VT55

ACCEPTANCE TEST

MD-11-DZVTD-B

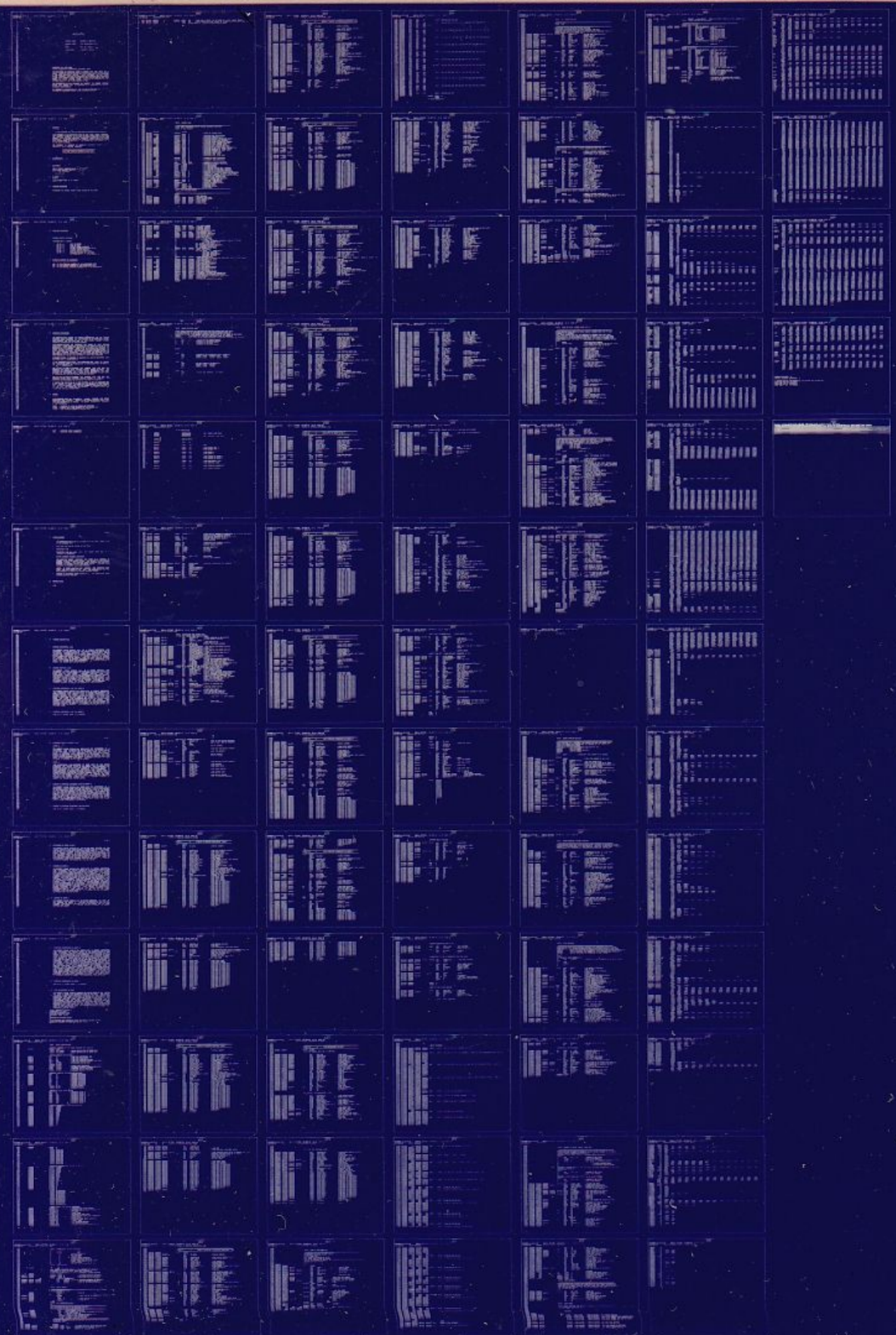
EP-DZVTD-B-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.A



.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZVTD-B

PRODUCT NAME: VT55 ACCEPTANCE TEST

DATE: AUG. 21, 1976

MAINTAINER: DIAGNOSTIC GROUP

1.0 ABSTRACT

** THE PROGRAM WILL RUN ON NON-SWITCH REGISTER CPU TYPES **

THIS PROGRAM IS AN ACCEPTANCE TEST OF THE VT55 VIDEO TERMINAL. THE PROGRAM CONSISTS OF 13 TEST PATTERNS DISPLAYED ON THE VT55 SCREEN. EACH PATTERN REQUIRES OPERATOR INSPECTION FOR ERROR DETECTION. A DESCRIPTION OF THE CORRECT VISUAL DISPLAY FOR EACH TEST CAN BE FOUND IN SECTION 9. THE SCREEN WILL BE COPIED IN 9 OF THE TEST PATTERNS.

THE PROGRAM IS CAPABLE OF HANDLING MULTIPLE VT55'S IN A SEQUENTIAL DL-11 FASHION, HOWEVER:

****
ONLY ONE VT55 IS TESTED AT ONE TIME.
****

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 FAMILY COMPUTER WITH 4K OF MEMORY.
VT55 VIDEO GRAPHIC TERMINAL
DL-11 TYPE INTERFACE

2.2 STORAGE

THIS PROGRAM USES 4K OF MEMORY.

3.0 LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

SW 15 = 1	HALT ON ERROR
SW 14 = 1	LOOP ON TEST
SW 13 = 1	INHIBIT ERROR TYPEOUTS
SW 12 = 1	INHIBIT PROGRAM SUB-TEST DELAY
SW 11 = 1	INHIBIT COPIER TESTING
SW 08 = 1	LOOP ON TEST IN SWR <4:0>
SW 07 = 1	KEYBOARD CONTROL OF THE TEST <SW 8 AND SW 7 = 1 IS AN ERROR>

4.2 STARTING ADDRESS OR ADDRESSES

200 IS THE STARTING ADDRESS OF THE ACCEPTANCE TEST
204 IS THE RESTART ADDRESS OF THE ACCEPTANCE TEST
210 IS THE SPECIAL STARTING ADDRESS FOR VT-55 PRODUCTION

5.0 OPERATING PROCEDURE

THE OPERATOR MUST INSERT THE CORRECT INFORMATION IN THE SWITCH REGISTER WHEN REQUIRED BY THE PROGRAM OR AN ERROR WILL OCCUR. ONCE STARTED, THE TEST WILL RUN IN ITS NORMAL MANNER WITHOUT OPERATOR INTERVENTION OR SWITCH CHANGES.

THIS PROGRAM ALLOWS THE OPERATOR TWO MODES OF TEST PATTERN SELECTION. THESE MODES ARE SELECTED BY THE STATE OF SW 07 AT THE BEGINNING OF THE PROGRAM. WHEN SW 07 IS A ZERO, THE PROGRAM IS UNDER SWITCH REGISTER CONTROL FOR TEST PATTERN SELECTION. IF SW 07 IS EQUAL TO A ONE, THE PROGRAM IS UNDER KEYBOARD CONTROL OF THE TEST PATTERN SELECTION. IN THIS MODE THE OPERATOR WILL BE REQUIRED TO TYPE IN ON THE CONSOLE TTY THE FIRST AND LAST OCTAL BASE ADDRESS OF THE DL-11'S TO WHICH VT-55'S ARE CONNECTED.

KEYBOARD CONTROL IS ASSUMED IF RUNNING ON A SWITCH REGISTER LESS CPU TYPE (I.E., 11/04 LSI-11).

IN THE KEYBOARD SELECT MODE, TWO CHARACTERS ARE USED TO SELECT THE "STARTING WITH" OR "LOOPING ON" A PARTICULAR TEST PATTERN BY "/" OR "\" RESPECTFULLY.

THE "/" KEY IS USED TO SUSPEND THE CURRENT TEST AND ASK THE OPERATOR AT WHICH TEST PATTERN HE/SHE WISHES TO START. THE OPERATOR NOW DEPRESSES THE LETTER WHICH REPRESENTS THE TEST PATTERN TO BE STARTED WITH. REFER TO THE PROGRAM LISTING TABLE OF CONTENTS FOR THE TEST LETTER OF EACH PATTERN.

THE "\" KEY IS USED TO SUSPEND THE CURRENT TEST AND ASK THE OPERATOR WHICH TEST PATTERN HE/SHE WISHES TO LOOP ON. THE OPERATOR NOW DEPRESSES THE LETTER OF THE TEST TO LOOP ON.

IF DURING THE EXECUTION OF A TEST PATTERN, A KEY IS DEPRESSED AND SW 07 EQUALS A ZERO, AN ERROR WILL BE REPORTED TO THE CONSOLE TTY. IF SW 07 EQUALS A ONE, AND THE CHARACTER RECEIVED WAS NOT A "/" OR "\", AN ERROR WILL BE REPORTED. THE CODES "X-OFF" AND "X-ON" ARE THE ONLY EXCEPTIONS.

6.0 ERRORS

THIS PROGRAMS USES THE DIAGNOSTIC 'SYSMAC' PACKAGE FOR ERROR REPORTING AND TYPEOUT. REFER TO THE "ERROR POINTER TABLE" FOR TYPE AND DESCRIPTION OF ERRORS. THE ERROR INFORMATION CONSISTS OF THE FOLLOWING:

ERRPC - LOCATION AT WHICH AN ERROR WAS DETECTED
VTNOW - CURRENT DL-11 BUS ADDRESS OF VT55 UNDER TEST
TSTNUM - TEST PATTERN NUMBER OF FAILING TEST

```
                                EXPT    - EXPECTED INPUT CHARACTER
                                RCVD    - RECEIVED INPUT CHARACTER
```

7.0 MISCELLANEOUS

1. THE OPERATOR SHOULD SET SW 15 AND 13 IF THE VT55 UNDER TEST IS THE CONSOLE TTY.

2. ONLY ONE VT55 CAN BE TESTED AT ONE TIME.

3. EXECUTION TIME

   EXECUTION TIME WILL VARY WITH THE "BAUD" RATE AND IF A "COPIER" IS CONNECTED.

4. DEVICE ADDRESS PROGRAM LOCATIONS

   THE LOCATION "SBASE" CONTAINS THE FIRST DL11 ADDRESS IF SEVERAL VT-55'S ARE BEING TESTED. THE DEFAULT IS THE CONSOLE ADDRESS (177560). THE LOCATION "LAST" CONTAINS THE LAST DL11 ADDRESS IF SEVERAL VT-55'S ARE BEING TESTED. LOCATION VTNOW CONTAINS THE CURRENT DL11 BASE ADDRESS.

   #NOTE: IF THESE LOCATIONS ARE CHANGED, THE OPERATOR MUST START THE TEST AGAIN AT LOC. 200. THE PROGRAM WILL USE THE BASE ADDRESS TO UPDATE THE ACTUAL PROGRAM VALUES.

5. PROGRAM IS CHAINABLE UNDER XXDP/ACT-11. APT HOOKS HAVE BEEN PROVIDED BUT NOT TESTED.

8.0 RESTRICTIONS

NONE.

9.0 PROGRAM DESCRIPTION

9.1 GROWING HORIZONTAL LINE

THE CORRECT VISUAL DISPLAY WILL BE A SINGLE HORIZONTAL LINE EXTENDING THE ENTIRE WIDTH OF THE SCREEN PLACED AT BASE ZERO OF THE SCREEN. ANOTHER HORIZONTAL LINE WILL SUCCESSIVELY APPEAR GIVING THE IMPRESSION OF A GROWING HORIZONTAL LINE, UNTIL THE ENTIRE SCREEN HAS BEEN FILLED. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED. THEN THE FIRST LINE AT THE BASE OF THE SCREEN WILL BE REMOVED, FOLLOWED BY EACH SUCCESSIVE LINE UNTIL THE ENTIRE BLOCK HAS DISAPPEARED. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.2 GROWING VERTICAL LINE

THE CORRECT VISUAL DISPLAY WILL BE A SINGLE VERTICAL LINE EXTENDING THE ENTIRE HEIGHT OF THE SCREEN AND PLACED AT THE FAR RIGHT SIDE OF THE SCREEN. ANOTHER VERTICAL LINE WILL SUCCESSIVELY APPEAR, GIVING THE IMPRESSION OF A GROWING VERTICAL LINE RIGHT TO LEFT, UNTIL THE ENTIRE SCREEN HAS BEEN FILLED. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED. THEN THE FIRST LINE AT THE RIGHT WILL BE REMOVED, FOLLOWED BY EACH SUCCESSIVE LINE UNTIL THE ENTIRE BLOCK HAS DISAPPEARED. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.3 STEPPING HORIZONTAL LINE FOR GRAPH 0

THE CORRECT VISUAL DISPLAY WILL BEGIN WITH A SINGLE HORIZONTAL LINE APPEARING NEAR THE CENTER OF THE SCREEN AND EXTENDING THE ENTIRE WIDTH OF THE SCREEN. THEN A SECOND HORIZONTAL LINE HALFWAY BETWEEN THE FIRST LINE AND THE BASE OF THE SCREEN WILL BEGIN TO GROW FROM THE LEFT SIDE OF THE SCREEN. AS THIS LINE GROWS, THE FIRST LINE DISAPPEARS AND YOU'RE LEFT WITH A SINGLE HORIZONTAL LINE ABOUT ONE FOURTH THE WAY UP THE SCREEN. THEN ANOTHER LINE BEGINS TO GROW FROM THE LEFT, HALFWAY BETWEEN THE PREVIOUS LINE AND THE BASE OF THE SCREEN. AS THIS ONE GROWS THE PREVIOUS LINE IS REMOVED. THIS PROCEDURE CONTINUES FOR A TOTAL OF EIGHT TIMES.

9.4 STEPPING HORIZONTAL LINE FOR GRAPH 1

SAME AS 9.3 EXCEPT GRAPH 1 IS ENABLED.

9.5 DIFFERENT DATA ON GRAPH 0 AND 1

GRAPH 0:

THE CORRECT VISUAL DISPLAY WILL BEGIN WITH THE APPEARANCE OF A HORIZONTAL LINE EXTENDING THE ENTIRE WIDTH OF THE SCREEN PLACED AT THE BASE ZERO. AS THIS LINE IS REMOVED FROM THE LEFT TO RIGHT, A DIAGONAL LINE, BEGINNING AT THE LEFT BOTTOM CORNER, BEGINS TO GROW UNTIL IT REACHES THE TOP OF THE SCREEN. AT THIS POINT A SECOND DIAGONAL LINE BEGINS TO GROW UP FROM THE BASE LINE ABOUT IN THE MIDDLE OF THE SCREEN. THIS LINE CONTINUES TO GROW UP UNTIL IT REACHES THE TOP OF THE SCREEN.

GRAPH 1:

THEN A HORIZONTAL LINE REAPPEARS AT BASE ZERO OF THE SCREEN EXTENDING THE ENTIRE WIDTH OF THE SCREEN. NOW A DIAGONAL LINE BEGINNING AT THE TOP OF THE LEFT CORNER OF THE SCREEN BEGINS TO DECAY DOWNWARD AS THE BASE HORIZONTAL LINE DISAPPEARS. IT CONTINUES UNTIL IT REACHES THE BASE LINE OF THE SCREEN WHEN A SECOND DIAGONAL LINE BEGINS TO DECAY DOWNWARD FROM THE TOP MIDDLE SECTION OF THE SCREEN AND CONTINUES UNTIL IT REACHES THE BASE LINE OF THE SCREEN. THE END RESULT SHOULD BE TWO LARGE X'S FILLING UP THE ENTIRE SCREEN (XX). IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.6 DISPLAY A STEPPING HISTOGRAM LINE ON GRAPH 0

THE CORRECT VISUAL DISPLAY WILL BEGIN WITH THE APPEARANCE OF A HORIZONTAL LINE AT BASE ZERO ON THE SCREEN. THEN A LINE HALFWAY UP THE SCREEN WILL BEGIN TO GROW FROM THE LEFT SIDE OF THE SCREEN WITH ALL THE AREA BETWEEN THE TWO LINES SHADED. THIS SHADED AREA WILL CONTINUE TO GROW UNTIL IT REACHES THE FAR RIGHT SIDE OF THE SCREEN. THEN A LINE WHICH BISECTS THE SHADED AREA BEGINS TO GROW FROM THE LEFT SIDE. AS THIS LINE GROWS IT REMOVES THE SHADED AREA ABOVE IT, CONTINUING UNTIL IT REACHES THE RIGHT SIDE OF THE SCREEN. THE SHADED AREA IS THEN CUT IN HALF AGAIN AND AGAIN UNTIL A SINGLE HORIZONTAL LINE REMAINS.

9.7 DISPLAY A STEPPING HISTOGRAM LINE ON GRAPH 1

SAME AS 9.6 EXCEPT GRAPH 1 IS ENABLED.

9.8 HISTOGRAM ON GRAPH 0 AND 1

THE CORRECT VISUAL DISPLAY WILL FOLLOW THE SAME PATTERN AS THE VISUAL DISPLAY FOR GRAPH 0 AND 1, EXCEPT THAT AS EACH DIAGONAL LINE GROWS, A TRIANGULAR SHADED AREA GROWS UNDER IT. THE FINAL RESULT SHOULD CONSIST OF FOUR OVERLAPPING RIGHT TRIANGLES, TWO WITH THE RIGHT ANGLE ON THE RIGHT BOTTOM OF THE SCREEN (MADE BY THE DIAGONAL LINES WHICH STARTED FROM THE BOTTOM LEFT) AND TWO WITH THE RIGHT ANGLE ON THE LEFT BOTTOM OF THE SCREEN (MADE BY THE DIAGONAL LINES WHICH STARTED FROM THE TOP LEFT). IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.9 CURSORS ON GRAPH 0

THE CORRECT VISUAL DISPLAY WILL BEGIN WITH A SINGLE HORIZONTAL LINE EXTENDING THE ENTIRE LENGTH OF THE SCREEN PLACED AT BASE ZERO. A DIAGONAL LINE WILL THEN BEGIN TO GROW FROM THE BOTTOM LEFT CORNER AS THE BASE HORIZONTAL LINE IS REMOVED. IT CONTINUES TO GROW UNTIL IT REACHES THE TOP OF THE SCREEN WHEN A SECOND DIAGONAL LINE BEGINS TO GROW FROM THE BOTTOM OF THE MIDDLE OF THE SCREEN. THIS LINE CONTINUES TO GROW AS THE BASE HORIZONTAL LINE CONTINUES TO BE REMOVED. WHEN EACH DIAGONAL LINE REACHES THE TOP OF THE SCREEN A SQUARE OF CURSORS GROWS AT THE BASE OF THE FIRST DIAGONAL LINE. IT IS FOLLOWED BY ANOTHER SQUARE, EVENTUALLY GIVING THE APPEARANCE OF A STAIRCASE. THIS PROCEDURE IS REPEATED ON THE SECOND DIAGONAL LINE AND WHEN THE LAST SQUARE IS DONE, THE ENTIRE PROCEDURE IS REVERSED. EACH SQUARE IS SUCCESSIVELY REMOVED STARTING AT THE TOP OF THE SECOND DIAGONAL LINE, CONTINUING DOWN IT, THEN STARTING AT THE TOP OF THE FIRST DIAGONAL LINE AND GOING DOWN IT. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.10 CURSORS ON GRAPH 1

THE CORRECT VISUAL DISPLAY WILL BE ALMOST IDENTICAL TO THAT OF THE CURSORS ON GRAPH 0. THE ONLY DIFFERENCE IS THAT THE TWO DIAGONAL LINES BEGIN AT THE TOP LEFT OF THE SCREEN AND GO DOWN TOWARDS THE RIGHT. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.11 STARTING COORDINATE ON GRAPH 0

THE CORRECT VISUAL DISPLAY WILL BEGIN WITH A SINGLE HORIZONTAL LINE EXTENDING THE ENTIRE WIDTH OF THE SCREEN PLACED AT BASE ZERO. A DIAGONAL LINE WILL THEN BEGIN TO DECAY FROM THE TOP LEFT CORNER AS THE HORIZONTAL LINE DISAPPEARS. IT CONTINUES TO DECAY UNTIL IT REACHES THE BOTTOM OF THE SCREEN WHEN A SECOND DIAGONAL LINE BEGINS TO DECAY FROM THE TOP OF THE MIDDLE OF THE SCREEN. THIS LINE CONTINUES TO DECAY AS THE HORIZONTAL LINE CONTINUES TO DISAPPEAR. WHEN THE SECOND DIAGONAL LINE REACHES THE BOTTOM OF THE SCREEN, A SMALL SECTION OF THE HORIZONTAL LINE SHOULD STILL BE VISIBLE. AT THIS POINT A DOTTED SINE CURVE BEGINS TO APPEAR FROM THE RIGHT EDGE. AS IT GROWS UPWARD THE DIAGONAL LINE IS REMOVED. AS THE SINE CURVE REACHES ITS PEAK, THE SECOND DIAGONAL LINE BEGINS TO DISAPPEAR. THE LINE CONTINUES TO DISAPPEAR AS THE SINE CURVE ROUNDS THE PEAK AND STARTS DOWNWARD. THEN THE SINE CURVE GROWS UPWARD AGAIN AND THE FIRST DIAGONAL LINE BEGINS TO DISAPPEAR AS THE CURVE REACHES ITS PEAK. THE SINE CURVE CONTINUES TO GROW UNTIL FOUR COMPLETE CYCLES HAVE BEEN FORMED. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

9.12 STARTING COORDINATE ON GRAPH 1

SAME AS 9.11 EXCEPT GRAPH 1 IS ENABLED.

9.13 VT55 ADJUSTMENT PATTERN

THE CORRECT VISUAL DISPLAY CONSISTS OF TWELVE ROWS OF THE LETTER "H" WITH A BLANK LINE SEPARATING EACH OF THE ROWS. THEN TWELVE HORIZONTAL LINES ARE DISPLAYED, STARTING AT THE BOTTOM AND OVERLAYING THE ROWS OF H'S TOUCHING THE BOTTOM OF THE H'S. THEN FORTY-ONE VERTICAL LINES ARE DISPLAYED, RESULTING IN A CHECKERBOARD OVER THE TWELVE ROWS OF H'S. THE FIRST VERTICAL LINE IS PLACED THROUGH THE MIDDLE OF THE FIRST ROW OF H'S. EACH SUCCESSIVE VERTICAL LINE IS 13 POINTS TO THE RIGHT OF THE PREVIOUS LINE. THE VERY LAST LINE IS EXACTLY ON TOP OF THE RIGHT SIDE OF THE LAST ROW OF H'S. IF "COPIER" TESTING IS ENABLE, THE SCREEN WILL NOW BE COPIED.

```
%
.TITLE  MAINDEC-11-DZVTD-B
;*COPYRIGHT (C) 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY RAYMOND SHOOP
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C0),MAR 21, 1976.
;*
```

```
                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
```

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

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
176500                  ABASE=176500
```

```
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        12            INHIBIT SUB-TEST DELAY'S
                                ;*        10            ENABLE "SAVE COPIER PAPER MODE"
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                .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  001346         JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001402         JMP     RBEGIN          ;JUMP TO RESTART ADDRESS
000210  000137  001412         JMP     BEGIN2          ;JUMP TO ADJUSTMENT PATTERN


                                .SBTTL  ACT11 HOOKS

                                ;;*************************************************************
                                ;HOOKS REQUIRED BY ACT11
                        000214          $SVPC=.                 ;SAVE PC
                        000046          .=46
000046  006776                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
                        000052          .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
                        000214          .=$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         $MB9AD: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000300         $TSTM:  .WORD   300     ;;RUN TIM OF LONGEST TEST
```

```
MAINDEC-11-DZVTD-B     MACY11 27(732)  25-SEP-76  10:10  PAGE 15
DZVTDB.SAC     APT PARAMETER BLOCK

   594  001006  000300                  SPASTM: .WORD   300     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
   595  001010  000300                  $UNITM: .WORD   300     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
   596  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                          $CNTAG:                 ;;START OF COMMON TAGS
001100  000000                  $TIMES: .WORD   0       ;;CONTAINS THE TEST NUMBER
001102  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001103  000                     $ICNT:  .BYTE   0       ;;CONTAINS SUBTEST ITERATION COUNT
001104  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001106  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001110  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001112  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001113  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001114  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001116  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001120  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001122  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001124  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001126  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001130  000000                          .WORD   0
001132  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001133  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001134  000000                          .WORD   0
001136  177570                  $SWR:   .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001140  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
                                TKS=    177560         ;;TTY KBD STATUS
                                TKB=    177562         ;;TTY KBD BUFFER
                                TPS=    177564         ;;TTY PRINTER STATUS REG. ADDRESS
                                TPB=    177566         ;;TTY PRINTER BUFFER REG. ADDRESS
001144  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001145  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001146  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001147  000                     $TTFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001150  000000                  $SREGAD: .WORD  0       ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($SREG0) WAS OBTAINED
001152  000000                  $SREG0:  .WORD  0       ;;CONTAINS ((SREGAD)+0)
001154  000000                  $SREG1:  .WORD  0       ;;CONTAINS ((SREGAD)+2)
001156  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001170  077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
001171  015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001172  000012                  $LF:    .ASCII  <12>    ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

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

```
                                        $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
                                        $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
001224  000                     $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001225  000                     $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                                                MEM.TYPE BYTE   --  (HIGH BYTE)
                                                                        900 NSEC CORE=001
                                                                        300 NSEC BIPOLAR=002
                                                                        500 NSEC MOS=003
001226  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001230  000                     $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001231  000                     $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001232  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001234  000                     $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001235  000                     $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001236  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001240  000                     $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001241  000                     $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001242  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001244  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001246  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001250  174500                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001252  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001254  000000                  $CON1:  .WORD   ACON1   ;;CONTROLLER DESCRIPTION WORD#1
001256                          $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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                        ;*NOTE1:        IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
                                        ;*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                                  SERRTB:


                                        ;ITEM   1
001256  011612                                  EM1             ;INCORRECT OR UNEXPECTED INPUT CHARACTER
001260  011745                                  DH1             ;ERRPC  VTNOW   TSTNUM  EXPT    RCVD
001262  012700                                  DT1             ;$ERRPC VTNOW   TSTNUM  $GDDAT  $BDDAT
001264  000000                                  0

                                        ;ITEM   2
001266  011563                                  EM2             ;NO HOST INPUT FLAG RECEIVED
001270  011714                                  DH1             ;ERRPC  VTNOW   TSTNUM
001272  012670                                  DT1             ;$ERRPC VTNOW   TSTNUM
001274  000000                                  0

                                        ;ITEM   3
001276  011655                                  EM3             ;INVALID BUS ADDRESS,  TRY AGAIN
001300  000000                                  0
001302  000000                                  0
001304  000000                                  0
```

```
MACRO11-DZVTD-A     MACY11 27(732)  25-SEP-76  10:10  PAGE 19
DZVTU8.SAC      ERROR POINTER TABLE

                                ;VT-55 EQUALITIES
                
000354                          MAXHOZ=236.             ;MAX. HORIZ LINE COUNT
000400                          MAXHZC=256.
001000                          MAXVRT=512.             ;MAX. VERTICAL LINE COUNT

002000                          ADDLIN=BIT10

000033                          ESC=    33

000100                          LNO=    100             ;NOP

000101                          LDE0=   101             ;LOAD ENABLE REG. 0
000111                          LDE1=   111             ;LOAD ENABLE REG. 1

000102                          LDG0=   102             ;LOAD GRAPH 0
000112                          LDG1=   112             ;LOAD GRAPH 1

000103                          LDC0=   103             ;LOAD CURSOR ON GRAPH 0
000113                          LDC1=   113             ;LOAD CURSOR ON GRAPH 1

000104                          LHV0=   104             ;LOAD HORIZONTAL LINE
000114                          LHV1=   114             ;LOAD VERTICAL LINE

000110                          LSC=    110             ;LOAD STARTING COORDINATE

000135                          COPY=   135             ;COPY SCREEN COMMAND
```

```

001306  177560                          FIRST:  177560                  ;FIRST DEVICE ADDRESS OF SEQUIENTAL DL-11-A/B TYPE DEVIC
                                                                        ;DEFAULT TO THE CONSOLE ADDRESS
001310  000000                          LAST:   0                       ;LAST DEVICE ADDRESS OF DL-11-A/B TYPE
001312  177560                          VTNOW:  177560                  ;CURRENT DEVICE BUSS ADDRESS
001314  000000                          TSTNUM: 0                       ;ERROR PATTERN

001316  000100                          TIMEO:  100                     ;CHARACTER FLAG TIMEOUT CONSTANT
001320  000012                          SUBTST: 10.                     ;SUBTEST DELAY CONSTANT
001322  000000                          WFTEST: 0
001324  177560                          VTIS:   177560                  ;DEVICE ADDRESSES
001326  177562                          VTIB:   177562                  ;IN DATA
001330  177564                          VTOS:   177564                  ;OUT STAT
001332  177566                          VTOB:   177566                  ;OUT DATA
001334  000000                          SAVE4:  0

001336  022626                          BUSSTR: CMP     (SP)+,(SP)+     ;POP STACK
001340  104003                                  ERROR   3               ;INVALID BUS ADDRESS
001342  000240                                  NOP
001344  000240                                  NOP

001346  012737  002064  002062          BEGIN:  MOV     #TST1,WHERE     ;STARTING ACCEPTANCE TEST ADDRESS
001354  005037  001322                          CLR     WFTEST
001360  005037  001334                          CLR     SAVE4
001364  012737  001336  000004                  MOV     #BUSSTR,@#ERRVEC
001372  012737  000340  000006                  MOV     #340,@#ERRVEC+2
001400  000412                                  BR      GINA
001402  012737  002064  002062          RBEGIN: MOV     #TST1,WHERE
001410  000403                                  BR      GIN
001412  012737  000001  001322          BEGIN2: MOV     #1,WFTEST
001420  012737  000001  001334          GIN:    MOV     #1,SAVE4
001426  000005                          GINA:   RESET
```

```
                                                .SBTTL  INITIALIZE THE COMMON TAGS
                                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001430  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001434  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001436  022706  001140                          CMP     #$WR,R6 ;;DONE?
001442  001374                                  BNE     .-6             ;;LOOP BACK IF NO
001444  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                                ;;INITIALIZE A FEW VECTORS
001450  012737  013644  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001456  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001464  012737  014306  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001472  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001500  012737  015344  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001506  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
001514  012737  015410  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001522  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001530  013737  006756  006750                  MOV     $ENDCT,$LOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001536  005037  001166                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001542  112737  000001  001115                  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001550  012737  001550  001106                  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001556  012737  001556  001110                  MOV     #.,$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.
001564  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001570  012737  001624  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001576  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001604  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001612  022777  177777  177320                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001620  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
001622  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
001624  012716  001632          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001630  000002                          RTI
001632  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001640  012737  000174  001142          MOV     #DISPREG,DISPLAY
001646  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001652  005037  001202                          CLR     $PASS           ;;CLEAR PASS COUNT
001656  132737  000200  001215                  BITB    #APTSIZE,SENVM  ;;TEST USER SIZE UNDER APT
001664  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
001666  012737  001216  001140                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
001674                          67$:
001674  013737  001250  001306          MOV     $BASE,FIRST             ;LOAD "APT" 'S  BASE ADDRESS
001702  005037  010544                  CLR     LOOP
001706  012737  013630  000020          MOV     #MSCOPE,@#IOTVEC
001714  005737  001334          TIST:   TST     SAVE4                   ;TEST FLAG
001720  001002                          BNE     11$                     ;BR IF NON-ZERO
001722  104400                          TYPE
001724  011220                          TITLE
```

```
                                        ;NOW TEST IF RUNNING ON NON-SWR CPU
001726  022737  000176  001140  11$:    CMP     @#SWREG,SWR             ;TEST IF NON-SWITCH REGISTER
001734  001403                          BEQ     2$                      ;BR IF YES - AND ASK ADDRESS
001736  105777  177176                  TSTB    @SWR
001742  100022                          BPL     RSTRTA                  ;BR IF CLEARED
001744  104400                  2$:     TYPE
001746  011466                          WHATO                           ;FIND OUT THE DEVICE ADDRESS
001750  104410                          RDOCT
001752  012637  001306                  MOV     (SP)+,FIRST             ;SAVE THE ADDRESS
001756  022737  160000  001306          CMP     #160000,FIRST
001764  101360                          BHI     11$                     ;BR IF INVALID
001766  005777  177314                  TST     @FIRST                  ;TEST IF VALID
001772  005037  001310                  CLR     LAST
001776  012737  000006  000004          MOV     #6,@#4
002004  005037  000006                  CLR     @#6
002010  013737  001306  001312  RSTRTA: MOV     FIRST,VTNOW             ;LOAD INITIAL DEVICE ADDRESS

002016  012700  001324          RSTRT:  MOV     #VTIS,R0                ;LOAD POINTER
002022  013720  001312                  MOV     VTNOW,(R0)+             ;LOAD INPUT STAT
002026  013710  001312                  MOV     VTNOW,(R0)
002032  062720  000002                  ADD     #2,(R0)+                ;LOAD INPUT BUFFER
002036  013710  001312                  MOV     VTNOW,(R0)
002042  062720  000004                  ADD     #4,(R0)+                ;LOAD OUTPUT STAT
002046  013710  001312                  MOV     VTNOW,(R0)
002052  062720  000006                  ADD     #6,(R0)+                ;LOAD OUTPUT BUFFER
002056  000177  000000                  JMP     @#WHERE                 ;JUMP TO STARTING ADDRESS
002062  002064                  WHERE:  TST1
```

```
                                ;;*****************************************************************
                                ;*TEST 1        A       DISPLAY A GROWING HORIZONTAL LINES
                                ;;*****************************************************************
002064  000004                  TST1:   SCOPE
002066  004537  011070                  JSR     R5,AMSG                 ;DISPLAY HEADER
002072  012013                          DAHL
                                
002074  012700  015562                  MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
002100  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
002104  112720  000061                  MOVB    #'1,(R0)+               ;LOAD "01" ENTER CODE
002110  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002114  112720  000040                  MOVB    #BITS,(R0)+             ;DISABLE DISPLAY
002120  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002124  112720  000050                  MOVB    #BITS!BIT4,(R0)+        ;CLEAR GRAPH, LILES, AND CURSORS
002130  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002134  112720  000041                  MOVB    #BITS!BIT0,(R0)+        ;LOAD DISPLAY ENABLE
002140  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002144  112720  000041                  MOVB    #BITS!BIT0,(R0)+        ;LOAD DISPLAY HORIZ. LINE
002150  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
002154  112720  000104                  MOVB    #LHVC,(R0)+             ;LOAD LOAD HORIZONTAL LINE
002160  012737  002000  007772          MOV     #ADDLIN,BASE            ;LOAD THE STARTING DATA VALUE
002166  004737  007776          1$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002172  010120                          MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002174  005237  007772                  INC     BASE                    ;UPDATE THE DATA
002200  022737  002354  007772          CMP     #ADDLIN!MAXHOZ,BASE     ;COMPARE TO LAST DATA LINE
002206  001367                          BNE     1$                      ;LOOP UNTIL DONE
002210  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
002214  112720  000135                  MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
002220  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002224  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002230  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002234  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002240  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002244  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002250  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002254  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002260  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002264  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002270  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002274  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
        112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
        112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002310  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002314  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002320  105020                          CLRB    (R0)+                   ;LOAD TERM
002322  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
002326  004737  010742                  JSR     PC,DELAY
                                
```

```
911
912  002332  012700  015562                MOV     #BUFFER,R0          ;LOAD BUFFER POINTER
913  002336  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP
914  002342  112720  000104                MOVB    #LHVD,(R0)+         ;LOAD HORIZONTAL LINE MODE AGAIN
915
916  002346  012737  000000  007772        MOV     #0,BASE             ;LOAD STARTING DATA VALUE TO REMOVE THE LINE
917  002354  004737  007776          2$:   JSR     PC,SHUFF            ;SHUFFEL THE DATA INTO VT-55 FORMAT
918  002360  010120                        MOV     R1,(R0)+            ;SAVE THE LSB,MSB BYTE
919  002362  005237  007772                INC     BASE                ;UPDATE THE DATA
920  002366  022737  000354  007772        CMP     #MAXH02,BASE        ;COMPARE TO LAST DATA LINE
921  002374  001367                        BNE     2$                  ;BR TILL DONE
922
923  002376  112720  000033                MOVB    #ESC,(R0)+          ;LOAD "ESC" CODE
924  002402  112720  000135                MOVB    #COPY,(R0)+         ;LOAD COPY SCREEN
925  002406  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
926  002412  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
927  002416  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
928  002422  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
929  002426  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
930  002432  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
931  002436  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
932  002442  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
933  002446  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
934  002452  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
935  002456  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
936  002462  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
937  002466  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
938  002472  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
939  002476  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
940  002502  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
941  002506  105020                        CLRB    (R0)+               ;LOAD TERMINATOR
942  002510  004737  010074                JSR     PC,XPRNT            ;DISPLAY
943  002514  004737  010742                JSR     PC,DELAY
944
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2        B       DISPLAY A GROWING VERTICAL LINES
                                        ;;*****************************************************************
002520  000004                  TST2:   SCOPE
002522  004537  011070                  JSR     R5,AMSG                 ;DISPLAY HEADER
002526  012045                          DAVL
002530  012700  015562                  MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
002534  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
002540  112720  000061                  MOVB    #61,(R0)+               ;LOAD "01" ENTER CODE
002544  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002550  112720  000040                  MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
002554  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002560  112720  000060                  MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LILES, AND CURSORS
002564  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002570  112720  000041                  MOVB    #BIT5!BIT0,(R0)+        ;LOAD DISPLAY ENABLE
002574  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002600  112720  000042                  MOVB    #BIT5!BIT1,(R0)+        ;LOAD DISPLAY VERTICAL LINE
002604  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
002610  112720  000114                  MOVB    #LHV1,(R0)+             ;LOAD VERTICAL LINE
002614  012737  002777  007772          MOV     #ADDLIN!MAXVRT-1,BASE   ;LOAD THE STARTING DATA VALUE
002622  004737  007776          1$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002626  010120                          MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002630  005337  007772                  DEC     BASE                    ;UPDATE THE DATA
002634  022737  002000  007772          CMP     #ADDLIN,BASE            ;COMPARE TO LAST DATA LINE
002642  001367                          BNE     1$                      ;LOOP UNTIL DONE
002644  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
002650  112720  000135                  MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
002654  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002660  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002664  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002670  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002674  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002700  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002704  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002710  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002714  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002720  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002724  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002730  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002734  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002740  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002744  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002750  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
002754  105020                          CLRB    (R0)+                   ;LOAD TERM
002756  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
002762  004737  010742                  JSR     PC,DELAY
```

```
  992  002766  012700  015562                MOV     #BUFFER,R0
  993  002772  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP
  994  002776  112720  000114                MOVB    #LHV1,(R0)+     ;LOAD VERTIACL LINE MODE AGAIN
  995
  996  003002  012737  001000  007772        MOV     #MAXVRT,BASE    ;LOAD STARTING DATA VALUE TO REMOVE THE LINE
  997  003010  004737  007776        2$:     JSR     PC,SHUFF        ;SHUFFEL THE DATA INTO VT-55 FORMAT
  998  003014  010120                        MOV     R1,(R0)+        ;SAVE THE LSB MSB BYTE
  999  003016  005337  007772                DEC     BASE            ;UPDATE THE DA T
 1000  003022  001372                        BNE     2$              ;BR TILL DONE
 1001
 1002  003024  112720  000033                MOVB    #ESC,(R0)+      ;LOAD "ESC" CODE
 1003  003030  112720  000135                MOVB    #COPY,(R0)+     ;LOAD COPY SCREEN
 1004  003034  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1005  003040  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1006  003044  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1007  003050  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1008  003054  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1009  003060  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1010  003064  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1011  003070  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1012  003074  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1013  003100  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1014  003104  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1015  003110  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1016  003114  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1017  003120  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1018  003124  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1019  003130  112720  000100                MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1020  003134  105020                        CLRB    (R0)+           ;LOAD TERMINATOR
 1021  003136  004737  010074                JSR     PC,XPRNT        ;DISPLAY
 1022  003142  004737  010742                JSR     PC,DELAY
 1023
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3        C       GRAPH 0 DISPLAY A STEPPING HORIZONTAL LINE
                                        ;;*****************************************************************
003146  000004                  TST3:   SCOPE
003150  004537  011070                  JSR     R5,AMSG                 ;DISPLAY HEADER
003154  012076                          SHLO
003156  012700  015562                  MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
003162  112720  000033                  MOVB    #ESC,(R0)+              ;ENTER VT-55 MODE
003166  112720  000061                  MOVB    #61,(R0)+
003172  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
003176  112720  000040                  MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
003202  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
003206  112720  000060                  MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LILES, AND CURSORS
003212  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
003214  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
003220  012737  000200  003400          MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
003226  012700  015562          2$:     MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
003232  112720  000033                  MOVB    #ESC,(R0)+              ;ENTER VT-55 FORMAT
003236  112720  000061                  MOVB    #61,(R0)+               ;LOAD ENTER '01' CODE
003242  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
003246  112720  000040                  MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
003252  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
003256  112720  000060                  MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LINES AND CURSORS

003262  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
003266  112720  000043                  MOVB    #BIT5!BIT1!BIT0,(R0)+   ;LOAD DISPLAY ENABLE AND GRAPH 0 ON
003272  112720  000100                  MOVB    #LN0,(R0)+              ;NOP
003276  112720  000110                  MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
003302  012737  000000  007772          MOV     #0,BASE                 ;GET BASE LINE
003310  004737  007776                  JSR     PC,SHUFF                ;CONVERT
003314  010120                          MOV     R1,(R0)+                ;SAVE COORD.
003316  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP
003322  112720  000102                  MOVB    #LDG0,(R0)+             ;LOAD "LOAD GRAPH"
003326  013737  003400  007772          MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
003334  004737  007776                  JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
003340  012737  001000  003402          MOV     #MAXVRT,101$            ;LOAD COUNTER
003346  010120                  1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
003350  005337  003402                  DEC     101$                    ;DONE FULL GRAPH
003354  001374                          BNE     1$

003356  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
003360  004737  010074                  JSR     PC,XPRNT                ;DISPLAY
003364  006037  003400                  ROR     100$                    ;CHANGE DATA VALUE
003370  001316                          BNE     2$                      ;NO
003372  004737  010742                  JSR     PC,DELAY
003376  000402                          BR      TST4            ;;NEXT TEST

003400  000000                  100$:   0
003402  000000                  101$:   0
```

```
1075                                    ;;*****************************************************************
1076                                    ;*TEST 4        D       GRAPH 1 DISPLAY A STEPPING HORIZONTAL LINE
1077                                    ;;*****************************************************************
1078  003404  000004                    TST4:   SCOPE
1079  003406  004537  011070                    JSR     R5,AMSG                 ;DISPLAY HEADER
1080  003412  012154                            SHL1
1081  003414  012700  015562                    MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
1082  003420  112720  000033                    MOVB    #ESC,(R0)+              ;ENTER VT-55 MODE
1083  003424  112720  000061                    MOVB    #61,(R0)+
1084  003430  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1085  003434  112720  000040                    MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1086  003440  112720  000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1087  003444  112720  000060                    MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LILES, AND CURSORS
1088  003450  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1089  003452  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1090
1091  003456  012737  000200  003636            MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
1092  003464  012700  015562            2$:     MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
1093  003470  112720  000033                    MOVB    #ESC,(R0)+              ;ENTER VT55  MODE
1094  003474  112720  000061                    MOVB    #61,(R0)+               ;LOAD ENTER "01" CODE
1095  003500  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1096  003504  112720  000040                    MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1097  003510  112720  000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1098  003514  112720  000060                    MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH,LINES AND CURSORS
1099  003520  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1100  003524  112720  000045                    MOVB    #BIT5!BIT2!BIT0,(R0)+   ;LOAD DISPLAY ENABLE AND GRAPH 1 ON
1101  003530  112720  000100                    MOVB    #LNO,(R0)+              ;NOP
1102  003534  112720  000110                    MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
1103  003540  012737  000000  007772            MOV     #0,BASE                 ;GET BASE LINE
1104  003546  004737  007776                    JSR     PC,SHUFF                ;CONVERT
1105  003552  010120                            MOV     R1,(R0)+                ;SAVE COORD.
1106  003554  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP
1107  003560  112720  000112                    MOVB    #LDG1,(R0)+             ;LOAD "LOAD GRAPH"
1108  003564  013737  003636  007772            MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
1109  003572  004737  007776                    JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
1110  003576  012737  001000  003640            MOV     #MAXVRT,101$            ;LOAD COUNTER
1111  003604  010120                    1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
1112  003606  005337  003640                    DEC     101$                    ;DONE FULL GRAPH
1113  003612  001374                            BNE     1$
1114
1115
1116  003614  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1117  003616  004737  010074                    JSR     PC,XPRNT                ;DISPLAY
1118  003622  006037  003636                    ROR     100$
1119  003626  001316                            BNE     2$                      ;NO
1120  003630  004737  010742                    JSR     PC,DELAY
1121  003634  000402                            BR      TST5            ;;NEXT TEST
1122
1123  003636  000000                    100$:   0
1124  003640  000000                    101$:   0
```

```
                                ;;*****************************************************************
                                ;*TEST 5        E       GRAPH 0 AND 1
                                ;;*****************************************************************
003642  000004                  TST5:   SCOPE
003644  004537  011070                  JSR     R5,AMSG                 ;DISPLAY HEADER
003650  012232                          GRDA1
003652  012700  015562                  MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
003656  112720  000033                  MOVB    #ESC,(R0)+              ;ENTER VT55 MODE
003662  112720  000061                  MOVB    #61,(R0)+               ;ENTER "01" CODE
003666  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
003672  112720  000040                  MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
003676  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
003702  112720  000060                  MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LINES AND CURSORS
003706  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
003710  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
003714  004537  007242                  JSR     R5,UPDWN                ;LOAD DATA PATTERN
003720     002     102                  .BYTE   BIT1,LDG0               ;GRAPH 0 INC. PAT.
003722  000000                          .WORD   0
003724  004737  010074                  JSR     PC,XPRNT
003730  004537  007242                  JSR     R5,UPDWN                ;LOAD DATA PATTERN
003734     006     112                  .BYTE   BIT2!BIT1,LDG1          ;GRAPH 0 DEC. PAT.
003736  100354                          .WORD   BIT15!MAXHOZ                    ;
003740  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
003744  112720  000143                  MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
003750  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
003754  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
003760  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
003764  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
003770  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
003774  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004000  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004004  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004010  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004014  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004020  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004024  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004030  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004034  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004040  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004044  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
004050  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
004052  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
004056  004737  010742                  JSR     PC,DELAY
```

```
                                        ;;*****************************************************************
                                        ;*TEST 6        F       GRAPH 0 DISPLAY A STEPPING HISTOGRAM LINE
                                        ;;*****************************************************************
004062  000004                  TST6:   SCOPE
004064  004537  011070                  JSR     R5,AMSG                 ;DISPLAY HEADER
004070  012276                          SHGLO
004072  012700  015562                  MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
004076  112720  000033                  MOVB    #ESC,(R0)+              ;ENTER VT55 MODE
004102  112720  000061                  MOVB    #61,(R0)+               ;ENTER "01" CODE
004106  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
004112  112720  000040                  MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
004116  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
004122  112720  000060                  MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LINES AND CURSORS
004126  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
004130  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
004134  012737  000200  004274          MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
004142  012700  015562          2$:     MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
004146  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
004152  112720  000061                  MOVB    #61,(R0)+               ;LOAD "01" ENTER CODE
004156  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
004162  112720  000053                  MOVB    #BIT5!BIT3!BIT1!BIT0,(R0)+      ;LOAD DISP. ENABLE , GRAPH 0, HISTO 1 ON
004166  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP
004172  112720  000110                  MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
004176  012737  000000  007772          MOV     #0,BASE                 ;GET BASE LINE
004204  004737  007776                  JSR     PC,SHUFF                ;CONVERT
004210  010120                          MOV     R1,(R0)+                ;SAVE COORD.
004212  112720  000100                  MOVB    #LN0,(R0)+              ;LOAD NOP
004216  112720  000102                  MOVB    #LDG0,(R0)+             ;LOAD "LOAD GRAPH"
004222  013737  004274  007772          MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
004230  004737  007776                  JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
004234  012737  001000  004276          MOV     #LNXVRT,101$            ;LOAD COUNTER
004242  010120                  1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
004244  005337  004276                  DEC     101$                    ;DONE FULL GRAPH
004250  001374                          BNE     1$
004252  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
004254  004737  010074                  JSR     PC,XPRNT                ;DISPLAY
004260  006037  004274                  ROR     100$                    ;CHANGE DATA VALUE
004264  001326                          BNE     2$                      ;NO
004266  004737  010742                  JSR     PC,DELAY
004272  000402                          BR      TST7            ;;NEXT TEST

004274  000000          100$:   0
004276  000000          101$:   0
```

```
1219                                    ;;#################################################################
1220                                    ;#TEST 7        G       GRAPH 1 DISPLAY A STEPPING HISTOGRAM LINE
1221                                    ;;#################################################################
1222  004300  000004                    TST7:   SCOPE
1223  004302  004537  011070                    JSR     R5,AMSG                 ;DISPLAY HEADER
1224  004306  012347                            SHGL1
1225  004310  012700  015562                    MOV     #BUFFER,R0              ;LOAD OUPTUT BUFFER POINTER
1226  004314  112720  000033                    MOVB    #ESC,(R0)+              ;ENTER VT55 MODE
1227  004320  112720  000061                    MOVB    #61,(R0)+               ;ENTER "01"CODE
1228  004324  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1229  004330  112720  000040                    MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1230  004334  112720  000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1231  004340  112720  000060                    MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRPAH,LINES AND CUROSRS
1232  004344  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1233  004346  004737  010074                    JSR     PC,XPRNT                ;EXECUTRE
1234
1235  004352  012737  000200  004512            MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
1236
1237  004360  012700  015562          2$:       MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
1238  004364  112720  000033                    MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
1239  004370  112720  000061                    MOVB    #61,(R0)+               ;LOAD "01" ENTER CODE
1240  004374  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1241  004400  112720  000065                    MOVB    #BIT5!BIT4!BIT2!BIT0,(R0)+      ;LOAD DISPLAY ENABLE AND GRAPH 1 ON
1242  004404  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP
1243  004410  112720  000110                    MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
1244  004414  012737  000000  007772            MOV     #0,BASE                 ;GET BASE LINE
1245  004422  004737  007776                    JSR     PC,SHUFF                ;CONVERT
1246  004426  010120                            MOV     R1,(R0)+                ;SAVE COORD.
1247  004430  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP
1248  004434  112720  000112                    MOVB    #LDG1,(R0)+             ;LOAD "LOAD GRAPH"
1249  004440  013737  004512  007772            MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
1250  004446  004737  007776                    JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
1251  004452  012737  001000  004514            MOV     #MAXVAT,101$            ;LOAD COUNTER
1252  004460  010120                  1$:       MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
1253  004462  005337  004514                    DEC     101$                    ;DONE FULL GRAPH
1254  004466  001374                            BNE     1$
1255
1256
1257  004470  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1258  004472  004737  010074                    JSR     PC,XPRNT                ;DISPLAY
1259  004476  006037  004512                    ROR     100$
1260  004502  001326                            BNE     2$                      ;NO
1261  004504  004737  010742                    JSR     PC,DELAY
1262  004510  000402                            BR      TST10           ;;NEXT TEST
1263
1264  004512  000000                  100$:     0
1265  004514  000000                  101$:     0
1266
```

```
1267
1268                                    ;;*****************************************************************
1269                                    ;*TEST 10       H       HISTOGRAM ON GRAPH 0 AND 1
1270                                    ;;*****************************************************************
1271  004516  000004                    TST10:  SCOPE
1272  004520  004537  011070                    JSR     R5,AMSG                 ;DISPLAY HEADER
1273  004524  012420                            HGGA1
1274  004526  012700  015562                    MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
1275  004532  112720  000033                    MOVB    #ESC,(R0)+              ;LOAD VT55 MODE
1276  004536  112720  000061                    MOVB    #61,(R0)+               ;ENTER "01" CODE
1277  004542  112720  000101                    MOVB    #LDG0,(R0)+             ;LOAD ENABLE 0
1278  004546  112720  000040                    MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1279  004552  112720  000111                    MOVB    #LDG1,(R0)+             ;LOAD ENABLE 1
1280  004556  112720  000060                    MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LINES AND CURSORS
1281  004562  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1282  004564  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1283
1284  004570  004537  007242                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
1285  004574     012     102                    .BYTE   BIT3!BIT1,LDG0          ;GRAPH 0 INC. PAT.
1286  004576  000000                            .WORD   0
1287  004600  004737  010074                    JSR     PC,XPRNT
1288
1289  004604  004537  007242                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
1290  004610     036     112                    .BYTE   BIT4!BIT3!BIT2!BIT1,LDG1        ;GRAPH 1
1291  004612  100354                            .WORD   BIT15!MAXHOZ
1292
1293  004614  112720  000033                    MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
1294  004620  112720  000135                    MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
1295  004624  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1296  004630  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1297  004634  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1298  004640  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1299  004644  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1300  004650  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1301  004654  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1302  004660  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1303  004664  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1304  004670  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1305  004674  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1306  004700  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1307  004704  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1308  004710  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1309  004714  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1310  004720  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1311  004724  105020                            CLRB    (R0)+                   ;TERMINATOR
1312  004726  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1313  004732  004737  010742                    JSR     PC,DELAY
1314
```

```
1315                                    ;;*****************************************************************
1316                                    ;;*TEST 11       I       CURSORS ON GRAPH 0
1317                                    ;;*****************************************************************
1318  004736  000004                    TST11:  SCOPE
1319  004740  004537  011070                    JSR     R5,AMSG                 ;DISPLAY HEADER
1320  004744  012441                            CURGRO
1321  004746  012700  015562                    MOV     #BUFFER,R0              ;LOAD OUPTUT BUFFER POINTER
1322  004752  112720  000033                    MOVB    #ESC,(R0)+              ;LOAD VT55 MODE
1323  004756  112720  000061                    MOVB    #61,(R0)+               ;ENTER "01" CODE
1324  004762  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1325  004766  112720  000040                    MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1326  004772  112720  000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1327  004776  112720  000060                    MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LINES AND CURSORS
1328  005002  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1329  005004  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1330
1331  005010  004537  007242                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
1332  005014     002     102                    .BYTE   BIT1,LDG0               ;GRAPH 0
1333  005016  000000                            .WORD   0                       ;DATA TO BE LOADED
1334  005020  004737  010074                    JSR     PC,XPRNT                ;EXECUTE IT
1335
1336  005024  004537  007574                    JSR     R5,CURSOR               ;ENABLE CURSORS
1337  005030     002     103                    .BYTE   BIT1,LDC0
1338  005032  002000                            .WORD   ADDLIN                  ;SC = 0   GRAPH 0
1339
1340  005034  112720  000033                    MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
1341  005040  112720  000135                    MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
1342  005044  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1343  005050  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1344  005054  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1345  005060  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1346  005064  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1347  005070  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1348  005074  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1349  005100  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1350  005104  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1351  005110  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1352  005114  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1353  005120  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1354  005124  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1355  005130  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1356  005134  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1357  005140  112720  000100                    MOVB    #LN0,(R0)+              ;LOAD NOP TO LOAD SILO
1358  005144  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1359  005146  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1360
1361  005152  004537  007574                    JSR     R5,CURSOR               ;REMOVE CURSORS
1362  005156     002     103                    .BYTE   BIT1,LDC0               ;ON GRAPH 0
1363  005160  100000                            .WORD   BIT15
1364  005162  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1365  005166  004737  010742                    JSR     PC,DELAY
1366
```

```
1367
1368                                    ;;*****************************************************************
1369                                    ;*TEST 12       J       CURSORS ON GRAPH 1
1370                                    ;;*****************************************************************
1371  005172  000004                   TST12:  SCOPE
1372  005174  004737  011070                   JSR     R5,AMSG                 ;DISPLAY HEADER
1373  005200  012512                           CURGR1
1374  005202  012700  015562                   MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
1375  005206  112720  000033                   MOVB    #ESC,(R0)+              ;ENTER VT-55 MODE
1376  005212  112720  000061                   MOVB    #61,(R0)+
1377  005216  112720  000101                   MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1378  005222  112720  000040                   MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1379  005226  112720  000111                   MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1380  005232  112720  000060                   MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LILES, AND CURSORS
1381  005236  105020                           CLRB    (R0)+                   ;LOAD TERMINATOR
1382  005240  004737  010074                   JSR     PC,XPRNT                ;EXECUTE
1383
1384  005244  004537  007242                   JSR     R5,UPDWN                ;LOAD DATA PATTERN
1385  005250     004     112                   .BYTE   BIT2,LDG1               ;GRAPH 1 DECREMENTING PAT.
1386  005252  100354                           .WORD   BIT15!MAXHOZ            ;MAX #
1387  005254  004737  010074                   JSR     PC,XPRNT                ;EXECUTE
1388
1389  005260  004537  007574                   JSR     R5,CURSOR               ;ENABLE CURSORS
1390  005264     004     113                   .BYTE   BIT2,LDC1
1391  005266  002000                           .WORD   ADDLIN
1392
1393  005270  112720  000033                   MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
1394  005274  112720  000135                   MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
1395  005300  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1396  005304  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1397  005310  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1398  005314  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1399  005320  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1400  005324  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1401  005330  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1402  005334  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1403  005340  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1404  005344  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1405  005350  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1406  005354  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1407  005360  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1408  005364  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1409  005370  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1410  005374  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
1411  005400  105020                           CLRB    (R0)+                   ;LOAD TERMINATOR
1412  005402  004737  010074                   JSR     PC,XPRNT                ;EXECUTE
1413  005406  004537  007574                   JSR     R5,CURSOR               ;REMOVE CURSOR ON GRAPH 1
1414  005412     004     113                   .BYTE   BIT2,LDC1
1415  005414  100000                           .WORD   BIT15
1416  005416  004737  010074                   JSR     PC,XPRNT                ;EXECUTE
1417  005422  004737  010742                   JSR     PC,DELAY
```

```
                                ;;*****************************************************************
                                ;*TEST 13,      K       TEST STARTING COORDINATE ON GRAPH 0
                                ;;*****************************************************************
005426  000004                  TST13:  SCOPE
005430  004537  011070                  JSR     R5,AMSG                 ;DISPLAY HEADER
005434  012543                          SCORDO
005436  012700  015562                  MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
005442  112720  000033                  MOVB    #ESC,(R0)+              ;ENTER VT-55 MODE
005446  112720  000061                  MOVB    #61,(R0)+
005452  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
005456  112720  000040                  MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
005462  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
005466  112720  000060                  MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LILOS, AND CURSORS
005472  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
005474  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
005500  004537  007242                  JSR     R5,UPDWN                ;LOAD GRAPH 0 DATA
005504      002      102                .BYTE   BIT1,LDG0
005506  100554                          .WORD   BIT15!MAXHOZ
005510  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
005514  012703  000004                  MOV     #4,R3                   ;LOAD SINE COUNTER
005520  012701  000777                  MOV     #MAXVRT-1,R1            ;LOAD STARTING COORD.
005524  012700  007201                  MOV     #SINEND-40,R0           ;LOAD SA FOR FIRST PASS
005530  000402                          BR      1$
005532  012700  007241          3$:     MOV     #SINEND,R0              ;LOAD SINE POINTER
005536  114037  005556          1$:     MOVB    -(R0),10$               ;LOAD SINE DATA  WORD
005542  001413                          BEQ     2$                      ;BR IF NO MORE DATA
005544  010137  005560                  MOV     R1,11$                  ;LOAD STARTING COORDINATE
005550  004537  007436                  JSR     R5,STCORD               ;LOAD DATA INTO BUFFER
005554      002      102                .BYTE   BIT1,LDG0               ;FOR GRAPH 0
005556  000000          10$:    .WORD   0                               ;DATA TO BE LOADED
005560  000000          11$:    .WORD   0                               ;STARTING COORD.
005562  004737  010074                  JSR     PC,XPRNT                ;EXECUTE
005566  005301                          DEC     R1                      ;DONE ALL COORD. ?
005570  000762                          BR      1$
005572  005303          2$:     DEC     R3                              ;FINISHED ALL LINES ?
005574  001356                          BNE     3$                      ;BR IF NOT
005576  012700  015562                  MOV     #BUFFER,R0
005602  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
005606  112720  000135                  MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
005612  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005616  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005622  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005626  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005632  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005636  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005642  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005646  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005652  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005656  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005662  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005666  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
005672  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
```

```
 1474  005676  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1475  005702  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1476  005706  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1477  005712  105020                        CLRB    (R0)+               ;LOAD TERMINATOR
 1478  005714  004737  010074                JSR     PC,XFRNT            ;EXECUTE
 1479  005720  004737  010742                JSR     PC,DELAY            ;
 1480
 1481                                ;;*****************************************************************
 1482                                ;*TEST 14       L       TEST STARTING COORDINATE ON GRAPH 1
 1483                                ;;*****************************************************************
 1484  005724  000004                TST14:  SCOPE
 1485  005726  004537  011070                JSR     R5,AMSG             ;DISPLAY HEADER
 1486  005732  012615                        SCORD1
 1487  005734  012700  015562                MOV     #BUFFER,R0          ;LOAD OUTPUT BUFFER POINTER
 1488  005740  112720  000033                MOVB    #ESC,(R0)+          ;ENTER VT-55 MODE
 1489  005744  112720  000061                MOVB    #61,(R0)+
 1490  005750  112720  000101                MOVB    #LDE0,(R0)+         ;LOAD ENABLE 0
 1491  005754  112720  000040                MOVB    #BIT5,(R0)+         ;DISABLE DISPLAY
 1492  005760  112720  000111                MOVB    #LDE1,(R0)+         ;LOAD ENABLE 1
 1493  005764  112720  000060                MOVB    #BIT5!BIT4,(R0)+    ;CLEAR GRAPH, LILES, AND CURSORS
 1494  005770  105020                        CLRB    (R0)+               ;LOAD TERMINATOR
 1495  005772  004737  010074                JSR     PC,XPRNT            ;EXECUTE
 1496
 1497  005776  004537  007242                JSR     R5,UPDWN            ;LOAD GRAPH 0 DATA
 1498  006002     004     112                .BYTE   BIT2,LDG1
 1499  006004  100354                        .WORD   BIT15!MAXHOZ
 1500  006006  004737  010074                JSR     PC,XPRNT            ;EXECUTE
 1501
 1502  006012  012703  000004                MOV     #4,R3               ;LOAD SINE COUNT
 1503  006016  012701  000777                MOV     #MAXVRT-1,R1        ;LOAD STARTING CORD.
 1504  006022  012700  007201                MOV     #SINEND-40,R0       ;LOAD SA POINTER
 1505  006026  000402                        BR      1$
 1506  006030  012700  007241        3$:     MOV     #SINEND,R0          ;LOAD SINE POINTER
 1507  006034  114037  006054        1$:     MOVB    -(R0),10$           ;LOAD SINE DATA  WORD
 1508  006040  001413                        BEQ     2$                  ;BR IF NO MORE DATA
 1509  006042  010137  006056                MOV     R1,11$              ;LOAD STARTING COORDINATE
 1510
 1511  006046  004537  007436                JSR     R5,STCORD           ;LOAD DATA INTO BUFFER
 1512  006052     004     112                .BYTE   BIT2,LDG1           ;FOR GRAPH 1
 1513  006054  000000                10$:    .WORD   0                   ;DATA TO BE LOADED
 1514  006056  000000                11$:    .WORD   0                   ;STARTING COORD.
 1515
 1516  006060  004737  010074                JSR     PC,XPRNT            ;EXECUTE
 1517  006064  005301                        DEC     R1                  ;DONE ALL COORD. ?
 1518  006066  000762                        BR      1$
 1519  006070  005303                2$:     DEC     R3                  ;DONE ALL LINES ?
 1520  006072  001356                        BNE     3$                  ;BR IF NOT
 1521  006074  012700  015562                MOV     #BUFFER,R0
 1522  006100  112720  000033                MOVB    #ESC,(R0)+          ;LOAD "ESC" CODE
 1523  006104  112720  000135                MOVB    #COPY,(R0)+         ;LOAD COPY SCREEN
 1524  006110  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1525  006114  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1526  006120  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1527  006124  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1528  006130  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
 1529  006134  112720  000100                MOVB    #LNO,(R0)+          ;LOAD NOP TO LOAD SILO
```

```
 1530  006140  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1531  006144  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1532  006150  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1533  006154  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1534  006160  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1535  006164  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1536  006170  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1537  006174  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1538  006200  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1539  006204  112720  000100          MOVB    #LNO,(R0)+      ;LOAD NOP TO LOAD SILO
 1540  006210  105020                  CLRB    (R0)+
 1541  006212  004737  010074          JSR     PC,XPRNT        ;EXECUTE
 1542  006216  004737  010742          JSR     PC,DELAY        ;
 1543
```

```
1544
1545                                    ;;*****************************************************************
1546                                    ;*TEST 15        M       VT55 ADJUSTMENT PATTERN
1547                                    ;;*****************************************************************
1548  006222  000004                    TST15:  SCOPE
1549                                    ;FILL THE SCREEN WITH THE H CHARACTER
1550
1551  006224  012702  000014                    MOV     #12.,R2                 ;LOAD COUNT
1552  006230  012700  015562                    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
1553  006234  112720  000033                    MOVB    #ESC,(R0)+              ;ENTER VT55 FORMAT
1554  006240  112720  000061                    MOVB    #61,(R0)+               ;LOAD ENTER "01"CODE
1555  006244  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1556  006250  112720  000040                    MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
1557  006254  112720  000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1558  006260  112720  000060                    MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LILES, AND CURSORS
1559  006264  112720  000033                    MOVB    #ESC,(R0)+              ;ENTER VT55 FORMAT
1560  006270  112720  000062                    MOVB    #62,(R0)+               ;EXIT "02" CODE
1561  006274  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
1562  006276  004737  010074                    JSR     PC,XPRNT                ;EXECUTE
1563  006302  012700  015562                    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
1564  006306  112720  000015                    MOVB    #15,(R0)+               ;LOAD CR
1565  006312  112720  000012                    MOVB    #12,(R0)+               ;LOAD LF
1566  006316  112720  000015                    MOVB    #15,(R0)+               ;LOAD CR
1567  006322  112720  000012                    MOVB    #12,(R0)+               ;LOAD LF
1568  006326  012701  000050                    MOV     #40.,R1                 ;LOAD SCREEN WIDTH
1569  006332  012720  044110          1$:       MOV     #44110,(R0)+            ;LOAD ASCII H
1570  006336  005301                            DEC     R1                      ;FINISHED ?
1571  006340  001374                            BNE     1$                      ;BR IF NOT
1572  006342  105020                            CLRB    (R0)+                   ;LOAD TERM.
1573  006344  004737  010074          2$:       JSR     PC,XPRNT                ;XFER TO SCREEN
1574  006350  005302                            DEC     R2                      ;FINISHED ALL LINES
1575  006352  001374                            BNE     2$                      ;BR IF NOT
1576
1577                                    ;NOW INSTALL THE HORIZONTAL LINES
1578
1579  006354  012700  015562                    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
1580  006360  112720  000033                    MOVB    #33,(R0)+               ;ENABLE CHART MODE
1581  006364  112720  000061                    MOVB    #61,(R0)+               ;
1582  006370  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
1583  006374  112720  000041                    MOVB    #BIT5!BIT0,(R0)+        ;LOAD DISPLAY ENABLE
1584  006400  112720  000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
1585  006404  112720  000043                    MOVB    #BIT5!BIT1!BIT0,(R0)+   ;ENABLE HORIZ.+ VERT LINES
1586  006410  112720  000100                    MOVB    #LN0,(R0)+
1587  006414  112720  000104                    MOVB    #LHV0,(R0)+             ;LOAD HORIZ INST
1588  006420  012737  002002  007772            MOV     #ADDLIN+2,BASE          ;LOAD BASE LINE VALUE
1589  006426  004737  007776          3$:       JSR     PC,SHUFF                ;SHUFFEL THE DATA
1590  006432  010120                            MOV     R1,(R0)+                ;SAVE THE DATA
1591  006434  062737  000024  007772            ADD     #24,BASE                ;UPDATE BASE LINE VALUE
1592  006442  022737  002340  007772            CMP     #ADDLIN+340,BASE        ;TEST FOR GREATER THAN VALID
1593  006450  100366                            BPL     3$                      ;BR IF OK
1594  006452  105020                            CLRB    (R0)+                   ;LOAD TERM
1595  006454  004737  010074                    JSR     PC,XPRNT                ;XMIT TO THE SCREEN
```

```
                                        ;NOW LOAD THE VERTICAL LINES
006460  012700  015562                  MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
006464  112720  000033                  MOVB    #33,(R0)+               ;LOAD ESC
006470  112720  000061                  MOVB    #61,(R0)+               ;ENTER CHART MODE
006474  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
006500  112720  000114                  MOVB    #LHV1,(R0)+             ;LOAD VERT LINE INST.
006504  012737  002000  007772          MOV     #ADDLIN,BASE            ;LOAD STARTING LINE #
006512  004737  007776          4$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA
006516  010120                          MOV     R1,(R0)+                ;SAVE DATA
006520  062737  000015  007772          ADD     #15,BASE                ;UPDATE DATA
006526  022737  003000  007772          CMP     #ADDLIN!1000,BASE       ;TEST FOR LAST DATA LINE
006534  100366                          BPL     4$                      ;BR IF NOT DONE
006536  012737  002777  007772          MOV     #ADDLIN!777,BASE        ;LOAD LAST LINE
006544  004737  007776                  JSR     PC,SHUFF                ;SHUFFEL THE DATA
006550  010120                          MOV     R1,(R0)+                ;LOAD LAST DATA LINE
006552  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
006556  112720  000135                  MOVB    #COPY,(R0)+             ;LOAD COPY SCREEN
006562  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006566  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006572  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006576  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006602  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006606  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006612  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006616  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006622  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006626  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006632  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006636  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006642  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006646  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006652  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006656  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP TO LOAD SILO
006662  105020                          CLRB    (R0)+                   ;LOAD TERM.
006664  004737  010074                  JSR     PC,XPRNT                ;SEND TO SCREEN

006670  004737  010742                  JSR     PC,DELAY                ;DELAY
```

```
                                .SBTTL  END OF PASS ROUTINE

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

006674                          $EOP:
006674  000004                          SCOPE
006676  005737  001310                  TST     LAST                    ;TEST IF MORE
006702  001411                          BEQ     1$                      ;BR IF NONE
006704  023737  001310  001312          CMP     LAST,VTNOW              ;IS THIS THE LAST ONE
006712  001405                          BEQ     1$                      ;BR IF YES
006714  062737  000010  001312          ADD     #10,VTNOW
006722  000137  002016                  JMP     RSTRT                   ;TEST NEXT ONE
006726  000240                  1$:     NOP
006730  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
006734  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
006740  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
006746  005327                          DEC     (PC)+           ;;LOOP?
006750  000001                  $EOPCT: .WORD   1
006752  003015                          BGT     $DOAGN          ;;YES
006754  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
006756  000001                  $ENDCT: .WORD   1
006760  006750                          $EOPCT
006762  104400  007015                  TYPE    ,$ENDMG         ;;TYPE "END PASS"
006766  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
006772  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
006774  000005                          RESET                   ;;CLEAR THE WORLD
006776  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
007000  000240                          NOP                     ;;SAVE ROOM
007002  000240                          NOP                     ;;FOR
007004  000240                          NOP                     ;;ACT11
007006                          $DOAGN:
007006  000137                          JMP     @(PC)+          ;;RETURN
007010  002010                  $RTNAD: .WORD   RSTRTA
007012     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
007015     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS/
007022  050040  051501  000123
```

```
 1680  007030  000  003  004          .BYTE   000,003,004,004,005
 1681  007033  004  005
 1682  007035  006  007  010          .BYTE   006,007,010,012,014,015,017,021,024,027,032,035,040
 1683  007040  012  014  015
 1684  007043  017  021  024
 1685  007046  027  032  035
 1686  007051  040
 1687  007052  043  047  053          .BYTE   043,047,053,056,062,066,073,077,103,110,115,121,126,133
 1688  007055  056  062  066
 1689  007060  073  077  103
 1690  007063  110  115  121
 1691  007066  126  133
 1692  007070  137  144  151          .BYTE   137,144,151,156,162,167,174,201,206,213,217,223,230,235
 1693  007073  156  162  167
 1694  007076  174  201  206
 1695  007101  213  217  223
 1696  007104  230  235
 1697  007106  241  245  251          .BYTE   241,245,251,255,261,265,271,274,277,303,305,310,313
 1698  007111  255  261  265
 1699  007114  271  274  277
 1700  007117  303  305  310
 1701  007122  313
 1702  007123  316  317  321          .BYTE   316,317,321,323,324,326,327,330,331,331,331,331,331
 1703  007126  323  324  326
 1704  007131  327  330  331
 1705  007134  331  331  331
 1706  007137  331
 1707  007140  330  327  326          .BYTE   330,327,326,325,323,322,320,317,314,312,307
 1708  007143  325  323  322
 1709  007146  320  317  314
 1710  007151  312  307
 1711  007153  304  301  275          .BYTE   304,301,275,272,267,263,257,253,247,243,237,232,225
 1712  007156  272  267  263
 1713  007161  257  253  247
 1714  007164  243  237  232
 1715  007167  225
 1716  007170  221  214  207          .BYTE   221,214,207,203,176,171,164,157,153,145,140
 1717  007173  203  176  171
 1718  007176  164  157  153
 1719  007201  145  140
 1720  007203  134  127  121          .BYTE   134,127,121,115,111,104,077,073,066,063,057,053,047
 1721  007206  115  111  104
 1722  007211  077  073  066
 1723  007214  063  057  053
 1724  007217  047
 1725  007220  043  040  034          .BYTE   043,040,034,031,027,023,021,017,015,013,011,010,007
 1726  007223  031  027  023
 1727  007226  021  017  015
 1728  007231  013  011  010
 1729  007234  007
 1730  007235  006  005  004          .BYTE   006,005,004,003
 1731  007240  003
 1732  007241  000              SINEND: .BYTE   0
 1733                                   .EVEN
 1734                           ;UP-DOWN SUBROUTINE
 1735
```

```
 1736  007242  012537  007432          UPDWN:  MOV     (R5)+,11$       ;GET GRAPH BIT AND INC/DEC WORD
 1737  007246  012537  007434                  MOV     (R5)+,12$       ;GET STARTING COORD.
 1738  007252  012700  015562                  MOV     #BUFFER,R0      ;LOAD THE POINTER
 1739  007256  112720  000033                  MOVB    #ESC,(R0)+      ;LOAD 'ESCAPE'
 1740  007262  112720  000061                  MOVB    #61,(R0)+       ;ENABLE GRAPHIC MODE
 1741  007266  112720  000101                  MOVB    #LOED,(R0)+     ;LOAD ENABLE D
 1742  007272  112710  000041                  MOVB    #BITS!BIT0,(R0) ;ENABLE DISPLAY
 1743  007276  153720  00743                   BISB    11$,(R0)+       ;LOAD GRAPH ENABLE BIT
 1744  007302  012737  000001  007430          MOV     #1,10$
 1745  007310  112720  000100                  MOVB    #LMO,(R0)+      ;LOAD NOP
 1746  007314  112720  000110                  MOVB    #LSC,(R0)+      ;LOAD STARTING COORD.
 1747  007320  005037  007772                  CLR     BASE            ;LOAD START. COORD.
 1748  007324  004737  007776                  JSR     PC,SHUFF
 1749  007330  010120                          MOV     R1,(R0)+        ;LOAD INTO BUFFER
 1750  007332  112720  000100                  MOVB    #LMO,(R0)+      ;LOAD NOP
 1751  007336  113720  007433                  MOVB    11$+1,(R0)+     ;LOAD LOAD GRAPH X
 1752  007342  013737  007434  007772  4$:     MOV     12$,BASE                ;LOAD DATA
 1753  007350  004737  007776          1$:     JSR     PC,SHUFF        ;SHUFFEL DATA
 1754  007354  010120                          MOV     R1,(R0)+        ;LOAD DATA
 1755  007356  005737  007434                  TST     12$             ;TEST FOR UP OR DOWN DATA
 1756  007362  100007                          BPL     2$              ;BR IF INC.
 1757  007364  005337  007772                  DEC     BASE            ;DEC. DATA
 1758  007370  042737  177400  007772          BIC     #177400,BASE
 1759  007376  001407                          BEQ     5$
 1760  007400  000763                          BR      1$
 1761  007402  005237  007772          2$:     INC     BASE            ;CHANGE DATA
 1762  007406  022737  000354  007772  3$:     CMP     #MAXHOZ,BASE    ;TEST FOR LAST
 1763  007414  001355                          BNE     1$              ;BR IF NOT
 1764  007416  005337  007430          5$:     DEC     10$
 1765  007422  100347                          BPL     4$              ;BR IF NOT
 1766  007424  105010                          CLRB    (R0)            ;LOAD TERMINATOR
 1767  007426  000205                          RTS     R5              ;EXIT
 1768  007430  000000          10$:    0
 1769  007432  000000          11$:    0
 1770  007434  000000          12$:    0
```

```
 1771                                           ;STARTING CORDINATE SUBROUTINE
 1772
 1773  007436  012537  007570          STCORD: MOV     (R5)+,11$               ;GET GRAPH BIT AND INC/DEC WORD
 1774  007442  012537  007572                  MOV     (R5)+,12$               ;GET STARTING CORD.
 1775  007446  012537  007566                  MOV     (R5)+,10$               ;GET ARG. WORD
 1776  007452  010046                          MOV     R0,-(SP)                ;SAVE R0
 1777  007454  010146                          MOV     R1,-(SP)                ;SAVE R1
 1778  007456  012700  015562                  MOV     #BUFFER,R0              ;LOAD THE POINTER
 1779  007462  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD 'ESCAPE'
 1780  007466  112720  000061                  MOVB    #61,(R0)+               ;ENABLE GRAPHIC MODE
 1781  007472  112720  000101                  MOVB    #LMO,(R0)+              ;LOAD ENABLE 0
 1782  007476  112710  000041                  MOVB    #BIT0!BIT5,(R0)         ;ENABLE DISPLAY
 1783  007502  153720  007570                  BISB    11$,(R0)+               ;LOAD GRAPH ENABLE BIT
 1784  007506  112720  000100                  MOVB    #LMO,(R0)+              ;LOAD MOP
 1785  007512  112720  000110                  MOVB    #LSC,(R0)+              ;LOAD STARTING CORD.
 1786  007516  013737  007566  007772          MOV     10$,BASE                ;LOAD START. CORD.
 1787  007524  004737  007776                  JSR     PC,SHUFF
 1788  007530  010120                          MOV     R1,(R0)+                ;LOAD INTO BUFFER
 1789  007532  112720  000100                  MOVB    #LMO,(R0)+              ;LOAD MOP
 1790  007536  113720  007571                  MOVB    11$+1,(R0)+             ;LOAD LOAD GRAPH X
 1791  007542  013737  007572  007772  4$:     MOV     12$,BASE                ;LOAD DATA
 1792  007550  004737  007776          1$:     JSR     PC,SHUFF                ;SHUFFEL DATA
 1793  007554  010120                          MOV     R1,(R0)+                ;LOAD DATA
 1794  007556  105010                          CLRB    (R0)                    ;LOAD TERMINATOR
 1795  007560  012601                          MOV     (SP)+,R1
 1796  007562  012600                          MOV     (SP)+,R0
 1797  007564  000205                          RTS     R5                      ;EXIT
 1798  007566  000000          10$:    0
 1799  007570  000000          11$:    0
 1800  007572  000000          12$:    0
```

```
                                        ;CURSOR SUBROUTINE
                                        ;
007574 012537 007766            CURSOR: MOV     (R5)+,11$               ;GET ARG. WORD
007600 012537 007770                    MOV     (R5)+,12$               ;GET ARG. WORD
007604 012700 015562                    MOV     #BUFFER,R0              ;LOAD THE POINTER
007610 112720 000033                    MOVB    #ESC,(R0)+              ;LOAD 'ESCAPE'
007614 112720 000061                    MOVB    #'1,(R0)+               ;ENABLE GRAPHIC MODE
007620 112720 000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
007624 112710 000041                    MOVB    #BIT5!BIT0,(R0)         ;ENABLE DISPLAY
007630 153720 007766                    BISB    11$,(R0)+               ;LOAD GRAPH ENABLE BIT
007634 112720 000111                    MOVB    #LDE1,(R0)+             ;LOAD ENABLE
007640 013701 007766                    MOV     11$,R1
007644 006301                           ASL     R1
007646 112710 000040                    MOVB    #BIT5,(R0)
007652 150120                           BISB    R1,(R0)+
007654 112720 000100                    MOVB    #LNO,(R0)+              ;LOAD NOP
007660 112720 000110                    MOVB    #LSC,(R0)+              ;LOAD STARTING CORD.
007664 005037 007772                    CLR     BASE                    ;LOAD START. CORD.
007670 004737 007776                    JSR     PC,SHUFF
007674 010120                           MOV     R1,(R0)+                ;LOAD INTO BUFFER
007676 112720 000100                    MOVB    #LNO,(R0)+              ;LOAD NOP
007702 113720 007767                    MOVB    11$+1,(R0)+             ;LOAD "LOAD CURSOR ON GRAPH X"
007706 013737 007770 007772             MOV     12$,BASE                ;LOAD CURSOR POSITION DATA
007714 004737 007776            1$:     JSR     PC,SHUFF                ;SHUFFEL DATA
007720 010120                           MOV     R1,(R0)+                ;LOAD DATA
007722 005737 007770                    TST     12$                     ;TEST FOR UP OR DOWN DATA
007726 100007                           BPL     2$                      ;BR IF INC.
007730 005337 007772                    DEC     BASE
007734 042737 177000 007772             BIC     #^177000,BASE
007742 001407                           BEQ     4$
007744 000763                           BR      1$
007746 005237 007772            2$:     INC     BASE                    ;CHANGE DATA
007752 022737 003000 007772     3$:     CMP     #ADDLIN!MAXVRT,BASE     ;TEST FOR LAST
007760 001355                           BNE     1$                      ;BR IF NOT
007762 105010                   4$:     CLRB    (R0)                    ;LOAD TERMINATOR
007764 000205                           RTS     R5                      ;EXIT
007766 000000                   11$:    0
007770 000000                   12$:    0
007772 000000                   BASE:   0
007774 000000                   BASE1:  0
```

```
 1843                                           ;SUBROUTINE TO SHUFFEL DATA INTO VT-55 DATA BYTE FORMAT
 1844
 1845  007776  013702  007772          SHUFF:  MOV     BASE,R2         ;LOAD VALUE TO BE SHUFFELED
 1846  010002  010237  007774                  MOV     R2,BASE1
 1847  010006  042737  177740  007774          BIC     #177740,BASE1
 1848  010014  010201                          MOV     R2,R1
 1849  010016  006001                          ROR     R1
 1850  010020  006001                          ROR     R1
 1851  010022  006001                          ROR     R1
 1852  010024  006001                          ROR     R1
 1853  010026  006001                          ROR     R1
 1854  010030  110137  007775                  MOVB    R1,BASE1+1              ;RELOAD R1
 1855  010034  042737  170340  007774          BIC     #170340,BASE1   ;MASK
 1856  010042  052737  020040  007774          BIS     #20040,BASE1    ;CONVERT TO ASCII
 1857  010050  032737  010000  007772          BIT     #BIT10,BASE
 1858  010056  001403                          BEQ     1$              ;BR IF NOT SET
 1859  010060  052737  010000  007774          BIS     #BIT12,BASE1    ;SET BIT
 1860  010066  013701  007774          1$:     MOV     BASE1,R1
 1861  010072  000207                          RTS     PC              ;EXIT
```

```
1862                                        ;DISPLAY SUBROUTINE
1863
1864  010074 010046                 XPRNT:  MOV     R0,-(SP)
1865  010076 010146                         MOV     R1,-(SP)
1866  010100 005037 010546                  CLR     ANESC           ;HOUSEKEEP
1867  010104 005037 010552                  CLR     TERM
1868  010110 005037 010550                  CLR     NOEXIT
1869  010114 012700 015562                  MOV     #BUFFER,R0      ;SETUP BUFFER POINTER
1870  010120 105777 171204          1$:     TSTB    @VTCS           ;TEST READY
1871  010124 100375                         BPL     1$
1872  010126 104405                         CKSWR                   ;TEST IF "CTRL G" ON CONSOLE TTY
1873  010130 000240                         NOP
1874  010132 000240                         NOP
1875  010134 000240                         NOP
1876  010136 000240                         NOP
1877  010140 000240                         NOP
1878  010142 112001                 2$:     MOVB    (R0)+,R1        ;GET A CHAR.
1879  010144 001562                         BEQ     16$             ;BR IF TERM
1880  010146 122701 000033                  CMPB    #33,R1          ;TEST FOR ESC
1881  010152 001031                         BNE     4$              ;BR IF NOT
1882  010154 122710 000135                  CMPB    #COPY,(R0)      ;TEST IF NEXT CHAR IS A "COPY"
1883  010160 001023                         BNE     3$              ;BR IF NOT
1884  010162 032777 004000 170750           BIT     #BIT11,@SWR     ;TEST IF "INHIBIT" COPIER TESTING
1885  010170 001402                         BEQ     13$             ;BR IF NOT
1886  010172 105720                 21$:    TSTB    (R0)+           ;YES - ADJUST POINTER
1887  010174 000762                         BR      2$
1888  010176 032777 002000 170734   13$:    BIT     #BIT10,@SWR     ;TEST IF "SAVE PAPER MODE" IS ENABLED
1889  010204 001407                         BEQ     20$             ;BR IF NOT
1890  010206 013737 001202 010542           MOV     SPASS,30$       ;GET PASS COUNT
1891  010214 042737 177770 010542           BIC     #177770,30$     ;MASK OF BITS
1892  010222 001363                         BNE     21$             ;BR IF NOT 1 OR 8 PASSES
1893  010224 105237 010550          20$:    INCB    NOEXIT          ;SET "DO NOT EXIT UNTILL X-ON"
1894  010230 005237 010546          3$:     INC     ANESC           ;SET SOFT FLAG
1895  010234 000402                         BR      5$
1896  010236 005037 010546          4$:     CLR     ANESC           ;CLEAR SOFT FLAG
1897  010242 110177 171064          5$:     MOVB    R1,@VTOB        ;LOAD CHAR
1898  010246 105777 171052                  TSTB    @VTIS           ;TEST INPUT FLAG
1899  010252 100322                         BPL     1$              ;BR IF CLEARED
1900  010254 005737 010546                  TST     ANESC           ;TEST IF "ESC" WAS JUST SENT
1901  010260 001317                         BNE     1$
1902  010262 005037 001124                  CLR     $GOODAT         ;CLEAR EXPECTED DATA
1903  010266 000420                         BR      10$
1904
```

```
                                        ;WAIT FOR A KEYBOARD FLAG  -  ERROR REPORT IF NONE

010270  013737  001316  011162  6$:     MOV     TIME0,TIME1
010276  005037  011164                  CLR     TIME2           ;LOAD DELAY
010302  105777  171016          7$:     TSTB    @VTIS           ;TEST IF INPUT FLAG
010306  100410                  BMI     10$             ;BR IF SET
010310  005337  011164                  DEC     TIME2           ;DELAY
010314  001372                  BNE     7$
010316  104405                  CKSWR                   ;TEST FOR "CTRG G" ON CONSOLE TTY
010320  005337  011162                  DEC     TIME1           ;DELAY
010324  001366                  BNE     7$
010326  000457                  BR      14$             ;REPORT ERROR

                                        ;INPUT FLAG SET - FIND OUT WHAT CHARACTER IT WAS

010330  017737  170772  001126  10$:     MOV     @VTIB,$BDDAT    ;READ CHAR
010336  042737  177600  001126          BIC     #177600,$BDDAT  ;MASK
010344  022737  000021  001126          CMP     #XON,$BDDAT     ;TEST FOR X ON
010352  001006                  BNE     11$
010354  005037  010550                  CLR     NOEXIT
010360  005737  010552                  TST     TERM            ;TEST IF TERMINATOR WAS SET
010364  001655                  BEQ     1$              ;BR IF NOT
010366  000462                  BR      17$             ;BR IF ONLY ONE
010370  022737  000023  001126  11$:     CMP     #XOFF,$BDDAT    ;TEST FOR X OFF
010376  001006                  BNE     12$             ;BR IF NOT
010400  105237  010550                  INCB    NOEXIT          ;SET "NO EXIT UNTIL X-ON" RCVD
010404  012737  000021  001124          MOV     #XON,$GDDAT     ;LOAD EXPECTED VALUE
010412  000736                  BR      6$              ;WAIT FOR X-ON
010414  005037  001124          12$:    CLR     $GDDAT          ;LOAD EXPECTED
010420  105777  170514                  TSTB    @SWR            ;TEST SWR
010424  100020                  BPL     14$             ;BR IF CLEARED
010426  022737  000057  001126          CMP     #'/,$BDDAT      ;COMPARE
010434  001417                  BEQ     15$             ;BR IF EQUAL
010436  022737  000134  001126          CMP     #'\,$BDDAT      ;COMPARE
010444  001010                  BNE     14$             ;BR IF NOT
010446  012737  000001  010544          MOV     #1,LOOP         ;SET SOFT FLAG
010454  012737  011277  010572          MOV     #MSG0,FINDTA    ;SETUP MESSAGE
010462  000137  010562                  JMP     FINDDT

010466  104001          14$:    ERROR   1               ;UNEXPECTED OR INCORRECT INPUT FLAG
010470  000137  010130                  JMP     1$

010474  005037  010544          15$:    CLR     LOOP
010500  012737  011362  010572          MOV     #MSG1,FINDTA    ;SETUP MESSAGE
010506  000137  010562                  JMP     FINDDT
010512  012737  000001  010552  16$:     MOV     #1,TERM         ;SET "TERMINATOR HAS BEEN FOUND" FLAG
010520  012737  000021  001124          MOV     #XON,$GDDAT     ;LOAD EXPECTED
010526  105737  010550                  TSTB    NOEXIT          ;TEST IF ALLOWED TO LEAVE ROUTINE ?
010532  001256                  BNE     6$              ;BR IF NOT ALLOWED
010534  012601          17$:    MOV     (SP)+,R1
010536  012600                  MOV     (SP)+,R0
010540  000207                  RTS     PC              ;EXIT

010542  000000          30$:    0
010544  000000          LOOP:   0
010546  000000          ^ESC:   0
```

```
 1961  010550  000000                  NOEXIT: 0
 1962  010552  000000                  TERM:   0
 1963
 1964  010554  004537  011070          FNDA:   JSR     R5,AMSG
 1965  010560  011446                          MQ2                     ;ERROR ASK AGAIN
 1966  010562  012706  001100          FINDOT: MOV     #STACK,SP
 1967  010566  004537  011070                  JSR     R5,AMSG
 1968  010572  011362                  FINDTA: MQ1
 1969  010574  004737  011110                  JSR     PC,GETCHR
 1970  010600  000770                          BR      FINDOT
 1971
 1972  010602  105777  170522          1$:     TSTB    @VTOS
 1973  010606  100375                          BPL     1$
 1974  010610  110077  170516                  MOVB    R0,@VTDB
 1975  010614  042700  177640                  BIC     #177640,R0      ;MASK
 1976  010620  122700  000101                  CMPB    #'A,R0          ;TEST FOR NUMBER
 1977  010624  101353                          BHI     FNDA
 1978  010626  122700  000116                  CMPB    #'N,R0          ;TEST FOR OTHERS
 1979  010632  103750                          BLO     FNDA
 1980  010634  042700  177740                  BIC     #177740,R0      ;MAKE 0-37
 1981  010640  005300                          DEC     R0
 1982  010642  110037  001102                  MOVB    R0,$TSTNM       ;LOAD THAT TEST #
 1983  010646  006300                          ASL     R0
 1984  010650  005760  010676                  TST     DSPCH(R0)       ,TEST IF VALID
 1985  010654  001737                          BEQ     FNDA                    ;BR IF NOT
 1986  010656  016037  010676  001100          MOV     DSPCH(R0),$LPADR        ;LOAD RETURN ADDRESS
 1987  010664  016037  010676  001110          MOV     DSPCH(R0),$LPERR        ;LOAD ERROR LOOP ADDRESS
 1988  010672  000170  010676                  JMP     @DSPCH(R0)      ;GO TO THAT TEST
 1989
 1990  010676  002066                  DSPCH:  TST1+2
 1991  010700  002522                          TST2+2
 1992  010702  003150                          TST3+2
 1993  010704  003406                          TST4+2
 1994  010706  003644                          TST5+2
 1995  010710  004064                          TST6+2
 1996  010712  004302                          TST7+2
 1997  010714  004520                          TST10+2
 1998  010716  004746                          TST11+2
 1999  010720  005174                          TST12+2
 2000  010722  005430                          TST13+2
 2001  010724  005726                          TST14+2
 2002  010726  006244                          TST15+2
 2003  010730  000000                          0
 2004  010732  000000                          0
 2005  010734  000000                          0
 2006  010736  000000                          0
 2007  010740  000000                          0
```

```
2008                                        ;PROGRAM DELAY ROUTINE
2009
2010  010742  013737  001320  011020  DELAY:  MOV     SUBTST,10$      ;LOAD COUNT
2011  010750  005037  011022                  CLR     11$
2012  010754  005737  001322                  TST     WFTEST          ;TEST IF W.F. MODE
2013  010760  001404                          BEQ     2$              ;BR IF NOT
2014  010762  006237  011020                  ASR     10$             ;CHANGE DELAY TIMER
2015  010766  006237  011020                  ASR     10$
2016  010772  032777  010000  170140  2$:     BIT     #BIT12,@SWR     ;TEST SR
2017  011000  001006                          BNE     3$              ;BR IF SET
2018  011002  005337  011022                  DEC     11$             ;DELAY
2019  011006  001371                          BNE     2$
2020  011010  005337  011020                  DEC     10$
2021  011014  100366                          BPL     2$              ;DELAY
2022  011016  000207                  3$:     RTS     PC              ;EXIT
2023
2024  011020  000002                  10$:    2
2025  011022  000000                  11$:    0
2026
2027  011024  013737  001320  011064  ADELAY: MOV     SUBTST,10$
2028  011032  005037  011066                  CLR     11$
2029  011036  006237  011064                  ASR     10$
2030  011042  006237  011064                  ASR     10$
2031  011046  005337  011066          2$:     DEC     11$
2032  011052  001375                          BNE     2$
2033  011054  005337  011064                  DEC     10$
2034  011060  100372                          BPL     2$
2035  011062  000207                          RTS     PC
2036  011064  000000                  10$:    0
2037  011066  000000                  11$:    0
```

```
2038
2039                                        ;HEADER SUBROUTINE FOR VT-50
2040
2041  011070  012537  011100          AMSG:   MOV     (R5)+,10$       ;GET POINTER
2042  011074  004537  011166                  JSR     R5,MT08         ;MOVE TO BUFFER
2043  011100  000000                  10$:    0
2044  011102  004737  010074                  JSR     PC,XPRNT        ;DISPLAY IT
2045  011106  000205                          RTS     R5              ;EXIT
2046
2047
2048                                        ;SUBROUTINE TO GET A CHARACTER FROM THE VT50
2049
2050  011110  013737  001316  011162  GETCHR: MOV     TIME0,TIME1     ;LOAD TIME COUNTER
2051  011116  005037  011164                  CLR     TIME2
2052
2053  011122  105777  170176          1$:     TSTB    @VTIS           ;TEST INPUT STATUS
2054  011126  100005                          BPL     2$              ;BR IF CLEARED
2055  011130  017700  170172                  MOV     @VTIB,R0        ;READ A CHAR
2056  011134  062716  000002                  ADD     #2,(SP)         ;UPDATE RETURN
2057  011140  000207                          RTS     PC              ;EXIT
2058
2059  011142  005337  011164          2$:     DEC     TIME2           ;DELAY
2060  011146  001365                          BNE     1$
2061  011150  015337  011162                  DEC     TIME1           ;FINISHED ?
2062  011154  100362                          BPL     1$              ;LOOP TILL TIME EXPIRED
2063  011156  104002                          ERROR   2               ;NO INPUT FLAG FROM DEVICE
2064  011160  000207                          RTS     PC              ;EXIT
2065
2066  011162  000000                  TIME1:  0
2067  011164  000000                  TIME2:  0
2068
2069                                        ;MOVE TO THE OUTPUT BUFFER
2070
2071  011166  012500                  MT08:   MOV     (R5)+,R0        ;LOAD DEST.
2072  011170  012701  015562                  MOV     #BUFFER,R1      ;LOAD R1
2073  011174  112021                  1$:     MOVB    (R0)+,(R1)+     ;LOAD BYTE
2074  011176  001376                          BNE     1$              ;BR UNTIL DONE
2075  011200  000205                          RTS     R5              ;EXIT
2076
2077                                        .SBTTL  ASCII MESSAGES
```

```
2078
2079                                        ;ASCII MESSAGES
2080
2081  011202  031033  005015  050042  PWRMSG: .ASCIZ  <33><62><15><12>/"POWER"/<15><12>
2082  011210  053517  051105  006442
2083  011216  000012
2084  011220  031033  006415  046412  TITLE:  .ASCIZ  <33><62><15><15><12>/MAINDEC-11-DZVTD-B VT55 ACCEPTANCE TEST/<15><12>
2085  011226  044501  042116  041505
2086  011234  030455  026461  055104
2087  011242  052126  026504  020102
2088  011250  052126  032465  040440
2089  011256  041503  050105  040524
2090  011264  041516  020105  042524
2091  011272  052123  005015      000
2092  011277      033  015462  015510  MQ0:    .ASCIZ  <33><62><33><110><33><112>/LOOP ON TEST PATTERN LETTER (A THRU M) ? =  /
2093  011304  046112  047517  020120
2094  011312  047117  052040  051505
2095  011320  020124  040520  052124
2096  011326  051105  020116  042514
2097  011334  052124  051105  024040
2098  011342  020101  044124  052522
2099  011350  046440  020051  020077
2100  011356  020075  000040
2101  011362  031033  044033  045033  MQ1:    .ASCIZ  <33><62><33><110><33><112>/START AT TEST PATTERN LETTER (A THRU M) ? =
2102  011370  052123  051101  020124
2103  011376  052101  052040  051505
2104  011404  020124  040520  052124
2105  011412  051105  020116  042514
2106  011420  052124  051105  024040
2107  011426  020101  044124  052522
2108  011434  046440  020051  020077
2109  011442  020075  000040
2110  011446  005015  052012  054522  MQ2:    .ASCIZ  <15><12><12>/TRY AGAIN /<15><12>
2111  011454  040440  040507  047111
2112  011462  006440  000012
2113  011466  005015  042012  053105  WHATO:  .ASCIZ  <15><12><12>/DEVICE ADDRESS ? = /.200>
2114  011474  041511  020105  042101
2115  011502  051104  051505  020123
2116  011510  020077  020075  000200
2117  011516  051105  047522  020122  EM1:    .ASCIZ  /ERROR FLAG SET ON TRANSMITTER STATUS/
2118  011524  046106  043501  051440
2119  011532  052105  047440  020116
2120  011540  051124  047101  046523
2121  011546  052111  042524  020122
2122  011554  052123  052101  051525
2123  011562      000
2124  011563      116  020117  047111  EM2:    .ASCIZ  /NO INPUT FLAG DETECTED/
2125  011570  052520  020124  046106
2126  011576  043501  042040  052105
2127  011604  041505  042524  000104
2128  011612  047125  054105  042520  EM4:    .ASCIZ  /UNEXPECTED OR INCORRECT INPUT CHAR/
2129  011620  052103  042105  047440
2130  011626  020122  047111  047503
2131  011634  051122  041505  020124
2132  011642  047111  052520  020124
2133  011650  044103  051101      000
```

```
2134  011655     111  053116  046101  EM3:    .ASCIZ  /INVALID BUS ADDRESS, TRY AGAIN/
2135  011662  042111  041040  051525
2136  011670  040440  042104  042522
2137  011676  051523  020054  051124
2138  011704  020131  043501  044501
2139  011712  000116
2140  011714  051105  050122  020103  DH1:    .ASCIZ  /ERRPC    VTNOW    TSTNUM/
2141  011722  020040  053040  047124
2142  011730  053517  020040  020040
2143  011736  051524  047124  046525
2144  011744     000
2145  011745     105  051122  041520  DH4:    .ASCIZ  /ERRPC   VTNOW   TSTNUM   EXPCT   RECV/
2146  011753  020040  053040  047124
2147  011760  053517  020040  052040
2148  011766  052123  052516  020115
2149  011774  020040  054105  041520
2150  012002  020124  020040  042522
2151  012010  053103     000
2152  012013     033  015462  015510  DAHL:   .ASCIZ  <33><62><33><110><33><112>/GROWING HORIZ. LINE/
2153  012020  043512  047522  044527
2154  012026  043516  044040  051117
2155  012034  055111  020056  044514
2156  012042  042516     000
2157  012045     033  015462  015510  DAVL:   .ASCIZ  <33><62><33><110><33><112>/GROWING VERT. LINE/
2158  012052  043512  047522  044527
2159  012060  043516  053040  051105
2160  012066  027124  046040  047111
2161  012074  000105
2162  012076     033     062     033  SHL0:   .BYTE   33,62,33,110,33,112
2163  012101     110     033     112
2164  012104  044504  050123  040514          .ASCIZ  /DISPLAY STEPPING HORIZ. LINE ON GRAPH 0/
2165  012112  020131  052123  050105
2166  012120  044520  043516  044040
2167  012126  051117  055111  020056
2168  012134  044514  042516  047440
2169  012142  020116  051107  050101
2170  012150  020110  000060
2171  012154     033     062     033  SHL1:   .BYTE   33,62,33,110,33,112
2172  012157     110     033     112
2173  012162  044504  050123  040514          .ASCIZ  /DISPLAY STEPPING HORIZ. LINE ON GRAPH 1/
2174  012170  020131  052123  050105
2175  012176  044520  043516  044040
2176  012204  051117  055111  020056
2177  012212  044514  042516  047440
2178  012220  020116  051107  050101
2179  012226  020110  000061
2180  012232     033     062     033  GROA1:  .BYTE   33,62,33,110,33,112
2181  012235     110     033     112
2182  012240  044504  050123  040514          .ASCIZ  /DISPLAY DATA ON GRAPH 0 AND 1/
2183  012246  020131  040504  040523
2184  012254  047440  020116  051107
2185  012262  050101  020110  020060
2186  012270  047101  020104  000061
2187  012276     033     062     033  SHGL0:  .BYTE   33,62,33,110,33,112
2188  012301     110     033     112
2189  012304  052123  050105  044520          .ASCIZ  /STEPPING HISTOGRAM LINE ON GRAPH 0/
```

```
2190  012312  043516  044040  051511
2191  012320  047524  051107  046501
2192  012326  046040  047111  020105
2193  012334  047117  043440  040522
2194  012342  044120  030040     000
2195  012347     033     062     033  SHGL1:  .BYTE   33,62,33,110,33,112
2196  012352     110     033     112
2197  012355     123  042524  050120          .ASCIZ  /STEPPING HISTOGRAM LINE ON GRAPH 1/
2198  012362  047111  020107  044510
2199  012370  052123  043517  040522
2200  012376  020115  044514  042516
2201  012404  047440  020116  051107
2202  012412  050101  020110  000061
2203  012420     033     062     033  HGDA1.  .BYTE   33,62,33,110,33,112
2204  012423     110     033     112
2205  012426  044510  052123  043517          .ASCIZ  /HISTOGRAM ON GRAPH 0 AND 1/
2206  012434  040522  020115  047117
2207  012442  043440  040522  044120
2208  012450  030040  040440  042116
2209  012456  030440     000
2210  012461     033     062     033  CURGR0: .BYTE   33,62,33,110,33,112
2211  012464     110     033     112
2212  012467     103  051125  047523          .ASCIZ  /CURSORS ON GRAPH 0/
2213  012474  051522  047440  020116
2214  012502  051107  050101  020110
2215  012510  000060
2216  012512     033     062     033  CURGR1: .BYTE   33,62,33,110,33,112
2217  012515     110     033     112
2218  012520  052503  051522  051117          .ASCIZ  /CURSORS ON GRAPH 1/
2219  012526  020123  047117  043440
2220  012534  040522  044120  030440
2221  012542     000
2222  012543     033     062     033  SCORD0: .BYTE   33,62,33,110,33,112
2223  012546     110     033     112
2224  012551     123  040524  052122          .ASCIZ  /STARTING COORDINATE TEST ON GRAPH 0/
2225  012556  047111  020107  047503
2226  012564  051117  044504  040516
2227  012572  042524  052040  051505
2228  012600  020124  047117  043440
2229  012606  040522  044120  030040
2230  012614     000
2231  012615     033     062     033  SCORD1: .BYTE   33,62,33,110,33,112
2232  012620     110     033     112
2233  012623     123  040524  052122          .ASCIZ  /STARTING COORDINATE TEST ON GRAPH 1/
2234  012630  047111  020107  047503
2235  012636  051117  044504  040516
2236  012644  042524  052040  051505
2237  012652  020124  047117  043440
2238  012660  040522  044120  030440
2239  012666     000
2240          012670                  .EVEN
2241  012670  001116  001312  001314  DT1:    SERRPC,VTNOW,TSTNUM,0
2242  012676  000000
2243  012700  001116  001312  001314  DT4:    SERRPC,VTNOW,TSTNUM,$GDDAT,$BDDAT,0
2244  012706  001124  001126  000000
2245
```

```
                                .SBTTL  TTY INPUT ROUTINE

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

                                ;;**********************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
012714  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED*
012722  001074                        BNE     15$             ;;BRANCH IF NO
012724  105777  166214                TSTB    @$TKS           ;;CHAR THERE?
012730  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
012732  117746  166210                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
012736  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
012742  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
012746  001062                        BNE     15$             ;;NO, RETURN TO USER
012750  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
012756  001456                        BEQ     15$             ;;BRANCH IF YES

012760  104400  013441                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
012764  104400  013446        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
012770  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
012774  104401                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012776  104400  013457                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
013002  005046                1$:     CLR     -(SP)           ;;CLEAR COUNTER
013004  005046                        CLR     -(SP)           ;;THE NEW SWR
013006  105777  166132        7$:     TSTB    @$TKS           ;;CHAR THERE?
013012  100375                        BPL     7$              ;;IF NOT TRY AGAIN

013014  117746  166126                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
013020  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


013024  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
013030  001005                        BNE     10$             ;;BRANCH IF NOT
013032  104400  013434                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
013036  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
013042  000757                        BR      1$              ;;LET'S TRY IT AGAIN


013044  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
013050  001022                        BNE     16$             ;;BRANCH IF NO
013052  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
013056  001403                        BEQ     11$             ;;BRANCH IF YES
013060  016677  000002  166052        MOV     2(SP),@SWR      ;;SAVE NEW SWR
013066  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
013072  104400  001171        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
013076  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
013104  001003                        BNE     15$             ;;BRANCH IF NOT
013106  012777  000100  166020        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
013114  000002        15$:    RTI                             ;;RETURN
013116  004737  015046        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
013122  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
```

```
2302  013126  002420                        BLT     18$             ;;BRANCH IF YES
2303  013130  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
2304  013134  003015                        BGT     18$             ;;BRANCH IF YES
2305  013136  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2306  013142  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
2307  013146  001403                        BEQ     17$             ;;BRANCH IF YES
2308  013150  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
2309  013152  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
2310  013154  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
2311  013156  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2312  013162  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2313  013166  000707                        BR      7$              ;;GET THE NEXT ONE
2314  013170  104400  001170        18$:    TYPE    ,$QUES          ;;TYPE ?(CR)(LF)
2315  013174  000720                        BR      20$             ;;SIMULATE CONTROL-U
2316                                .DSABL  LSB
2317
2318
2319                                ;;*****************************************************************
2320                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2321                                ;*CALL:
2322                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2323                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2324                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
2325                                ;
2326
2327  013176  011646                SRDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2328  013200  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
2329  013206  105777  165732        1$:     TSTB    @#TKS           ;;WAIT FOR
2330  013212  100375                        BPL     1$              ;;A CHARACTER
2331  013214  117766  165726  000004        MOVB    @#TKB,4(SP)     ;;READ THE TTY
2332  013222  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
2333  013230  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
2334  013236  001013                        BNE     3$              ;;BRANCH IF NO
2335  013240  105777  165700        2$:     TSTB    @#TKS           ;;WAIT FOR A CHARACTER
2336  013244  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
2337  013246  117746  165674                MOVB    @#TKB,-(SP)     ;;GET CHARACTER
2338  013252  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2339  013256  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
2340  013262  001366                        BNE     2$              ;;IF NOT DISCARD IT
2341  013264  000750                        BR      1$              ;;YES, RESUME
2342  013266  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
2343  013274  002407                        BLT     4$              ;;BRANCH IF YES
2344  013276  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
2345  013304  003003                        BGT     4$              ;;BRANCH IF YES
2346  013306  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
2347  013314  000002                4$:     RTI                     ;;GO BACK TO USER
2348                                ;;*****************************************************************
2349                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2350                                ;*CALL:
2351                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2352                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2353                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2354
2355  013316  010346                SRDLIN: MOV     R3,-(SP)        ;;SAVE R3
2356  013320  012703  013424        1$:     MOV     #TTYIN,R3       ;;GET ADDRESS
2357  013324  022703  013434        2$:     CMP     #TTYINW.,R3     ;;BUFFER FULL?
```

```
2358  013330  101405                          BLOS    4$              ;;BR IF YES
2359  013332  104406                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2360  013334  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
2361  013336  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2362  013342  001003                          BNE     3$              ;;SKIP IF NOT
2363  013344  104400  001170          4$:     TYPE    $QUES           ;;TYPE A '?'
2364  013350  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2365  013352  111337  013422          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2366  013356  104400  013422                  TYPE    ,9$
2367  013362  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2368  013366  001356                          BNE     2$              ;;LOOP IF NOT RETURN
2369  013370  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2370  013374  104400  001172                  TYPE    $LF             ;;TYPE A LINE FEED
2371  013400  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2372  013402  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2373  013404  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2374  013412  012766  013424  000004          MOV     #STTYIN,4(SP)
2375  013420  000002                          RTI                     ;;RETURN
2376  013422     000          9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2377  013423     000                  .BYTE   0               ;;TERMINATOR
2378  013424  000010          STTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2379  013434  052536  005015  000   $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
2380  013441     136  006507  000012$CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
2381  013446  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
2382  013454  020075     000
2383  013457     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
2384  013464  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.
                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED, THE COMPLETE NUMBER MUST
                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

013470  011646                 $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
013472  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
013500  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013502  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013504  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
013506  104407                 1$:     RDLIN                   ;;READ AN ASCIZ LINE
013510  012600                         MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
013512  010037  013616                 MOV     R0,5$           ;;AND SAVE IT
013516  005001                         CLR     R1              ;;CLEAR DATA WORD
013520  005002                         CLR     R2
013522  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
013524  001420                         BEQ     3$              ;;IF ZERO GET OUT
013526  122716  000060                 CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
013532  003026                         BGT     4$              ;;IS AN OCTAL DIGIT
013534  122716  000067                 CMPB    #'7,(SP)
013540  002423                         BLT     4$
013542  006301                         ASL     R1              ;;*2
013544  006102                         ROL     R2
013546  006301                         ASL     R1              ;;*4
013550  006102                         ROL     R2
013552  006301                         ASL     R1              ;;*8
013554  006102                         ROL     R2
013556  042716  177770                 BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
013562  062601                         ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
013564  000756                         BR      2$              ;;LOOP
013566  005726                 3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
013570  010166  000012                 MOV     R1,12(SP)       ;;SAVE THE RESULT
013574  010237  013626                 MOV     R2,$HIOCT
013600  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
013602  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
013604  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
013606  000002                         RTI                     ;;RETURN
013610  005726                 4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
013612  105010                         CLRB    (R0)            ;;SET A TERMINATOR
013614  104400                         TYPE                    ;;TYPE UP THRU THE BAD CHAR.
013616  000000                 5$:     .WORD   0
013620  104400  001170                 TYPE    $QUES           ;;"?" "CR" & "LF"
013624  000730                         BR      1$              ;;TRY AGAIN
013626  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
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

013644                          $SCOPE:
013644  104405                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
013646  104405                          CKSWR
013650  032777  040000  165262  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
013656  001062                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
013660  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
013662  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
013666  012737  013706  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
013674  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
013700  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
013704  000431                          BR      $SVLAD          ;;GO TO THE NEXT TEST
013706  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
013710  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
013714  000417                          BR      7$              ;;LOOP ON THE PRESENT TEST
013716                          6$:;#####END OF CODE FOR THE XOR TESTER#####
013716  032777  000400  165214          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
013724  001404                          BEQ     2$              ;;BR IF NO
013726  127737  165206  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
013734  001433                          BEQ     $OVER           ;;BR IF YES
013736  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
013742  001412                          BEQ     $SVLAD          ;;BR IF NO
013744  032777  001000  165166          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
013752  001404                          BEQ     4$              ;;BR IF NO
013754  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
013762  000420                          BR      $OVER
013764  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
013770  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
013774  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
014002  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
014006  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
014012  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
014016  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
014024  013777  001102  165110  $OVER:  MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER
014032  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
014036  000002                          RTI                     ;;FIXES PS
```

```
2492                                    .SBTTL  APT COMMUNICATIONS ROUTINE
2493
2494                                    ;;*************************************************************
2495  014040  112737  000001  014304  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2496  014046  112737  000001  014302  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2497  014054  000403                  BR      $ATYC
2498  014056  112737  000001  014304  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2499  014064                          $ATYC:
2500  014064  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2501  014066  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2502  014070  105737  014302                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2503  014074  001450                          BEQ     5$              ;;IF NOT:  BR
2504  014076  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2505  014104  001031                          BNE     3$              ;;IF NOT:  BR
2506  014106  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2507  014114  001425                          BEQ     3$              ;;IF NOT:  BR
2508  014116  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2509  014122  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2510  014130  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2511  014134  001375                          BNE     1$              ;;IF NOT:  WAIT
2512  014136  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2513  014142  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2514  014144  001376                          BNE     2$
2515  014146  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2516  014152  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2517  014154  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2518  014160  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2519  014166  000413                          BR      5$
2520  014170  017637  000004  014214  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2521  014176  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2522  014204  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2523  014210  004737  014634                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
2524  014214  000000                  4$:     .WORD   0
2525  014216                          5$:
2526  014216  105737  014304          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2527  014222  001416                          BEQ     12$             ;;IF NOT:  BR
2528  014224  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
2529  014230  001413                          BEQ     12$             ;;IF NOT:  BR
2530  014232  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2531  014236  001375                          BNE     11$             ;;IF NOT:  WAIT
2532  014240  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
2533  014246  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2534  014254  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2535  014260  105037  014304          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
2536  014264  105037  014303                  CLRB    $LFLG           ;;CLEAR LOG FLAG
2537  014270  105037  014302                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2538  014274  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2539  014276  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2540  014300  000207                          RTS     PC              ;;RETURN
2541  014302  000             $MFLG:  .BYTE   0               ;;MESSG. FLAG
2542  014303  000             $LFLG:  .BYTE   0               ;;LOG FLAG
2543  014304  000             $FFLG:  .BYTE   0               ;;FATAL FLAG
2544          014306                  .EVEN
2545          000200          APTSIZE=200
2546          000001          APTENV=001
2547          000100          APTSPOOL=100
```

```
25:18          000040                  APTCSUP=040
```

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

014306                                  $ERROR:
014306  104405                                 CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
014310  113737  001102  001314                 MOVB    $TSTNM,TSTNUM
014316  105237  001103          7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
014322  001775                          BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
014324  013777  001102  164610          MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
014332  005237  001112                  INC     $ERTTL                  ;;INC THE ERROR COUNT
014336  011637  001116                  MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
014342  162737  000002  001116          SUB     #2,$ERRPC
014350  117737  164542  001114          MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
014356  032777  020000  164554          BIT     #BIT13,@$WR             ;;SKIP TYPEOUT IF SET
014364  001004                          BNE     20$                     ;;SKIP TYPEOUTS
014366  004737  014500                  JSR     PC,$ERRTYP              ;;GO TO USER ERROR ROUTINE
014372  104400  001171                  TYPE    ,$CRLF
014376                                  20$:
014376  122737  000001  001214                 CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
014404  001007                          BNE     2$                      ;;NO,SKIP APT ERROR REPORT
014406  113737  001114  014420          MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
014414  004737  014056                  JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
014420  000                     21$:    .BYTE   0
014421  000                             .BYTE   0
014422  000777                  22$:    BR      22$                     ;;APT ERROR LOOP
014424  005777  164510          2$:     TST     @$WR                    ;;HALT ON ERROR
014430  100002                          BPL     3$                      ;;SKIP IF CONTINUE
014432  000000                          HALT                            ;;HALT ON ERROR!
014434  104405                          CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
014436  032777  001000  164474  3$:     BIT     #BIT09,@$WR             ;;LOOP ON ERROR SWITCH SET?
014444  001402                          BEQ     4$                      ;;BR IF NO
014446  013716  001110                  MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
014452  005737  001166          4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
014456  001402                          BEQ     5$                      ;;BR IF NONE
014460  013716  001166                  MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
014464                                  5$:
014464  022737  006776  000042                 CMP     #$ENDAD,@#42            ;;ACT-11 AUTO-ACCEPT?
014472  001001                          BNE     6$                      ;;BRANCH IF NO
014474  000000                          HALT                            ;;YES
014476                                  6$:
014476  000002                          RTI                             ;;RETURN
```

```
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

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

014500                          $ERRTYP:
014500  104400  001171                  TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
014504  010046                  MOV     R0,-(SP)        ;;SAVE R0
014506  005000                  CLR     R0              ;;PICKUP THE ITEM INDEX
014510  153700  001114          BISB    @#$ITEMB,R0
014514  001004                  BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                        ;;TYPE THE PC OF THE ERROR
014516  013746  001116          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                        ;;ERROR ADDRESS
014522  104401                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014524  000426                  BR      6$              ;;GET OUT
014526  005300          1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
014530  006300                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
014532  006300                  ASL     R0
014534  006300                  ASL     R0
014536  062700  001256          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
014542  012037  014552          MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
014546  001404                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
014550  104400                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
014552  000000          2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
014554  104400  001171          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
014560  012037  014570          MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
014564  001404                  BEQ     5$              ;;SKIP TYPEOUT IF 0
014566  104400                  TYPE                    ;;TYPE THE "DATA HEADER"
014570  000000          4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
014572  104400  001171          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
014576  011000          5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
014600  001004                  BNE     7$              ;;GO TYPE THE DATA
014602  012600          6$:     MOV     (SP)+,R0        ;;RESTORE R0
014604  104400  001171          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
014610  000207                  RTS     PC              ;;RETURN
014612                  7$:
014612  013[illegible]          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
014614  104401                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014616  005710                  TST     (R0)            ;;IS THERE ANOTHER NUMBER?
014620  001770                  BEQ     6$              ;;BR IF NO
014622  104400  014630          TYPE    ,8$             ;;TYPE TWO(2) SPACES
014626  000771                  BR      7$              ;;LOOP
014630  020040      000 8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
014634                          .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
                                ;*

014634  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
014640  100002                          BPL     1$              ;;BR IF YES
014642  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
014644  000430                          BR      3$              ;;LEAVE
014646  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
014650  017600  000002                  MOV     2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
014654  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
014662  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
014664  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
014672  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
014674  010037  014704                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
014700  004737  014046                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
014704  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
014706  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
014714  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
014716  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
014720  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
014722  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
014724  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
014726  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
014732  000002                          RTI                     ;;RETURN
014734  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
014740  001430                          BEQ     8$
014742  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
014746  001006                          BNE     5$
014750  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
014752  104400                          TYPE                    ;;TYPE A CR AND LF
014754  001171                          $CRLF
014756  105037  015112                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
014762  000755                          BR      2$              ;;GET NEXT CHARACTER
014764  004737  015046          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
014770  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
014774  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
014776  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
015002  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
015006  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
015010  004737  015046                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
```

```
2705  015014  105337  015112          DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2706  015020  000770                  BR      7$              ;;LOOP
2707
2708                                  ;HORIZONTAL TAB PROCESSOR
2709
2710  015022  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2711  015026  004737  015046          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2712  015032  132737  000007  015112          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2713  015040  001372                          BNE     9$              ;;TAB STOP
2714  015042  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2715  015044  000724                          BR      2$              ;;GET NEXT CHARACTER
2716  015046  105777  164076          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2717  015052  100375                          BPL     $TYPEC
2718  015054  116677  000002  164070          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2719  015062  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2720  015070  001003                          BNE     1$              ;;BRANCH IF NO
2721  015072  105037  015112                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2722  015076  000406                          BR      $TYPEX          ;;EXIT
2723  015100  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2724  015106  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2725  015110  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2726  015112  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2727  015114  000207          $TYPEX: RTS     PC
2728
```

```
                                .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

015116  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
015122  116637  000001  015341          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
015130  112637  015343                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
015134  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
015140  000406                          BR      $TYPON
015142  112737  000001  015341  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
015150  112737  000006  015343          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
015156  112737  000005  015340  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
015164  010346                          MOV     R3,-(SP)        ;;SAVE R3
015166  010446                          MOV     R4,-(SP)        ;;SAVE R4
015170  010546                          MOV     R5,-(SP)        ;;SAVE R5
015172  113704  015343                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
015176  005404                          NEG     R4
015200  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
015204  110437  015342                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
015210  113704  015341                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
015214  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
015220  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
015222  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
015224  000404                          BR      3$              ;;GO DO MSB
015226  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
015230  006105                          ROL     R5
015232  006105                          ROL     R5
015234  010503                          MOV     R5,R3
015236  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
015240  105337  015342                  DECB    $OMODE          ;;TYPE THIS DIGIT?
015244  100016                          BPL     7$              ;;BR IF NO
015246  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
015252  001002                          BNE     4$              ;;TEST FOR 0
015254  005704                          TST     R4              ;;SUPPRESS THIS 0?
```

```
2785  015256  001403                          BEQ     5$              ;;BR IF YES
2786  015260  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2787  015262  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2788  015266  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2789  015272  110337  015336                  MOVB    R3,8$           ;;SAVE FOR TYPING
2790  015276  104400  015336                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2791  015302  105337  015340          7$:     DECB    $OCNT           ;;COUNT BY 1
2792  015306  003347                          BGT     2$              ;;BR IF MORE TO DO
2793  015310  002402                          BLT     6$              ;;BR IF DONE
2794  015312  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2795  015314  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2796  015316  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2797  015320  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2798  015322  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2799  015324  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2800  015332  012616                          MOV     (SP)+,(SP)
2801  015334  000002                          RTI                     ;;RETURN
2802  015336     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2803  015337     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2804  015340     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2805  015341     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2806  015342  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2807                                  .SBTTL  TRAP DECODER
2808
2809                                  ;;**********************  **************************************
2810                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2811                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2812                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2813                                  ;*GO TO THAT ROUTINE.
2814
2815  015344  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2816  015346  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
2817  015352  005740                          TST     -(R0)           ;;BACKUP BY 2
2818  015354  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2819  015356  006300                          ASL     R0              ;;POSITION FOR INDEXING
2820  015360  016000  015366                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2821  015364  000200                          RTS     R0              ;;GO TO ROUTINE
2822
2823                                  .SBTTL  TRAP TABLE
2824
2825                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2826                                  ;*BY THE "TRAP" INSTRUCTION.
2827
2828                                  ;       ROUTINE
2829                                  ;       -------
2830  015366                          $TRPAD:
2831  015366  014634                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
2832  015370  015142                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2833  015372  015116                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2834  015374  015156                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2835
2836  015376  ^12764                          $GTSWR  ;;CALL=GTSWR    TRAP+4(104404)  GET SOFT-SWR SETTING
2837
2838  015400  012714                          $CKSWR  ;;CALL=CKSWR    TRAP+5(104405)  TEST FOR CHANGE IN SOFT-SWR
2839  015402  013176                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
2840  015404  013316                          $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
```

```
2841  015406  013470                          $RDOCT  ;;CALL=RDOCT   TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
2842                                  .SBTTL  POWER DOWN AND UP ROUTINES
2843
2844                                  ;;**********************************************************************
2845                                  ;POWER DOWN ROUTINE
2846  015410  012737  015554  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2847  015416  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2848  015424  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2849  015426  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2850  015430  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2851  015432  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2852  015434  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2853  015436  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2854  015440  017746  163474                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
2855  015444  010637  015560                  MOV     SP,$SAVR6       ;;SAVE SP
2856  015450  012737  015462  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2857  015456  000000                          HALT
2858  015460  000776                          BR      .-2             ;;HANG UP
2859
2860                                  ;;**********************************************************************
2861                                  ;POWER UP ROUTINE
2862  015462  012737  015554  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
2863  015470  013706  015560                  MOV     $SAVR6,SP       ;;GET SP
2864  015474  005037  015560                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
2865  015500  005237  015560          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
2866  015504  001375                          BNE     1$              ;;OF  WORD
2867  015506  012677  163426                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
2868  015512  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2869  015514  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
2870  015516  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2871  015520  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2872  015522  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2873  015524  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2874  015526  012737  015410  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2875  015534  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2876  015542  104400                          TYPE                    ;;REPORT THE POWER FAILURE
2877  015544  011202                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
2878  015546  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RBEGIN
2879  015550  001402                  $PWRAD: .WORD   RBEGIN          ;;RESTART ADDRESS
2880  015552  000002                          RTI
2881  015554  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE HAS STARTED
2882  015556  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2883  015560  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
2884  015562  000000                  BUFFER: 0
2885          000001                          .END
```

```
ABASE = 176500      537#   646    687
ACOM1 = 000000      646    689
ACOM2 = 000000      646
ACPUOP= 000000      646    661
ADDLIN= 002000      733#   883    887    965    969    1338   1391   1588   1592   1605   1609   1611   1834
ADDWD = 000777      646
ADDW1 = 000000      646
ADDW10= 000000      646
ADDW11= 000000      646
ADDW12= 000000      646
ADDW13= 000000      646
ADDW14= 000000      646
ADDW15= 000000      646
ADDW2 = 000000      646
ADDW3 = 000000      646
ADDW4 = 000000      646
ADDW5 = 000000      646
ADDW6 = 000000      646
ADDW7 = 000000      646
ADDW8 = 000000      646
ADDW9 = 000000      646
ADELAY  011024      2027#
ADEVCT= 000000      646    652
ADEVM = 000000      646    688
AENV  = 000000      646    657
AENVM = 000000      646    658
AFATAL= 000000      646    649
AMADR1= 000000      646    674
AMADR2= 000000      646    678
AMADR3= 000000      646    681
AMADR4= 000000      646    684
AMAMS1= 000000      646    668
AMAMS2= 000000      646    676
AMAMS3= 000000      646    679
AMAMS4= 000000      646    682
AMSG    011070      867    949    1028   1079   1129   1176   1223   1272   1319   1372   1422   1485   1964
                    1967   2041#
AMSGAD= 000000      646    654
AMSGLG= 000000      646    655
AMSGTY= 000000      646    648
AMTYP1= 000000      646    669
AMTYP2= 000000      646    677
AMTYP3= 000000      646    680
AMTYP4= 000000      646    683
ANESC   010546      1866#  1894#  1896#  1900   1960#
APASS = 000000      646    651
APRIOR= 000000      646
APTCSU= 000040      2548#  2680
APTENV= 000001      2504   2546#  2578   2673
APTSIZ= 000200      824    2545#
APTSPO= 000100      2506   2547#  2675
ASWREG= 000000      646    659
ATESTN= 000000      646    650
AUNIT = 000000      646    653
AUSWR = 000000      646    660
AVECT1= 000000      646    685
```

```
AVECT2= 000000            646     686
BRGE    007772            883*    886*    887     916*    919*    920     965*    968*    969     996*    999*    1053*   1058*
                         1103*   1107*   1197*   1202*   1244*   1249*   1588*   1591*   1592    1605*   1608*   1609    1611*
                         1747*   1752*   1757*   1758*   1761*   1762    1786*   1791*   1819*   1824*   1829*   1830*   1833*
                         1834    1840*   1845    1857
BASE1   007774            1841*   1846*   1847*   1854*   1855*   1856*   1859*   1860
BEGIN   001346            559     775*
BEGIN2  001412            561     783*
BIT0  = 000001            522*    878     880     960     1050    1100    1194    1241    1583    1585    1742    1782    1810
BIT00 = 000001            512*    522
BIT01 = 000002            511*    521
BIT02 = 000004            510*    520
BIT03 = 000010            509*    519
BIT04 = 000020            508*    518
BIT05 = 000040            507*    517
BIT06 = 000100            506*    516
BIT07 = 000200            505*    515
BIT08 = 000400            504*    514     2472
BIT09 = 001000            503*    513     2478    2589
BIT1  = 000002            521*    962     1050    1143    1148    1194    1285    1290    1332    1337    1362    1435    1449
                         1585
BIT10 = 002000            502*    733     1857    1888
BIT11 = 004000            501*    1884
BIT12 = 010000            500*    1859    2016
BIT13 = 020000            499*    2573
BIT14 = 040000            498*    2458
BIT15 = 100000            497*    1149    1291    1363    1386    1415    1436    1499
BIT2  = 000004            520*    1100    1148    1241    1290    1385    1390    1414    1498    1512
BIT3  = 000010            519*    1194    1285    1290
BIT4  = 000020            518*    876     958     1036    1047    1087    1098    1138    1184    1231    1241    1280    1290
                         1327    1380    1430    1493    1558
BIT5  = 000040            517*    874     876     878     880     956     958     960     962     1034    1036    1045    1047
                         1050    1085    1087    1096    1098    1100    1136    1138    1182    1184    1194    1229    1231
                         1241    1278    1280    1325    1327    1378    1380    1426    1430    1491    1493    1556    1558
                         1583    1585    1742    1782    1810    1815
BIT6  = 000100            516*
BIT7  = 000200            515*    1040    1091    1188    1235
BIT8  = 000400            514*
BIT9  = 001000            513*
BPTVEC= 003014            529*
BUFFER  013562            870     912     952     992     1030    1041    1081    1092    1132    1178    1190    1225    1237
                         1274    1321    1374    1424    1458    1487    1521    1552    1563    1579    1600    1738    1778
                         1806    1669    2072    2684*
BUSSTR  001336            770*    778
CH_AR = 104405            1872    1913    2456    2457    2564    2588    2838*
COPY  = 000135            753*    890     924     972     1003    1152    1294    1341    1394    1460    1523    1615    1882
CP    = 000015            437*    2719    2729
CRLF  = 000200            438*    2690    2729
CUFCRO  012461            1320    2210*
CUFCR1  012512            1373    2216*
CURSOR  007574            1336    1361    1389    1413    1804*
DAHL    012013            868     2152*
DAVL    012045            950     2157*
DDISP = 177570            444*    625     812
DELAY   010742            909     943     991     1022    1070    1120    1171    1214    1261    1313    1365    1417    1479
                         1542    1635    2010*
```

DZVTDB.SRC CROSS REFERENCE TABLE -- USER SYMBOLS

```
DH1     011714            717    2140#
DH4     011745            711    2145#
DISPLA  001142            625#    912*    820*   2489*   2568*
DISPRE  000174            556#    820
DSPCH   010676           1534    1986    1987    1988    1990#
DSWR  = 177570            443#    624     811
DT1     012670            718    2241#
DT4     012700            712    2243#
EMTVEC= 000030            532#    796*    797*
EM1     011516           2117#
EM2     011563            716    2124#
EM3     011655            722    2134#
EM4     011612            710    2128#
ERRVEC= 000004            525#    778*    779*    809     810*    821*   2463    2464*   2466*   2469*
ESC   = 000033            735#    871     889     923     953     971    1002    1031    1042    1086    1093    1133    1151
                         1179    1191    1226    1238    1275    1293   1322    1340    1375    1393    1425    1459    1488
                         1522    1553    1559    1614    1739    1779   1807
FINDT   010562           1942    1949    1966#   1970
FINTA   010572           1941*   1948*   1968#
FIRST   001036            755#    828*    844*    845     847     851
FINDA   010554           1964#   1977    1979    1985
GETCHR  011110           1969    2050#
GIN     001420            782     784#
GINA    001426            780     785#
GMS   = ****** U          555    2831    2832    2833    2834    2836   2838    2839    2840    2841
GMDA1   012232           1130    2180#
GTSWR = 104404           2836#
HGDA1   012420           1273    2203#
HT    = 000011            435#   2688    2729
IOTVEC= 000020            530#    794*    795*    830*
LAST    001310            757#    848*   1649    1651
LDC0  = 000103            745#   1337    1362
LDC1  = 000113            746#   1390    1414
LDE0  = 000101            739#    873     877     955     959    1033    1044   1049    1084    1095    1099    1135    1181
                         1193    1228    1240    1277    1324   1377    1427   1490    1555    1582    1741    1781    1809
LDE1  = 000111            740#    875     879     957     961    1035    1046   1086    1097    1137    1183    1230    1279
                         1326    1379    1429    1492    1557   1584    1812
LDG0  = 000102            742#   1057    1143    1201    1285   1332    1435   1449
LDG1  = 000112            743#   1107    1148    1248    1290   1385    1498   1512
LF    = 000012            436#   2723    2729
LHV0  = 000104            748#    892     914    1587
LHV1  = 000114            749#    894     994    1634
LN0   = 000100            737#    881     892     893     894     895    896     897     898     899     900     901
                          902     923     924     925     926     927    928     929     930     931
                          932     933     934     935     936     937    938     939     940     963     973     974     975
                          976     977     978     979     980     981    982     983     984     985     986     987     988
                          993    1004    1005    1006    1007    1008   1009    1010    1011    1012    1013    1014    1015
                         1016    1017    1018    1019    1051    1056   1101    1106    1153    1154    1155    1156    1157
                         1158    1159    1160    1161    1162    1163   1164    1165    1166    1167    1168    1195    1200
                         1242    1247    1295    1296    1297    1298   1299    1300    1301    1302    1303    1304    1305
                         1306    1307    1308    1309    1310    1342   1343    1344    1345    1346    1347    1348    1349
                         1350    1351    1352    1353    1354    1355   1356    1357    1395    1396    1397    1398    1399
                         1400    1401    1402    1403    1404    1405   1406    1407    1408    1409    1410    1461    1462
                         1463    1464    1465    1466    1467    1468   1469    1470    1471    1472    1473    1474    1475
                         1476    1524    1525    1526    1527    1528   1529    1530    1531    1532    1533    1534    1535
                         1536    1537    1538    1539    1586    1603   1616    1617    1618    1619    1620    1621    1622
```

```
                        1623    1624    1625    1626    1627    1628    1629    1630    1631    1745    1750    1784    1789
                         181    1822
LOOP    010544             6    1940*   1947*   1959*   2440
LSC     = 000112        751*    1052    1132    1196    1243    1746    1785    1818
M30HOZ= 000354          729*     687     920    1149    1291    1386    1436    1499    1762
MAXHZC= 000400          730*
MAXVRT= 001000          731*     965     996    1060    1110    1204    1251    1440    1503    1834
MD0     011277         1941    2042*
MD1     011362         1948    1958    2101*
MD2     011446         1965    2110*
MSCOPE  013630          830    2438*
MTOB    011166         2042    2071*
NOEXIT  010550         1878*   1893*   1924*   1930*   1952    1971*
PC      =X000007        456*    884*    908*    904*    917*    943*    966*    990*    991*    997*   1021*   1022*
                       1038*   1054*   1059*   1067*   1070*   1083*   1104*   1109*   1117*   1120*   1140*   1145*   1170*
                       1171*   1186*   1198*   1203*   1211*   1214*   1233*   1245*   1250*   1258*   1261*   1282*   1287*
                       1312*   1313*   1329*   1334*   1359*   1364*   1385*   1392*   1387*   1412*   1416*   1417*   1432*
                       1437*   1453*   1478*   1479*   1495*   1500*   1516*   1541*   1542*   1562*   1573*   1563*   1595*
                       1606*   1612*   1633*   1654*   1659*   1662*   1669*   1674    1748*   1753*   1787*   1792*   1820*
                       1825*   1861*   1956*   1964*   2022*   2035*   2044*   2057*   2064*   2300*   2523*   2540*   2575*
                       2581*   2639*   2678*   2697*   2704*   2711*   2725*   2727*   2878
PIRQ    = 177772        442*
PIRQVE= 000240          536*
PR0     = 000000        459*
PR1     = 000040        460*
PR2     = 000100        461*
PR3     = 000140        462*
PR4     = 000200        463*
PR5     = 000240        464*
PR6     = 000300        465*
PR7     = 000340        465*
PS      = 177776        439*     440
PSW     = 177776        440*
PWRMSG  011202         2081*   2877
PWRVEC= 000024          531*     800*    801*   2846*   2847*   2856*   2862*   2874*   2875*
REGIN   001402          560      781*   2879
ROCHR   = 104406       2359    2839*
ROLIN   = 104407       2404    2840*
ROOCT   = 104410        843    2841*
RESVEC= 000010          526*
RSTRT   002016          853*    1654
RSTRTA  002010          840     851*    1675
R0      =X000000        447*    853*    854*    855*    856*    857*    858*    859*    860*    870*    871*    872*    873*
                        874*    875*    876*    877*    878*    879*    880*    881*    882*    885*    889*    890*    891*
                        892*    893*    894*    895*    896*    897*    898*    899*    900*    901*    902*    903*    904*
                        905*    906*    907*    912*    913*    914*    918*    923*    924*    925*    926*    927*    928*
                        929*    930*    931*    932*    933*    934*    935*    936*    937*    938*    939*    940*    941*
                        952*    953*    954*    955*    956*    957*    958*    959*    960*    961*    962*    963*    964*
                        967*    971*    972*    973*    974*    975*    976*    977*    978*    979*    980*    981*    982*
                        983*    984*    985*    986*    987*    988*    989*    992*    993*    994*    998*   1002*   1003*
                       1004*   1005*   1006*   1007*   1008*   1009*   1010*   1011*   1012*   1013*   1014*   1015*   1016*
                       1017*   1018*   1019*   1020*   1030*   1031*   1032*   1033*   1034*   1035*   1036*   1037*   1041*
                       1042*   1043*   1044*   1045*   1046*   1047*   1049*   1050*   1051*   1052*   1055*   1056*   1057*
                       1061*   1066*   1081*   1082*   1083*   1084*   1085*   1086*   1087*   1088*   1092*   1093*   1094*
                       1095*   1096*   1097*   1098*   1099*   1100*   1101*   1102*   1105*   1106*   1107*   1111*   1116*
                       1132*   1133*   1134*   1135*   1136*   1137*   1138*   1139*   1151*   1152*   1153*   1154*   1155*
```

```
                    1156* 1157* 1158* 1159* 1160* 1161* 1162* 1163* 1164* 1165* 1166* 1167* 1168*
                    1169* 1178* 1179* 1180* 1181* 1182* 1183* 1184* 1185* 1190* 1191* 1192* 1193*
                    1194* 1195* 1196* 1199* 1200* 1201* 1205* 1210* 1225* 1226* 1227* 1228* 1229*
                    1230* 1231* 1232* 1237* 1238* 1239* 1240* 1241* 1242* 1243* 1246* 1247* 1248*
                    1252* 1257* 1274* 1275* 1276* 1277* 1278* 1279* 1280* 1281* 1293* 1294* 1295*
                    1296* 1297* 1298* 1299* 1300* 1301* 1302* 1303* 1304* 1305* 1306* 1307* 1308*
                    1309* 1310* 1311* 1321* 1322* 1323* 1324* 1325* 1326* 1327* 1328* 1340* 1341*
                    1342* 1343* 1344* 1345* 1346* 1347* 1348* 1349* 1350* 1351* 1352* 1353* 1354*
                    1355* 1356* 1357* 1358* 1374* 1375* 1376* 1377* 1378* 1379* 1380* 1381* 1393*
                    1394* 1395* 1396* 1397* 1398* 1399* 1400* 1401* 1402* 1403* 1404* 1405* 1406*
                    1407* 1408* 1409* 1410* 1411* 1424* 1425* 1426* 1427* 1428* 1429* 1430* 1431*
                    1441* 1443* 1444  1458* 1459* 1460* 1461* 1462* 1463* 1464* 1465* 1466* 1467*
                    1468* 1469* 1470* 1471* 1472* 1473* 1474* 1475* 1476* 1477* 1487* 1488* 1489*
                    1490* 1491* 1492* 1493* 1494* 1504* 1506* 1507  1521* 1522* 1523* 1524* 1525*
                    1526* 1527* 1528* 1529* 1530* 1531* 1532* 1533* 1534* 1535* 1536* 1537* 1538*
                    1539* 1540* 1552* 1553* 1554* 1555* 1556* 1557* 1558* 1559* 1560* 1561* 1562*
                    1564* 1565* 1566* 1567* 1569* 1572* 1579* 1580* 1581* 1582* 1583* 1584* 1585*
                    1586* 1587* 1590* 1594* 1600* 1601* 1602* 1603* 1604* 1607* 1613* 1614* 1615*
                    1616* 1617* 1618* 1619* 1620* 1621* 1622* 1623* 1624* 1625* 1626* 1627* 1629*
                    1629* 1630* 1631* 1632* 1666* 1669  1738* 1739* 1740* 1741* 1742* 1743* 1745*
                    1746* 1749* 1750* 1751* 1754* 1766* 1776  1778* 1779* 1780* 1781* 1782* 1783*
                    1784* 1785* 1788* 1789* 1790* 1793* 1791* 1795* 1806* 1807* 1808* 1809* 1810*
                    1811* 1812* 1815* 1816* 1817* 1818* 1821* 1822* 1823* 1826* 1836* 1854  1869*
                    1878  1882  1885  1955* 1974  1975* 1976  1978  1980* 1981* 1982  1983* 1984
                    1986  1987  1988  2055* 2071* 2073  2401  2405* 2406  2409  2429* 2432* 2500
                    2508* 2512  2513  2515* 2516* 2517  2539* 2611  2612* 2613* 2620* 2621* 2622*
                    2623* 2624* 2625  2630  2635* 2637* 2641  2643  2671  2672* 2677  2682  2685*
                    2815  2816* 2817  2818* 2819* 2820* 2821* 2848  2873*
R1      =%000001     448*  835   918   967   998   1035  1061  1105  1111  1199  1205  1246  1252
                    1440* 1446  1454* 1503* 1509  1517* 1568* 1570* 1590  1607  1613  1749  1754
                    1777  1786  1793  1795* 1813* 1814* 1816  1821  1826  1848* 1849* 1850* 1851*
                    1852* 1853* 1854  1860* 1865  1879* 1940  1897  1954* 2072* 2073* 2402  2407*
                    2415* 2417* 2419* 2422* 2425  2428* 2501  2538* 2849  2872*
R2      =%000002     449*  1551* 1574* 1645* 1846  1848  2403  2408* 2416* 2418* 2420* 2426  2427*
                    2850  2871*
R3      =%000003     450*  1439* 1456* 1503* 1519* 2355  2356* 2357  2360* 2361  2365  2367  2369*
                    2371* 2763  2772* 2774* 2779* 2782* 2787* 2788* 2789  2798* 2851  2870*
R4      =%000004     451*  2764  2766* 2767* 2768* 2783  2770* 2784  2796* 2794* 2797* 2852  2869*
R5      =%000005     452*  867*  949*  1028* 1078* 1129* 1142* 1147* 1176* 1223* 1272* 1284* 1289*
                    1319* 1331* 1336* 1361* 1372* 1384* 1393* 1413* 1422* 1434* 1448* 1485* 1497*
                    1511* 1736  1737  1767* 1773  1804  1775  1797* 1804  1805  1837* 1964* 1967*
                    2041  2042* 2045* 2071  2075* 2765  2771* 2773* 2775* 2776* 2777* 2778  2796*
                    2853  2858*
R6      =%000006     453*  455   788*  783*  790
R7      =%000007     454*  456
SAVE4   001334       768*  777*  784*  831
SCOPE0  012543       1423  2222*
SCOPE1  012615       1486  2231*
SHALO   012276       1177  2187*
SHALI   012347       1224  2195*
SHLO    012076       1029  2162*
SHLI    012154       1080  2171*
SHUFF   007776       834   917   956   997   1054  1059  1104  1109  1198  1203  1245  1250  1589
                    1606  1612  1748  1753  1787  1792  1820  1825  1845*
SINEND  007241       1441  1443  1504  1536  1732*
SP      =%000006     455*  770   792*  809*  817*  821   844   1776* 1777* 1795  1796  1864* 1865*
```

```
                    1954    1955    1966*   2056*   2260*   2261*   2262    2269*   2272*   2273*   2277*   2278*   2282
                    2285*   2289    2291    2293    2294*   2301    2303    2305*   2306    2308*   2309*   2310*   2311*
                    2312*   2327*   2328*   2331*   2332*   2333    2337*   2338*   2339    2342    2344    2346*   2355*
                    2360    2371    2372*   2373*   2374*   2399*   2400*   2401*   2402*   2403*   2405    2409*   2411
                    2413    2421*   2422    2424    2425*   2427    2428    2429    2431    2463*   2466    2468    2469
                    2485    2486    2490*   2500*   2501*   2508    2509*   2520    2521*   2522*   2532    2533*   2538
                    2539    2570    2591*   2594*   2611*   2616*   2637    2641*   2671*   2672    2682*   2684    2685
                    2686*   2698    2690    2692    2698    2700*   2702*   2710*   2714    2718    2719    2723    2755*
                    2756    2757    2758*   2763*   2764*   2765*   2771    2796    2797    2798    2799*   2800*   2815*
                    2816    2848*   2849*   2850*   2851*   2952*   2853*   2854*   2855    2863*   2967    2868    2869
                    2870    2871    2872    2673    2878*
STACK = 001100       430*    792     1966
STCOMD  007436      1448    1511    1773*
STKLMT= 177774       441*
SUBTST  001320       762*    2010    2027
SVR     001140       624*    790     811*    813     819*    826*    837     839     1894    1888    1934    2016    2256
                    2293*   2438    2458    2472    2474    2478    2573    2585    2589    2854    2867*
SWREG   000176       557*    819     837     2256    2269
SW0     = 000001     494*
SW00    = 000001     484*    494
SW01    = 000002     483*    493
SW02    = 000004     482*    492
SW03    = 000010     481*    491
SW04    = 000020     480*    490
SW05    = 000040     479*    489
SW06    = 000100     478*    488
SW07    = 000200     477*    487
SW08    = 000400     476*    486
SW09    = 001000     475*    485
SW1     = 000002     493*
SW10    = 002000     474*
SW11    = 004000     473*
SW12    = 010000     472*
SW13    = 020000     471*
SW14    = 040000     470*
SW15    = 100000     469*
SW2     = 000004     492*
SW3     = 000010     491*
SW4     = 000020     490*
SW5     = 000040     489*
SW6     = 000100     488*
SW7     = 000200     487*
SW8     = 000400     486*
SW9     = 001000     485*
TBITVE= 000014       527*
TERM    010552      1867*   1925    1950*   1962*
TIMEO   001316       761*   1907    2050
TIME1   011162      1907*   1914*   2050*   2061*   2066*
TIME2   011164      1908*   1911*   2051*   2059*   2067*
TITLE   011220       834    2084*
TITST   001714       831*
TKVEC = 000060       534*
TPVEC = 000064       535*
TRAPVE= 000034       533*    798*    799*
TRTVEC= 000014       528*
TSTNUM  011314       759*   2241    2243    2565*
```

```
TST1    002064          775     781     862     866#    1990
TST10   004516          1262    1271#   1997
TST11   004736          1318#   1998
TST12   005172          1371#   1999
TST13   005426          1421#   2000
TST14   005724          1484#   2001
TST15   006222          1548#   2002
TST2    002520          948#    1991
TST3    003146          1027#   1992
TST4    003404          1071    1078#   1993
TST5    003642          1121    1138#   1994
TST6    004062          1175#   1995
TST7    004300          1215    1222#   1996
TYPE  = 104400          833     841     1665    2267    2268    2271    2284    2295    2314    2363    2366    2370    2433
                        2435    2576    2610    2627    2629    2632    2634    2638    2645    2693    2790    2831#   2876
TYPOC = 104401          2270    2618    2642    2832#
TYPON = 104403          2834#
TYPOS = 104402          2833#
UPDWN   007242          1142    1147    1284    1289    1331    1384    1434    1497    1736#
VTI9    001326          745#    1920    2055
VTIS    001324          764#    853     1898    1909    2053
VTNOW   001312          758#    851*    854     855     857     859     1651    1653*   2241    2243
VTO8    001332          767#    1897*   1974#
VTOS    001330          766#    1870    1972
WFTEST  001322          763#    776*    783*    2012
WHATO   011466          842     2113#
WHERE   002052          775*    781*    861     862#
XOFF  = 000023          763#    1928
XON   = 000021          763#    1922    1931    1951
XPRNT   010074          908     942     990     1021    1038    1067    1089    1117    1140    1145    1170    1186    1211
                        1233    1258    1282    1287    1312    1329    1334    1359    1364    1382    1387    1412    1416
                        1432    1437    1453    1478    1495    1500    1516    1541    1562    1573    1595    1633    1864#
                        2044
$APTHD  001000          584     590#
$HSTAT= ****** U        2526    2541
$ATYC   014064          2497    2439#
$ATY1   014040          2495#
$ATY3   014046          2496#   2678
$ATY4   014056          2498#   2581
$AUTOB  001134          621#    2264    2385
$POSE   001250          687#    828
$XOB    001122          616#
$JAT    001126          618#    1920*   1921*   1922    1928    1936    1938    2243
$LOW1   001254          689#
$CHARC  015112          2695*   2705*   2712    2721*   2726#
$CRSWR  012714          2256#   2838
$CMTAG  00110C          604#    787     788     796     802     803
$CM1  = 000002          636#    637#    638#
$CM2  = 000004          636#    637#    638#
$CM3  = 000002          634#    636
$CNTLG  013441          2267    2380#
$CNTLU  013434          2284    2379#
$CPUOP  001222          661#
$CRLF   001171          640#    2295    2379    2438    2576    2601    2610    2629    2634    2638    2694    2729
$DEVCT  001204          652#
$DEVM   001252          668#
```

```
$DOAGN  007006          1661    1667    1673#
$ENDAD  006776           570    1669#   2596
$ENDCT  006756           802    1663#
$ENDMG  007015          1665    1677#
$ENULL  007C12          1676#
$ENV    001214           657#   2504    2528    2578    2673
$ENVM   001215           658#    824    2506    2675    2680
$EOP    006674          1647#
$EOPCT  006750           802#   1660#   1664
$ERFLG  001103           607#   2447    2476    2482*   2492    2566#   2601
$ERMAX  001115           613#    804#   2488#   2492
$ERROR  01430C           796    2563#
$ERRPC  001116           614#   2241    2243    2570#   2571#   2572    2601    2616
$ERRTB  001256           706#   2624
$ERRTY  014500          2575    2609#
$ERTTL  001112           611#   2569#   2601
$ESCAP  001166           638#    803#   2487#   2592    2594    2601
$ETABL  001214           656#
$ETEND  001256           596     690#
$FATAL  001176           649#   2532#
$FFLG   014304          2495#   2498#   2526    2535#   2543#
$FILLC  001156           632#    698    2729
$FILLS  001155           631#   2729
$GDADR  001120           615#
$GDDAT  001124           617#   1902#   1931#   1933#   1951#   2243
$GET42  006766          1666#
$GTSWR  012764          2268#   2836
$HD   = 000000           427
$HIBTS  001000           591#
$HIOCT  013626          2426#   2437#
$ICNT   001104           608#
$ILLUP  015554          2846    2862    2881#
$INTAG  001135           622#   2296    2385
$ITEMB  001114           612#   2572#   2580    2601    2613
$LF     001172           641#   2370    2379    2438    2601    2729
$LFLG   014303          2536#   2512#
$LPADR  001106           609#    805#   1966#   2480#   2485#   2490    2492
$LPERR  001110           610#    806#   1987#   2480    2486#   2492    2591
$MADR1  001226           674#
$MADR2  001232           678#
$MADR3  001236           681#
$MADR4  001242           684#
$MAIL   001174           592     596     647#    823     2484    2578    2673
$MAMS1  001224           668#
$MAMS2  001230           676#
$MAMS3  001234           679#
$MAMS4  001240           682#
$MBADR  001002           592#
$MFLG   014306          2496#   2502    2537#   2541#
$MTNUM  013457          2271    2383#
$MSGAD  001210           654#   2512#   2515
$MSGLG  001212           655#   2517#
$MSGTY  001174           648#   2510    2518#   2530    2534#
$MSWR   013446          2268    2381#
$MTYP1  001225           669#
$MTYP2  001231           677#
```

```
$MTYP3  001235            680#
$MTYP4  001241            683#
$NULL   001154            630#  2700    2729
$NWTST= 000001            863#   945#   1024#  1075#  1125#  1172#  1219#  1268#  1315#  1368#  1418#  1481#  1545#
$OCNT   015340           2762*  2791*   2804#
$OMODE  015342           2757*  2761*   2766   2769*  2780*  2806#
$OVER   014024           2441   2459    2475   2481   2489#
$PASS   001202            651#   823*   1657*  1658*  1676   1890
$PASTM  001006            594#
$PWRAD  015550           2879#
$PWRDN  015410            800   2846#   2874
$PWRMG  015544           2877#
$PWRUP  015462           2856   2862#
$QUES   001170            639#  2314    2363   2379   2435   2438   2601   2729
$RDCHR  013176           2327#  2839
$RDDEC= ****** U         2842
$RDLIN  013316           2355#  2340
$RDOCT  013470           2399#  2841
$RDSZ = 000010           2348#
$REGAD  001160            634#
$REGO   001162            636#
$REG1   001164            637#
$STNAD  007010           1675#
$R2A  = ****** U         2642
$SAVRE= ****** U         2842
$SAVR6  015560           2855*  2863    2864*  2865*  2883#
$SCOPE  013644            794   2439    2455#
$SETUP= 000137            708#   793     794    796    798    800    802    803    805   1656   2251   2385   2456
                         2564   2588    2596
$STUP = 177777            708#
$SVLAD  013770           2467   2477    2483#
$SVPC = 000214            568#   573
$SWR  = 161400            417#   427     542    543    544    545    546    547    548    638    639    803    805
                          806    867     949   1028   1079   1129   1176   1223   1272   1319   1372   1422   1485
                         1549   1642    1657   1668   1674   1676   2448   2449   2450   2451   2458   2470   2472
                         2473   2476    2477   2478   2483   2486   2489   2492   2556   2557   2558   2559   2569
                         2573   2585    2589   2601   2880
$SWPEG  001216            659#   826
$SWMSK= 000000            548    549    2451   2452   2474
$TESTN  001200            650#  2484*
$TKB    001146            627#  2249    2260   2277   2331   2337
$TKS    001144            626#  2249    2258   2274   2298*  2329   2335
$TN   = 000016            417#   427     863    867#   945    949#   1024   1028#  1071   1075   1079#  1121   1125
                         1129#  1172    1176#  1215   1219   1223#  1262   1268   1272#  1315   1319#  1368   1372#
                         1418   1422#   1481   1485#  1545   1549#
$TPB    001152            629#  2718*   2729
$TPFLG  001157            633#  2667    2729
$TPS    001150            628#  2716    2729
$TRAP   015344            798   2815#
$TRP  = 000011           2823#  2832#   2833#  2834#  2835#  2836   2837#  2838   2839#  2840#  2841#  2842#
$TPPAD  015366           2820   2830#
$TSTM   001004            593#
$TSTNM  001102            606#  1656*   1982*  2447   2474   2483*  2484   2489   2492   2565   2568   2601
$TTYIN  013424           2356   2357    2374   2378#
$TYPBN= ****** U         2835
$TYPOS= ****** U         2835
```

```
$TYPE   014634            2523    2667#   2823    2831
$TYPEC  015046            2300    2697    2704    2711    2716#   2717
$TYPEX  015114            2722    2724    2727#
$TYPOC  015142            2760#   2832
$TYPON  015156            2759    2762#   2834
$TYPOS  015116            2755#   2833
$UNIT   001206             653#
$UNITM  001010             595#
$USWR   001220             660#
$VECT1  001244             685#
$VECT2  001246             686#
$XTSTR  013660            2461#
$$GET4= 000000            1668#
$DFILL  015341            2756*   2760*   2770    2805#
$40CAT= ****** U          2458    2575
.       = 015564           551#    555#    568     569#    571#    573#    574#    580     581#    583#    585#    603#    642
                           791     805     806     1676    1679    2240#   2249    2378#   2379    2385    2438    2492    2544#
                          2601    2648#   2729    2858    2882
.$ASTA= ****** U          2496    2499
.$X     = 001000           580#    585
```

```
ANYMOR    1544#    1648
CLRDAT     754#     873     955    1033    1084    1377    1427    1490    1555
CLRUNT     754#    1030    1081    1374    1424    1487
COMMEN       1#     537#
COPYSC     754#     889     923     971    1002    1151    1293    1340    1393    1459    1522    1614
ENDCOM       1#     537#
ERROR      431#     771    1944    2063
ESCAPE       1#     537#
GETPRI       1#     537#
GETSWR       1#     537#
MULT         1#     537#
NEWTST       1#     537#     863     945    1024    1075    1125    1172    1219    1268    1315    1368    1418    1481    1545
POP          1#     537#    2427    2538    2539    2867    2868
PUSH         1#     537#    2401    2499    2501    2522    2848    2854
REPORT       1#     537#
SCOPE      432#     866     948    1027    1078    1128    1175    1222    1271    1318    1371    1421    1484    1548    1648
SETPRI       1#     537#
SETTRA    2823#    2832    2833    2834    2836    2838    2839    2840    2841
SETUP        1#     537#     786
SKIP         1#     537#    1071    1121    1215    1262    1937
SLASH        1#     537#
SPACE      537#
STARS        1#     537#     566     577     579     586     599     642     645     863     865     945     947    1024    1026
          1075     1077    1125    1127    1172    1174    1219    1221    1268    1270    1315    1317    1368    1370    1418
          1420     1481    1483    1545    1547    1639    2248    2251    2319    2348    2387    2444    2494    2552    2604
          2652     2732    2809    2844    2860
TWRSU        1#     537#     807#
TRMTRP    2823#
TYPBIN       1#     537#
TYPDEC       1#     537#
TYPNAM       1#     537#
TYPNUM       1#     537#
TYPOCS       1#     537#
TYPOCT       1#     537#    2269    2616    2640
TYPTXT       1#     537#
$$CMRE     597#     636     637
$$CMTM     597#
$$ESCA       1#     537#
$$NEWT       1#     537#     863     945    1024    1075    1125    1172    1219    1268    1315    1368    1418    1481    1545
$$SET     2823#    2832    2833    2834    2836    2838    2839    2840    2841
$$SETM     823#
$$SKIP       1#     537#    1071    1121    1215    1262
.EQUAT       1#     417#     427
.HEADE       1#     417#
.KT11        1#
.SETUP       1#     417#     708
.SWRHI       1#     417#     538
.SWRLO     549#
.$ACT1       1#     417#     564
.$APTB       1#     643#
.$APTH       1#     417#     575
.$APTY       1#     417#    2492
.$ASTA       1#
.$CATC       1#     417#     549
.$CMTA       1#     417#     597
.$DB2D       1#
```

```
.$DB2O         1#
.$DIV          1#
.$EOP          1#     417#    1637
.$ERRO         1#     417#    2[illegible]0
.$ERRT         1#     417#    2602
.$MULT         1#
.$PARM       417#
.$POWE         1#     417#    2842
.$RAND         1#
.$RDDE         1#
.$RDOC         1#     417#    2385
.$READ         1#     417#    2246
.$R2AZ         1#
.$SAVE         1#     417#
.$SB2D         1#
.$SB2O         1#
.$SCOP         1#     417#    2442
.$SIZE         1#
.$SPAC       417#
.$SUPR         1#
.$SWDO       417#
.$TRAP         1#     417#    2807
.$TYPB         1#
.$TYPD         1#
.$TYPE         1#     417#    2650
.$TYPO         1#     417#    2730
.$40CA         1#
.1170          1#
```

DZVTDB.SRC CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
ADD      856  858  860 1591 1608 1653 2056 2285 2294 2422 2509 2521 2533 2624 2686
        2758 2768
ASL      814 1983 2308 2309 2310 2415 2417 2419 2621 2622 2623 2819
ASR     2014 2015 2029 2030 2516
BEQ      845  838 1445 1506 1650 1652 1667 1759 1831 1858 1879 1885 1899 1926 1937
        1965 2013 2265 2292 2307 2410 2473 2475 2477 2479 2503 2507 2527 2529 2567
        2590 2593 2626 2631 2644 2676 2689 2724 2785
BGT     1661 2304 2345 2412 2792
BHI      846 1977
BIC     1658 1758 1830 1847 1855 1891 1921 1975 1980 2261 2278 2305 2332 2339 2346
        2421 2782
BIS     1856 1859 2312 2787 2788
BISB    1743 1783 1811 1816 2613
BIT     1857 1864 1888 2016 2458 2472 2478 2573 2582
BITB     824 2506 2675 2680 2712
BLO     1979
BLOS    2358
BLT     2302 2343 2414 2703 2753
BMI     1910
BNE      791  814  832  888  921  970 1000 1063 1069 1113 1119 1207 1213 1254 1260
        1457 1520 1571 1575 1763 1835 1881 1883 1892 1901 1912 1915 1923 1929 1939
        1953 2017 2019 2032 2060 2074 2257 2263 2283 2290 2297 2334 2340 2362 2368
        2441 2459 2505 2511 2514 2531 2574 2579 2597 2614 2636 2674 2691 2683 2691
        2699 2713 2720 2783 2866
BPL      840 1593 1610 1756 1765 1828 1871 1899 1935 1973 2021 2034 2054 2062 2259
        2275 2330 2336 2439 2586 2668 2717 2781
BR       790  762  816 1071 1121 1215 1262 1442 1455 1505 1518 1760 1832 1887 1895
        1903 1916 1927 1932 1970 2286 2313 2315 2341 2364 2423 2436 2461 2467 2470
        2481 2497 2519 2584 2619 2646 2670 2696 2706 2715 2722 2759 2774 2795 2858
        2862
CLR      776  777  789  803  823  829  848  850 1656 1747 1819 1866 1867 1868 1896
        1902 1908 1924 1933 1947 2011 2028 2051 2272 2273 2407 2408 2487 2612 2772
        2864
CLRB     907  941  989 1020 1037 1066 1088 1116 1139 1169 1185 1210 1232 1257 1281
        1311 1328 1358 1381 1411 1431 1477 1434 1540 1561 1572 1594 1632 1766 1794
        1836 2369 2432 2482 2535 2536 2537 2695 2721
CMP      770  790  813  837  845  887  920  969 1592 1609 1651 1762 1834 1922 1928
        1936 1938 2256 2262 2282 2289 2301 2303 2333 2339 2342 2344 2357 2468 2596
        1830 1852 1976 1978 2264 2296 2361 2367 2411 2413 2474 2504 2578 2673 2688
        2690 2698 2719 2723
DEC      968  999 1062 1112 1206 1253 1454 1456 1517 1519 1570 1574 1659 1757 1764
        1829 1911 1914 1981 2018 2020 2031 2033 2059 2061 2620
DECB    2702 2705 2780 2791
EMT      431
HALT     555 2587 2598 2669 2857 2881
INC      866  919 1657 1761 1833 1894 2311 2534 2569 2786 2794 2865
INCB    1843 1930 2483 2566 2725
IOT      432
JMP      559  560  561  861 1654 1674 1942 1945 1949 1988
JSR      867  884  908  909  917  942  943  949  966  990  991  997 1021 1022 1028
        1038 1054 1059 1067 1070 1079 1099 1104 1109 1117 1120 1129 1140 1142 1145
        1147 1170 1171 1176 1186 1198 1203 1211 1214 1223 1233 1245 1250 1258 1261
        1272 1282 1284 1287 1289 1312 1313 1319 1329 1331 1334 1336 1359 1361 1364
        1365 1372 1382 1384 1387 1389 1412 1413 1416 1417 1422 1432 1434 1437 1448
        1453 1478 1479 1485 1495 1497 1500 1511 1516 1541 1542 1562 1573 1589 1595
        1606 1612 1633 1635 1669 1748 1753 1787 1792 1820 1825 1964 1967 1969 2042
```

C07

	2044	2300	2523	2575	2581	2678	2697	2704	2711						
MOV	775	778	779	781	783	784	788	792	794	795	796	797	798	799	800
	801	802	805	806	809	810	811	812	817	819	820	821	826	829	830
	844	849	851	853	854	855	857	859	870	883	885	912	916	918	952
	965	967	992	996	998	1030	1040	1041	1053	1055	1058	1060	1061	1081	1091
	1092	1103	1105	1108	1110	1111	1132	1178	1188	1190	1197	1199	1202	1204	1205
	1225	1235	1237	1244	1246	1249	1251	1252	1274	1321	1374	1424	1439	1440	1441
	1443	1446	1458	1487	1502	1503	1504	1506	1509	1521	1551	1552	1563	1568	1569
	1579	1588	1590	1600	1605	1607	1611	1613	1662	1666	1736	1737	1738	1744	1749
	1752	1754	1773	1774	1775	1776	1777	1778	1786	1788	1791	1793	1795	1796	1804
	1805	1806	1813	1821	1824	1826	1845	1846	1848	1860	1864	1865	1869	1890	1907
	1920	1931	1940	1941	1948	1950	1951	1954	1955	1966	1986	1987	2010	2027	2041
	2050	2055	2071	2072	2269	2293	2298	2327	2328	2355	2356	2371	2372	2373	2374
	2399	2400	2401	2402	2403	2405	2406	2425	2426	2427	2428	2429	2463	2464	2466
	2469	2480	2485	2486	2489	2490	2500	2501	2508	2512	2517	2518	2520	2522	2532
	2538	2539	2568	2570	2591	2594	2611	2616	2625	2630	2635	2637	2641	2671	2672
	2677	2685	2700	2755	2763	2764	2765	2771	2778	2796	2797	2798	2799	2800	2815
	2816	2820	2846	2847	2848	2849	2850	2851	2852	2853	2854	2855	2856	2862	2863
	2867	2868	2869	2870	2871	2872	2873	2874	2875	2878					
MOVB	804	871	872	873	874	875	876	877	878	879	880	881	882	889	890
	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905
	906	913	914	923	924	925	926	927	928	929	930	931	932	933	934
	935	936	937	938	939	940	953	954	955	956	957	958	959	960	961
	962	963	964	971	972	973	974	975	976	977	978	979	980	981	982
	983	984	985	986	987	988	993	994	1002	1003	1004	1005	1006	1007	1008
	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1031	1032	1033	1034
	1035	1036	1042	1043	1044	1045	1046	1047	1049	1050	1051	1052	1056	1057	1082
	1083	1084	1085	1086	1087	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102
	1106	1107	1133	1134	1135	1136	1137	1138	1151	1152	1153	1154	1155	1156	1157
	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167	1168	1179	1180	1181	1182
	1183	1184	1191	1192	1193	1194	1195	1196	1200	1201	1226	1227	1228	1229	1230
	1231	1238	1239	1240	1241	1242	1243	1247	1248	1275	1276	1277	1278	1279	1280
	1293	1294	1295	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307
	1308	1309	1310	1322	1323	1324	1325	1326	1327	1340	1341	1342	1343	1344	1345
	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1375	1376	1377
	1378	1379	1380	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404
	1405	1406	1407	1408	1409	1410	1425	1426	1427	1428	1429	1430	1444	1459	1460
	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471	1472	1473	1474	1475
	1476	1488	1489	1490	1491	1492	1493	1507	1522	1523	1524	1525	1526	1527	1528
	1529	1530	1531	1532	1533	1534	1535	1536	1537	1538	1539	1553	1554	1555	1556
	1557	1558	1559	1560	1564	1565	1566	1567	1580	1581	1582	1583	1584	1585	1586
	1587	1601	1602	1603	1604	1614	1615	1616	1617	1618	1619	1620	1621	1622	1623
	1624	1625	1626	1627	1628	1629	1630	1631	1739	1740	1741	1742	1745	1746	1750
	1751	1779	1780	1781	1782	1784	1785	1789	1790	1807	1808	1809	1810	1812	1815
	1817	1818	1822	1823	1854	1878	1897	1974	1982	2073	2260	2277	2331	2337	2360
	2365	2409	2484	2488	2495	2496	2498	2565	2572	2580	2682	2710	2718	2756	2757
	2760	2761	2762	2766	2769	2770	2789	2818							
NEG	2767														
NOP	772	773	1655	1670	1671	1672	1873	1874	1875	1876	1877				
RESET	785	1668													
ROL	2416	2418	2420	2773	2775	2776	2777	2779							
ROR	1068	1118	1212	1259	1849	1850	1851	1852	1853						
RTI	818	2299	2347	2375	2430	2491	2600	2687	2801	2880					
RTS	1767	1797	1837	1861	1966	2022	2036	2045	2057	2064	2075	2540	2639	2727	2821
SUB	2515	2571													
TRAP	2823	2832	2833	2834	2836	2838	2839	2840	2841						

TST	831	847	1649	1755	1827	1900	1925	1984	2012	2291	2305	2424	2431	2440	2465
	2510	2528	2530	2585	2592	2643	2684	2692	2714	2784	2817				
TSTB	839	1870	1886	1898	1909	1934	1952	1972	2053	2258	2274	2329	2335	2438	2476
	2502	2513	2526	2667	2716										
.ASCII	639	640													
.ASCIZ	641	1677	2081	2084	2092	2101	2110	2113	2117	2124	2128	2134	2140	2145	2152
	2157	2164	2173	2182	2189	2197	2205	2212	2218	2224	2233	2379	2380	2381	2383
	2647														
.BLKB	2378														
.BYTE	606	607	612	613	621	622	630	631	632	633	657	658	668	669	676
	677	679	680	682	683	1143	1148	1285	1290	1332	1337	1362	1365	1390	1414
	1435	1449	[illegible]	1512	1676	1680	1682	1687	1692	1697	1702	1707	1711	1716	1720
	1725	1730	1732	2162	2171	2180	2187	2195	2203	2210	2216	2222	2231	2376	2377
	2541	2542	2543	2582	2583	2802	2803	2804	2805						
.DSABL	2316														
.ENABL	1	417	2249												
.END	2885														
.ENDC	422	431	523	537	546	547	548	549	560	567	571	573	578	580	587
	600	604	606	634	638	639	643	646	668	676	679	682	685	686	687
	688	689	692	708	792	793	796	798	800	802	803	805	807	828	864
	865	866	867	945	947	948	949	1025	1026	1027	1028	1072	1076	1077	1078
	1079	1122	1126	1127	1128	1129	1173	1174	1175	1176	1216	1220	1221	1222	1223
	1263	1269	1270	1271	1272	1316	1317	1318	1319	1369	1370	1371	1372	1419	1420
	1421	1422	1482	1483	1484	1485	1546	1547	1548	1549	1640	1641	1642	1644	1646
	1656	1661	1664	1665	1666	1668	1674	1676	1679	1938	2249	2250	2252	2280	2316
	2320	2348	2349	2356	2358	2361	2363	2379	2385	2388	2394	2438	2445	2448	2452
	2458	2460	2471	2474	2475	2476	2478	[illegible]	[illegible]	2489	2492	2495	2496	2499	2526
	2541	2553	2556	2566	2570	2575	2576	2577	[illegible]	2596	2600	2601	2605	2620	2649
	2653	[illegible]	2733	2810	2816	2819	2831	2832	2833	2834	2835	2836	2837	2838	2839
	2840	2841	2842	2845	2854	2855	2861	2867	2868	2878	2880	2884			
.EQUIV	431	432	440	455	456	485	486	487	488	489	490	491	492	493	494
	513	514	515	516	517	518	519	520	521	522					
.EVEN	646	1733	2240	2544	2648										
.IF	418	429	495	523	545	546	547	548	549	558	566	569	571	577	579
	586	599	603	605	634	638	639	642	643	645	668	676	679	682	685
	686	687	[illegible]	689	690	692	708	787	792	794	[illegible]	798	800	802	803
	805	823	[illegible]	865	867	945	947	949	1024	1026	1029	1071	1075	1077	1079
	1121	1125	1127	1129	1172	1174	1176	1215	1219	1221	1223	1262	1268	1270	1272
	1315	1317	1319	1368	1370	1372	1418	1420	1422	1481	1483	[illegible]	1545	1547	1549
	1639	1640	1641	1642	1643	1644	1648	1660	1663	1665	1666	[illegible]	1674	1676	1677
	1679	1937	2248	2250	2251	2252	2290	2319	[illegible]	2348	[illegible]	[illegible]	2361	2362	2378
	2379	2385	2387	[illegible]	2406	2444	2447	2451	2457	2458	[illegible]	[illegible]	2473	2474	2476
	2477	2478	[illegible]	2484	2485	2491	2492	2494	2496	2499	[illegible]	[illegible]	2552	2555	2565
	2569	2573	2575	2576	[illegible]	[illegible]	[illegible]	2596	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2652	2673
	2732	[illegible]	2815	2819	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2838	2839	2840	2841	2842
	2844	[illegible]	[illegible]	2860	2867	2868	2876	2878	[illegible]	[illegible]					
.IFF	429	545	546	548	549	567	571	573	578	580	587	600	603	606	634
	643	646	792	864	865	866	867	946	947	948	949	1025	1026	1027	1028
	1072	1076	1077	1078	1079	1122	1126	1127	1128	1129	1173	1174	1175	1176	1216
	1220	1221	1222	1223	1263	1269	1270	1271	1272	1316	1317	1318	1319	1369	1370
	1371	1372	1419	1420	1421	1422	1482	1483	1484	1485	1546	1547	1548	1549	1640
	1643	1648	1661	1664	1676	1937	2249	2252	2320	2322	2327	2348	2349	2358	2362
	2379	[illegible]	2445	2471	2474	2475	2477	[illegible]	2495	2553	2555	2570	[illegible]	2601	2605
	[illegible]	2649	2653	2733	2810	2816	2845	2861	2876						
.IFT	2322	2327	2411	2431	2438	2483	2576								
.IFTF	2267	2320	2323	2407	2415	2437	2482	2575							

E07

DZVTDB.SRC CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*,DZVTDB.SEQ/SOL/CRF/PAGNUM/NL:TOC=DZVTDB.SML,DZVTDB.SRC
RUN-TIME: 36 46 6 SECONDS
RUN-TIME RATIO: 184/89=2.0
CORE USED: 34K (67 PAGES)