ADV11

PERFORMANCE TEST

MD-11-DVADA-A

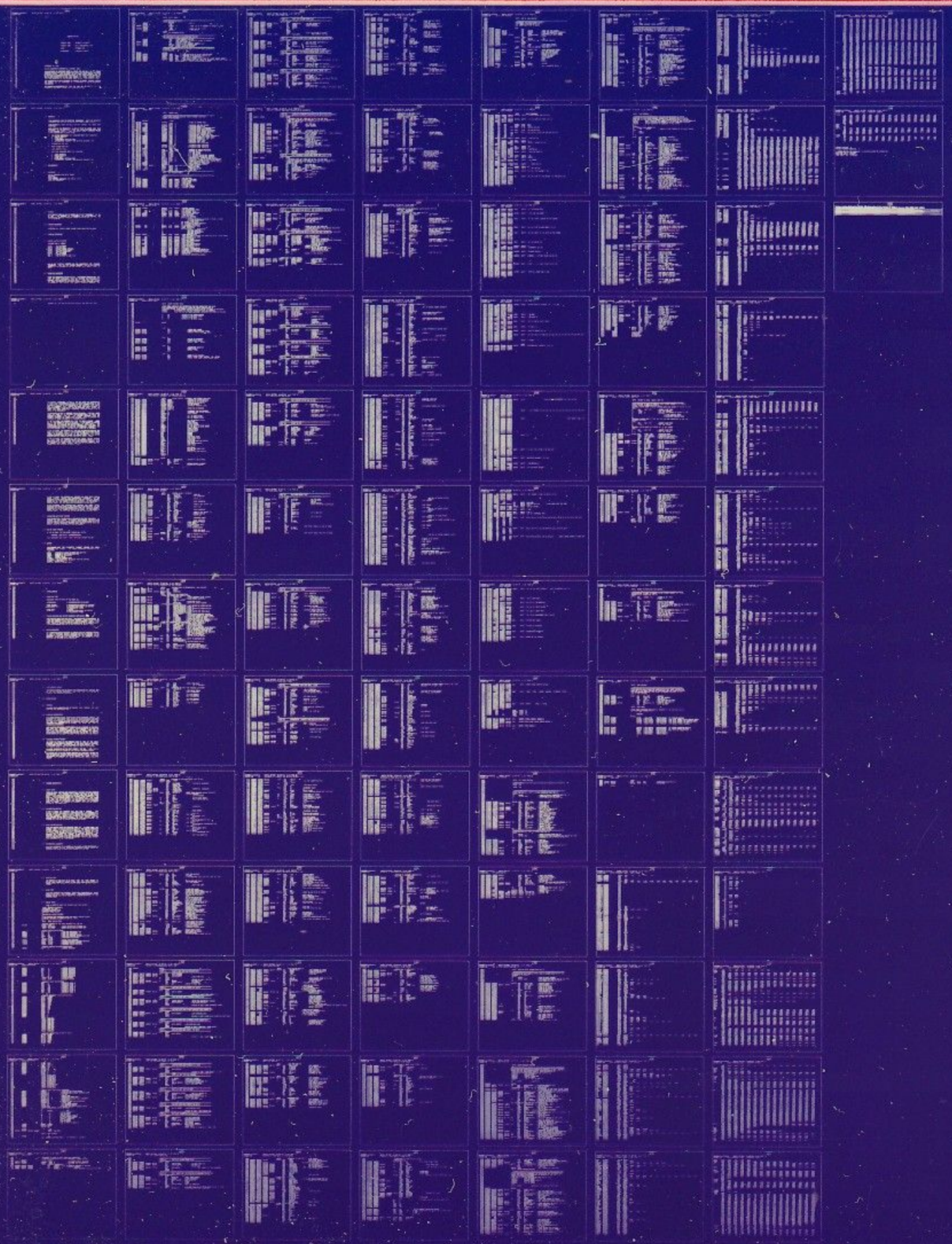
EP-DVADA-A-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.A



```
                        .REM  %
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DVADA-A
PRODUCT NAME:	ADV11 PERFORMANCE TEST
DATE:	OCTOBER 1976
MAINTAINER:	DIAGNOSTIC GROUP

1.0 ABSTRACT

THIS DIAGNOSTIC HAS TWO STARTING ADDRESSES: 200 FOR STANDARD TOLERANCES AND 210 FOR THE OPTION TEST AREA'S BURN IN TEST.

THIS DIAGNOSTIC TESTS THE ADV11 WITH OR WITHOUT THE BERG TEST CONNECTOR.

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 BERG TEST CONNECTOR)
- A. ANALOG SUBTESTS
- B. NOISE TEST
- C. INTERCHANNEL SETTLING TEST
- D. DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST

C: CALIBRATION TEST ONLY

P: PRINT VALUES TEST ONLY

L: LOGIC SUBTESTS ONLY

A: AUTO TEST (REQUIRES BERG TEST CONNECTOR)
- 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

LSI-11 COMPUTER WITH 8K OF MEMORY
TELETYPE
ADV11 MODULE
VT55 TERMINAL SUPPORTED FOR GRAPHIC OUTPUT
BERG TEST CONNECTOR

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 BURN IN TEST.

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 LETTER AND <CR> FOR TEST:". THEN TYPE THE LETTER YOU WANT, ACCORDING TO THE TABLE LISTED AND HIT CARRIAGE RETURN. IF STARTED AT THE OPTION TEST AREA'S STARTING ADDRESS, THE

```
  161                    PROGRAM WILL NOT ASK FOR THE TEST BUT WILL RUN THE LOGIC TEST.
```

```
162
163
164
165                     TWO CONTROL CHARACTERS, ↑A AND ↑C, ARE SET ASIDE FOR INTERRUPTING
166                     A  TEST  AND  TRANSFERRING CONTROL TO EITHER THE BEGINNING OF THE
167                     DIAGNOSTIC (↑C) OR TO THE BEGINNING OF THE  SPECIFIC  TEST  WHICH
168                     WAS  IN  PROGRESS  (↑A).  DURING THE LOGIC TESTS WHILE A RESET IS
169                     BEING PERFORMED, ↑C OR ↑A WILL NOT BE EXECUTED  UNTIL  AFTER  THE
170                     RESET  HAS  BEEN  COMPLETED,  THEREFORE  CONTINUE TYPING ↑C OR ↑A
171                     UNTIL IT IS SUCCESSFUL.
172
173                     FOR MACHINES WITHOUT A HARDWARE SWITCH REGISTER,  LOCATION  SWREG
174                     (176)  IS  USED  AS  A  SOFTWARE  SWITCH REGISTER.  TO MODIFY THE
175                     CONTENTS OF SWREG,  TYPE  ↑G.   THE  PROGRAM  RESPONDS  WITH  THE
176                     CURRENT  CONTENTS  OF  SWREG  AND  A SLASH.  TYPE THE DESIRED NEW
177                     CONTENTS OF SWREG FOLLOWED BY A CARRIAGE RETURN.
178
179                     IF "W" IS TYPED, THE PROGRAM WILL TYPE "XX ADV11'S FOUND".  WHERE
180                     XX  IS  THE NUMBER OF ADV11'S IN OCTAL.  IF THE NUMBER IS GREATER
181                     THAN 1, THE TEST WILL BE RUN SUCCESSIVELY  ON  EACH  ADV11.   THE
182                     PROGRAM WILL RUN THROUGH THE ANALOG SUBTESTS, THE NOISE TEST, THE
183                     INTERCHANNEL SETTLING TEST, AND THE  DIFFERENTIAL  LINEARITY  AND
184                     RELATIVE ACCURACY TEST.  THE BERG TEST CONNECTOR IS REQUIRED.
185
186                     IF "C" IS TYPED, THE PROGRAM WILL RUN THE CALIBRATION ROUTINE AND
187                     LOOP  ON  THE  TEST  UNTIL IT IS CALIBRATED AND A CARRIAGE RETURN
188                     TYPED.  IF A CERTAIN  ADV11  IS  TO  BE  CALIBRATED,  ITS  STATUS
189                     REGISTER ADDRESS MUST BE LOADED INTO $BASE (1250), AND ITS VECTOR
190                     ADDRESS MUST BE LOADED INTO THE LOW BYTE OF $VECT1 (1244).
191
192                     IF "P" IS TYPED, THE PROGRAM WILL RUN THE  PRINT  VALUES  ROUTINE
193                     AND  WILL  LOOP  ON  THAT TEST UNTIL THE OPERATOR HALTS IT.  IF A
194                     CERTAIN ADV11 IS TO BE TESTED, ITS STATUS REGISTER  ADDRESS  MUST
195                     BE  LOADED  INTO  $BASE  (1250),  AND  ITS VECTOR ADDRESS MUST BE
196                     LOADED INTO THE LOW BYTE OF $VECT1 (1244).
```

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 RUN SUCCESSIVELY ON EACH ADV11.

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 ADV11.

5.1 INHIBITING AUTO-SIZE FEATURE

THIS PROGRAM WILL AUTOMATICALLY AUTO-SIZE AND TEST EACH ADV11 IT DETECTS ON THE SYSTEM. TO INHIBIT THIS FEATURE, SET BIT 15 OF LOCATION $ENVM (1214). ALSO, LOAD LOCATION $BASE (1250) WITH THE ADV11'S STATUS REGISTER ADDRESS AND THE LOW BYTE OF LOCATION $VECT1 (1244) WITH THE ADV11'S VECTOR ADDRESS.

5.2 END OF PASS TYPEOUTS

AT END OF PASS, THE FOLLOWING TYPEOUT WILL OCCUR:

"ENDPASS GOOD UNITS 0000000000000011

THIS INDICATES THAT UNITS 1 AND 2 HAVE RUN WITHOUT FAILURE.

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:	5 CONVERSIONS/MIN @110 BAUD
PRINT VALUES:	8 CONVERSIONS/8 SECONDS @ 110 BAUD
WRAPAROUND TEST:	7 MINUTES FIRST PASS; 32 MINUTES FOR SUCCESSIVE PASSES
LOGIC TEST:	1 MINUTE
AUTO TEST:	8 MINUTES FIRST PASS, 33 MINUTES FOR SUCCESSIVE PASSES

7.2 STATUS REGISTER AND VECTOR ADDRESSES AND PRIORITY

WHEN TESTING MORE THAN ONE ADV11, THE DIFFERENCE IN ADDRESSES IS 4 FOR BUS ADDRESS AND 10 FOR VECTOR ADDRESS. THESE VALUES ARE IN VADR (BUS ADDRESS) (1332) AND VVCT (VECTOR ADDRESS) (1334). THE FIRST ADV11'S STATUS REGISTER ADDRESS MUST BE IN $BASE (1250), ITS VECTOR ADDRESS MUST BE IN THE LOW BYTE OF $VECT1 (1244).

7.3 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 SWITCH REGISTER.

7.4 VT55 GRAPHIC OUTPUT

THE SCREEN DISPLAY MAY BE HALTED FOR EXAMINATION BY SETTING BIT 12 OF THE SWITCH REGISTER. AND THEN JUST HIT CONTINUE TO COMPLETE THE PROGRAM'S EXECUTION.

8.0 RESTRICTIONS

8.1 TESTING

THE BERG TEST CONNECTOR MUST BE PRESENT WHEN RUNNING THE AUTO TEST AND THE WRAPAROUND TEST.

8.2 STARTING RESTRICTION

IF A FREE-RUNNING CLOCK, SUCH AS 60HZ FROM THE POWER SUPPLY, IS ATTACHED TO THE BEVNT BUS LINE ON BOTH REV LEVEL C/D AND E SYSTEMS, AN INTERRUPT TO LOCATION 100 WILL OCCUR WHEN USING THE "G" AND "L" COMMANDS PRIOR TO EXECUTING THE FIRST INSTRUCTION. THEREFORE THIS PROGRAM CAN NOT DISABLE THE BEVNT BUS LINE BY INHIBITING INTERRUPTS.

USER SYSTEMS REQUIRING A FREE-RUNNING CLOCK ATTACHED TO THE BEVNT BUS LINE CAN TEMPORARILY AVOID THIS SITUATION BY SETTING THE PSW(R5) TO 200, INSTEAD OF USING THE "G" COMMAND, LOAD THE PC (R7) WITH THE STARTING ADDRESS AND USE THE PROCEED "P" COMMAND. BEFORE USING THE "L" COMMAND, THE PSW(R5) CAN BE SET TO 200 TO AVOID RECEIVING THE BEVNT INTERRUPT AFTER LOADING THE ABS LOADER.

8.3 POSSIBLE PROGRAM "BOMBS"

THE FIRST TWO TESTS OF THIS PROGRAM CHECK TO SEE IF THE ADV11 RESPONDS TO THE EXPECTED ADDRESS. IF THE ADV11 DOES NOT RESPOND, A BUSS ERROR OCCURS. ALSO BUS ERRORS CAN OCCUR DURING THE TIME THE PROGRAM SIZES TO SEE HOW MANY ADV11'S ARE ON YOUR SYSTEM.

FOR MORE INFORMATION ON THE NEXT SUBJECT, SEE JAN. 1976 LSI-11 ENGINEERING BULLETIN ISSUED BY THE DIGITAL COMPONENTS GROUP.

BUS ERRORS MAY ALTER THE PRESET CONTENTS OF LOCATION 4 BEFORE THE TRAP IS EXECUTED, THEREBY TRANSFERRING PROGRAM CONTROL TO AREA IN THE PROGRAM THAT WAS NOT SET UP TO HANDLE THE TRAP. IF THIS HAPPENS, THE PROGRAM WILL "BOMB" AND POSSIBLY REWRITE PARTS OF ITSELF.

9.0 PROGRAM DESCRIPTION

9.1 LOGIC TESTS

THESE 23 LOGIC SUBTESTS RUN SEQUENTIALLY WITHOUT FURTHER OPERATOR INTERVENTION. ITS PURPOSE IS TO CHECK THAT EACH OF THE STATUS REGISTER BITS THAT ARE READ/WRITE 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 ROUTINE

IF "C" IS TYPED, THE PROGRAM WILL ASK FOR A CHANNEL. TYPE CHANNEL NUMBER FOLLOWED BY A CARRIAGE RETURN. THE PROGRAM WILL ASK YOU IF YOU WANT OFFSET OR GAIN. APPLY VOLTAGE REQUESTED TO SELECTED CHANNEL. ADJUST POT REQUESTED FOR 0.00 LSB TYPEOUT. TYPE CARRIAGE RETURN WHEN ADJUSTED. THE LAST TYPEOUT WILL BE CHECKED FOR 0.00 LSB WITH A TOLERANCE OF 0.04 LSB IF OUTSIDE, THE PROGRAM WILL ASK YOU TO ADJUST THE SAME POT AGAIN.

9.3 PRINT VALUES ROUTINE

THIS TEST BEGINS WHEN THE OPERATOR TYPES "P". 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 (0), IT PRINTS OUT THE CONVERTED VALUE ON THE TELETYPE; OTHERWISE, IF SWR BIT 13 IS UP (1), 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.

9.4 DIFFERENTIAL LINEARITY

THIS TEST DETERMINE IF A CHANGE IN THE INPUT VOLTAGE REPRESENTS A SIMILAR CHANGE IN THE RESULTING CONVERTED BINARY VALUE, BY MEASURING THE WIDTH OF EACH STATE CORRECT TO 0.01 LSB.

```
                                9.5  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.


                                9.6  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.


                                9.7  ANALOG TESTS

                                     THESE 6 SUBTESTS CHECK THE CHANNELS AND THEIR OUTPUT.
                                     %
                                .TITLE  MAINDEC-11-DVADA-A
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY GEORGE STEVENS
                                ;*
                                ;*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
```

```
                                ;
                                R1=     %1              ;;GENERAL REGISTER
                                R2=     %2              ;;GENERAL REGISTER
                                R3=     %3              ;;GENERAL REGISTER
                                R4=     %4              ;;GENERAL REGISTER
                                R5=     %5              ;;GENERAL REGISTER
                                R6=     %6              ;;GENERAL REGISTER
                                R7=     %7              ;;GENERAL REGISTER
                                .EQUIV  R6,SP           ;;STACK POINTER
                                .EQUIV  R7,PC           ;;PROGRAM COUNTER
                                ;
                                ;*PRIORITY LEVEL DEFINITIONS
                                ;
                                PR0=    0               ;;PRIORITY LEVEL 0
                                PR1=    40              ;;PRIORITY LEVEL 1
                                PR2=    100             ;;PRIORITY LEVEL 2
                                PR3=    140             ;;PRIORITY LEVEL 3
                                PR4=    200             ;;PRIORITY LEVEL 4
                                PR5=    240             ;;PRIORITY LEVEL 5
                                PR6=    300             ;;PRIORITY LEVEL 6
                                PR7=    340             ;;PRIORITY LEVEL 7
                                ;
                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                                ;
                                SW15=   100000
                                SW14=   40000
                                SW13=   20000
                                SW12=   10000
                                SW11=   4000
                                SW10=   2000
                                SW09=   1000
                                SW08=   400
                                SW07=   200
                                SW06=   100
                                SW05=   40
                                SW04=   20
                                SW03=   10
                                SW02=   4
                                SW01=   2
                                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)
                                ;
                                BIT15=  100000
                                BIT14=  40000
                                BIT13=  20000
                                BIT12=  10000
                                BIT11=  4000
                                BIT10=  2000
                                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
                        100400          AVECTI= 100400
                        000200          APRIOR= 200

                        000100                  .=100
000100  000104  000200  000002                  .WORD   104,200,2

                                        .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 001644      JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137 002262      JMP     @#BEG2          ;RESTART ADDRESS
000210  000137 001652      JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;**********************************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  011744                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000214                                  .=$SVPC                 ;; RESTORE PC
001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

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

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  000000  SCMTAG:                 ;;START OF COMMON TAGS
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
```

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

```
                                .SBTTL  ERROR POINTER TABLE

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

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

001256                          $ERRTB:


                                ;ITEM   1
001256  014261                          EM1                     ;STATUS REG, ERROR
001260  014401                          DH1                     ;ERRPC STREG EXPECTED ACTUAL
001262  014560                          DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
001264  014620                          DF1


                                ;ITEM   2
001266  014303                          EM2                     ;FAILED TO INTERRUPT
001270  014520                          DH3                     ;ERRPC STREG ACTUAL
001272  014610                          DT3                     ;$ERRPC, STREG,  $BDDAT
001274  014620                          DF1

                                ;ITEM   3
001276  014327                          EM3                     ;UNEXPECTED INTERRUPT
001300  014520                          DH3                     ;ERRPC STREG
001302  014610                          DT3                     ;$ERRPC, STREG
001304  014620                          DF1

                                ;ITEM   4
001306  014354                          EM4                     ;ERROR ON A/D CHANNEL
001310  014435                          DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
001312  014572                          DT2                     ;$ERRPC,STREG,CHANL,$GDDAT,$PREAD,$BDDAT
001314  014620                          DF1
```

```
                        .SBTTL  MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
001316  170400          STREG:  ABASE                   ;ADDRESS OF STATUS REGISTER
001320  170401          ADSTI:  ABASE+1                 ;UPPER BYTE OF STATUS REG.
001322  170402          ADBUFF: ABASE+2                 ;ADDRESS OF A/D BUFFER
001324  100400          VECTOR: AVECT1                  ;VECTOR ADDRESS
001326  000200          BASEBR: APRIOR                  ;INTERRUPT PRIORITY LEVEL
001330  100402          VECTR1: AVECT1+2
001332  100404          VECTR2: AVECT1+4                ;ERROR VECTOR ADDRESS
001334  100406          VECTR3: AVECT1+6
001336  000004          VADR:   4                       ;INCREMENT FOR BUS ADDRESS
001340  000010          VVCT:   10                      ;INCREMENT FOR VECTOR ADDRESS
001342  000000          BASECH: 0                       ;BASE CHANNEL
001344  000060          KBVECT: 60
001346  000000          WIDE:   0                       ;NO. OF WIDE STATES
001350  000000          NARROW: 0                       ;NO. OF NARROW STATES
001352  000000          FIRST:  0
001354  000000          SKIPST: 0                       ;NO. OF SKIPPED STATES
001356  000000          TEMP:   0                       ;WORK AREA
001360  000000          CH1:    0                       ;FIRST CHANNEL
001362  000000          CH2:    0                       ;SECOND CHANNEL
001364  000000          NBEXT:  0                       ;NO. OF ADV11'S TO BE TESTED
001366  000000          NMBEXT: 0                       ;NO. OF ADV11'S TO BE TESTED
001370  000000          DUMMY:  0                       ;DUMMY CHANNEL
001372  000000          CHANL:  0                       ;CHANNEL VALUE
001374  000000          TADDR:  0                       ;TEST ADDRESS
001376  000000          RNA:    0                       ;RANDOM
001400  000000          RNB:    0                       ;NUMBER
001402  000000          RNC:    0                       ;VALUES
001404  000000          RMS:    0                       ;RMS NOISE VALUE
001406  000000          PEAK:   0                       ;PEAK NOISE VALUE
001410  000000          FLAG:   0                       ;VT55 FLAG
001412  000000          SPREAD: 0                       ;DEVIATION FROM THE NOMINAL
001414  000000          DAC:    0                       ;SAR VALUE
001416  000000          DELAY:  0                       ;TIME DELAY COUNTER
001420  000000          EDGE:   0                       ;EDGE VALUE
001422  000000          BITPNT: 0
001424  000000          MIN:    0                       ;MIN VALUE
001426  000000          WFTEST: 0                       ;OPTION TEST AREA FLAG
001430  000000          MAX:    0                       ;MAX VALUE
001432  000000          PERCNT: 0                       ;PERCENT FOR SAR ROUTINE
001434  000000          OUT:    0
001436  000000          GUNITS: 0
001440  000001          TSTBIT: 1

001442                  UNEXP:
001442  012737  001456  001162        MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
001450  005237  001103                INC     $ERFLG
001454  104003                      ERROR   3
001456  005037  001162  1$:         CLR     $ESCAPE         ;RETURN ESCAPE TO NORMAL
001462  000002                      RTI                     ;UNEXPECTED INTERRUPT
```

```
                                        .SBTTL  CONTROL A AND C DECODERS
001464 010046                   ISERV:  MOV     RO,-(SP)        ;SAVE RO
001466 017700 177454                    MOV     @#TKB,RO        ;GET CHARACTER
001472 042700 177600                    BIC     #177600,RO
001476 120027 000003                    CMPB    RO,#3           ;IS IT ↑C?
001502 001010                           BNE     1$
001504 104400 012036                    TYPE    ,CMSG           ;ECHO CHARACTER
001510 012706 001100                    MOV     #STACK,SP
001514 004737 011300                    JSR     PC,RST          ;RESET & SET INTRPT. EN.
001520 000137 002262                    JMP     BEG2
001524 120027 000001            1$:     CMPB    RO,#1           ;IS IT ↑A?
001530 001010                           BNE     2$
001532 104400 012031                    TYPE    ,AMSG           ;ECHO CHARACTER
001536 012706 001100                    MOV     #STACK,SP
001542 004737 011300                    JSR     PC,RST          ;RESET & SET INTRPT. EN.
001546 000177 177622                    JMP     @TADDR          ;RETURN TO TEST
001552 120027 000007            2$:     CMPB    RO,#7           ;IS IT ↑G?
001556 001027                           BNE     NONE
001560 023727 001140 177570             CMP     SWR,#177570     ;HARDWARE SWREG?
001566 001423                           BEQ     NONE
001570 104400 012043                    TYPE    ,GMSG           ;ECHO CHARACTER
001574 017746 177340                    MOV     @SWR,-(SP)      ;;SAVE @SWR FOR TYPEOUT
                                                                ;;TYPE SWREG
001600 104402                           TYPOS                   ;;GO TYPE--OCTAL ASCII
001602    006                           .BYTE   6               ;;TYPE 6 DIGITS
001603    001                           .BYTE   1               ;;TYPE LEADING ZEROS
001604 104400 012223                    TYPE    ,SLASH
001610 104407                           RDOCT                   ;READ NEW VALUE
001612 012677 177322                    MOV     (SP)+,@SWR      ;LOAD NEW SWREG VALUE
001616 012600                   POPRO:  MOV     (SP)+,RO
001620 022776 000001 000000     RETURN: CMP     #1,@0(SP)       ;DOES IT RETURN TO A WAIT?
001626 001002                           BNE     1$              ;NO
001630 062716 000002                    ADD     #2,(SP)         ;BUMP RETURN ADDRESS
001634 000002                   1$:     RTI
001636 104400 012027            NONE:   TYPE    ,QUEST          ;TYPE "?"
001642 000765                           BR      POPRO
```

```
                                        .SBTTL          INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
001644  005037  001426          BEGIN:  CLR     @#WFTEST
001650  000403                  BEGIN2: BR      RBEG
001652  012737  000001  001426          MOV     #1,@#WFTEST
001660  000005                  RBEG:   RESET
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001662  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001666  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001670  022706  001140                  CMP     #SWR,R6 ;;DONE?
001674  001374                          BNE     .-6             ;;LOOP BACK IF NO
001676  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001702  012737  015216  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001710  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001716  012737  015474  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001724  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001732  012737  017064  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001740  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001746  013737  011664  011656          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001754  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001760  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001764  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001772  012737  001772  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002000  012737  002000  001110          MOV     #.,$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.
002006  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002012  012737  002046  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002020  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002026  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002034  022777  177777  177076          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002042  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002044  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002046  012716  002054          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002052  000002                          RTI
002054  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002062  012737  000174  001142          MOV     #DISPREG,DISPLAY
002070  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002074  005037  001202                  CLR     $PASS           ;;CLEAR PASS COUNT
002100  132737  000200  001215          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002106  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002110  012737  001216  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002116                  67$:
```

```
  873  002116  005037  001410                    CLR     FLAG            ;CLEAR VT55 FLAG
  874  002122  005737  000042                    TST     @#42                    ;IS IT CHAINED?
  875  002126  001033                            BNE     REST1
  876                                .SBTTL      DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
  877  002130  042777  000100  177006            BIC     #100,@$TKS
  878  002136  104400  013724                    TYPE    CO              ;TYPE ASCIZ STRING
  879  002142  004737  002432                    JSR     PC,VTFLG        ;GET A CHARACTER
  880  002146  020027  000033                    CMP     R0,#33
  881  002152  001017                            BNE     NOVT55          ;NO VT55 PRESENT
  882  002154  004737  002432                    JSR     PC,VTFLG        ;GET A CHARACTER
  883  002160  020027  000057                    CMP     R0,#57
  884  002164  001012                            BNE     NOVT55          ;NO VT55 PRESENT
  885  002166  004737  002432                    JSR     PC,VTFLG        ;GET A CHARACTER
  886  002172  020027  000103                    CMP     R0,#103
  887  002176  001403                            BEQ     VT55            ;VT55 IS PRESENT
  888  002200  020027  000105                    CMP     R0,#105
  889  002204  001002                            BNE     NOVT55
  890  002206  005237  001410            VT55:   INC     FLAG
```

```
                                        .SBTTL  DIALOGUE TO DETERMINE WHICH TEST TO RUN
002212  104400  014067          NOVT55: TYPE    ,READI
002216  000005                  RESTI:  RESET
002220  004737  006310                  JSR     PC,FIXONE               ;INITIALIZE ADDRESSES
002224  013700  001344                  MOV     KBVECT,R0
002230  012720  001464                  MOV     #ISERV,(R0)+
002234  012710  000340                  MOV     #340,(R0)
002240  012737  062341  001376          MOV     #62341,RNA              ;RANDOM NO, VARIABLES
002246  012737  142315  001400          MOV     #142315,RNB
002254  012737  127623  001402          MOV     #127623,RNC
002262  012706  001100          BEG2:   MOV     #STACK,SP       ;RESET STACK POINTER INCASE RESTARTED
002266  000005                          RESET                   ;RESTART ADDRESS
002270  005737  000042                  TST     @#42                    ;IS IT CHAINED?
002274  001405                          BEQ     1$
002276  000137  006030          2$:     JMP     BEGL                    ;GO TO LOGIC TESTS
002302  005737  001426                  TST     WFTEST                  ;TEST FOR OPTION TEST
002306  001373                          BNE     2$              ;;
002310  104400  013541          1$:     TYPE    ,MSG71
002314  104406                  TRYAG:  RDLIN
002316  052777  000100  176620          BIS     #100,@STKS
002324  005046                          CLR     -(SP)                   ;CLEAR PSW
002326  012746  002334                  MOV     #1$,-(SP)
002332  000002                          RTI
002334  012600                  1$:     MOV     (SP)+,R0                ;READ ANSWER
002336  142710  000040                  BICB    #40,(R0)
002342  121027  000101                  CMPB    (R0),#'A                ;IS IT A?
002346  001002                          BNE     2$
                                ;;NO, TRY C
002350  000137  006066                  JMP     BEGINA                  ;GO TO AUTO TEST
002354  121027  000103          2$:     CMPB    (R0),#'C                ;IS IT C?
002360  001002                          BNE     3$
                                ;;NO, TRY P
002362  000137  005364                  JMP     BEGINC                  ;GO TO CALIBRATION TEST
002366  121027  000120          3$:     CMPB    (R0),#'P                ;IS IT P?
002372  001002                          BNE     4$
                                ;;NO, TRY L
002374  000137  005640                  JMP     BEGINP                  ;GO TO DISPLAY CONVERSIONS TEST
002400  121027  000114          4$:     CMPB    (R0),#'L                ;IS IT L?
002404  001002                          BNE     5$
                                ;;NO, TRY W
002406  000137  006030                  JMP     BEGL                    ;GO TO LOGIC TESTS
002412  121027  000127          5$:     CMPB    (R0),#'W                ;IS IT W?
002416  001002                          BNE     6$
                                ;;NO,TRY AGAIN
002420  000137  006150                  JMP     BEGINW                  ;GO TO WRAPAROUND TEST
002424  104400  012027          6$:     TYPE    ,QUEST
002430  000731                          BR      TRYAG                   ;WAIT FOR CHARACTER
```

```
                                VTFLG:  CLR     R0              ;TEST FOR PRESENCE
                                1$:     TSTB    @$TKS           ;OF VT55
                                        BMI     2$              ;;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
                                        DEC     R0
                                        BNE     1$              ;;
                                        TST     (SP)+           ;POP A WORD OFF STACK
                                        BR      NOVT55          ;;NO VT55 PRESENT
                                2$:     MOV     @$TKB,R0
                                        BIC     #177600,R0              ;TEST VT55 CODE
                                        RTS     PC

                                TESTAD: CLR     $PASS           ;CLEAR PASS COUNT
                                        CLR     GUNITS          ;CLEAR UNIT ERROR BITS
                                        MOV     #1,TSTBIT       ;INITIALIZE MODULE ERROR TEST BIT
                                        MOV     #1,TEMP         ;SET UP FOR ONLY ONE A/D
                                        TSTB    $ENVM           ;TESTING ONLY ONE A/D?
                                        BMI     3$              ;;YES
                                        MOV     #4,TEMP         ;SET UP MAX NO OF A/D'S
                                        TST     WFTEST          ;IS IT IN OPTION TEST
                                        BEQ     3$              ;;NOT IN OPTION TEST
                                        MOV     #16.,TEMP       ;SET UP OPTION MAX NO OF A/D'S
                                3$:     MOV     $BASE,$BDDAT    ;SETUP TO TEST FOR ADV11'S
                                        MOV     @#ERRVEC,-(SP)  ;SAVE ERRVEC
                                        MOV     #2$,ERRVEC      ;SET UP FOR TIME OUT ERROR
                                        CLR     NBEXT           ;CLEAR ADV11 COUNTER
                                1$:     TST     @$BDDAT         ;ADDRESS ADV11
                                        INC     NBEXT           ;INCREMENT ADV11 COUNTER
                                        BIS     TSTBIT,GUNITS   ;SET A/D BIT UNDER TEST
                                        ASL     TSTBIT          ;SET TEST BIT FOR NEXT UNIT
                                        DEC     TEMP            ;REACHED MAX?
                                        BEQ     4$              ;;REACHED MAX NO OF A/D'S
                                        ADD     VADR,$BDDAT     ;GET NEXT ADV11
                                        BR      1$              ;;TRY NEXT ADV11
                                2$:     CMP     (SP)+,(SP)+     ;POP 2 WORDS OFF STACK
                                4$:
                                        MOV     NBEXT,-(SP)             ;;SAVE NBEXT FOR TYPEOUT
                                                                ;;TYPE NUMBER OF ADV11'S
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   2               ;;TYPE 2 DIGIT(S)
                                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        TYPE    ,MSG50
                                        DEC     NBEXT           ;ADJUST ADV11 COUNT
                                        MOV     NBEXT,NMBEXT    ;KEEP COUNT OF NUMBER
                                        MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
                                        MOV     #1,TSTBIT       ;INITIALIZE MODULE ERROR TEST BIT
                                        RTS     PC
```

```
                                        BEGINL:
                                        ;;*****************************************************************
                                        ;*TEST 1         FLOAT A ONE THRU MULTIPLEXER BITS
                                        ;;*****************************************************************
002670  012737  002670  001106  TST1:   MOV     #TST1,$LPADR
002676  012737  002670  001110          MOV     #TST1,$LPERR
002704  012737  000400  001124          MOV     #BIT8,$GDDAT            ;LOAD FIRST BIT
002712  104411                  2$:     CHKIT
002714  104001                          ERROR   1                       ;FAILED TO LOAD + READ BIT
002716  006337  001124          1$:     ASL     $GDDAT                  ;GET NEXT BIT
002722  023727  001124  010000          CMP     $GDDAT,#BIT12           ;FINISHED?
002730  001370                          BNE     2$              ;;NO,GO TO NEXT TEST

                                        ;;*****************************************************************
                                        ;*TEST 2         LOAD AND READ BACK ERROR I.E. BIT14
                                        ;;*****************************************************************
002732  000004                  TST2:   SCOPE
002734  012737  040000  001124          MOV     #BIT14,$GDDAT
002742  104411                          CHKIT
002744  104001                          ERROR   1                       ;FAILED TO LOAD + READ ERROR I.E.
                                        ;;*****************************************************************
                                        ;*TEST 3         LOAD AND READ BACK INTERRUPT ENABLE BIT6
                                        ;;*****************************************************************
002746  000004                  TST3:   SCOPE
002750  012777  001442  176346          MOV     #UNEXP,@VECTOR          ;SETUP FOR UNEXPECTED INTERUPT
002756  012737  000100  001124          MOV     #BIT6,$GDDAT            ;LOAD EXPECTED DATA
002764  104411                          CHKIT
002766  104001                          ERROR   1                       ;FAILED TO LOAD + READ INTERRUPT ENABLE

                                        ;;*****************************************************************
                                        ;*TEST 4         LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
                                        ;;*****************************************************************
002770  000004                  TST4:   SCOPE
002772  012737  000040  001124          MOV     #BIT5,$GDDAT            ;LOAD EXPECTED DATA
003000  104411                          CHKIT
003002  104001                          ERROR   1                       ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB

                                        ;;*****************************************************************
                                        ;*TEST 5         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
                                        ;;*****************************************************************
003004  000004                  TST5:   SCOPE
003006  012737  000020  001124          MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
003014  104411                          CHKIT
003016  104001                          ERROR   1               ;FAILED TO LOAD + READ EXT. START ENABLE
                                        ;;*****************************************************************
                                        ;*TEST 6         LOAD AND READ BACK MAINT. TST BIT2
                                        ;;*****************************************************************
003020  000004                  TST6:   SCOPE
003022  012737  000004  001124          MOV     #BIT2,$GDDAT
003030  104411                          CHKIT
003032  104001                          ERROR   1                       ;FAILED TO LOAD + READ BACK MAINT. TST
```

```
                                ;;*****************************************************************
                                ;*TEST 7         LOAD AND READ BACK ENABLE I.D. BIT3
                                ;;*****************************************************************
003034  000004                  TST7:   SCOPE
003036  012737  000010  001124          MOV     #BIT3,$GDDAT
003044  104411                        CHKIT
003046  104001                        ERROR   1                       ;FAILED TO LOAD + READ ENABLE I.D. BIT
                                ;;*****************************************************************
                                ;*TEST 10        TEST I.D. BIT (BIT 12) CLEARED
                                ;;*****************************************************************
003050  000004                  TST10:  SCOPE
003052  012777  000001  176236          MOV     #1,@STREG               ;CLEAR I.D. ENABLE
003060  105777  176232          1$:     TSTB    @STREG                  ;WAIT FOR CONVERSION
003064  100375                        BPL     1$                      ;;CONVERSION IS NOT DONE YET
003066  032777  010000  176226          BIT     #BIT12,@ADBUFF          ;IS I.D. BIT CLEARED?
003074  001401                        BEQ     TST11                   ;;YES - GOTO NEXT TEST
003076  104001                        ERROR   1

                                ;;*****************************************************************
                                ;*TEST 11        TEST I.D. BIT (BIT 12) SET
                                ;;*****************************************************************
003100  000004                  TST11:  SCOPE
003102  012777  000011  176206          MOV     #BIT3!BIT0,@STREG       ;SET I.D. ENABLE BIT
003110  105777  176202          1$:     TSTB    @STREG                  ;WAIT FOR CONVERSION
003114  100375                        BPL     1$                      ;;CONVERSION IS NOT DONE YET
003116  032777  010000  176176          BIT     #BIT12,@ADBUFF          ;IS I.D. BIT SET?
003124  001001                        BNE     TST12                   ;;YES - GOTO NEXT TEST
003126  104001                        ERROR   1

                                ;;*****************************************************************
                                ;*TEST 12        LOAD AND READ BACK ERROR FLAG BIT15
                                ;;*****************************************************************
003130  000004                  TST12:  SCOPE
003132  012737  100000  001124          MOV     #BIT15,$GDDAT           ;LOAD EXPECTED DATA
003140  104411                        CHKIT
003142  104001                        ERROR   1                       ;FAILED TO LOAD + READ ERROR FLAG
                                ;;*****************************************************************
                                ;*TEST 13        TEST INIT CLEARS BITS 2-6,8-11,14
                                ;;*****************************************************************
003144  000004                  TST13:  SCOPE
003146  012737  000300  001160          MOV     #300,$TIMES             ;;DO 300 ITERATIONS
003154  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED DATA
003160  012777  047574  176130  2$:     MOV     #47574,@STREG           ;SET STATUS REGISTER
003166  000005                        RESET                           ;INITIALIZE
003170  052777  000100  175746          BIS     #100,@STKS              ;SET INTRPT. ENABLE
003176  017737  176114  001126          MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
003204  001401                        BEQ     TST14                   ;;NEXT TEST
003206  104001                        ERROR   1                       ;RESET FAILED TO CLEAR AD ST. REG. BITS
```

```
 1079                                   ;;*****************************************************************
 1080                                   ;*TEST 14       TEST INIT CLEARS ERROR FLAG
 1081                                   ;;*****************************************************************
 1082  003210  000004                   TST14:  SCOPE
 1083  003212  012737  000300  001160           MOV     #300,$TIMES             ;DO 300 ITERATIONS
 1084  003220  012777  100000  176070           MOV     #BIT15,@STREG           ;SET BIT 15
 1085  003226  000005                           RESET                           ;ISSUE INIT
 1086  003230  052777  000100  175706           BIS     #100,@STKS                      ;SET INTRPT. EN. FOR KEYBOARD
 1087  003236  104410                           CHECK
 1088  003240  104001                           ERROR   1
 1089                                   ;;*****************************************************************
 1090                                   ;*TEST 15       TEST  DONE FLAG  SETS AND BITO CLEARS ON END OF CONV.
 1091                                   ;;*****************************************************************
 1092  003242  000004                   TST15:  SCOPE
 1093  003244  012700  001000                   MOV     #BIT9,R0        ;STALL TIME COUNTER
 1094  003250  005277  176042                   INC     @STREG          ;START CONVERSION
 1095  003254  012737  000200  001124           MOV     #BIT7,$GDDAT    ;LOAD EXPECTED
 1096  003262  005300                   1$:     DEC     R0              ;STALL
 1097  003264  001376                           BNE     1$              ;TIME
 1098  003266  042777  100000  176022           BIC     #BIT15,@STREG   ;MASK OUT ERROR BIT
 1099  003274  104410                           CHECK
 1100  003276  104001                           ERROR   1               ;A/D DONE FLAG FAILED TO SET;BITO FAILED TO CLEAR
 1101  003300  017700  176016                   MOV     @ADBUFF,R0      ;CLEAR DONE FLAG FOR ITERATIONS
```

```
                                ;;*****************************************************************
                                ;*TEST 16        TEST INIT CLEARS DONE FLAG
                                ;;*****************************************************************
003304  000004                  TST16:  SCOPE
003306  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
003314  005037  001124          .       CLR     $GDDAT          ;CLEAR EXPECTED
003320  005277  175772          .       INC     @STREG          ;START CONVERSION
003324  105777  175766          2$:     TSTB    @STREG
003330  100375                  .       BPL     2$
003332  000005                  .       RESET
003334  104410                  .       CHECK
003336  104001                  .       ERROR   1               ;DONE FLAG FAILED TO CLEAR
003340  052777  000100  175576  .       BIS     #100,@STKS              ;SET INTRPT. EN. BIT

                                ;;*****************************************************************
                                ;*TEST 17        TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
                                ;;*****************************************************************
003346  000004                  TST17:  SCOPE
003350  005277  175742          .       INC     @STREG          ;SET A/D START CONVERSION BIT
003354  105777  175736          1$:     TSTB    @STREG          ;WAIT FOR FLAG
003360  100375                  .       BPL     1$
003362  017700  175734          .       MOV     @ADBUFF,R0      ;READ CONVERTED VALLUE
003366  104410                  .       CHECK
003370  104001                  .       ERROR   1               ;DONE FLAG FAILED TO CLEAR

                                ;;*****************************************************************
                                ;*TEST 20        TEST ALL '0'S RESULTS USING MAINT. ADTST. BIT
                                ;;*****************************************************************
003372  000004                  TST20:  SCOPE
003374  005037  001124          .       CLR     $GDDAT          ;CLEAR EXPECTED VALUE
003400  005037  001372          .       CLR     CHANL           ;SET CHANL = 0
003404  005037  001412          .       CLR     SPREAD          ;SET SPREAD = 0
003410  012777  000005  175700  .       MOV     #5,@STREG       ;CONVERT EVEN CHANNEL WITH MAINT. BIT SET
003416  105777  175674          1$:     TSTB    @STREG          ;WAIT FOR DONE
003422  100375                  .       BPL     1$
003424  017737  175672  001126  .       MOV     @ADBUFF,$BDDAT  ;RESULTS TO BDDAT FOR CHECKING
003432  001401                  .       BEQ     TST21           ;;GOTO NEXT TEST
003434  104004                  .       ERROR   4               ;DID NOT GET ALL '0'S RSULT WITH MAINT. ADTST

                                ;;*****************************************************************
                                ;*TEST 21        TEST ALL '1'S RESULT USING MAINT. ADTST. BIT
                                ;;*****************************************************************
003436  000004                  TST21:  SCOPE
003440  012737  007777  001124  .       MOV     #7777,$GDDAT    ;EXPECT ALL '1'S RESULT
003446  012737  000001  001372  .       MOV     #1,CHANL        ;SET CHANL = 1
003454  005037  001412          .       CLR     SPREAD          ;SET SPREAD = 0
003460  012777  000405  175630  .       MOV     #405,@STREG     ;CONVERT ODD CHANNEL WITH MAINT. BIT SET
003466  105777  175624          1$:     TSTB    @STREG          ;WAIT FOR DONE
003472  100375                  .       BPL     1$
003474  017737  175622  001126  .       MOV     @ADBUFF,$BDDAT  ;RESULTS TO BDDAT FOR CHECKING
003502  023737  001124  001126  .       CMP     $GDDAT,$BDDAT   ;EQUAL?
003510  001401                  .       BEQ     TST22           ;;GOTO NEXT TEST
003512  104004                  .       ERROR   4               ;DID NOT GET ALL '1'S RESULT WITH MAINT. ADTST
```

```
                                ;;*****************************************************************
                                ;*TEST 22       GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
                                ;;*****************************************************************
003514  000004                  TST22:  SCOPE
                                        ;* "ENTERING TEST 22" TYPED OUT TO TELL YOU THE NEXT
                                        ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
                                        ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
                                        ;*EXECUTING TEST "22".
003516  012700  000022                  MOV     #22,R0          ;GET TEST NO.
003522  004737  011176                  JSR     PC,$DMW         ;PRINT MESSAGE
003526  005046                          CLR     -(SP)           ;RESET PRIORITY
003530  012746  003536                  MOV     #3$,-(SP)
003534  000002                          RTI
003536  012777  003612  175560  3$:     MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
003544  012777  000200  175556          MOV     #200,@VECTR1    ;SET UP NEW PSW
003552  012777  000101  175536          MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
003560  105777  175532          2$:     TSTB    @STREG          ;WAIT FOR DONE
003564  100375                          BPL     2$              ;FLAG TO SET
003566  017737  175524  001126          MOV     @STREG,$BDDAT   ;READ STATUS REGISTER
003574  012737  000300  001124          MOV     #BIT7!BIT6,$GDDAT ;GOOD DATA
003602  104002                          ERROR   2               ;FAILED TO INTERRUPT ON DONE
003604  004737  011250                  JSR     PC,DUMC         ;TYPE COMPLETED
003610  000414                          BR      TST23           ;;BRANCH TO NEXT TEST
003612  022626                  1$:     CMP     (SP)+,(SP)+     ;RESET STACK POINTER
003614  012777  001442  175502          MOV     #UNEXP,@VECTOR  ;SET UP FOR UNEXPECTED INTERRUPT
003622  005046                          CLR     -(SP)           ;CLEAR PSW
003624  012746  003632                  MOV     #4$,-(SP)
003630  000002                          RTI
003632  004737  011250          4$:     JSR     PC,DUMC         ;TYPE COMPLETED
003636  005777  175460                  TST     @ADBUFF         ;CLEAR DONE BIT
                                ;;*****************************************************************
                                ;*TEST 23       TEST INTERRUPT OCCURS WHEN ERROR AND I.E.E. IS SET
                                ;;*****************************************************************
003642  000004                  TST23:  SCOPE
                                        ;* "ENTERING TEST 23" TYPED OUT TO TELL YOU THE NEXT
                                        ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
                                        ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
                                        ;*EXECUTING TEST "23".
003644  012700  000023                  MOV     #23,R0          ;GET TEST NO.
003650  004737  011176                  JSR     PC,$DMW         ;PRINT MESSAGE
003654  012777  003714  175450          MOV     #1$,@VECTR2     ;SETUP VECTOR ADDRESS
003662  012777  140000  175426          MOV     #BIT15!BIT14,@STREG     ;CAUSE AN INTERRUPT
003670  017737  175422  001126          MOV     @STREG,$BDDAT   ;BAD DATA
003676  012737  140000  001124          MOV     #BIT15!BIT14,$GDDAT     ;GOOD DATA
003704  104002                          ERROR   2
003706  004737  011250                  JSR     PC,DUMC         ;TYPE COMPLETED
003712  000627                          BR      TST2D
003714  022626                  1$:     CMP     (SP)+,(SP)+     ;POP STACK
003716  004737  011250                  JSR     PC,DUMC
003722  005077  175370                  CLR     @STREG
```

```
1204                                    ;;*****************************************************************
1205                                    ;*TEST 24       TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
1206                                    ;;*****************************************************************
1207  003726  000004                    TST24:  SCOPE
1208  003730  012777  000001  175360            MOV     #BIT0,@STREG    ;START CONVERSION
1209  003736  105777  175354            1$:     TSTB    @STREG          ;WAIT FOR
1210  003742  100375                            BPL     1$
1211  003744  012737  100200  001124    2$:     MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
1212  003752  012777  000001  175336            MOV     #BIT0,@STREG    ;START 2ND CONVERSION
1213  003760  012700  001000                    MOV     #BIT9,R0        ;WAIT FOR 2ND
1214  003764  005300                    3$:     DEC     R0              ;CONVERSION TO END
1215  003766  001376                            BNE     3$
1216  003770  104410                    4$:     CHECK
1217  003772  104001                            ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
1218                                                                    ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
1219  003774  017700  175322                    MOV     @ADBUFF,R0      ;CLEAR DONE FLAG
1220                                    ;;*****************************************************************
1221                                    ;*TEST 25       TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
1222                                    ;;*****************************************************************
1223  004000  000004                    TST25:  SCOPE
1224  004002  012737  100000  001124            MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
1225  004010  012777  000001  175300            MOV     #BIT0,@STREG    ;START CONVERSION
1226  004016  112777  000001  175272            MOVB    #BIT0,@STREG            ;START NEXT CONVERSION
1227  004024  112777  000001  175264            MOVB    #BIT0,@STREG            ;ONCE AGAIN IN CASE REFRESH INTERVENED
1228  004032  017737  175260  001126            MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
1229  004040  042737  077777  001126            BIC     #77777,$BDDAT           ;MASK OUT BIT 15
1230  004046  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1231  004054  001401                            BEQ     1$              ;;BRANCH OVER ERROR
1232  004056  104001                            ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
1233                                                                    ;CONVERT BEGINS BEFORE FIRST DONE
1234  004060  017700  175236            1$:     MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
1235  004064  005077  175226                    CLR     @STREG                  ;CLEAR STATUS REGISTER
1236  004070  000004                            SCOPE
1237  004072  000207                            RTS     PC              ;RETURN TO TEST SECTION
1238
1239
1240                                    ;;SUBROUTINE FOR LOGIC TESTS;;
1241  004074  013777  001124  175214    TESTIT: MOV     $GDDAT,@STREG           ;LOAD EXPECTED VALUE
1242  004102  017737  175210  001126    TEST:   MOV     @STREG,$BDDAT           ;READ ST. REG.
1243  004110  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1244  004116  001002                            BNE     RETERR          ;;ERROR RETURN
1245  004120  062716  000002                    ADD     #2,(SP)                 ;BUMP RETURN ADDRESS TO GET AROUND ERROR
1246  004124  000002                    RETERR: RTI
```

```
                                .SBTTL      WRAPAROUND TEST SECTION
004126                          WRAP:
                                ;;**************************************************************
                                ;*TEST 26       TEST CH0 GROUND
                                ;;**************************************************************
004126  012737  000026  001102  TST26:  MOV     #STN,$TSTNM
004134  012737  000010  001160         MOV     #10.,$TIMES     ;;DO 10 ITERATIONS
004142  012737  004126  001110         MOV     #TST26,$LPERR
004150  012737  004126  001106         MOV     #TST26,$LPADR
004156  004537  011016                 JSR     R5,CONVRT               ;CONVERT 8 TIMES
004162  000000                         0
004164  004537  011130                 JSR     R5,COMPAR               ;COMPARE RESULTS
004170  004000                         4000                            ;NOMINAL
004172  011576                         V12                             ;TOLERANCE
004174  104004                         ERROR   4                       ;ERROR ON A/D CHANNEL
                                ;;**************************************************************
                                ;*TEST 27       TEST CH1 +4.5 VOLT
                                ;;**************************************************************
004176  000004                 TST27:  SCOPE
004200  012737  000010  001160         MOV     #10.,$TIMES     ;;DO 10 ITERATIONS
004206  004537  011016                 JSR     R5,CONVRT               ;CONVERT 8 TIMES
004212  000001                         1                               ;CHANNEL 1
004214  004537  011130                 JSR     R5,COMPAR               ;COMPARE RESULTS
004220  007344                         7344                            ;NOMINAL
004222  011602                         V326                            ;TOLERANCE
004224  104004                         ERROR   4                       ;ERROR ON A/D CHANNEL
                                ;;**************************************************************
                                ;*TEST 30       TEST CH2 -4.5 VOLT
                                ;;**************************************************************
004226  000004                 TST30:  SCOPE
004230  012737  000010  001160         MOV     #10.,$TIMES     ;;DO 10 ITERATIONS
004236  004537  011016                 JSR     R5,CONVRT               ;CONVERT 8 TIMES
004242  000002                         2                               ;CHANNEL 2
004244  004537  011130                 JSR     R5,COMPAR               ;COMPARE RESULTS
004250  000434                         434                             ;NOMINAL
004252  011602                         V326                            ;TOLERANCE
004254  104004                         ERROR   4                       ;ERROR ON A/D CHANNEL
                                ;;**************************************************************
                                ;*TEST 31       TEST GROUND ON CHANNELS 4 - 17
                                ;;**************************************************************
004256  000004                 TST31:  SCOPE
004260  012737  000010  001160         MOV     #10.,$TIMES     ;;DO 10 ITERATIONS
004266  012737  000004  004300         MOV     #4,2$           ;SET UP FIRST CHANNEL
004274  004537  011016         1$:     JSR     R5,CONVRT       ;CONVERT CHANNEL
004300  000004                 2$:     4
004302  004537  011130                 JSR     R5,COMPAR       ;TEST RESULTS
004306  004000                         4000
004310  011576                         V12
004312  104004                         ERROR   4
004314  005237  004300                 INC     2$              ;GET NEXT CHANNEL
004320  022737  000017  004300         CMP     #17,2$          ;DONE?
004326  001362                         BNE     1$              ;;NO
```

```
1300                                    ;;*****************************************************************
1301                                    ;*TEST 32       TEST VERNIER OFFSET DAC ON CH0
1302                                    ;;*****************************************************************
1303  004330  000004                    ↑ST32:  SCOPE
1304  004332  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1305  004340  005077  174756                    CLR     @ADBUFF                 ;SET VERNIER DAC = 0
1306  004344  004537  011016                    JSR     R5,CONVRT               ;CONV. CH0, DIRECT VERNIER DAC
1307  004350  000000                            0
1308  004352  013704  001356                    MOV     TEMP,R4                 ;SAVE VALUE IN R4
1309  004356  012777  000377  174736  1$:       MOV     #377,@ADBUFF            ;SET VERNIER DAC = 377
1310  004364  004537  011016                    JSR     R5,CONVRT               ;CONVERT IT
1311  004370  000000                            0
1312  004372  160437  001356                    SUB     R4,TEMP                 ;TEMP=DIFF. BETWEEN VALUE & PREVIOUS
1313  004376  004537  011130                    JSR     R5,COMPAR               ;COMPARE RESULTS
1314  004402  000005                            5
1315  004404  011572                            V2
1316  004406  104004                            ERROR   4
1317                                    ;;*****************************************************************
1318                                    ;*TEST 33       OFFSET ON CH0
1319                                    ;;*****************************************************************
1320  004410  000004                    ↑ST33:  SCOPE
1321  004412  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1322  004420  013737  001342  001372            MOV     BASECH,CHANL            ;LOAD CHANNEL
1323  004426  013737  001342  001370            MOV     BASECH,DUMMY            ;LOAD DUMMY
1324  004434  004737  005160                    JSR     PC,OFFSET       ;FIND OFFSET
1325  004440  104400  013736                    TYPE    ,MOFSET         ;TYPE "OFFSET="
1326  004444  004737  005234                    JSR     PC,TOFF         ;TYPE OFFSET
1327  004450  004537  011130                    JSR     R5,COMPAR       ;IS RESULT WITHIN LIMITS?
1328  004454  000000                            0
1329  004456  011600                            VSOO
1330  004460  000401                            BR      OFFERR          ;NO-ERROR
1331  004462  000403                            BR      OFFOK           ;YES-OK
1332  004464  104400  012547    OFFERR: TYPE    ,ERMSG
1333  004470  000402                            BR      ↑ST34           ;;GO TO NEXT TEST
1334  004472  104400  012225    OFFOK:  TYPE    ,OKMSG
```

```
                                        ;;*****************************************************************
                                        ;*TEST 34        TEST RAMP RANGE, CH3
                                        ;;*****************************************************************
004476  000004                  TST34:  SCOPE
004500  012737  000001  001160          MOV     #1,$TIMES               ;DO THIS ONCE
004506  012703  007777                  MOV     #7777,R3                ;INIT R3 VALUE
004512  005004                          CLR     R4                      ;AND R4
004514  012777  001400  174574          MOV     #1400,@STREG            ;SETUP FOR CH3
004522  012702  047040                  MOV     #20000.,R2              ;SETUP FOR 20,000 CONVERSIONS
004526  105277  174564          1$:     INCB    @STREG
004532  105777  174560          2$:     TSTB    @STREG
004536  100375                          BPL     2$
004540  027704  174556                  CMP     @ADBUFF,R4
004544  003402                          BLE     3$
004546  017704  174550                  MOV     @ADBUFF,R4              ;HIT A NEW HIGH
004552  027703  174544          3$:     CMP     @ADBUFF,R3
004556  002002                          BGE     4$
004560  017703  174536                  MOV     @ADBUFF,R3              ;HIT A NEW LOW
004564  005302                  4$:     DEC     R2
004566  001357                          BNE     1$
004570  010337  001356                  MOV     R3,TEMP
004574  004537  011130                  JSR     R5,COMPAR
004600  000000                          0
004602  011570                          V0
004604  104004                          ERROR   4                       ;RAMP DIDN'T REACH LOW END OF RANGE
004606  010437  001356                  MOV     R4,TEMP
004612  004537  011130                  JSR     R5,COMPAR
004616  007777                          7777
004620  011570                          V0
004622  104004                          ERROR   4                       ;RAMP DIDN'T REACH HIGH END OF RANGE
```

```
                                        ;;*****************************************************************
                                        ;*TEST 35        NOISE TEST, 1 EDGE
                                        ;*****************************************************************
004624  000004                  TST35:  SCOPE
004626  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004634  104400  011764                  TYPE    ,NOIMSG
004640  005037  001372                  CLR     CHANL           ;LOAD CHANNEL 0
004644  013737  001372  001370  1$:     MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
004652  004737  006714                  JSR     PC,GETEDG               ;GET EDGE VALUE
004656  005037  001404                  CLR     RMS                     ;CLEAR RMS VLAUE
004662  005037  001406                  CLR     PEAK                    ;CLEAR PEAK VALUE
004666  004537  007074                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 16%
004672  000020                          16.
004674  063737  001414  001404          ADD     DAC,RMS         ;ADD RESULT TO RMS
005002  004537  007074                  JSR     R5,$ARSUB               ;DO SAR ROUTINE AT 84%
005006  000124                          84.
005010  163737  001414  001404          SUB     DAC,RMS         ;SUBTRACT RESULT FROM RMS
005016  004537  007074                  JSR     R5,$ARSUB               ;DO SAR ROUTINE AT 1%
005022  000001                          1
005024  063737  001414  001406          ADD     DAC,PEAK                ;ADD RESULT TO PEAK
005032  004537  007074                  JSR     R5,$ARSUB               ;DO SAR ROUTINE AT 99%
005036  000143                          99.
005040  163737  001414  001406          SUB     DAC,PEAK                ;SUBTRACT RESULT FROM PEAK
005046  012737  000001  007072          MOV     #1,EDGFLG
005054  004737  010666                  JSR     PC,TYPRP                ;TYPE RMS AND PEAK VALUES
005060  005237  001372                  INC     CHANL                   ;GET NEXT CHANNEL
005064  022737  000003  001372          CMP     #3,CHANL                ;CHANNEL 3?
005072  001002                          BNE     2$              ;;NO
005074  005237  001372                  INC     CHANL           ;CHANNEL 3 IS SKIPED
005000  022737  000017  001372  2$:     CMP     #17,CHANL       ;DONE?
005006  001316                          BNE     1$              ;;NO
```

```
                                        ;;*********************************************************************
                                        ;*TEST 36        INTERCHANNEL SETTLING TEST, 1 EDGE
                                        ;;*********************************************************************
005010  000004                  TST36:  SCOPE
005012  012737  000001  001160          MOV     #1,STIMES       ;;DO 1 ITERATION
005020  104400  012003                  TYPE    ,SETMSG         ;TYPE "SETTLING TEST"
005024  012737  000001  001360          MOV     #1,CH1          ;DO TEST BETWEEN CHANNEL 1 AND 2
005032  012737  000002  001362          MOV     #2,CH2
005040  013737  001362  001372  1$:     MOV     CH2,CHANL
005046  004737  006714                  JSR     PC,GETEDG       ;GET EDGE VALUES
005052  005002                          CLR     R2
005054  004737  006652                  JSR     PC,SET1A        ;SCALING = .02 LSB
005060  004737  006652                  JSR     PC,SET1A        ;MAKE IT .01 LSB
005064  100001                          BPL     2$
005066  005402                          NEG     R2              ;MAKE IT POSITIVE
005070  010204                  2$:     MOV     R2,R4
005072  012737  000001  007072          MOV     #1,EDGFLG
005100  004737  006522                  JSR     PC,TYPSET       ;TYPE SETTLING INFORMATION
005104  022737  000002  001360          CMP     #2,CH1          ;DONE?
005112  001410                          BEQ     TST37           ;;YES
005114  013702  001360                  MOV     CH1,R2          ;SETTLE THE OTHER WAY
005120  013737  001362  001360          MOV     CH2,CH1
005126  010237  001362                  MOV     R2,CH2
005132  000742                          BR      1$              ;;
005134                          3$:
                                        ;;*********************************************************************
                                        ;*TEST 37        DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
                                        ;;*********************************************************************
005134  000004                  TST37:  SCOPE
005136  012737  000001  001160          MOV     #1,STIMES       ;;DO 1 ITERATION
005144  005737  001202                  TST     $PASS                   ;FIRST TIME-SKIP DIFLIN
005150  001402                          BEQ     LEND
005152  004737  007266                  JSR     PC,DIFLIN
005156  000207                  LEND:   RTS     PC                      ;RETURN TO TEST SECTION

005160  012737  004001  001420  OFFSET: MOV     #4001,EDGE              ;4000,4001 EDGE
005166  004537  007074                  JSR     R5,SARSUB
005172  000062                          50.
005174  013737  001414  001356          MOV     DAC,TEMP
005202  012737  004000  001420          MOV     #4000,EDGE              ;3777,4000 EDGE
005210  004537  007074                  JSR     R5,SARSUB
005214  000062                          50.
005216  063737  001414  001356          ADD     DAC,TEMP
005224  162737  000400  001356          SUB     #400,TEMP
005232  000207                          RTS     PC
```

```
1441  005234  013702  001356          TOFF:  MOV   TEMP,R2
1442  005240  100402                         BMI   1$              ;;IS THE NUMBER POSITIVE?
1443  005242  104400  012545                 TYPE  ,POSITV
1444  005246  104412                  1$:    TYPOC
1445  005250  104400  013751                 TYPE  ,MLSB           ;TYPE ASCIZ STRING
1446  005254  000207                         RTS   PC
1447  005256  005303          TCHK:  DEC   R3               ;DECREMENT COUNT
1448  005260  001005                         BNE   1$               ;;
1449  005262  012703  000005                 MOV   #5,R3           ;RESET COUNT
1450  005266  104400  001171                 TYPE  ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
1451  005272  000402                         BR    2$               ;;
1452  005274  104400  012114           1$:   TYPE  ,SPACE          ;TYPE FOUR (4) SPACES
1453  005300  005037  001416           2$:   CLR   DELAY           ;CLEAR DELAY
1454  005304  005077  173634                 CLR   @#STKS          ;CLEAR INTERRUPT ENABLE
1455  005310  105777  173630           3$:   TSTB  @#STKS          ;IS KEYBOARD FLAG SET?
1456  005314  100404                         BMI   4$               ;;YES
1457  005316  005237  001416                 INC   DELAY           ;IS DELAY ZERO?
1458  005322  001372                         BNE   3$               ;;NO
1459  005324  000416                         BR    6$               ;;
1460  005326  005777  173614           4$:   TST   @#STKB          ;CLEAR FLAG
1461  005332  012777  000100  173604       MOV   #100,@#STKS     ;SET INTERRUPT ENABLE
1462  005340  004537  011130               JSR   R5,COMPAR       ;TEST LAST CONVERSION
1463  005344  000000                       0
1464  005346  011574                       V4                    ;TOLERANCE .04 LSB
1465  005350  000402                       BR    5$               ;;
1466  005352  062716  000002               ADD   #2,(SP)         ;BUMP RETURN ADDRESS
1467  005356  062716  000002           5$:   ADD   #2,(SP)         ;BUMP RETURN ADDRESS 2 WORDS
1468  005362  000207                 6$:   RTS   PC
1469  005364  104400  012236          BEGINC:TYPE  ,CCHAN           ;ASK FOR CHANNEL
1470  005370  104407                       RDOCT                 ;READ CHANNEL NUMBER
1471  005372  012637  001372               MOV   (SP)+,CHANL     ;STORE CHANNEL NUMBER
1472  005376  013737  001372  001370       MOV   CHANL,DUMMY     ;LOAD DUMMY
1473  005404  104400  012264           1$:   TYPE  ,SEL            ;SELECT OFFSET OR GAIN ADJUST
1474  005410  104406                       RDLIN                 ;GET TEST
1475  005412  012600                       MOV   (SP)+,R0        ;MOVE POINTER TO R0
1476  005414  121027  000117               CMPB  (R0),#'O        ;IS IT "O"?
1477  005420  001406                       BEQ   AJOFF            ;;YES, GO TO ADJUST OFFSET
1478  005422  121027  000107               CMPB  (R0),#'G        ;IS IT "G"?
1479  005426  001430                       BEQ   AJGAIN           ;;YES, GO TO ADJUST GAIN
1480  005430  104400  001170               TYPE  ,$QUES          ;TYPE "?"
1481  005434  000763                       BR    1$               ;;
```

```
1482  005436  104400  012377          AJOFF:  TYPE    ,IGND           ;GROUND CHANNEL
1483  005442  104406                          RDLIN                   ;WAIT FOR CR
1484  005444  005726                          TST     (SP)+           ;POP 1 WORD OFF STACK
1485  005446  104400  012337          1$:     TYPE    ,XADJ           ;ADJUST MESSAGE
1486  005452  104400  012436                  TYPE    ,CRWR           ;TYPE "TYPE CR WHEN READY"
1487  005456  012703  000005                  MOV     #5,R3           ;SET UP COUNT
1488  005462  004737  005160          2$:     JSR     PC,OFFSET       ;TEST AND TYPE OFFSET ERROR
1489  005466  004737  005234                  JSR     PC,TOFF         ;TYPE OFFSET
1490  005472  004737  005256                  JSR     PC,TCHK         ;CHECK FOR A CHARACTER AND DELAY
1491  005476  000771                          BR      2$              ;;
1492  005500  000762                          BR      1$              ;;NOT WITHIN TOLLERANCE, TRY AGAIN
1493  005502  000005                          RESET
1494  005504  000137  002262                  JMP     BEG2
1495  005510  104400  012465          AJGAIN: TYPE    ,IVOLT          ;INPUT +5.115 VOLTS ON CHANNEL
1496  005514  104400  012436                  TYPE    ,CRWR
1497  005520  104406                          RDLIN                   ;WAIT FOR CR
1498  005522  005726                          TST     (SP)+           ;POP 1 WORD OFF STACK
1499  005524  104400  012531          1$:     TYPE    ,YADJ           ;ADJUST MESSAGE
1500  005530  104400  012353                  TYPE    ,MOLSB          ;TYPE " FOR 0.00 LSB ERROR"
1501  005534  104400  012436                  TYPE    ,CRWR
1502  005540  012703  000005                  MOV     #5,R3           ;SET UP COUNT
1503  005544  012737  007777  001420  2$:     MOV     #7777,EDGE      ;LOOK FOR 7776,7777 EDGE
1504  005552  004537  007074                  JSR     R5,SARSUB
1505  005556  000062                          50.
1506  005560  013737  001414  001356          MOV     DAC,TEMP        ;SAVE DAC
1507  005566  012737  007776  001420          MOV     #7776,EDGE      ;LOOK FOR 7775,7776 EDGE
1508  005574  004537  007074                  JSR     R5,SARSUB
1509  005600  000062                          50.
1510  005602  063737  001414  001356          ADD     DAC,TEMP        ;ADD RESULTS
1511  005610  162737  000400  001356          SUB     #400,TEMP       ;OFFSET RESULT
1512  005616  004737  005234                  JSR     PC,TOFF         ;TYPE GAIN
1513  005622  004737  005256                  JSR     PC,TCHK         ;CHECK FOR CHARACTER AND DELAY
1514  005626  000746                          BR      2$              ;;
1515  005630  000735                          BR      1$              ;;NOT WITHIN TOLLERANCE, TRY AGAIN
1516  005632  000005                          RESET
1517  005634  000137  002262                  JMP     BEG2
```

```
                                        .SBTTL      PRINT VALUES ROUTINE
005640  012737  005640  001374  BEGINP: MOV     #BEGINP,TADDR           ;TEST ADDRESS IN TADDR
005646  005077  173444                  CLR     @STREG                  ;CLEAR STATUS REGISTER
005652  104400  013645                  TYPE    ,HEADS                  ;TYPE OUT HEADING
005656  005046                          CLR     -(SP)                   ;CLEAR PSW
005660  012746  005666                  MOV     #1$,-(SP)
005664  000002                          RTI
005666  017700  173246          1$:     MOV     @SWR,R0                 ;READ CHANNEL FROM SWITCH REG.
005672  042700  177700                  BIC     #177700,R0              ;ISOLATE MUX BITS
005676  032777  020000  173234          BIT     #BIT13,@SWR             ;IS BIT 13 SET?
005704  001005                          BNE     2$              ;;YES,SKIP TYPEOUT
005706  104400  012111                  TYPE    ,CH
005712  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                                                ;;TYPE CHANNEL
005714  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
005716     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
005717     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
005720  012777  001620  173376  2$:     MOV     #RETURN,@VECTOR         ;ADDRESS AFTER INTRPT.
005726  000300                          SWAB    R0                      ;SWITCH BYTES
005730  052700  000100                  BIS     #BIT6,R0
005734  010077  173356                  MOV     R0,@STREG               ;LOAD THE CHANNEL
005740  012702  000010                  MOV     #10,R2                  ;TYPEOUT COUNTER
005744  005277  173346          3$:     INC     @STREG                  ;START CONVERSION
005750  000001                          WAIT                            ;WAIT FOR INTRPT.
005752  017700  173344                  MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
005756  032777  020000  173154          BIT     #BIT13,@SWR             ;IS BIT 13 SET?
005764  001403                          BEQ     4$                      ;NOT SET, TYPE OUT LIST
005766  010077  173150                  MOV     R0,@DISPLAY                     ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
005772  000735                          BR      1$                      ;REPEAT CONVERSION
005774  104400  012114          4$:     TYPE    ,SPACE
006000  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                                                ;;PRINT OCTAL CONVERTED VALUE
006002  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006004     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
006005     001                          .BYTE   1               ;;TYPE LEADING ZEROS
006006  012701  010000                  MOV     #10000,R1
006012  005301          5$:     DEC     R1
006014  001376                  BNE     5$
006016  005302                  DEC     R2                      ;DECREMENT THE COUNTER
006020  001351                  BNE     3$                      ;NO CARRIAGE RETURN
006022  104400  001171                  TYPE    ,$CRLF                  ;CARRIAGE RETURN
006026  000717                          BR      1$                      ;REPEAT CONVERSION
```

```
1560                                            .SBTTL      LOGIC TEST SECTION
1561  006030  012737  006030  001374  BEGL:     MOV       #BEGL,TADDR       ;TEST ADDRESS
1562  006036  004737  002464                   JSR       PC,TESTAD         ;NO OF ADDITIONAL AD'S
1563  006042  004737  002670          1$:      JSR       PC,BEGINL         ;LOGIC TESTS
1564  006046  004737  006206                   JSR       PC,BUMPAD         ;MORE TO TEST?
1565  006052  000773                           BR        1$                ;TEST NEXT A/D
1566  006054  012737  006042  011626           MOV       #1$,AGTST         ;ADDRESS FOR EOP
1567  006062  000137  011630                   JMP       $EOP              ;TYPE END OF PASS
1568
1569                                            .SBTTL      AUTO TEST
1570  006066  012737  006066  001374  BEGINA:  MOV       #BEGINA,TADDR     ;TEST ADDRESS
1571  006074  004737  002464                   JSR       PC,TESTAD         ;NO. OF AD'S TO BE TESTED
1572  006100  004737  002670          1$:      JSR       PC,BEGINL         ;LOGIC TESTS
1573  006104  104400  013037                   TYPE      ,MEND             ;TYPE END OF LOGIC TEST
1574  006110  013746  001316                   MOV       $TREG,-(SP)       ;SAVE STREG FOR TYPEOUT
1575  006114  104402                           TYPOS                       ;TYPE OCTAL NUMBER
1576  006116     006                           .BYTE     6                 ;TYPE 6 DIGITS
1577  006117     001                           .BYTE     1                 ;TYPE LEADING ZEROS
1578  006120  104400  001171                   TYPE      ,$CRLF            ;TYPE A CR,LF
1579  006124  004737  004126                   JSR       PC,WRAP
1580  006130  004737  006206                   JSR       PC,BUMPAD         ;TEST NEXT A/D
1581  006134  000761                           BR        1$                ;TEST NEXT AD
1582  006136  012737  006100  011626           MOV       #1$,AGTST         ;ADDRESS FOR EOP
1583  006144  000137  011630                   JMP       $EOP              ;TYPE END OF PASS
1584
1585                                            .SBTTL      WRAPAROUND TEST
1586  006150  012737  006150  001374  BEGINW:  MOV       #BEGINW,TADDR     ;TEST ADDRESS
1587  006156  004737  002464                   JSR       PC,TESTAD         ;NO. OF AD'S TO BE TESTED
1588  006162  004737  004126          1$:      JSR       PC,WRAP           ;WRAPAROUND TESTS
1589  006166  004737  006206                   JSR       PC,BUMPAD         ;MORE A/D'S TO BE TESTED?
1590  006172  000773                           BR        1$                ;YES-GO TEST NEXT ADV11
1591  006174  012737  006162  011626           MOV       #1$,AGTST
1592  006202  000137  011630                   JMP       $EOP              ;INCREMENTS $PASS
```

```
1593                                            .SBTTL      DETERMINE IF MORE ADV11'S TO BE TESTED
1594   006206  005737  001364           BUMPAD: TST     NBEXT                   ;ADDITIONAL AD'S?
1595   006212  001434                           BEQ     FIXADR                  ;NO-INITIALIZE ADDRESSES
1596   006214  006337  001440                   ASL     TSTBIT          ;MOVE BIT TO NEXT MODULE
1597   006220  063737  001336  001316           ADD     VADR,STREG              ;SET UP NEW ST. REG.
1598   006226  063737  001336  001320           ADD     VADR,ADST1              ;SET UP NEW ADST1
1599   006234  063737  001336  001322           ADD     VADR,ADBUFF             ;SET UP NEW BUFFER ADDRESS
1600   006242  063737  001340  001324           ADD     VVCT,VECTOR             ;SET UP NEW VECTOR
1601   006250  063737  001340  001330           ADD     VVCT,VECTR1
1602   006256  063737  001340  001332           ADD     VVCT,VECTR2
1603   006264  063737  001340  001334           ADD     VVCT,VECTR3
1604   006272  005077  173032                   CLR     @VECTR1
1605   006276  005337  001364                   DEC     NBEXT                   ;ONE LESS ADV11
1606   006302  000465                           BR      BYPASS
1607   006304  062716  000002           FIXADR: ADD     #2,(SP)
1608   006310  012737  000001  001440  FIXONE: MOV     #1,TSTBIT       ;INITIALIZE MODULE ERROR TEST BIT
1609   006316  013737  001250  001316           MOV     $BASE,STREG             ;RELOAD INITIAL ADDRESSES
1610   006324  013737  001250  001320           MOV     $BASE,ADST1
1611   006332  013737  001250  001322           MOV     $BASE,ADBUFF
1612   006340  005237  001320                   INC     ADST1
1613   006344  062737  000002  001322           ADD     #2,ADBUFF
1614   006352  013737  001244  001324           MOV     $VECT1,VECTOR
1615   006360  042737  170000  001324           BIC     #170000,VECTOR
1616   006366  113737  001245  001326           MOVB    $VECT1+1,BASEBR
1617   006374  105037  001327                   CLRB    BASEBR+1                ;CLEAR HIGH BYTE
1618   006400  013737  001324  001330           MOV     VECTOR,VECTR1
1619   006406  062737  000002  001330           ADD     #2,VECTR1
1620   006414  013737  001324  001332           MOV     VECTOR,VECTR2
1621   006422  062737  000004  001332           ADD     #4,VECTR2
1622   006430  013737  001324  001334           MOV     VECTOR,VECTR3
1623   006436  062737  000006  001334           ADD     #6,VECTR3
1624   006444  005077  172660                   CLR     @VECTR1
1625   006450  013737  001366  001364           MOV     NMBEXT,NBEXT            ;RESET COUNTER
1626                                   ;;LOAD .+2 AND HALT TRAP CATCH;;
1627   006456  012700  000216           BYPASS: MOV     #216,R0                 ;FILL .+2
1628   006462  012701  000214                   MOV     #214,R1                 ;LOAD HALT
1629   006466  020137  001344           1$:     CMP     R1,KBVECT
1630   006472  001410                           BEQ     2$
1631   006474  010021                           MOV     R0,(R1)+
1632   006476  005021                           CLR     (R1)+
1633   006500  010100                           MOV     R1,R0
1634   006502  005720                           TST     (R0)+
1635   006504  020027  001002                   CMP     R0,#1002
1636   006510  001366                           BNE     1$
1637   006512  000207                           RTS     PC                      ;TEST NEXT A/D
1638   006514  022021           2$:     CMP     (R0)+,(R1)+
1639   006516  022021                   CMP     (R0)+,(R1)+
1640   006520  000762                   BR      1$
```

```
1641  006522  104412                  TYPSET: TYPDC
1642  006524  104400  012121                  TYPE    ,LSB
1643  006530  013746  001362                  MOV     CH2,-(SP)       ;;SAVE CH2 FOR TYPEOUT
1644                                                                  ;;TYPE CH
1645  006534  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1646  006536     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1647  006537     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1648  006540  104400  013757                  TYPE    ,MAT            ;TYPE ASCIZ STRING
1649  006544  004737  007030                  JSR     PC,TYPEDG
1650  006550  104400  012134                  TYPE    ,SETCH
1651  006554  013746  001360                  MOV     CH1,-(SP)       ;;SAVE CH1 FOR TYPEOUT
1652                                                                  ;;TYPE CH
1653  006560  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1654  006562     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1655  006563     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1656  006564  104400  012156                  TYPE    ,ATMSG
1657  006570  013737  001360  006616          MOV     CH1,1$
1658  006576  163737  001342  006616          SUB     BASECH,1$
1659  006604  012777  000200  172510          MOV     #200,@ADBUFF
1660  006612  004537  011016                  JSR     R5,CONVRT
1661  006616  000000                  1$:     0
1662  006620  013746  001356                  MOV     TEMP,-(SP)      ;;SAVE TEMP FOR TYPEOUT
1663                                                                  ;;TYPE VALUE
1664  006624  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1665  006626     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
1666  006627     001                          .BYTE   1               ;;TYPE LEADING ZEROS
1667  006630  020437  011610                  CMP     R4,VSET
1668  006634  003003                          BGT     ERR
1669  006636  104400  012225                  TYPE    ,OKMSG
1670  006642  000207                          RTS     PC
1671  006644  104400  012547          ERR:    TYPE    ,ERMSG
1672  006650  000207                          RTS     PC
1673
1674                                  ;;SUBROUTINE FOR SETTLING TESTS;;
1675  006652  013737  001362  001370  SETIA:  MOV     CH2,DUMMY               ;LOAD DUMMY
1676  006660  004537  007074                  JSR     R5,$ARSUB               ;DO SAR ROUTINE AT 50%
1677  006664  000062                          50.
1678  006666  063702  001414                  ADD     DAC,R2          ;ADD RESULT TO R2
1679  006672  013737  001360  001370          MOV     CH1,DUMMY               ;CHANGE DUMMY VALUE
1680  006700  004537  007074                  JSR     R5,$ARSUB               ;DO SAR ROUTINE AT 50%
1681  006704  000062                          50.
1682  006706  163702  001414                  SUB     DAC,R2          ;SUBTRACT RESULT FROM R2
1683  006712  000207                          RTS     PC              ;RETURN
```

```
                                        ;SUBROUTINE TO GET EDGE VALUE
                                        ;CALL=JSR   PC,GETEDG
                                        ;CONVERSIONS ON A/D CHANNEL 'CHANL'
                                        ;RESULT IN EDGE, USES R0
006714  012777  000200  172400  GETEDG: MOV     #200,@ADBUFF    ;LOAD VERNIER DAC
006722  113700  001372                  MOVB    CHANL,R0                ;GET CHANNEL
006726  000300                          SWAB    R0              ;SET UP A.D STATUS REG.
006730  052700  000100                  BIS     #100,R0         ;ENABLE INTRPT.
006734  010077  172356                  MOV     R0,@STREG
006740  012700  000100                  MOV     #100,R0         ;DAC SETTLING DELAY
006744  005300                  1$:     DEC     R0
006746  001376                          BNE     1$
006750  005037  001420                  CLR     EDGE
006754  012700  000010                  MOV     #10,R0
006760  012777  001620  172336          MOV     #RETURN,@VECTOR         ;RETURN ADDRESS
006766  005277  172324          CONV:   INC     @STREG                  ;START CONVERSION
006772  000001                          WAIT                            ;WAIT FOR INTERRUPT
006774  067737  172322  001420          ADD     @ADBUFF,EDGE
007002  005300                          DEC     R0
007004  001370                          BNE     CONV
007006  006237  001420                  ASR     EDGE
007012  006237  001420                  ASR     EDGE
007016  006237  001420                  ASR     EDGE
007022  005537  001420                  ADC     EDGE
007026  000207                          RTS     PC

                                        ;;SUBROUTINE TO TYPE EDGE VALUES;;
007030  013703  001420          TYPEDG: MOV     EDGE,R3
007034  010346                          MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
                                                                ;;TYPE OCTAL VALUE OF EDGE
007036  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
007040     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
007041     001                          .BYTE   1               ;;TYPE LEADING ZEROS
007042  023727  007072  000001          CMP     EDGFLG,#1
007050  001407                          BEQ     RET
007052  062703  000007                  ADD     #7,R3
007056  104400  012025                  TYPE    ,MINUS          ;TYPE ASCIZ STRING
007062  010346                          MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
                                                                ;;TYPE EDGE VALUE
007064  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
007066     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
007067     001                          .BYTE   1               ;;TYPE LEADING ZEROS
007070  000207          RET:    RTS     PC
007072  000000          EDGFLG: 0
```

```
1728                                    ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
1729                                    ;CALL=JSR   R5,SARSUB
1730                                    ;       XXX;XXX=PERCENT
1731                                    ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
1732  007074  012537  001432            SARSUB: MOV     (R5)+,PERCNT            ;GET PERCENT
1733  007100  006337  001432                    ASL     PERCNT
1734  007104  006337  001432                    ASL     PERCNT
1735  007110  006337  001432                    ASL     PERCNT                  ;RESCALE PERCENT FOR 1600.
1736  007114  006337  001432                    ASL     PERCNT                  ;POINTS PER BURST
1737  007120  012737  000200  001422    SAR1:   MOV     #200,BITPNT             ;INITIALIZE BIT POINTER AT MSB
1738  007126  005037  001414                    CLR     DAC                     ;INITIALIZE DAC VALUE
1739  007132  005000                    TRY:    CLR     R0
1740  007134  063737  001422  001414            ADD     BITPNT,DAC              ;TRY BIT
1741  007142  013777  001414  172152            MOV     DAC,@ADBUFF
1742  007150  012701  003100                    MOV     #1600.,R1               ;SET UP FOR 1600. CONVERSIONS
1743  007154  113777  001370  172136  NXTCVT:   MOVB    DUMMY,@ADST1            ;PRESET MUX TO DUMMY CHANNEL
1744  007162  012777  001620  172134            MOV     #RETURN,@VECTOR         ;RETURN ADDRESS
1745  007170  052777  000101  172120            BIS     #101,@STREG             ;CONVERSION ON DUMMY CHANNEL
1746  007176  000001                            WAIT                            ;WAIT FOR INTERRUPT
1747  007200  017704  172116                    MOV     @ADBUFF,R4              ;DUMMY READ
1748  007204  013704  001372                    MOV     CHANL,R4
1749  007210  000304                            SWAB    R4
1750  007212  052704  000101                    BIS     #101,R4                 ;INTERRUPT ENABLE START
1751  007216  010477  172074                    MOV     R4,@STREG               ;JUMP TO CHANNEL + START CONVERT
1752  007222  000001                            WAIT                            ;WAIT FOR INTERRUPT
1753  007224  027737  172072  001420            CMP     @ADBUFF,EDGE
1754  007232  002001                            BGE     2$
1755  007234  005200                            INC     R0                      ;COUNT RESULTS .LT. EDGE
1756  007236  005301                    2$:     DEC     R1
1757  007240  001345                            BNE     NXTCVT
1758  007242  020037  001432                    CMP     R0,PERCNT
1759  007246  003003                            BGT     SHIFT
1760  007250  163737  001422  001414            SUB     BITPNT,DAC              ;TAKE THE BIT OUT
1761  007256  006237  001422            SHIFT:  ASR     BITPNT
1762  007262  001323                            BNE     TRY
1763  007264  000205                            RTS     R5
```

```
1764
1765  007266  104400  013162          ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
1766  007272  013702  001376  DIFLIN: TYPE   ,MSG20
1767  007276  013704  001400          MOV    RNA,R2          ;SET UP RANDOM NUMBER GENERATOR
1768  007302  013705  001402          MOV    RNB,R4
1769  007306  012700  017754          MOV    RNC,R5
1770  007312  012701  010000          MOV    #BUFFER,R0
1771  007316  005020                  MOV    #4096.,R1       ;4096 WORDS FOR HISTOGRAM
1772  007320  005301          CLEAR1: CLR    (R0)+           ;CLEAR BUFFER AREA
1773  007322  001375                  DEC    R1
1774  007324  012700  017134          BNE    CLEAR1
1775  007330  012701  000310          MOV    #DIST,R0        ;DISTRIBUTION BUFFER POINTER
1776  007334  005003                  MOV    #200.,R1        ;200. WORDS FOR DISTRIBUTION
1777  007336  005037  001434          CLR    R3
1778  007342  005037  001346          CLR    OUT
1779  007346  005037  001350          CLR    WIDE
1780  007352  005037  001352          CLR    NARROW
1781  007356  005037  001354          CLR    FIRST
1782  007362  005020          CLEAR2: CLR    SKIPST
1783  007364  005301                  CLR    (R0)+           ;CLEAR DISTRIBUTION BUFFER AREA
1784  007366  001375                  DEC    R1
1785  007370  012700  000003          BNE    CLEAR2
1786  007374  063700  001342  CHANNL: MOV    #3,R0           ;CHANNEL 3
1787  007400  000300                  ADD    BASECH,R0
1788  007402  052700  000100          SWAB   R0                              ;LOAD MUX BITS
1789  007406  010077  171704          BIS    #100,R0
1790  007412  012700  001440          MOV    R0,@STREG
1791  007416  012777  001620  171700  MOV    #800.,R0        ;NOMINAL STATE WIDTH - 1 LSB
1792  007424  012701  007776  AGAIN:  MOV    #RETURN,@VECTOR
1793  007430  060402          NEXT:   MOV    #4094.,R1
1794  007432  060502                  ADD    R4,R2
1795  007434  005502                  ADD    R5,R2
1796  007436  060204                  ADC    R2
1797  007440  060504                  ADD    R2,R4
1798  007442  005504                  ADD    R5,R4
1799  007444  060205                  ADC    R4
1800  007446  060405                  ADD    R2,R5
1801  007450  005505                  ADD    R4,R5
1802  007452  010246                  ADC    R5
1803  007454  042702  177760          MOV    R2,-(SP)
1804  007460  001402                  BIC    #177760,R2      ;MASK IT TO 4 BITS ONLY
1805  007462  005302          DELAY3: BEQ    CONVR
1806  007464  001376                  DEC    R2              ;STALL
1807  007466  005277  171624  CONVR:  BNE    DELAY3          ;TIME
1808  007472  000001                  INC    @STREG          ;START CONVERSION
1809  007474  017702  171622          WAIT
1810  007500  001413                  MOV    @ADBUFF,R2      ;GET CONVERTED VALUE
1811  007502  020227  007777          BEQ    DELAY1          ;IGNORE IF =0
1812  007506  001413                  CMP    R2,#7777        ;IGNORE IF =7777
1813  007510  006302                  BEQ    DELAY2
1814  007512  005262  017754          ASL    R2
1815  007516  100013                  INC    BUFFER(R2)      ;MAKE HISTOGRAM
1816  007520  012762  077777  017754  BPL    OKAY
1817  007526  000407                  MOV    #077777,BUFFER(R2)      ;PREVENT OVERFLOW
                                      BR     OKAY
```

```
 1818  007530  020227  007777      DELAY1: CMP     R2,#7777        ;EQUALIZE LOOP TIME
 1819  007534  001400                  BEQ     DELAY2          ;WITH DUMMY INSTR.
 1820  007536  005201              DELAY2: INC     R1
 1821  007540  005263  001356          INC     TEMP(R3)
 1822  007544  100404                  BMI     NOTOK
 1823  007546  012602              OKAY:   MOV     (SP)+,R2        ;POP RANDOM NUMBER FROM STACK
 1824  007550  005301                  DEC     R1
 1825  007552  001326                  BNE     NEXT
 1826  007554  000403                  BR      AROUND
 1827  007556  005037  001356      NOTOK:  CLR     TEMP
 1828  007562  000771                  BR      OKAY
 1829  007564  005300              AROUND: DEC     R0
 1830  007566  001316                  BNE     AGAIN
 1831  007570  012700  007776          MOV     #4094.,R0
 1832  007574  012701  017756          MOV     #BUFFER+2,R1
 1833  007600  012102              READ:   MOV     (R1)+,R2        ;GET STATE WIDTH
 1834  007602  006202                  ASR     R2              ;1 LSB = 800.
 1835  007604  006202                  ASR     R2
 1836  007606  006202                  ASR     R2
 1837  007610  005502                  ADC     R2              ;1 LSB = 100.
 1838  007612  020227  000310          CMP     R2,#200.        ;OUT OF RANGE?
 1839  007616  002403                  BLT     INRNGE
 1840  007620  005237  001434          INC     OUT             ;YES - INCREMENT COUNTER
 1841  007624  000423                  BR      TYPBAD
 1842  007626  006302              INRNGE: ASL     R2
 1843  007630  005262  017134          INC     DIST(R2)        ;MAKE STATE WIDTH DISTRIBUTION
 1844  007634  006202                  ASR     R2
 1845  007636  020227  000062          CMP     R2,#50.         ;IS IT 1/2 LSB?
 1846  007642  002007                  BGE     NOTNAR
 1847  007644  005237  001350          INC     NARROW
 1848  007650  005702                  TST     R2              ;IS IT A SKIPPED STATE?
 1849  007652  001002                  BNE     31$
 1850  007654  005237  001354          INC     SKIPST
 1851  007660  000405              31$:    BR      TYPBAD
 1852  007662  020227  000226      NOTNAR: CMP     R2,#150.        ;IS IT 1.5 LSB?
 1853  007666  003425                  BLE     LAST
 1854  007670  005237  001346          INC     WIDE
 1855  007674  005737  001352      TYPBAD: TST     FIRST
 1856  007700  001004                  BNE     60$
 1857  007702  005237  001352          INC     FIRST
 1858  007706  104400  012071          TYPE    ,STATE
 1859  007712  010103              60$:    MOV     R1,R3
 1860  007714  162703  017756          SUB     #BUFFER+2,R3
 1861  007720  006203                  ASR     R3
 1862  007722  010346                  MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
 1863                                                          ;;TYPE STATE
 1864  007724  104402                  TYPOS                   ;;GO TYPE--OCTAL ASCII
 1865  007726     004                  .BYTE   4               ;;TYPE 4 DIGIT(S)
 1866  007727     001                  .BYTE   1               ;;TYPE LEADING ZEROS
 1867  007730  104400  012065          TYPE    ,DASH
 1868  007734  104412                  TYPDC
 1869  007736  104400  012056          TYPE    ,LSBMSG
```

```
 1870  007742  005300                    LAST:   DEC     R0
 1871  007744  001315                            BNE     READ
 1872  007746  112737  000177  014554            MOVB    #177,DECPNT
 1873  007754  013702  001354                    MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
 1874  007760  104412                            TYPDC                           ;TYPE IT
 1875  007762  104400  012564                    TYPE    ,SKPMSG         ;TYPE MESSAGE
 1876  007766  005737  001354                    TST     SKIPST
 1877  007772  001403                            BEQ     1$
 1878  007774  104400  012547                    TYPE    ,ERMSG          ;TYPE "ERROR"
 1879  010000  000402                            BR      NAR
 1880  010002  104400  012225            1$:     TYPE    ,OKMSG          ;TYPE #OK#
 1881  010006  013702  001350            NAR:    MOV     NARROW,R2               ;GET NO. OF NARROW STATES
 1882  010012  104412                            TYPDC                           ;TYPE IT
 1883  010014  104400  012606                    TYPE    ,NARMSG         ;TYPE MESSAGE
 1884  010020  013702  001346                    MOV     WIDE,R2
 1885  010024  063702  001434                    ADD     OUT,R2
 1886  010030  104412                            TYPDC                           ;TYPE NO. OF WIDE STATES
 1887  010032  104400  012645                    TYPE    ,WIDMSG         ;TYPE MESSAGE
 1888  010036  013702  001434                    MOV     OUT,R2
 1889  010042  104412                            TYPDC                           ;TYPE NO. OF STATES OUTSIDE 2 LSB
 1890  010044  104400  012704                    TYPE    ,OUTMSG         ;TYPE MESSAGE
 1891  010050  005737  001434                    TST     OUT
 1892  010054  001403                            BEQ     11$
 1893  010056  104400  012547                    TYPE    ,ERMSG          ;TYPE "ERROR"
 1894  010062  000402                            BR      HALF
 1895  010064  104400  012225            11$:    TYPE    ,OKMSG          ;TYPE "OK"
 1896  010070  013702  001350            HALF:   MOV     NARROW,R2
 1897  010074  063702  001346                    ADD     WIDE,R2
 1898  010100  063702  001434                    ADD     OUT,R2
 1899  010104  010200                            MOV     R2,R0
 1900  010106  104412                            TYPDC                           ;TYPE NO. OF STATES OUTSIDE LIMITS
 1901  010110  112737  000056  014554            MOVB    #56,DECPNT
 1902  010116  104400  012737                    TYPE    ,HAFMSG
 1903  010122  020027  000051                    CMP     R0,#41.         ;COMPARE IT TO NOMINAL
 1904  010126  003403                            BLE     21$
 1905  010130  104400  012547                    TYPE    ,ERMSG          ;TYPE "ERROR"
 1906  010134  000402                            BR      SWDIST
 1907  010136  104400  012225            21$:    TYPE    ,OKMSG          ;TYPE "OK"
 1908  010142  005737  001410            SWDIST: TST     FLAG            ;VT55?
 1909  010146  001426                            BEQ     RELACC
 1910  010150  004737  010626                    JSR     PC,DELCLR       ;WAIT AWHILE, THEN CLEAR VT55
 1911  010154  104400  013214                    TYPE    ,MSG16
 1912  010160  104400  014006                    TYPE    ,BUFF1          ;TYPE BUFF1-PRINT GRID
 1913  010164  012700  017134                    MOV     #DIST,R0        ;POINTER TO STATE WIDTH DISTRIBUTION
 1914  010170  012701  000310                    MOV     #200.,R1        ;GO 200. TIMES UP TO 2 LSB
 1915  010174  012002            NXTY1:  MOV     (R0)+,R2
 1916  010176  004737  011322                    JSR     PC,LOADY
 1917  010202  005002                            CLR     R2
 1918  010204  004737  011322                    JSR     PC,LOADY
 1919  010210  005301                            DEC     R1
 1920  010212  001370                            BNE     NXTY1
 1921  010214  104400  013727                    TYPE    ,C2             ;TYPE ASCIZ STRING
 1922  010220  004737  010626                    JSR     PC,DELCLR
```

```
                                        ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR

010224  005001                RELACC: CLR     R1              ;RUNNING ERROR = 0
010226  005003                        CLR     R3              ;MAXIMUM ERROR = 0
010230  104400  013577                TYPE    ,MSG21
010234  012700  017756                MOV     #BUFFER+2,R0
010240  011002                NXTSTA: MOV     (R0),R2         ;STATE WIDTH = R2
010242  162702  001440                SUB     #800.,R2        ;STATE WIDTH ERROR IN R2
010246  060201                        ADD     R2,R1           ;UPDATE RUNNING ERROR
010250  010120                        MOV     R1,(R0)+        ;SAVE IN BUFFER
010252  010104                        MOV     R1,R4           ;SAVE IN R4 ALSO
010254  100001                        BPL     PLUS            ;IS IT POSITIVE?
010256  005404                        NEG     R4              ;NO - MAKE IT POSITIVE
010260  020403                PLUS:   CMP     R4,R3           ;CHECK AGAINST PREVIOUS MAX. ERROR
010262  003405                        BLE     NOTNEW          ;NOT A NEW MAXIMUM
010264  010403                        MOV     R4,R3           ;UPDATE MAXIMUM IN R3
010266  010005                        MOV     R0,R5
010270  162705  017756                SUB     #BUFFER+2,R5
010274  006205                        ASR     R5              ;R5=EDGE VALUE AT MAX. RELACC
010276  020027  037752        NOTNEW: CMP     R0,#BUFFER+8190.  ;DONE?
010302  001356                        BNE     NXTSTA          ;NO - REPEAT
010304  006203                        ASR     R3              ;RESCALE FROM 1 LSB = 800. SCALING
010306  006203                        ASR     R3              ;TO 1 LSB = 100. SCALING
010310  006203                        ASR     R3
010312  005503                        ADC     R3
010314  010302                        MOV     R3,R2
010316  104412                        TYPDC
010320  104400  013624                TYPE    ,LINEA
010324  010546                        MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
                                                                ;;TYPE VALUE
010326  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
010330      004                       .BYTE   4               ;;TYPE 4 DIGIT(S)
010331      001                       .BYTE   1               ;;TYPE LEADING ZEROS
010332  104400  012223                TYPE    ,SLASH          ;PRINT '/'
010336  005205                        INC     R5
010340  010546                        MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
                                                                ;;TYPE VALUE
010342  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
010344      004                       .BYTE   4               ;;TYPE 4 DIGIT(S)
010345      001                       .BYTE   1               ;;TYPE LEADING ZEROS
010346  020337  011612                CMP     R3,VLIN
010352  003403                        BLE     41$
010354  104400  012547                TYPE    ,ERMSG
010360  000402                        BR      42$
010362  104400  012225        41$:    TYPE    ,OKMSG
010366  005737  001410        42$:    TST     FLAG            ;VT55?
010372  001503                        BEQ     LO2
010374  012700  017754                MOV     #BUFFER,R0
010400  012701  010000                MOV     #4096.,R1
```

```
 1972  010404  011002                GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
 1973  010406  006202                        ASR     R2              ;RESCALE IT TO 1 LSB = 100.
 1974  010410  006202                        ASR     R2
 1975  010412  006202                        ASR     R2
 1976  010414  005502                        ADC     R2
 1977  010416  062702  000166              ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
 1978  010422  010220                        MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
 1979  010424  005301                        DEC     R1
 1980  010426  001366                        BNE     GETDAT
 1981  010430  012700  017754              MOV     #BUFFER,R0
 1982  010434  012704  017754              MOV     #BUFFER,R4
 1983  010440  012705  017756              MOV     #BUFFER+2,R5
 1984  010444  012701  001000              MOV     #512.,R1
 1985  010450  012702  000007      NXT8:   MOV     #7.,R2
 1986  010454  012003                        MOV     (R0)+,R3
 1987  010456  010337  001424              MOV     R3,MIN          ;MINIMUM
 1988  010462  010337  001430              MOV     R3,MAX          ;MAXIMUM
 1989  010466  012003              NXTCMP: MOV     (R0)+,R3
 1990  010470  020337  001424              CMP     R3,MIN
 1991  010474  002002                        BGE     MAXTST
 1992  010476  010337  001424              MOV     R3,MIN          ;NEW MINIMUM
 1993  010502  020337  001430      MAXTST: CMP     R3,MAX
 1994  010506  003402                        BLE     TST8
 1995  010510  010337  001430              MOV     R3,MAX          ;NEW MAXIMUM
 1996  010514  005302              TST8:   DEC     R2
 1997  010516  001363                        BNE     NXTCMP
 1998  010520  013724  001424              MOV     MIN,(R4)+
 1999  010524  013725  001430              MOV     MAX,(R5)+
 2000  010530  022425                        CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
 2001  010532  005301                        DEC     R1
 2002  010534  001345                        BNE     NXT8
 2003  010536  104400  013122              TYPE    ,MSG18
 2004  010542  104400  014034              TYPE    ,BUFF2          ;TYPE BUFF2
 2005  010546  012700  017754              MOV     #BUFFER,R0
 2006  010552  004737  010604              JSR     PC,LOAD
 2007  010556  104400  013734              TYPE    ,C3             ;TYPE ASCIZ STRING
 2008  010562  012700  017756              MOV     #BUFFER+2,R0
 2009  010566  004737  010604              JSR     PC,LOAD
 2010  010572  104400  013727              TYPE    ,C2             ;TYPE ASCIZ STRING
 2011  010576  004737  010626              JSR     PC,DELCLR
 2012  010602  000207              L02:    RTS     PC
 2013  010604  012701  001000      LOAD:   MOV     #512.,R1
 2014  010610  012002              LOADO:  MOV     (R0)+,R2
 2015  010612  005720                        TST     (R0)+
 2016  010614  004737  011322              JSR     PC,LOADY
 2017  010620  005301                        DEC     R1
 2018  010622  001372                        BNE     LOADO
 2019  010624  000207                        RTS     PC
```

```
 2020  010626  032777  010000  170304  DELCLR: BIT     #BIT12,@SWR     ;TEST FOR HALT FOR DISPLAY
 2021  010634  001402                          BEQ     1$              ;;DON'T HALT FOR DISPLAY
 2022  010636  000000                          HALT
 2023  010640  000407                          BR      3$              ;;
 2024  010642  005000                  1$:     CLR     R0
 2025  010644  012701  000020                  MOV     #20,R1          ;DELAY BEFORE CLEANING SCREEN
 2026  010650  005300                  2$:     DEC     R0
 2027  010652  001376                          BNE     2$
 2028  010654  005301                          DEC     R1
 2029  010656  001374                          BNE     2$
 2030  010660  104400  014054          3$:     TYPE    ,VTINIT
 2031  010664  000207                          RTS     PC
 2032                                  ;;TYPE RMS AND PEAK VALUES;;
 2033  010666  104400  012163          TYPRP:  TYPE    ,NOI
 2034  010672  005737  001404                  TST     RMS
 2035  010676  100002                          BPL     POSRMS
 2036  010700  005037  001404                  CLR     RMS             ;RMS<0,SET RMS=0
 2037  010704  005737  001406          POSRMS: TST     PEAK
 2038  010710  100002                          BPL     POSPEA
 2039  010712  005037  001406                  CLR     PEAK            ;PEAK<0,SET PEAK=0
 2040  010716  013702  001404          POSPEA: MOV     RMS,R2
 2041  010722  104412                          TYPDC
 2042  010724  104400  013006                  TYPE    ,MESR           ;TYPE " LSB RMS, "
 2043  010730  013702  001406                  MOV     PEAK,R2
 2044  010734  104412                          TYPDC
 2045  010736  104400  013021                  TYPE    ,MESP           ;TYPE " LSB PEAK AT "
 2046  010742  004737  007030                  JSR     PC,TYPEDG
 2047  010746  104400  012173                  TYPE    ,CHAN           ;TYPE " ON CHANNEL "
 2048  010752  013746  001372                  MOV     CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
 2049                                                                  ;;TYPE CHANL
 2050  010756  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 2051  010760     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
 2052  010761     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 2053  010762  023737  001404  011604          CMP     RMS,VNR         ;WITHIN LIMITS?
 2054  010770  003007                          BGT     ER
 2055  010772  023737  001406  011606          CMP     PEAK,VNP        ;WITHIN LIMITS?
 2056  011000  003003                          BGT     ER
 2057  011002  104400  012225                  TYPE    ,OKMSG
 2058  011006  000207                          RTS     PC
 2059  011010  104400  012547          ER:     TYPE    ,ERMSG
 2060  011014  000207                          RTS     PC
```

```
                                        ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
011016  012500                  CONVRT: MOV     (R5)+,R0                ;GET CHANNEL VALUE
011020  063700  001342                  ADD     BASECH,R0
011024  010037  001372                  MOV     R0,CHANL
011030  000300                          SWAB    R0
011032  005037  001356                  CLR     TEMP
011036  010077  170254                  MOV     R0,@STREG               ;LOAD CHANNEL INTO MIX BITS
011042  012700  010000                  MOV     #10000,R0
011046  005300                  2$:     DEC     R0
011050  001376                          BNE     2$
011052  012777  001620  170244          MOV     #RETURN,@VECTOR         ;LOAD VECTOR
011060  012700  000010                  MOV     #10,R0                  ;SET UP COUNTER
011064  152777  000101  170224  1$:     BISB    #101,@STREG             ;SET INTRPT. EN., START CONV.
011072  000001                          WAIT                            ;WAIT FOR CONVERSION
011074  067737  170222  001356          ADD     @ADBUFF,TEMP            ;READ BUFFER
011102  005300                          DEC     R0
011104  001367                          BNE     1$                      ;DO 8 TIMES
011106  006237  001356                  ASR     TEMP                    ;AVERAGE VALUE
011112  006237  001356                  ASR     TEMP
011116  006237  001356                  ASR     TEMP
011122  005537  001356                  ADC     TEMP
011126  000205                          RTS     R5                      ;RETURN

                                        ;COMPARE SGDDAT AND SBDDAT;;
011130  012537  001124          COMPAR: MOV     (R5)+,$GDDAT            ;GET GOOD DATA
011134  013537  001412                  MOV     @(R5)+,SPREAD           ;GET SPREAD
011140  013737  001356  001126          MOV     TEMP,$BDDAT             ;GET BAD(ACTUAL) DATA
011146  013701  001126                  MOV     $BDDAT,R1
011152  013700  001124                  MOV     $GDDAT,R0
011156  160100                          SUB     R1,R0                   ;GET DIFFERENCE
011160  100001                          BPL     7$
011162  005400                          NEG     R0
011164  020037  001412          7$:     CMP     R0,SPREAD               ;COMPARE IT TO SPREAD
011170  003001                          BGT     10$                     ;GO TO ERROR PRINTOUT
011172  005725                          TST     (R5)+                   ;BUMP RETURN POINTER AROUND ERROR CALL
011174  000205                  10$:    RTS     R5
```

```
2097                                        ;;SUBROUTINE TO TYPE INTRPT. TST MSG.;;
2098  011176  005737  001202          DUMW:  TST     $PASS
2099  011202  001021                        BNE     20$
2100  011204  012737  011246  001110        MOV     #20$,$LPERR
2101  011212  012737  011246  001106        MOV     #20$,$LPADR
2102  011220  104400  013764                TYPE    ,METST          ;TYPE ASCIZ STRING
2103  011224  010046                        MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
2104                                                                ;;TYPE TEST NO.
2105  011226  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
2106  011230     002                        .BYTE   2               ;;TYPE 2 DIGIT(S)
2107  011231     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
2108  011232  104400  013063                TYPE    ,ONAD
2109  011236  013746  001316                MOV     $TREG,-(SP)     ;;SAVE STREG FOR TYPEOUT
2110                                                                ;;TYPE BUS ADDRESS
2111  011242  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
2112  011244     006                        .BYTE   6               ;;TYPE 6 DIGITS
2113  011245     001                        .BYTE   1               ;;TYPE LEADING ZEROS
2114  011246  000207          20$:   RTS     PC
2115
2116  011250  005737  001202          DUMC:  TST     $PASS
2117  011254  001010                        BNE     30$
2118  011256  012737  011276  001110        MOV     #30$,$LPERR
2119  011264  012737  011276  001106        MOV     #30$,$LPADR
2120  011272  104400  012210                TYPE    ,DONE
2121  011276  000207          30$:   RTS     PC
```

```
                                        ;SUBROUTINE TO RESET & SET INTRPT. EN.;
011300  000005                  RST:    RESET
011302  052777  000100  167634          BIS     #100,@$TKS
011310  005046                          CLR     -(SP)           ;CLEAR PSW
011312  012746  011320                  MOV     #1$,-(SP)
011316  000002                          RTI
011320  000207                  1$:     RTS     PC

                                        ;SUBROUTINE LOADY;
011322  005702                  LOADY:  TST     R2              ;ROUTINE TO LOAD VLAUE INTO R2
011324  100001                          BPL     PLUSR2          ;AS A VT55 Y-VALUE
011326  005002                          CLR     R2
011330  020227  000353          PLUSR2: CMP     R2,#235.
011334  002402                          BLT     LESS
011336  012702  000353                  MOV     #235.,R2
011342  010203                  LESS:   MOV     R2,R3
011344  042702  177740                  BIC     #177740,R2
011350  052702  000040                  BIS     #40,R2
011354  105777  167570          B10:    TSTB    @$TPS           ;PRINT CHARACTER
011360  100375                          BPL     B10
011362  110277  167564                  MOVB    R2,@$TPB
011366  006203                          ASR     R3
011370  006203                          ASR     R3
011372  006203                          ASR     R3
011374  006203                          ASR     R3
011376  006203                          ASR     R3
011400  042703  177770                  BIC     #177770,R3
011404  052703  000040                  BIS     #40,R3
011410  105777  167534          B11:    TSTB    @$TPS           ;PRINT CHARACTER
011414  100375                          BPL     B11
011416  110377  167530                  MOVB    R3,@$TPB
011422  000207                          RTS     PC
```

```
2154                                        ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
2155                                        ;;IN R2 AS X.XX;;
2156  011424  005702                DECTYP: TST     R2              ;TEST VALUE TO BE TYPED
2157  011426  100003                        BPL     POS
2158  011430  104400  012025              TYPE    ,MINUS                  ;TYPE MINUS SIGN
2159  011434  005402                        NEG     R2
2160  011436  020227  001747        POS:    CMP     R2,#999.                ;>999. REPLACE IT WITH 999.
2161  011442  003402                        BLE     OKAYD
2162  011444  012702  001747              MOV     #999.,R2
2163  011450  105037  014556        OKAYD:  CLRB    ONES            ;CLEAR ONES
2164  011454  105037  014555              CLRB    TENS            ;CLEAR TENS
2165  011460  105037  014553              CLRB    HUNS            ;CLEAR HUNS
2166  011464  005702                TESTR2: TST     R2                      ;CONVERT VALUE TO A DECIMAL VALUE
2167  011466  001424                        BEQ     TYPOUT
2168  011470  005302                        DEC     R2
2169  011472  105237  014556              INCB    ONES
2170  011476  123727  014556  000012        CMPB    ONES,#10.
2171  011504  001367                        BNE     TESTR2
2172  011506  105037  014556              CLRB    ONES
2173  011512  105237  014555              INCB    TENS
2174  011516  123727  014555  000012        CMPB    TENS,#10.
2175  011524  001357                        BNE     TESTR2
2176  011526  105037  014555              CLRB    TENS
2177  011532  105237  014553              INCB    HUNS
2178  011536  000752                        BR      TESTR2
2179  011540  152737  000060  014553  TYPOUT: BISB    #60,HUNS                ;PREPARE FOR TYPOUT
2180  011546  152737  000060  014555          BISB    #60,TENS
2181  011554  152737  000060  014556          BISB    #60,ONES
2182  011562  104400  014553              TYPE    ,HUNS           ;TYPE VALUE
2183  011566  000002                        RTI
2184  011570  000000                VO:     0                       ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
2185  011572  000002                V2:     2
2186  011574  000004                V4:     4
2187  011576  000012                V12:    12
2188  011600  000062                V500:   50.
2189  011602  000326                V326:   326
2190
2191  011604  000041                VNR:    33.             ;.33 LSB,NORMAL LIMITS FOR SYSTEM
2192  011606  000310                VNP:    200.            ;2 LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
2193  011610  000144                VSET:   100.            ;1 LSB
2194  011612  000175                VLIN:   125.            ;1.25 LSB
2195  011614  100000                        BITI5
2196
2197  011616  052777  000100  167320  AGATST: BIS     #100,@$TKS
2198  011624  000137                        JMP     @(PC)+
2199  011626  001644                AGTST:  BEGIN
```

```
                                .SBTTL  END OF PASS ROUTINE
                                ;;*****************************************************************
                                ;#INCREMENT THE PASS NUMBER ($PASS)
                                ;#IF THERES A MONITOR GO TO IT
                                ;#IF THERE ISN'T JUMP TO AGATST
                                $EOP:
011630  000240                          NOP
011632  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
011636  005037  001160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
011642  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
011646  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011654  005327                          DEC     (PC)+           ;;LOOP?
011656  000001          $EOPCT: .WORD   1
011660  003035                          BGT     $DOAGN          ;;YES
011662  012737                          MOV     (PC)+,2(PC)+    ;;RESTORE COUNTER
011664  000001          $ENDCT: .WORD   1
011666  011656                          $EOPCT
011670  104400  011676                  TYPE    ,65$            ;;TYPE ASCIZ STRING
011674  000414                          BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <15><12>/ENDPASS  GOOD UNITS  /
011726                  64$:
011726  013746  001436                  MOV     GUNITS,-(SP)    ;;SAVE GUNITS FOR TYPEOUT
011732  104404                          TYPBN                   ;;GO TYPE--BINARY ASCII
011734  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011740  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011742  000005                          RESET                   ;;CLEAR THE WORLD
011744  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011746  000240                          NOP                     ;;SAVE ROOM
011750  000240                          NOP                     ;;FOR
011752  000240                          NOP                     ;;ACT11
011754                          $DOAGN:
011754  000137                          JMP     @(PC)+          ;;RETURN
011756  011616                  $RTNAD: .WORD   AGATST
011760     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
011764                          .EVEN
```

```
                                                .SBTTL          ASCII MESSAGES
011764  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST/<15><12>
011772  020105  042524  052123
012000  005015      000
012003      015  051412  052105  SETMSG: .ASCIZ  <15><12>/SETTLING TEST/<15><12>
012010  046124  047111  020107
012016  042524  052123  005015
012024      000
012025      055      000        MINUS:  .BYTE   55,0
012027      077      000        QUEST:  .BYTE   77,0
012031      136      101   040  AMSG:   .BYTE   136,101,40,40,0
012034      040      000
012036      136      103   040  CMSG:   .BYTE   136,103,40,40,0
012041      040      000
012043      136      107   015  GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
012046      012      123   127
012051      122      105   107
012054      072      000
012056  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
012064      000
012065      055  020055      000  DASH:   .ASCIZ  /-- /
012071      123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
012076  026455  053440  042111
012104  044124  005015      000
012111      103  000110           CH:     .ASCIZ  /CH/
012114  020040  020040      000  SPACE:  .ASCIZ  /   /
012121      040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
012126  047117  041440  000110
012134  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
012142  047111  020107  051106
012150  046517  041440  000110
012156  040440  020124      000  ATMSG:  .ASCIZ  / AT /
012163      116  044517  042523  NOI:    .ASCIZ  /NOISE: /
012170  020072      000
012173      040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
012200  040510  047116  046105
012206  000040
012210  020040  020040  047504  DONE:   .ASCIZ  /    DONE/<15><12>
012216  042516  005015      000
012223      057      000        SLASH:  .ASCIZ  #/#
012225      040  020040  047440  OKMSG:  .ASCIZ  /    OK/<15><12>
012232  006513  000012
012236  005015  054524  042520  CCHAN:  .ASCIZ  <15><12>/TYPE CHANNEL & CR: /
012244  041440  040510  047116
012252  046105  023040  041440
012260  035122  000040
012264  005015  054524  042520  SEL:    .ASCIZ  <15><12>/TYPE "O" FOR OFFSET, "G" FOR GAIN & CR: /
012272  021040  021117  043040
012300  051117  047440  043106
012306  042523  026124  021040
012314  021107  043040  051117
012322  043440  044501  020116
012330  020046  051103  020072
012336      000
```

```
2291  012337  015  040412  045104  XADJ:   .ASCII  <15><12>/ADJUST R15/
2292  012344  051525  020124  030522
2293  012352  065
2294  012353  040  047506  020122  MOLSB:  .ASCIZ  / FOR 0.00 LSB ERROR/
2295  012360  027060  030060  046040
2296  012366  041123  042440  051122
2297  012374  051117  000
2298  012377  015  044412  050116  IGND:   .ASCII  <15><12>/INPUT A GROUND ON THE CHANNEL/
2299  012404  052125  040440  043440
2300  012412  047522  047125  020104
2301  012420  047117  052040  042510
2302  012426  041440  040510  047116
2303  012434  046105
2304  012436  005015  054524  042520  CRWR:   .ASCIZ  <15><12>/TYPE CR WHEN READY/<15><12>
2305  012444  041440  020122  044127
2306  012452  047105  051040  040505
2307  012460  054504  005015  000
2308  012465  015  044412  050116  IVOLT:  .ASCIZ  <15><12>/INPUT +5.115 VOLTS ON THE CHANNEL/
2309  012472  052125  025440  027065
2310  012500  030461  020065  047526
2311  012506  052114  020123  047117
2312  012514  052040  042510  041440
2313  012522  040510  047116  046105
2314  012530  000
2315  012531  015  040412  045104  YADJ:   .ASCIZ  <15><12>/ADJUST R3/
2316  012536  051525  020124  031522
2317  012544  000
2318  012545  053  000  POSITV: .ASCIZ  /+/
2319  012547  040  025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
2320  012554  047522  025122  006452
2321  012562  000012
2322  012564  051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
2323  012572  042105  051440  040524
2324  012600  042524  051450  000051
2325  012606  047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
2326  012614  020127  036050  030440
2327  012622  031057  046040  041123
2328  012630  020051  052123  052101
2329  012636  024105  024523  005015
2330  012644  000
2331  012645  040  044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
2332  012652  024040  020076  020061
2333  012660  027461  020062  051514
2334  012666  024502  051440  040524
2335  012674  042524  051450  006451
2336  012702  000012
2337  012704  051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
2338  012712  051450  020051  044527
2339  012720  042504  020122  044124
2340  012726  047101  031040  046040
2341  012734  041123  000
```

```
2342  012737  000040  052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
2343  012744  026505  044527  052104
2344  012752  024110  024523  047440
2345  012760  052125  044523  042504
2346  012766  025440  047440  020122
2347  012774  020055  027461  020062
2348  013002  051514  000102
2349  013006  046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
2350  013014  051515  020054  000
2351  013021  000040  051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
2352  013026  042520  045501  040440
2353  013034  020124  000
2354  013037  000015  042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
2355  013044  047440  020106  047514
2356  013052  044507  020103  042524
2357  013060  052123  000123
2358  013063  000040  047117  040440  ONAD:   .ASCIZ  / ON ADV11 AT /
2359  013070  053104  030461  040440
2360  013076  020124  000
2361  013101  000040  042101  030526  MSG50:  .ASCIZ  / ADV11'S FOUND/<15><12>
2362  013106  023461  020123  047506
2363  013114  047125  006504  000012
2364  013122  005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><12><1
2365  013130  020062  051514  006502
2366  013136  005012  005012  005012
2367  013144  005012  005012  005012
2368  013152  030455  031057  051514          .ASCIZ  \-1/2LSB\
2369  013160  000102
2370
2371                                          .EVEN
2372  013162  044504  043106  051105  MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
2373  013170  047105  044524  046101
2374  013176  046040  047111  040505
2375  013204  044522  054524  006472
2376  013212  000012
```

```
2377  013214  020040  020040  020040  MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
2378  013222  020040  020040  020040
2379  013230  020040  020040  020040
2380  013236  020040  052123  052101
2381  013244  026505  044527  052104
2382  013252  020110  044504  052123
2383  013260  044522  052502  044524
2384  013266  047117  005015  005012
2385  013274  020040  020043  043117          .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><12><
2386  013302  051440  040524  042524
2387  013310  005123  005012  005012
2388  013316  005012  005012  005012
2389  013324  005012  005012  005012
2390  013332  005012
2391  013334  020040  020040  020040          .ASCII  /                                                STATE WIDTH (LSB)/<15>
2392  013342  020040  020040  020040
2393  013350  020040  020040  020040
2394  013356  020040  020040  020040
2395  013364  020040  020040  020040
2396  013372  020040  020040  020040
2397  013400  020040  020040  020040
2398  013406  020040  020040  020040
2399  013414  051440  040524  042524
2400  013422  053440  042111  044124
2401  013430  024040  051514  024502
2402  013436  005015
2403  013440  030040  020040  020040          .ASCIZ  # 0          1/2          1          1 1/2          2#
2404  013446  020040  020040  020040
2405  013454  020040  020040  027461
2406  013462  020062  020040  020040
2407  013470  020040  020040  020040
2408  013476  020040  020061  020040
2409  013504  020040  020040  020040
2410  013512  020040  030440  030440
2411  013520  031057  020040  020040
2412  013526  020040  020040  020040
2413  013534  020040  031040      000
2414  013541      015  052012  050131  MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR TEST: /
2415  013546  020105  042514  052124
2416  013554  051105  023040  041440
2417  013562  020122  047506  020122
2418  013570  042524  052123  020072
2419  013576      000
2420  013577      122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
2421  013604  053111  020105  041501
2422  013612  052503  040522  054503
2423  013620  006472  000012
2424  013624  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
2425  013632  054101  046511  046525
2426  013640  040440  020124      000
2427  013645      015  050012  044522  HEADS:  .ASCII  <15><12>/PRINT VALUES--/
2428  013652  052116  053040  046101
2429  013660  042525  026523      055
```

```
2430  013665  040     042523  020124  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
2431  013672  044103  047101  042516
2432  013700  020114  047111  051440
2433  013706  051127  046040  053517
2434  013714  041040  052131  006505
2435  013722  000012
2436  013724  055033  000             C0:     .ASCIZ  <33><132>
2437  013727  033     015462  000110  C2:     .ASCIZ  <33><62><33><110>       ;CLEAR GRAPH MODE AND HOME
2438  013734  000112                  C3:     .ASCIZ  <112>
2439  013736  005015  043117  051506  MOFSET: .ASCIZ  <15><12>/OFFSET =/
2440  013744  052105  036440  000
2441  013751  040     051514  020102  MLSB:   .ASCIZ  / LSB /
2442  013756  000
2443  013757  040     052101  000040  MAT:    .ASCIZ  / AT /
2444  013764  005015  042440  052116  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
2445  013772  051105  047111  020107
2446  014000  042524  052123  000040
2447  014006  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
2448  014011  061     111     062
2449  014014  114     041     060
2450  014017  045     063     051
2451  014022  066     055     071
2452  014025  061     074     110
2453  014030  041     040     112
2454  014033  000
2455  014034  033     061     101     BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
2456  014037  047     111     061
2457  014042  104     050     065
2458  014045  044     062     110
2459  014050  040     040     102
2460  014053  000
2461  014054  033     110     033     VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0      ;HOME & ERASE SCREEN & CLEAR GRA
2462  014057  112     033     061
2463  014062  101     040     033
2464  014065  062     000
```

```
2465  014067  015     005012  042115  HEAD1:  .ASCII  <15><12><12>/MD-11-DVADA-A    ADV11 DIAGNOSTIC/<15><12>
2466  014074  030455  026461  053104
2467  014102  042101  026501  020101
2468  014110  020040  040440  053104
2469  014116  030461  042040  040511
2470  014124  047107  051517  044524
2471  014132  006503  012
2472  014135  012     035101  040440          .ASCII  <12>/A: AUTO TEST/
2473  014142  052125  020117  042524
2474  014150  052123
2475  014152  005015  035103  041440          .ASCII  <15><12>/C: CALIBRATION/
2476  014160  046101  041111  040522
2477  014166  044524  047117
2478  014172  005015  035120  050040          .ASCII  <15><12>/P: PRINT VALUES/
2479  014200  044522  052116  053040
2480  014206  046101  042525  123
2481  014213  015     046012  020072          .ASCII  <15><12>/L: LOGIC TEST/
2482  014220  047514  044507  020103
2483  014226  042524  052123
2484  014232  005015  035127  053440          .ASCIZ  <15><12>/W: WRAPAROUND TEST/<15><12>
2485  014240  040522  040520  047522
2486  014246  047125  020104  042524
2487  014254  052123  005015  000
2488  014261  123     040524  052524  EM1:    .ASCIZ  /STATUS REG. ERROR/
2489  014266  020123  042522  027107
2490  014274  042440  051122  051117
2491  014302  000
2492  014303  106     044501  042514  EM2:    .ASCIZ  /FAILED TO INTERRUPT/
2493  014310  020104  047524  044440
2494  014316  052116  051105  052522
2495  014324  052120  000
2496  014327  125     042516  050130  EM3:    .ASCIZ  /UNEXPECTED INTERRUPT/
2497  014334  041505  042524  020104
2498  014342  047111  042524  051122
2499  014350  050125  000124
2500  014354  051105  047522  020122  EM4:    .ASCIZ  #ERROR ON A/D CHANNEL#
2501  014362  047117  040440  042057
2502  014370  041440  040510  047116
2503  014376  046105  000
```

```
2504  014401  000105  051122  041520  DH1:    .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/
2505  014406  051440  051124  043505
2506  014414  042440  050130  041505
2507  014422  042524  020104  041501
2508  014430  052524  046101  000000
2509  014436  000105  051122  041520  DH2:    .ASCIZ  /ERRPC  STREG   CHANNEL  NOMINAL  TOLERANCE  ACTUAL/
2510  014444  020040  052123  042522
2511  014450  020107  020040  044103
2512  014456  047101  042516  020114
2513  014464  047040  046517  047111
2514  014472  046101  020040  047524
2515  014500  042514  040522  041516
2516  014506  020105  040440  052103
2517  014514  040525  000114
2518  014520  051105  050122  020103  DH3:    .ASCIZ  /ERRPC      STREG    ACTUAL/
2519  014526  020040  020040  051440
2520  014534  051124  043505  020040
2521  014542  020040  041501  052524
2522  014550  046101  000000
2523  014553  000000                  MUNS:   .BYTE   0
2524  014554  000056                  DECPNT: .BYTE   56
2525  014555  000000                  TENS:   .BYTE   0
2526  014556  000000  000000          ONES:   .BYTE   0,0
2527                                  .EVEN
2528
2529  014560  001116  001316  001124  DT1:    $ERRPC, STREG, $GDDAT, $BDDAT,0
2530  014566  001126  000000
2531  014572  001116  001316  001372  DT2:    $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
2532  014600  001124  001412  001126
2533  014606  000000
2534  014610  001116  001316  001126  DT3:    $ERRPC,STREG,$BDDAT,0
2535  014616  000000
2536  014620  000000                  DF1:    0
```

```
                                .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
                                ;
014622  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
014624  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
014632  105777  164306          1$:     TSTB    @$TKS           ;;WAIT FOR
014636  100375                          BPL     1$              ;;A CHARACTER
014640  117766  164302  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
014646  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
014654  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
014662  001013                          BNE     3$              ;;BRANCH IF NO
014664  105777  164254          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
014670  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
014672  117746  164250                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
014676  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
015002  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
015006  001366                          BNE     2$              ;;IF NOT DISCARD IT
015010  000750                          BR      1$              ;;YES, RESUME
015012  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
015020  002407                          BLT     4$              ;;BRANCH IF YES
015022  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
015030  003003                          BGT     4$              ;;BRANCH IF YES
015032  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
015040  000002                  4$:     RTI                     ;;GO BACK TO USER
                                ;;*********************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

015042  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
015044  012703  015050          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
015050  022703  015060          2$:     CMP     #$TTYIN+80.,R3  ;;BUFFER FULL?
015054  101405                          BLOS    4$              ;;BR IF YES
015056  104405                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
015060  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
015062  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
015066  001003                          BNE     3$              ;;SKIP IF NOT
015070  104400  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
015074  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
015076  111337  015046          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
015102  104400  015046                  TYPE    ,9$
```

```
 2593  015006  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 2594  015012  001356                            BNE     2$              ;;LOOP IF NOT RETURN
 2595  015014  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 2596  015020  104400  001172                    TYPE    $LF             ;;TYPE A LINE FEED
 2597  015024  012603                            MOV     (SP)+,R3        ;;RESTORE R3
 2598  015026  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 2599  015030  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 2600  015036  012766  015050  000004            MOV     #STTYIN,4(SP)
 2601  015044  000002                            RTI                     ;;RETURN
 2602  015046     000                  9$:       .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 2603  015047     000                            .BYTE   0               ;;TERMINATOR
 2604  015050  000010                  STTYIN:   .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
 2605  015060  052536  005015     000  $CNTLU:   .ASCIZ  /↑U/<15><12>   ;;CONTROL "U"
 2606  015065     136  006507  000012  $CNTLG:   .ASCIZ  /↑G/<15><12>   ;;CONTROL "G"
 2607  015072  005015  053523  020122  $MSWR:    .ASCIZ  <15><12>/SWR = /
 2608  015100  020075     000
 2609  015103     040  047040  053505  $MNEW:    .ASCIZ  /  NEW = /
 2610  015110  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

015114  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
015116  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
015124  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015126  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015130  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015132  104406                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
015134  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
015136  005001                          CLR     R1              ;;CLEAR DATA WORD
015140  005002                          CLR     R2
015142  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
015144  001412                          BEQ     3$              ;;IF ZERO GET OUT
015146  006301                          ASL     R1              ;;*2
015150  006102                          ROL     R2
015152  006301                          ASL     R1              ;;*4
015154  006102                          ROL     R2
015156  006301                          ASL     R1              ;;*8
015160  006102                          ROL     R2
015162  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
015166  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
015170  000764                          BR      2$              ;;LOOP
015172  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
015174  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
015200  010237  015214                  MOV     R2,$HIOCT
015204  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
015206  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
015210  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
015212  000002                          RTI                     ;;RETURN
015214  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

015216                          $SCOPE:
015216  032777  040000  163714  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
015224  001114                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
015226  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
015230  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
015234  012737  015254  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
015242  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
015246  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015252  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
015254  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
015256  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015262  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
015264                          6$:;#####END OF CODE FOR THE XOR TESTER#####
015264  032777  000400  163646          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
015272  001404                          BEQ     2$              ;;BR IF NO
015274  127737  163640  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
015302  001465                          BEQ     $OVER           ;;BR IF YES
015304  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
015310  001421                          BEQ     3$              ;;BR IF NO
015312  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015320  101015                          BHI     3$              ;;BR IF NO
015322  032777  001000  163610          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
015330  001404                          BEQ     4$              ;;BR IF NO
015332  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
015340  000446                          BR      $OVER
015342  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
015346  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015352  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
015354  032777  004000  163556  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
015362  001011                          BNE     1$              ;;BR IF YES
015364  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
015370  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
015372  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
015376  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015404  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
015406  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
015414  013737  015472  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
015422  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
015426  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
015434  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

```
 2705  015440  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 2706  015444  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 2707  015450  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 2708  015456  013777  001102  163456  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 2709  015464  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 2710  015470  000002                          RTI                     ;;FIXES PS
 2711  015472  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
 2712                                  .SBTTL  ERROR HANDLER ROUTINE
 2713
 2714                                  ;;**************************************************************
 2715                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 2716                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 2717                                  ;*AND GO TO $ERRTYP ON ERROR
 2718                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 2719                                  ;*SW15=1         HALT ON ERROR
 2720                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
 2721                                  ;*SW10=1         BELL ON ERROR
 2722                                  ;*SW09=1         LOOP ON ERROR
 2723                                  ;*CALL
 2724                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 2725
 2726  015474                          $ERROR:
 2727  015474  043737  001440  001436          BIC     TSTBIT,GUNITS
 2728  015502  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 2729  015506  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 2730  015510  013777  001102  163424          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 2731  015516  032777  002000  163414          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
 2732  015524  001402                          BEQ     1$              ;;NO - SKIP
 2733  015526  104400  001164                  TYPE    ,$BELL          ;;RING BELL
 2734  015532  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 2735  015536  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 2736  015542  162737  000002  001116          SUB     #2,$ERRPC
 2737  015550  117737  163342  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 2738  015556  032777  020000  163354          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
 2739  015564  001004                          BNE     20$             ;;SKIP TYPEOUTS
 2740  015566  004737  015676                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
 2741  015572  104400  001171                  TYPE    ,$CRLF
 2742  015576                          20$:
 2743  015576  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 2744  015604  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
 2745  015606  113737  001114  015620          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
 2746  015614  004737  016332                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
 2747  015620     000                  21$:    .BYTE   0
 2748  015621     000                          .BYTE   0
 2749  015622  000777                  22$:    BR      22$             ;;APT ERROR LOOP
 2750  015624  005777  163310          2$:     TST     @SWR            ;;HALT ON ERROR
 2751  015630  100001                          BPL     3$              ;;SKIP IF CONTINUE
 2752  015632  000000                          HALT                    ;;HALT ON ERROR!
 2753  015634  032777  001000  163276  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
 2754  015642  001402                          BEQ     4$              ;;BR IF NO
 2755  015644  013716  CC::10                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 2756  015650  005737  001162          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
 2757  015654  001402                          BEQ     5$              ;;BR IF NONE
 2758  015656  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 2759  015662                          5$:
 2760  015662  022737  011744  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
```

```
                                        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" (SERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

                                SERRTYP:
                                        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$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
                                3$:     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                        MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
                                        BEQ     5$              ;;SKIP TYPEOUT IF 0
                                        TYPE                    ;;TYPE THE "DATA HEADER"
                                4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
                                5$:     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                        MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
                                        BNE     7$              ;;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
                                ;*

016032  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
016036  100002                          BPL     1$              ;;BR IF YES
016040  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
016042  000430                          BR      3$              ;;LEAVE
016044  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
016046  017600  000002                  MOV     2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
016052  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
016060  001011                          BNE     6$              ;;NO,GO CHECK FOR APT CONSOLE
016062  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
016070  001405                          BEQ     6$              ;;NO,GO CHECK FOR CONSOLE
016072  010037  016102                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
016076  004737  016322                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
016102  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
016104  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
016112  001003                          BNE     6O$             ;;YES,SKIP TYPE OUT
016114  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016116  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
016120  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
016122  012600                  6O$:    MOV     (SP)+,R0        ;;RESTORE R0
016124  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
016130  000002                          RTI                     ;;RETURN
016132  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
016136  001430                          BEQ     8$
016140  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
016144  001006                          BNE     5$
016146  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
016150  104400                          TYPE                    ;;TYPE A CR AND LF
016152  001171                          $CRLF
016154  105037  016310                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
016160  000755                          BR      2$              ;;GET NEXT CHARACTER
016162  004737  016244          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
016166  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
016172  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
016174  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
016200  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
016204  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
016206  004737  016244                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
016212  105337  016310                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
2868  016216  000770                          BR      7$              ;;LOOP
2869
2870                                  ;HORIZONTAL TAB PROCESSOR
2871
2872  016220  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2873  016224  004737  016244          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2874  016230  132737  000007  016310          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2875  016236  001372                          BNE     9$              ;;TAB STOP
2876  016240  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2877  016242  000724                          BR      2$              ;;GET NEXT CHARACTER
2878  016244  105777  162700          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2879  016250  100375                          BPL     $TYPEC
2880  016252  116677  000002  162672        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2881  016260  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2882  016266  001003                          BNE     1$              ;;BRANCH IF NO
2883  016270  105037  016310                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2884  016274  000406                          BR      $TYPEX          ;;EXIT
2885  016276  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2886  016304  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2887  016306  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2888  016310  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2889  016312  000207          $TYPEX: RTS     PC
2890
2891                                  .SBTTL  APT COMMUNICATIONS ROUTINE
2892
2893                                  ;;**************************************************************
2894  016314  112737  000001  016560  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2895  016322  112737  000001  016556  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2896  016330  000403                          BR      $ATYC
2897  016332  112737  000001  016560  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2898  016340                          $ATYC:
2899  016340  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2900  016342  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2901  016344  105737  016556                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2902  016350  001450                          BEQ     5$              ;;IF NOT:  BR
2903  016352  122737  000001  001214        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2904  016360  001031                          BNE     3$              ;;IF NOT:  BR
2905  016362  132737  000100  001215        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2906  016370  001425                          BEQ     3$              ;;IF NOT:  BR
2907  016372  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2908  016376  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2909  016404  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2910  016410  001375                          BNE     1$              ;;IF NOT:  WAIT
2911  016412  010037  001210                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2912  016416  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2913  016420  001376                          BNE     2$
2914  016422  163700  001210                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2915  016426  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2916  016430  010037  001212                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2917  016434  012737  000004  001174        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2918  016442  000413                          BR      5$
2919  016444  017637  000004  016470  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2920  016452  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2921  016460  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2922  016464  004737  016032                JSR     PC,$TYPE        ;;CALL TYPE MACRO
2923  016470  000000          4$:     .WORD   0
```

```
 2924  016472                          5$:
 2925  016472  105737  016560          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
 2926  016476  001416                          BEQ     12$             ;;IF NOT:  BR
 2927  016500  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
 2928  016504  001413                          BEQ     12$             ;;IF NOT:  BR
 2929  016506  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
 2930  016512  001375                          BNE     11$             ;;IF NOT:  WAIT
 2931  016514  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
 2932  016522  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 2933  016530  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
 2934  016534  105037  016560          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
 2935  016540  105037  016557                  CLRB    $LFLG           ;;CLEAR LOG FLAG
 2936  016544  105037  016556                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
 2937  016550  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 2938  016552  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 2939  016554  000207                          RTS     PC              ;;RETURN
 2940  016556     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
 2941  016557     000          $LFLG:  .BYTE   0               ;;LOG FLAG
 2942  016560     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
 2943          016562                  .EVEN
 2944          000200          APTSIZE=200
 2945          000001          APTENV=001
 2946          000100          APTSPOOL=100
 2947          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
                                
016562 017646 000000            $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016566 116637 000001 017005             MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
016574 112637 017007                    MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
016600 062716 000002                    ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
016604 000406                           BR      $TYPON
016606 112737 000001 017005     $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
016614 112737 000006 017007             MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
016622 112737 000005 017004     $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
016630 010346                           MOV     R3,-(SP)        ;;SAVE R3
016632 010446                           MOV     R4,-(SP)        ;;SAVE R4
016634 010546                           MOV     R5,-(SP)        ;;SAVE R5
016636 113704 017007                    MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
016642 005404                           NEG     R4
016644 062704 000006                    ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
016650 110437 017006                    MOVB    R4,$OMODE       ;;SAVE IT FOR USE
016654 113704 017005                    MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
016660 016605 000012                    MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
016664 005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
016666 006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
016670 000404                           BR      3$              ;;GO DO MSB
016672 006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
016674 006105                           ROL     R5
016676 006105                           ROL     R5
016700 010503                           MOV     R5,R3
016702 006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
016704 105337 017006                    DECB    $OMODE          ;;TYPE THIS DIGIT?
016710 100016                           BPL     7$              ;;BR IF NO
016712 042703 177770                    BIC     #177770,R3      ;;GET RID OF JUNK
016716 001002                           BNE     4$              ;;TEST FOR 0
016720 005704                           TST     R4              ;;SUPPRESS THIS 0?
016722 001403                           BEQ     5$              ;;BR IF YES
```

```
 3004  016724  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 3005  016726  052703  000060          5$:     BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 3006  016732  052703  000040                  BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 3007  016736  110337  017002                  MOVB    R3,8$           ;;SAVE FOR TYPING
 3008  016742  104400  017002                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
 3009  016746  105337  017004          7$:     DECB    $OCNT           ;;COUNT BY 1
 3010  016752  003347                          BGT     2$              ;;BR IF MORE TO DO
 3011  016754  002402                          BLT     6$              ;;BR IF DONE
 3012  016756  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 3013  016760  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 3014  016762  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
 3015  016764  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 3016  016766  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 3017  016770  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 3018  016776  012616                          MOV     (SP)+,(SP)
 3019  017000  000002                          RTI                     ;;RETURN
 3020  017002     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 3021  017003     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 3022  017004     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 3023  017005     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 3024  017006  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  BINARY TO ASCII AND TYPE ROUTINE

                                ;;*********************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
                                ;*BINARY-ASCII NUMBER AND TYPE IT.
                                ;*CALL:
                                ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
                                ;*      TYPBN                   ;;TYPE IT

017010  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
017012  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
017016  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
017020  112737  000060  017062  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
017026  006101                          ROL     R1              ;;GET THIS BIT
017030  001406                          BEQ     2$              ;;DONE?
017032  105537  017062                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
017036  104400  017062                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
017042  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
017044  000765                          BR      1$              ;;GO DO THE NEXT BIT
017046  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
017050  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
017056  012616                          MOV     (SP)+,(SP)
017060  000002                          RTI                     ;;RETURN TO USER
017062     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
```

```
                                .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.

017064  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
017066  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
017072  005740                          TST     -(R0)           ;;BACKUP BY 2
017074  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
017076  006300                          ASL     R0              ;;POSITION FOR INDEXING
017100  016000  017106                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
017104  000200                          RTS     R0              ;;GO TO ROUTINE

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
017106                          $TRPAD:
017106  016032                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
017110  016606                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
017112  016562                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
017114  016622                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
017116  017010                          $TYPBN  ;;CALL=TYPBN    TRAP+4(104404)  TYPE BINARY (ASCII) NUMBER

017120  014622                          $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
017122  014742                          $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
017124  015114                          $RDOCT  ;;CALL=RDOCT    TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
017126  004102                          TEST    ;;CALL=CHECK    TRAP+10(104410)
017130  004074                          TESTIT  ;;CALL=CHKIT    TRAP+11(104411)
017132  011424                          DECTYP  ;;CALL=TYPDC    TRAP+12(104412)
```

```
                                .EVEN
012134  000310                  DISI:   :BLKW   200.            ;STATE-WIDTH DISTRIBUTION
017754  010000                  BUFFER: :BLKW   4096.           ;BUFFER AREA

        000001                  .END
```

```
ABASE = 170400          553*   653     694     742     743     744
ACDW1 = 000000          653    696
ACDW2 = 000000          653
ACPUOP= 000000          653    668
ADBUFF  001322          744*  1044    1055    1101    1123    1136    1150    1183    1219    1234    1305*   1309*   1347
                       1349   1350    1352    1542    1599*   1611*   1613*   1659*   1688*   1701    1741*   1747    1753
                       1809   2075
ADDW0 = 000000          653
ADDW1 = 000000          653
ADDW10= 000000          653
ADDW11= 000000          653
ADDW12= 000000          653
ADDW13= 000000          653
ADDW14= 000000          653
ADDW15= 000000          653
ADDW2 = 000000          653
ADDW3 = 000000          653
ADDW4 = 000000          653
ADDW5 = 000000          653
ADDW6 = 000000          653
ADDW7 = 000000          653
ADDW8 = 000000          653
ADDW9 = 000000          653
ADEVCT= 000000          653    659
ADEVM = 000000          653    695
ADST1   001320          743*  1598*   1610*   1612*   1743*
AENV  = 000000          653    664
AENVM = 000000          653    665
AFATAL= 000000          653    656
AGAIN   007424         1792*  1830
AGATST  011616         2197*  2234
AGTST   011626         1566*  1582*   1591*   2199*
AJGAIN  005510         1479   1495*
AJOFF   005436         1477   1482*
AMADR1= 000000          653    681
AMADR2= 000000          653    685
AMADR3= 000000          653    688
AMADR4= 000000          653    691
AMAMS1= 000000          653    675
AMAMS2= 000000          653    683
AMAMS3= 000000          653    686
AMAMS4= 000000          653    689
AMSG    012031          803   2247*
AMSGAD= 000000          653    661
AMSGLG= 000000          653    662
AMSGTY= 000000          653    655
AMTYP1= 000000          653    676
AMTYP2= 000000          653    684
AMTYP3= 000000          653    687
AMTYP4= 000000          653    690
APASS = 000000          653    658
APRIOR= 000200          555*   653     746
APTCSU= 000040         2842   2947*
APTENV= 000001         2743   2835    2903    2945*
APTSIZ= 000200          869   2944*
APTSPO= 000100         2837   2905    2946*
```

```
AROUND  007564          1826    1829#
ASKCH   013665          2430#
ASWREG= 000000           653     666
ATESTN= 000000           653     657
ATMSG   012156          1656    2268#
AUNIT = 000000           653     660
AUSWR = 000000           653     667
AVECT1= 100400           554#    653     692     745     747     748     749
AVECT2= 000000           653     693
BASEBR  001326           746#    1616*   1617*
BASECH  001342           752#    1322    1323    1658    1786    2063
BEGIN   001644           570     828#    2199
BEGINA  006066           918    1570#
BEGINC  005364           921    1469#
BEGINL  002670           979#   1563    1572
BEGINP  005640           924    1519#
BEGINW  006150           930    1586#
BEGIN2  001652           572     830#
BEGL    006030           905     927    1561#
BEG2    002262           571     800     901#    1494    1517
BITPNT  001422           776#   1737*   1740    1760    1761*
BIT0  = 000001           526#   1052    1169    1208    1212    1225    1226    1227
BIT00 = 000001           516#    526
BIT01 = 000002           515#    525
BIT02 = 000004           514#    524
BIT03 = 000010           513#    523
BIT04 = 000020           512#    522
BIT05 = 000040           511#    521
BIT06 = 000100           510#    520
BIT07 = 000200           509#    519
BIT08 = 000400           508#    518    2678
BIT09 = 001000           507#    517    2686    2753
BIT1  = 000002           525#
BIT10 = 002000           506#   2731
BIT11 = 004000           505#   2693
BIT12 = 010000           504#    989    1044    1055    2020
BIT13 = 020000           503#   1527    1543    2738
BIT14 = 040000           502#    996    1195    1197    2664
BIT15 = 100000           501#   1063    1084    1098    1195    1197    1211    1224    2195
BIT2  = 000004           524#   1027
BIT3  = 000010           523#   1034    1052
BIT4  = 000020           522#   1020
BIT5  = 000040           521#   1012
BIT6  = 000100           520#   1004    1169    1173    1537
BIT7  = 000200           519#   1095    1173    1211
BIT8  = 000400           518#    985
BIT9  = 001000           517#   1093    1213
BPTVEC= 000014           533#
BUFFER  017754          1769    1814*   1816*   1832    1860    1928    1940    1942    1970    1981    1982    1983    2005
                        2008    3088#
BUFF1   014006          1912    2447#
BUFF2   014034          2004    2455#
BUMPAD  006206          1564    1580    1589    1594#
BYPASS  006456          1606    1627#
B10     011354          2140#   2141
B11     011410          2150#   2151
```

```
CCHAN   012236          1469    2279#
CH      012111          1529    2261#
CHAN    012173          2047    2271#
CHANL   001372           764#   1131*   1145*   1322*   1371*   1372    1390*   1391    1393*   1394    1404*   1471*   1472
                        1689    1748    2048    2064*   2531
CHANNL  007370          1785#
CHECK = 104410          1087    1099    1112    1124    1216    3083#
CHKIT = 104411           986     997    1005    1013    1021    1028    1035    1064    3084#
CH1     001360           759#   1402*   1414    1416    1417*   1651    1657    1679
CH2     001362           760#   1403*   1404    1417    1418*   1643    1675
CLEAR1  007316          1771#   1773
CLEAR2  007362          1782#   1784
CMSG    012036           797    2249#
COMPAR  011130          1258    1269    1281    1293    1313    1327    1356    1361    1462    2085#
CONV    006766          1699#   1703
CONVR   007466          1804    1807#
CONVRT  011016          1256    1267    1279    1291    1306    1310    1660    2062#
CR      = 000015         441#   2881    2891
CRLF    = 000200         442#   2852    2891
CRWR    012436          1486    1496    1501    2304#
C0      013724           878    2436#
C2      013727          1921    2010    2437#
C3      013734          2007    2438#
DAC     001414           773#   1378    1381    1384    1387    1434    1438    1506    1510    1678    1682    1738*   1740*
                        1741    1760*
DASH    012065          1867    2257#
DDISP = 177570           448#    634     857
DECPNT  014554          1872*   1901*   2524#
DECTYP  011424          2156#   3085
DELAY   001416           774#   1453*   1457*
DELAY1  007530          1810    1818#
DELAY2  007536          1812    1819    1820#
DELAY3  007462          1805#   1806
DELCLR  010626          1910    1922    2011    2020#
DF1     014620           721     728     734     740    2536#
DH1     014401           719    2504#
DH2     014435           738    2509#
DH3     014520           726     732    2518#
DIFLIN  007266          1428    1765#
DISPLA  001142           634#    857*    865*   1545*   2708*   2730*
DISPRE  000174           567#    865
DIST    017134          1774    1843*   1913    3087#
DONE    012210          2120    2274#
DSWR    = 177570         447#    633     856
DT1     014560           720    2529#
DT2     014572           739    2531#
DT3     014610           727     733    2534#
DUMC    011250          1175    1182    1199    1202    2116#
DUMMY   001370           763#   1323*   1372*   1472*   1675*   1679*   1743
DUMW    011176          1163    1193    2098#
EDGE    001420           775#   1431*   1435*   1503*   1507*   1696*   1701*   1704*   1705*   1706*   1707*   1711    1753
EDGFLG  007072          1388*   1412*   1717    1727#
EMTVEC= 000030           536#    842*    843*
EM1     014261           718    2488#
EM2     014303           725    2492#
EM3     014327           731    2496#
```

```
EM4     014354            737    2500#
ER      011010           2054    2056    2059#
ERMSG   012547           1332    1671    1878    1893    1905    1965    2059    2319#
ERR     006644           1668    1671#
ERRVEC= 000004            529#    854     855*    866*    955     956*    976*    2669    2670*   2672*   2675*
FIRST   001352            756#   1780*   1855    1857*
FIXADR  006304           1595    1607#
FIXONE  006310            894    1608#
FLAG    001410            771#    873*    890*    1908    1968
GETDAT  010404           1972#   1980
GETEDG  006714           1373    1405    1688#
GMSG    012043            811    2251#
GNS   = ****** U          566    2221    3073    3074    3075    3076    3077    3080    3081    3082    3083    3084    3085
GUNITS  001436            782#    945*    960*    2223    2727*
HAFMSG  012737           1902    2342#
HALF    010070           1894    1896#
HEAD1   014067            892    2465#
HEADS   013645           1521    2427#
HT    = 000011            439#   2850    2891
HUNS    014553           2165*   2177*   2179*   2182    2523#
IGND    012377           1482    2298#
INRNGE  007626           1839    1842#
IOTVEC= 000020            534#    840*    841*
ISERV   001464            792#    896
IVOLT   012465           1495    2308#
KBVECT  001344            753#    895    1629
LAST    007742           1853    1870#
LEND    005156           1427    1429#
LESS    011342           2135    2137#
LF    = 000012            440#   2885    2891
LINEA   013624           1950    2424#
LOAD    010604           2006    2009    2013#
LOADY   011322           1916    1918    2016    2131#
LOADO   010610           2014#   2018
LO2     010602           1969    2012#
LSB     012121           1642    2263#
LSBMSG  012056           1869    2255#
MAT     013757           1648    2443#
MAX     001430            779#   1988*   1993    1995*   1999
MAXTST  010502           1991    1993#
MEND    013037           1573    2354#
MESP    013021           2045    2351#
MESR    013006           2042    2349#
METST   013764           2102    2444#
MIN     001424            777#   1987*   1990    1992*   1998
MINUS   012025           1720    2158    2245#
MLSB    013751           1445    2441#
MOFSET  013736           1325    2439#
MSG16   013214           1911    2377#
MSG18   013122           2003    2364#
MSG20   013162           1765    2372#
MSG21   013377           1927    2420#
MSG50   013101            973    2361#
MSG71   013541            908    2414#
MOLSB   012353           1500    2294#
NAR     010006           1879    1881#
```

```
NARMSG  012606          1883    2325#
NARROW  001350           755#   1779#   1847#   1881    1896
NBEXT   001364           761#    957#    959#    968     974#    975    1594    1605#   1625#
NEXT    007430          1793#   1825
NMBEXT  001366           762#    975#   1625
NOI     012163          2033    2269#
NOIMSG  011764          1370    2238#
NONE    001636           808     810    825#
NOTNAR  007662          1846    1852#
NOTNEW  010276          1937    1942#
NOTOK   007556          1822    1827#
NOVTSS  002212           881     884    889     892#    939
NXTCMP  010466          1989#   1997
NXTCVT  007154          1743#   1757
NXTSTA  010240          1929#   1943
NXTY1   010174          1915#   1920
NXTB    010450          1985#   2002
OFFERR  004464          1330    1332#
OFFOK   004472          1331    1334#
OFFSET  005160          1324    1431#   1488
OKAY    007546          1815    1817    1823#   1828
OKAYD   011450          2161    2163#
OKMSG   012225          1334    1669    1880    1895    1907    1967    2057    2277#
ONAD    013063          2108    2358#
ONES    014556          2163#   2169#   2170    2172#   2181#   2526#
OUT     001434           781#   1777#   1840#   1885    1888    1891    1898
OUTMSG  012704          1890    2337#
PC      =000007          460#    799#    805#    879#    882#    885#    894#    942#    978#   1163#   1175#   1182#   1193#
                        1199#   1202#   1237#   1324#   1326#   1373#   1389#   1405#   1407#   1408#   1413#   1428#   1429#
                        1440#   1446#   1468#   1488#   1489#   1490#   1512#   1513#   1562#   1563#   1564#   1571#   1572#
                        1579#   1580#   1587#   1588#   1589#   1637#   1649#   1670#   1672#   1683#   1708#   1726#   1910#
                        1916#   1918#   1922#   2006#   2009#   2011#   2012#   2016#   2019#   2031#   2046#   2058#   2060#
                        2114#   2121#   2128#   2153#   2198    2213#   2216#   2228#   2233    2740#   2746#   2802#   2840#
                        2859#   2866#   2873#   2887#   2889#   2922#   2939#
PEAK    001406           770#   1375#   1384#   1387#   2037    2039#   2043    2055
PERCNT  001432           780#   1732#   1733#   1734#   1735#   1736#   1758
PIRQ    =177772          446#
PIRQVE= 000240           540#
PLUS    010260          1934    1936#
PLUSR2  011330          2132    2134#
POPRO   001616           820#    826
POS     011436          2157    2160#
POSITV  012545          1443    2318#
POSPEA  010716          2038    2040#
POSRMS  010704          2035    2037#
PR0     =000000          463#
PR1     =000040          464#
PR2     =000100          465#
PR3     =000140          466#
PR4     =000200          467#
PR5     =000240          468#
PR6     =000300          469#
PR7     =000340          470#
PS      =177776          443#    444
PSW     =177776          444#
PWRVEC= 000024           535#
```

```
QUEST   012027            825     931    2246#
RBEG    001660            829     831#
RDCHR = 104405           2585    3080#
RDLIN = 104406            909    1474    1483    1497    2626    3081#
RDOCT = 104407            818    1470    3082#
READ    007600           1833#   1871
RELACC  010224           1909    1925#
REST1   002216            875     893#
RESVEC= 000010            530#
RET     007070           1718    1726#
RETERR  004124           1244    1246#
RETURN  001620            821#   1535    1698    1744    1791    2071
RMS     001404            769#   1374*   1378*   1381*   2034    2036*   2040    2053
RNA     001376            766#    898*   1766
RNB     001400            767#    899*   1767
RNC     001402            768#    900*   1768
RST     011300            799     805    2123#
R0    =%000000            451#    792     793*    794*    795     801     807     820*    880     883     886     888     895*
                          896*    897*    914*    915*    916     919     922     925     928     933*    936*    940*    941*
                         1093*   1096*   1101*   1123*   1162*   1192*   1213*   1214*   1219*   1234*   1475*   1476    1478
                         1525*   1526*   1530    1536*   1537*   1538    1542*   1545    1548    1627*   1631    1633*   1634
                         1635    1638    1639    1689*   1690*   1691*   1692    1693*   1694*   1697*   1702*   1739*   1755*
                         1758    1769*   1771*   1774*   1782*   1785*   1786*   1787*   1788*   1789    1790*   1829*   1831*
                         1870*   1899*   1903    1913*   1915    1928*   1929    1932*   1939    1942    1970*   1972    1978*
                         1981*   1986    1989    2005*   2008*   2014    2015    2024*   2026*   2062*   2063*   2064    2065*
                         2067    2068*   2069*   2072*   2076*   2089*   2090*   2092*   2093    2103    2225*   2228    2623
                         2627*   2630    2646*   2774    2775*   2776*   2783*   2784*   2785*   2786*   2787*   2788    2793
                         2798*   2800*   2804    2806    2833    2834*   2839    2844    2847*   2899    2907*   2911    2912
                         2914*   2915*   2916    2938*   3057    3058*   3059    3060*   3061*   3062*   3063*
R1    =%000001            452#   1553*   1554*   1628*   1629    1631*   1632*   1633    1638    1639    1742*   1756*   1770*
                         1772*   1775*   1783*   1792*   1820*   1824*   1832*   1833    1859    1914*   1919*   1925*   1931*
                         1932    1933    1971*   1979*   1984*   2001*   2013*   2017*   2025*   2028*   2088*   2090    2624
                         2628*   2632*   2634*   2636*   2639*   2642    2645*   2900    2937*   3034    3035*   3038*   3044*
R2    =%000002            453#   1343*   1353*   1406*   1410*   1411    1416*   1418    1441*   1539*   1556*   1678*   1682*
                         1766*   1793*   1794*   1795*   1796    1799    1802    1803*   1805*   1809*   1811    1813*   1814*
                         1816*   1818    1823*   1833*   1834*   1835*   1836*   1837*   1838    1842*   1843*   1844*   1845
                         1848    1852    1873*   1881*   1884*   1885*   1888*   1896*   1897*   1898*   1899    1915*   1917*
                         1929*   1930*   1931    1948*   1972*   1973*   1974*   1975*   1976*   1977*   1978    1985*   1996*
                         2014*   2040*   2043*   2131    2133*   2134    2136*   2137    2138*   2139*   2142    2156    2159*
                         2160    2162*   2166    2168*   2625    2629*   2633*   2635*   2637*   2643    2644*
R3    =%000003            454#   1340*   1350    1352*   1355    1447*   1449*   1487*   1502*   1711*   1712    1719*   1721
                         1776*   1821*   1859*   1860*   1861*   1862    1926*   1936    1938*   1944*   1945*   1946*   1947*
                         1948    1963    1986*   1987    1988    1989*   1990    1992    1993    1995    2137*   2143*   2144*
                         2145*   2146*   2147*   2148*   2149*   2152    2581    2582*   2583    2586*   2587    2591    2593
                         2595*   2597*   2981    2990*   2996*   2997*   3000*   3005*   3006*   3007    3016*
R4    =%000004            455#   1308*   1312    1341*   1347    1349*   1360    1411*   1667    1747*   1748*   1749*   1750*
                         1751    1767*   1793    1796*   1797*   1798*   1800    1933*   1935*   1936    1938    1982*   1998*
                         2000    2982    2984*   2985*   2986*   2987    2988*   3002    3004*   3012*   3015*
R5    =%000005            456#   1256*   1258*   1267*   1269*   1279*   1281*   1291*   1293*   1306*   1310*   1313*   1327*
                         1356*   1361*   1376*   1379*   1382*   1385*   1432*   1436*   1462*   1504*   1508*   1660*   1676*
                         1680*   1732    1763*   1768*   1794    1797    1799*   1800*   1801*   1939*   1940*   1941*   1951
                         1957*   1958    1983*   1999*   2000    2062    2082*   2085    2086    2095    2096*   2983    2989*
                         2991*   2993*   2994*   2995*   2996    3014#
R6    =%000006            457#    459     834*    835*    836
R7    =%000007            458#    460
SARSUB  007074           1376    1379    1382    1385    1432    1436    1504    1508    1676    1680    1732#
```

```
SAR1    007120          1737#
SEL     012264          1473    2283#
SETCH   012134          1650    2265#
SETMSG  012003          1401    2241#
SETIA   006652          1407    1408    1675#
SHIFT   007256          1759    1761#
SKIPST  001354           757#   1781*   1850*   1873    1876
SKPMSG  012564          1875    2322#
SLASH   012223           817    1956    2276#
SP    =x000006           459#    792*    798*    804*    812*    819     820     821     823*    838*    854*    862*    866
                         901*    911*    912*    914     938     955*    966     968*    976     1164*   1165*   1177    1179*
                        1180*   1201    1245*   1466*   1467*   1471    1475    1484    1498    1522*   1523*   1530*   1548*
                        1574*   1607*   1643*   1651*   1662*   1712*   1721*   1802*   1823    1862*   1951*   1958*   2048*
                        2103*   2109*   2125*   2126*   2223*   2553*   2554*   2557*   2558*   2559    2563*   2564*   2565
                        2568    2570    2572*   2581*   2586    2597    2598*   2599*   2600*   2621*   2622*   2623*   2624*
                        2625*   2627    2630*   2638*   2639    2641    2642*   2644    2645    2646    2669*   2672    2674
                        2675    2704    2705    2709*   2735    2755*   2758*   2774*   2779*   2800    2804*   2833*   2834
                        2844*   2846    2847    2848*   2850    2852    2854    2860    2862*   2864*   2872*   2876    2880
                        2881    2885    2899*   2900*   2907    2908*   2919    2920*   2921*   2931    2932*   2937    2938
                        2973*   2974    2975    2976*   2981*   2982*   2983*   2989    3014    3015    3016    3017*   3018*
                        3034*   3035    3044    3045*   3046*   3057*   3058
SPACE   012114          1452    1547    2262#
SPREAD  001412           772#   1132*   1146*   2086*   2093    2531
STACK = 001100           434#    798     804     838     901
STATE   012071          1858    2258#
STKLMT= 177774           445#
STREG   001316           742#   1041*   1042    1052*   1053    1072*   1075    1084*   1094*   1098*   1108*   1109    1120*
                        1121    1133*   1134    1147*   1148    1169*   1170    1172    1195*   1196    1203*   1208*   1209
                        1212*   1225*   1226*   1227*   1228    1235*   1241*   1242    1342*   1344*   1345    1520*   1538*
                        1540*   1574    1597*   1609*   1692*   1699*   1745*   1751*   1789*   1807*   2067*   2073*   2109
                        2529    2531    2534
SWDIST  010142          1906    1908#
SWR     001140           633#    809     812     819*    836     856*    858     864*    871*    1525    1527    1543    2020
                        2664    2678    2680    2686    2693    2731    2738    2750    2753
SWREG   000176           568#    864
SW0   = 000001           498#
SW00  = 000001           488#    498
SW01  = 000002           487#    497
SW02  = 000004           486#    496
SW03  = 000010           485#    495
SW04  = 000020           484#    494
SW05  = 000040           483#    493
SW06  = 000100           482#    492
SW07  = 000200           481#    491
SW08  = 000400           480#    490
SW09  = 001000           479#    489
SW1   = 000002           497#
SW10  = 002000           478#
SW11  = 004000           477#
SW12  = 010000           476#
SW13  = 020000           475#
SW14  = 040000           474#
SW15  = 100000           473#
SW2   = 000004           496#
SW3   = 000010           495#
SW4   = 000020           494#
```

DVADAA.CMB CROSS REFERENCE TABLE -- USER SYMBOLS

```
SW5    = 000040            493#
SW6    = 000100            492#
SW7    = 000200            491#
SW8    = 000400            490#
SW9    = 001000            489#
TADDR    001374            765#    806    1519*  1561*  1570*  1586*
TBITVE= 000014             531#
TCHK     005256           1447#   1490    1513
TEMP     001356            758#    947*    950*   953*   962*  1308   1312*  1355*  1360*  1434*  1438*  1439*  1441
                          1506*   1510*   1511*  1662   1821*  1827*  2066*  2075*  2078*  2079*  2080*  2081*  2087
TENS     014555           2164*   2173*   2174   2176*  2180*  2525#
TEST     004102           1242#   3083
TESTAD   002464            944#   1562    1571   1587
TESTIT   004074           1241#   3084
TESTR2   011464           2166#   2171    2175   2178
TKVEC  = 000060            538#
TOFF     005234           1326    1441#   1489   1512
TPVEC  = 000064            539#
TRAPVE= 000034             537#    844*    845*
TRTVEC= 000014             532#
TRY      007132           1739#   1762
TRYAG    002314            909#    932
TSTBIT   001440            783#    946*    960    961*   977*   1596*  1608*  2727
TST1     002670            983#    984
TST10    003050           1040#
TST11    003100           1045    1051#
TST12    003130           1056    1062#
TST13    003144           1069#
TST14    003210           1076    1082#
TST15    003242           1092#
TST16    003304           1105#
TST17    003346           1119#
TST2     002732            995#
TST20    003372           1129#   1200
TST21    003436           1137    1143#
TST22    003514           1152    1157#
TST23    003642           1176    1187#
TST24    003726           1207#
TST25    004000           1223#
TST26    004126           1252#   1254    1255
TST27    004176           1265#
TST3     002746           1002#
TST30    004226           1277#
TST31    004256           1288#
TST32    004330           1303#
TST33    004410           1320#
TST34    004476           1333    1338#
TST35    004624           1368#
TST36    005010           1399#
TST37    005134           1415    1424#
TST4     002770           1011#
TST5     003004           1019#
TST6     003020           1026#
TST7     003034           1033#
TST8     010514           1994    1996#
TYPBAD   007674           1841    1851    1855#
```

```
TYPBN = 104404          2224    3077#
TYPDC = 104412          1444    1641    1868    1874    1882    1886    1889    1900    1949    2041    2044    3085#
TYPE  = 104400           797     803     811     817     825     878     892     908     931     973    1325    1332    1334
                        1370    1401    1443    1445    1450    1452    1469    1473    1480    1482    1485    1486    1495
                        1496    1499    1500    1501    1521    1529    1547    1558    1573    1578    1642    1648    1650
                        1656    1669    1671    1720    1765    1858    1867    1869    1875    1878    1880    1883    1887
                        1890    1893    1895    1902    1905    1907    1911    1912    1921    1927    1950    1956    1965
                        1967    2003    2004    2007    2010    2030    2033    2042    2045    2047    2057    2059    2102
                        2108    2120    2158    2182    2219    2589    2592    2596    2733    2741    2773    2790    2792
                        2795    2797    2801    2808    2855    3008    3041    3073#
TYPEDG  007030          1649    1711#   2046
TYPOC = 104401          2781    2805    3074#
TYPON = 104403          3076#
TYPOS = 104402           814     970    1532    1550    1575    1645    1653    1664    1714    1723    1864    1953    1960
                        2050    2105    2111    3075#
TYPOUT  011540          2167    2179#
TYPRP   010666          1389    2033#
TYPSET  006522          1413    1641#
UNEXP   001442           785#    1003    1178
VADR    001336           750#     964    1597    1598    1599
VECTOR  001324           745#    1003#   1167#   1178#   1535#   1600#   1614#   1615#   1618    1620    1622    1698#   1744#
                        1791#   2071#
VECTR1  001330           747#    1168#   1601#   1604#   1618#   1619#   1624#
VECTR2  001332           748#    1194#   1602#   1620#   1621#
VECTR3  001334           749#    1603#   1622#   1623#
VLIN    011612          1963    2194#
VMP     011606          2055    2192#
VMR     011604          2053    2191#
VSET    011610          1667    2193#
VTFLG   002432           879     882     885     933#
VTINIT  014054          2030    2461#
VTSS    002206           887     890#
VVCT    001340           751#    1600    1601    1602    1603
V0      011570          1358    1363    2184#
V12     011576          1260    1295    2187#
V2      011572          1315    2185#
V326    011602          1271    1283    2189#
V4      011574          1464    2186#
V500    011600          1329    2188#
WFTEST  001426           778#    828#    830#    906     951
WIDE    001346           754#   1778#   1854#   1884    1897
WIDMSG  012645          1837    2331#
WRAP    004126          1248#   1579    1588
XADJ    012337          1485    2291#
YADJ    012531          1499    2315#
SAPTHD  001000           593     599#
$ASTAT= ******  U       2925    2940
SATYC   016340          2896    2898#
SATY1   016314          2894#
SATY3   016322          2840    2895#
SATY4   016332          2746    2897#
$AUTOB  001134           630#
$BASE   001250           694#    954    1609    1610    1611
$BOADR  001122           625#
$BODAT  001126           627#    954#    958     964#   1075#   1136#   1150#   1151    1172#   1196#   1228#   1229#   1230
                        1242#   1243    2087#   2088    2529    2531    2534
```

```
$BELL   001164            645#  2733  2765
$BIN    017062           3037*  3040*  3041  3048#
$CDWI   001254            696#
$CHARC  016310           2857*  2867*  2874  2883*  2888#
$CKSWR= ****** U         3080
$CMTAG  001100            613#   833   834   842   846   847   848
$CM3    = 000000          643#
$CNTLG  015065           2606#
$CNTLU  015060           2605#
$CPUOP  001222            668#
$CRLF   001171            647#  1450  1558  1578  2605  2741  2765  2773  2792  2797  2801  2856  2891
$DEVCT  001204            659#
$DEVM   001252            695#
$DOAGN  011754           2215   2226  2232#
$ENDAD  011744            579   2228#  2760
$ENDCT  011664            846   2217#
$ENULL  011760           2235#
$ENV    001214            664#  2743  2835  2903  2927
$ENVM   001215            665#   869   948  2837  2842  2905
$EOP    011630           1567   1583  1592  2207#
$EOPCT  011656            846#  2214#  2218
$ERFLG  001103            616#   787*  2654  2682  2684  2690*  2712  2728*  2765
$ERMAX  001115            622#   849*  2684  2707*  2712
$ERROR  015474            842   2726#
$ERRPC  001116            623#  2529   2531  2534  2735*  2736*  2737  2765  2779
$ERRTB  001256            713#  2787
$ERRTY  015676           2740   2772#
$ERTTL  001112            620#  2734*  2765
$ESCAP  001162            644#   786*   789*   848*  2706*  2756  2758  2765
$ETABL  001214            663#
$ETEND  001256            605    697#
$FATAL  001176            656#  2931#
$FFLG   016560           2894*  2897*  2925  2934*  2942#
$FILLC  001156            641#  2860   2891
$FILLS  001155            640#  2891
$GDADR  001120            624#
$GDDAT  001124            626#   985*   988*   989   996*  1004*  1012*  1020*  1027*  1034*  1063*  1071*  1095*
                         1107*  1130*  1144*  1151  1173*  1197*  1211*  1224*  1230   1241   1243   2085*  2089
                         2529   2531
$GET42  011734           2225#
$GTSWR= ****** U         3079
$HD     = 000000          431
$HIBTS  001000            600#
$HIOCT  015214           2643*  2648#
$ICNT   001104            617#  2697*  2698  2700*  2711
$INTAG  001135            631#
$ITEMB  001114            621#  2737*  2745  2765  2776
$LF     001172            648#  2596   2605  2765  2891
$LFLG   016557           2935*  2941#
$LPADR  001106            618#   850*   983*  1255*  2101*  2119*  2688*  2704*  2709   2711
$LPERR  001110            619#   851*   984*  1254*  2100*  2118*  2688   2705*  2711   2755
$MADR1  001226            681#
$MADR2  001232            685#
$MADR3  001236            688#
$MADR4  001242            691#
$MAIL   001174            601    605   654#   868  2703  2743  2835
```

```
SMAMS1  001224          675#
SMAMS2  001230          683#
SMAMS3  001234          686#
SMAMS4  001240          689#
SMBADR  001002          601#
SMFLG   016556         2895*   2901    2936*   2940#
SMNEW   015103         2609#
SMSGAD  001210          661#   2911*   2914
SMSGLG  001212          662#   2916*
SMSGTY  001174          655#   2909    2917*   2929    2933*
SMSWR   015072         2607#
SMTYP1  001225          676#
SMTYP2  001231          684#
SMTYP3  001235          687#
SMTYP4  001241          690#
SMXCNT  015472         2701    2711#
SNULL   001154          639#   2862    2891
SNWTST= 000001          980#    992#    999#   1008#   1016#   1023#   1030#   1037#   1048#   1059#   1066#   1079#   1089#
                       1102#   1116#   1126#   1140#   1154#   1184#   1204#   1220#   1249#   1262#   1274#   1285#   1300#
                       1317#   1335#   1365#   1396#   1421#
SOCNT   017004         2980*   3009*   3022#
SOMODE  017006         2975*   2979*   2984    2987*   2998*   3024#
SOVER   015456         2665    2681    2689    2699    2708#
SPASS   001202          658#    868*    944#   1426    2098    2116    2211*   2212*   2235    2695    2712
SPASTM  001006          603#
SQUES   001170          646#   1480    2589    2605    2765    2891
SRDCHR  014622         2553#   3080
SRDDEC= ****** U       3083
SRDLIN  014742         2581#   3081
SRDOCT  015114         2621#   3082
SRDSZ = 000010         2574#
SRTNAD  011756         2234#
SR2A  = ****** U       3083
SSAVRE= ****** U       3083
SSCOPE  015216          840   2663#
SSETUP= 000027          714#    839     840     842     844     846     847     848     850    2209    2542    2611    2664
                       2727    2753    2760
SSTUP = 177777          714#
SSVLAD  015422         2673    2702#
SSVPC = 000214          577#    582
SSWR  = 167400          421#    431     545     546     547     548     549     550     551     552     643     644     645
                        847     848     850     851     984     996    1003    1012    1020    1027    1034    1041    1052
                       1063    1070    1083    1093    1106    1120    1130    1144    1158    1188    1208    1224    1253
                       1266    1278    1289    1304    1321    1339    1369    1400    1425    2204    2210    2227    2233
                       2235    2655    2656    2657    2658    2659    2664    2676    2678    2679    2682    2683    2684
                       2691    2692    2693    2705    2708    2711    2718    2719    2720    2721    2722    2731    2738
                       2750    2753    2765
SSWREG  001216          666#    871
SSWRMK= 000000          552     553    2659    2660    2680
STESTN  001200          657#   2703*
STIMES  001160          643#    847#   1070#   1083#   1106#   1253#   1266#   1278#   1289#   1304#   1321#   1339#   1369#
                       1400#   1425#   2210#   2691#   2698   2701#   2711
STKB    001146          636#    793     940    1460    2540    2557    2563
STKS    001144          635#    877#    910#    934    1074#   1086#   1114#   1454#   1455    1461#   2124#   2197#   2540
                       2555    2561
STN   = 000040          421#    431     980     984#    992     996#    999    1003#   1008    1012#   1016    1020#   1023
```

```
                    1027#  1030   1034#  1037   1041#  1045   1048   1052#  1056   1059   1063#  1066   1070#
                    1076   1079   1083#  1089   1093#  1102   1106#  1116   1120#  1126   1130#  1137   1140
                    1144#  1152   1154   1158#  1176   1184   1188#  1204   1208#  1220   1224#  1249   1252
                    1253#  1262   1266#  1274   1278#  1285   1289#  1300   1304#  1317   1321#  1333   1335
                    1339#  1365   1369#  1396   1400#  1415   1421   1425#
STPB    001152       638#  2142*  2152*  2880*  2891
STPFLG  001157       642#  2829   2891
STPS    001150       637#  2140   2150   2878   2891
STRAP   017064       844   3057#
STRP  = 000013      3065#  3074#  3075#  3076#  3077#  3078#  3080   3081#  3082#  3083#  3084#  3085#  3086#
STRPAD  017106      3062   3072#
STSTM   001004       602#
STSTNM  001102       615#  1252*  2209*  2654   2680   2702*  2703   2708   2712   2730   2765
STTYIN  015050      2582   2583   2600   2604#
STYPBN  017010      3034#  3077
STYPOS= ****** U    3077
STYPE   016032      2829#  2922   3065   3073
STYPEC  016244      2859   2866   2873   2878#  2879
STYPEX  016312      2884   2886   2889#
STYPOC  016606      2978#  3074
STYPON  016622      2977   2980#  3076
STYPOS  016562      2973#  3075
SUNIT   001206       660#
SUNITM  001010       604#
SUSWR   001220       667#
SVECT1  001244       692#  1614   1616
SVECT2  001246       693#
SXTSTR  015226      2667#
SSGET4= 000000      2227#
SOFILL  017005      2974*  2978*  2988   3023#
S40CAT= ****** U    2664   2740
.     = 037754       557#   562#   566#   577    578#   580#   582#   583#   589    590#   592#   594#   612#
                     649    837    850    851   2235   2236#  2540   2604#  2605   2611   2711   2712   2765
                    2811#  2891   2943#  3087#  3088#
.SASTA= ****** U    2895   2898
.SX   = 001000       589#   594
```

DVADAA.CMB CROSS REFERENCE TABLE -- MACRO NAMES

```
BERROR    421#   2727
COMMEN      1#    541#
DUMWRN    717#   1158   1188
ENDCOM      1#    541#
ENDPAS    421#   2219
ERROR     435#    788    987    998   1006   1014   1022   1029   1036   1046   1057   1065   1077   1088   1100
         1113   1125   1138   1153   1174   1198   1217   1232   1261   1272   1284   1296   1316   1359   1364
ESCAPE      1#    541#    785
GETPRI      1#    541#
GETSWR      1#    541#
MULT        1#    541#
NEXTST      1#    541#    980    992    999   1008   1016   1023   1030   1037   1048   1059   1066   1079   1089
         1102   1116   1126   1140   1154   1184   1204   1220   1249   1262   1274   1285   1300   1317   1335
         1365   1396   1421
POP         1#    541#   2644   2937   2938
PUSH        1#    541#   2623   2898   2900   2921
REPORT      1#    541#
SCOPE     436#    995   1002   1011   1019   1026   1033   1040   1051   1062   1069   1082   1092   1105   1119
         1129   1143   1157   1187   1207   1223   1236   1265   1277   1288   1303   1320   1338   1368   1399
         1424
SETPRI      1#    541#
SETTRA   3065#   3074   3075   3076   3077   3080   3081   3082   3083   3084   3085
SETUP       1#    541#    832
SKIP        1#    541#    907    917    920    923    926    929    935    937    939    949    952    963    965
          990   1043   1045   1054   1056   1076   1137   1152   1176   1231   1244   1299   1333   1392   1395
         1415   1419   1442   1448   1451   1456   1458   1459   1465   1477   1479   1481   1491   1492   1514
         1515   1528   2021   2023
SLASH       1#    541#
SPACE     541#
STARS       1#    541#    575    586    588    595    608    649    652    980    982    992    994    999   1001
         1008   1010   1016   1018   1023   1025   1030   1032   1037   1039   1048   1050   1059   1061   1066
         1068   1079   1081   1089   1091   1102   1104   1116   1118   1126   1128   1140   1142   1154   1156
         1184   1186   1204   1206   1220   1222   1249   1251   1262   1264   1274   1276   1285   1287   1300
         1302   1317   1319   1335   1337   1365   1367   1396   1398   1421   1423   2202   2539   2545   2574
         2613   2651   2714   2767   2814   2893   2950   3027   3051
SWRSU       1#    541#    852#
TRMTRP    306S#
TYPBIN      1#    541#   2223
TYPDEC      1#    541#
TYPNAM      1#    541#
TYPNUM      1#    541#
TYPOCS      1#    541#    812    967   1530   1548   1643   1651   1662   1712   1721   1862   1951   1958   2048
         2103   2109
TYPOCT      1#    541#   2779   2803
TYPTXT      1#    541#   2219
$$CMRE    606#
$$CMTM    606#
$$ESCA      1#    541#
$$NEXT      1#    541#    980    992    999   1008   1016   1023   1030   1037   1048   1059   1066   1079   1089
         1102   1116   1126   1140   1154   1184   1204   1220   1249   1262   1274   1285   1300   1317   1335
         1365   1396   1421
$$SET     3065#   3074   3075   3076   3077   3080   3081   3082   3083   3084   3085
$$SETM     868#
$$SKIP      1#    541#   1045   1056   1076   1137   1152   1176   1333   1415
.EQUAT      1#    421#    431
.MEADE      1#    421#
```

```
.KT11        1#
.SETUP       1#    421#    714
.SWRHI       1#    421#    541
.SWRLO     553#
.SACTI       1#    421#    573
.SAPTB       1#    421#    650#
.SAPTH       1#    421#    584
.SAPTY       1#    421#   2891
.SASTA       1#
.SCATC       1#    421#    560
.SCMTA       1#    421#    606
.SDB2D       1#
.SDB2O       1#
.SDIV        1#
.SEOP        1#    421#   2200
.SERRO       1#    421#   2712
.SERRT       1#    421#   2765
.SMULT       1#
.SPARM     421#
.SPOWE       1#    421#
.SRAND       1#    421#
.SRODE       1#
.SRODC       1#    421#   2611
.SREAD       1#    421#   2537
.SR2AZ       1#
.SSAVE       1#    421#
.SSB2D       1#
.SSB2O       1#
.SSCOP       1#    421#   2649
.SSIZE       1#
.SSPAC     421#
.SSUPR       1#
.SSWDO     421#
.STRAP       1#    421#   3049
.STYPB       1#    421#   3025
.STYPD       1#    421#
.STYPE       1#    421#   2812
.STYPO       1#    421#   2948
.S40CA       1#
.1170        1#
```

```
ADC     1707  1795  1798  1801  1837  1947  1976  2081
ADCB    3040
ADD      823   964  1245  1378  1384  1438  1466  1467  1510  1597  1598  1599  1600  1601  1602
        1603  1607  1613  1619  1621  1623  1678  1701  1719  1740  1786  1793  1794  1796  1797
        1799  1800  1885  1897  1898  1931  1977  2063  2075  2639  2787  2848  2908  2920  2932
        2976  2986
ASL      961   988  1596  1733  1734  1735  1736  1813  1842  2632  2634  2636  2784  2785  2786
        3061
ASR     1704  1705  1706  1761  1834  1835  1836  1844  1861  1941  1944  1945  1946  1973  1974
        1975  2078  2079  2080  2143  2144  2145  2146  2147  2915
BEQ      810   870   887   904   952   963  1045  1076  1137  1152  1231  1415  1427  1477  1479
        1544  1595  1630  1718  1804  1810  1812  1819  1877  1892  1909  1969  2021  2167  2226
        2631  2679  2681  2683  2687  2696  2729  2732  2754  2757  2789  2794  2807  2838  2851
        2886  2902  2906  2926  2928  3003  3039
BGE     1351  1754  1846  1991  2699
BGT     1668  1759  2054  2056  2094  2215  2571  3010
BHI     2685
BIC      794   877   941  1098  1229  1526  1615  1803  2138  2148  2212  2558  2564  2572  2638
        2727  3000
BICB     915
BIS      910   960  1074  1086  1114  1537  1691  1745  1750  1788  2124  2139  2149  2197  3005
        3006
BISB    2073  2179  2180  2181  2776
BIT     1044  1055  1527  1543  2020  2664  2678  2686  2693  2731  2738  2753
BITB     869  2837  2842  2874  2905
BLE     1348  1853  1904  1937  1964  1994  2161
BLOS    2584
BLT     1839  2135  2569  2865  3011
BMI      935   949  1442  1456  1822
BNE      796   802   808   822   837   859   875   881   884   889   907   917   920   923   926
         929   937   990  1056  1097  1215  1244  1299  1354  1392  1395  1448  1458  1528  1555
        1557  1636  1695  1703  1757  1762  1773  1784  1806  1825  1830  1849  1856  1871  1920
        1943  1980  1997  2002  2018  2027  2029  2070  2077  2099  2117  2171  2175  2560  2566
        2588  2594  2665  2694  2739  2744  2761  2777  2799  2836  2843  2845  2853  2861  2875
        2882  2904  2910  2913  2930  3001
BPL     1043  1054  1110  1122  1135  1149  1171  1210  1346  1409  1815  1934  2035  2038  2091
        2132  2141  2151  2157  2556  2562  2751  2830  2879  2999
BR       826   829   861   932   939   965  1176  1200  1330  1331  1333  1419  1451  1459  1465
        1481  1491  1492  1514  1515  1546  1559  1565  1581  1590  1606  1640  1817  1826  1828
        1841  1851  1879  1894  1906  1966  2023  2178  2220  2567  2590  2640  2667  2673  2676
        2689  2692  2749  2782  2809  2832  2858  2868  2877  2884  2896  2918  2977  2992  3013
        3043
CLC     3042
CLR      789   828   835   847   848   868   873   911   933   944   945   957  1071  1107  1130
        1131  1132  1146  1164  1179  1203  1235  1305  1341  1371  1374  1375  1406  1453  1454
        1520  1522  1604  1624  1632  1696  1738  1739  1771  1776  1777  1778  1779  1780  1781
        1782  1827  1917  1925  1926  2024  2036  2039  2066  2125  2133  2209  2210  2628  2629
        2691  2706  2775  2990
CLRB    1617  2163  2164  2165  2172  2176  2595  2690  2857  2883  2934  2935  2936
CMP      809   821   836   858   880   883   886   888   966   989  1151  1177  1201  1230  1243
        1298  1347  1350  1391  1394  1414  1629  1635  1638  1639  1667  1717  1753  1758  1811
        1818  1838  1845  1852  1903  1936  1942  1963  1990  1993  2000  2053  2055  2093  2134
        2160  2559  2565  2568  2570  2583  2674  2698  2760
CMPB     795   801   807   916   919   922   925   928  1476  1478  2170  2174  2587  2593  2680
        2684  2743  2835  2850  2852  2860  2881  2885  2903
DEC      936   962   974  1096  1214  1353  1447  1554  1556  1605  1694  1702  1756  1772  1783
```

M07

	1805	1824	1829	1870	1919	1979	1996	2001	2017	2026	2028	2069	2076	2168	2213
	2783														
DECB	2864	2867	2998	3009											
EMT	435														
HALT	566	2022	2752	2762	2831										
INC	787	890	959	1094	1108	1120	1297	1390	1393	1457	1540	1612	1699	1755	1807
	1814	1820	1821	1840	1843	1847	1850	1854	1857	1957	2211	2697	2734	2933	3004
	3012														
INCB	1344	2169	2173	2177	2702	2728	2887								
IOT	436														
JMP	570	571	572	800	806	905	918	921	924	927	930	1494	1517	1567	1583
	1592	2198	2233												
JSR	799	805	879	882	885	894	1163	1175	1182	1193	1199	1202	1256	1258	1267
	1269	1279	1281	1291	1293	1306	1310	1313	1324	1326	1327	1356	1361	1373	1376
	1379	1382	1385	1389	1405	1407	1408	1413	1428	1432	1436	1462	1488	1489	1490
	1504	1508	1512	1513	1562	1563	1564	1571	1572	1579	1580	1587	1588	1589	1649
	1660	1676	1680	1910	1916	1918	1922	2006	2009	2011	2016	2046	2228	2740	2746
	2840	2859	2866	2873	2922										
MOV	786	792	793	798	804	812	819	820	830	834	838	840	841	842	843
	844	845	846	850	851	854	855	856	857	862	864	865	866	871	895
	896	897	898	899	900	901	912	914	940	946	947	950	953	954	955
	956	968	975	976	977	983	984	985	996	1003	1004	1012	1020	1027	1034
	1041	1052	1063	1070	1072	1075	1083	1084	1093	1095	1101	1106	1123	1133	1136
	1144	1145	1147	1150	1162	1165	1167	1168	1169	1172	1173	1178	1180	1192	1194
	1195	1196	1197	1208	1211	1212	1213	1219	1224	1225	1228	1234	1241	1242	1252
	1253	1254	1255	1266	1278	1289	1290	1304	1308	1309	1321	1322	1323	1339	1340
	1342	1343	1349	1352	1355	1360	1369	1372	1388	1400	1402	1403	1404	1411	1412
	1416	1417	1418	1425	1431	1434	1435	1441	1449	1461	1471	1472	1475	1487	1502
	1503	1506	1507	1519	1523	1525	1530	1535	1538	1539	1542	1545	1548	1553	1561
	1566	1570	1574	1582	1586	1591	1608	1609	1610	1611	1614	1618	1620	1622	1625
	1627	1628	1631	1633	1643	1651	1657	1659	1662	1675	1679	1688	1692	1693	1697
	1698	1711	1712	1721	1732	1737	1741	1742	1744	1747	1748	1751	1766	1767	1768
	1769	1770	1774	1775	1785	1789	1790	1791	1792	1802	1809	1816	1823	1831	1832
	1833	1859	1862	1873	1881	1884	1888	1896	1899	1913	1914	1915	1928	1929	1932
	1933	1938	1939	1948	1951	1958	1970	1971	1972	1978	1981	1982	1983	1984	1985
	1986	1987	1988	1989	1992	1995	1998	1999	2005	2008	2013	2014	2025	2040	2043
	2048	2062	2064	2067	2068	2071	2072	2085	2086	2087	2088	2089	2100	2101	2103
	2109	2118	2119	2126	2136	2137	2162	2216	2223	2225	2553	2554	2581	2582	2597
	2598	2599	2600	2621	2622	2623	2624	2625	2627	2642	2643	2644	2645	2646	2669
	2670	2672	2675	2688	2700	2701	2704	2705	2708	2709	2730	2735	2755	2758	2774
	2779	2788	2793	2798	2800	2804	2833	2834	2839	2847	2862	2899	2900	2907	2911
	2916	2917	2919	2921	2931	2937	2938	2973	2981	2982	2983	2989	2996	3014	3015
	3016	3017	3018	3034	3035	3044	3045	3046	3057	3058	3062				
MOVB	849	1226	1227	1616	1689	1743	1872	1901	2142	2152	2557	2563	2586	2591	2630
	2703	2707	2737	2745	2844	2872	2880	2894	2895	2897	2974	2975	2978	2979	2980
	2984	2987	2988	3007	3037	3060									
NEG	1410	1935	2092	2159	2985										
NOP	2208	2229	2230	2231											
RESET	831	893	902	1073	1085	1111	1493	1516	2123	2227					
ROL	2633	2635	2637	2991	2993	2994	2995	2997	3038						
RTI	790	824	863	913	1166	1181	1246	1524	2127	2183	2573	2601	2647	2710	2764
	2849	3019	3047												
RTS	942	978	1237	1429	1440	1446	1468	1637	1670	1672	1683	1708	1726	1763	2012
	2019	2031	2058	2060	2082	2096	2114	2121	2128	2153	2802	2889	2939	3063	
SEC	3036														
SUB	1312	1381	1387	1439	1511	1658	1682	1760	1860	1930	1940	2090	2736	2914	

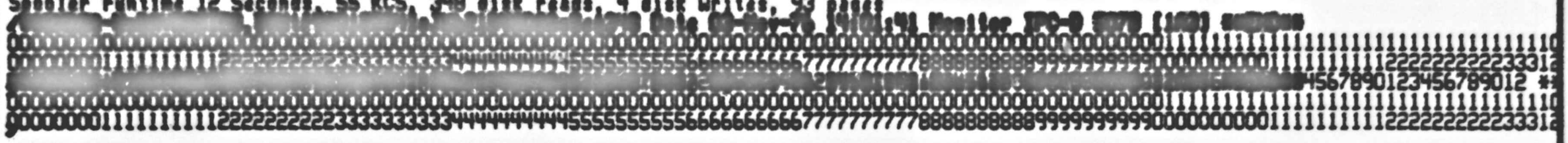
```
SWAB    1536  1690  1749  1787  2065
TRAP    3065  3074  3075  3076  3077  3080  3081  3082  3083  3084  3085
TST      874   903   906   938   951   958  1183  1426  1460  1484  1498  1594  1634  1848  1855
        1876  1891  1908  1968  2015  2034  2037  2095  2098  2116  2131  2156  2166  2641  2671
        2695  2750  2756  2806  2846  2854  2876  2909  2927  2929  3002  3059
TSTB     934   948  1042  1053  1109  1121  1134  1148  1170  1209  1345  1455  2140  2150  2555
        2561  2682  2829  2878  2901  2912  2925
WAIT    1541  1700  1746  1752  1808  2074
.ASCII   646   647  2291  2298  2354  2364  2377  2385  2391  2427  2465  2472  2475  2478  2481
.ASCIZ   645   648  2222  2238  2241  2255  2257  2258  2261  2262  2263  2265  2268  2269  2271
        2274  2276  2277  2279  2283  2294  2304  2308  2315  2318  2319  2322  2325  2331  2337
        2342  2349  2351  2358  2361  2368  2372  2403  2414  2420  2424  2430  2436  2437  2438
        2439  2441  2443  2444  2484  2488  2492  2496  2500  2504  2509  2518  2605  2606  2607
        2609  2810
.BLKB   2604
.BLKW   3087  3088
.BYTE    615   616   621   622   630   631   639   640   641   642   664   665   675   676   683
         684   686   687   689   690   815   816   971   972  1533  1534  1551  1552  1576  1577
        1646  1647  1654  1655  1665  1666  1715  1716  1724  1725  1865  1866  1954  1955  1961
        1962  2051  2052  2106  2107  2112  2113  2235  2245  2246  2247  2249  2251  2447  2455
        2461  2523  2524  2525  2526  2602  2603  2747  2748  2940  2941  2942  3020  3021  3022
        3023  3048
.DSABL  2542
.ENABL     1   421  2540
.END    3090
.ENDC    426   435   527   541   549   551   552   553   571   576   580   582   587   589   596
         609   613   615   643   644   645   646   650   653   675   683   686   689   692   693
         694   695   696   699   714   787   816   817   838   839   842   844   846   847   848
         850   852   873   908   918   921   924   927   930   936   938   940   950   953   964
         966   972   973   981   982   983   984   991   993   994   995   996  1000  1001  1002
        1003  1009  1010  1011  1012  1017  1018  1019  1020  1024  1025  1026  1027  1031  1032
        1033  1034  1038  1039  1040  1041  1044  1046  1049  1050  1051  1052  1055  1057  1060
        1061  1062  1063  1067  1068  1069  1070  1071  1077  1080  1081  1082  1083  1090  1091
        1092  1093  1103  1104  1105  1106  1107  1117  1118  1119  1120  1127  1128  1129  1130
        1138  1141  1142  1143  1144  1153  1155  1156  1157  1158  1177  1185  1186  1187  1188
        1205  1206  1207  1208  1221  1222  1223  1224  1232  1245  1250  1251  1252  1253  1254
        1263  1264  1265  1266  1267  1275  1276  1277  1278  1279  1286  1287  1288  1289  1290
        1300  1301  1302  1303  1304  1305  1318  1319  1320  1321  1322  1334  1336  1337  1338
        1339  1366  1367  1368  1369  1370  1393  1396  1397  1398  1399  1400  1401  1416  1420
        1422  1423  1424  1425  1426  1443  1449  1452  1457  1459  1460  1466  1478  1480  1482
        1492  1493  1515  1516  1529  1534  1535  1552  1553  1647  1648  1655  1656  1666  1667
        1716  1717  1725  1726  1866  1867  1955  1956  1962  1963  2022  2024  2052  2053  2107
        2108  2113  2114  2203  2204  2206  2209  2215  2218  2219  2222  2225  2227  2233  2235
        2236  2540  2541  2542  2546  2574  2575  2582  2584  2587  2589  2605  2611  2614  2616
        2649  2652  2655  2660  2664  2666  2677  2680  2681  2682  2684  2686  2693  2697  2702
        2704  2708  2711  2712  2715  2718  2728  2735  2740  2741  2742  2750  2760  2764  2765
        2768  2783  2812  2915  2844  2894  2895  2898  2925  2940  2951  3028  3052  3058  3061
        3073  3074  3075  3076  3077  3078  3079  3080  3081  3082  3083  3084  3085
.EQUIV   435   436   444   459   460   489   490   491   492   493   494   495   496   497   498
         517   518   519   520   521   522   523   524   525   526
.EVEN    653  2222  2236  2371  2527  2811  2943  3086
.IF      422   433   498   527   548   550   551   552   553   569   575   578   580   586   588
         595   608   612   614   643   644   645   649   650   652   675   683   686   689   692
         693   694   695   696   697   698   714   786   815   816   833   838   840   842   844
         846   847   848   850   868   907   917   920   923   926   929   935   937   939   949
         952   963   965   971   972   980   982   984   990   992   994   996   999  1001  1003
```

```
        1008  1010  1012  1016  1018  1020  1023  1025  1027  1030  1032  1034  1037  1039  1041
        1043  1045  1048  1050  1052  1054  1056  1059  1061  1063  1066  1068  1070  1071  1076
        1079  1081  1083  1089  1091  1093  1102  1104  1106  1107  1116  1118  1120  1126  1128
        1130  1137  1140  1142  1144  1152  1154  1156  1158  1176  1184  1186  1188  1204  1206
        1208  1220  1222  1224  1231  1244  1249  1251  1253  1254  1262  1264  1266  1267  1274
        1276  1278  1279  1285  1287  1289  1290  1299  1300  1302  1304  1305  1317  1319  1321
        1322  1333  1335  1337  1339  1365  1367  1369  1370  1392  1395  1396  1398  1400  1401
        1415  1419  1421  1423  1425  1426  1442  1448  1451  1456  1458  1459  1465  1477  1479
        1481  1491  1492  1514  1515  1528  1533  1534  1551  1552  1646  1647  1654  1655  1665
        1666  1715  1716  1724  1725  1865  1866  1954  1955  1961  1962  2021  2023  2051  2052
        2106  2107  2112  2113  2202  2203  2204  2205  2206  2208  2214  2217  2219  2221  2225
        2227  2233  2235  2236  2539  2541  2542  2545  2546  2574  2582  2583  2587  2588  2604
        2605  2611  2613  2616  2628  2651  2654  2659  2664  2676  2678  2679  2680  2682  2683
        2684  2693  2695  2703  2705  2710  2711  2712  2714  2717  2727  2731  2738  2740  2741
        2743  2750  2753  2760  2764  2765  2767  2782  2798  2814  2835  2893  2895  2898  2925
        2940  2950  3027  3051  3057  3061  3065  3074  3075  3076  3077  3079  3080  3081  3082
        3083  3084  3085
.IFF     433   548   551   552   553   576   580   582   587   589   596   609   612   615   [illegible]
         650   653   786   815   817   838   907   917   920   923   926   929   935   937   939
         949   952   963   965   972   980   981   982   983   984   990   993   994   995   996
        1000  1001  1002  1003  1009  1010  1011  1012  1017  1018  1019  1020  1024  1025  1026
        1027  1031  1032  1033  1034  1038  1039  1040  1041  1043  1046  1049  1050  1051  1052
        1054  1057  1060  1061  1062  1063  1067  1068  1069  1070  1077  1080  1081  1082  1083
        1090  1091  1092  1093  1103  1104  1105  1106  1117  1118  1119  1120  1127  1128  1129
        1130  1138  1141  1142  1143  1144  1153  1155  1156  1157  1158  1177  1185  1186  1187
        1188  1205  1206  1207  1208  1221  1222  1223  1224  1231  1244  1249  1250  1251  1252
        1253  1263  1264  1265  1266  1275  1276  1277  1278  1286  1287  1288  1289  1299  1301
        1302  1303  1304  1305  1318  1319  1320  1321  1322  1334  1336  1337  1338  1339  1366
        1367  1368  1369  1370  1392  1395  1397  1398  1399  1400  1401  1416  1419  1422  1423
        1424  1425  1426  1442  1448  1451  1456  1458  1459  1465  1477  1479  1481  1491  1492
        1514  1515  1528  1534  1552  1553  1647  1655  1666  1667  1716  1717  1725  1726  1866
        1867  1955  1956  1962  1963  2021  2023  2052  2107  2112  2114  2203  2205  2208  2215
        2218  2235  2540  2542  2546  2548  2553  2574  2575  2584  2588  2605  2614  2652  2677
        2680  2681  2684  2711  2712  2715  2717  2731  2760  2765  2768  2783  2812  2815  2894
        2951  3028  3052  3058
.IFT    2222  2548  2553  2632  2648  2649  2692  2741
.IFTF   2222  2542  2546  2549  2628  2632  2648  2690  2740
.IIF     421   426   431   545   546   547   549   552   553   566   649   653   813   839   842
         846   847   848   850   851   969  1531  1549  1644  1652  1663  1713  1722  1863  1952
        1959  2049  2104  2110  2204  2209  2210  2224  2235  2236  2540  2597  2605  2611  2655
        2656  2657  2658  2659  2660  2664  2691  2692  2708  2711  2712  2718  2719  2720  2721
        2722  2727  2753  2760  2765  2780  2805  2891  3073  3074  3075  3076  3077  3080  3081
        3082  3083  3084  3085
.IRP     714   980   992   999  1008  1016  1023  1030  1037  1048  1059  1066  1079  1089  1102
        1116  1126  1140  1154  1184  1204  1220  1249  1262  1274  1285  1300  1317  1335  1365
        1396  1421  2208  2623  2644  2727  2899  2900  2921  2937  2938
.LIST      1   421   541   552   566   643   650   653   714   852   980   984   992   996   999
        1003  1008  1012  1016  1020  1023  1027  1030  1034  1037  1041  1048  1052  1059  1063
        1066  1070  1079  1083  1089  1093  1102  1106  1116  1120  1126  1130  1140  1144  1154
        1158  1184  1188  1204  1208  1220  1224  1249  1253  1262  1266  1274  1278  1285  1289
        1300  1304  1317  1321  1335  1339  1365  1369  1396  1400  1421  1425  2209  2222  2227
        2574  2659  2760  3065  3073  3074  3075  3076  3077  3078  3080  3081  3082  3083  3084
        3085  3086
.MACRO     1   421   553   606   717   868  3065
.MCALL   421   541   650   852
.MEXIT   698
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

```
*,DVADAA.SEQ/SOL/CRF/PAGNUM/NL:TOC/DS:ERFZ=DVADAA.SML,DVADAA.CMB
RUN-TIME:  39 51 7 SECONDS
RUN-TIME RATIO:  173/98=1.7
CORE USED:  34K  (67 PAGES)
```

Spooler runtime 12 Seconds, 55 KCS, 348 disk reads, 4 disk writes, 93 pages



```
                                ...C1     2246    012027       077    ...C5
  114                           ...D1     2300    012412    047522    ...D5
                                ...E1     2351    013021       040    ...E5
  171                           ...F1     2386    013302    051440    ...F5
  206                           ...G1     2439    013736    005015    ...G5
  258                           ...H1     2474    014150    052123    ...H5
  296                           ...I1     2513    014464    047040    ...I5
  352                           ...J1     2546                        ...J5
  405                           ...K1     2602    015046       000    ...K5
  461                           ...L1     2620                        ...L5
  517                           ...M1     2658                        ...M5
                                ...N1     2714                        ...N5

  582          000214           ...B2     2770                        ...B6
  615  001102      000           ...C2     2821                        ...C6
  671                           ...D2     2877    016242    000724    ...D6
  708                           ...E2     2933    016530    005237    ...E6
  750  001336  000004           ...F2     2957                        ...F6
  800  001520  000137           ...G2     3013    016760    000744    ...G6
  836  001670  022706           ...H2     3034    017010    010146    ...H6
  882  002154  004737           ...I2     3058    017066    016600    ...I6
  900  002254  012737           ...J2                                 ...J6
  942  002462  000207           ...K2  ADDW10=  000000                ...K6
  988  002716  006337           ...L2  BASEBR   001326                ...L6
 1039                           ...M2  CH2      001362                ...M6
 1088  003240  104001           ...N2  GETDAT   010404                ...N6

 1111  003332  000005           ...B3  NOTNEW   010276                ...B7
 1163  003522  004737           ...C3  RET      007070                ...C7
 1213  003760  012700           ...D3  SP    =%000006                ...D7
 1256  004120  004537           ...E3                                 ...E7
 1309  004356  012777           ...F3                                 ...F7
 1344  004526  105277           ...G3  $CPUOP   001222                ...G7
 1374  004656  005037           ...H3  $MSGTY   001174                ...H7
 1405  005046  004737           ...I3  $TRP  =  000013                ...I7
 1450  005266  104400           ...J3  GETSWR       1#      541#      ...J7
 1491  005476  000771           ...K3  .$CATC       1#      421#      ...K7
 1527  005676  032777           ...L3                1975    2078     ...L7
 1569                           ...M3  IOT           436     2169     ...M7
 1602  006256  063737           ...N3  .ASCIZ       645      648     ...N7

 1650  006550  104400           ...B4                1666    1715     ...B8
 1693  006740  012700           ...C4  .REPT         566      699     ...C8
 1737  007120  012737           ...D4   **END** USER DAVIES,TOM       ...D8
 1773  007322  001375           ...E4
 1827  007556  005037           ...F4
 1879  010000  000402           ...G4
 1932  010250  010120           ...H4
 1981  010430  012700           ...I4
 2029  010656  001374           ...J4
 2070  011050  001376           ...K4
 2106  011230      002           ...L4
 2131  011322  005702           ...M4
 2163  011450  105037           ...N4
```