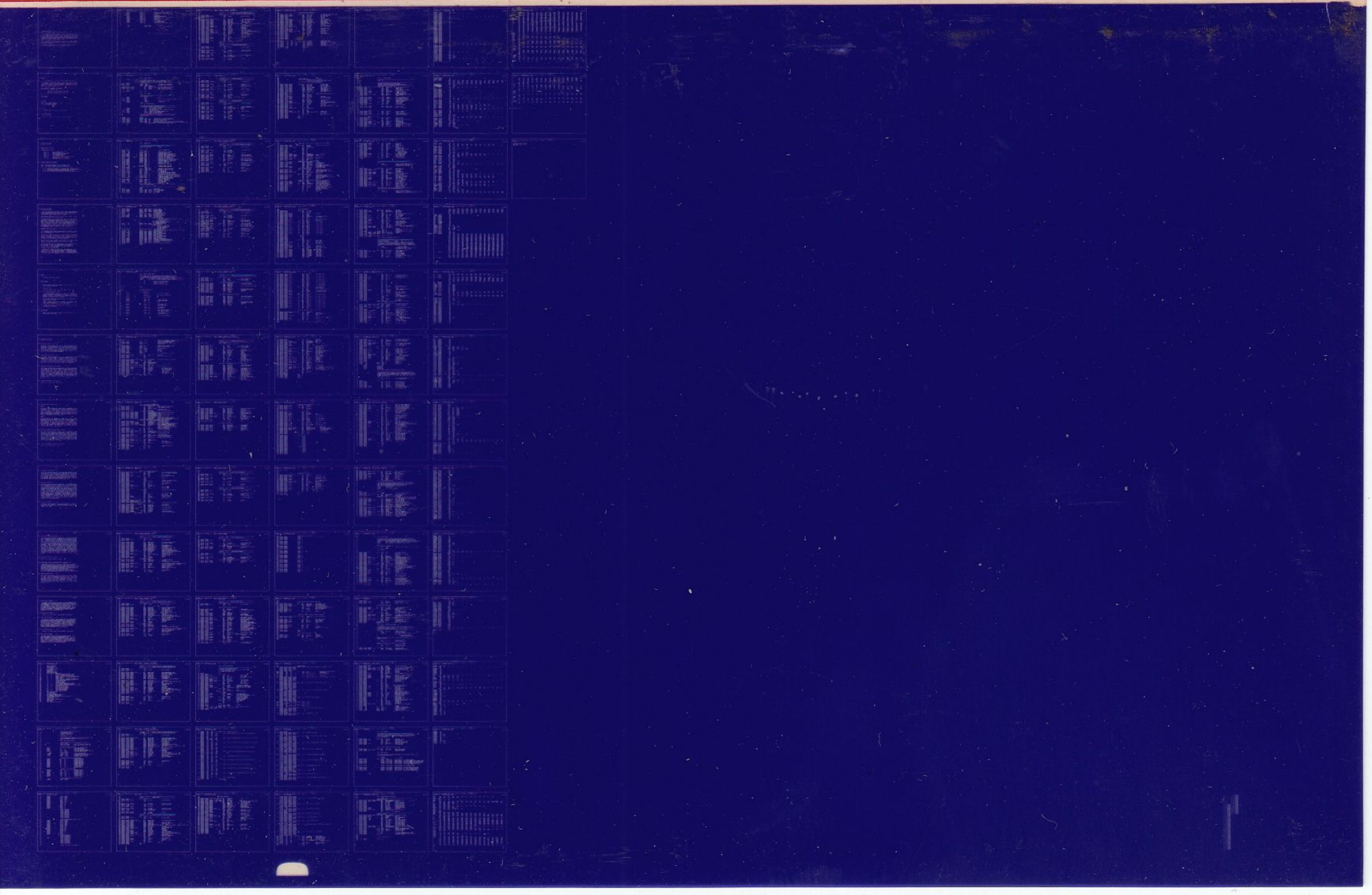
VT105

ACCEPTANCE TEST
CZVTNA0

AH-E760A-MC

FICHE 1 OF 1

DEC 1978
digital
MADE IN USA



IDENTIFICATION

PRODUCT CODE: AC-E759A-MC

PRODUCT NAME: CZVTNA0 VT105 ACCEPTANCE TEST

DATE: AUGUST 1978

MAINTAINER: DIAGNOSTIC GROUP

1.0 ABSTRACT

** The program will run on non-switch register CPU types **

This program is an acceptance test of the VT105 video terminal. The program consists of 19 test patterns displayed on the VT105 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 program is capable of handling multiple VT105's in a sequential DL-11 fashion, however:

****
ONLY ONE VT105 IS TESTED AT ONE TIME.
****

2.0 REQUIREMENTS

2.1 Equipment

PDP-11 family computer with 9K of memory.
VT105 VIDEO GRAPHIC TERMINAL
DL-11 TYPE SERIAL INTERFACE

2.2 Storage

This program LOADS IN 4K BUT uses 9K 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 14 = 1		Loop on CURRENT Test
SW 12 = 1		Inhibit Program Sub-Test Delay
SW 11 = 1		FORCE ASPECT RATIO VIA SW 10
SW 10 = 0		VT55 (RECTANGLE) RATIO
SW 10 = 1		VT105 (SQUARE) RATIO
SW 08 = 1		Loop on Test in SWR <4:0> WHEN EXECUTED
SW 07 = 1		Keyboard Control of the Test
		<SW 8 AND SW 7 = 1 is INCORRECT>

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 VT105 Production

SA 200 INFORM THE OPERATOR OF THE PROGRAM NAME, REQUEST DEVICE BUS ADDRESS IF NON-SWITCH REGISTER CPU, AND THEN RUN THE TEST.
SA 204 USE THE SAME DEVICE BUS ADDRESS AND RUN THE TEST.
SA 210 USE THE DEFAULT BUS ADDRESS AND SHORTEN THE SUB-TEST DELAY.

5.0 OPERATING PROCEDURE

THE VT105 SHOULD BE INITILIZED TO ''ASCII'' MODE. WHEN RUNNING ON A NON-VT105 OR A VT105 ASCII MODE CONSOLE, THE OPERATOR MAY NOTICE THE FOLLOWING CHARACTERS ARE TYPED ''?'', ''2 AND A LOWER CASE ''L''. THESE ARE THE CODES TO SWITCH INTO ASCII MODE.

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 VT105's are connected.
THE PROGRAM WILL USE THE DEFAULT IF A ''CR'' IS TYPED.

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, THE CHARACTER WILL BE IGONRED.
If SW 07 equals a one, and the character received was not a ''/'' or ''\'', IT WILL BE IGNORED. The codes ''X-OFF'' and ''X-ON'' are the only exceptions.

CHANGING THE SOFTWARE SWITCH REGISTER, THE OPERATOR MUST FIRST TYPE ''/'' OR ''\''. THESE KEYS WILL SUSPEND THE CURRENT TEST AND ENSURE THE VT105 IS IN A NON-GRAPHIC MODE. THE OPERATOR MAY NOW TYPE ''CTRL G'' TO CHANGE THE SWITCH REGISTER. UPON COMPLETION OF THE SWITCH CHANGE, THE PROGRAM WILL RETYPE THE PROMPT HEADER MESSAGE.

6.0 ERRORS

NO HARDWARE ERRORS ARE REPORTED.

7.0 MISCELLANEOUS

1. UNLESS USING THE DEDICATED HARDWARE TESTER, Only one VT105 can be tested at one time.

2. Execution Time

   Execution time will vary with the ''BAUD'' rate.

3. Device Address Program Locations

   The location ''$BASE'' contains the first DL11 address if several VT105's are being tested. The default is the Console address <177560>. The location ''LAST'' contains the last DL11 address if several VT105's are being tested. Location VTNOW contains the current DL11 base address.

   THE LOCATION ''$CDW1'' CONTAINS THE TESTER'S FIRST ADDRESS. THE DEFAULT VALUE IS 176500.

   *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.

4. Program is chainable under XXDP/ACT-11. APT Hooks have been provided but not tested.

8.0 RESTRICTIONS

UNLESS USING THE DEDICATED HARDWARE TESTER, ONLY ONE VT105 CAN BE TESTED AT ONE TIME.

SEQ 0006

9.0 PROGRAM DESCRIPTION

9.1 Growing Horizontal Line (A)

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. Then the first line at the base of the screen will be removed, followed by each successive line until the entire block has disappeared.

9.2 Growing Vertical Line (B)

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. Then the first line at the right will be removed, followed by each successive line until the entire block has disappeared.

9.3 Stepping Horizontal Line for Graph 0 (C)

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 (D)

Same as 9.3 except Graph 1 is enabled.

9.5 Different Data on Graph 0 and 1 (E)

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).

9.6 Display a Stepping Histogram Line on Graph 0 (F)

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 (G)

Same as 9.6 except Graph 1 is enabled.

9.8 Histogram on Graph 0 and 1 (H)

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).

9.9 Cursors on Graph 0 (I)

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 the 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.

9.10 Cursors on Graph 1 (J)

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.

9.11 Starting Coordinate on Graph 0 (K)

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 ap-- 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.

9.12 Starting Coordinate on Graph 1 (L)

Same as 9.11 except Graph 1 is enabled.

9.13 GRAPH ASPECT RATIO AND INTERACTIVE TEST ENABLE (M)

THE PURPOSE OF THE TEST IS TO VERIFY OPERATION OF THE INTERNAL INTERACTIVE TEST AND ASPECT RATIO LOGIC. THE VISUAL DISPLAY PATTERN IS GENERATED BY ENABLING THE INTERNAL INTERACTIVE TEST LOGIC. THE VISUAL DISPLAY CONSISTS OF TWO SECTIONS. THE FIRST IS WITH VT55 ASPECT RATIO. ONE HISTOGRAMED TRIANGLE ON THE LEFT SIDE AND A VERTICAL RECTANGLS ON THE RIGHT SIDE SHOULD APPEAR.
WITH THE SECTION SECTION, THE VT105 ASPECT RATIO IS ENABLED.
THE HEIGHT SHOULD REMAIN NEAR CONSTANT BUT THE WIDTH SHOULD CONTRACT.

9.14 CHARACTER ASPECT RATIO (N)

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.

9.15 BASELINE ON HISTOGRAM 0 (O)

THE CORRECT VISUAL DISPLAY WILL BEGIN WITH TWO ACCENDING SHADED HISTOGRAMED RIGHT TRIANGLES ON GRAPH 0. BASE LINE FOR GRAPH 0 IS THEN ENABLED. THE CONTENTS OF THE BASE REGISTER IS INCREMENTED. THE HISTOGRAMED LINE WILL TERMINATE AT THE BASE LINE VALUE. IF THE PLOTTED POINT IS LESS THAN THE BASE REGISTER VALUE, THE HISTOGRAM LINE WILL ASCEND TO THE BASE REGISTER VALUE. THE BASE REGISTER IS INCREMENTED TO MAXIMUM VALUE AND THEN DECREMENTED TO THE VALUE OF ZERO. THE OPERATOR SHOULD OBSERVE A SMOOTH MOVEMENT OF THE PLOTTED HISTOGRAM LINES.

9.16 BASE LINE ON HISTOGRAM 1 (P)

SAME AS 9.15 EXCEPT GRAPH 1 AND BASELINE 1 ARE ENABLED.

9.17 STRIP CHART ON GRAPH 0 (Q)

THE START OF THE CORRECT VISUAL DISPLAY WILL BE THE APPEARANCE OF TWO ACCENDING RIGHT TRIANGLES ON GRAPH 0. THE TRIANGLES ARE DISPLAYED WITH STRIP CHART 0 ENABLED BUT THE TRIANGLES SHOULD START ON THE LEFT EDGE OF THE SCREEN. WHEN THE SECOND TRIANGLE HAS BEEN DISPLAYED, THE PROGRAM WILL CHANGE THE PATTERN TO THAT OF A "SINE WAVE". THE VT105 SHOULD STRIPCHART THE "SINEWAVE" ACROSS THE SCREEN FROM RIGHT TO LEFT. THE OPERATOR SHOULD OBSERVE A SMOOTH MOVEMENT FROM RIGHT TO LEFT. THE INITIAL TWO TRIANGLES SHOULD STRIPCHART LEFT OFF THE SCREEN.

9.18 STRIP CHART ON GRAPH 1 (R)

SAME AS 9.17 EXCEPT GRAPH 1 AND STRIPCHART 1 ARE ENABLED.

9.19 DUAL STRIP CHART MODE (S)

THE CORRECT VISUAL DISPLAY PATTERN WILL BEGIN WITH DUAL STRIP CHART IS ENABLED. A TOTAL OF TEN HISTOGRAMED "SINE WAVE" PATTERNS ARE STRIPCHARTED FROM RIGHT TO LEFT. ONE "SINE WAVE" BEGINS ON A POSITIVE SWING AND THE OTHER ON A NEGATIVE SWING. WHERE THE TWO WAVEFORMS OVERLAP, THE INTENSITY WILL BE BRIGHTER. THE OPERATOR SHOULD OBSERVE A SMOOTH MOVEMENT AND 3 VARIATIONS OF INTENSITY. NO VISUAL LINES PAST THE LEFT EDGE OR RANDOM DOTS PAST THE RIGHT EDGE SHOULD APPEAR.

```
                                .TITLE  CZVTNA  VT105 ACCEPTANCE TEST
                                ;*COPYRIGHT (C) 1978
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY R. SHOOP
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
                                .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
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;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
```

```
(1)             020000                  SW13=   20000
(1)             010000                  SW12=   10000
(1)             004000                  SW11=   4000
(1)             002000                  SW10=   2000
(1)             001000                  SW09=   1000
(1)             000400                  SW08=   400
(1)             000200                  SW07=   200
(1)             000100                  SW06=   100
(1)             000040                  SW05=   40
(1)             000020                  SW04=   20
(1)             000010                  SW03=   10
(1)             000004                  SW02=   4
(1)             000002                  SW01=   2
(1)             000001                  SW00=   1
(1)                                     .EQUIV  SW09,SW9
(1)                                     .EQUIV  SW08,SW8
(1)                                     .EQUIV  SW07,SW7
(1)                                     .EQUIV  SW06,SW6
(1)                                     .EQUIV  SW05,SW5
(1)                                     .EQUIV  SW04,SW4
(1)                                     .EQUIV  SW03,SW3
(1)                                     .EQUIV  SW02,SW2
(1)                                     .EQUIV  SW01,SW1
(1)                                     .EQUIV  SW00,SW0
(1)
(1)                                     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)             100000                  BIT15=  100000
(1)             040000                  BIT14=  40000
(1)             020000                  BIT13=  20000
(1)             010000                  BIT12=  10000
(1)             004000                  BIT11=  4000
(1)             002000                  BIT10=  2000
(1)             001000                  BIT09=  1000
(1)             000400                  BIT08=  400
(1)             000200                  BIT07=  200
(1)             000100                  BIT06=  100
(1)             000040                  BIT05=  40
(1)             000020                  BIT04=  20
(1)             000010                  BIT03=  10
(1)             000004                  BIT02=  4
(1)             000002                  BIT01=  2
(1)             000001                  BIT00=  1
(1)                                     .EQUIV  BIT09,BIT9
(1)                                     .EQUIV  BIT08,BIT8
(1)                                     .EQUIV  BIT07,BIT7
(1)                                     .EQUIV  BIT06,BIT6
(1)                                     .EQUIV  BIT05,BIT5
(1)                                     .EQUIV  BIT04,BIT4
(1)                                     .EQUIV  BIT03,BIT3
(1)                                     .EQUIV  BIT02,BIT2
(1)                                     .EQUIV  BIT01,BIT1
(1)                                     .EQUIV  BIT00,BIT0
(1)
(1)                                     ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
```

```
    (1)         000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
    (1)         000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
    (1)         000014                  TBITVEC=14              ;;'T'' BIT
    (1)         000014                  TRTVEC= 14              ;;TRACE TRAP
    (1)         000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
    (1)         000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
    (1)         000024                  PWRVEC= 24              ;;POWER FAIL
    (1)         000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
    (1)         000034                  TRAPVEC=34              ;;'TRAP'' TRAP
    (1)         000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
    (1)         000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
    (1)         000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
     14
     15
     16         177560                          ABASE=177560
     17         176500                          ACDW1=176500            ;TESTER STARTING ADDRESS
```

```
                                .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 001320          JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137 001354          JMP     RBEGIN          ;JUMP TO RESTART ADDRESS
000210  000137 001364          JMP     BEGIN2          ;JUMP TO ADJUSTMENT PATTERN

                        000176          .=176
000176  000200                  200

                                .SBTTL  ACT11 HOOKS
                                ;;**************************************************************************
                                ;HOOKS REQUIRED BY ACT11
                        000200          $SVPC=.                 ;SAVE PC
                        000046          .=46
000046  005332                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        000052          .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
                        000200          .=$SVPC                 ;; RESTORE PC
                        001000  .=1000
                                .SBTTL  APT PARAMETER BLOCK
                                ;;**************************************************************************
                                ;
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;**************************************************************************
                        001000          .$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  001172          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000300          $TSTM:  .WORD   300     ;;RUN TIM OF LONGEST TEST
001006  000300          $PASTM: .WORD   300     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000300          $UNITM: .WORD   300     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
 29                                     .SBTTL  COMMON TAGS
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)                                     ;*USED IN THE PROGRAM.
(1)
(1)             001100                          .=1100
(1)  001100                             $CMTAG:                         ;;START OF COMMON TAGS
(1)  001100  000000                             .WORD   0
(1)  001102     000             $TSTNM: .BYTE   0                       ;;CONTAINS THE TEST NUMBER
(1)  001103     000             $ERFLG: .BYTE   0                       ;;CONTAINS ERROR FLAG
(1)  001104  000000             $ICNT:  .WORD   0                       ;;CONTAINS SUBTEST ITERATION COUNT
(1)  001106  000000             $LPADR: .WORD   0                       ;;CONTAINS SCOPE LOOP ADDRESS
(1)  001110  000000             $LPERR: .WORD   0                       ;;CONTAINS SCOPE RETURN FOR ERRORS
(1)  001112  000000             $ERTTL: .WORD   0                       ;;CONTAINS TOTAL ERRORS DETECTED
(1)  001114     000             $ITEMB: .BYTE   0                       ;;CONTAINS ITEM CONTROL BYTE
(1)  001115     001             $ERMAX: .BYTE   1                       ;;CONTAINS MAX. ERRORS PER TEST
(1)  001116  000000             $ERRPC: .WORD   0                       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)  001120  000000             $GDADR: .WORD   0                       ;;CONTAINS ADDRESS OF 'GOOD' DATA
(1)  001122  000000             $BDADR: .WORD   0                       ;;CONTAINS ADDRESS OF 'BAD' DATA
(1)  001124  000000             $GDDAT: .WORD   0                       ;;CONTAINS 'GOOD' DATA
(1)  001126  000000             $BDDAT: .WORD   0                       ;;CONTAINS 'BAD' DATA
(1)  001130  000000                     .WORD   0                       ;;RESERVED--NOT TO BE USED
(1)  001132  000000                     .WORD   0
(1)  001134     000             $AUTOB: .BYTE   0                       ;;AUTOMATIC MODE INDICATOR
(1)  001135     000             $INTAG: .BYTE   0                       ;;INTERRUPT MODE INDICATOR
(1)  001136  000000                     .WORD   0
(1)  001140  177570             SWR:    .WORD   DSWR                    ;;ADDRESS OF SWITCH REGISTER
(1)  001142  177570             DISPLAY: .WORD  DDISP                   ;;ADDRESS OF DISPLAY REGISTER
(1)  001144  177560             $TKS:   177560                          ;;TTY KBD STATUS
(1)  001146  177562             $TKB:   177562                          ;;TTY KBD BUFFER
(1)  001150  177564             $TPS:   177564                          ;;TTY PRINTER STATUS REG. ADDRESS
(1)  001152  177566             $TPB:   177566                          ;;TTY PRINTER BUFFER REG. ADDRESS
(1)  001154     000             $NULL:  .BYTE   0                       ;;CONTAINS NULL CHARACTER FOR FILLS
(1)  001155     002             $FILLS: .BYTE   2                       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)  001156     012             $FILLC: .BYTE   12                      ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
(1)  001157     000             $TPFLG: .BYTE   0                       ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
(1)  001160  000000             $REGAD: .WORD   0                       ;;CONTAINS THE ADDRESS FROM
(1)                                                                     ;;WHICH  ($REG0) WAS OBTAINED
(3)  001162  000000             $REG0:  .WORD   0                       ;;CONTAINS (($REGAD)+0)
(3)  001164  000000             $REG1:  .WORD   0                       ;;CONTAINS (($REGAD)+2)
(1)  001166     077             $QUES:  .ASCII  /?/                     ;;QUESTION MARK
(1)  001167     015             $CRLF:  .ASCII  <15>                    ;;CARRIAGE RETURN
(1)  001170  000012             $LF:    .ASCIZ  <12>                    ;;LINE FEED
(2)                                     ;;**********************************************************************
(2)                                     .SBTTL  APT MAILBOX-ETABLE
(2)
(3)                                     ;;**********************************************************************
(2)                                     .EVEN
(2)  001172                             $MAIL:                          ;;APT MAILBOX
(2)  001172  000000             $MSGTY: .WORD   AMSGTY                  ;;MESSAGE TYPE CODE
(2)  001174  000000             $FATAL: .WORD   AFATAL                  ;;FATAL ERROR NUMBER
(2)  001176  000000             $TESTN: .WORD   ATESTN                  ;;TEST NUMBER
(2)  001200  000000             $PASS:  .WORD   APASS                   ;;PASS COUNT
```

```
(2)  001202  000000                $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
(2)  001204  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
(2)  001206  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2)  001210  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001212                        $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  001212     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001213     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
(2)  001214  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001216  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001220  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)                                ;*                      BITS 15-11=CPU TYPE
(2)                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)                                ;*                              11/70=06,PDQ=07,Q=10
(2)                                ;*                      BIT 10=REAL TIME CLOCK
(2)                                ;*                      BIT  9=FLOATING POINT PROCESSOR
(2)                                ;*                      BIT  8=MEMORY MANAGEMENT
(2)  001222     000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001223     000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
(2)                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                                ;*                              900 NSEC CORE=001
(2)                                ;*                              300 NSEC BIPOLAR=002
(2)                                ;*                              500 NSEC MOS=003
(2)  001224  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)  001226     000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001227     000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001230  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001232     000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001233     000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001234  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001236     000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001237     000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001240  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001242  000000                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001244  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001246  177560                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001250  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001252  176500                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001254                        $ETEND:
(2)                                .MEXIT
```

```
(1)                                 .SBTTL  ERROR POINTER TABLE
(1)
(1)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                 ;*      DT              ;;POINTS TO THE DATA
(1)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  001254                         $ERRTB:
 30
 32
 33                                 ;NO ERRORS ARE REPORTED
 34
 35                                 ;VT-55 EQUALITIES
 36
 37          000354                         MAXHOZ=236.             ;MAX. HORIZ LINE COUNT
 38
 39          001000                         MAXVRT=512.             ;MAX. VERTICAL LINE COUNT
 40
 41          002000                         ADDLIN=BIT10
 42
 43          000033                         ESC=    33
 44          000062                         GROF=   62              ;DISABLE GRAPH MODE
 45          000061                         GRON=   61              ;ENABLE GRAPH MODE
 46
 47          000100                         LNO=    100             ;NOP
 48
 49          000101                         LDE0=   101             ;LOAD ENABLE REG. 0
 50          000111                         LDE1=   111             ;LOAD ENABLE REG. 1
 51
 52          000102                         LDG0=   102             ;LOAD GRAPH 0
 53          000112                         LDG1=   112             ;LOAD GRAPH 1
 54
 55          000103                         LDC0=   103             ;LOAD CURSOR ON GRAPH 0
 56          000113                         LDC1=   113             ;LOAD CURSOR ON GRAPH 1
 57
 58          000104                         LHV0=   104             ;LOAD HORIZONTAL LINE
 59          000114                         LHV1=   114             ;LOAD VERTICAL LINE
 60
 61          000110                         LSC=    110             ;LOAD STARTING COORDINATE
 64
```

```
 66
 67  001254  177560                  FIRST:  177560                  ;FIRST DEVICE ADDRESS OF SEQUIENTAL DL-11-A/B TYPE DEVIC
 68                                                                  ;DEFAULT TO THE CONSOLE ADDRESS
 69  001256  000000                  LAST:   0                       ;LAST DEVICE ADDRESS OF DL-11-A/B TYPE
 70  001260  177560                  VTNOW:  177560                  ;CURRENT DEVICE BUSS ADDRESS
 71  001262  000000                  TSTNUM: 0                       ;ERROR PATTERN
 72
 73  001264  000200                  TIMEO:  200                     ;CHARACTER FLAG TIMEOUT CONSTANT
 74  001266  000003                  SUBTST: 3.                      ;SUBTEST DELAY CONSTANT
 80  001270  000000                  WFTEST: 0
 81  001272  177560                  VTIS:   177560                  ;DEVICE ADDRESSES
 82  001274  177562                  VTIB:   177562                  ;IN DATA
 83  001276  177564                  VTOS:   177564                  ;OUT STAT
 84  001300  177566                  VTOB:   177566                  ;OUT DATA
 85  001302  000000                  SAVE4:  0
 86  001304  000040                  ASPTRB: 40                      ;40 = VT55 RATIO  41 = 105 RATIO
 87
 88  001306  022626                  BUSSTR: CMP     (SP)+,(SP)+     ;POP STACK
 89  001310  104401  010641                  TYPE,   EM3             ;REPORT BUS TIMEOUT TO OPER.
 90  001314  000240                          NOP
 91  001316  000240                          NOP
 92
 93  001320  012737  002122  002120  BEGIN:  MOV     #TST1,WHERE     ;STARTING ACCEPTANCE TEST ADDRESS
 94  001326  005037  001270                  CLR     WFTEST
 95  001332  005037  001302                  CLR     SAVE4
 96  001336  012737  C01306  000004          MOV     #BUSSTR,@#ERRVEC
 97  001344  012737  000340  000006          MOV     #340,@#ERRVEC+2
 98  001352  000424                          BR      GINA
 99  001354  012737  002122  002120  RBEGIN: MOV     #TST1,WHERE
100  001362  000415                          BR      GIN
101  001364  012737  000001  001270  BEGIN2: MOV     #1,WFTEST
102  001372  012700  010064                  MOV     #VTIS0,R0               ;GET POINTER TO DLV11'S
103  001376  013701  001252                  MOV     $CDW1,R1                ;GET INITIAL VALUE
104  001402  010120                  1$:     MOV     R1,(R0)+                ;LOAD THE ADDRESSES
105  001404  022700  010164                  CMP     #VTOB7+2,R0             ;CHECK IF DONE
106  001410  001405                          BEQ     GINA                    ;BR IF DONE
107  001412  005721                          TST     (R1)+                   ;BUMP THE ADDRESS VALUE
108  001414  000772                          BR      1$                      ;TRY AGAIN
109  001416  012737  000001  001302  GIN:    MOV     #1,SAVE4
110  001424  000005                  GINA:   RESET
```

```
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001426  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001432  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001434  022706  001140                  CMP     #SWR,R6 ;;DONE?
001440  001374                          BNE     .-6             ;;LOOP BACK IF NO
001442  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001446  012737  013444  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001454  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001462  012737  014304  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001470  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001476  012737  014366  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001504  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001512  013737  005300  005272          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001520  012737  001520  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001526  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001532  012737  001566  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001540  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001546  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001554  022777  177777  177356          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001562  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;/ ND  THE HARDWARE SWR IS NOT = -1
001564  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001566  012716  001574          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001572  000002                          RTI
001574  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001602  012737  000174  001142          MOV     #DISPREG,DISPLAY
001610  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001614  005037  001200                  CLR     $PASS           ;;CLEAR PASS COUNT
001620  132737  000200  001213          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
001626  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
001630  012737  001214  001140          MOV     #$SWREG,SWR     ;;NO,USE API SWITCH REGISTER
001636                          67$:
001636  005037  007532                  CLR     LOOP
001642  012737  013376  000020          MOV     #MSCOPE,@#IOTVEC
001650  005737  001302          TITST:  TST     SAVE4                   ;TEST FLAG
001654  001067                          BNE     4$                      ;BR IF NON-ZERO
001656  013737  001246  001254          MOV     $BASE,FIRST             ;LOAD DEFAULT ADDRESS
001664  104401                          TYPE
001666  010346                          TITLE
001670  005037  001704                  CLR     66$                     ;SETUP STARTUP DELAY
001674  005337  001704          1$:     DEC     66$                     ;DELAY
001700  001375                          BNE     1$
001702  000401                          BR      11$
001704  000000          66$:    0
```

```
                                        ;NOW TEST IF RUNNING ON NON-SWR CPU
001706  022737  000176  001140  11$:    CMP     #SWREG,SWR      ;TEST IF NON-SWITCH REGISTER
001714  001403                          BEQ     2$              ;BR IF YES - AND ASK ADDRESS
001716  105777  177216                  TSTB    @SWR
001722  100044                          BPL     4$              ;BR IF CLEARED
001724  005737  000042          2$:     TST     @#42            ;TEST IF XXDP CHAIN MODE
001730  001037                          BNE     3$              ;BR IF YES
001732  104401                          TYPE
001734  010616                          WHAT8                   ;FIRST
001736  104401                          TYPE
001740  010561                          WHAT0                   ;FIND OUT THE DEVICE ADDRESS
001742  017700  177200                  MOV     @$TKB,R0        ;ENSURE CLEAR CHAR.
001746  104412                          RDOCT
001750  012600                          MOV     (SP)+,R0        ;SAVE THE ADDRESS
001752  001426                          BEQ     3$              ;BR IF ''CR''
001754  010037  001254                  MOV     R0,FIRST
001760  022737  160000  001254          CMP     #160000,FIRST
001766  101347                          BHI     11$             ;BR IF INVALID
001770  005777  177260                  TST     @FIRST          ;TEST IF VALID
001774  104401                          TYPE
001776  010630                          WHAT9                   ;FIND OUT THE LAST ADDRESS
002000  104401                          TYPE
002002  010561                          WHAT0
002004  104412                          RDOCT                   ;GET OPR INPUT
002006  012600                          MOV     (SP)+,R0        ;
002010  001407                          BEQ     3$              ;BR IF NONE
002012  005710                          TST     (R0)            ;SEE IF IT EXISTS
002014  010037  001256                  MOV     R0,LAST         ;IT MUST, SAVE IT
002020  022700  160000                  CMP     #160000,R0      ;TEST IF I/O VALUE
002024  101330                          BHI     11$             ;BR IF NOT
002026  000402                          BR      4$
002030  005037  001256          3$:     CLR     LAST
002034  012737  000006  000004  4$:     MOV     #6,@#4
002042  005037  000006                  CLR     @#6
002046  013737  001254  001260  RSTRTA: MOV     FIRST,VTNOW     ;LOAD INITIAL DEVICE ADDRESS
                                        ;
002054  012700  001272          RSTRT:  MOV     #VTIS,R0        ;LOAD POINTER
002060  013720  001260                  MOV     VTNOW,(R0)+     ;LOAD INPUT STAT
002064  013710  001260                  MOV     VTNOW,(R0)
002070  062720  000002                  ADD     #2,(R0)+        ;LOAD INPUT BUFFER
002074  013710  001260                  MOV     VTNOW,(R0)
002100  062720  000004                  ADD     #4,(R0)+        ;LOAD OUTPUT STAT
002104  013710  001260                  MOV     VTNOW,(R0)
002110  062720  000006                  ADD     #6,(R0)+        ;LOAD OUTPUT BUFFER
002114  000177  000000                  JMP     @WHERE          ;JUMP TO STARTING ADDRESS
002120  002122                  WHERE:  TST1
```

```
                                ;;*******************************************************************
                                ;*TEST 1         A       DISPLAY A GROWING HORIZONTAL LINES
                                ;;*******************************************************************
                                ;
002122  000004                  TST1:   SCOPE
002124  004537  010164                  JSR     R5,AMSG                 ;DISPLAY HEADER
002130  010700                          DAHL
                                
002132  012700  014540                  MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
002136  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD GRAPHIC ENABLE
002142  112720  000061                  MOVB    #GRON,(R0)+             ;
002146  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002152  112720  000041                  MOVB    #BIT5!BIT0,(R0)+        ;LOAD DISPLAY ENABLE
002156  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002162  112720  000041                  MOVB    #BIT5!BIT0,(R0)+        ;LOAD DISPLAY HORIZ. LINE
002166  113720  001304                  MOVB    ASPTRB,(R0)+            ;LOAD ASPECT RATIO
002172  112720  000104                  MOVB    #LHV0,(R0)+             ;LOAD LOAD HORIZONTAL LINE
002176  012737  002000  006374          MOV     #ADDLIN,BASE            ;LOAD THE STARTING DATA VALUE
002204  004737  006400          1$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002210  010120                          MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002212  005237  006374                  INC     BASE                    ;UPDATE THE DATA
002216  022737  002354  006374          CMP     #ADDLIN!MAXHOZ,BASE     ;COMPARE TO LAST DATA LINE
002224  001367                          BNE     1$                      ;LOOP UNTIL DONE
002226  105020                          CLRB    (R0)+                   ;TERM
002230  004737  006632                  JSR     PC,XPRNT                ;EXECUTE
002234  004737  007756                  JSR     PC,DELAY
                                ;NOW REMOVE THE LINE
002240  012700  014540                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
002244  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
002250  112720  000104                  MOVB    #LHV0,(R0)+             ;LOAD HORIZONTAL LINE MODE AGAIN
                                
002254  012737  000000  006374          MOV     #0,BASE                 ;LOAD STARTING DATA VALUE TO REMOVE THE LINE
002262  004737  006400          2$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002266  010120                          MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002270  005237  006374                  INC     BASE                    ;UPDATE THE DATA
002274  022737  000354  006374          CMP     #MAXHOZ,BASE            ;COMPATE TO LAST DATA LINE
002302  001367                          BNE     2$                      ;BR TOLL DONE
                                
002304  105020                          CLRB    (R0)+                   ;LOAD TERMINATOR
002306  004737  006632                  JSR     PC,XPRNT                ;DISPLAY
002312  004737  007756                  JSR     PC,DELAY
                                
```

```
                                        ;;*******************************************************************
                                        ;*TEST 2        B       DISPLAY A GROWING VERTICAL LINES
                                        ;;*******************************************************************
                                        TST2:   SCOPE
002316  000004
002320  004537  010164                          JSR     R5,AMSG                 ;DISPLAY HEADER
002324  010724                                  DAVL
002326  012700  014540                          MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
002332  112720  000033                          MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
002336  112720  000061                          MOVB    #GRON,(R0)+             ;LOAD "01" ENTER CODE
002342  112720  000101                          MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002346  112720  000041                          MOVB    #BIT5!BIT0,(R0)+        ;LOAD DISPLAY ENABLE
002352  112720  000111                          MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002356  112720  000042                          MOVB    #BIT5!BIT1,(R0)+        ;LOAD DISPLAY VERTICAL LINE
002362  113720  001304                          MOVB    ASPTRB,(R0)+            ;LOAD NOP
002366  112720  000114                          MOVB    #LHV1,(R0)+             ;LOAD VERTICAL LINE
002372  012737  002777  006374                  MOV     #ADDLIN!MAXVRT-1,BASE   ;LOAD THE STARTING DATA VALUE
002400  004737  006400                  1$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002404  010120                                  MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002406  005337  006374                          DEC     BASE                    ;UPDATE THE DATA
002412  022737  002000  006374                  CMP     #ADDLIN,BASE            ;COMPARE TO LAST DATA LINE
002420  001367                                  BNE     1$                      ;LOOP UNTIL DONE
002422  105020                                  CLRB    (R0)+                   ;LOAD TERM
002424  004737  006632                          JSR     PC,XPRNT                ;EXECUTE
002430  004737  007756                          JSR     PC,DELAY
                                        ;NOW REMOVE THE LINE
002434  012700  014540                          MOV     #BUFFER,R0
002440  112720  000100                          MOVB    #LNO,(R0)+              ;LOAD NOP
002444  112720  000114                          MOVB    #LHV1,(R0)+             ;LOAD VERTIACL LINE MODE AGAIN

002450  012737  001000  006374                  MOV     #MAXVRT,BASE            ;LOAD STARTING DATA VALUE TO REMOVE THE LINE
002456  004737  006400                  2$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002462  010120                                  MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002464  005337  006374                          DEC     BASE                    ;UPDATE THE DATA
002470  001372                                  BNE     2$                      ;BR TOLL DONE

002472  105020                                  CLRB    (R0)+                   ;LOAD TERMINATOR
002474  004737  006632                          JSR     PC,XPRNT                ;DISPLAY
002500  004737  007756                          JSR     PC,DELAY
```

```
                        ;;*****************************************************************
                        ;*TEST 3        C       GRAPH 0 DISPLAY A STEPPING HORIZONTAL LINE
                        ;;*****************************************************************
002504  000004          TST3:   SCOPE
002506  004537  010164          JSR     R5,AMSG                 ;DISPLAY HEADER
002512  010747                  SHL0
002514  012737  000200  002674          MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
002522  012700  014540  2$:     MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
002526  112720  000033          MOVB    #ESC,(R0)+              ;ENTER VT-55 FORMAT
002532  112720  000061          MOVB    #GRON,(R0)+             ;LOAD ENTER '01' CODE
002536  112720  000101          MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002542  112720  000040          MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
002546  112720  000111          MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002552  112720  000060          MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH, LINES AND CURSORS
002556  112720  000101          MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002562  112720  000043          MOVB    #BIT5!BIT1!BIT0,(R0)+   ;LOAD DISPLAY ENABLE AND GRAPH 0 ON
002566  112720  000100          MOVB    #LNO,(R0)+              ;NOP
002572  112720  000110          MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
002576  012737  000000  006374          MOV     #0,BASE                 ;GET BASE LINE
002604  004737  006400          JSR     PC,SHUFF                ;CONVERT
002610  010120                  MOV     R1,(R0)+                ;SAVE COORD.
002612  112720  000100          MOVB    #LNO,(R0)+              ;LOAD NOP
002616  112720  000102          MOVB    #LDG0,(R0)+             ;LOAD 'LOAD GRAPH'
002622  013737  002674  006374          MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
002630  004737  006400          JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
002634  012737  001000  002676          MOV     #MAXVRT,101$            ;LOAD COUNTER
002642  010120          1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
002644  005337  002676          DEC     101$                    ;DONE FULL GRAPH
002650  001374                  BNE     1$

002652  105020                  CLRB    (R0)+                   ;LOAD TERMINATOR
002654  004737  006632          JSR     PC,XPRNT                ;DISPLAY
002660  006037  002674          ROR     100$                    ;CHANGE DATA VALUE
002664  001316                  BNE     2$                      ;NO
002666  004737  007756          JSR     PC,DELAY
002672  000402                  BR      TST4            ;;NEXT TEST

002674  000000          100$:   0
002676  000000          101$:   0
```

```
                        ;;*****************************************************************
                        ;*TEST 4        D       GRAPH 1 DISPLAY A STEPPING HORIZONTAL LINE
                        ;;*****************************************************************
002700  000004          TST4:   SCOPE
002702  004537  010164          JSR     R5,AMSG                 ;DISPLAY HEADER
002706  011017          SHL1
002710  012737  000200  003070          MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
002716  012700  014540  2$:     MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
002722  112720  000033          MOVB    #ESC,(R0)+              ;ENTER VT55  MODE
002726  112720  000061          MOVB    #GRON,(R0)+             ;LOAD ENTER '01'' CODE
002732  112720  000101          MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002736  112720  000040          MOVB    #BIT5,(R0)+             ;DISABLE DISPLAY
002742  112720  000111          MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
002746  112720  000060          MOVB    #BIT5!BIT4,(R0)+        ;CLEAR GRAPH,LINES NAD CURSORS
002752  112720  000101          MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
002756  112720  000045          MOVB    #BIT5!BIT2!BIT0,(R0)+   ;LOAD DISPLAY ENABLE AND GRAPH 1 ON
002762  112720  000100          MOVB    #LNO,(R0)+              ;NOP
002766  112720  000110          MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
002772  012737  000000  006374          MOV     #0,BASE                 ;GET BASE LINE
003000  004737  006400          JSR     PC,SHUFF                ;CONVERT
003004  010120          MOV     R1,(R0)+                ;SAVE COORD.
003006  112720  000100          MOVB    #LNO,(R0)+              ;LOAD NOP
003012  112720  000112          MOVB    #LDG1,(R0)+             ;LOAD 'LOAD GRAPH''
003016  013737  003070  006374          MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
003024  004737  006400          JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
003030  012737  001000  003072          MOV     #MAXVRT,101$            ;LOAD COUNTER
003036  010120          1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
003040  005337  003072          DEC     101$                    ;DONE FULL GRAPH
003044  001374          BNE     1$

003046  105020          CLRB    (R0)+                   ;LOAD TERMINATOR
003050  004737  006632          JSR     PC,XPRNT                ;DISPLAY
003054  006037  003070          ROR     100$
003060  001316          BNE     2$                      ;NO
003062  004737  007756          JSR     PC,DELAY
003066  000402          BR      TST5            ;;NEXT TEST

003070  000000  100$:   0
003072  000000  101$:   0
```

```
  332                                   ;;*****************************************************************
  (3)                                   ;*TEST 5         E       GRAPH 0 AND 1
  (3)                                   ;;*****************************************************************
  (2)  003074  000004                   TST5:   SCOPE
  333  003076  004537  010164                   JSR     R5,AMSG                 ;DISPLAY HEADER
  334  003102  011067                           GR0A1
  335
  337
  338  003104  004537  005610                   JSR     R5,UPDWN                ;LOAD DATA PATTERN
  339  003110     002     102                    .BYTE   BIT1,LDG0               ;GRAPH 0 INC. PAT.
  340  003112  000000                           .WORD   0
  341  003114  004737  006632                   JSR     PC,XPRNT
  342
  343  003120  004537  005610                   JSR     R5,UPDWN                ;LOAD DATA PATTERN
  344  003124     006     112                    .BYTE   BIT2!BIT1,LDG1          ;GRAPH 0 DEC. PAT.
  345  003126  100354                           .WORD   BIT15!MAXHOZ                    ;
  346
  347  003130  004737  006632                   JSR     PC,XPRNT                ;EXECUTE
  348  003134  004737  007756                   JSR     PC,DELAY
  349                                   ;;*****************************************************************
  (3)                                   ;*TEST 6         F       GRAPH 0 DISPLAY A STEPPING HISTOGRAM LINE
  (3)                                   ;;*****************************************************************
  (2)  003140  000004                   TST6:   SCOPE
  350  003142  004537  010164                   JSR     R5,AMSG                 ;DISPLAY HEADER
  351  003146  011125                           SHGL0
  353
  354  003150  012737  000200  003310           MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
  355  003156  012700  014540           2$:     MOV     #BUFFER,R0
  356  003162  112720  000033                   MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
  357  003166  112720  000061                   MOVB    #GRON,(R0)+             ;LOAD "01" ENTER CODE
  358  003172  112720  000101                   MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
  359  003176  112720  000053                   MOVB    #BIT5!BIT3!BIT1!BIT0,(R0)+      ;LOAD DISP. ENABLE , GRAPH 0, HISTO 1 ON
  360  003202  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP
  361  003206  112720  000110                   MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
  362  003212  012737  000000  006374           MOV     #0,BASE                 ;GET BASE LINE
  363  003220  004737  006400                   JSR     PC,SHUFF                ;CONVERT
  364  003224  010120                           MOV     R1,(R0)+                ;SAVE COORD.
  365  003226  112720  000100                   MOVB    #LNO,(R0)+              ;LOAD NOP
  366  003232  112720  000102                   MOVB    #LDG0,(R0)+             ;LOAD "LOAD GRAPH"
  367  003236  013737  003310  006374           MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
  368  003244  004737  006400                   JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
  369  003250  012737  001000  003312           MOV     #MAXVRT,101$            ;LOAD COUNTER
  370  003256  010120                   1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
  371  003260  005337  003312                   DEC     101$                    ;DONE FULL GRAPH
  372  003264  001374                           BNE     1$
  373  003266  105020                           CLRB    (R0)+                   ;LOAD TERMINATOR
  374  003270  004737  006632                   JSR     PC,XPRNT                ;DISPLAY
  375  003274  006037  003310                   ROR     100$                    ;CHANGE DATA VALUE
  376  003300  001326                           BNE     2$                      ;NO
  377  003302  004737  007756                   JSR     PC,DELAY
  378  003306  000402                           BR      TST7            ;;NEXT TEST
  379  003310  000000                   100$:   0
  380  003312  000000                   101$:   0
```

```
382                                     ;;*****************************************************************
(3)                                     ;*TEST 7        G       GRAPH 1 DISPLAY A STEPPING HISTOGRAM LINE
(3)                                     ;;*****************************************************************
(2)  003314  000004                     TST7:   SCOPE
383  003316  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
384  003322  011170                            SHGL1
386
387  003324  012737  000200  003464            MOV     #BIT7,100$              ;LOAD STARTING BASE LINE
388  003332  012700  014540            2$:     MOV     #BUFFER,R0              ;LOAD THE STARTING ADDRESS
389  003336  112720  000033                    MOVB    #ESC,(R0)+              ;LOAD "ESC" CODE
390  003342  112720  000061                    MOVB    #GRON,(R0)+             ;LOAD "01" ENTER CODE
391  003346  112720  000101                    MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
392  003352  112720  000065                    MOVB    #BIT5!BIT4!BIT2!BIT0,(R0)+      ;LOAD DISPLAY ENABLE AND GRAPH 1 ON
393  003356  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP
394  003362  112720  000110                    MOVB    #LSC,(R0)+              ;LOAD STARTING COORD.
395  003366  012737  000000  006374            MOV     #0,BASE                 ;GET BASE LINE
396  003374  004737  006400                    JSR     PC,SHUFF                ;CONVERT
397  003400  010120                            MOV     R1,(R0)+                ;SAVE COORD.
398  003402  112720  000100                    MOVB    #LNO,(R0)+              ;LOAD NOP
399  003406  112720  000112                    MOVB    #LDG1,(R0)+             ;LOAD "LOAD GRAPH"
400  003412  013737  003464  006374            MOV     100$,BASE               ;LOAD THE STARTING DATA VALUE
401  003420  004737  006400                    JSR     PC,SHUFF                ;SHUFFEL THE DATA INTO VT-55 FORMAT
402  003424  012737  001000  003466            MOV     #MAXVRT,101$            ;LOAD COUNTER
403  003432  010120                    1$:     MOV     R1,(R0)+                ;SAVE THE LSB MSB BYTE
404  003434  005337  003466                    DEC     101$                    ;DONE FULL GRAPH
405  003440  001374                            BNE     1$
406  003442  105020                            CLRB    (R0)+                   ;LOAD TERMINATOR
407  003444  004737  006632                    JSR     PC,XPRNT                ;DISPLAY
408  003450  006037  003464                    ROR     100$
409  003454  001326                            BNE     2$                      ;NO
410  003456  004737  007756                    JSR     PC,DELAY
411  003462  000402                            BR      TST10           ;;NEXT TEST
412  003464  000000                    100$:   0
413  003466  000000                    101$:   0
414                                     ;;*****************************************************************
(3)                                     ;*TEST 10       H       HISTOGRAM ON GRAPH 0 AND 1
(3)                                     ;;*****************************************************************
(2)  003470  000004                     TST10:  SCOPE
415  003472  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
416  003476  011233                            HG0A1
418
419  003500  004537  005610                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
420  003504     012     102                    .BYTE   BIT3!BIT1,LDG0          ;GRAPH 0 INC. PAT.
421  003506  000000                            .WORD   0
422  003510  004737  006632                    JSR     PC,XPRNT
423
424  003514  004537  005610                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
425  003520     036     112                    .BYTE   BIT4!BIT3!BIT2!BIT1,LDG1        ;GRAPH 1
426  003522  100354                            .WORD   BIT15!MAXHOZ
427
428  003524  004737  006632                    JSR     PC,XPRNT                ;EXECUTE
429  003530  004737  007756                    JSR     PC,DELAY
430
```

```
 432                                    ;;*************************************************************************
 (3)                                    ;*TEST 11        I       CURSORS ON GRAPH 0
 (3)                                    ;;*************************************************************************
 (2)  003534  000004                    TST11:  SCOPE
 433  003536  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
 434  003542  011266                            CURGR0
 436
 437  003544  004537  005610                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
 438  003550     002     102                    .BYTE   BIT1,LDG0               ;GRAPH 0
 439  003552  000000                            .WORD   0                       ;DATA TO BE LOADED
 440  003554  004737  006632                    JSR     PC,XPRNT                ;EXECUTE IT
 441
 442  003560  004537  006176                    JSR     R5,CURSOR               ;ENABLE CURSORS
 443  003564     002     103                    .BYTE   BIT1,LDC0
 444  003566  002000                            .WORD   ADDLIN                  ;SC = 0   GRAPH 0
 445
 446  003570  004737  006632                    JSR     PC,XPRNT                ;EXECUTE
 447
 448  003574  004537  006176                    JSR     R5,CURSOR               ;REMOVE CURSORS
 449  003600     002     103                    .BYTE   BIT1,LDC0               ;ON GRAPH 0
 450  003602  100000                            .WORD   BIT15
 451  003604  004737  006632                    JSR     PC,XPRNT                ;EXECUTE
 452  003610  004737  007756                    JSR     PC,DELAY
 453
 454                                    ;;*************************************************************************
 (3)                                    ;*TEST 12        J       CURSORS ON GRAPH 1
 (3)                                    ;;*************************************************************************
 (2)  003614  000004                    TST12:  SCOPE
 455  003616  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
 456  003622  011311                            CURGR1
 458
 459  003624  004537  005610                    JSR     R5,UPDWN                ;LOAD DATA PATTERN
 460  003630     004     112                    .BYTE   BIT2,LDG1               ;GRAPH 1 DECREMENTING PAT.
 461  003632  100354                            .WORD   BIT15!MAXHOZ            ;MAX #
 462  003634  004737  006632                    JSR     PC,XPRNT                ;EXECUTE
 463
 464  003640  004537  006176                    JSR     R5,CURSOR               ;ENABLE CURSORS
 465  003644     004     113                    .BYTE   BIT2,LDC1
 466  003646  002000                            .WORD   ADDLIN
 467
 468  003650  004737  006632                    JSR     PC,XPRNT                ;EXECUTE
 469  003654  004537  006176                    JSR     R5,CURSOR               ;REMOVE CURSOR ON GRAPH 1
 470  003660     004     113                    .BYTE   BIT2,LDC1
 471  003662  100000                            .WORD   BIT15
 472  003664  004737  006632                    JSR     PC,XPRNT                ;EXECUTE
 473  003670  004737  007756                    JSR     PC,DELAY
```

```
                        ;;*****************************************************************
                        ;*TEST 13        K       TEST STARTING COORDINATE ON GRAPH 0
                        ;;*****************************************************************
                        ;
003674  000004          TST13:  SCOPE
003676  004537  010164          JSR     R5,AMSG                 ;DISPLAY HEADER
003702  011334          SCORD0
                        ;
003704  004537  005610          JSR     R5,UPDWN                ;LOAD GRAPH 0 DATA
003710     002     102          .BYTE   BIT1,LDG0
003712  100354                  .WORD   BIT15!MAXHOZ
003714  004737  006632          JSR     PC,XPRNT                ;EXECUTE
                        ;
003720  012703  000004          MOV     #4,R3                   ;LOAD SINE COUNTER
003724  012701  000777          MOV     #MAXVRT-1,R1            ;LOAD STARTING CORD.
003730  012700  005537          MOV     #SINEND-40,R0           ;LOAD SA FOR FIRST PASS
003734  000402                  BR      1$
003736  012700  005577  3$:     MOV     #SINEND,R0              ;LOAD SINE POINTER
003742  114037  003762  1$:     MOVB    -(R0),10$               ;LOAD SINE DATA  WORD
003746  001413                  BEQ     2$                      ;BR IF NO MORE DATA
003750  010137  003764          MOV     R1,11$                  ;LOAD STARTING COORDINATE
                        ;
003754  004537  006014          JSR     R5,STCORD               ;LOAD DATA INTO BUFFER
003760     002     102          .BYTE   BIT1,LDG0               ;FOR GRAPH 0
003762  000000          10$:    .WORD   0                       ;DATA TO BE LOADED
003764  000000          11$:    .WORD   0                       ;STARTING COORD.
                        ;
003766  004737  006632          JSR     PC,XPRNT                ;EXECUTE
003772  005301                  DEC     R1                      ;DONE ALL COORD. ?
003774  000762                  BR      1$
003776  005303          2$:     DEC     R3                      ;FINISHED ALL LINES ?
004000  001356                  BNE     3$                      ;BR IF NOT
004002  004737  007756          JSR     PC,DELAY                ;
                        ;
```

```
                                ;;*****************************************************************
                                ;*TEST 14       L       TEST STARTING COORDINATE ON GRAPH 1
                                ;;*****************************************************************
004006  000004                  TST14:  SCOPE
004010  004537  010164                  JSR     R5,AMSG                 ;DISPLAY HEADER
004014  011400                          SCORD1
004016  004537  005610                  JSR     R5,UPDWN                ;LOAD GRAPH 0 DATA
004022     004     112                  .BYTE   BIT2,LDG1
004024  100354                          .WORD   BIT15!MAXHOZ
004026  004737  006632                  JSR     PC,XPRNT                ;EXECUTE
004032  012703  000004                  MOV     #4,R3                   ;LOAD SINE COUNT
004036  012701  000777                  MOV     #MAXVRT-1,R1            ;LOAD STARTINC CORD.
004042  012700  005537                  MOV     #SINEND-40,R0           ;LOAD SA POINTER
004046  000402                          BR      1$
004050  012700  005577          3$:     MOV     #SINEND,R0              ;LOAD SINE POINTER
004054  114037  004074          1$:     MOVB    -(R0),10$               ;LOAD SINE DATA  WORD
004060  001413                          BEQ     2$                      ;BR IF NO MORE DATA
004062  010137  004076                  MOV     R1,11$                  ;LOAD STARTING COORDINATE
004066  004537  006014                  JSR     R5,STCORD               ;LOAD DATA INTO BUFFER
004072     004     112                  .BYTE   BIT2,LDG1               ;FOR GRAPH 1
004074  000000                  10$:    .WORD   0                       ;DATA TO BE LOADED
004076  000000                  11$:    .WORD   0                       ;STARTING COORD.
004100  004737  006632                  JSR     PC,XPRNT                ;EXECUTE
004104  005301                          DEC     R1                      ;DONE ALL COORD. ?
004106  000762                          BR      1$
004110  005303                  2$:     DEC     R3                      ;DONE ALL LINES ?
004112  001356                          BNE     3$                      ;BR IF NOT
004114  004737  007756                  JSR     PC,DELAY                ;
```

```
                                        ;;********************* ********************************************
                                        ;*TEST 15        M       GRAPH ASPECT RATIO AND INTERACTIVE TEST ENABLE
                                        ;;*****************************************************************
(2)  004120  000004                     TST15:  SCOPE
555  004122  005737  001200                       TST     $PASS                   ;TEST IF FIRST PASS
556  004126  001455                               BEQ     TST16           ;;BYPASS IF FIRST PASS
557  004130  004537  010164                       JSR     R5,AMSG                 ;DISPLAY HEADER
558  004134  011627                               GAITE
559                                     ;ENABLE TEST MODE AND RECTANGLE RATIO
561  004136  012700  014540                       MOV     #BUFFER,R0              ;LOAD OUTPUT POINTER
562  004142  112720  000033                       MOVB    #ESC,(R0)+              ;LOAD ENTER GRAPH MODE
563  004146  112720  000061                       MOVB    #GRON,(R0)+
564  004152  112720  000111                       MOVB    #LDE1,(R0)+             ;LOAD 55 RATIO AND TEST ENABLE
565  004156  112720  000040                       MOVB    #40,(R0)+
566  004162  112720  000042                       MOVB    #42,(R0)+
567  004166  112720  000101                       MOVB    #LDE0,(R0)+             ;AND NOW ENABLE DISPLAY
568  004172  112720  000051                       MOVB    #51,(R0)+               ;ON GRAPH 0 HISTOGRAM
569  004176  105020                               CLRB    (R0)+                   ;LOAD TERM
570  004200  004737  006632                       JSR     PC,XPRNT                ;DISPLAY
571  004204  004737  007756                       JSR     PC,DELAY
572
573                                     ;NOW CHANGE TO SQUARE RATIO AND GRAPH 1
574
575  004210  012700  014540                       MOV     #BUFFER,R0              ;LOAD OUTPUT POINTER
576  004214  112720  000033                       MOVB    #ESC,(R0)+              ;LOAD ENTER GRAPH MODE
577  004220  112720  000061                       MOVB    #GRON,(R0)+
578  004224  112720  000111                       MOVB    #LDE1,(R0)+             ;ENABLE VT105 RATIO
579  004230  112720  000040                       MOVB    #40,(R0)+
580  004234  112720  000041                       MOVB    #41,(R0)+
581  004240  112720  000101                       MOVB    #LDE0,(R0)+             ;AND NOW ENABLE DISPLAY
582  004244  112720  000061                       MOVB    #61,(R0)+               ;ON GRPAH 1 HISTOGRAM
583  004250  105020                               CLRB    (R0)+                   ;LOAD TERM
584  004252  004737  006632                       JSR     PC,XPRNT                ;DISPLAY
585  004256  004737  007756                       JSR     PC,DELAY
```

```
                                ;;********************* ***************************************
                                ;*TEST 16       N       CHARACTER ASPECT RATIO
                                ;;*************************************************************
                                TST16:  SCOPE
                                ;FILL THE SCREEN WITH THE H CHARACTER

004264  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS
004270  001545                          BEQ     TST17           ;;BYPASS IF FIRST PASS
004272  004537  010164                  JSR     R5,AMSG                 ;DISPLAY HEADER
004276  011706                          CARAT
004300  012702  000014                  MOV     #12.,R2                 ;LOAD COUNT
004304  012700  014540                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
004310  112720  000015                  MOVB    #15,(R0)+               ;LOAD CR
004314  112720  000012                  MOVB    #12,(R0)+               ;LOAD LF
004320  112720  000015                  MOVB    #15,(R0)+               ;LOAD CR
004324  112720  000012                  MOVB    #12,(R0)+               ;LOAD LF
004330  012701  000050                  MOV     #40.,R1                 ;LOAD SCREEN WIDTH
004334  012720  044110          1$:     MOV     #44110,(R0)+            ;LOAD ASCII H
004340  005301                          DEC     R1                      ;FINISHED ?
004342  001374                          BNE     1$                      ;BR IF NOT
004344  105020                          CLRB    (R0)+                   ;LOAD TERM.
004346  004737  006632          2$:     JSR     PC,XPRNT                ;XFER TO SCREEN
004352  005302                          DEC     R2                      ;FINISHED ALL LINES
004354  001374                          BNE     2$                      ;BR IF NOT

                                ;NOW INSTALL THE HORIZONTAL LINES AT A SQUARE RATIO

004356  012700  014540                  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
004362  112720  000033                  MOVB    #ESC,(R0)+              ;ENABLE CHART MODE
004366  112720  000061                  MOVB    #GRON,(R0)+             ;
004372  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
004376  112720  000041                  MOVB    #BIT5!BIT0,(R0)+        ;LOAD DISPLAY ENABLE
004402  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
004406  112720  000043                  MOVB    #BIT5!BIT1!BIT0,(R0)+   ;ENABLE HORIZ.+ VERT LINES
004412  112720  000041                  MOVB    #BIT5!BIT0,(R0)+        ;ENSURE 105 RATIO
004416  112720  000104                  MOVB    #LHV0,(R0)+             ;LOAD HORIZ INST
004422  012737  002002  006374          MOV     #ADDLIN+2,BASE          ;LOAD BASE LINE VALUE
004430  004737  006400          3$:     JSR     PC,SHUFF                ;SHUFFEL THE DATA
004434  010120                          MOV     R1,(R0)+                ;SAVE THE DATA
004436  062737  000024  006374          ADD     #24,BASE                ;UPDATE BASE LINE VALUE
004444  022737  002340  006374          CMP     #ADDLIN+340,BASE        ;TEST FOR GREATER THAN VALID
004452  100366                          BPL     3$                      ;BR IF OK
004454  105020                          CLRB    (R0)+                   ;LOAD TERM
004456  004737  006632                  JSR     PC,XPRNT                ;XMIT TO THE SCREEN
```

```
                                        ;NOW LOAD THE VERTICAL LINES
                                
004462  012700  014540                        MOV     #BUFFER,R0              ;LOAD OUTPUT BUFFER POINTER
004466  112720  000033                        MOVB    #ESC,(R0)+              ;LOAD ESC
004472  112720  000061                        MOVB    #GRON,(R0)+             ;ENTER CHART MODE
004476  112720  000100                        MOVB    #LNO,(R0)+              ;LOAD NOP
004502  112720  000114                        MOVB    #LHV1,(R0)+             ;LOAD VERT LINE INST.
004506  012737  002000  006374                MOV     #ADDLIN,BASE            ;LOAD STARTING LINE #
004514  004737  006400              4$:       JSR     PC,SHUFF                ;SHUFFEL THE DATA
004520  010120                              MOV     R1,(R0)+                ;SAVE DATA
004522  062737  000015  006374              ADD     #15,BASE                ;UPDATE DATA
004530  022737  003000  006374              CMP     #ADDLIN!1000,BASE       ;TEST FOR LAST DATA LINE
004536  100366                              BPL     4$                      ;BR IF NOT DONE
004540  105020                              CLRB    (R0)+                   ;LOAD TERM.
004542  004737  006632                      JSR     PC,XPRNT                ;SEND TO SCREEN
                                
004546  004737  007756                      JSR     PC,DELAY                ;DELAY
                                
                                        ;NOW CHANGE TO VT55 RATIO
                                
004552  012700  014540                        MOV     #BUFFER,R0              ;LOAD POINTER
004556  112720  000111                        MOVB    #LDE1,(R0)+             ;LOAD ENABLE 1
004562  112720  000043                        MOVB    #BIT5!BIT1!BIT0,(R0)+   ;LOAD WORD 1
004566  112720  000040                        MOVB    #BIT5,(R0)+             ;LOAD 55 RATIO
004572  105020                              CLRB    (R0)+                   ;LOAD TERM
004574  004737  006632                      JSR     PC,XPRNT                ;SEND TO SCREEN
004600  004737  007756                      JSR     PC,DELAY                ;DELAY
```

```
  660
  661                                   ;;********************* ***************************************************
  (3)                                   ;*TEST 17       0       BASE LINE ON HISTOGRAM 0
  (3)                                   ;;*************************************************************************
  (2)  004604  000004                   TST17:  SCOPE
  662  004606  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
  663  004612  011444                            BSLGR0
  665
  666  004614  004537  005610                    JSR     R5,UPDWN                ;LOAD ACCENDING VECTOR ON
  667  004620     010     102                     .BYTE   BIT3,LDG0               ;GRAPH 0
  668  004622  000000                            .WORD   0                       ;STARTING AT C
  669
  670  004624  004537  006476                    JSR     R5,LDBASE               ;LOAD MOVING BASE LINE
  671  004630     041     042                     .BYTE   41,42
  672
  673  004632  004737  006632                    JSR     PC,XPRNT                ;DISPLAY IT
  674  004636  004737  007756                    JSR     PC,DELAY                ;DELAY
  675
  676                                   ;;*************************************************************************
  (3)                                   ;*TEST 20       P       BASE LINE ON HISTOGRAM 1
  (3)                                   ;;*************************************************************************
  (2)  004642  000004                   TST20:  SCOPE
  677  004644  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
  678  004650  011475                            BSLGR1
  680
  681  004652  004537  005610                    JSR     R5,UPDWN                ;LOAD DECENDING VECTOR ON
  682  004656     020     112                     .BYTE   BIT4,LDG1               ;GRAPH 1
  683  004660  100354                            .WORD   BIT15!MAXHOZ            ;STARTING AT MAX
  684
  685  004662  004537  006476                    JSR     R5,LDBASE               ;LOAD MOVING BASE LINE
  686  004666     041     045                     .BYTE   41,45
  687
  688  004670  004737  006632                    JSR     PC,XPRNT                ;DISPLAY IT
  689  004674  004737  007756                    JSR     PC,DELAY                ;DELAY
```

```
  691
  692                                   ;;********************* ***************************************************
  (3)                                   ;*TEST 21        Q       STRIP CHART ON GRAPH 0
  (3)                                   ;;***************************************************************************
  (2)  004700  000004                   TST21:  SCOPE
  693  004702  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
  694  004706  011526                            STCGR0
  696
  697  004710  004537  005600                    JSR     R5,UPDWNS               ;LOAD ACCENDING GRAPH
  698  004714     012     102                    .BYTE   BIT3!BIT1,LDG0          ;ON GRAPH 0
  699  004716  000000                            .WORD   0                       ;STARTING AT MIN.
  700                                   ;NOW CONTINUE LOADING DATA INTO GRAPH 0
  701  004720  004737  006570                    JSR     PC,STRIPG               ;LOAD MORE DATA <SINE WAVE>
  702  004724  004737  006632                    JSR     PC,XPRNT                ;DISPLAY IT
  703  004730  004737  007756                    JSR     PC,DELAY
  704
  705                                   ;;***************************************************************************
  (3)                                   ;*TEST 22        R       STRIP CHART ON GRAPH 1
  (3)                                   ;;***************************************************************************
  (2)  004734  000004                   TST22:  SCOPE
  706  004736  004537  010164                    JSR     R5,AMSG                 ;DISPLAY HEADER
  707  004742  011554                            STCGR1
  709
  710  004744  004537  005600                    JSR     R5,UPDWNS               ;LOAD DECENDING GRAPH
  711  004750     024     112                    .BYTE   BIT4!BIT2,LDG1          ;ON GRAPH 1
  712  004752  100354                            .WORD   BIT15!MAXHOZ            ;STARTING AT MAX.
  713                                   ;NOW CONTINUE LOADING DATA INTO GRAPH 1
  714  004754  004737  006570                    JSR     PC,STRIPG               ;LOAD MORE DATA <SINE WAVE>
  715  004760  004737  006632                    JSR     PC,XPRNT                ;DISPLAY IT
  716  004764  004737  007756                    JSR     PC,DELAY                ;DELAY
```

```
                        ;;*****************************************************************
                        ;*TEST 23       S       DUAL STRIPCHART MODE
                        ;;*****************************************************************
004770  000004          TST23:  SCOPE
004772  004537  010164          JSR     R5,AMSG                 ;DISPLAY HEADER
004776  011602                  DUSTC
                        ;NOW LOAD DATA IN STRIP CHART MODE
005000  012700  014540          MOV     #BUFFER,R0              ;LOAD POINTER
005004  004537  010302          JSR     R5,MTOB                 ;LOAD MESSAGE INTO BUFFER
005010  006152                  SETBAS
005012  105740                  TSTB    -(R0)                   ;BACK UP OVER TERM.
005014  112720  000100          MOVB    #LNO,(R0)+
005020  112720  000101          MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
005024  112720  000047          MOVB    #47,(R0)+               ;DISPLAY ENABLE
005030  112720  000066          MOVB    #66,(R0)+               ;DUAL STRIPCHART ENABLE
005034  012703  005367          MOV     #SINBEG+1,R3            ;LOAD GRAPH 0 POINTER
005040  012704  005472          MOV     #SINBEG+104,R4          ;LOAD GRAPH 1 POINTER
005044  012737  000012  001302  MOV     #10.,SAVE4              ;LOAD # OF WAVES TO BE SEEN
005052  112337  006374  1$:     MOVB    (R3)+,BASE              ;GET BYTE OF DATA
005056  001425                  BEQ     4$                      ;BR IF DONE
005060  004737  006400          JSR     PC,SHUFF                ;CONVERT DATA INTO FORMAT
005064  112720  000100          MOVB    #LNO,(R0)+              ;LOAD NOP TO ENSURE WORD EDGE
005070  112720  000102          MOVB    #LDG0,(R0)+             ;LOAD 'LOAD GRAPH 0 DATA''
005074  010120                  MOV     R1,(R0)+                ;LOAD SHUFFLED DATA
005076  112437  006374  2$:     MOVB    (R4)+,BASE              ;GET BYTE OF GRAPH 1 DATA
005102  001410                  BEQ     3$                      ;BR IF NO MORE
005104  004737  006400          JSR     PC,SHUFF                ;CONVERT DATA INTO FORMAT
005110  112720  000100          MOVB    #LNO,(R0)+              ;LOAD NOP TO ENSURE WORD EDGE
005114  112720  000112          MOVB    #LDG1,(R0)+             ;LOAD 'LOAD GRAPH 1 DATA''
005120  010120                  MOV     R1,(R0)+                ;LOAD SHUFFLED DATA
005122  000753                  BR      1$
005124  012704  005367  3$:     MOV     #SINBEG+1,R4            ;RELOAD GRAPH 1 POINTER
005130  000762                  BR      2$
005132  012703  005367  4$:     MOV     #SINBEG+1,R3            ;RELOAD GRAPH 0 POINTER
005136  005337  001302          DEC     SAVE4                   ;FINISHED ?
005142  001343                  BNE     1$
005144  112720  000033          MOVB    #ESC,(R0)+              ;ENSURE TEXT MODE
005150  112720  000062          MOVB    #GROF,(R0)+             ;SO END OF PASS IS OK
005154  105020                  CLRB    (R0)+                   ;TERM
005156  004737  006632          JSR     PC,XPRNT                ;DISPLAY
005162  004737  007756          JSR     PC,DELAY
                        ;NOW CLEAR THE GRAPH OFF THE SCREEN
005166  004537  010164          JSR     R5,AMSG                 ;USE THE HEADER ROUTINE TO
005172  011766                  ZAPITB                          ;CLEAR THE SCREEN
```

```
   763                                  .SBTTL  END OF PASS ROUTINE
   (1)
   (2)                                  ;;**********************************************************************
   (1)                                  ;*INCREMENT THE PASS NUMBER ($PASS)
   (1)                                  ;*TYPE 'END PASS #XXXXX'' (WHERE XXXXX IS A DECIMAL NUMBER)
   (1)                                  ;*IF THERES A MONITOR GO TO IT
   (1)                                  ;*IF THERE ISN'T JUMP TO RSTRTA
   (1)
   (1)  005174                          $EOP:
   (3)  005174  000004                          SCOPE
   (3)  005176  005737  001256                  TST     LAST                    ;TEST IF MORE
   (3)  005202  001411                          BEQ     1$                      ;BR IF NONE
   (3)  005204  023737  001256  001260          CMP     LAST,VTNOW              ;IS THIS THE LAST ONE
   (3)  005212  001405                          BEQ     1$                      ;BR IF YES
   (3)  005214  062737  000010  001260          ADD     #10,VTNOW
   (3)  005222  000137  002054                  JMP     RSTRT                   ;TEST NEXT ONE
   (3)  005226  112737  000040  001304  1$:     MOVB    #40,ASPTRB              ;LOAD VT55 RATIO
   (3)  005234  032737  000001  001200          BIT     #BIT0,$PASS             ;TEST IF EVEN
   (3)  005242  001002                          BNE     2$                      ;BR IF NOT
   (3)  005244  105237  001304                  INCB    ASPTRB                  ;MAKE 105 RATIO
   (3)  005250  000240                  2$:     NOP
   (1)  005252  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
   (1)  005256  005237  001200                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
   (1)  005262  042737  100000  001200          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
   (1)  005270  005327                          DEC     (PC)+           ;;LOOP?
   (1)  005272  000001                  $EOPCT: .WORD   1
   (1)  005274  003022                          BGT     $DOAGN          ;;YES
   (1)  005276  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
   (1)  005300  000001                  $ENDCT: .WORD   1
   (1)  005302  005272                          $EOPCT
   (1)  005304  104401  005351                  TYPE    ,$ENDMG         ;;TYPE 'END PASS #''
   (2)  005310  013746  001200                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
   (2)  005314  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
   (1)  005316  104401  005346                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
   (1)  005322  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
   (1)  005326  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
   (1)  005330  000005                          RESET                   ;;CLEAR THE WORLD
   (1)  005332  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
   (1)  005334  000240                          NOP                     ;;SAVE ROOM
   (1)  005336  000240                          NOP                     ;;FOR
   (1)  005340  000240                          NOP                     ;;ACT11
   (1)  005342                          $DOAGN:
   (1)  005342  000137                          JMP     @(PC)+          ;;RETURN
   (1)  005344  002046                  $RTNAD: .WORD   RSTRTA
   (1)  005346     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
   (1)  005351     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
   (1)  005356  050040  051501  020123
   (1)  005364  000043
```

```
765  005366  000  003  004  SINBEG: .BYTE  000,003,004,004,005
     005371  004  005
766  005373  006  007  010          .BYTE  006,007,010,012,014,015,017,021,024,027,032,035,040
     005376  012  014  015
     005401  017  021  024
     005404  027  032  035
     005407  040
767  005410  043  047  053          .BYTE  043,047,053,056,062,066,073,077,103,110,115,121,126,133
     005413  056  062  066
     005416  073  077  103
     005421  110  115  121
     005424  126  133
768  005426  137  144  151          .BYTE  137,144,151,156,162,167,174,201,206,213,217,223,230,235
     005431  156  162  167
     005434  174  201  206
     005437  213  217  223
     005442  230  235
769  005444  241  245  251          .BYTE  241,245,251,255,261,265,271,274,277,303,305,310,313
     005447  255  261  265
     005452  271  274  277
     005455  303  305  310
     005460  313
770  005461  316  317  321          .BYTE  316,317,321,323,324,326,327,330,331,331,331,331,331
     005464  323  324  326
     005467  327  330  331
     005472  331  331  331
     005475  331
771  005476  330  327  326          .BYTE  330,327,326,325,323,322,320,317,314,312,307
     005501  325  323  322
     005504  320  317  314
     005507  312  307
772  005511  304  301  275          .BYTE  304,301,275,272,267,263,257,253,247,243,237,232,225
     005514  272  267  263
     005517  257  253  247
     005522  243  237  232
     005525  225
773  005526  221  214  207          .BYTE  221,214,207,203,176,171,164,157,153,145,140
     005531  203  176  171
     005534  164  157  153
     005537  145  140
774  005541  134  127  121          .BYTE  134,127,121,115,111,104,077,073,066,063,057,053,047
     005544  115  111  104
     005547  077  073  066
     005552  063  057  053
     005555  047
775  005556  043  040  034          .BYTE  043,040,034,031,027,023,021,017,015,013,011,010,007
     005561  031  027  023
     005564  021  017  015
     005567  013  011  010
     005572  007
776  005573  006  005  004          .BYTE  006,005,004,003
     005576  003
777  005577  000              SINEND: .BYTE  0
778                                   .EVEN
```

```
                                        ;UP-DOWN SUBROUTINE

005600  012737  000050  006012  UPDWNS: MOV     #50,UPDWNZ              ;LOAD
005606  000403                          BR      UPDWNX
005610  012737  000040  006012  UPDWN:  MOV     #40,UPDWNZ              ;LOAD
005616  012537  006006          UPDWNX: MOV     (R5)+,11$               ;GET GRAPH BIT AND INC/DEC WORD
005622  012537  006010                  MOV     (R5)+,12$               ;GET STARTING CORD.
005626  012700  014540                  MOV     #BUFFER,R0              ;LOAD THE POINTER
005632  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD 'ESCAPE'
005636  112720  000061                  MOVB    #GRON,(R0)+             ;ENABLE GRAPHIC MODE
005642  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
005646  112710  000041                  MOVB    #BIT5!BIT0,(R0)         ;ENABLE DISPLAY
005652  153720  006006                  BISB    11$,(R0)+               ;LOAD GRAPH ENABLE BIT
005656  012737  000001  006004          MOV     #1,10$
005664  113720  006012                  MOVB    UPDWNZ,(R0)+            ;LOAD NOP
005670  112720  000110                  MOVB    #LSC,(R0)+              ;LOAD STARTING CORD.
005674  005037  006374                  CLR     BASE                    ;LOAD START. CORD.
005700  004737  006400                  JSR     PC,SHUFF
005704  010120                          MOV     R1,(R0)+                ;LOAD INTO BUFFER
005706  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
005712  113720  006007                  MOVB    11$+1,(R0)+             ;LOAD LOAD GRAPH X
005716  013737  006010  006374  4$:     MOV     12$,BASE                        ;LOAD DATA
005724  004737  006400          1$:     JSR     PC,SHUFF                ;SHUFFEL DATA
005730  010120                          MOV     R1,(R0)+                ;LOAD DATA
005732  005737  006010                  TST     12$                     ;TEST FOR UP OR DOWN DATA
005736  100007                          BPL     2$                      ;BR IF INC.
005740  005337  006374                  DEC     BASE                    ;DEC. DATA
005744  042737  177400  006374          BIC     #177400,BASE
005752  001407                          BEQ     5$
005754  000763                          BR      1$
005756  005237  006374          2$:     INC     BASE                    ;CHANGE DATA
005762  022737  000354  006374  3$:     CMP     #MAXHOZ,BASE            ;TEST FOR LAST
005770  001355                          BNE     1$                      ;BR IF NOT
005772  005337  006004          5$:     DEC     10$
005776  100347                          BPL     4$                      ;BR IF NOT
006000  105010                          CLRB    (R0)                    ;LOAD TERMINATOR
006002  000205                          RTS     R5                      ;EXIT
006004  000000                  10$:    0
006006  000000                  11$:    0
006010  000000                  12$:    0
006012  000040                  UPDWNZ: 40
```

```
                                        ;STARTING CORDINATE SUBROUTINE
                                        
006014  012537  006146          STCORD: MOV     (R5)+,11$               ;GET GRAPH BIT AND INC/DEC WORD
006020  012537  006150                  MOV     (R5)+,12$               ;GET STARTING CORD.
006024  012537  006144                  MOV     (R5)+,10$               ;GET ARG. WORD
006030  010046                          MOV     R0,-(SP)                ;SAVE R0
006032  010146                          MOV     R1,-(SP)                ;SAVE R1
006034  012700  014540                  MOV     #BUFFER,R0              ;LOAD THE POINTER
006040  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD 'ESCAPE'
006044  112720  000061                  MOVB    #GRON,(R0)+             ;ENABLE GRAPHIC MODE
006050  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
006054  112710  000041                  MOVB    #BIT5!BIT0,(R0)         ;ENABLE DISPLAY
006060  153720  006146                  BISB    11$,(R0)+               ;LOAD GRAPH ENABLE BIT
006064  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
006070  112720  000110                  MOVB    #LSC,(R0)+              ;LOAD STARTING CORD.
006074  013737  006144  006374          MOV     10$,BASE                ;LOAD START. CORD.
006102  004737  006400                  JSR     PC,SHUFF
006106  010120                          MOV     R1,(R0)+                ;LOAD INTO BUFFER
006110  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
006114  113720  006147                  MOVB    11$+1,(R0)+             ;LOAD LOAD GRAPH X
006120  013737  006150  006374  4$:     MOV     12$,BASE                ;LOAD DATA
006126  004737  006400          1$:     JSR     PC,SHUFF                ;SHUFFEL DATA
006132  010120                          MOV     R1,(R0)+                ;LOAD DATA
006134  105010                          CLRB    (R0)                    ;LOAD TERMINATOR
006136  012601                          MOV     (SP)+,R1
006140  012600                          MOV     (SP)+,R0
006142  000205                          RTS     R5                      ;EXIT
006144  000000                  10$     0
006146  000000                  11$:    0
006150  000000                  12$:    0
                                        
006152      033     061         SETBAS: .BYTE   ESC,GRON                ;ENABLE GRAPH MODE
006154      101     040     042         .BYTE   LDE0,40,42              ;ENABLE BASE 0 TO BE LOADED
006157      100     060     043         .BYTE   LNO,60,43               ;LOAD BASE 0
006162      101     040     043         .BYTE   LDE0,40,43              ;ENABLE BASE 1 TO BE LOADED
006165      100     060     043         .BYTE   LNO,60,43               ;LOAD BASE 1
006170      110     040     040         .BYTE   LSC,40,40               ;LOAD STARTING CORD. TO 0
006173      100                         .BYTE   LNO
006174      000                         .BYTE   0
        006176                          .EVEN
```

```
                                ;CURSOR SUBROUTINE
                                ;THE FIRST ARG. LOW BYTE IS AN 'LDC0 OR LDC1''
                                ;                HIGH BYTE IS BIT 1 OR BIT 2 TO ENABLE CURSOR
                                ;
                                ;THE SECOND ARG. = BIT 15 WILL REMOVE CURSORS
                                ;                = BIT 10 WILL ADD CURSORS
                                ;
006176  012537  006370          CURSOR: MOV     (R5)+,11$               ;GET ARG. WORD
006202  012537  006372                  MOV     (R5)+,12$               ;GET ARG. WORD
006206  012700  014540                  MOV     #BUFFER,R0              ;LOAD THE POINTER
006212  112720  000033                  MOVB    #ESC,(R0)+              ;LOAD 'ESCAPE'
006216  112720  000061                  MOVB    #GRON,(R0)+             ;ENABLE GRAPHIC MODE
006222  112720  000101                  MOVB    #LDE0,(R0)+             ;LOAD ENABLE 0
006226  112710  000041                  MOVB    #BIT5!BIT0,(R0)         ;ENABLE DISPLAY
006232  153720  006370                  BISB    11$,(R0)+               ;LOAD GRAPH ENABLE BIT
006236  112720  000111                  MOVB    #LDE1,(R0)+             ;LOAD ENABLE
006242  013701  006370                  MOV     11$,R1
006246  006301                          ASL     R1              ;MAKE PROPER CURSOR ENABLE DATA BYTE
006250  112710  000040                  MOVB    #BIT5,(R0)
006254  150120                          BISB    R1,(R0)+
006256  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
006262  112720  000110                  MOVB    #LSC,(R0)+              ;LOAD STARTING CORD.
006266  005037  006374                  CLR     BASE                    ;LOAD START. CORD.
006272  004737  006400                  JSR     PC,SHUFF
006276  010120                          MOV     R1,(R0)+                ;LOAD INTO BUFFER
006300  112720  000100                  MOVB    #LNO,(R0)+              ;LOAD NOP
006304  113720  006371                  MOVB    11$+1,(R0)+             ;LOAD 'LOAD CURSOR ON GRAPH X''
006310  013737  006372  006374          MOV     12$,BASE                ;LOAD CURSOR POSITION DATA
006316  004737  006400          1$:     JSR     PC,SHUFF                ;SHUFFEL DATA
006322  010120                          MOV     R1,(R0)+                ;LOAD DATA
006324  005737  006372                  TST     12$                     ;TEST FOR UP OR DOWN DATA
006330  100007                          BPL     2$                      ;BR IF INC.
006332  005337  006374                  DEC     BASE
006336  042737  177000  006374          BIC     #177000,BASE
006344  001407                          BEQ     4$
006346  000763                          BR      1$
006350  005237  006374          2$:     INC     BASE                    ;CHANGE DATA
006354  022737  003000  006374  3$:     CMP     #ADDLIN!MAXVRT,BASE     ;TEST FOR LAST
006362  001355                          BNE     1$                      ;BR IF NOT
006364  105010                  4$:     CLRB    (R0)                    ;LOAD TERMINATOR
006366  000205                          RTS     R5                      ;EXIT
006370  000000                  11$:    0
006372  000000                  12$:    0
006374  000000                  BASE:   0
006376  000000                  BASE1:  0
```

```
                                        ;SUBROUTINE TO SHUFFEL DATA INTO GRAPHIC DATA BYTE FORMAT
                                        
006400 013702 006374            SHUFF:  MOV     BASE,R2         ;LOAD VALUE TO BE SHUFFELED
006404 010237 006376                    MOV     R2,BASE1
006410 042737 177740 006376             BIC     #177740,BASE1
006416 010201                           MOV     R2,R1
006420 006001                           ROR     R1
006422 006001                           ROR     R1
006424 006001                           ROR     R1
006426 006001                           ROR     R1
006430 006001                           ROR     R1
006432 110137 006377                    MOVB    R1,BASE1+1              ;RELOAD R1
006436 042737 170340 006376             BIC     #170340,BASE1   ;MASK
006444 052737 020040 006376             BIS     #20040,BASE1    ;CONVERT TO ASCII
006452 032737 002000 006374             BIT     #BIT10,BASE
006460 001403                           BEQ     1$              ;BR IF NOT SET
006462 052737 010000 006376             BIS     #BIT12,BASE1    ;SET BIT
006470 013701 006376            1$:     MOV     BASE1,R1
006474 000207                           RTS     PC              ;EXIT
                                        ;SUBROUTINE TO LOAD VARIABLE BASE LINE
006476 112720 000033            LDBASE: MOVB    #ESC,(R0)+      ;LOAD ESC
006502 112720 000061                    MOVB    #GRON,(R0)+     ;LOAD GRAPH MODE
006506 112720 000101                    MOVB    #LDE0,(R0)+     ;LOAD ENABLE 0
006512 112520                           MOVB    (R5)+,(R0)+     ;LOAD ENABLE 0 DATA BITS
006514 112520                           MOVB    (R5)+,(R0)+     ;LOAD ENABLE 0 2ND WORD DATA BITS
006516 005037 006374                    CLR     BASE            ;CLEAR BASE VALUE
006522 112720 000100                    MOVB    #LNO,(R0)+      ;ENSURE WORD ADDR.
                                        ;NOW RAISE THE BASE LINE
006526 004737 006400            1$:     JSR     PC,SHUFF        ;CONVERT BASE VALUE TO DATA
006532 010120                           MOV     R1,(R0)+        ;SAVE BASE LINE VALUE
006534 005237 006374                    INC     BASE            ;UPDATE TO NEXT BASE LINE VALUE
006540 022737 000354 006374             CMP     #MAXHOZ,BASE    ;TEST IF DONE
006546 001367                           BNE     1$              ;BR IF NOT
                                        ;NOW LOWER THE BASE LINE
006550 004737 006400            2$:     JSR     PC,SHUFF        ;CONVERT BASE VALUE INTO DATA
006554 010120                           MOV     R1,(R0)+        ;SAVE BASE LINE VALUE
006556 005337 006374                    DEC     BASE            ;UPDATE TO PREVIOUS BASE LINE VALUE
006562 001372                           BNE     2$              ;BR IF NOT DONE
006564 005010                           CLR     (R0)            ;LOAD TERM
006566 000205                           RTS     R5              ;EXIT
                                        ;SUBROUTINE TO LOAD SINE WAVE TO THE END OF CURRENT BUFFER
006570 005037 006374            STRIPG: CLR     BASE            ;PRIME THE BASE VALUE
006574 012704 000003                    MOV     #3,R4           ;LOAD # OF SINE WAVES SEEN
006600 012703 005367            1$:     MOV     #SINBEG+1,R3    ;LOAD SINE WAVE POINTER
006604 112337 006374            2$:     MOVB    (R3)+,BASE      ;GET A SINE WAVE DATA BYTE
006610 001404                           BEQ     3$              ;BR IF END OF SINE WAVE DATA
006612 004737 006400                    JSR     PC,SHUFF        ;CONVERT BASE INTO DATA
006616 010120                           MOV     R1,(R0)+        ;LOAD THE DATA INTO THE BUFFER
006620 000771                           BR      2$              ;BR AND DO MORE DATA
006622 005304                   3$:     DEC     R4              ;FINISHED ALL SINE WAVES
006624 100365                           BPL     1$              ;BR IF NOT
006626 005010                           CLR     (R0)            ;LOAD TERM
006630 000207                           RTS     PC              ;EXIT
```

```
                                        ;DISPLAY SUBROUTINE
                                        
006632  010046                  XPRNT:  MOV     R0,-(SP)
006634  010146                          MOV     R1,-(SP)
006636  010246                          MOV     R2,-(SP)
006640  005037  007534                  CLR     ANESC           ;HOUSEKEEP
006644  005037  007540                  CLR     TERM
006650  005037  007536                  CLR     NOEXIT
006654  012700  014540                  MOV     #BUFFER,R0      ;SETUP BUFFER POINTER
006660  105777  172412          1$:     TSTB    @VTOS           ;TEST READY
006664  100375                          BPL     1$
006666  005737  001270                  TST     WFTEST          ;TEST IF TESTER MODE
006672  001435                          BEQ     2$              ;BR IF NOT
006674  105777  001170          60$:    TSTB    @VTOS0          ;LINE 0 READY
006700  100375                          BPL     60$
006702  105777  001172          61$:    TSTB    @VTOS1          ;LINE 1 READY
006706  100375                          BPL     61$
006710  105777  001174          62$:    TSTB    @VTOS2          ;LINE 2 READY
006714  100375                          BPL     62$
006716  105777  001176          63$:    TSTB    @VTOS3          ;LINE 3 READY
006722  100375                          BPL     63$
006724  105777  001200          64$:    TSTB    @VTOS4          ;LINE 4 READY
006730  100375                          BPL     64$
006732  105777  001202          65$:    TSTB    @VTOS5          ;LINE 5 READY
006736  100375                          BPL     65$
006740  105777  001204          66$:    TSTB    @VTOS6          ;LINE 6 READY
006744  100375                          BPL     66$
006746  105777  001206          67$:    TSTB    @VTOS7          ;LINE 7 READY
006752  100375                          BPL     67$
006754  000240                          NOP
006756  000240                          NOP
006760  000240                          NOP
006762  000240                          NOP
006764  000240                          NOP
006766  112001                  2$:     MOVB    (R0)+,R1        ;GET A CHAR.
006770  001500                          BEQ     14$             ;BR IF TERM
006772  122701  000033                  CMPB    #33,R1          ;TEST FOR ESC
006776  001003                          BNE     4$              ;BR IF NOT
007000  005237  007534          3$:     INC     ANESC           ;SET SOFT FLAG
007004  000402                          BR      5$
007006  005037  007534          4$:     CLR     ANESC           ;CLEAR SOFT FLAG
007012  110177  172262          5$:     MOVB    R1,@VTOB        ;LOAD CHAR
007016  005737  001270                  TST     WFTEST          ;TEST IF TESTER MODE
007022  001450                          BEQ     68$             ;BR IF NOT
007024  110177  001042                  MOVB    R1,@VTOB0       ;LOAD LINE 0
007030  110177  001046                  MOVB    R1,@VTOB1       ;LOAD LINE 1
007034  110177  001052                  MOVB    R1,@VTOB2       ;LOAD LINE 2
007040  110177  001056                  MOVB    R1,@VTOB3       ;LOAD LINE 3
007044  110177  001062                  MOVB    R1,@VTOB4       ;LOAD LINE 4
007050  110177  001066                  MOVB    R1,@VTOB5       ;LOAD LINE 5
007054  110177  001072                  MOVB    R1,@VTOB6       ;LOAD LINE 6
007060  110177  001076                  MOVB    R1,@VTOB7       ;LOAD LINE 7
007064  105777  000774                  TSTB    @VTIS0          ;TEST INPUT LINE 0
007070  100442                          BMI     70$
```

```
1017  007072  105777  000776              TSTB    @VTIS1          ;TEST INPUT LINE 1
1018  007076  100442                      BMI     71$
1019  007100  105777  001000              TSTB    @VTIS2          ;TEST INPUT LINE 2
1020  007104  100442                      BMI     72$
1021  007106  105777  001002              TSTB    @VTIS3          ;TEST INPUT LINE 3
1022  007112  100442                      BMI     73$
1023  007114  105777  001004              TSTB    @VTIS4          ;TEST INPUT LINE 4
1024  007120  100442                      BMI     74$
1025  007122  105777  001006              TSTB    @VTIS5          ;TEST INPUT LINE 5
1026  007126  100442                      BMI     75$
1027  007130  105777  001010              TSTB    @VTIS6          ;TEST INPUT LINE 6
1028  007134  100442                      BMI     76$
1029  007136  105777  001012              TSTB    @VTIS7          ;TEST INPUT LINE 7
1030  007142  100442                      BMI     77$
1031  007144  105777  172122      68$:    TSTB    @VTIS           ;TEST INPUT FLAG
1032  007150  100243                      BPL     1$              ;BR IF CLEARED
1033  007152  005737  007534              TST     ANESC           ;TEST IF "ESC" WAS JUST SENT
1034  007156  001240                      BNE     1$
1035  007160  005037  001124              CLR     $GDDAT          ;CLEAR EXPECTED DATA
1036  007164  013702  001272              MOV     VTIS,R2         ;GET CONSOLE ADDRESS
1037  007170  000452                      BR      10$
1038  007172  000137  007476      14$:    JMP     16$
1039  007176  013702  010064      70$:    MOV     VTIS0,R2        ;GET LINE 0 STATUS
1040  007202  000445                      BR      10$
1041  007204  013702  010074      71$:    MOV     VTIS1,R2        ;GET LINE 1 STATUS
1042  007210  000442                      BR      10$
1043  007212  013702  010104      72$:    MOV     VTIS2,R2        ;GET LINE 2 STATUS
1044  007216  000437                      BR      10$
1045  007220  013702  010114      73$:    MOV     VTIS3,R2        ;GET LINE 3 STATUS
1046  007224  000434                      BR      10$
1047  007226  013702  010124      74$:    MOV     VTIS4,R2        ;GET LINE 4 STATUS
1048  007232  000431                      BR      10$
1049  007234  013702  010134      75$:    MOV     VTIS5,R2        ;GET LINE 5 STATUS
1050  007240  000426                      BR      10$
1051  007242  013702  010144      76$:    MOV     VTIS6,R2        ;GET LINE 6 STATUS
1052  007246  000423                      BR      10$
1053  007250  013702  010154      77$:    MOV     VTIS7,R2        ;GET LINE 7 STATUS
1054  007254  000420                      BR      10$
1055
1056                              ;WAIT FOR A KEYBOARD FLAG  -  EXIT IF NONE
1057
1058  007256  013737  001264  010276 6$:    MOV     TIME0,TIME1
1059  007264  005037  010300              CLR     TIME2           ;LOAD DELAY
1060  007270  105712              7$:     TSTB    (R2)            ;TEST IF INPUT FLAG
1061  007272  100411                      BMI     10$             ;BR IF SET
1062  007274  005337  010300              DEC     TIME2           ;DELAY
1063  007300  001373                      BNE     7$
1064  007302  104407                      CKSWR                   ;TEST FOR "CTRG G" ON CONSOLE TTY
1065  007304  005337  010276              DEC     TIME1           ;DELAY
1066  007310  001367                      BNE     7$
1067  007312  000137  006660      13$:    JMP     1$              ;TRY AGAIN
1068
1069                              ;INPUT FLAG SET - FIND OUT WHAT CHARACTER IT WAS
1070
```

```
1071  007316  010201                  10$:    MOV     R2,R1           ;COPY R2
1072  007320  005721                          TST     (R1)+           ;BUMP VALUE
1073  007322  011137  001126                  MOV     (R1),$BDDAT     ;READ CHAR
1074  007326  042737  177600  001126          BIC     #177600,$BDDAT  ;MASK
1075  007334  022737  000021  001126          CMP     #XON,$BDDAT     ;TEST FOR X ON
1076  007342  001006                          BNE     11$
1077  007344  005037  007536                  CLR     NOEXIT
1078  007350  005737  007540                  TST     TERM            ;TEST IF TERMINATOR WAS SET
1079  007354  001756                          BEQ     13$             ;BR IF NOT
1080  007356  000460                          BR      17$             ;BR IF ONLY ONE
1081  007360  022737  000023  001126  11$:    CMP     #XOFF,$BDDAT    ;TEST FOR X OFF
1082  007366  001006                          BNE     12$             ;BR IF NOT
1083  007370  105237  007536                  INCB    NOEXIT          ;SET 'NO EXIT UNTIL X-ON'' RCVD
1084  007374  012737  000021  001124          MOV     #XON,$GDDAT     ;LOAD EXPECTED VALUE
1085  007402  000725                          BR      6$              ;WAIT FOR X-ON
1086  007404  005037  001124          12$:    CLR     $GDDAT          ;LOAD EXPECTED
1087  007410  105777  171524                  TSTB    @SWR            ;TEST SWR
1088  007414  100336                          BPL     13$             ;BR IF CLEARED
1089  007416  020237  001272                  CMP     R2,VTIS         ;TEST IF NON-TESTER
1090  007422  001333                          BNE     13$             ;BR IF TESTER
1091  007424  022737  000057  001126          CMP     #'/,$BDDAT      ;COMPARE
1092  007432  001413                          BEQ     15$             ;;BR IF EQUAL
1093  007434  022737  000134  001126          CMP     #'\,$BDDAT      ;COMPARE
1094  007442  001323                          BNE     13$             ;BR IF NOT
1095  007444  012737  000001  007532          MOV     #1,LOOP         ;SET SOFT FLAG
1096  007452  012737  010426  007552          MOV     #MQ0,FINDTA     ;SETUP MESSAGE
1097  007460  000430                          BR      FINDOT
1098  007462  005037  007532          15$:    CLR     LOOP
1099  007466  012737  010503  007552          MOV     #MQ1,FINDTA     ;SETUP MESSAGE
1100  007474  000422                          BR      FINDOT
1101  007476  012737  000001  007540  16$:    MOV     #1,TERM         ;SET 'TERMINATOR HAS BEEN FOUND'' FLAG
1102  007504  012737  000021  001124          MOV     #XON,$GDDAT     ;LOAD EXPECTED
1103  007512  105737  007536                  TSTB    NOEXIT          ;TEST IF ALLOWED TO LEAVE ROUTINE ?
1104  007516  001257                          BNE     6$              ;BR IF NOT ALLOWED
1105  007520  012602                  17$:    MOV     (SP)+,R2
1106  007522  012601                          MOV     (SP)+,R1
1107  007524  012600                          MOV     (SP)+,R0
1108  007526  000207                          RTS     PC              ;EXIT
1109  007530  000000                  30$:    0
1110  007532  000000                  LOOP:   0
1111  007534  000000                  ANESC:  0
1112  007536  000000                  NOEXIT: 0
1113  007540  000000                  TERM:   0
```

```
                                        ;DETERMINE WHAT THE OPR. WANTS TO DO NOW
007542  012706  001100          FINDOT: MOV     #STACK,SP
007546  004537  010164                  JSR     R5,AMSG
007552  010503                  FINDTA: MQ1
007554  004737  010226                  JSR     PC,GETCHR
007560  000770                          BR      FINDOT

007562  105777  171510          1$:     TSTB    @VTOS
007566  100375                          BPL     1$
007570  110077  171504                  MOVB    R0,@VTOB
007574  042700  177640                  BIC     #177640,R0      ;MASK
007600  122700  000007                  CMPB    #7,R0           ;TEST FOR CTRL G
007604  001432                          BEQ     2$
007606  122700  000003                  CMPB    #3,R0           ;TEST FOR CTRL C
007612  001431                          BEQ     3$
007614  122700  000101                  CMPB    #'A,R0          ;TEST FOR NUMBER
007620  101350                          BHI     FINDOT
007622  122700  000124                  CMPB    #'T,R0          ;TEST FOR OTHERS
007626  103745                          BLO     FINDOT
007630  042700  177740                  BIC     #177740,R0      ;MAKE 0-37
007634  005300                          DEC     R0
007636  110037  001102                  MOVB    R0,$TSTNM       ;LOAD THAT TEST #
007642  006300                          ASL     R0
007644  005760  007702                  TST     DSPCH(R0)       ;TEST IF VALID
007650  001734                          BEQ     FINDOT                  ;BR IF NOT
007652  016037  007702  001106          MOV     DSPCH(R0),$LPADR        ;LOAD RETURN ADDRESS
007660  016037  007702  001110          MOV     DSPCH(R0),$LPERR        ;LOAD ERROR LOOP ADDRESS
007666  000170  007702                  JMP     @DSPCH(R0)              ;GO TO THAT TEST
007672  104406                  2$:     GTSWR                           ;READ SWITCHES
007674  000722                          BR      FINDOT
007676  000000                  3$:     HALT                            ;LSI-11 ODT ENTRY
007700  000720                          BR      FINDOT

007702  002124          DSPCH:  TST1+2
007704  002320                  TST2+2
007706  002506                  TST3+2
007710  002702                  TST4+2
007712  003076                  TST5+2
007714  003142                  TST6+2
007716  003316                  TST7+2
007720  003472                  TST10+2
007722  003536                  TST11+2
007724  003616                  TST12+2
007726  003676                  TST13+2
007730  004010                  TST14+2
007732  004122                  TST15+2
007734  004264                  TST16+2
007736  004606                  TST17+2
007740  004644                  TST20+2
007742  004702                  TST21+2
007744  004736                  TST22+2
007746  004772                  TST23+2
007750  000000                  0
007752  000000                  0
```

```
 1169  007754  000000                          0
 1170
 1171
 1172                                  ;PROGRAM DELAY ROUTINE
 1173
 1174  007756  013737  001266  010060  DELAY:  MOV     SUBTST,10$      ;LOAD COUNT
 1175  007764  005037  010062                  CLR     11$
 1176  007770  005737  001270                  TST     WFTEST          ;TEST IF W.F. MODE
 1177  007774  001410                          BEQ     2$              ;BR IF NOT
 1178  007776  006237  010060                  ASR     10$             ;CHANGE DELAY TIMER
 1179  010002  006237  010060                  ASR     10$
 1180  010006  001003                          BNE     2$              ;BR IF SOME BITS ARE SET
 1181  010010  012737  000001  010060          MOV     #1,10$          ;ENSURE SOME DELAY
 1182  010016  032777  010000  171114  2$:     BIT     #BIT12,@SWR     ;TEST SR
 1183  010024  001014                          BNE     3$              ;BR IF SET
 1184  010026  005737  001200                  TST     $PASS           ;FIRST PASS ?
 1185  010032  001411                          BEQ     3$              ;BR IF YES
 1186  010034  105777  171232                  TSTB    @VTIS           ;CHECK FOR INPUT FLAG
 1187  010040  100406                          BMI     3$              ;BR IF FLAG
 1188  010042  005337  010062                  DEC     11$             ;DELAY
 1189  010046  001363                          BNE     2$              ;BR IF NOT DONE
 1190  010050  005337  010060                  DEC     10$
 1191  010054  100360                          BPL     2$              ;DELAY
 1192  010056  000207                  3$:     RTS     PC              ;EXIT
 1193
 1194  010060  000002                  10$:    2
 1195  010062  000000                  11$:    0
```

```
 1197                                   ;TESTER DEVICE BUS ADDRESSES
 1198  010064  000000                   VTIS0:  0                       ;LINE 0 ADDRESSES
 1199  010066  000000                   VTIB0:  0
 1200  010070  000000                   VTOS0:  0
 1201  010072  000000                   VTOB0:  0
 1202
 1203  010074  000000                   VTIS1:  0
 1204  010076  000000                   VTIB1:  0
 1205  010100  000000                   VTOS1:  0
 1206  010102  000000                   VTOB1:  0
 1207
 1208  010104  000000                   VTIS2:  0
 1209  010106  000000                   VTIB2:  0
 1210  010110  000000                   VTOS2:  0
 1211  010112  000000                   VTOB2:  0
 1212
 1213  010114  000000                   VTIS3:  0
 1214  010116  000000                   VTIB3:  0
 1215  010120  000000                   VTOS3:  0
 1216  010122  000000                   VTOB3:  0
 1217
 1218  010124  000000                   VTIS4:  0
 1219  010126  000000                   VTIB4:  0
 1220  010130  000000                   VTOS4:  0
 1221  010132  000000                   VTOB4:  0
 1222
 1223  010134  000000                   VTIS5:  0
 1224  010136  000000                   VTIB5:  0
 1225  010140  000000                   VTOS5:  0
 1226  010142  000000                   VTOB5:  0
 1227
 1228  010144  000000                   VTIS6:  0
 1229  010146  000000                   VTIB6:  0
 1230  010150  000000                   VTOS6:  0
 1231  010152  000000                   VTOB6:  0
 1232
 1233  010154  000000                   VTIS7:  0
 1234  010156  000000                   VTIB7:  0
 1235  010160  000000                   VTOS7:  0
 1236  010162  000000                   VTOB7:  0
```

```
 1238
 1239                                   ;HEADER SUBROUTINE FOR TERMINAL UNDER TEST
 1240
 1241  010164  012537  010216          AMSG:   MOV     (R5)+,10$       ;GET POINTER
 1242  010170  012700  014540                  MOV     #BUFFER,R0      ;LOAD OUTPUT POINTER
 1243  010174  113737  001304  011752          MOVB    ASPTRB,ZAPITA   ;GET ASPECT RATIO
 1244  010202  004537  010302                  JSR     R5,MTOB         ;MOVE TO OUTPUT BUFFER
 1245  010206  011743                          ZAPIT                   ;CLEAR SCREEN AND SET ASPECT RATIO
 1246  010210  105740                          TSTB    -(R0)           ;BACK OVER TERMINATOR
 1247  010212  004537  010302                  JSR     R5,MTOB         ;MOVE TO BUFFER
 1248  010216  000000                  10$:    0
 1249  010220  004737  006632                  JSR     PC,XPRNT        ;DISPLAY IT
 1250  010224  000205                          RTS     R5              ;EXIT
 1251
 1252
 1253                                   ;SUBROUTINE TO GET A CHARACTER FROM THE TERMINAL
 1254
 1255  010226  013737  001264  010276  GETCHR: MOV     TIME0,TIME1     ;LOAD TIME COUNTER
 1256  010234  005037  010300                  CLR     TIME2
 1257
 1258  010240  105777  171026          1$:     TSTB    @VTIS           ;TEST INPUT STATUS
 1259  010244  100005                          BPL     2$              ;BR IF CLEARED
 1260  010246  017700  171022                  MOV     @VTIB,R0        ;READ A CHAR
 1261  010252  062716  000002                  ADD     #2,(SP)         ;UPDATE RETURN
 1262  010256  000207                          RTS     PC              ;EXIT
 1263
 1264  010260  005337  010300          2$:     DEC     TIME2           ;DELAY
 1265  010264  001365                          BNE     1$
 1266  010266  005337  010276                  DEC     TIME1           ;FINISHED ?
 1267  010272  100362                          BPL     1$              ;LOOP TILL TIME EXPIRED
 1268  010274  000207                          RTS     PC              ;EXIT
 1269
 1270  010276  000000                  TIME1:  0
 1271  010300  000000                  TIME2:  0
 1272
 1273                                   ;MOVE TO THE OUTPUT BUFFER
 1274
 1275  010302  012501                  MTOB:   MOV     (R5)+,R1        ;LOAD DEST.
 1276  010304  112120                  1$:     MOVB    (R1)+,(R0)+     ;LOAD BYTE
 1277  010306  001376                          BNE     1$              ;BR UNTIL DONE
 1278  010310  000205                          RTS     R5              ;EXIT
 1279
 1280                                           .SBTTL  ASCII MESSAGES
```

```
1282                                                    ;ASCII MESSAGES
1283  010312  031033  005015  050042  PWRMSG: .ASCIZ  <33><62><15><12>/'POWER FAIL OCCURRED''/<15><12>
      010320  053517  051105  043040
      010326  044501  020114  041517
      010334  052503  051122  042105
      010342  006442  000012
1284  010346     033     062          TITLE:  .BYTE   33,62                  ;ENSURE EXIT FROM GRAPH MODE
1285  010350     033     074                  .BYTE   33,'<                  ;ENTER ANSCII
1286  010352     033     133     077          .BYTE   33,133,77,63,154       ;ENSURE 80 COLUMN
      010355     063     154
1287  010357     033     133     077          .BYTE   33,133,77,62,154       ;ENSURE ASCII
      010362     062     154
1288  010364  006415  041412  053132          .ASCIZ  <15><15><12>/CZVTNA     VT105 ACCEPTANCE TEST/<15><12>
      010372  047124  004501  052126
      010400  030061  020065  041501
      010406  042503  052120  047101
      010414  042503  052040  051505
      010422  006524  000012
1289  010426  047514  050117  047440  MQ0:    .ASCIZ  /LOOP ON TEST PATTERN LETTER (A THRU S) ? =  /
      010434  020116  042524  052123
      010442  050040  052101  042524
      010450  047122  046040  052105
      010456  042524  020122  040450
      010464  052040  051110  020125
      010472  024523  037440  036440
      010500  020040     000
1290  010503     123  040524  052122  MQ1:    .ASCIZ  /START AT TEST PATTERN LETTER (A THRU S) ? =  /
      010510  040440  020124  042524
      010516  052123  050040  052101
      010524  042524  047122  046040
      010532  052105  042524  020122
      010540  040450  052040  051110
      010546  020125  024523  037440
      010554  036440  020040     000
1291  010561     104  053105  041511  WHAT0:  .ASCIZ  /DEVICE STARTING ADDRESS ?   /
      010566  020105  052123  051101
      010574  044524  043516  040440
      010602  042104  042522  051523
      010610  037440  036440  000040
1292  010616     015     012     012  WHAT8:  .BYTE   15,12,12
1293  010621     106  051111  052123          .ASCIZ  /FIRST /
      010626  000040
1294  010630     015     012     012  WHAT9:  .BYTE   15,12,12
1295  010633     114  051501  020124          .ASCIZ  /LAST /
      010640     000
1296  010641     111  053116  046101  EM3:    .ASCIZ  /INVALID BUS ADDRESS, TRY AGAIN/
      010646  042111  041040  051525
      010654  040440  042104  042522
      010662  051523  020054  051124
      010670  020131  043501  044501
      010676  000116
1297  010700  051107  053517  047111  DAHL:   .ASCIZ  /GROWING HORIZ. LINE/
      010706  020107  047510  044522
      010714  027132  046040  047111
```

```
                010722  000105
1298            010724  051107  053517  047111  DAVL:   .ASCIZ  /GROWING VERT. LINE/
                010732  020107  042526  052122
                010740  020056  044514  042516
                010746     000
1299            010747     104  051511  046120  SHL0:   .ASCIZ  /DISPLAY STEPPING HORIZ. LINE ON GRAPH 0/
                010754  054501  051440  042524
                010762  050120  047111  020107
                010770  047510  044522  027132
                010776  046040  047111  020105
                011004  047117  043440  040522
                011012  044120  030040     000
1300            011017     104  051511  046120  SHL1:   .ASCIZ  /DISPLAY STEPPING HORIZ. LINE ON GRAPH 1/
                011024  054501  051440  042524
                011032  050120  047111  020107
                011040  047510  044522  027132
                011046  046040  047111  020105
                011054  047117  043440  040522
                011062  044120  030440     000
1301            011067     104  051511  046120  GROA1:  .ASCIZ  /DISPLAY DATA ON GRAPH 0 AND 1/
                011074  054501  042040  052101
                011102  020101  047117  043440
                011110  040522  044120  030040
                011116  040440  042116  030440
                011124     000
1302            011125     123  042524  050120  SHGL0:  .ASCIZ  /STEPPING HISTOGRAM LINE ON GRAPH 0/
                011132  047111  020107  044510
                011140  052123  043517  040522
                011146  020115  044514  042516
                011154  047440  020116  051107
                011162  050101  020110  000060
1303            011170  052123  050105  044520  SHGL1:  .ASCIZ  /STEPPING HISTOGRAM LINE ON GRAPH 1/
                011176  043516  044040  051511
                011204  047524  051107  046501
                011212  046040  047111  020105
                011220  047117  043440  040522
                011226  044120  030440     000
1304            011233     110  051511  047524  HG0A1:  .ASCIZ  /HISTOGRAM ON GRAPH 0 AND 1/
                011240  051107  046501  047440
                011246  020116  051107  050101
                011254  020110  020060  047101
                011262  020104  000061
1305            011266  052503  051522  051117  CURGR0: .ASCIZ  /CURSORS ON GRAPH 0/
                011274  020123  047117  043440
                011302  040522  044120  030040
                011310     000
1306            011311     103  051125  047523  CURGR1: .ASCIZ  /CURSORS ON GRAPH 1/
                011316  051522  047440  020116
                011324  051107  050101  020110
                011332  000061
1307            011334  052123  051101  044524  SCORD0: .ASCIZ  /STARTING COORDINATE TEST ON GRAPH 0/
                011342  043516  041440  047517
                011350  042122  047111  052101
                011356  020105  042524  052123
```

```
                011364  047440  020116  051107
                011372  050101  020110  000060
1308            011400  052123  051101  044524  SCORD1: .ASCIZ  /STARTING COORDINATE TEST ON GRAPH 1/
                011406  043516  041440  047517
                011414  042122  047111  052101
                011422  020105  042524  052123
                011430  047440  020116  051107
                011436  050101  020110  000061
1309            011444  040502  042523  046040  BSLGR0: .ASCIZ  /BASE LINE ON HISTOGRAM 0/
                011452  047111  020105  047117
                011460  044040  051511  047524
                011466  051107  046501  030040
                011474     000
1310            011475     102  051501  020105  BSLGR1: .ASCIZ  /BASE LINE ON HISTOGRAM 1/
                011502  044514  042516  047440
                011510  020116  044510  052123
                011516  043517  040522  020115
                011524  000061
1311            011526  052123  044522  041520  STCGR0: .ASCIZ  /STRIPCHART ON GRAPH 0/
                011534  040510  052122  047440
                011542  020116  051107  050101
                011550  020110  000060
1312            011554  052123  044522  041520  STCGR1: .ASCIZ  /STRIPCHART ON GRAPH 1/
                011562  040510  052122  047440
                011570  020116  051107  050101
                011576  020110  000061
1313            011602  052504  046101  051440  DUSTC:  .ASCIZ  /DUAL STRIPCHART MODE/
                011610  051124  050111  044103
                011616  051101  020124  047515
                011624  042504     000
1314            011627     107  040522  044120  GAITE:  .ASCIZ  /GRAPH ASPECT RATIO AND INTERACTIVE ENABLE TEST/
                011634  040440  050123  041505
                011642  020124  040522  044524
                011650  020117  047101  020104
                011656  047111  042524  040522
                011664  052103  053111  020105
                011672  047105  041101  042514
                011700  052040  051505  000124
1315            011706  051501  042520  052103  CARAT:  .ASCIZ  /ASPECT RATIO WITH CHARACTERS/
                011714  051040  052101  047511
                011722  053440  052111  020110
                011730  044103  051101  041501
                011736  042524  051522     000
1316            011743     033     061          ZAPIT:  .BYTE   ESC,GRON         ;ENTER GRAPHIC
1317            011745     101     040     040          .BYTE   LDE0,40,40      ;DISABLE THE SCREEN
1318            011750     111     060                  .BYTE   LDE1,60         ;CLEAR GRAPH DATA
1319            011752     040                  ZAPITA: .BYTE   40              ;LOAD ASPECT RATIO
1320            011753     033     062                  .BYTE   ESC,GROF        ;ENTER TEXT MODE
1321            011755     033     133     077          .BYTE   ESC,'[,'?,'2,154        ;ENSURE ASCII MODE
                011760     062     154
1322            011762     033     110                  .BYTE   ESC,'H          ;HOME THE SCREEN
1323            011764     033     112                  .BYTE   ESC,'J          ;ERASE THE SCREEN
1324            011766     000                  ZAPITB: .BYTE   0               ;TERM.
1325                    011770                          .EVEN
```

```
1326
```

```
1328                                    .SBTTL  TTY INPUT ROUTINE
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    .ENABL  LSB
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 (1)                                    ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 (1)                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
 (1)                                    ;*WHEN OPERATING IN TTY FLAG MODE.
 (1)  011770  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
 (1)  011776  001074                         BNE     15$             ;;BRANCH IF NO
 (1)  012000  105777  167140                 TSTB    @$TKS           ;;CHAR THERE?
 (1)  012004  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
 (1)  012006  117746  167134                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
 (1)  012012  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
 (1)  012016  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
 (1)  012022  001062                         BNE     15$             ;;NO, RETURN TO USER
 (1)  012024  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 (1)  012032  001456                         BEQ     15$             ;;BRANCH IF YES
 (1)
 (1)  012034  104401  012515                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
 (1)  012040  104401  012522          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
 (2)  012044  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
 (2)  012050  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  012052  104401  012533                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
 (1)  012056  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
 (1)  012060  005046                          CLR     -(SP)           ;;THE NEW SWR
 (1)  012062  105777  167056          7$:     TSTB    @$TKS           ;;CHAR THERE?
 (1)  012066  100375                         BPL     7$              ;;IF NOT TRY AGAIN
 (1)
 (1)  012070  117746  167052                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
 (1)  012074  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 (1)
 (1)
 (1)
 (1)  012100  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
 (1)  012104  001005                          BNE     10$             ;;BRANCH IF NOT
 (1)  012106  104401  012510                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
 (1)  012112  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 (1)  012116  000757                          BR      19$             ;;LET'S TRY IT AGAIN
 (1)
 (1)
 (1)  012120  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
 (1)  012124  001022                          BNE     16$             ;;BRANCH IF NO
 (1)  012126  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
 (1)  012132  001403                          BEQ     11$             ;;BRANCH IF YES
 (1)  012134  016677  000002  166776         MOV     2(SP),@SWR      ;;SAVE NEW SWR
 (1)  012142  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
 (1)  012146  104401  001167          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
 (1)  012152  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
 (1)  012160  001003                          BNE     15$             ;;BRANCH IF NOT
 (1)  012162  012777  000100  166754         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
 (1)  012170  000002                  15$:    RTI                     ;;RETURN
```

```
(1)  012172  004737  014006          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  012176  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
(1)  012202  002420                          BLT     18$             ;;BRANCH IF YES
(1)  012204  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
(1)  012210  003015                          BGT     18$             ;;BRANCH IF YES
(1)  012212  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  012216  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  012222  001403                          BEQ     17$             ;;BRANCH IF YES
(1)  012224  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  012226  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
(1)  012230  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
(1)  012232  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  012236  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  012242  000707                          BR      7$              ;;GET THE NEXT ONE
(1)  012244  104401  001166          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  012250  000720                          BR      20$             ;;SIMULATE CONTROL-U
(1)                                  .DSABL  LSB
(1)
(1)
(2)                                  ;;**************************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
(1)                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  012252  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  012254  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  012262  105777  166656          1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  012266  100375                          BPL     1$              ;;A CHARACTER
(1)  012270  117766  166652  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  012276  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  012304  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  012312  001013                          BNE     3$              ;;BRANCH IF NO
(1)  012314  105777  166624          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  012320  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  012322  117746  166620                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  012326  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  012332  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  012336  001366                          BNE     2$              ;;IF NOT DISCARD IT
(1)  012340  000750                          BR      1$              ;;YES, RESUME
(1)  012342  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  012350  002407                          BLT     4$              ;;BRANCH IF YES
(1)  012352  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  012360  003003                          BGT     4$              ;;BRANCH IF YES
(1)  012362  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  012370  000002                  4$:     RTI                     ;;GO BACK TO USER
(2)                                  ;;**************************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
(1)                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
```

```
(1)
(1)  012372  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  012374  012703  012500           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  012400  022703  012510           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  012404  101405                           BLOS    4$              ;;BR IF YES
(1)  012406  104410                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  012410  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  012412  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  012416  001003                           BNE     3$              ;;SKIP IF NOT
(1)  012420  104401  001166           4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  012424  000763                           BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  012426  111337  012476           3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  012432  104401  012476                   TYPE    ,9$
(1)  012436  122723  000015                   CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  012442  001356                           BNE     2$              ;;LOOP IF NOT RETURN
(1)  012444  105063  177777                   CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  012450  104401  001170                   TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  012454  012603                           MOV     (SP)+,R3        ;;RESTORE R3
(1)  012456  011646                           MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  012460  016666  000004  000002           MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  012466  012766  012500  000004           MOV     #$TTYIN,4(SP)
(1)  012474  000002                           RTI                     ;;RETURN
(1)  012476     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  012477     000                          .BYTE   0               ;;TERMINATOR
(1)  012500  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  012510  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
(1)  012515     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
(1)  012522  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  012530  020075     000
(1)  012533     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
(1)  012540  036440  000040
1329                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(2)                                   ;;******************************************************************
(1)                                   ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                   ;*CHANGE IT TO BINARY.
(1)                                   ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
(1)                                   ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
(1)                                   ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
(1)                                   ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
(1)                                   ;*CALL:
(1)                                   ;*      RDOCT                   ;;READ AN OCTAL NUMBER
(1)                                   ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                   ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
(1)
(1)  012544  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  012546  016666  000004  000002           MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  012554  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  012556  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  012560  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  012562  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  012564  012600                           MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  012566  010037  012672                   MOV     R0,5$           ;;AND SAVE IT
(1)  012572  005001                           CLR     R1              ;;CLEAR DATA WORD
```

```
   (1)  012574  005002                        CLR     R2
   (1)  012576  112046                2$:      MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
   (1)  012600  001420                        BEQ     3$              ;;IF ZERO GET OUT
   (1)  012602  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
   (1)  012606  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
   (1)  012610  122716  000067                CMPB    #'7,(SP)
   (1)  012614  002423                        BLT     4$
   (1)  012616  006301                        ASL     R1              ;;*2
   (1)  012620  006102                        ROL     R2
   (1)  012622  006301                        ASL     R1              ;;*4
   (1)  012624  006102                        ROL     R2
   (1)  012626  006301                        ASL     R1              ;;*8
   (1)  012630  006102                        ROL     R2
   (1)  012632  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
   (1)  012636  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
   (1)  012640  000756                        BR      2$              ;;LOOP
   (1)  012642  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
   (1)  012644  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
   (1)  012650  010237  012702                MOV     R2,$HIOCT
   (3)  012654  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
   (3)  012656  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  012660  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  012662  000002                        RTI                     ;;RETURN
   (1)  012664  005726                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
   (1)  012666  105010                        CLRB    (R0)            ;;SET A TERMINATOR
   (1)  012670  104401                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
   (1)  012672  000000                5$:     .WORD   0
   (1)  012674  104401  001166                TYPE    ,$QUES          ;;''?'' 'CR' & 'LF''
   (1)  012700  000730                        BR      1$              ;;TRY AGAIN
   (1)  012702  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
1330                                  .SBTTL  APT COMMUNICATIONS ROUTINE
   (1)
   (2)                                ;;**************************************************************************
   (1)  012704  112737  000001  013150 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
   (1)  012712  112737  000001  013146 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
   (1)  012720  000403                        BR      $ATYC
   (1)  012722  112737  000001  013150 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
   (2)  012730                        $ATYC:
   (3)  012730  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  012732  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (1)  012734  105737  013146                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
   (1)  012740  001450                        BEQ     5$              ;;IF NOT:  BR
   (1)  012742  122737  000001  001212        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
   (1)  012750  001031                        BNE     3$              ;;IF NOT:  BR
   (1)  012752  132737  000100  001213        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
   (1)  012760  001425                        BEQ     3$              ;;IF NOT:  BR
   (1)  012762  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
   (1)  012766  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
   (1)  012774  005737  001172        1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
   (1)  013000  001375                        BNE     1$              ;;IF NOT:  WAIT
   (1)  013002  010037  001206                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
   (1)  013006  105720                2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
   (1)  013010  001376                        BNE     2$
   (1)  013012  163700  001206                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
```

```
(1)  013016  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(1)  013020  010037  001210                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(1)  013024  012737  000004  001172          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(1)  013032  000413                          BR      5$
(1)  013034  017637  000004  013060  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(1)  013042  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  013050  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  013054  004737  013574                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  013060  000000                  4$:     .WORD   0
(1)  013062                          5$:
(1)  013062  105737  013150          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  013066  001416                          BEQ     12$             ;;IF NOT:  BR
(1)  013070  005737  001212                  TST     $ENV            ;;RUNNING UNDER APT?
(1)  013074  001413                          BEQ     12$             ;;IF NOT:  BR
(1)  013076  005737  001172          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  013102  001375                          BNE     11$             ;;IF NOT:  WAIT
(1)  013104  017637  000004  001174          MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  013112  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  013120  005237  001172                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  013124  105037  013150          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  013130  105037  013147                  CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  013134  105037  013146                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  013140  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013142  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013144  000207                          RTS     PC              ;;RETURN
(1)  013146     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
(1)  013147     000                  $LFLG:  .BYTE   0               ;;LOG FLAG
(1)  013150     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
(1)          013152                          .EVEN
(1)          000200                  APTSIZE=200
(1)          000001                  APTENV=001
(1)          000100                  APTSPOOL=100
(1)          000040                  APTCSUP=040
1331                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 (1)
 (2)                                 ;;******************************************************************
 (1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 (1)                                 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 (1)                                 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 (1)                                 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 (1)                                 ;*REPLACED WITH SPACES.
 (1)                                 ;*CALL:
 (1)                                 ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
 (1)                                 ;*      TYPDS                   ;;GO TO THE ROUTINE
 (1)
 (2)  013152                         $TYPDS:
 (3)  013152  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 (3)  013154  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (3)  013156  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 (3)  013160  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 (3)  013162  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 (1)  013164  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 (1)  013170  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 (1)  013174  100004                         BPL     1$              ;;BR IF INPUT IS POS.
```

```
(1)  013176  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  013200  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  013206  005000                    1$:   CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  013210  012703  013366                    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  013214  112723  000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  013220  005002                    2$:   CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  013222  016001  013356                    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  013226  160105                    3$:   SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  013230  002402                          BLT     4$              ;;BR IF DONE
(1)  013232  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  013234  000774                          BR      3$
(1)  013236  060105                    4$:   ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  013240  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  013242  001002                          BNE     5$              ;;FALL THROUGH IF 0
(1)  013244  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  013246  100407                          BMI     7$              ;;BR IF YES
(1)  013250  106316                    5$:   ASLB    (SP)            ;;MSD?
(1)  013252  103003                          BCC     6$              ;;BR IF NO
(1)  013254  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  013262  052702  000060            6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  013266  052702  000040            7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  013272  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  013274  005720                          TST     (R0)+           ;;JUST INCREMENTING
(1)  013276  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  013302  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  013304  003002                          BGT     8$              ;;GO TO EXIT
(1)  013306  010502                          MOV     R5,R2           ;;GET THE LSD
(1)  013310  000764                          BR      6$              ;;GO CHANGE TO ASCII
(1)  013312  105726                    8$:   TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  013314  100003                          BPL     9$              ;;BR IF NO
(1)  013316  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  013324  105013                    9$:   CLRB    (R3)            ;;SET THE TERMINATOR
(3)  013326  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  013330  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  013332  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  013334  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013336  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013340  104401  013366                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  013344  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  013352  012616                          MOV     (SP)+,(SP)
(1)  013354  000002                          RTI                     ;;RETURN TO USER
(1)  013356  023420                    $DTBL: 10000.
(1)  013360  001750                          1000.
(1)  013362  000144                          100.
(1)  013364  000012                          10.
(1)  013366  000004                    $DBLK: .BLKW   4
```

```
 1333
 1334  013376  032777  004000  165534  MSCOPE: BIT     #SW11,@SWR      ;TEST IF FORCED MODE SW 11 IS SET
 1335  013404  001411                          BEQ     1$              ;BR IF NOT
 1336  013406  012737  000040  001304          MOV     #40,ASPTRB      ;PRIME THE RATIO
 1337  013414  032777  002000  165516          BIT     #SW10,@SWR      ;TEST IF FORCE VT55
 1338  013422  001402                          BEQ     1$              ;BR IF YES
 1339  013424  105237  001304                  INCB    ASPTRB          ;FORCE 105 RATIO
 1340  013430  105777  165504          1$:     TSTB    @SWR            ;TEST BIT 7
 1341  013434  100003                          BPL     $SCOPE          ;NOT SET
 1342  013436  005737  007532                  TST     LOOP            ;TEST LOOP
 1343  013442  001046                          BNE     $OVER           ;SET LOOP ON TEST
 1344                                  .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (2)                                 ;;*******************************************************************
  (1)                                 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                                 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
  (1)                                 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
  (1)                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                 ;*SW14=1        LOOP ON TEST
  (1)                                 ;*SW08=1        LOOP ON TEST IN SWR<7:0>
  (1)                                 ;*CALL
  (1)                                 ;*      SCOPE           ;;SCOPE=IOT
  (1)
  (1)  013444                         $SCOPE:
  (1)  013444  104407                         CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
  (2)  013446  104407                         CKSWR
  (1)  013450  032777  040000  165462  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
  (1)  013456  001040                          BNE     $OVER           ;;YES IF SW14=1
  (1)                                 ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  013460  000416                 $XTSTR: BR      6$              ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
  (1)                                                                 ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
  (1)  013462  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  013466  012737  013506  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
  (1)  013474  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
  (1)  013500  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  013504  000414                          BR      $SVLAD          ;;GO TO THE NEXT TEST
  (1)  013506  022626                 5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  013510  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  013514  000421                          BR      $OVER           ;;LOOP ON THE PRESENT TEST
  (1)  013516                         6$:;#####END OF CODE FOR THE XOR TESTER#####
  (1)  013516  032777  000400  165414          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
  (1)  013524  001404                          BEQ     $SVLAD          ;;BR IF NO
  (1)  013526  127737  165406  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
  (1)  013534  001411                          BEQ     $OVER           ;;BR IF YES
  (1)  013536  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
  (1)  013542  113737  001102  001176          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
  (1)  013550  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
  (1)  013554  105037  001103                  CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
  (1)  013560  013777  001102  165354  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
  (1)  013566  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
  (1)  013572  000002                          RTI                     ;;FIXES PS
```

```
 1346
 1347                                   .SBTTL  TYPE ROUTINE
  (1)
  (2)                                   ;;*********************************************************************
  (1)                                   ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
  (1)                                   ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
  (1)                                   ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
  (1)                                   ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
  (1)                                   ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
  (1)                                   ;*
  (1)                                   ;*CALL:
  (1)                                   ;*1) USING A TRAP INSTRUCTION
  (1)                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
  (1)                                   ;*OR
  (1)                                   ;*      TYPE
  (1)                                   ;*      MESADR
  (1)                                   ;*
  (1)
  (1)  013574  105737  001157           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
  (1)  013600  100002                           BPL     1$              ;;BR IF YES
  (1)  013602  000000                           HALT                    ;;HALT HERE IF NO TERMINAL
  (1)  013604  000430                           BR      3$              ;;LEAVE
  (1)  013606  010046                   1$:     MOV     R0,-(SP)        ;;SAVE R0
  (1)  013610  017600  000002                   MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
  (1)  013614  122737  000001  001212           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
  (1)  013622  001011                           BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
  (1)  013624  132737  000100  001213           BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
  (1)  013632  001405                           BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
  (1)  013634  010037  013644                   MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
  (1)  013640  004737  012712                   JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
  (1)  013644  000000                   61$:    .WORD   0               ;;MESSAGE ADDRESS
  (1)  013646  132737  000040  001213   62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
  (1)  013654  001003                           BNE     60$             ;;YES,SKIP TYPE OUT
  (1)  013656  112046                   2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
  (1)  013660  001005                           BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
  (1)  013662  005726                           TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
  (1)  013664  012600                   60$:    MOV     (SP)+,R0        ;;RESTORE R0
  (1)  013666  062716  000002           3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
  (1)  013672  000002                           RTI                     ;;RETURN
  (1)  013674  122716  000011           4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
  (1)  013700  001430                           BEQ     8$
  (1)  013702  122716  000200                   CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
  (1)  013706  001006                           BNE     5$
  (1)  013710  005726                           TST     (SP)+           ;;POP  <CR><LF> EQUIV
  (1)  013712  104401                           TYPE                    ;;TYPE A CR AND LF
  (1)  013714  001167                           $CRLF
  (1)  013716  105037  014052                   CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
  (1)  013722  000755                           BR      2$              ;;GET NEXT CHARACTER
  (1)  013724  004737  014006           5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
  (1)  013730  123726  001156           6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
  (1)  013734  001350                           BNE     2$              ;;IF NO GO GET NEXT CHAR.
  (1)  013736  013746  001154                   MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
  (1)                                           ;;AND THE NULL CHAR.
  (1)  013742  105366  000001           7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
```

```
(1)  013746  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  013750  004737  014006                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  013754  105337  014052                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  013760  000770                        BR      7$              ;;LOOP
(1)
(1)                                 ;HORIZONTAL TAB PROCESSOR
(1)
(1)  013762  112716  000040          8$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  013766  004737  014006          9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  013772  132737  000007  014052        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  014000  001372                        BNE     9$              ;;TAB STOP
(1)  014002  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(1)  014004  000724                        BR      2$              ;;GET NEXT CHARACTER
(1)  014006  105777  165136          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  014012  100375                        BPL     $TYPEC
(1)  014014  116677  000002  165130        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  014022  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  014030  001003                        BNE     1$              ;;BRANCH IF NO
(1)  014032  105037  014052                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  014036  000406                        BR      $TYPEX          ;;EXIT
(1)  014040  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  014046  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
(1)  014050  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  014052  000000          $CHARCNT:.WORD 0               ;;CHARACTER COUNT STORAGE
(1)  014054  000207          $TYPEX: RTS    PC
(1)
1348                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)                                 ;;**********************************************************************
(1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPOS                   ;;CALL FOR TYPEOUT
(1)                                 ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                 ;*      .BYTE   M               ;;M=1 OR 0
(1)                                 ;*                                      ;;1=TYPE LEADING ZEROS
(1)                                 ;*                                      ;;0=SUPPRESS LEADING ZEROS
(1)                                 ;*
(1)                                 ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                 ;*$TYPOS OR $TYPOC
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPON                   ;;CALL FOR TYPEOUT
(1)                                 ;*
(1)                                 ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  014056  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  014062  116637  000001  014301        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  014070  112637  014303                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
```

```
(1)  014074  062716  000002                    ADD     #2,(SP)          ;;ADJUST RETURN ADDRESS
(1)  014100  000406                            BR      $TYPON
(1)  014102  112737  000001  014301  $TYPOC:   MOVB    #1,$OFILL        ;;SET THE ZERO FILL SWITCH
(1)  014110  112737  000006  014303            MOVB    #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
(1)  014116  112737  000005  014300  $TYPON:   MOVB    #5,$OCNT         ;;SET THE ITERATION COUNT
(1)  014124  010346                            MOV     R3,-(SP)         ;;SAVE R3
(1)  014126  010446                            MOV     R4,-(SP)         ;;SAVE R4
(1)  014130  010546                            MOV     R5,-(SP)         ;;SAVE R5
(1)  014132  113704  014303                    MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  014136  005404                            NEG     R4
(1)  014140  062704  000006                    ADD     #6,R4            ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  014144  110437  014302                    MOVB    R4,$OMODE        ;;SAVE IT FOR USE
(1)  014150  113704  014301                    MOVB    $OFILL,R4        ;;GET THE ZERO FILL SWITCH
(1)  014154  016605  000012                    MOV     12(SP),R5        ;;PICKUP THE INPUT NUMBER
(1)  014160  005003                            CLR     R3               ;;CLEAR THE OUTPUT WORD
(1)  014162  006105                  1$:       ROL     R5               ;;ROTATE MSB INTO "C"
(1)  014164  000404                            BR      3$               ;;GO DO MSB
(1)  014166  006105                  2$:       ROL     R5               ;;FORM THIS DIGIT
(1)  014170  006105                            ROL     R5
(1)  014172  006105                            ROL     R5
(1)  014174  010503                            MOV     R5,R3
(1)  014176  006103                  3$:       ROL     R3               ;;GET LSB OF THIS DIGIT
(1)  014200  105337  014302                    DECB    $OMODE           ;;TYPE THIS DIGIT?
(1)  014204  100016                            BPL     7$               ;;BR IF NO
(1)  014206  042703  177770                    BIC     #177770,R3       ;;GET RID OF JUNK
(1)  014212  001002                            BNE     4$               ;;TEST FOR 0
(1)  014214  005704                            TST     R4               ;;SUPPRESS THIS 0?
(1)  014216  001403                            BEQ     5$               ;;BR IF YES
(1)  014220  005204                  4$:       INC     R4               ;;DON'T SUPPRESS ANY[illegible] J'S
(1)  014222  052703  000060                    BIS     #'0,R3           ;;MAKE THIS DIGIT ASC'[illegible]
(1)  014226  052703  000040            5$:     BIS     #' ,R3           ;;MAKE ASCII IF NOT ALRE[illegible]DY
(1)  014232  110337  014276                    MOVB    R3,8$            ;;SAVE FOR TYPING
(1)  014236  104401  014276                    TYPE    ,8$              ;;GO TYPE THIS DIGIT
(1)  014242  105337  014300            7$:     DECB    $OCNT            ;;COUNT BY 1
(1)  014246  003347                            BGT     2$               ;;BR IF MORE TO DO
(1)  014250  002402                            BLT     6$               ;;BR IF DONE
(1)  014252  005204                            INC     R4               ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  014254  000744                            BR      2$               ;;GO DO THE LAST DIGIT
(1)  014256  012605                  6$:       MOV     (SP)+,R5         ;;RESTORE R5
(1)  014260  012604                            MOV     (SP)+,R4         ;;RESTORE R4
(1)  014262  012603                            MOV     (SP)+,R3         ;;RESTORE R3
(1)  014264  016666  000002  000004            MOV     2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
(1)  014272  012616                            MOV     (SP)+,(SP)
(1)  014274  000002                            RTI                      ;;RETURN
(1)  014276     000                  8$:       .BYTE   0                ;;STORAGE [illegible]R ASCII DIGIT
(1)  014277     000                            .BYTE   0                ;;TERMINA[illegible] FOR TYPE ROUTINE
(1)  014300     000            $OCNT:  .BYTE   0                ;;OCTAL [illegible]IGIT COUNTER
(1)  014301     000            $OFILL: .BYTE   0                ;;ZERO FILL SWITCH
(1)  014302  000000            $OMODE: .WORD   0                ;;NUMBER OF DIGITS TO TYPE
```

```
 1350                                   .SBTTL  TRAP DECODER
  (1)
  (2)                                   ;;**********************************************************************
  (1)                                   ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
  (1)                                   ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
  (1)                                   ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
  (1)                                   ;*GO TO THAT ROUTINE.
  (1)
  (1)  014304  010046                   $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
  (1)  014306  016600  000002                   MOV     2(SP),R0        ;;GET TRAP ADDRESS
  (1)  014312  005740                           TST     -(R0)           ;;BACKUP BY 2
  (1)  014314  111000                           MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
  (1)  014316  006300                           ASL     R0              ;;POSITION FOR INDEXING
  (1)  014320  016000  014340                   MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
  (1)  014324  000200                           RTS     R0              ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                   ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
  (1)
  (1)  014326  011646                   $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
  (1)  014330  016666  000004  000002           MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
  (1)  014336  000002                           RTI                     ;;RESTORE THE PSW
  (1)
  (3)                                   .SBTTL  TRAP TABLE
  (3)
  (3)                                   ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
  (3)                                   ;*BY THE 'TRAP' INSTRUCTION.
  (3)
  (3)                                   ;       ROUTINE
  (3)                                   ;       -------
  (3)  014340  014326                   $TRPAD: .WORD   $TRAP2
  (3)  014342  013574                           $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)  014344  014102                           $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)  014346  014056                           $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)  014350  014116                           $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)  014352  013152                           $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  (1)
  (3)  014354  012040                           $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
  (1)
  (3)  014356  011770                           $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
  (3)  014360  012252                           $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
  (3)  014362  012372                           $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
  (3)  014364  012544                           $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
```

```
 1352                                   .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;POWER DOWN ROUTINE
  (1)  014366  012737  014532  000024   $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  014374  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (3)  014402  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  014404  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  014406  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  014410  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  014412  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)  014414  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (3)  014416  017746  164516                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
  (1)  014422  010637  014536                  MOV     SP,$SAVR6       ;;SAVE SP
  (1)  014426  012737  014440  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)  014434  000000                          HALT
  (1)  014436  000776                          BR      .-2             ;;HANG UP
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;POWER UP ROUTINE
  (1)  014440  012737  014532  000024   $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
  (1)  014446  013706  014536                  MOV     $SAVR6,SP       ;;GET SP
  (1)  014452  005037  014536                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
  (1)  014456  005237  014536          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
  (1)  014462  001375                          BNE     1$              ;;OF  WORD
  (3)  014464  012677  164450                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
  (3)  014470  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  014472  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
  (3)  014474  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  014476  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  014500  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  014502  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  014504  012737  014366  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  (1)  014512  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (1)  014520  104401                          TYPE                    ;;REPORT THE POWER FAILURE
  (1)  014522  010312                  $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
  (1)  014524  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RBEGIN
  (1)  014526  001354                  $PWRAD: .WORD   RBEGIN           ;;RESTART ADDRESS
  (1)  014530  000002                          RTI
  (1)  014532  000000                  $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
  (1)  014534  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
  (1)  014536  000000                  $SAVR6: 0                        ;;PUT THE SP HERE
 1353
 1354
 1355                                   ;START OF BUFFER SPACE
 1356
 1357  014540  000000                  BUFFER: 0
 1358          000001                          .END
```

```
ABASE = 177560              16#     29
ACDW1 = 176500              17#     29
ACDW2 = 000000              29
ACPUOP= 000000              29
ADDLIN= 002000              41#    187    191    226    230    444    466    621    625    639    643    899
ADDW0 = 000000              29
AD' J1 = 000000             29
AC  10= 000000              29
AD. .11= 000000             29
ADDW12= 000000              29
ADDW13= 000000              29
ADDW14= 000000              29
ADDW15= 000000              29
ADDW2 = 000000              29
ADDW3 = 000000              29
ADDW4 = 000000              29
ADDW5 = 000000              29
ADDW6 = 000000              29
ADDW7 = 000000              29
ADDW8 = 000000              29
ADDW9 = 000000              29
ADEVCT= 000000              29
ADEVM = 000000              29
AENV  = 000000              29
AENVM = 000000              29
AFATAL= 000000              29
AMADR1= 000000              29
AMADR2= 000000              29
AMADR3= 000000              29
AMADR4= 000000              29
AMAMS1= 000000              29
AMAMS2= 000000              29
AMAMS3= 000000              29
AMAMS4= 000000              29
AMSG    010164             175    214    252    293    333    350    383    415    433    455    476    508    557
                           593    662    677    693    706    719    760   1117   1241#
AMSGAD= 000000              29
AMSGLG= 000000              29
AMSGTY= 000000              29
AMTYP1= 000000              29
AMTYP2= 000000              29
AMTYP3= 000000              29
AMTYP4= 000000              29
ANESC   007534             968*   1001*  1003*  1033   1111#
APASS = 000000              29
APRIOR= 000000              29
AP.CSU= 000040             1330#   1347
APTENV= 000001             1330#   1347
APTSIZ= 000200              112    1330#
APTSPO= 000100             1330#   1347
ASPTRB  001304              86#    185    224    763*   1243   1336*  1339*
ASWREG= 000000              29
ATESTN= 000000              29
AUNIT = 000000              29
```

```
DDISP = 177570            13#    29    112
DELAY   007756           195    210    234    248    286    326    348    377    410    429    452    473    504
                         536    571    585    648    658    674    689    703    716    758   1174#
DISPLA  001142            29#   112*  1344*
DISPRE  000174            19#   112
DSPCH   007702          1138   1140   1141   1142   1148#
DSWR  = 177570            13#    29    112
DUSTC   011602           720   1313#
EMTVEC= 000030            13#
EM3     010641            89   1296#
ERRVEC= 000004            13#    96*    97*   112*  1344*
ESC   = 000033            43#   179    218    258    299    356    389    562    576    613    635    754    787
                         829    852    872    929   1316   1320   1321   1322   1323
FINDOT  007542          1097   1100   1116#  1120   1131   1133   1139   1144   1146
FINDTA  007552          1096*  1099*  1118#
FIRST   001254            67#   117*   142*   143    145    161
GAITE   011627           558   1314#
GETCHR  010226          1119   1255#
GIN     001416           100    109#
GINA    001424            98    106    110#
GNS   = ****** U          19   1350
GROF  = 000062            44#   755   1320
GRON  = 000061            45#   180    219    259    300    357    390    563    577    614    636    788    830
                         852    873    930   1316
GROA1   011067           334   1301#
GTSWR = 104406          1143   1350#
HGOA1   011233           416   1304#
HT    = 000011            13#   1347
IOTVEC= 000020            13#   112*   114*
LAST    001256            69#   154*   158*   763
LDBASE  006476           670    685    929#
LDC0  = 000103            55#   443    449
LDC1  = 000113            56#   465    470
LDE0  = 000101            49#   181    220    260    265    301    305    358    391    567    581    615    729
                         789    831    853    855    874    931   1317
LDE1  = 000111            50#   183    222    262    303    564    578    617    653    877   1318
LDG0  = 000102            52#   273    339    366    420    438    481    495    667    698    740
LDG1  = 000112            53#   313    344    399    425    460    513    527    682    711    746
LF    = 000012            13#   1347
LHV0  = 000104            58#   186    199    620
LHV1  = 000114            59#   225    238    638
LNO   = 000100            47#   198    237    267    272    307    312    360    365    393    398    637    728
                         739    745    798    834    839    854    856    858    882    887    935
LOOP    007532           113*  1095*  1098*  1110#  1342
LSC   = 000110            61#   268    308    361    394    794    835    857    883
MAXHOZ= 000354            37#   191    205    345    426    461    482    514    683    712    810    940
MAXVRT= 001000            39#   226    240    276    316    369    402    486    518    899
MQ0     010426          1096   1289#
MQ1     010503          1099   1118   1290#
MSCOPE  013376           114   1334#
MTOB    010302           725   1244   1247   1275#
NOEXIT  007536           970*  1077*  1083*  1103   1112#
PC      -%000007          13#   188*   194*   195*   202*   209*   210*   227*   233*   234*   241*   247*   248*
                         270*   275*   283*   286*   310*   315*   323*   326*   341*   347*   348*   363*   368*
```

```
                        374*   377*   396*   401*   407*   410*   422*   428*   429*   440*   446*   451*   452*
                        462*   468*   472*   473*   483*   499*   504*   515*   531*   536*   570*   571*   584*
                        585*   606*   622*   628*   640*   646*   648*   657*   658*   673*   674*   688*   689*
                        701*   702*   703*   714*   715*   716*   738*   744*   757*   758*   763*   796*   801*
                        837*   842*   885*   890*   927*   937*   943*   955*   961*  1108*  1119*  1192*  1249*
                       1262*  1268*  1328*  1330*  1347*  1352
PIRQ   = 177772          13#
PIRQVE= 000240           13#
PR0    = 000000          13#
PR1    = 000040          13#
PR2    = 000100          13#
PR3    = 000140          13#
PR4    = 000200          13#
PR5    = 000240          13#
PR6    = 000300          13#
PR7    - 000340          13#
PS     = 177776          13#
PSW    = 177776          13#
PWRMSG  010312         1283#  1352
PWRVEC= 000024           13#   112*  1352*
RBEGIN  001354           20     99#  1352
RDCHR  = 104410        1328   1350#
RDLIN  = 104411        1329   1350#
RDOCT  = 104412         139    150  1350#
RESVEC= 000010           13#
RSTRT   002054          163#   763
RSTRTA  002046          161#   763
R0     -%000000          13#   102*   104*   105    138*   140*   142    151*   153    154    155    163*   164*
                        165*   166*   167*   168*   169*   170*   178*   179*   180*   181*   182*   183*   184*
                        185*   186*   189*   193*   197*   198*   199*   203*   208*   217*   218*   219*   220*
                        221*   222*   223*   224*   225*   228*   232*   236*   237*   238*   242*   246*   257*
                        258*   259*   260*   261*   262*   263*   265*   266*   267*   268*   271*   272*   273*
                        277*   282*   298*   299*   300*   301*   302*   303*   304*   305*   306*   307*   308*
                        311*   312*   313*   317*   322*   355*   356*   357*   358*   359*   360*   361*   364*
                        365*   366*   370*   373*   388*   389*   390*   391*   392*   393*   394*   397*   398*
                        399*   403*   406*   487*   489*   490    519*   521*   522    561*   562*   563*   564*
                        565*   566*   567*   568*   569*   575*   576*   577*   578*   579*   580*   581*   582*
                        583*   596*   597*   598*   599*   600*   602*   605*   612*   613*   614*   615*   616*
                        617*   618*   619*   620*   623*   627*   634*   635*   636*   637*   638*   641*   645*
                        652*   653*   654*   655*   656*   724*   727    728*   729*   730*   731*   739*   740*
                        741*   745*   746*   747*   754*   755*   756*   763*   786*   787*   788*   789*   790*
                        791*   793*   794*   797*   798*   799*   802*   814*   826    828*   829*   830*   831*
                        832*   833*   834*   835*   838*   839*   840*   843*   844*   846*   871*   872*   873*
                        874*   875*   876*   877*   880*   881*   882*   883*   886*   887*   888*   891*   901*
                        929*   930*   931*   932*   933*   935*   938*   944*   947*   956*   960*   965    971*
                        997   1107*  1124   1125*  1126   1128   1130   1132   1134*  1135*  1136   1137*  1138
                       1140   1141   1142   1242*  1246   1269*  1276*  1329*  1330*  1331*  1347*  1350*  1352*
R1     =%000001          13#   103*   104    107    189    203    228    242    271    277    311    317    364
                        370    397    403    486*   492    500*   518*   524    532*   601*   603*   623    641
                        741    747    797    802    827    838    843    845*   878*   879*   881    886    891
                        914*   915*   916*   917*   918*   919*   920    926*   938    944    956    966    997*
                        999   1004   1007   1008   1009   1010   1011   1012   1013   1014   1071*  1072   1073
                       1106*  1275*  1276   1329*  1330*  1331*  1352*
R2     -%000002          13#   595*   607*   911*   912    914    967   1036*  1039*  1041*  1043*  1045*  1047*
```

```
                    1049*  1051*  1053*  1060   1071   1089   1105*  1329*  1331*  1352*
R3      =%000003      13#   485*   502*   517*   534*   733*   736    751*   952*   953    1328*  1331*  1348*
                    1352*
R4      -%000004      13#   734*   742    749*   951*   958*   1348*  1352*
R5      =%000005      13#   175*   214*   252*   293*   333*   338*   343*   350*   383*   415*   419*   424*
                     433*   437*   442*   448*   455*   459*   464*   469*   476*   480*   494*   508*   512*
                     526*   557*   593*   662*   666*   670*   677*   681*   685*   693*   697*   706*   710*
                     719*   725*   760*   784    785    815*   823    824    825    847*   869    870    902*
                     932    933    948*   1117*  1241   1244*  1247*  1250*  1275   1278*  1331*  1348*  1352*
R6      =%000006      13#   112*
R7      =%000007      13#
SAVE4     001302      85#    95*    109*   115    735*   752*
SCORD0    011334     477    1307#
SCORD1    011400     509    1308#
SETBAS    006152     726    852#
SHGL0     011125     351    1302#
SHGL1     011170     384    1303#
SHL0      010747     253    1299#
SHL1      011017     294    1300#
SHUFF     006400     188    202    227    241    270    275    310    315    363    368    396    401    622
                     640    738    744    796    801    837    842    885    890    911#   937    943    955
SINBEG    005366     733    734    749    751    765#   952
SINEND    005577     487    489    519    521    777#
SP      -%000006      13#    88     112*   140    151    763*   826*   827*   845    846    965*   966*   967*
                    1105   1106   1107   1116*  1261*  1328*  1329*  1330*  1331*  1344*  1347*  1348*  1350*
                    1352*
STACK   = 001100      13#    112    1116
STCGR0    011526     694    1311#
STCGR1    011554     707    1312#
STCORD    006014     494    526    823#
STKLMT= 177774        13#
STRIPG    006570     701    714    950#
SUBTST    001266      74#   1174
SWR       001140      29#    112*   128    130    1087   1182   1328*  1334   1337   1340   1344   1352*
SWREG     000176      19#    112    128    1328
SW0     = 000001      13#
SW00    = 000001      13#
SW01    = 000002      13#
SW02    = 000004      13#
SW03    = 000010      13#
SW04    = 000020      13#
SW05    = 000040      13#
SW06    = 000100      13#
SW07    = 000200      13#
SW08    = 000400      13#
SW09    = 001000      13#
SW1     = 000002      13#
SW10    = 002000      13#    1337
SW11    = 004000      13#    1334
SW12    = 010000      13#
SW13    = 020000      13#
SW14    = 040000      13#
SW15    = 100000      13#
SW2     = 000004      13#
```

```
SW3    = 000010              13#
SW4    = 000020              13#
SW5    = 000040              13#
SW6    = 000100              13#
SW7    = 000200              13#
SW8    = 000400              13#
SW9    = 001000              13#
TBITVE= 000014               13#
TERM    007540              969*   1078    1101*   1113#
TIME0   001264               73#   1058    1255
TIME1   010276             1058*   1065*   1255*   1266*   1270#
TIME2   010300             1059*   1062*   1256*   1264*   1271#
TITLE   010346              119    1284#
TITST   001650              115#
TKVEC  = 000060              13#
TPVEC  = 000064              13#
TRAPVE= 000034               13#    112*
TRTVEC= 000014               13#
TSTNUM  001262               71#
TST1    002122               93      99     172     174#   1148
TST10   003470              411     414#   1155
TST11   003534              432#   1156
TST12   003614              454#   1157
TST13   003674              475#   1158
TST14   004006              507#   1159
TST15   004120              554#   1160
TST16   004262              556     588#   1161
TST17   004604              592     661#   1162
TST2    002316              213#   1149
TST20   004642              676#   1163
TST21   004700              692#   1164
TST22   004734              705#   1165
TST23   004770              718#   1166
TST3    002504              251#   1150
TST4    002700              287     292#   1151
TST5    003074              327     332#   1152
TST6    003140              349#   1153
TST7    003314              378     382#   1154
TYPDS  = 104405              763    1350#
TYPE   = 104401               89     118     134     136     146     148     763    1328    1329    1331    1347    1348    1350#
                             1352
TYPOC  = 104402             1328    1350#
TYPON  = 104404             1350#
TYPOS  = 104403             1350#
UPDWN   005610              338     343     419     424     437     459     480     512     666     681     783#
UPDWNS  005600              697     710     781#
UPDWNX  005616              782     784#
UPDWNZ  006012              781*    783*    793     819#
VTIB    001274               82#   1260
VTIB0   010066             1199#
VTIB1   010076             1204#
VTIB2   010106             1209#
VTIB3   010116             1214#
VTIB4   010126             1219#
```

```
VTIB5   010136            1224#
VTIB6   010146            1229#
VTIB7   010156            1234#
VTIS    001272             81#    163    1031    1036    1089    1186    1258
VTISO   010064             102    1015    1039    1198#
VTIS1   010074            1017    1041    1203#
VTIS2   010104            1019    1043    1208#
VTIS3   010114            1021    1045    1213#
VTIS4   010124            1023    1047    1218#
VTIS5   010134            1025    1049    1223#
VTIS6   010144            1027    1051    1228#
VTIS7   010154            1029    1053    1233#
VTNOW   001260             70#     161*    164     165     167     169     763*
VTOB    001300             84#    1004*   1124*
VTOB0   010072           1007*    1201#
VTOB1   010102           1008*    1206#
VTOB2   010112           1009*    1211#
VTOB3   010122           1010*    1216#
VTOB4   010132           1011*    1221#
VTOB5   010142           1012*    1226#
VTOB6   010152           1013*    1231#
VTOB7   010162            105     1014*   1236#
VTOS    001276             83#     972    1122
VTOS0   010070            976     1200#
VTOS1   010100            978     1205#
VTOS2   010110            980     1210#
VTOS3   010120            982     1215#
VTOS4   010130            984     1220#
VTOS5   010140            986     1225#
VTOS6   010150            988     1230#
VTOS7   010160            990     1235#
WFTEST  001270             80#     94*    101*    974     1005    1176
WHATO   010561            137     149    1291#
WHAT8   010616            135    1292#
WHAT9   010630            147    1294#
WHERE   002120             93*     99*    171     172#
XOFF  = 000023             76#    1081
XON   = 000021             77#    1075    1084    1102
XPRNT   006632            194     209     233     247     283     323     341     347     374     407     422     428     440
                          446     451     462     468     472     483     499     515     531     570     584     606     628
                          646     657     673     688     702     715     757     965#    1249
ZAPIT   011743           1245    1316#
ZAPITA  011752           1243*   1319#
ZAPITB  011766            761    1324#
$APTHD  001000             28#
$ASTAT= ****** U          1330
$ATYC   012730           1330#
$ATY1   012704           1330#
$ATY3   012712           1330#   1347
$ATY4   012722           1330#
$AUTOB  001134             29#   1328
$BASE   001246             29#    117
$BDADR  001122             29#
$BDDAT  001126             29#   1073*   1074*   1075    1081    1091    1093
```

```
$CDW1   001252            29#    103
$CHARC  014052          1347#*
$CKSWR  011770          1328#   1350
$CMTAG  001100            29#    112
$CM1   = 000002            29#
$CM2   = 000004            29#
$CM3   = 000002            29#
$CNTLG  012515          1328#
$CNTLU  012510          1328#
$CPUOP  001220            29#
$CRLF   001167            29#   1328    1329    1347
$DBLK   013366          1331#
$DEVCT  001202            29#
$DEVM   001250            29#
$DOAGN  005342           763#
$DTBL   013356          1331#
$ENDAD  005332             26     763#
$ENDCT  005300            112     763#
$ENDMG  005351           763#
$ENULL  005346           763#
$ENV    001212            29#   1330    1347
$ENVM   001213            29#    112    1330    1347
$EOP    005174           763#
$EOPCT  005272           112*    763#
$ERFLG  001103            29#   1344*
$ERMAX  001115            29#   1344
$ERRPC  001116            29#
$ERRTB  001254            29#
$ERTTL  001112            29#
$ETABL  001212            29#
$ETEND  001254            28      29#
$FATAL  001174            29#   1330*
$FFLG   013150          1330#*
$FILLC  001156            29#   1347
$FILLS  001155            29#   1347
$GDADR  001120            29#
$GDDAT  001124            29#   1035*   1084*   1086*   1102*
$GET42  005322           763#
$GTSWR  012040          1328#   1350
$HD    = 000000            12
$HIBTS  001000            28#
$HIOCT  012702          1329#*
$ICNT   001104            29#
$ILLUP  014532          1352#
$INTAG  001135            29#   1328
$ITEMB  001114            29#
$LF     001170            29#   1328    1329    1347
$LFLG   013147          1330#*
$LPADR  001106            29#    112*   1140*   1344*
$LPERR  001110            29#   1141*
$MADR1  001224            29#
$MADR2  001230            29#
$MADR3  001234            29#
$MADR4  001240            29#
```

```
$MAIL   001172              28      29#     112     1344    1347
$MAMS1  001222              29#
$MAMS2  001226              29#
$MAMS3  001232              29#
$MAMS4  001236              29#
$MBADR  001002              28#
$MFLG   013146            1330#*
$MNEW   012533            1328#
$MSGAD  001206              29#   1330*
$MSGLG  001210              29#   1330*
$MSGTY  001172              29#   1330*
$MSWR   012522            1328#
$MTYP1  001223              29#
$MTYP2  001227              29#
$MTYP3  001233              29#
$MTYP4  001237              29#
$NULL   001154              29#   1347
$NWTST= 000001             174#    213#    251#    292#    332#    349#    382#    414#    432#    454#    475#    507#    554#
                           588#    661#    676#    692#    705#    718#
$OCNT   014300            1348#*
$OMODE  014302            1348#*
$OVER   013560            1343    1344#
$PASS   001200              29#    112*    555     591     763*    1184
$PASTM  001006              28#
$PWRAD  014526            1352#
$PWRDN  014366             112    1352#
$PWRMG  014522            1352#
$PWRUP  014440            1352#
$QUES   001166              29#   1328    1329    1347
$RDCHR  012252            1328#   1350
$RDDEC= ****** U          1350
$RDLIN  012372            1328#   1350
$RDOCT  012544            1329#   1350
$RDSZ = 000010            1328#
$REGAD  001160              29#
$REG0   001162              29#
$REG1   001164              2?#
$RTNAD  005344             763#
$R2A  = ****** U          1350
$SAVRE= ****** U          1350
$SAVR6  014536            1352#*
$SCOPE  013444             112    1341    1344#
$SETUP= 000135              31#    112     763     1328    1344
$STUP = 177777              31#
$SVLAD  013536            1344#
$SVPC = 000200              26#
$SWR  = 040400               5#     12      29      112     174     213     251     292     332     349     382     414     432
                           454     475     507     554     588     661     676     692     705     718     763     1344    1352
$SWREG  001214              29#    112
$SWRMK= 000000            1344
$TESTN  001176              29#   1344*
$TKB    001146              29#    138    1328
$TKS    001144              29#   1328*
$TN   = 000024               6#     12     174#    213#    251#    287     292#    327     332#    349#    378     382#    411
```

```
                        414#    432#    454#    475#    507#    554#    556     588#    592     661#    676#    692#    705#
                        718#
$TPB    001152           29#    1347*
$TPFLG  001157           29#    1347
$TPS    001150           29#    1347
$TRAP   014304          112     1350#
$TRAP2  014326         1350#
$TRP  = 000013         1350#
$TRPAD  014340         1350#
$TSTM   001004           28#
$TSTNM  001102           29#     763*    1136*   1344*
$TTYIN  012500         1328#
$TYPBN= ****** U        1350
$TYPDS  013152         1331#    1350
$TYPE   013574         1330     1347#   1350
$TYPEC  014006         1328     1347#
$TYPEX  014054         1347#
$TYPOC  014102         1348#    1350
$TYPON  014116         1348#    1350
$TYPOS  014056         1348#    1350
$UNIT   001204           29#
$UNITM  001010           28#
$USWR   001216           29#
$VECT1  001242           29#
$VECT2  001244           29#
$XTSTR  013460         1344#
$$GET4= 000000          763#
$OFILL  014301         1348#*
$40CAT= ****** U        1344
.     = 014542           19#     23#     26#     27#     28#     29#     112     763     860#    1325#   1328#   1329    1330#
                        1331#   1344    1347    1352
.$ASTA= ****** U        1330
.$X   = 001000           28#
```

```
ANYMOR    538#    763
CLRUNT     62#    254    295    336    352    385    417    435    457    478    510    560    664    679    695
          708
COMMEN     13#
ENDCOM     13#
ERROR      13#
ESCAPE     13#
GETPRI     13#
GETSWR     13#
MULT       13#
NEWTST     13#    174    213    251    292    332    349    382    414    432    454    475    507    554    588
          661    676    692    705    718
POP        13#   1329   1330   1331   1352
PUSH       13#   1329   1330   1331   1352
REPORT     13#
SCOPE      13#    174    213    251    292    332    349    382    414    432    454    475    507    554    588
          661    676    692    705    718    763
SETPRI     13#
SETTRA   1350#
SETUP      13#    112
SKIP       13#    287    327    378    411    556    592   1092
SLASH      13#
SPACE      13#
STARS      13#     26     28     29    174    213    251    292    332    349    382    414    432    454    475
          507    554    588    661    676    692    705    718    763   1328   1329   1330   1331   1344   1347
         1348   1350   1352
SWRSU      13#    112#
TRMTRP   1350#
TYPBIN     13#
TYPDEC     13#    763
TYPNAM     13#
TYPNUM     13#
TYPOCS     13#
TYPOCT     13#   1328
TYPTXT     13#
$$CMRE     29#
$$CMTM     29#
$$ESCA     13#
$$NEWT     13#    174    213    251    292    332    349    382    414    432    454    475    507    554    588
          661    676    692    705    718
$$SET    1350#
$$SETM    112#
$$SKIP     13#    287    327    378    411    556    592
.EQUAT      7#     13
.HEADE      7#     12
.SETUP      8#     31
.SWRHI      9#
.$ACT1     10#     26
.$APTB     29#
.$APTH     10#     28
.$APTY     10#   1330
.$CATC      7#     19
.$CMTA      7#     29
.$EOP       7#    763
```

ADD	166	168	170	624	642	763	1261	1328	1329	1330	1331	1347	1348		
ASL	879	1137	1328	1329	1350										
ASLB	1331														
ASR	1178	1179	1330												
BCC	1331														
BEQ	106	112	129	141	152	491	523	556	592	737	743	763	807	896	924
	954	975	998	1006	1079	1092	1127	1129	1139	1177	1185	1328	1329	1330	1335
	1338	1344	1347	1348											
BGT	763	1328	1329	1331	1348										
BHI	144	156	1131												
BIC	763	806	895	913	921	1074	1125	1134	1328	1329	1348				
BIS	922	925	1328	1331	1348										
BISB	791	833	876	881											
BIT	763	923	1182	1334	1337	1344									
BITB	112	1330	1347												
BLO	1133														
BLOS	1328														
BLT	1328	1329	1331	1347	1348										
BMI	1016	1018	1020	1022	1024	1026	1028	1030	1061	1187	1331				
BNE	112	116	122	133	192	206	231	244	279	285	319	325	372	376	405
	409	503	535	604	608	753	763	811	900	941	946	1000	1034	1063	1066
	1076	1082	1090	1094	1104	1180	1183	1189	1265	1277	1328	1330	1331	1343	1344
	1347	1348	1352												
BPL	131	626	644	804	813	893	959	973	977	979	981	983	985	987	989
	991	1032	1088	1123	1191	1259	1267	1328	1331	1341	1347	1348			
BR	98	100	108	112	123	157	287	327	378	411	488	501	520	533	748
	750	782	808	897	957	1002	1037	1040	1042	1044	1046	1048	1050	1052	1054
	1080	1085	1097	1100	1120	1144	1146	1328	1329	1330	1331	1344	1347	1348	1352
CLR	94	95	112	113	120	158	160	763	795	884	934	947	950	960	968
	969	970	1003	1035	1059	1077	1086	1098	1175	1256	1328	1329	1331	1348	1352
CLRB	193	208	232	246	282	322	373	406	569	583	605	627	645	656	756
	814	844	901	1328	1329	1330	1331	1344	1347						
CMP	88	105	112	128	143	155	191	205	230	625	643	763	810	899	940
	1075	1081	1089	1091	1093	1328	1331	1344							
CMPB	999	1126	1128	1130	1132	1328	1329	1330	1344	1347					
DEC	121	229	243	278	318	371	404	500	502	532	534	603	607	752	763
	805	812	894	945	958	1062	1065	1135	1188	1190	1264	1266			
DECB	1347	1348													
EMT	13														
HALT	19	1145	1347	1352											
INC	190	204	763	809	898	939	1001	1328	1330	1331	1348	1352			
INCB	763	1083	1339	1344	1347										
IOT	13														
JMP	19	20	21	171	763	1038	1067	1142							
JSR	175	188	194	195	202	209	210	214	227	233	234	241	247	248	252
	270	275	283	286	293	310	315	323	326	333	338	341	343	347	348
	350	363	368	374	377	383	396	401	407	410	415	419	422	424	428
	429	433	437	440	442	446	448	451	452	455	459	462	464	468	469
	472	473	476	480	483	494	499	504	508	512	515	526	531	536	557
	570	571	584	585	593	606	622	628	640	646	648	657	658	662	666
	670	673	674	677	681	685	688	689	693	697	701	702	703	706	710
	714	715	716	719	725	738	744	757	758	760	763	796	801	837	842
	885	890	937	943	955	1117	1119	1244	1247	1249	1328	1330	1347		
MOV	93	96	97	99	101	102	103	104	109	112	114	117	138	140	142

	[illegible]	154	159	161	163	164	165	167	169	178	187	189	197	201	203
	217	226	228	236	240	242	256	257	269	271	274	276	277	297	298
	[illegible]	311	314	316	317	354	355	362	364	367	369	370	387	388	395
	39[illegible]	400	402	403	485	486	487	489	492	517	518	519	521	524	561
	575	595	596	601	602	612	621	623	634	639	641	652	724	733	734
	735	741	747	749	751	763	781	783	784	785	786	792	797	800	802
	823	824	825	826	827	828	836	838	841	843	845	846	869	870	871
	878	886	889	891	911	912	914	926	938	944	951	952	956	965	966
	967	971	1036	1039	1041	1043	1045	1047	1049	1051	1053	1058	1071	1073	1084
	1095	1096	1099	1101	1102	1105	1106	1107	1116	1140	1141	1174	1181	1241	1242
	1255	1260	1275	1328	1329	1330	1331	1336	1344	1347	1348	1350	1352		
MOVB	179	180	181	182	183	184	185	186	198	199	218	219	220	221	222
	223	224	225	237	238	258	259	260	261	262	263	265	266	267	268
	272	273	299	300	301	302	303	304	305	306	307	308	312	313	356
	357	358	359	360	361	365	366	389	390	391	392	393	394	398	399
	490	522	562	563	564	565	566	567	568	576	577	578	579	580	581
	582	597	598	599	600	613	614	615	616	617	618	619	620	635	636
	637	638	653	654	655	728	729	730	731	736	739	740	742	745	746
	754	755	763	787	788	789	790	793	794	798	799	829	830	831	832
	834	835	839	840	872	873	874	875	877	880	882	883	887	888	920
	929	930	931	932	933	935	953	997	1004	1007	1008	1009	1010	1011	1012
	1013	1014	1124	1136	1243	1276	1328	1329	1330	1331	1344	1347	1348	1350	
NEG	1331	1348													
NOP	90	91	763	992	993	994	995	996							
RESET	110	763													
ROL	1329	1348													
ROR	284	324	375	408	915	916	917	918	919						
RTI	112	1328	1329	1331	1344	1347	1348	1350	1352						
RTS	815	847	902	927	948	961	1108	1192	1250	1262	1268	1278	1330	1347	1350
SUB	1330	1331													
TRAP	1350														
TST	107	115	132	145	153	555	591	763	803	892	974	1005	1033	1072	1078
	1[illegible]38	1[illegible]75	1184	1328	1329	1330	1331	1342	1344	1347	134[illegible]	1350			
TSTB	130	[illegible]27	972	976	978	980	982	984	986	988	[illegible]90	1015	1017	1019	1021
	1023	1025	1027	1029	1031	1060	1087	1103	1122	1186	[illegible]46	1258	1328	1330	1331
	1340	1347													
.ASCII	29														
.ASCIZ	29	763	1283	1288	1289	1290	1291	1293	1295	1296	1297	1298	1299	1300	1301
	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311	1312	1313	1314	1315	1328
.BLKB	1328														
.BLKW	1331														
.BYTE	29	339	344	420	425	438	443	449	460	465	470	481	495	513	527
	667	671	682	686	698	711	763	765	766	767	768	769	770	771	772
	773	774	775	776	777	852	853	854	855	856	857	858	859	1284	1285
	1286	1287	1292	1294	1316	1317	1318	1319	1320	1321	1322	1323	1324	1328	1330
	1348														
.DSABL	1328														
.ENABL	4	1328													
.END	1358														
.ENDC	12	13	19	26	28	29	31	112	174	213	251	287	292	327	332
	349	378	382	411	414	432	454	475	507	554	556	588	592	661	676
	692	705	718	763	1092	1328	1329	1330	1331	1344	1347	1348	1350	1352	
.EQUIV	13														
.EVEN	29	778	860	1325	1330										

ERRORS DETECTED: 0

```
*CZVTNA,CZVTNA/CRF=CZVTNA
RUN-TIME: 23 9 1 SECONDS
CORE USED:  25K
```