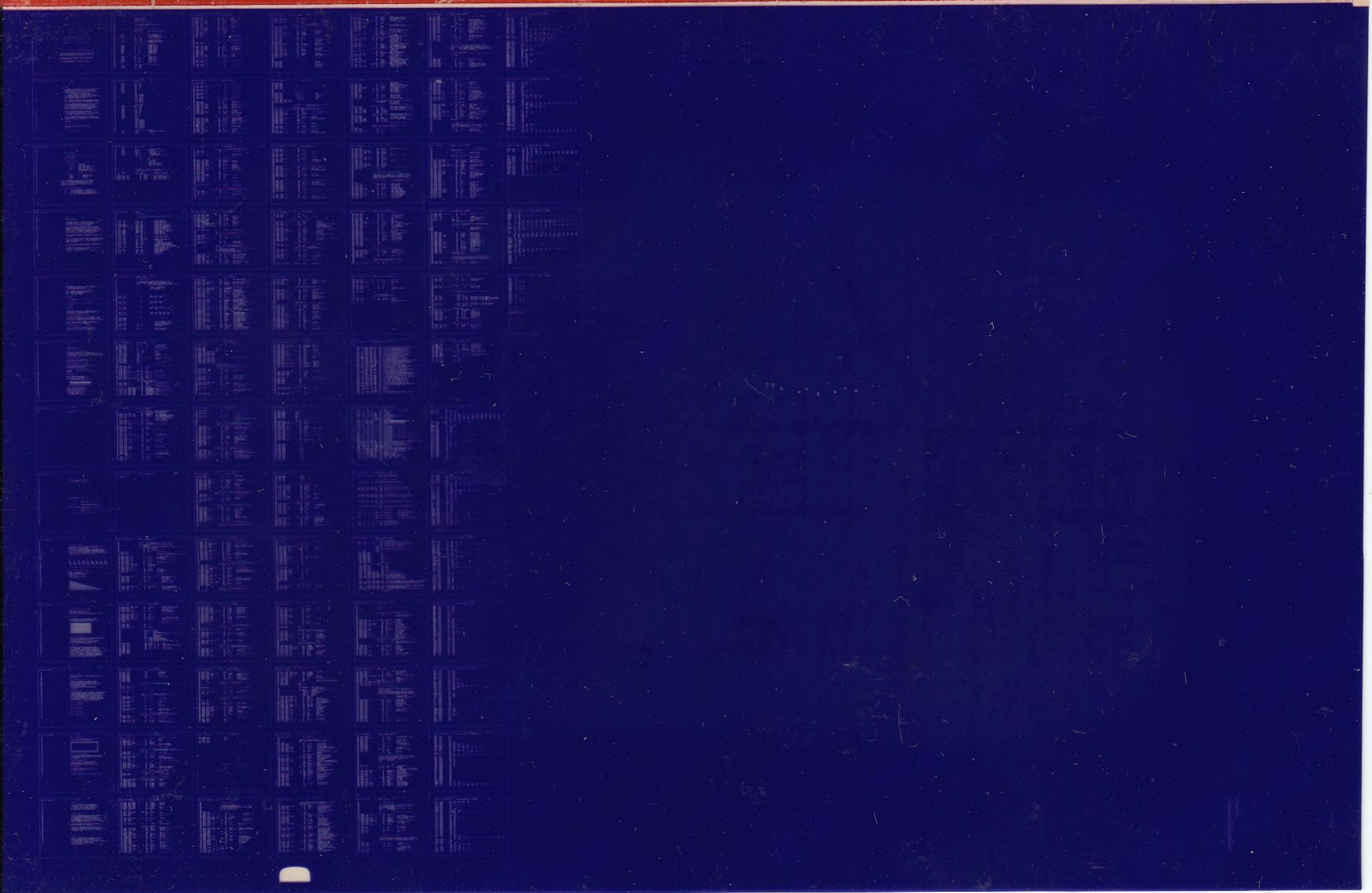
VT50,52,55

VT50A,B,H/52 ACCPT
CZVTCF0

AH-9508F-MC

FICHE 1 OF 1

SEP 1979
digital
MADE IN USA



```
                        .REM ^
```

IDENTIFICATION

PRODUCT CODE: AC-9506F-MC

PRODUCT NAME: CZVTCF0 VT50A,B,H/52 ACCPT

PRODUCT DATE: APRIL 1979

MAINTAINER: DIAGNOSTIC ENGINEERING

1. ABSTRACT

THIS PROGRAM IS AN ACCEPTANCE TEST OF THE VT50/52 VIDEO TERMINAL.
THE PROGRAM CONSISTS OF FOUR PARTS, ALL OF WHICH REQUIRE
OPERATOR INSPECTION OR INTERACTION. THE PROGRAM IS CAPABLE
OF HANDLING MULTIPLE UNITS IN A SEQUENTIAL DL11 FASHION (REF 8.2).

****
ONLY ONE VT50/52 IS TESTED AT ONE TIME.
****

THE PROGRAM WILL DEFAULT TO THE CONSOLE TTY (REF 5.0 AND 8.2).
ALL CHARACTERS AND COMMANDS ARE TESTED. IN THE KEYBOARD CHARACTER
TEST THE FOLLOWING 'FUNCTION'' KEYS ARE NOT TESTED:
BREAK, REPEAT, AUTO-PRINT, AND SCROLL.

PART 1 CONSISTS OF A SERIES OF TEST PATTERNS DISPLAYED ON THE
VT50/52 SCREEN AND COPIER (REF 9. FOR DESCRIPTION). THE OPERATOR
MUST VISUALLY INSPECT EACH TEST PATTERN FOR ERROR DETECTION.

PART 2 IS A KEYBOARD CHARACTER TEST. THIS TEST IS TO DETERMINE
THAT THE TERMINAL IS GENERATING THE EXPECTED ASCII CODES.
IN THIS TEST AN OPERATOR WILL BE REQUIRED TO FOLLOW THE
INSTRUCTIONS DISPLAYED ON THE VT50/52 SCREEN AND EXECUTE THEM.
DUE TO THE FLEXIBILITY OF DIFFERENT PROCESSORS OR OPTIONS,
PARITY BIT TESTING MUST BE SELECTED BY THE OPERATOR. THE OPERATOR
SELECTS THE TYPE OF PARITY TO BE TESTED BY SW00-01.

PART 3 IS A KEYBOARD OCTAL VALUE LOOP. WHEN A KEY IS DEPRESSED,
THE OCTAL VALUE WILL BE DISPLAYED ON THE SCREEN.
IF THE KEY DEPRESSED WAS PRINTABLE, IT WILL ALSO BE DISPLAYED
ON THE SCREEN. IF A 'DEFINED'' CHARACTER, A TWO LETTER EQUALIVENT
(I.E., BL=BELL, ES=ESCAPE, ETC.) WILL BE DISPLAYED.

PART 4 IS A KEYBOARD ECHO LOOP. WHEN A KEY IS DEPRESSED, THE
CHARACTER IS ECHOED TO THE SCREEN. NO TESTING OF THE CHARACTER
IS PERFORMED. THIS ALLOWS THE OPERATOR A 'LOCAL' MODE OF OPERATION
BETWEEN THE VT50/52 AND THE HOST COMPUTER.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 FAMILY COMPUTER WITH 8K WORDS OF MEMORY.
VT50A, B, H, 52 VIDEO TERMINAL CONNECTED VIA A DL11A/B TYPE
INTERFACE.

2.2 STORAGE

THIS PROGRAM USES 8K OF MEMORY.

3. LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

SW 15 = 1	HALT ON ERROR
SW 14 = 1	LOOP ON TEST
SW 13 = 1	INHIBIT ERROR TYPEOUTS
SW 12 = 1	INHIBIT PROGRAM SUB-TEST DELAY
SW 11 = 1	INHIBIT COPIER TESTING
SW 10 = 1	ENABLE ''SAVE COPIER PAPER'' MODE
SW 08 = 1	LOOP ON TEST IN SWR <4:0>
SW 07 = 1	KEYBOARD CONTROL OF THE TEST <SW 8 AND SW 7 = 1 IS AN ERROR>

KEYBOARD CHARACTER TEST ONLY

SW02 =	1	ENABLE PARITY BIT TEST
SW00-01=	00	EVEN PARITY CHECK
SW00-01=	01	ODD PARITY CHECK
SW00-01=	10	ALWAYS A 0
SW00-01=	11	ALWAYS A 1

SPECIAL NOTE: IF THE COMPUTER UTILIZED IS AN LSI-11 OR A COMPUTER WITHOUT A SWITCH REGISTER,THE PROGRAM WILL UTILIZE LOCATIONS 174 AND 176 AS A ''DISPLAY'' REGISTER AND A ''SWITCH'' REGISTER RESPECTIVELY. THE OPERATOR WILL BE RESPONSIBLE FOR THE LOADING OF THE ''SWITCH'' REGISTER LOCATION PRIOR TO STARTING OR RESTARTING THE PROGRAM.

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 STARTING ADDRESS OF THE KEYBOARD CHARACTER TEST
214	IS THE STARTING ADDRESS OF THE KEYBOARD OCTAL VALUE LOOP
220	IS THE STARTING ADDRESS OF THE KEYBOARD ECHO LOOP
224	IS THE SPECIAL STARTING ADDRESS FOR VT50 PRODUCTION

5. OPERATING PROCEDURE

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

THIS PROGRAM ALLOWS THE OPERATOR TWO MODES OF TEST PATTERN SELECTION. THESE MODES ARE SELECTED BY THE STATE OF SW07 AT THE BEGINNING OF THE PROGRAM. WHEN SW07 IS A ZERO, THE PROGRAM IS UNDER SWITCH REGISTER CONTROL FOR TEST PATTERN SELECTION. IF SW07 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 DL11'S TO WHICH VT50'S ARE CONNECTED.

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

THE ''/'' KEY IS USED TO SUSPEND THE CURRENT TEST AND ASK THE OPERATOR A 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 TFST 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 SW07 EQUALS A ZERO, AN ERROR WILL BE REPORTED TO THE CONSOLE TTY. IF SW07 EQUALS A ONE, AND THE CHARACTER RECEIVED WAS NOT A ''/'' OR ''\'', AN ERROR WILL BE REPORTED. THE CODES ''X-OFF'' AND ''X-ON'' ARE THE ONLY EXCEPTIONS.

6. ERRORS

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

ERRPC - LOCATION AT WHICH AN ERROR WAS DETECTED
VTNOW - CURRENT DL11 BUS ADDRESS OF VT50/52 UNDER TEST
TSTNUM- TEST PATTERN NUMBER OF FAILING TEST
EXPT - EXPECTED INPUT CHARACTER
RCVD - RECEIVED INPUT CHARACTER

7. RESTRICTIONS

A. THE OPERATOR SHOULD SET SW15 AND SW13 IF THE VT50/52 UNDER TEST
   IS THE CONSOLE TTY.

B. ONLY ONE VT50/52 CAN BE TESTED AT ONE TIME.

C. THE FIRST TIME AFTER LOADING THE PROGRAM, THE TERMINAL IDENTIFIER
   MUST BE RUN.

8. MISCELLANEOUS

8.1 EXECUTION TIME

EXECUTION TIME WILL VARY WITH THE 'BAUD'' RATE, AND IF A COPIER
IS CONNECTED. THE PROGRAM WILL TYPE 'END PASS' WHEN A PASS
HAS BEEN COMPLETED. THE KEYBOARD LOOP AND CHARACTER TEST WILL
NOT EXIT UNTIL THE PROGRAM IS RESTARTED.

8.2 DEVICE ADDRESS PROGRAM LOCATIONS (AT APPROX. 1240)

THE LOCATION 'FIRST'' CONTAINS THE FIRST DL11 ADDRESS IF SEVERAL VT50'S
ARE BEING TESTED. THE DEFAULT IS THE CONSOLE ADDRESS <177560>
THE LOCATION 'LAST'' CONTAINS THE LAST DL11 ADDRESS IF SEVERAL VT50'S
ARE BEING TESTED. LOCATION VTNOW CONTAINS THE CURRENT DL11 BASE ADDRESS.

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

8.3 COPIER SAVE PAPER SWITCH

IF SW10 = 1 AND A COPIER IS INSTALLED, THE COPIER TESTS
WILL BE EXECUTED ON THE FIRST PASS AND THEN BYPASSED FOR
THE NEXT TEN PASSES. THIS REDUCES PAPER USAGE WHEN THE
PROGRAM IS RUN FOR AN EXTENDED PERIOD. (LOC. PTCT IS # OF PASSES.)

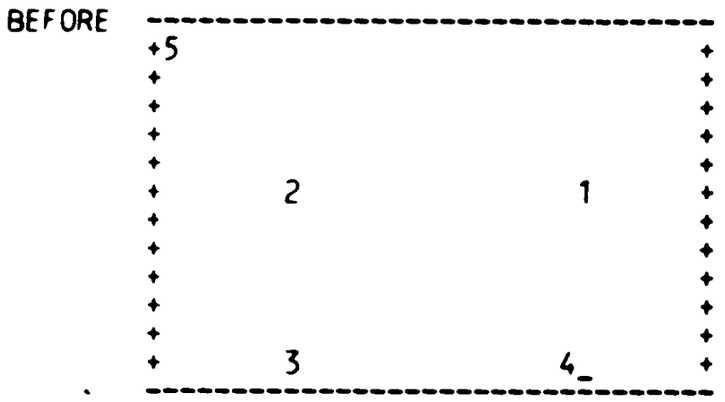
```
250
251
252
253                     9.      PROGRAM DESCRIPTION <SCREEN>
254                             ----------------------
255
256                     9.1  A  TERMINAL IDENTIFICATION TEST
257
258                             THIS TEST WILL INTERROGATE THE VT50/52 UNDER TEST AS TO ITS TYPE.
259                             THE RESPONSE RECEIVED IS USED TO DETERMINE THE MODES TO BE TESTED
260                             (I.E., COPIER, 12/24 LINES, D.C.A. <DIRECT CURSOR ADDRESSING>, ETC.).
261                             IF AN UNDEFINED OR AN INCORRECT RESPONSE IS RECEIVED, THE PROGRAM
262                             ASSUMES VT50A MODEL (I.E., 12 LINES, NO COPIER, NO D.C.A., NO KEYPAD).
263
264                     9.2  B  FULL SCREEN OF THE LETTER E
265
266                             THIS TEST WILL FILL THE SCREEN WITH THE LETTER E.
267                             THIS TEST WILL LOAD THE SCREEN RAM WITH A SINGLE
268                             CHARACTER IN ALL LOCATIONS.
269
270                     9.3  C  DATA PATH AND RAM NOISE TEST
271
272                             THIS TEST WILL PRODUCE TWO FULL SCREENS OF A WORST CASE
273                             NOISE PATTERN FOR THE SCREEN RAM AND THE DATA PATH.
274                             ALTERNATING LINES OF THE FOLLOWING
275                             PATTERNS SHOULD BE DISPLAYED.
276
277                             *U*U*U*U*U          CODES 52 AND 125
278                             @?@?@?@?@?          CODES 77 AND 100
279                             *U*U*U*U*U
280                             @?@?@?@?@?
281
282                     9.4  D  SINGLE CHARACTER SET
283
284                             ONE FULL LINE OF EACH CHARACTER
285                             (CODES 40 THRU 137) OF THE CHARACTER
286                             SET.  THIS WILL ALSO TEST THAT ALL WORDS
287                             IN THE SCREEN RAM CAN BE LOADED.
288                             THIS WILL ALSO EXERCISE THE SCROLLING FUNCTION.
289
290                             IE:  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
291                                  BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
292                                  CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
293                                  DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
294
295                     9.5  E  INCREMENTING AND SLIDING CHARACTER SET
296
297                             ONE FULL LINE OF AN INCREMENTING
298                             CHARACTER SET ACROSS THE FULL SCREEN
299                             STARTING WITH THE CODE 40 (SPACE)
300                             THE NEXT LINE SHOULD BEGIN WITH THE
301                             "!" CHARACTER (CODE 41), ETC.  CONTINUE
302                             THE PATTERN BY INCREMENTING THE
303                             FIRST COLUMN CHARACTER UNTIL THE
304                             FULL CHARACTER SET HAS BEEN EXHAUSTED.
305
```

```
306                                     THIS PATTERN WILL VERIFY THAT THE FULL
307                                     CHARACTER SET CAN BE DISPLAYED IN EACH
308                                     SCREEN WORD.
309
```

9.6 F CURSOR MOTION

IN THIS TEST THE BASIC CURSOR MOTIONS ARE
TESTED. THE FOLLOWING TEST PATTERN
IS GENERATED TO TEST THE CURSOR FUNCTIONS

BEFORE



UPON COMPLETION OF THE SETUP THE
BLINKING CURSOR WILL BE TO THE RIGHT
OF #4.

TO TEST "CURSOR UP": GENERATE CURSOR UP 6/12 TIMES AND
DISPLAY A 'X'

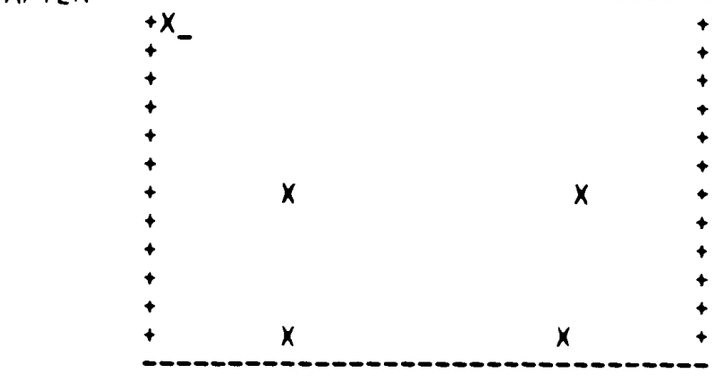
TO TEST "CURSOR LEFT": GENERATE CURSOR LFFT (BACKSPACE)
FORTY-FOUR TIMES AND DISPLAY A 'X'

TO TEST "CURSOR DOWN": GENERATE CURSOR DOWN 6/12 TIMES AND
DISPLAY A 'X'

TO TEST "CURSOR RIGHT": GENERATE CURSOR RIGHT FORTY-TWO TIMES AND
DISPLAY A 'X'

TO TEST "CURSOR HOME": GENERATE CURSOR HOME AND DISPLAY A 'X'

AFTER



```
9.7  G  TAB, BACKSPACE AND BELL TEST

        THIS PATTERN WILL TEST THAT THE 'TAB' FUNCTIONS CORRECTLY FROM EACH
        COLUMN POSITION. GENERATE A REFERENCE LINE USING SPACES
        AND THE LETTER 'I'.   ON THE NEXT LINE
        RING THE BELL AND DISPLAY ONE CHARACTER IN COLUMN ONE AND EXECUTE A 'TAB
        THE CURSOR SHOULD THEN BE POSITIONED BELOW THE REFERENCE TAB STOP.
        REPEAT THE ONE CHARACTER IN THE FIRST TAB COLUMN ACROSS THE SCREEN.
        ON THE NEXT LINES USE INCREASING NUMBERS OF CHARACTERS BEFORE THE 'TAB'
        THE BELL WILL RING AT THE BEGINNING OF EACH LINE. THE SECOND SECTION TES
        THE BACKSPACE CODE. A FULL LINE OF THE CHARACTER "X" WILL BE DISPLAYED
        STARTING AT THE RIGHT EDGE AND EXTENDING TO THE LEFT.


        ------------------------------------------------------------------------
       +I       I       I       I       I       I       I       I       I
       +I       I       I       I       I       I       I       I       I
       +IA      IA      IA      IA      IA      IA      IA      IA      IA
       +IAA     IAA     IAA     IAA     IAA     IAA     IAA     IAA     IAA
       +IAAA    IAAA    IAAA    IAAA    IAAA    IAAA    IAAA    IAAA    IAAA
       +IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA
       +IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA
       +IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA


        XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX


9.10 H  ERASE FROM CURSOR TO END OF LINE:

   H.1  GENERATE A FULL SCREEN OF THE CHARACTER "E"
   H.2  EXECUTE A "CURSOR HOME"
   H.3  EXECUTE "ERASE FROM CURSOR TO END OF LINE"
   H.4  EXECUTE "CURSOR RIGHT" 6/3 TIMES
   H.5  EXECUTE "LF"
        REPEAT H.3 THRU H.5 12/24 TIMES
        THE END DISPLAY SHOULD APPEAR TO BE A
        STAIR STEP PATTERN


        EEEEEE
        EEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
        EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

```


9.11 I  ERASE FROM CURSOR TO END OF THE SCREEN:

        GENERATE A FULL SCREEN OF THE CHARACTER ''E''
        EXECUTE FOURTY ''CURSORS LEFT''
        EXECUTE ''ERASE SCREEN'' AND ''CURSOR UP''
        REPEAT THIS 12 OR 24 TIMES, UPON COMPLETION THE PATTERN WILL
        BE A LINE OF FOURTY CHARACTERS AT THE TOP LEFT OF THE SCREEN.


9.12 J  VIDEO COUPLING:

        THIS PATTERN WILL ALLOW FOR THE CHECKING OF VIDEO
        COUPLING BETWEEN THE VIDEO AND VERTICAL CIRCUITS.
        IF COUPLING OCCURS, THE VERTICAL DOTS OF THE CHARACTERS
        WILL BE UNEVENLY SPACED OR DOTS WILL SPARKLE.

        T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
        T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
        T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
        T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
        T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
        T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
        T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
        T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
        T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
        T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
        T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
        T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T


9.13 K  DIRECT CURSOR ADDRESSING (D.C.A.)

        THIS TEST IS ONLY EXECUTED ON A VT50H. WHEN EXECUTED ON A VT50H
        THIS TEST WILL BE RUN TWO TIMES. THE FIRST TIME WILL BE WITH
        AN ''ESC-Y'' AND THE SECOND WITH AN ''CODE 16''.
        IF AN INCORRECT I.D., THIS TEST WILL NOT BE EXECUTED.
        THIS TEST WILL RANDOMLY FILL THE SCREEN. THE END RESULT
        WILL BE THE SAME STATEMENT ON ALL VERTICAL LINES.
        EACH OF THE LINES SHOULD APPEAR THE SAME.
9.14 L  HOLD SCREEN TEST

        NOT EXECUTED IF VT50/52 PRODUCTION STARTING ADDRESS.
        THIS TEST DETERMINES THAT THE ''XON-XOFF'' FEATURE IS FUNCTIONING.
        A FULL SCREEN SCROLL IS EXECUTED AND A SUB-TEST TITLE
        WILL BE DISPLAYED. THE HOLD SCREEN MODE IS ENABLED AND
        THE PROGRAM WILL ATTEMP TO SCROLL THE SCREEN AGAIN AND SEND A MESSAGE.
        THIS WILL CAUSE AN ''X-OFF'' TO BE SENT AND THE SCREEN IS
        INHIBITED FROM SCROLLING. ANY ADDITIONAL CHARACTERS RECEIVED
        WILL BE PLACED INTO THE SILO STORAGE BUFFER. UPON RECEIPT
        OF AN ''X-OFF'', THE PROGRAM WILL DISABLE HOLD SCREEN MODE AND
        SHOULD RECEIVE AN ''X-ON''. TO CHECK PROPER SILO OPERATION,
        THE MESSAGE ''TESTING SILO REG'' SHOULD APPEAR UNDER THE SUB-TEST TITLE
        FOLLOWED BY ''SILO TEST DONE'' ON THE NEXT LINE.
```

COPIER TEST PATTERNS

THE TEST PATTERNS USED TO TEST THE COPIER ARE THE SAME PATTERNS
AS SOME OF THE SCREEN TESTS. TWO ADDITIONAL PATTERNS HAVE
BEEN INCLUDED.

9.15 M GRAPHICS MODE/REVERSE LINE FEED TEST

THIS TEST IS EXECUTED ONLY IF THE UNIT IS A VT52.
GRAPHICS MODE IS ENTERED AND 37 LINES OF GRAPHICS
CHARACTERS(LINE LENGTH IS DECREMENTED BY ONE CHAR. AFTER
A LINE IS PRINTED)ARE GENERATED. THE BOTTOM LINE OF THE
WILL FORM A COMPLETE SERIES OF GRAPHIC CHARACTERS WHICH
CAN BE VERIFIED FOR ACCURACY.
IF REVERSE LINE FEED IS NOT OPERATIONAL A SINGLE
LINE OF GRAPHICS CHARACTERS WILL BE GENERATED
ON LINE 0 ONLY.

9.16 N COPIER - AUTO COPY MODE

THIS TEST DETERMINES IF THE AUTO COPY MODE IS FUNCTIONING CORRECTLY.
THE SUB-TEST TITLE AND A ROW OF THE LETTER 'E' WILL BE DISPLAYED.
THE AUTO-COPY MODE IS ENABLED. THE COPIER SHOULD THEN START.
A PROGRAM DELAY IS EXECUTED BETWEEN EACH LINE. THE RESULT
IS THE COPIER SHOULD COPY ONE LINE AND THEN STOP FOR THE PROGRAM
DELAY TIME. WHEN THE PROGRAM DELAY IS COMPLETED ANOTHER LINE
OF 'E'S WILL BE DISPLAYED AND THE COPIER SHOULD START AGAIN.
UPON COMPLETION OF THE 12 OR 24 LINES, DISABLE AUTO-COPY MODE
IS SENT TO THE VT50. IF THE AUTO COPY MODE IS NOT DISABLED
THE NEXT SUB-TEST WOULD PERFORM IN A SIMILAR MANNER.

9.17 O COPIER - FULL SCREEN OF THE LETTER E

SAME AS SCREEN PATTERN

9.20 P COPIER - DATA PATH TEST PATTERN

SAME AS SCREEN PATTERN

9.21 Q COPIER - SINGLE CHARACTER PER LINE

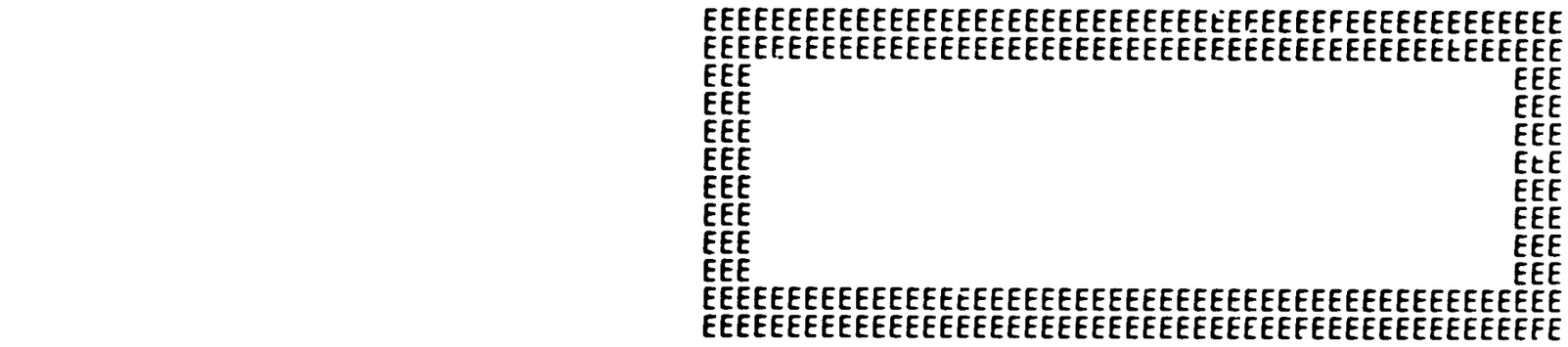
SAME AS SCREEN PATTERN

9.22 R COPIER - ROTATING CHARACTER TEST

SAME AS SCREEN PATTERN

9.23 S COPIER - PERIMETER TEST

THIS TEST COPIES THE PERIMETER OF THE SCREEN AND THE
RESULTING PICTURE SHOULD RESEMBLE BELOW



9.24 T COPIER - DISCLAIMER STATEMENT

IN THIS TEST THE DISCLAMER STATEMENT FROM THE COVER PAGE
OF THIS DOCUMENT IS DISPLAYED ON THE SCREEN AND THEN COPIED.
THE COPIED VERSION SHOULD BE COMPARED TO THE FRONT COVER TO
CHECK FOR ERRORS.

9.25 U PRINTER CONTROLLER MODE TEST

A 'ROLLING' PATTERN OF INCREMENTING CHARACTERS IS
ISSUED TO THE UNIT AFTER PRINTER CONTROLLER
MODE HAS BEEN ENTERED. 92 LINES(OF 132 CHAR. EACH)
SHOULD BE PRINTED WITH NO DATA APPERING ON SCREEN.

9.26 V PRINT SCREEN TEST

THE SCREEN IS FILLED WITH 24 LINES OF 'E'S. THE
PRINT SCREEN ESCAPE SEQUENCE IS ISSUED AND ALL
24 LINES SHOULD BE PRINTED.

9.27 W AUTO PRINT TEST

OPERATION IS SAME AS AUTO COPY TEST EXCEPT THAT
THE PRINTER, RATHER THAN THE COPIER, IS THE DESTINATION.

568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615

9.30 X KEYBOARD OCTAL VALUE LOOP

THIS LOOP IS PROVIDED TO ENABLE THE OPERATOR TO
EXAMINE THE OCTAL VALUE OF A CHARACTER. WHEN A KEY
IS DEPRESSED, THE OCTAL VALUE WILL BE DISPLAYED.
IF THE CHARACTER WAS A PRINTABLE CHARACTER, IT WILL
BE DISPLAYED. THOSE CODES DEFINED AS ''CONTROL''
WILL BE DISPLAYED AS A TWO LETTER MNEMONIC, E.G.,
DE=DELETE, BL=BELL, CL=CURSOR LEFT, ETC.

9.31 Y KEYBOARD CHARACTER TEST

THIS TEST IS DESIGNED TO VERIFY THAT CORRECT CHARACTER CODES
AND PARITY BIT ARE GENERATED WHEN A KEY IS DEPRESSED.
THIS TEST REQUIRES THE OPERATOR TO EXECUTE THE
INSTRUCTIONS DISPLAYED ON THE SCREEN. THE OPERATOR SHOULD
ONLY DEPRESS ONE KEY AT A TIME WITH SOME EXCEPTIONS.
THE OPERATOR WILL BE REQUIRED TO SKIP THOSE KEYS THAT
ARE NOT IMPLEMENTED IF THE UNIT IS A VT50. THE PROGRAM WILL
INFORM THE OPERATOR WHICH ROW TO TEST.

IN TESTING THE PARITY BIT, SW0 AND SW1 ARE USED TO
INFORM THE PROGRAM OF THE EXPECTED PARITY. AN INCORRECT
SWITCH SETTING WILL RESULT IN AN ERROR.

A KEYPAD TEST IS INCLUDED FOR THE VT50H AND VT52. IF AN
INCORRECT KEY IS DEPRESSED DURING THE TEST, THE BAD KEY
MNEMONIC AND THE BAD KEY CODE ARE DISPLAYED AS WELL AS A PROMPT
TO THE OPERATOR INDICATING THE EXPECTED KEY MNEMONIC AND
THE EXPECTED KEY CODE.

9.32 Z KEYBOARD ECHO LOOP

WHEN A KEY IS DEPRESSED, THE CHARAC .R WILL BE DISPLAYED.
NO MODIFICATION OR DATA TEST IS PERFORMED. THIS TEST CAN
BE USED TO DETERMINE IF THERE IS A ''UART'' OR SERIAL
LINE PROBLEM. WHEN THE VT50/52 IS IN LOCAL MODE (NO UART/SERIAL LINE)
THE CHARACTERS SHOULD BE ECHOED CORRECTLY. IF NOT, THE PROBLEM
IS IN THE VT50 UNIT. IF CHANGED TO REMOTE
MODE AND THE CHARACTERS ARE IN ERROR, THERE IS A UART/
SERIAL LINE PROBLEM.

```
                        .TITLE  CZVTCF0 VT50A,B,H/52 ACCPT
                        ;*COPYRIGHT (C) 1979
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY TERMINAL DIAGNOSTICS
                        ;*
                        ;*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
020000                  SW13=   20000
010000                  SW12=   10000
```

```
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;''T'' BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
```

```
                        000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        000024                  PWRVEC= 24              ;;POWER FAIL
                        000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034                  TRAPVEC=34              ;;''TRAP'' TRAP
                        000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                                ;*
                                                ;*      SWITCH                  USE
                                                ;*      ------          --------------------
                                                ;*        15            HALT ON ERROR
                                                ;*        14            LOOP ON TEST
                                                ;*        13            INHIBIT ERROR TYPEOUTS
                                                ;*        12            INHIBIT SUB-TEST DELAY'S
                                                ;*        10            ENABLE SAVE COPIER PAPER MODE
                                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                                .SBTTL  TRAP CATCHER

                        000000                          .=0
                                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
                                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                          .=174
  000174  000000        DISPREG: .WORD  0                ;;SOFTWARE DISPLAY REGISTER
  000176  000000        SWREG:   .WORD  0                ;;SOFTWARE SWITCH REGISTER
                        .SBTTL  STARTING ADDRESS(ES)
  000200  000137  001334        JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
  000204  000137  001364        JMP     RBEGIN          ;JUMP TO RESTART ADDRESS
  000210  000137  001410        JMP     BEGIN1          ;JUMP TO KEYBOARD CHARACTER TEST
  000214  000137  001420        JMP     BEGIN2          ;JUMP TO CHAR OCTAL VALUE LOOP
  000220  000137  001400        JMP     BEGIN3          ;JUMP TO ASCII ECHO LOOP
  000224  000137  001360        JMP     MANFU           ;JUMP TO W.F. SPECIAL TEST PARAM.
```

```
                        .SBTTL  COMMON TAGS

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

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

```
                        .SBTTL  ERROR POINTER TABLE

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

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

001172                  $ERRTB:

                        ;ITEM   1
001172  021102                  EM1             ;ERROR FLAG ON HOST TRANSMIT STATUS
001174  021325                  DH1             ;ERRPC  VTNOW   TSTNUM
001176  021472                  DT1             ;$ERRPC VTNOW   TSTNUM
001200  000000                  0

                        ;ITEM   2
001202  021147                  EM2             ;NO HOST INPUT FLAG RECEIVED
001204  021325                  DH1             ;ERRPC  VTNOW   TSTNUM
001206  021472                  DT1             ;$ERRPC VTNOW   TSTNUM
001210  000000                  0

                        ;ITEM   3
001212  021176                  EM3             ;INCORRECT I.D. RESPONSE
001214  021354                  DH3             ;ERRPC  VTNOW   1ST WD  2ND WD  3RD WD
001216  021502                  DT3             ;$ERRPC VTNOW   SAVE4   SAVE2   SAVE3
001220  000000                  0

                        ;ITEM   4
001222  021222                  EM4             ;INCORRECT OR UNEXPECTED INPUT CHARACTER
001224  021423                  DH4             ;ERRPC  VTNOW   TSTNUM  EXPT    RCVD
001226  021516                  DT4             ;$ERRPC VTNOW   TSTNUM  $GDDAT  $BDDAT
001230  000000                  0

                        ;ITEM   5
001232  021265                  EM5             ;INVALID BUSS ADDRESS, TRY AGAIN
001234  000000                  0
001236  000000                  0
001240  000000                  0
001242  177560  FIRST:  177560          ;FIRST DEVICE ADDRESS OF SEQUENTIAL DL11-A/B TYPE DEVICE
                                        ;DEFAULT TO THE CONSOLE ADDRESS
001244  000000  LAST:   0               ;LAST DEVICE ADDRESS OF DL11-A/B TYPE
001246  177560  VTNOW:  177560          ;CURRENT DEVICE BUSS ADDRESS
001250  100011  PTCT:   100011          ;COPIER PAPER SAVE COUNT <LOW BYTE>
001252  000000  TSTNUM: 0               ;ERROR PATTERN

001254  000300  TIMEO:  300             ;CHARACTER FLAG TIMEOUT CONSTANT
001256  000005  SUBTST: 5               ;SUBTEST DELAY CONSTANT
001260  000000  VT5XX:  0               ;I.D. AND CHARACTERISTICS
```

```
                        LASTLN: 53      ;OR 67          ;LAST VALID LINE # +40
                        TOTALC: 960.    ;OR 1920.       ;TOTAL CHARACTER COUNT
                        VH0:    12.     ;24.            ;VERTICAL LINE COUNT
                        VH1:    6.      ;12.            ;1/2 VERTICAL LINE COUNT
                        VH2:    3.      ;6.             ;1/4 VERTICAL LINE COUNT
                        PRTCNT: 0
                        VTIS:   177560                  ;DEVICE ADDRESSES
                        VTIB:   177562                  ;IN DATA
                        VTOS:   177564                  ;OUT STAT
                        VTOB:   177566                  ;OUT DATA
                        STCHAR: 0                       ;TEMP REG'S
                        LASTCH: 140     ;OR 200         ;FIRST NON-VALID CHARACTER
                        TEMP:   0                       ;TEMP REG'S
                        TEMP0:  0                       ;
                        PNTWID: 132.                    ;COLUMN COUNT FOR LINE PRINTER.
                        WIDTH:  80.                     ;COLUMN WIDTH
                        SAVE1:  0                       ;TEMP REG'S
                        SAVE2:  0
                        SAVE3:  0
                        SAVE4:  0
                        WFTEST: 0                       ;NON-ZERO IF SA = 224

002402  BEGIN:  MOV     #TST2,WHERE     ;STARTING ACCEPTANCE TEST ADDRESS
                CLR     SAVE4
                CLR     WFTEST
                CLR     PRTCNT
                BR      GINA
MANFU:  INC     WFTEST          ;SET W.F. PARAM.
RBEGIN: CLR     PRTCNT          ;RESTART ADDRESS
                MOV     #TST2,WHERE
                BR      GIN
BEGIN3: MOV     #KRBECH,WHERE   ;START AT ECHO LOOP
                BR      GIN
BEGIN1: MOV     #KRBTST,WHERE   ;STARTING CHARACTER TEST ADDRESS
                BR      GIN
BEGIN2: MOV     #KRBECO,WHERE   ;STARTING CHARACTER LOOP ADDRESS
GIN:    MOV     #1,SAVE4
GINA:   RESET
        .SBTTL  INITIALIZE THE COMMON TAGS
        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
                MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
                CLR     (R6)+           ;;CLEAR MEMORY LOCATION
                CMP     #SWR,R6 ;;DONE?
                BNE     .-6             ;;LOOP BACK IF NO
                MOV     #STACK,SP       ;;SETUP THE STACK POINTER
        ;;INITIALIZE A FEW VECTORS
                MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
                MOV     #340,@#IOTVEC+2 ;;LEVEL 7
                MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
                MOV     #340,@#EMTVEC+2 ;;LEVEL 7
                MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
                MOV     #340,@#TRAPVEC+2;LEVEL 7
                MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
                MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
```

```
 920  001544  012737  001544  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 921                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 922                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 923  001552  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
 924  001556  012737  001612  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
 925  001564  012737  177570  001140  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
 926  001572  012737  177570  001142  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
 927  001600  022777  177777  177332  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
 928  001606  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 929                                                          ;;AND  THE HARDWARE SWR IS NOT = -1
 930  001610  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
 931  001612  012716  001620  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
 932  001616  000002                  RTI
 933  001620  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 934  001626  012737  000174  001142          MOV     #DISPREG,DISPLAY
 935  001634  012637  000004  66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
 936
 937  001640  005037  010676          CLR     IGNORE
 938  001644  005037  010660          CLR     LOOP
 939  001650  012737  024700  000020  MOV     #MSCOPE,@#IOTVEC
 940  001656  005737  001330          TST     SAVE4           ;TEST FLAG
 941  001662  001051                  BNE     SBEGIN          ;BR IF NON-ZERO
 942  001664  104401                  TYPE
 943  001666  013704                  TITLE
 944  001670  105777  177244          TSTB    @SWR
 945  001674  100044                  BPL     SBEGIN          ;BR IF CLEARED
 946  001676  012737  001776  000004  MOV     #12$,4          ;SET UP SLAVE TIMEOUT
 947  001704  012737  000340  000006  MOV     #340,6          ;LOCK OUT ANY INTERRUPTS
 948  001712  104401          11$:    TYPE
 949  001714  020713                  WHAT0                   ;FIND OUT THE DE.ICE ADDRESS
 950  001716  104410                  RDOCT
 951  001720  012637  001242          MOV     (SP)+,FIRST     ;SAVE THE ADDRESS
 952  001724  022737  160000  001242  CMP     #160000,FIRST
 953  001732  101367                  BHI     11$             ;BR IF INVALID
 954  001734  005777  177302          TST     @FIRST          ;TEST IF VALID
 955  001740  005037  001244          CLR     LAST
 956  001744  104401                  TYPE
 957  001746  020756                  WHAT1                   ;FIND OUT THE LAST ADDRESS
 958  001750  104410                  RDOCT
 959  001752  012637  001244          MOV     (SP)+,LAST      ;SAVE LAST ADDRESS
 960  001756  005777  177262          TST     @LAST           ;TEST IF VALID
 961  001762  012737  000006  000004  MOV     #6,@#4
 962  001770  005037  000006          CLR     @#6
 963  001774  000404                  BR      SBEGIN
 964  001776  022626          12$:    CMP     (SP)+,(SP)+     ;ADJUST STACK FROM TIMEOUT
 965  002000  104401                  TYPE
 966  002002  016676                  INVLID                  ;INVALID ADDRESS MESSAGE
 967  002004  000742                  BR      11$             ;LET OPERATOR HAVE ANO TRY
 968  002006  013737  001242  001246  SBEGIN: MOV     FIRST,VTNOW     ;LOAD INITIAL DEVICE ADDRESS
 969
 970  002014  012700  001276  RSTRT:  MOV     #VTIS,R0        ;LOAD POINTER
 971  002020  013701  001246          MOV     VTNOW,R1        ;LOAD INPUT STAT
 972  002024  010120                  MOV     R1,(R0)+
 973  002026  005721                  TST     (R1)+
 974  002030  010120                  MOV     R1,(R0)+
 975  002032  005721                  TST     (R1)+
```

```
   976  002034  010120                    MOV     R1,(R0)+
   977  002036  005721                    TST     (R1)+
   978  002040  010110                    MOV     R1,(R0)
```

```
                                        ;;*************************************** **************************
                                        ;*TEST 1        A       TERMINAL IDENTIFICATION TEST
                                        ;;*************************************************************************
002042  000004                          TST1:   SCOPE
                                        ;IDENTIFY TERMINAL TYPE (VT50,VT50H,VT52) INVOKED BY ESC Z (033 132).
                                        ;ESCAPE SEQUENCES TRANSMITTED BY THE VARIOUS MODELS WHEN THEY RECEIVE THIS COMMAND:
                                        ;       VT50    ESC/A   033 057 101
                                        ;       VT50H   ESC/H   033 057 110
                                        ;       VT52    ESC/K   033 057 113
                                        ;       VT55    ESC/C   033 057 103
002044  004537  011454                          JSR     R5,AMSG         ;DISPLAY HEADER
002050  014343                          M914
002052  004537  011454                          JSR     R5,AMSG         ;SEND REQUEST FOR IDENTIFICATION
002056  020312                          RFI                             ;INVOKE ESC Z ESCAPE SEQUENCE TO REQUEST TERMINAL TYPE

002060  004737  013612                          JSR     PC,GETCHR       ;GET A CHARACTER FOUND IN R0 AFTER EXIT
002064  000537                          BR      2$                      ;;BR BACK IF NO INPUT
002066  010037  001330                  MOV     R0,SAVE4        ;SAVE RESPONSE
002072  004737  013612                  JSR     PC,GETCHR       ;GET A CHAR. (2ND CHAR. OF RESPONSE)
002076  000532                          BR      2$                      ;;BR IF NO INPUT
002100  010037  001324                  MOV     R0,SAVE2
002104  004737  013612                  JSR     PC,GETCHR       ;GET A CHAR. (3RD CHAR. OF RESPONSE)
002110  000525                          BR      2$                      ;;BR IF NO INPUT
002112  010037  001326                  MOV     R0,SAVE3
002116  042737  177600  001330          BIC     #177600,SAVE4   ;MASK BIT 7
002124  042737  177600  001324          BIC     #177600,SAVE2
002132  042737  177600  001326          BIC     #177600,SAVE3
002140  005737  001332                  TST     WFTEST
002144  001402                          BEQ     10$
002146  004737  011402                  JSR     PC,ADELAY       ;EXTRA DELAY
002152  122737  000033  001330  10$:    CMPB    #33,SAVE4       ;TEST FIRST CHAR.
002160  001015                          BNE     1$              ;BR TO ERROR
002162  122737  000057  001324          CMPB    #'/,SAVE2       ;TEST SECOND CHAR.
002170  001011                          BNE     1$              ;BR TO ERROR

                                        ;NOW DETERMINE WHICH VT5XX AND ITS CHARACTERISTICS

002172  005000                          CLR     R0              ;CLEAR INITIAL POINTER
002174  126037  002404  001326  3$:     CMPB    TYPEPT(R0),SAVE3        ;TEST I.D. TO KNOWN VALUE
002202  001406                          BEQ     4$              ;BR IF CORRECT
002204  005720                          TST     (R0)+           ;BUMP THE INDEX
002206  105760  002404                  TSTB    TYPEPT(R0)      ;CHECK IF MORE VALID I.D.'S
002212  001370                          BNE     3$              ;BR IF MORE
002214  104003                  1$:     ERROR   3               ;INCORRECT I.D. CODE
002216  005000                  11$:    CLR     R0              ;DEFAULT TO VT50A MODE

                                        ;HAVE NOW FOUND THE I.D. - REPORT TO CONSOLE

002220  016037  002404  001260  4$:     MOV     TYPEPT(R0),VT5XX        ;SAVE I.D. AND CHARACTERISTICS
002226  016037  002436  002242          MOV     MSGTYP(R0),5$   ;SAVE ASCII MESSAGE POINTER
002234  001403                          BEQ     13$             ;BR IF ZERO MESSAGE POINTER
002236  004537  011454                  JSR     R5,AMSG         ;TELL THE UUT
002242  017520                  5$:     VT50A                   ;POINTER TO VT5XX MESSAGE

002244  012737  001700  001264  13$:    MOV     #960.,TOTALC            ;LOAD TOTAL CHARACTER COUNT
```

```
1035  002252  012737  000014  001266          MOV     #12.,VH0                ;LOAD MAX VERTICAL LINE COUNT
1036  002260  012737  000053  001262          MOV     #53,LASTLN              ;LOAD LAST LINE VALUE +40 FOR DCA TESTING
1037  002266  005737  001260                  TST     VT5XX                   ;TEST IF 24 LINES AVAIL
1038  002272  100007                          BPL     6$                      ;BR IF NOT
1039  002274  006337  001264                  ASL     TOTALC                  ;ADJUST CHARACTER COUNT
1040  002300  006337  001266                  ASL     VH0                     ;ADJUST LINE COUNT
1041  002304  062737  000014  001262          ADD     #12.,LASTLN             ;ADJUST VALID LINE # +40
1042
1043  002312  013737  001266  001270  6$:     MOV     VH0,VH1                 ;LOAD OTHER LINE COUNTS
1044  002320  006237  001270                  ASR     VH1
1045  002324  013737  001270  001272          MOV     VH1,VH2
1046  002332  006237  001272                  ASR     VH2
1047  002336  012737  000140  001310          MOV     #140,LASTCH             ;LOAD FIRST NON-VALID CHARACTER
1048  002344  032737  004000  001260          BIT     #BIT11,VT5XX            ;TEST IF UPPER-LOWER CASE TERM.
1049  002352  001403                          BEQ     7$                      ;BR IF NOT
1050  002354  062737  000037  001310          ADD     #37,LASTCH              ;UPDATE TO ALLOW UP-LW CASE CHARACTER SET
1051  002362                          7$:
1052  002362  000403                          BR      12$             ;;BR AND START TESTING
1053
1054  002364  104002                  2$:     ERROR   2                       ;NO RESPONSE FROM UUT AFTER ASKING FOR IDENTIFY
1055  002366  000713                          BR      11$
1056  002370  000240                          NOP
1057  002372  004737  011274          12$:    JSR     PC,DELAY
1058  002376  000177  000000                  JMP     @WHERE
1059  002402  002466                  WHERE:  TST2
1060
1061                                  ;I.D. VALUES AND CHARACTERISTICS
1062                                  ;       BIT15 = 1       24 LINES
1063                                  ;       BIT14 = 1       COPIER CONNECTED
1064                                  ;       BIT13 = 1       DIRECT CURSOR ADDRESSING (ESC Y) + 'ESC-B' + 'ESC-D'
1065                                  ;       BIT12 = 1       KEYPAD
1066                                  ;       BIT11 = 1       UPPER AND LOWER CASE CHARACTERS
1067                                  ;       BIT10 = 1       PRINTER CONNECTED
1068                                  ;       BIT09 = 1       VT52X MODEL
1069                                  ;       VT52    24-LINE, UPPER AND LOWER CASE
1070                                  ;       VT50H   12-LINE, UPPER CASE WITH KEYPAD
1071                                  ;       VT50    12-LINE, UPPER CASE WITHOUT KEYPAD
1072                                  ;       VT55    SAME AS VT50 EXCEPT FOR GRAPHIC CAPABILITIES
1073                                  ;       VT52 I.D.;LOW BYTE CONTAINS THE I.D. FOR EACH KNOWN VT5??
1074
1075  002404  000101                  TYPEPT: .WORD   000101  ;I.D. = 101     ;VT50A
1076  002406  040102                          .WORD   040102  ;I.D. = 102     ;VT50B COPIER
1077  002410  140103                          .WORD   140103  ;I.D. = 103     ;VT55 24. LINES, COPIER(VT50 BASED)
1078  002412  070110                          .WORD   070110  ;I.D. = 110     ;VT50H COPIER DCA VT50H KEYPAD
1079  002414  135113                          .WORD   135113  ;I.D. = 113     ;VT52
1080  002416  175114                          .WORD   175114  ;I.D. = 114     ;VT52 WITH COPIER
1081  002420  137115                          .WORD   137115  ;I.D. = 115     ;VT52 WITH PRINTER.
1082  002422  175105                          .WORD   175105  ;I.D. = 105     ;VT55,24LINES,WITH COPIER(VT52 BASED)
1083  002424  000000                          0
1084  002426  000000                          0
1085  002430  000000                          0
1086  002432  000000                          0
1087  002434  000000                          0
1088
1089                                  ;ASCII MESSAGE POINTERS
1090
```

```
1091  002436  017520                  MSGTYP: VT50A                   ;VT50A NO COPIER
1092  002440  017572                          VT50B                   ;VT50B COPIER
1093  002442  017641                          VT55                    ;VT55 COPIER
1094  002444  017723                          VT50H                   ;VT50H COPIER
1095  002446  020001                          VT52K                   ;VT52
1096  002450  020065                          VT52L                   ;VT52 WITH COPIER
1097  002452  020146                          VT52M                   ;VT52 WITH PRINTER
1098  002454  020227                          VT55E
1099  002456  000000                          0
1100  002460  000000                          0
1101  002462  000000                          0
1102  002464  000000                          0
1103                                  ;;*****************************************************************
1104                                  ;*TEST 2        B       FULL SCREEN OF A CHARACTER
1105                                  ;;*****************************************************************
1106  002466  000004                  TST2:   SCOPE
1107
1108  002470  004537  011454                  JSR     R5,AMSG
1109  002474  013745                          M91
1110
1111  002476  004737  013550                  JSR     PC,FILLWC       ;FILL SCREEN WITH A 'E'S
1112
1113  002502  004737  011274                  JSR     PC,DELAY
1114
1115                                  ;;*****************************************************************
1116                                  ;*TEST 3        C       DATA TRANSFER PATH TEST
1117                                  ;;*****************************************************************
1118  002506  000004                  TST3:   SCOPE
1119  002510  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADING
1120  002514  014013                          M92
1121  002516  013737  001270  001312          MOV     VH1,TEMP        ;SET-UP A COUNTER
1122
1123  002524  004537  011166                  JSR     R5,DTPSR        ;SET-UP BUFFER
1124  002530  000077                          77                      ;OCTAL '?'
1125  002532  000100                          100                     ;OCTAL '@'
1126
1127  002534  004537  011172                  JSR     R5,DTPSRB       ;SET-UP BUFFER
1128  002540  000125                          125                     ;OCTAL 'U'
1129  002542  000052                          52                      ;OCTAL '*'
1130
1131  002544  004737  010246          1$:     JSR     PC,XPRNT        ;DISPLAY THIS LINE
1132  002550  005337  001312                  DEC     TEMP            ;COMPLETED FULL COUNT?
1133  002554  001373                          BNE     1$              ;BRANCH IF NOT COMPLETED
1134
1135  002556  004737  011274                  JSR     PC,DELAY        ;TEST DELAY SWITCH
1136                                  ;;*****************************************************************
1137                                  ;*TEST 4        D       SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
1138                                  ;;*****************************************************************
1139  002562  000004                  TST4:   SCOPE
1140  002564  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADING
1141  002570  014041                          M93
1142  002572  012737  000040  001306          MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
1143
1144  002600  013737  001266  001314          MOV     VH0,TEMP0       ;LOAD COUNT
1145  002606  013701  001306          1$:     MOV     STCHAR,R1       ;LOAD R1= TO CHARACTER
1146  002612  004737  010144                  JSR     PC,FILBUF       ;LOAD A BUFFER WITH THAT CHARACTER
```

```
1147
1148  002616  004737  010246                  JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
1149
1150  002622  005337  001314                  DEC     TEMP0           ;DONE ?
1151  002626  001005                          BNE     2$              ;FINISHED
1152  002630  004737  011274                  JSR     PC,DELAY
1153  002634  013737  001266  001314          MOV     VH0,TEMP0
1154  002642  005237  001306          2$:     INC     STCHAR          ;UPDATE THE CHARACTER
1155  002646  023737  001310  001306          CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
1156  002654  001354                          BNE     1$              ;BRANCH IF NOT COMPLETED
1157
1158  002656  004737  011274                  JSR     PC,DELAY        ;TEST DELAY SWITCH
1159
1160                                  ;;*****************************************************************
1161                                  ;*TEST 5         E       ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
1162                                  ;;*****************************************************************
1163  002662  000004                  TST5:   SCOPE
1164  002664  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADING
1165  002670  014101                          M94
1166  002672  012737  000040  001306          MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
1167
1168  002700  013737  001266  001314          MOV     VH0,TEMP0       ;LOAD TEMP
1169  002706  013701  001306          1$:     MOV     STCHAR,R1       ;LOAD R1=TO CHARACTER
1170  002712  004537  010200                  JSR     R5,LIC          ;LOAD A BUFFER STARTING WITH
1171  002716  001320                          WIDTH                   ; THAT CHARACTER AND WIDTH <BYTE>
1172
1173  002720  004737  010246                  JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
1174
1175  002724  005337  001314                  DEC     TEMP0           ;DONE ?
1176  002730  001005                          BNE     2$              ;BR IF YES
1177  002732  004737  011274                  JSR     PC,DELAY
1178  002736  013737  001266  001314          MOV     VH0,TEMP0
1179  002744  005237  001306          2$:     INC     STCHAR          ;UPDATE THE STARTING CHARACTER
1180  002750  023737  001310  001306          CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
1181  002756  001353                          BNE     1$              ;BRANCH IF NOT COMPLETED
1182
1183  002760  004737  011274                  JSR     PC,DELAY        ;TEST DELAY SWITCH
1184                                  ;;*****************************************************************
1185                                  ;*TEST 6         F       CURSOR MOTION TEST
1186                                  ;;*****************************************************************
1187  002764  000004                  TST6:   SCOPE
1188  002766  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADING
1189  002772  014131                          M96
1190
1191                                  ;ROUTINE TO LOAD REFERENCE LINES FOR CURSOR MOTION TEST
1192
1193  002774  012700  026346          LBMT:   MOV     #BUFFER,R0      ;LOAD POINTER
1194  003000  012720  000065                  MOV     #65.,(R0)+      ;LOAD #5
1195  003004  013701  001270                  MOV     VH1,R1          ;LOAD R1
1196  003010  005301                          DEC     R1
1197  003012  004737  003110                  JSR     PC,MOVDN1       ;MOVE CURSOR DOWN
1198  003016  004737  003126                  JSR     PC,MOVRIG       ;MOVE RIGHT
1199  003022  112720  000062                  MOVB    #62.,(R0)+      ;LOAD #2
1200  003026  004737  003126                  JSR     PC,MOVRIG       ;MOVE RIGHT
1201  003032  004737  003126                  JSR     PC,MOVRIG       ;MOVE RIGHT
1202  003036  112720  000040                  MOVB    #40.,(R0)+
```

```
1203  003042  112720  000061                MOVB    #61,(R0)+       ;LOAD #1
1204  003046  004737  003104                JSR     PC,MOVDWN       ;MOVE DOWN
1205  003052  004737  003126                JSR     PC,MOVRIG       ;MOVE RIGHT
1206  003056  112720  000063                MOVB    #63,(R0)+       ;LOAD #3
1207  003062  004737  003126                JSR     PC,MOVRIG
1208  003066  004737  003126                JSR     PC,MOVRIG       ;MOVE RIGHT
1209  003072  112720  000064                MOVB    #64,(R0)+       ;LOAD #4
1210  003076  112710  000377                MOVB    #377,(R0)       ;TERM
1211  003102  000420                        BR      LBMT1           ;;BR TO NEXT PART
1212
1213  003104  013701  001270        MOVDWN: MOV     VH1,R1
1214  003110  112720  000015        MOVDN1: MOVB    #15,(R0)+       ;LOAD CR
1215  003114  112720  000012                MOVB    #12,(R0)+       ;LOAD LF
1216  003120  005301                        DEC     R1
1217  003122  001372                        BNE     MOVDN1          ;LOOP UNTIL DONE
1218  003124  000207                        RTS     PC              ;ESIT
1219
1220  003126  012701  000024        MOVRIG: MOV     #20.,R1         ;LOAD R1
1221  003132  112720  000040        1$:     MOVB    #40,(R0)+       ;LOAD SPACES
1222  003136  005301                        DEC     R1
1223  003140  100374                        BPL     1$              ;LOOP UNTIL DONE
1224  003142  000207                        RTS     PC              ;EXIT
1225
1226  003144  004737  010246        LBMT1:  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1227  003150  004737  011274                JSR     PC,DELAY        ;TEST DELAY SWITCH
1228
1229                                ;CURSOR MOTION SUBROUTINE
1230                                ;IF VT50H TYPE - USE 'ESC-D'' FOR CURSOR LEFT AND USE 'ESC-B'' FOR CURSOR DOWN
1231  003154  013701  001270        LCM:    MOV     VH1,R1          ;LOAD COUNT
1232  003160  012700  026346                MOV     #BUFFER,R0      ;LOAD BUFFER POINTER
1233  003164  112720  000033        1$:     MOVB    #33,(R0)+       ;LOAD 'ESC'
1234  003170  112720  000101                MOVB    #101,(R0)+      ;LOAD 'A' CURSOR UP
1235  003174  005301                        DEC     R1
1236  003176  001372                        BNE     1$              ;LOOP UNTIL DONE
1237  003200  112720  000130                MOVB    #130,(R0)+      ;LOAD 'X'
1238  003204  012701  000054                MOV     #44.,R1         ;LOAD COUNT
1239  003210  032737  020000  001260        BIT     #BIT13,VT5XX    ;TEST IF VT50H TYPE
1240  003216  001407                        BEQ     20$             ;BR IF NOT
1241  003220  112720  000033        2$:     MOVB    #33,(R0)+       ;LOAD 'ESC'
1242  003224  112720  000104                MOVB    #104,(R0)+      ;LOAD 'CURSOR LEFT'
1243  003230  005301                        DEC     R1
1244  003232  100372                        BPL     2$              ;LOOP UNTIL DONE
1245  003234  000404                        BR      21$
1246  003236  112720  000010        20$:    MOVB    #10,(R0)+       ;LOAD 'BACKSPACE
1247  003242  005301                        DEC     R1              ;DONE ALL ?
1248  003244  100374                        BPL     20$             ;BR IF NOT
1249  003246  112720  000130        21$:    MOVB    #130,(R0)+      ;LOAD 'X'
1250  003252  112720  000010                MOVB    #10,(R0)+       ;LOAD BACKSPACE
1251  003256  013701  001270                MOV     VH1,R1          ;LOAD COUNT
1252  003262  032737  020000  001260        BIT     #BIT13,VT5XX    ;TEST IF VT50H TYPE
1253  003270  001407                        BEQ     30$             ;BR IF NOT
1254  003272  112720  000033        3$:     MOVB    #33,(R0)+       ;LOAD 'ESC' CURSOR DOWN
1255  003276  112720  000102                MOVB    #102,(R0)+      ;
1256  003302  005301                        DEC     R1
1257  003304  001372                        BNE     3$              ;LOOP UNTIL DONE
1258  003306  000404                        BR      31$
```

```
1259  003310  112720  000012          30$:    MOVB    #12,(R0)+       ;LOAD CURSOR DOWN (LF)
1260  003314  005301                          DEC     R1              ;DONE ?
1261  003316  001374                          BNE     30$             ;BR IF NOT
1262  003320  112720  000130          31$:    MOVB    #130,(R0)+      ;LOAD 'X'
1263  003324  112720  000010                  MOVB    #10,(R0)+       ;LOAD BACKSPACE
1264  003330  012701  000052                  MOV     #42.,R1         ;LOAD COUNT
1265  003334  112720  000033          4$:     MOVB    #33,(R0)+       ;LOAD 'ESC'
1266  003340  112720  000103                  MOVB    #103,(R0)+      ;LOAD 'C' CURSOR RIGHT
1267  003344  005301                          DEC     R1
1268  003346  100372                          BPL     4$              ;LOOP UNTIL DONE
1269  003350  112720  000130                  MOVB    #130,(R0)+      ;LOAD 'X'
1270  003354  112720  000033                  MOVB    #33,(R0)+       ;LOAD 'ESC'
1271  003360  112720  000110                  MOVB    #110,(R0)+      ;LOAD 'H' CURSOR HOME
1272  003364  112720  000130                  MOVB    #130,(R0)+
1273  003370  112720  000377                  MOVB    #377,(R0)+
1274  003374  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1275  003400  004737  011274                  JSR     PC,DELAY        ;DELAY
1276  003404  004537  011454                  JSR     R5,AMSG         ;
1277  003410  017223                          CRLFB
1278  003412  005737  001260                  TST     VT5XX           ;TEST IF 24 LINES
1279  003416  100003                          BPL     TST7            ;;BR IF 12 LINES
1280  003420  004537  011454                  JSR     R5,AMSG         ;SCROLL MORE LINES
1281  003424  017223                          CRLFB
1282
1283                                  ;;*******************************************************************
1284                                  ;*TEST 7         G       TAB, BACKSPACE AND BELL TEST
1285                                  ;;*******************************************************************
1286  003426  000004                  TST7:   SCOPE
1287  003430  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADING
1288  003434  014163                          M97
1289
1290  003436  012700  026346          LRL:    MOV     #BUFFER,R0      ;LOAD BUFFER POINTER
1291  003442  112720  000007                  MOVB    #7,(R0)+        ;LOAD 'BELL'
1292  003446  012701  000011                  MOV     #9.,R1          ;LOAD LINE COUNT
1293  003452  012702  000006          1$:     MOV     #6.,R2          ;LOAD COLUMN COUNT
1294  003456  112720  000111                  MOVB    #111,(R0)+      ;LOAD COLUMN MARK
1295  003462  112720  000040          2$:     MOVB    #40,(R0)+       ;LOAD SPACE
1296  003466  005302                          DEC     R2              ;
1297  003470  100374                          BPL     2$              ;BR UNTIL DONE
1298  003472  005301                          DEC     R1              ;
1299  003474  100366                          BPL     1$              ;BR UNTIL FINISHED ALL TAB STOPS
1300  003476  112720  000015                  MOVB    #15,(R0)+       ;LOAD CR
1301  003502  112720  000012                  MOVB    #12,(R0)+       ;LOAD LF
1302  003506  112720  000377                  MOVB    #377,(R0)+      ;LOAD TERM
1303
1304  003512  004737  010246                  JSR     PC,XPRNT
1305
1306  003516  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #1
1307  003522  000000                          0
1308  003524  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1309
1310  003530  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #2
1311  003534  000001                          1
1312  003536  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1313
1314  003542  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #3
```

```
1315  003546  000002                          2
1316  003550  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1317
1318  003554  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #4
1319  003560  000003                          3
1320  003562  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1321
1322  003566  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #5
1323  003572  000004                          4
1324  003574  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1325
1326  003600  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #6
1327  003604  000005                          5
1328  003606  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1329
1330  003612  004537  003706                  JSR     R5,LTAB         ;LOAD TAB LINE #7
1331  003616  000006                          6
1332  003620  004737  010246                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
1333                                  ;NOW EXECUTE THE BACKSPACE SECTION
1334
1335  003624  013702  001320                  MOV     WIDTH,R2        ;LOAD WIDTH COUNT
1336  003630  005302                          DEC     R2              ;ADJUST WIDTH
1337  003632  005302                          DEC     R2              ;  BY 2
1338  003634  012701  000040                  MOV     #40,R1          ;LOAD ''SPACE'' INTO THE LINE
1339  003640  004737  010150                  JSR     PC,FILBFB       ;LOAD LINE
1340  003644  013701  001320                  MOV     WIDTH,R1        ;LOAD # OF CHARACTER POSITIONS
1341  003650  112720  000130          3$:     MOVB    #'X,(R0)+       ;LOAD ASCII 'X''
1342  003654  112720  000010                  MOVB    #10,(R0)+       ;LOAD BACKSPACE CODE
1343  003660  112720  000010                  MOVB    #10,(R0)+
1344  003664  005301                          DEC     R1              ;DONE ALL POSITIONS ?
1345  003666  001370                          BNE     3$              ;BR IF NOT
1346  003670  112720  000377                  MOVB    #377,(R0)+      ;LOAD TERM.
1347  003674  004737  010246                  JSR     PC,XPRNT        ;EXECUTE ASCII CODE
1348  003700  004737  011274                  JSR     PC,DELAY        ;TEST DELAY SWITCH
1349  003704  000434                          BR      TST10           ;;BR TO NEXT TEST
1350
1351                                  ;SUBROUTINE TO LOAD THE TAB TEST INTO THE BUFFER
1352
1353  003706  012537  001322          LTAB:   MOV     (R5)+,SAVE1     ;LOAD # OF CHARACTERS
1354  003712  012702  026346                  MOV     #BUFFER,R2      ;LOAD BUFFER POINTER
1355  003716  012701  000011                  MOV     #9.,R1          ;LOAD WIDTH COUNTER
1356  003722  112722  000007                  MOVB    #7,(R2)+        ;LOAD BELL AT START OF LINE
1357  003726  112722  000111          3$:     MOVB    #111,(R2)+      ;LOAD 'I
1358  003732  013700  001322                  MOV     SAVE1,R0        ;GET THE  NO. OF THE CHAR
1359  003736  001404                          BEQ     1$              ;BR IF 0
1360  003740  112722  000101          2$:     MOVB    #101,(R2)+      ;LOAD THE 'A' CHAR.
1361  003744  005300                          DEC     R0              ;
1362  003746  001374                          BNE     2$              ;LOOP UNTIL DONE
1363  003750  112722  000011          1$:     MOVB    #11,(R2)+       ;LOAD 'TAB' CHAR
1364  003754  005301                          DEC     R1              ;
1365  003756  100363                          BPL     3$              ;BR TO NEXT TAB COLUMN CHAR
1366  003760  112722  000015                  MOVB    #15,(R2)+       ;LOAD 'CR'
1367  003764  112722  000012                  MOVB    #12,(R2)+       ;LOAD 'LF'
1368  003770  112722  000377                  MOVB    #377,(R2)+      ;LOAD TERM
1369  003774  000205                          RTS     R5              ;EXIT
1370                                  ;;**************************************************************************
```

```
1371                                        ;*TEST 10       H       ERASE FROM CURSOR TO END OF LINE
1372                                        ;;*****************************************************************
1373  003776  000004                        TST10:  SCOPE
1374  004000  004537  011454                        JSR     R5,AMSG         ;DISPLAY HEADING
1375  004004  014230                                M910
1376
1377  004006  004737  013550                        JSR     PC,FILLWC       ;FILL BUFFER WITH A CHAR
1378                                                                        ;DISPLAY THIS LINE
1379  004012  004737  011274                        JSR     PC,DELAY        ;DELAY
1380
1381                                        ;LOAD ERASE LINE TEST INTO BUFFER
1382
1383  004016  012700  026346                LODERL: MOV     #BUFFER,R0
1384  004022  112720  000033                        MOVB    #33,(R0)+       ;LOAD ESC
1385  004026  112720  000110                        MOVB    #'H,(R0)+       ;LOAD HOME
1386  004032  013737  001266  004134                MOV     VH0,2$          ;LOAD COUNT
1387  004040  005337  004134                        DEC     2$
1388  004044  112720  000033                3$:     MOVB    #33,(R0)+       ;LOAD ESC
1389  004050  112720  000113                        MOVB    # K,(R0)+       ;LOAD ERASE LINE CHAR
1390  004054  005337  004134                        DEC     2$              ;FINISHED ?
1391  004060  100427                                BMI     1$              ;BR WHEN DONE
1392  004062  012737  000006  004136                MOV     #6,,10$         ;LOAD COUNT
1393  004070  005737  001260                        TST     VT5XX           ;TEST IF 12 LINES
1394  004074  100005                                BPL     4$              ;BR IF 12 LINES
1395  004076  006237  004136                        ASR     10$             ;ADJUST HORIZ. COUNT
1396  004102  000240                                NOP
1397  004104  000240                                NOP
1398  004106  000240                                NOP
1399  004110  112720  000033                4$:     MOVB    #33,(R0)+
1400  004114  112720  000103                        MOVB    #'C,(R0)+       ;CURSOR RIGHT
1401  004120  005337  004136                        DEC     10$             ;
1402  004124  001371                                BNE     4$              ;LOOP
1403  004126  112720  000012                        MOVB    #12,(R0)+       ;CURSOR DOWN
1404  004132  000744                                BR      3$
1405  004134  000000                2$:     0
1406  004136  000000                10$:    0
1407  004140  112720  000377                1$:     MOVB    #377,(R0)+      ;LOAD TERM
1408
1409
1410  004144  004737  010246                        JSR     PC,XPRNT        ;DISPLAY THIS TEST
1411
1412  004150  004737  011274                        JSR     PC,DELAY        ;TEST DELAY SWITCH
1413
1414                                        ;;*****************************************************************
1415                                        ;*TEST 11       I       ERASE FROM CURSOR TO END OF SCREEN
1416                                        ;;*****************************************************************
1417  004154  000004                        TST11:  SCOPE
1418  004156  004537  011454                        JSR     R5,AMSG         ;DISPLAY HEADING
1419  004162  014257                                M911
1420  004164  004737  013550                        JSR     PC,FILLWC       ;FILL BUFFER WITH A CHAR
1421                                                                        ;DISPLAY THIS LINE
1422  004170  004737  011274                        JSR     PC,DELAY        ;DELAY
1423
1424                                        ;LOAD ERASE SCREEN TEST INTO BUFFER
1425
1426  00417?  013737  001320  004270        LODERS: MOV     WIDTH,2$        ;LOAD COUNT
```

```
1427  004202  006237  004270                    ASR     2$
1428  004206  012700  026346                    MOV     #BUFFER,R0
1429  004212  112720  000010            1$:     MOVB    #10,(R0)+       ;LOAD CURSOR LEFT
1430  004216  005337  004270                    DEC     2$              ;DONE ?
1431  004222  100373                            BPL     1$              ;BR UNTIL DONE
1432  004224  013737  001266  004270            MOV     VH0,2$          ;LOAD COUNT
1433  004232  112720  000033            4$:     MOVB    #33,(R0)+       ;LOAD ESC
1434  004236  112720  000112                    MOVB    #'J,(R0)+       ;LOAD ERASE SCREEN
1435  004242  005337  004270                    DEC     2$              ;DONE ?
1436  004246  100405                            BMI     3$              ;BR WHEN DONE
1437  004250  112720  000033                    MOVB    #33,(R0)+       ;LOAD ESC
1438  004254  112720  000101                    MOVB    #'A,(R0)+       ;LOAD CURSOR UP
1439  004260  000764                            BR      4$              ;LOOP
1440  004262  112720  000377            3$:     MOVB    #377,(R0)+      ;LOAD TERM
1441  004266  000401                            BR      5$
1442  004270  000000            2$:     0
1443
1444  004272  004737  010246            5$:     JSR     PC,XPRNT        ;DISPLAY THIS TEST
1445  004276  004737  011274                    JSR     PC,DELAY        ;TEST DELAY SWITCH
1446
1447                                    ;;**********************************************************************
1448                                    ;*TEST 12        J       VIDEO COUPLING TEST
1449                                    ;;**********************************************************************
1450  004302  000004                    TST12:  SCOPE
1451  004304  004537  011454                    JSR     R5,AMSG         ;DISPLAY HEADER
1452  004310  014310                            M912
1453  004312  013737  001270  001312            MOV     VH1,TEMP        ;LOAD COUNTER
1454  004320  004537  011454            1$:     JSR     R5,AMSG
1455  004324  017251                            PATH
1456
1457  004326  005337  001312                    DEC     TEMP            ;FINISHED COUNT ?
1458  004332  001372                            BNE     1$              ;BR UNTIL DONE
1459  004334  004737  011274                    JSR     PC,DELAY        ;TEST DELAY SWITCH
1460
1461                                    ;;**********************************************************************
1462                                    ;*TEST 13        K       DIRECT CURSOR ADDRESS TEST
1463                                    ;;**********************************************************************
1464  004340  000004                    TST13:  SCOPE
1465                                    ;       RANDOM NUMBERS ARE GENERATED AND USED AS 'X'' AND 'Y'' COORDINATES
1466                                    ;ADDRESSING A 960 CHARACTER PRINTOUT.
1467                                    ;VERFICATION OF DISPLAY IS PERFORMED VISUALLY BY THE USER
1468                                    ;EXECUTE 1ST TIME USING 'ESC-Y'' SEQUENCE AND IF I.D. IS AN 'H''
1469                                    ;       EXECUTE 2ND TIME USING ''CODE 16'' SEQUENCE
1470
1471  004342  032737  020000  001260            BIT     #BIT13,VT5XX    ;TEST IF DCA TYPE TERMINAL
1472  004350  001002                            BNE     3$              ;;BR IF CURSOR ADDRESSING
1473  004352  000137  005100            1$:     JMP     TST14           ;BYPASS THIS TEST
1474  004356  005737  001332            3$:     TST     WFTEST          ;TEST IF IN W.F. MODE
1475  004362  001373                            BNE     1$              ;BYPASS IF W.F. MODE
1476  004364  004537  011454                    JSR     R5,AMSG         ;ERASE SCREEN AND IDENTIFY TEST
1477  004370  020650                            M98
1478  004372  112737  000033  026346            MOVB    #33,BUFFER      ;LOAD 'ESC'' CODE
1479  004400  112737  000131  026347            MOVB    #'Y,BUFFER+1    ;LOAD ASCII 'Y'' (D.C.A. ENABLE)
1480  004406  004737  004462                    JSR     PC,DCATST       ;RUN TEST WITH ESC Y SEQUENCE
1481  004412  004737  011274                    JSR     PC,DELAY        ;DELAY TESTING
1482  004416  122737  000110  001260            CMPB    #'H,VT5XX       ;TEST IF VT50H TYPE DISPLAY
```

```
1483  004424  001352                          BNE     1$              ;;BR IF NOT VT50H TYPE
1484  004426  004537  011454                  JSR     R5,AMSG         ;ERASE SCREEN AND IDENTIFY TEST
1485  004432  020650                          M98
1486  004434  112737  000000  026346          MOVB    #0,BUFFER       ;LOAD ''SPACE'' AS FIRST VALUE
1487  004442  112737  000016  026347          MOVB    #16,BUFFER+1    ;LOAD CODE-16 SEQUENCE
1488  004450  004737  004462                  JSR     PC,DCATST       ;RUN TEST WITH CODE 16 SEQUENCE
1489  004454  004737  011274                  JSR     PC,DELAY
1490  004460  000734                          BR      1$              ;;BR TO NEXT TEST
1491
1492  004462  012737  123456  026046  DCATST: MOV     #123456,$LONUM  ;PRIME RANDOM NUMBER GENERATOR
1493  004470  012737  176543  026044          MOV     #176543,$HINUM
1494  004476  013737  001264  004754          MOV     TOTALC,OVRAL    ;LOAD CHAR COUNT
1495  004504  013737  001266  004752          MOV     VH0,SET         ;LOAD LINE COUNT
1496  004512  012700  026366                  MOV     #BUFFER+20,R0   ;LOAD DESTINATION BUFFER
1497  004516  012701  004756          2$:     MOV     #MSGTXT,R1      ;LOAD MESSAGE POINTER
1498  004522  012120                  1$:     MOV     (R1)+,(R0)+     ;LOAD 2 CHARACTERS
1499  004524  022701  005076                  CMP     #MSGTND,R1      ;TEST FOR LAST CHAR
1500  004530  001374                          BNE     1$              ;BR UNTIL DONE 1 LINE
1501  004532  005337  00475?                  DEC     SET             ;FINISHED ALL LINES
1502  004536  001367                          BNE     2$              ;BR IF NOT
1503  004540  012737  177777  026352          MOV     #-1,BUFFER+4    ;LOAD MESSAGE TERMINATOR
1504
1505  004546  004737  025746          GE'NER: JSR     %7,$RAND        ;GENERATE RANDOM NUMBER
1506  004552  013700  026044                  MOV     $HINUM,R0       ;GET RNADOM NUMBER
1507  004556  042700  177700                  BIC     #177700,%0      ;RANDOM NO. MUST BE TWO DIGITS
1508  004562  020027  000037                  CMP     %0,#37          ;NO. MUST BE LESS THAN 40
1509  004566  101767                          BLOS    GENER           ;LOWER, REGENERATION
1510  004570  020037  001262                  CMP     %0,LASTLN               ;NO. MUST NOT BE GREA'ER THAN 54 OR 67
1511  004574  101364                          BHI     GENER           ;GREATER, REGENERATION
1512  004576  010037  004746                  MOV     %0,YADDS        ;STORE RANDOM Y COORDINATE
1513  004602  010001                          MOV     %0,%1           ;COPY DATA
1514  004604  012737  026366  004752          MOV     #BUFFER+20,SET  ;LOAD BASE POINTER
1515  004612  162701  000040                  SUB     #40,%1          ;MINIMUM Y INDEX
1516  004616  001405                          BEQ     GENRX           ;RESULT, MINIMUM Y COORDINATE
1517  004620  062737  000120  004752  1$:     ADD     #80.,SET        ;SETUP  Y INDEX LOCATION FOR PRINTOU'
1518  004626  005301                          DEC     %1
1519  004630  001373                          BNE     1$              ;Y COORDINATE IS SET
1520  004632  004737  025746          GENRX:  JSR     %7,$RAND        ;GENERATE RANDOM NUMBER
1521  004636  013700  026044                  MOV     $HINUM,R0       ;GET A RANDOM NUMBER
1522  004642  042700  177600                  BIC     #177600,%0      ;RANDOM NO. MAY BE LESS THAN 200
1523  004646  020027  000037                  CMP     %0,#37          ; MUST NOT BE LESS THAN 40
1524  004652  101767                          BLOS    GENRX           ;LOWER, REGENERATION
1525  004654  020027  000157                  CMP     %0,#157         ; MUST NOT BE GREATER THAN 157
1526  004660  101364                          BHI     GENRX           ;GREATER, REGENERATION
1527  004662  010037  004750                  MOV     %0,XADDS        ;STORE RANDOM X COORDINATE
1528  004666  162700  000040                  SUB     #40,%0          ;SETUP MINIMUM X INDEX
1529  004672  060037  004752                  ADD     %0,SET          ;SETUP X COOR, FOR PNTOUT.
'530  004676  013701  004752                  MOV     SET,%1          ;SETUP CHECK
1531  004702  105711                          TSTB    (1)             ;HAS CURRENT CHAR, ALREADY BEEN USED?
1532  004704  100720                          BMI     GENER           ;YES, REGENERATE
1533  004706  113737  004746  026350          MOVB    YADDS,BUFFER+2  ;LOAD Y COORDINATE
1534  004714  113737  004750  026351          MOVB    XADDS,BUFFER+3  ;LOAD X COORDINATE
1535  004722  111137  026352                  MOV'3   (1),BUFFER+4    ;LOAD CHARACTER TO BE PRINTED
1536  004726  152711  000200                  BISB    #200,(1)        ;INDICATE USE OF CURSOR POSITION
1537  004732  004737  010246                  JSR     PC,XPRNT        ;EXECUTE AND PRINT CHARACTER
1538  004736  005537  004754                  DEC     OVRAL           ;MAXIMUM NO. OF COORDINATES
```

```
1539  004742  001301                        BNE     GENER           ;BR BACK UNTIL DONE
1540  004744  000207                        RTS     PC              ;EXIT
1541
1542  004746  000000                YADDS:  0
1543  004750  000000                XADDS:  0
1544  004752  000000                SET:    0
1545  004754  000000                OVRAL:  0
1546  004756  052126  030065  050055 MSGTXT: .ASCII  \VT50-PLUS-DIRECT-CURSOR-ADDRESSING-TEST\
1547  004764  052514  026523  044504
1548  004772  042522  052103  041455
1549  005000  051125  047523  026522
1550  005006  042101  051104  051505
1551  005014  044523  043516  052055
1552  005022  051505     124
1553  005025     055  044504  044507        .ASCII  \-DIGITAL-EQUIPMENT-CORP.-MAYNARD-MA.-VT50\
1554  005032  040524  026514  050505
1555  005040  044525  046520  047105
1556  005046  026524  047503  050122
1557  005054  026456  040515  047131
1558  005062  051101  026504  040515
1559  005070  026456  052126  030065
1560  005076  100000                MSGTND: BIT15
1561                                ;ONLY 12 LINE TERMINALS WILL RUN THIS TEST
1562                                ;;**************************************************************
1563                                ;*TEST 14       L       HOLD SCREEN TEST
1564                                ;;**************************************************************
1565  005100  000004                TST14:  SCOPE
1566  005102  005037  010c66               CLR     AXOFF
1567  005106  005037  010670               CLR     XOFFRC          ;CLEAR SOFT FLAG
1568  005112  004537  011454               JSR     R5,AMSG         ;DISPLAY
1569  005116  017223                       CRLFB
1570  005120  004537  011454               JSR     R5,AMSG         ;CR-LF
1571  005124  017223                       CRLFB
1572
1573  005126  005737  001332               TST     WFTEST          ;TEST IF IN W.F. MODE
1574  005132  001046                       BNE     TST15           ;;BR IF W.F. MODE
1575  005134  005737  001260               TST     VT5XX           ;TEST IF 12 LINE
1576  005140  100443                       BMI     TST15           ;;BR IF NOT 12 LINE
1577  005142  004537  011454               JSR     R5,AMSG         ;DISPLAY MESSAGE
1578  005146  014472                       M922
1579
1580  005150  012737  000001  010664       MOV     #1,XOFFOK
1581  005156  012737  000001  010662       MOV     #1,XOFFBR       ;ENABLE XOFF
1582  005164  004537  011454               JSR     R5,AMSG         ;TRY TO  SCROLL THE SCREEN
1583  005170  020540                       GOHDSC                  ;ENABLE HOLD SCREEN
1584
1585  005172  005737  010670               TST     XOFFRC          ;TEST IF XOFF SENSED
1586  005176  001001                       BNE     1$              ;BR IF SENSED
1587
1588  005200  104002                       ERROR   2               ;ENABLE HOLD SCREEN MODE FAILED TO
1589                                                                ;INHIBIT THE SCREEN FROM SCROLLING
1590                                                                ;BY SENDING 'X-OFF''
1591
1592  005202  005037  010672        1$:    CLR     XONRC           ;CLEAR SOFT FLAG
1593  005206  005037  010662               CLR     XOFFBR
1594  005212  012737  000001  010666       MOV     #1,AXOFF
```

```
1595  005220  004537  011454                JSR     R5,AMSG
1596  005224  020566                        GODSHS                  ;DISABLE HOLD SCREEN MODE
1597
1598  005226  004537  011454                JSR     R5,AMSG
1599  005232  017234                        CRLFA                   ;TRY SCROLLING THE SCREEN
1600
1601  005234  005737  010672                TST     XONRC           ;TEST SOFT FLAG (X-ON)
1602  005240  001001                        BNE     2$              ;BR IF SENSED
1603
1604  005242  104002                        ERROR   2               ;DISABLE HOLD SCREEN MODE FAILED TO ENABLE
1605                                                                ;THE SCREEN TO SCROLL BY SENDING AN 'X-ON'
1606
1607  005244  004737  011274        2$:     JSR     PC,DELAY        ;PROGRAM DELAY
1608
1609                                ;;*******************************************************************
1610                                ;*TEST 15       M       TEST GRAPHICS MODE AND REV. LINE FEED
1611                                ;;*******************************************************************
1612  005250  000004                TST15:  SCOPE
1613  005252  032737  001000  001260        BIT     #BIT09,VT5XX    ;IS UNIT VT52?
1614  005260  001002                        BNE     GRPHST          ;YES-TEST GRAPHICS AND REV. LINE FEED
1615  005262  000137  005362                JMP     COPTST          ;NO-GO CHECK FOR COPIER
1616  005266  012704  000120        GRPHST: MOV     #80.,R4
1617  005272  004537  011454        1$:     JSR     R5,AMSG         ;HOME THE CURSOR AND CLEAR THE SCREEN.
1618  005276  017243                        HOMERS
1619  005300  004537  011454                JSR     R5,AMSG         ;DISPLAY TEST MESSAGE
1620  005304  014525                        M9221
1621  005306  004537  011454                JSR     R5,AMSG         ;ENTER GRAPHICS MODE.
1622  005312  022313                        ENGRAF
1623  005314  012737  000045  001314        MOV     #37.,TEMPO      ;SET UP TC XMIT 80 LINES
1624  005322  004737  011232        2$:     JSR     PC,GBBUF        ;LOAD BUFFER WITH GRAPHICS
1625  005326  004737  010246                JSR     PC,XPRNT        ;DISPLAY 1 LINE
1626  005332  004537  011454                JSR     R5,AMSG         ;ISSUE REV. LINE FEED AND
1627  005336  022274                        REVLF                   ;A CARRIAGE RETURN
1628  005340  005304                        DEC     R4              ;DECREMENT BUFFER COUNT
1629  005342  005337  001314        3$:     DEC     TEMPO           ;DONE?
1630  005346  001365                        BNE     2$              ;NO-LOOP
1631  005350  004537  011454                JSR     R5,AMSG         ;YES-EXIT GRAPHICS MODE
1632  005354  022320                        EXGRAF
1633  005356  004737  011274                JSR     PC,DELAY        ;AND GO CHECK DELAY
1634  005362                        COPTST:
1635                                ;;*******************************************************************
1636                                ;*TEST 16       N       COPIER - AUTO COPY TEST
1637                                ;;*******************************************************************
1638  005362  000004                TST16:  SCOPE
1639  005364  032737  040000  001260        BIT     #BIT14,VT5XX    ;TEST IF COPIER IS AVAILABLE
1640  005372  001002                        BNE     2$              ;;BR IF AVAILABLE
1641  005374  000137  006342        3$:     JMP     PRNTST          ;NOT AVAILABLE SO BYPASS COPIER TESTS
1642  005400  032777  004000  173532  2$:     BIT     #BIT11,@SWR     ;TEST IF INHIBIT COPIER TEST SWITCH IS SET
1643  005406  001372                        BNE     3$              ;BR IF SET
1644  005410  032777  002000  173522        BIT     #BIT10,@SWR     ;TEST IF PAPER SAVE SWITCH IS SET
1645  005416  001406                        BEQ     6$              ;BR IF NOT SET AND START COPIER TESTING
1646  005420  105737  001274                TSTB    PRTCNT          ;TEST IF TIME TO TEST COPIER
1647  005424  001363                        BNE     3$              ;NOT TIME TO RUN THE COPIER
1648  005426  013737  001250  001274        MOV     PTCT,PRTCNT     ;RELOAD COUNTER AND TEST COPIER
1649
1650  005434  004537  011454        6$:     JSR     R5,AMSG
```

```
1651  005440  014600                          M923                    ;DISPLAY HEADER
1652  005442  004537  011454                  JSR     R5,AMSG         ;ENABLE AUTO-COPY
1653  005446  020614                          GOAPMD
1654
1655  005450  013737  001266  001314          MOV     VH0,TEMP0       ;LOAD A EXECUTION COUNT
1656  005456  012701  000101                  MOV     #101,R1         ;LOAD A CHARACTER
1657  005462  004737  010144                  JSR     PC,FILBUF       ;LOAD CHARACTER INTO BUFFER
1658  005466  012737  000001  010664  1$:     MOV     #1,XOFFOK
1659  005474  004737  010246                  JSR     PC,XPRNT        ;DISPLAY IT
1660  005500  004737  011274                  JSR     PC,DELAY        ;PROGRAM DELAY
1661  005504  005337  001314                  DEC     TEMP0           ;FINISHED ?
1662  005510  100366                          BPL     1$              ;BR IF NOT
1663  005512  012737  000001  010664          MOV     #1,XOFFOK
1664  005520  004537  011454                  JSR     R5,AMSG         ;DISABLE AUTO-COPY
1665  005524  020621                          GONAPM
1666  005526  004737  011274                  JSR     PC,DELAY        ;ANOTHER DELAY
1667
1668                                  ;;*****************************************************************
1669                                  ;*TEST 17        O       COPIER - FULL SCREEN OF A CHARACTER
1670                                  ;;*****************************************************************
1671  005532  000004                  ST17:   SCOPE
1672  005534  004537  011454          1$:     JSR     R5,AMSG         ;DISPLAY HEADER
1673  005540  013745                          M91
1674  005542  004737  013550                  JSR     PC,FILLWC       ;FILL BUFFER WITH CHAR
1675                                                                  ;DISPLAY IT
1676  005546  012737  000001  010664          MOV     #1,XOFFOK       ;ALLOW XOFF
1677  005554  004537  011454                  JSR     R5,AMSG         ;COPY INST
1678  005560  020512                          GOPRNT
1679  005562  004737  011274                  JSR     PC,DELAY
1680
1681
1682                                  ;;*****************************************************************
1683                                  ;*TEST 20        P       COPIER - DATA PATH TEST
1684                                  ;;*****************************************************************
1685  005566  000004                  TST20:  SCOPE
1686  005570  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADING
1687  005574  014013                          M92
1688  005576  013737  001270  001312          MOV     VH1,TEMP        ;SET-UP A COUNTER
1689  005604  004537  011166                  JSR     R5,DTPSR        ;SET-UP BUFFER
1690  005610  000077                          77                      ;OCTAL '?'
1691  005612  000100                          100                     ;OCTAL '@'
1692  005614  004537  011172                  JSR     R5,DTPSRB       ;SET-UP BUFFER
1693  005620  000125                          125                     ;OCTAL 'U'
1694  005622  000052                          52                      ;OCTAL '*'
1695  005624  004737  010246          1$:     JSR     PC,XPRNT        ;DISPLAY THIS LINE
1696  005630  005337  001312                  DEC     TEMP            ;COMPLETED FULL COUNT?
1697  005634  001373                          BNE     1$              ;BRANCH IF NOT COMPLETED
1698  005636  012737  000001  010664          MOV     #1,XOFFOK
1699  005644  004537  011454                  JSR     R5,AMSG
1700  005650  020512                          GOPRNT                  ;COPY INST
1701  005652  004737  011274                  JSR     PC,DELAY        ;TEST DELAY SWITCH
1702
1703                                  ;;*****************************************************************
1704                                  ;*TEST 21        Q       COPIER - SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
1705                                  ;;*****************************************************************
1706  005656  000004                  TST21:  SCOPE
```

```
1707  005660  004537  011454            JSR     R5,AMSG         ;DISPLAY HEADING
1708  005664  014041                    M93
1709  005666  012737  000040  001306    MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
1710  005674  013737  001266  001314    MOV     VH0,TEMP0
1711  005702  013701  001306      1$:   MOV     STCHAR,R1       ;LOAD R1= TO CHARACTER
1712  005706  004737  010144            JSR     PC,FILBUF       ;LOAD A BUFFER WITH THAT CHARACTER
1713  005712  004737  010246            JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
1714  005716  005337  001314            DEC     TEMP0           ;DONE
1715  005722  001011                    BNE     3$
1716  005724  013737  001266  001314    MOV     VH0,TEMP0
1717  005732  012737  000001  010664    MOV     #1,XOFFOK       ;ALLOW XOFF
1718  005740  004537  011454            JSR     R5,AMSG
1719  005744  020512                    GOPRNT                  ;COPY INST
1720  005746  005237  001306      3$:   INC     STCHAR          ;UPDATE THE CHARACTER
1721  005752  023737  001310  001306    CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
1722  005760  001350                    BNE     1$              ;BRANCH IF NOT COMPLETED
1723  005762  004537  011454            JSR     R5,AMSG         ;CRLF
1724  005766  017232                    CRLFA-2
1725  005770  012737  000001  010664    MOV     #1,XOFFOK       ;ENABLE XOFF
1726  005776  004537  011454            JSR     R5,AMSG         ;COPY INS
1727  006002  020512                    GOPRNT
1728  006004  004737  011274            JSR     PC,DELAY        ;TEST DELAY SWITCH
1729
1730                                ;;*****************************************************************
1731                                ;*TEST 22       R       COPIER - ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
1732                                ;;*****************************************************************
1733  006010  000004            TST22:  SCOPE
1734  006012  004537  011454            JSR     R5,AMSG         ;DISPLAY HEADING
1735  006016  014101                    M94
1736  006020  012737  000040  001306    MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
1737  006026  013737  001266  001314    MOV     VH0,TEMP0       ;
1738  006034  013701  001306      1$:   MOV     STCHAR,R1       ;LOAD R1=TO CHARACTER
1739  006040  004537  010200            JSR     R5,LIC          ;LOAD A BUFFER STARTING WITH
1740  006044  001320                    WIDTH                   ; THAT CHARACTER AND WIDTH <BYTE>
1741  006046  004737  010246            JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
1742  006052  005337  001314            DEC     TEMP0           ;DONE ?
1743  006056  001011                    BNE     3$
1744  006060  013737  001266  001314    MOV     VH0,TEMP0
1745  006066  012737  000001  010664    MOV     #1,XOFFOK
1746  006074  004537  011454            JSR     R5,AMSG
1747  006100  020512                    GOPRNT                  ;COPY INST
1748  006102  005237  001306      3$:   INC     STCHAR          ;UPDATE THE STARTING CHARACTER
1749  006106  023737  001310  001306    CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
1750  006114  001347                    BNE     1$              ;BRANCH IF NOT COMPLETED
1751  006116  004537  011454            JSR     R5,AMSG         ;CRLF
1752  006122  017232                    CRLFA-2
1753  006124  012737  000001  010664    MOV     #1,XOFFOK       ;ENABLE XOFF
1754  006132  004537  011454            JSR     R5,AMSG         ;COPY INST
1755  006136  020512                    GOPRNT
1756  006140  004737  011274            JSR     PC,DELAY        ;TEST DELAY SWITCH
1757
1758                                ;;*****************************************************************
1759                                ;*TEST 23       S       COPIER - PERIMETER PATTERN
1760                                ;;*****************************************************************
1761  006144  000004            TST23:  SCOPE
1762  006146  004537  011454            JSR     R5,AMSG         ;DISPLAY HEADER
```

```
1763  006152  014412                          M920
1764  006154  012701  000105                  MOV     #'E,R1          ;LOAD STARTING CHARACTER
1765  006160  004737  010144                  JSR     PC,FILBUF       ;FILL THE BUFFER
1766  006164  004737  010246                  JSR     PC,XPRNT        ;DISPLAY A LINE
1767  006170  004737  010246                  JSR     PC,XPRNT        ;DISPLAY A LINE
1768  006174  013737  001270  001312          MOV     VH1,TEMP        ;LOAD VERT COUNT
1769  006202  062737  000002  001312          ADD     #2,TEMP         ;UPDATE BY 2
1770  006210  004737  011072          1$:     JSR     PC,FIRLST       ;FILL FIRST AND LAST
1771  006214  004737  010246                  JSR     PC,XPRNT        ;DISPLAY A LINE
1772  006220  005337  001312                  DEC     TEMP            ;DONE ?
1773  006224  001371                          BNE     1$
1774  006226  012701  000105                  MOV     #'E,R1          ;LOAD STARTING CHAR
1775  006232  004737  010144                  JSR     PC,FILBUF       ;LOAD LINE WITH E'S
1776  006236  004737  010246                  JSR     PC,XPRNT        ;DISPLAY A LINE
1777  006242  004737  010246                  JSR     PC,XPRNT
1778  006246  012737  000001  010664          MOV     #1,XOFFOK
1779  006254  004537  011454                  JSR     R5,AMSG
1780  006260  020512                          GOPRNT                  ;COPY SCREEN
1781  006262  004737  011274                  JSR     PC,DELAY
1782
1783                                  ;;*****************************************************************
1784                                  ;*TEST 24        T       COPIER - DISCLAIMER STATEMENT
1785                                  ;;*****************************************************************
1786  006266  000004                  TST24:  SCOPE
1787
1788  006270  004537  011454                  JSR     R5,AMSG         ;DISPLAY HEADER
1789  006274  014440                          M921
1790
1791  006276  005004                          CLR     R4
1792  006300  016437  006616  006314  1$:     MOV     DISPCH(R4),10$  ;LOAD A POINTER
1793  006306  001405                          BEQ     2$              ;BR IF DONE
1794  006310  004537  011454                  JSR     R5,AMSG         ;DISPLAY COPYRIGHT
1795  006314  023221                  10$:    MTEXTO
1796  006316  005724                          TST     (R4)+           ;UPDATE POINTER
1797  006320  000767                          BR      1$
1798
1799
1800  006322  012737  000001  010664  2$:     MOV     #1,XOFFOK       ;ENABLE XOFF
1801  006330  004537  011454                  JSR     R5,AMSG         ;COPY SCREEN
1802  006334  020512                          GOPRNT
1803  006336  004737  011274                  JSR     PC,DELAY
1804  006342  000240                  PRNTST: NOP                     ;TRY PRINTER TESTS
1805
1806                                  ;;*****************************************************************
1807                                  ;*TEST 25        U       PRINTER CONTROLLER MODE TEST
1808                                  ;;*****************************************************************
1809  006344  000004                  TST25:  SCOPE
1810  006346  032737  002000  001260          BIT     #BIT10,VT5XX    ;IS UNIT EQUIPPED WITH PRINTER.
1811  006354  001002                          BNE     1$              ;YES-TEST IT
1812  006356  000137  006644                  JMP     $EOP            ;NO-EXIT TEST
1813  006362  004537  011454          1$:     JSR     R5,AMSG         ;DISPLAY HEADER
1814  006366  014635                          MPTCNT
1815  006370  004537  011454                  JSR     R5,AMSG         ;ENABLE PRINTER CONTROLLER MODE.
1816  006374  022325                          ENPNTM
1817  006376  012737  000040  001306          MOV     #40,STCHAR      ;LOAD INITIAL CHAR.
1818  006404  013701  001306          2$:     MOV     STCHAR,R1
```

```
1819  006410  004537  010200            JSR     R5,LIC          ;BUILD A 132 COL. LINE.
1820  006414  001316                    PNTWID
1821  006416  012737  000001  010664    MOV     #1,XOFFOK       ;ALLOW XOFF/XON PROTOCOL
1822  006424  004737  010246            JSR     PC,XPRNT        ;DISPLAY IT.
1823  006430  005237  001306            INC     STCHAR          ;INCREMENT 1ST CHAR.
1824  006434  023737  001306  001310    CMP     STCHAR,LASTCH   ;ISSUED 92 LINES?
1825  006442  001360                    BNE     2$              ;NO-LOOP
1826  006444  004537  011454            JSR     R5,AMSG         ;YES-DISABLE PRINTER
1827  006450  022332                    EXPNTM                  ;CONTROLLER MODE.
1828  006452  004737  011274            JSR     PC,DELAY        ;TEST DELAY SWITCH AND EXIT.
1829
1830                                ;;*****************************************************************
1831                                ;*TEST 26        V       PRINT SCREEN TEST
1832                                ;;*****************************************************************
1833  006456  000004                TST26.  SCOPE
1834
1835
1836  006460  004537  011454            JSR     R5,AMSG         ;DISPLAY PRINT SCREEN MESSAGE
1837  006464  014704                    MPTSCN
1838
1839  006466  004737  013550            JSR     PC,FILLWC       ;FILL THE SCREEN WITH E
1840
1841  006472  012737  000001  010664    MOV     #1,XOFFOK       ;ALLOW XOFF/XON PROTOCOL
1842  006500  004537  011454            JSR     R5,AMSG         ;PRINT THEM
1843  006504  020512                    GOPRNT
1844
1845  006506  004737  011274            JSR     PC,DELAY        ;TEST DELAY SWITCH.
1846
1847                                ;;*****************************************************************
1848                                ;*TEST 27        W       AUTO PRINT TEST
1849                                ;;*****************************************************************
1850  006512  000004                TST27:  SCOPE
1851
1852  006514  004537  011454            JSR     R5,AMSG
1853  006520  014741                    M923A                   ;DISPLAY HEADER
1854  006522  004537  011454            JSR     R5,AMSG         ;ENABLE AUTO-PRINT
1855  006526  020614                    GOAPMD
1856
1857  006530  013737  001266  001314    MOV     VH0,TEMP0       ;LOAD A EXECUTION COUNT
1858  006536  012701  000101            MOV     #101,R1         ;LOAD A CHARACTER
1859  006542  004737  010144            JSR     PC,FILBUF       ;LOAD CHARACTER INTO BUFFER
1860  006546  012737  000001  010664  1$:   MOV     #1,XOFFOK
1861  006554  004737  010246            JSR     PC,XPRNT        ;DISPLAY IT
1862  006560  004737  011274            JSR     PC,DELAY        ;PROGRAM DELAY
1863  006564  005337  001314            DEC     TEMP0           ;FINISHED ?
1864  006570  100366                    BPL     1$              ;BR IF NOT
1865  006572  012737  000001  010664    MOV     #1,XOFFOK
1866  006600  004537  011454            JSR     R5,AMSG         ;DISABLE AUTO-PRINT
1867  006604  020621                    GONAPM
1868  006606  004737  011274            JSR     PC,DELAY        ;ANOTHER DELAY
1869
1870  006612  000137  006644            JMP     $EOP            ;END OF PASS
1871  006616  023221                DISPCH: MTEXT0
1872  006620  023325                    MTEXT1
1873  006622  023426                    MTEXT2
1874  006624  023530                    MTEXT3
```

```
 1875  006626  023613                    MTEXT4
 1876  006630  023715                    MTEXT5
 1877  006632  024016                    MTEXT6
 1878  006634  024050                    MTEXT7
 1879  006636  024150                    MTEXT8
 1880  006640  000000                    0
 1881  006642  000000                    0
 1882
```

```
1883
1884                                    .SBTTL  END OF PASS ROUTINE
1885
1886                                    ;;**********************************************************************
1887                                    ;*INCREMENT THE PASS NUMBER ($PASS)
1888                                    ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
1889                                    ;*TYPE 'END PASS #XXXXX'' (WHERE XXXXX IS A DECIMAL NUMBER)
1890                                    ;*IF THERES A MONITOR GO TO IT
1891                                    ;*IF THERE ISN'T JUMP TO NOWEOP
1892
1893  006644                            $EOP:
1894  006644  000004                           SCOPE
1895  006646  005737  001244                  TST     LAST                    ;TEST IF MORE
1896  006652  001414                          BEQ     1$                      ;BR IF NONE
1897  006654  023737  001244  001246          CMP     LAST,VTNOW              ;IS THIS THE LAST ONE
1898  006662  001410                          BEQ     1$                      ;BR IF YES
1899  006664  062737  000010  001246          ADD     #10,VTNOW
1900  006672  012737  002466  002402          MOV     #TST2,WHERE
1901  006700  000137  002014                  JMP     RSTRT                   ;TEST NEXT ONE
1902  006704  005737  001100          1$:     TST     $PASS                   ;TEST IF FIRST PASS
1903  006710  001002                          BNE     2$                      ;BR IF NOT
1904  006712  104401  021035                  TYPE    ,PASHED                 ;TYPE EOP HEADER
1905  006716  000005                  2$:     RESET
1906  006720  005737  001332                  TST     WFTEST          ;TEST IF W.F. MODE
1907  006724  001402                          BEQ     3$              ;BR IF NOT
1908  006726  004737  011402                  JSR     PC,ADELAY       ;EXTRA DELAY
1909  006732  000240                  3$:     NOP
1910
1911  006734  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
1912  006740  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
1913  006744  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
1914  006752  005527                          DEC     (PC)+           ;;LOOP?
1915  006754  000001          $EOPCT: .WORD   1
1916  006756  003022                          BGT     $DOAGN          ;;YES
1917  006760  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
1918  006762  000001          $ENDCT: .WORD   1
1919  006764  006754                          $EOPCT
1920  006766  104401  007033                  TYPE    ,$ENDMG         ;;TYPE 'END PASS #''
1921  006772  013746  001100                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
1922  006776  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
1923  007000  104401  007030                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
1924  007004  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
1925  007010  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
1926  007012  000005                          RESET                   ;;CLEAR THE WORLD
1927  007014  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
1928  007016  000240                          NOP                     ;;SAVE ROOM
1929  007020  000240                          NOP                     ;;FOR
1930  007022  000240                          NOP                     ;;ACT11
1931  007024                          $DOAGN:
1932  007024  000137                          JMP     @(PC)+          ;;RETURN
1933  007026  007050                  $RTNAD: .WORD   NOWEOP
1934  007030     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
1935  007033     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
1936  007040  050040  051501  020123
1937  007046  000043
1938  007050  005737  001274          NOWEOP: TST     PRTCNT
```

```
1939  007054  100002                          BPL     11$
1940  007056  105337  001274                  DECB    PRTCNT
1941  007062  012737  002466  002402  11$:    MOV     #TST2,WHERE
1942  007070  013746  001112                  MOV     $ERTTL,-(SP)
1943  007074  104405                          TYPDS
1944  007076  104401  007030                  TYPE    ,$ENULL
1945  007102  000137  002006                  JMP     $BEGIN
1946                                ;;*****************************************************************
1947                                ;*TEST 30        X       KEYBOARD OCTAL VALUE LOOP
1948                                ;;*****************************************************************
1949  007106  000004                TST30:  SCOPE
1950  007110  012706  001100        KRBECO: MOV     #STACK,SP
1951  007114  004537  011454                JSR     R5,AMSG                 ;DISPLAY HEADER
1952  007120  016744                        MKE
1953  007122  004737  013612        1$:     JSR     PC,GETCHR               ;GET CHAR
1954  007126  000775                        BR      1$                  ;;BR BACK IF NO INPUT
1955  007130  004737  011474                JSR     PC,OCTAL                ;CONVERT R0 TO OCTAL
1956  007134  113737  011564  017165        MOVB    DIG0,MKEB               ;LOAD DIGIT
1957  007142  113737  011566  017166        MOVB    DIG1,MKEB+1             ;LOAD DIGIT
1958  007150  113737  011570  017167        MOVB    DIG2,MKEB+2             ;LOAD DIGIT
1959  007156  042700  177600                BIC     #177600,R0
1960  007162  001001                        BNE     10$
1961  007164  005200                        INC     R0
1962  007166  012701  007276        10$:    MOV     #BFCHR,R1               ;LOAD POINTER
1963  007172  121100                5$:     CMPB    (R1),R0                 ;TEST IF = TO VALUE IN TABLE ?
1964  007174  001403                        BEQ     3$                      ;BR IF FOUND
1965  007176  005721                        TST     (R1)+                   ;MOVE POINTER
1966  007200  001374                        BNE     5$                      ;BR IF MORE
1967  007202  000407                        BR      2$                      ;BR IF NOT IN LIST
1968  007204  062701  000040        3$:     ADD     #BFCHAR-BFCHR,R1        ;UPDATE POINTER
1969  007210  112137  017160                MOVB    (R1)+,MKEA1             ;LOAD 1ST CHAR
1970  007214  112137  017161                MOVB    (R1)+,MKEA1+1           ;LOAD 2ND
1971  007220  000420                        BR      4$
1972  007222  120027  000040        2$:     CMPB    R0,#40                  ;TEST IF LESS THAN 40
1973  007226  101010                        BHI     6$                      ;BR IF ABOVE
1974  007230  062700  000100                ADD     #100,R0                 ;MAKE PRINTABLE
1975  007234  110037  017161                MOVB    R0,MKEA1+1              ;SAVE CHAR
1976  007240  112737  000136  017160        MOVB    #136,MKEA1              ;ADD A '^' BEFORE CHARACTER
1977  007246  000405                        BR      4$
1978  007250  112737  000040  017161  6$:   MOVB    #40,MKEA1+1             ;LOAD SPACE
1979  007256  110037  017160                MOVB    R0,MKEA1                ;LOAD CHARACTER
1980  007262  005237  010676        4$:     INC     IGNORE                  ;IGNORE DOUBLE CHARACTER FLAG
1981  007266  004537  011454                JSR     R5,AMSG                 ;DISPLAY MESSAGE
1982  007272  017147                        MKEA
1983  007274  000712                        BR      1$                      ;LOOP BACK
1984
1985                                ;TABLE OF DEFINED CHARACTERS
1986
1987  007276  000007                BFCHR:  7                               ;BELL CODE
1988  007300  000010                        10                              ;CURSOR LEFT CODE
1989  007302  000011                        11                              ;TAB CODE
1990  007304  000012                        12                              ;LINE FEED CODE
1991  007306  000015                        15                              ;CARRIAGE RETURN CODE
1992  007310  000033                        33                              ;ESCAPE CODE
1993  007312  000040                        40                              ;SPACE CODE
1994  007314  000177                        177                             ;DELETE CODE
```

```
 1995  007316  000000                          0
 1996  007320  000000                          0
 1997  007322  000000                          0
 1998  007324  000000                          0
 1999  007326  000000                          0
 2000  007330  000000                          0
 2001  007332  000000                          0
 2002  007334  000000                          0
 2003
 2004                                  ;DEFINED CHARACTER EQUIL
 2005
 2006  007336  046102                  BFCHAR: .ASCII  /BL/                    ;BELL
 2007  007340  046103                          .ASCII  /CL/                    ;CURSOR LEFT
 2008  007342  052110                          .ASCII  /HT/                    ;H TAB
 2009  007344  043114                          .ASCII  /LF/                    ;LINE FEED
 2010  007346  051103                          .ASCII  /CR/                    ;CARIAGE RETURN
 2011  007350  051505                          .ASCII  /ES/                    ;ESCAPE
 2012  007352  050123                          .ASCII  /SP/                    ;SPACE
 2013  007354  042504                          .ASCII  /DE/                    ;DELETE
 2014  007356  000000  000000  000000          0,0,0,0,0,0,0,0,0,0
 2015  007364  000000  000000  000000
 2016  007372  000000  000000  000000
 2017  007400  000000
 2018                                          .EVEN
 2019                                  ;;*****************************************************************
 2020                                  ;*TEST 31        Y       KEYBOARD CHARACTER TEST
 2021                                  ;;*****************************************************************
 2022  007402  000004                  TST31:  SCOPE
 2023                                  ;CHECKS THAT TERMINAL IS GENERATING THE EXPECTED ASCII CODES
 2024  007404  032737  001000  001260  KRBTST: BIT     #BIT09,VT5XX    ;IS UNIT A VT52?
 2025  007412  001403                          BEQ     1$
 2026  007414  012703  021550                  MOV     #V52RW,R3       ;YES-SET UP FOR LOWER CASE CHAR.
 2027  007420  000402                          BR      2$
 2028  007422  012703  021532          1$:     MOV     #V50RW,R3       ;NO-SET UP FOR UPPER CASE CHAR.
 2029  007426  012706  001100          2$:     MOV     #STACK,SP
 2030  007432  004537  011454          A:      JSR     R5,AMSG         ;DISPLAY HEADER
 2031  007436  014774                          MKB
 2032  007440  012302                          MOV     (R3)+,R2        ;LOAD ROW #
 2033  007442  032737  001000  001260          BIT     #BIT09,VT5XX    ;UNIT A VT52?
 2034  007450  001404                          BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
 2035  007452  004537  011454                  JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
 2036  007456  015275                          MKBB2
 2037  007460  000403                          BR      2$
 2038  007462  004537  011454          1$:     JSR     R5,AMSG
 2039  007466  015160                          MKBB                    ;TOP ROW
 2040  007470  004737  011572          2$:     JSR     PC,TSTROW       ;CHECK THE ROW
 2041
 2042  007474  012302                  B:      MOV     (R3)+,R2        ;LOAD ROW #
 2043  007476  004537  011454                  JSR     R5,AMSG         ;2ND ROW
 2044  007502  015361                          MKBC
 2045  007504  004737  011572                  JSR     PC,TSTROW       ;CHECK 2ND ROW
 2046
 2047  007510  012302                  C:      MOV     (R3)+,R2        ;LOAD ROW #
 2048  007512  032737  001000  001260          BIT     #BIT09,VT5XX    ;UNIT A VT52?
 2049  007520  001404                          BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
 2050  007522  004537  011454                  JSR     R5,AMSG         ;ISSUE VT52 ROW 3 MESSAGE.
```

```
2051  007526  015526                        MKBD2
2052  007530  000403                        BR      2$
2053  007532  004537  011454        1$:     JSR     R5,AMSG
2054  007536  015421                        MKBD
2055  007540  004737  011572        2$:     JSR     PC,TSTROW       ;CHECK ROW 3
2056
2057  007544  012302                        MOV     (R3)+,R2        ;LOAD ROW #
2058  007546  004537  011454                JSR     R5,AMSG
2059  007552  015615                        MKBE
2060  007554  004737  011572                JSR     PC,TSTROW       ;CHECK ROW 4
2061
2062  007560  012702  021734                MOV     #ROW5,R2
2063  007564  004537  011454                JSR     R5,AMSG         ;DISPLAY MSG
2064  007570  015756                        MKBF                    ;''DEPRESS SPACE BAR''
2065  007572  000261                        SEC                     ;SET FLAG DENOTING FIFTH ROW TEST
2066  007574  004737  011572                JSR     PC,TSTROW       ;CHECK ROW 5
2067
2068                          ;TEST THE ''LEFT-SHIFT'' KEY
2069
2070  007600  004537  011454        D:      JSR     R5,AMSG         ;DEPRESS THE ''LEFT-SHIFT'' KEY
2071  007604  016013                        MKBG
2072  007606  012302                        MOV     (R3)+,R2        ;LOAD ROW 1 SHIFTED TABLE
2073  007610  032737  001000  001260        BIT     #BIT09,VT5XX    ;UNIT A VT52?
2074  007616  001404                        BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
2075  007620  004537  011454                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
2076  007624  015275                        MKBB2
2077  007626  000403                        BR      2$
2078  007630  004537  011454        1$:     JSR     R5,AMSG         ;TEST ROW 1
2079  007634  015160                        MKBB
2080  007636  004737  011572        2$:     JSR     PC,TSTROW       ;TEST THE ROW
2081  007642  004537  011454                JSR     R5,AMSG         ;RELEASE THE SHIFT KEY
2082  007646  015032                        MKB1
2083
2084                          ;TEST THE ''RIGHT-SHIFT'' KEY
2085
2086  007650  004537  011454        E:      JSR     R5,AMSG         ;SET THE ''RIGHT-SHIFT'' KEY
2087  007654  016062                        MKBGA
2088  007656  012302                        MOV     (R3)+,R2        ;LOAD TABLE POINTER
2089  007660  032737  001000  001260        BIT     #BIT09,VT5XX    ;UNIT A VT52?
2090  007666  001404                        BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
2091  007670  004537  011454                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
2092  007674  015275                        MKBB2
2093  007676  000403                        BR      2$
2094  007700  004537  011454        1$:     JSR     R5,AMSG
2095  007704  015160                        MKBB
2096  007706  004737  011572        2$:     JSR     PC,TSTROW       ;TEST THE ROW AGAIN WITH THE RIGHT-SHIFT SET
2097  007712  004537  011454                JSR     R5,AMSG
2098  007716  015032                        MKB1                    ;RELEASE SKIFT KEY
2099                          ;TEST THE CONTROL MODE
2100
2101  007720  004537  011454        F:      JSR     R5,AMSG         ;SET CTRL
2102  007724  016133                        MKBH
2103  007726  012302                        MOV     (R3)+,R2        ;LOAD ROW 1 CTRL TABLE
2104  007730  032737  001000  001260        BIT     #BIT09,VT5XX    ;UNIT A VT52?
2105  007736  001404                        BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
2106  007740  004537  011454                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
```

```
2107  007744  015275                        MKBB2
2108  007746  000403                        BR      2$
2109  007750  004537  011454        1$:     JSR     R5,AMSG
2110  007754  015160                        MKBB
2111  007756  004737  011572        2$:     JSR     PC,TSTROW       ;TEST THE ROW
2112
2113                                ;KEYPAD TEST FOR VT50H AND VT52
2114
2115  007762  032737  010000  001260        BIT     #BIT12,VT5XX            ;KEYPAD ON UUT?
2116  007770  001441                        BEQ     KRBDON                  ;NO EXIT TEST
2117  007772  004537  011454                JSR     R5,AMSG                 ;TYPE TEST ID
2118  007776  016406                        MKBN                            ;'VT50H/VT52 KEYPAD TEST IN NORMAL MODE''
2119  010000  004537  011454                JSR     R5,AMSG                 ;TYPE INSTRUCTIONS
2120  010004  023134                        INSTR                           ;''START TOP ROW LEFT TO RIGHT
2121                                                                        ;/PROCEED TO BOTTOM ROW''
2122  010006  004537  011454                JSR     R5,AMSG                 ;DISABLE ALTERNATE KEYPAD MODE
2123  010012  022306                        EXAKP
2124  010014  012702  022340                MOV     #KPTAB,R2               ;SET TABLE POINTER TO NORMAL MODE CODES
2125  010020  004737  012202                CALL    TSTKPS                  ;DO THE TEST
2126  010024  032737  001000  001260        BIT     #1000,VT5XX             ;IS UUT VT52?
2127  010032  001420                        BEQ     KRBDON                  ;NO EXIT TEST
2128  010034  004537  011454                JSR     R5,AMSG                 ;TYPE TEST ID
2129  010040  016463                        MKB52                           ;'VT52 ALTERNATE KEYPAD MODE TEST''
2130  010042  004537  011454                JSR     R5,AMSG                 ;TYPE INSTRUCTIONS
2131  010046  023134                        INSTR
2132  010050  004537  011454                JSR     R5,AMSG                 ;ENTER ALT-MODE
2133  010054  022301                        ENAKP
2134  010056  012702  022416                MOV     #AKPTAB,R2              ;SET TABLE POINTER TO ALTERNATE MODE CODES
2135  010062  004737  012202                CALL    TSTKPS                  ;DO THE TEST
2136  010066  004537  011454                JSR     R5,AMSG                 ;EXIT ALT-MODE
2137  010072  022306                        EXAKP
2138                                ;COMPLETION OF KEYBOARD-KEYPAD TEST
2139
2140  010074  004537  011454        KRBDON: JSR     R5,AMSG         ;END OF KEYBOARD TEST
2141  010100  016625                        MKBR
2142  010102  000137  007404                JMP     KRBTST          ;LOOP
2143                                ;;*****************************************************************
2144                                ;*TEST 32        Z       KEYBOARD ECHO LOOP
2145                                ;;*****************************************************************
2146  010106  000004                *ST32:  SCOPE
2147  010110  012706  001100        KRBECH: MOV     #STACK,SP
2148  010114  004537  011454                JSR     R5,AMSG         ;DISPLAY HEADER
2149  010120  017174                        MKEH
2150
2151  010122  004737  013612        1$:     JSR     PC,GETCHR       ;GET CHARACTER
2152  010126  000775                        BR      1$
2153  010130  110077  171150                MOVB    R0,@VTOB        ;LOAD THE CHARACTER
2154  010134  105777  171142        2$:     TSTB    @VTOS           ;WAIT FOR DONE
2155  010140  100375                        BPL     2$
2156  010142  000767                        BR      1$              ;LOOP BACK
2157
2158                                ;LOAD A SINGLE CHARACTER ACROSS THE SCREEN WIDTH
2159                                ;
2160
2161  010144  013702  001320        FILBUF: MOV     WIDTH,R2        ;LOAD WIDTH VALUE
2162  010150  012700  026346        FILBFB: MOV     #BUFFER,R0      ;SET-UP BUFFER POINTER
```

```
2163  010154  112720  000015                  MOVB    #15,(R0)+       ;LOAD 'CR'
2164  010160  112720  000012                  MOVB    #12,(R0)+       ;LOAD 'FL'
2165  010164  110120                FILBFA:   MOVB    R1,(R0)+        ;SAVE THE CHARACTER IN THE BUFFER
2166  010166  005302                          DEC     R2              ;FINISHED?
2167  010170  001375                          BNE     FILBFA          ;BRANCH IF NOT COMPLETED
2168  010172  112710  000377                  MOVB    #377,(R0)       ;LOAD TERM.
2169  010176  000207                          RTS     PC              ;EXIT
2170
2171                                  ;LOAD AN INCREMENTING CHARACTER ACROSS THE SCREEN WIDTH
2172                                  ;ONLY 40 THRU 177 ARE LEGAL CHARACTERS
2173
2174  010200  012700  026346         LIC:     MOV     #BUFFER,R0      ;SET-UP BUFFER POINTER
2175  010204  013502                          MOV     @(5)+,R2        ;SET-UP WIDTH
2176  010206  112720  000015                  MOVB    #15,(R0)+       ;LOAD 'CR'
2177  010212  112720  000012                  MOVB    #12,(R0)+       ;LOAD 'LF'
2178  010216  110120                LICA:     MOVB    R1,(0)+         ;SAVE A CHARACTER IN THE BUFFER
2179  010220  005201                          INC     R1              ;UPDATE THE CHARACTER
2180  010222  023701  001310                  CMP     LASTCH,R1       ;TEST FOR
2181  010226  001002                          BNE     LICB            ;BRANCH IF NOT
2182  010230  012701  000040                  MOV     #40,R1          ;MAKE A LEGAL CHARACTER
2183  010234  005302                LICB:     DEC     R2              ;DECREMENT COUNT
2184  010236  001367                          BNE     LICA            ;BRANCH IF NOT COMPLETED
2185  010240  112710  000377                  MOVB    #377,(R0)       ;LOAD TERM
2186  010244  000205                          RTS     R5              ;EXIT
2187
2188                                  ;DISPLAY SUBROUTINE
2189                                  ;CALLED WITH DATA TO BE DISPLAYED LOADED IN THE BUFFER
2190
2191  010246  012700  026346         XPRNT:   MOV     #BUFFER,R0      ;SETUP BUFFER POINTER
2192  010252  105777  171024         XPRNTA:  TSTB    @VTOS           ;TEST READY
2193  010256  100404                          BMI     XPRNTB          ;BRANCH IF SET
2194  010260  005777  171016                  TST     @VTOS           ;TEST ERROR
2195  010264  100372                          BPL     XPRNTA          ;BRANCH IF RESET
2196  010266  104001                          ERROR   1               ;ERROR FLAG SET ON TRANSMITTER STATUS
2197  010270  112001                XPRNTB:   MOVB    (0)+,R1
2198  010272  100563                          BMI     1$              ;BR IF MINUS
2199  010274  122701  000033         5$:      CMPB    #33,R1          ;TEST FOR ESC
2200  010300  001003                          BNE     3$              ;BR IF NOT
2201  010302  005237  010674                  INC     ANESC           ;SET SOFT FLAG
2202  010306  000402                          BR      4$
2203  010310  005037  010674         3$:      CLR     ANESC           ;CLEAR SOFT FLAG
2204  010314  110177  170764         4$:      MOVB    R1,@VTOB        ;LOAD CHAR
2205  010320  105777  170752                  TSTB    @VTIS           ;TEST INPUT FLAG
2206  010324  100352                          BPL     XPRNTA          ;BR IF CLEARED
2207  010326  005737  010674                  TST     ANESC           ;TEST IF ESC
2208  010332  001347                          BNE     XPRNTA
2209  010334  013737  001254  013664  52$:    MOV     TIME0,TIME1
2210  010342  005037  013666                  CLR     TIME2           ;LOAD DELAY
2211  010346  105777  170724         2$:      TSTB    @VTIS           ;TEST IF INPUT FLAG
2212  010352  100407                          BMI     53$             ;BR IF SET
2213  010354  005337  013666                  DEC     TIME2           ;DELAY
2214  010360  001372                          BNE     2$
2215  010362  005337  013664                  DEC     TIME1           ;DELAY
2216  010366  001367                          BNE     2$
2217  010370  000440                          BR      60$             ;REPORT ERROR
2218  010372  005737  010676         53$:     TST     IGNORE          ;IGNORE KEYBOARD CHARACTERS ?
```

```
2219  010376  001104                          BNE     15$             ;BR IF YES
2220  010400  017737  170674  001126          MOV     @VTIB,$BDDAT    ;READ CHAR
2221  010406  042737  177600  001126          BIC     #177600,$BDDAT  ;MASK
2222  010414  022737  000021  001126          CMP     #XON,$BDDAT     ;TEST FOR X ON
2223  010422  001003                          BNE     50$
2224  010424  005237  010672                  INC     XONRC           ;SET FLAG
2225  010430  000710                          BR      XPRNTA          ;START AGAIN
2226  010432  022737  000023  001126  50$:    CMP     #XOFF,$BDDAT    ;TEST FOR X OFF
2227  010440  001020                          BNE     51$             ;BR IF NOT
2228  010442  005237  010670                  INC     XOFFRC
2229  010446  005737  010664                  TST     XOFFOK          ;XOFF ENABLED ?
2230  010452  001730                          BEQ     52$
2231  010454  012737  000001  010666          MOV     #1,AXOFF        ;SET X OFF SOFT FLAG
2232  010462  005737  010662                  TST     XOFFBR          ;TEST
2233  010466  001271                          BNE     XPRNTA          ;BR BACK
2234  010470  000721                          BR      52$             ;BR
2235  010472  012737  000000  001124  60$:    MOV     #0,$GDDAT
2236  010500  000437                          BR      13$
2237  010502  105777  170432          51$:    TSTB    @SWR            ;TEST SWR
2238  010506  100371                          BPL     60$             ;BR IF CLEARED
2239  010510  012737  000057  001124          MOV     #'/,$GDDAT      ;LOAD EXPECTED
2240  010516  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
2241  010524  001437                          BEQ     14$             ;;BR IF EQUAL
2242  010526  012737  000134  001124          MOV     #'\,$GDDAT      ;LOAD EXPECTED
2243  010534  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
2244  010542  001016                          BNE     13$             ;BR IF NOT
2245
2246                                  ;'\' OR LOOP EXIT
2247
2248  010544  012737  000001  010660          MOV     #1,LOOP         ;SET SOFT FLAG
2249  010552  012737  020317  010716          MOV     #MQ0,FINDTA     ;SETUP MESSAGE
2250  010560  005037  010662          16$:    CLR     XOFFBR
2251  010564  005037  010664                  CLR     XOFFOK
2252  010570  005037  010676                  CLR     IGNORE
2253  010574  000137  010706                  JMP     FINDOT
2254
2255  010600  005737  010676          13$:    TST     IGNORE          ;IGNORE INPUT CHARACTER
2256  010604  001001                          BNE     15$
2257  010606  104004                          ERROR   4               ;UNEXPECTED OR INCORRECT INPUT FLAG
2258  010610  000240          15$:    NOP
2259  010612  000240                  NOP
2260  010614  000240                  NOP
2261  010616  000240                  NOP
2262  010620  000137  010252                  JMP     XPRNTA
2263
2264                                  ;'/' OR STANDARD EXIT
2265
2266  010624  005037  010660          14$:    CLR     LOOP
2267  010630  012737  020404  010716          MOV     #MQ1,FINDTA     ;SETUP MESSAGE
2268  010636  000137  010560                  JMP     16$
2269
2270                                  ;NORMAL EXIT
2271
2272  010642  005037  010664          1$:     CLR     XOFFOK
2273  010646  005037  010676                  CLR     IGNORE
2274  010652  005037  010662                  CLR     XOFFBR
```

```
2275  010656  000207                        RTS     PC              ;EXIT
2276
2277  010660  000000                LOOP:   0
2278  010662  000000                XOFFBR: 0
2279  010664  000000                XOFFOK: 0
2280  010666  000000                AXOFF:  0
2281  010670  000000                XOFFRC: 0
2282  010672  000000                XONRC:  0
2283  010674  000000                ANESC:  0
2284  010676  000000                IGNORE: 0               ;WHEN SET IGNORE KEYBOARD FLAGS
2285
2286                                ;DETERMINE THE  TEST TO GO TO
2287  010700  004537  011454        FNDA:   JSR     R5,AMSG
2288  010704  020472                        MQ2                     ;ERROR ASK AGAIN
2289  010706  012706  001100        FINDOT: MOV     #STACK,SP
2290  010712  004537  011454                JSR     R5,AMSG
2291  010716  020404                FINDTA: MQ1
2292  010720  004737  013612                JSR     PC,GETCHR
2293  010724  000770                        BR      FINDOT
2294  010726  042700  100600                BIC     #100600,R0      ;MASK
2295  010732  122700  000101                CMPB    #'A,R0          ;TEST FOR NUMBER
2296  010736  101360                        BHI     FNDA
2297  010740  122700  000132                CMPB    #'Z,R0          ;TEST FOR OTHERS
2298  010744  103755                        BLO     FNDA
2299  010746  042700  177740                BIC     #177740,R0      ;MAKE 0-32
2300  010752  005300                        DEC     R0
2301  010754  110037  001102                MOVB    R0,$TSTNM       ;LOAD THAT TEST #
2302  010760  006300                        ASL     R0
2303  010762  005760  011006                TST     DSPCH(R0)       ;TEST IF VALID
2304  010766  001744                        BEQ     FNDA                    ;BR IF NOT
2305  010770  000240                        NOP
2306  010772  000240                        NOP
2307  010774  016037  011006  001106        MOV     DSPCH(R0),$LPADR        ;LOAD LOOP ADDRESS
2308  011002  000170  011006                JMP     @DSPCH(R0)      ;GO TO THAT TEST
2309
2310                                ;SUBTEST DISPATCH TABLE
2311
2312  011006  002044                DSPCH:  TST1+2
2313  011010  002470                        TST2+2
2314  011012  002510                        TST3+2
2315  011014  002564                        TST4+2
2316  011016  002664                        TST5+2
2317  011020  002766                        TST6+2
2318  011022  003430                        TST7+2
2319  011024  004000                        TST10+2
2320  011026  004156                        TST11+2
2321  011030  004304                        TST12+2
2322  011032  004342                        TST13+2
2323  011034  005102                        TST14+2
2324  011036  005252                        TST15+2
2325  011040  005364                        TST16+2
2326  011042  005534                        TST17+2
2327  011044  005570                        TST20+2
2328  011046  005660                        TST21+2
2329  011050  006012                        TST22+2
2330  011052  006146                        TST23+2
```

```
2331  011054  006270                    TST24+2
2332  011056  006346                    TST25+2
2333  011060  006460                    TST26+2
2334  011062  006514                    TST27+2
2335  011064  007110                    TST30+2
2336  011066  007404                    TST31+2
2337  011070  010110                    TST32+2
2338
2339                              ;SUBROUTINE TO LOAD COPIER TEST
2340
2341  011072  012700  026346     FIRLST: MOV     #BUFFER,R0
2342  011076  112720  000015             MOVB    #15,(R0)+       ;LOAD CR
2343  011102  112720  000012             MOVB    #12,(R0)+       ;LOAD LF
2344  011106  112720  000105             MOVB    #'E,(R0)+
2345  011112  112720  000105             MOVB    #'E,(R0)+
2346  011116  112720  000105             MOVB    #'E,(R0)+
2347  011122  013702  001320             MOV     WIDTH,R2        ;LOAD WIDTH
2348  011126  163702  001270             SUB     VH1,R2
2349  011132  005302                     DEC     R2
2350  011134  112720  000040     1$:     MOVB    #40,(R0)+       ;LOAD A SPACE
2351  011140  005302                     DEC     R2              ;DONE ?
2352  011142  100374                     BPL     1$              ;BR UNTIL DONE
2353  011144  112720  000105             MOVB    #'E,(R0)+
2354  011150  112720  000105             MOVB    #'E,(R0)+
2355  011154  112720  000105             MOVB    #'E,(R0)+       ;LOAD 80 TH
2356  011160  112710  000377             MOVB    #377,(R0)       ;LOAD TERM
2357  011164  000207                     RTS     PC              ;EXIT
2358
2359                              ;SUBROUTINE FOR THE DATA PATH TEST
2360
2361  011166  012700  026346     DTPSR:  MOV     #BUFFER,R0
2362  011172  012501             DTPSRB: MOV     (5)+,R1         ;GET FIRST CHARACTER
2363  011174  012502                     MOV     (5)+,R2         ;GET SECOND CHARACTER
2364  011176  013703  001320             MOV     WIDTH,R3                ;SET THE WIDTH
2365  011202  006203                     ASR     R3              ;DIVIDE BY 2
2366  011204  112720  000015             MOVB    #15,(R0)+       ;LOAD 'CR'
2367  011210  112720  000012             MOVB    #12,(R0)+       ;LOAD 'LF'
2368  011214  110120             DTPSRA: MOVB    R1,(0)+         ;
2369  011216  110220                     MOVB    R2,(0)+
2370  011220  005303                     DEC     R3
2371  011222  100374                     BPL     DTPSRA
2372  011224  112710  000377             MOVB    #377,(R0)       ;LOAD TERM
2373  011230  000205                     RTS     R5
2374
2375                              ;SUBROUTINE TO LOAD BUFFER WITH GRAPHICS CHARACTERS
2376
2377  011232  012700  026346     GBBUF:  MOV     #BUFFER,R0      ;LOAD BUFFER ADDRESS
2378  011236  012701  000136             MOV     #136,R1         ;LOAD INITIAL CHAR.
2379  011242  010402                     MOV     R4,R2           ;LOAD BUFFER COUNT
2380  011244  110120             1$:     MOVB    R1,(R0)+        ;INSERT A CHAR. IN THE BUFFER
2381  011246  005201                     INC     R1              ;INCREMENT CHAR.
2382  011250  122701  000177             CMPB    #177,R1         ;AT END OF GRAPHICS STRING?
2383  011254  001002                     BNE     2$              ;NO
2384  011256  012701  000136             MOV     #136,R1         ;YES-RESET IT TO 1ST GRAPH. CHAR.
2385  011262  005302             2$:     DEC     R2              ;DECREMENT BUFFER COUNT.
2386  011264  001367                     BNE     1$              ;NOT AT END-LOOP
```

```
2387  011266  112710  000377                    MOVB    #377,(R0)       ;END OF BUFFER-INSERT TERMINATOR
2388  011272  000207                            RTS     PC              ;AND EXIT.
2389
2390                                    ;PROGRAM DELAY ROUTINE
2391
2392  011274  013737  001256  011376    DELAY:  MOV     SUBTST,10$      ;LOAD COUNT
2393  011302  005037  011400                    CLR     11$
2394  011306  005737  001332                    TST     WFTEST          ;TEST IF W.F. MODE
2395  011312  001413                            BEQ     2$              ;BR IF NOT
2396  011314  006237  011376                    ASR     10$             ;CHANGE DELAY TIMER
2397  011320  006237  011376                    ASR     10$
2398  011324  006237  011376                    ASR     10$
2399  011330  000240                            NOP
2400  011332  000240                            NOP
2401  011334  000240                            NOP
2402  011336  000240                            NOP
2403  011340  000240                            NOP
2404  011342  032777  010000  167570    2$:     BIT     #BIT12,@SWR     ;TEST SR
2405  011350  001006                            BNE     3$              ;BR IF SET
2406  011352  005337  011400                    DEC     11$             ;DELAY
2407  011356  001371                            BNE     2$
2408  011360  005337  011376            1$:     DEC     10$
2409  011364  100375                            BPL     1$              ;DELAY
2410  011366  000240                    3$:     NOP
2411  011370  000240                            NOP
2412  011372  000240                            NOP
2413  011374  000207                            RTS     PC              ;EXIT
2414
2415  011376  000002                    10$:    2
2416  011400  000000                    11$:    0
2417
2418  011402  013737  001256  011450    ADELAY: MOV     SUBTST,10$
2419  011410  005037  011452                    CLR     11$
2420  011414  006237  011450                    ASR     10$
2421  011420  006237  011450                    ASR     10$
2422  011424  005337  011452            2$:     DEC     11$
2423  011430  001375                            BNE     2$
2424  011432  005337  011450                    DEC     10$
2425  011436  100372                            BPL     2$
2426  011440  000240                            NOP
2427  011442  000240                            NOP
2428  011444  000240                            NOP
2429  011446  000207                            RTS     PC
2430  011450  000000                    10$:    0
2431  011452  000000                    11$:    0
2432
2433
2434                                    ;HEADER SUBROUTINE FOR VT50 AND VT52
2435
2436  011454  012537  011464            AMSG:   MOV     (R5)+,10$       ;GET POINTER
2437  011460  004537  013670            1$:     JSR     R5,MTOB         ;MOVE TO BUFFER
2438  011464  000000                    10$:    0
2439  011466  004737  010246            11$:    JSR     PC,XPRNT        ;DISPLAY IT
2440  011472  000205                            RTS     R5              ;EXIT
2441
2442
```

```
                                        ;OCTAL - 3 BIT CONVERSION

011474  010001                OCTAL:  MOV     R0,R1           ;LOAD R1
011476  042701  177770                BIC     #177770,R1      ;MASK
011502  062701  000060                ADD     #60,R1
011506  110137  011570                MOVB    R1,DIG2         ;SAVE LSD
011512  010001                        MOV     R0,R1
011514  006001                        ROR     R1
011516  006001                        ROR     R1
011520  006001                        ROR     R1
011522  042701  177770                BIC     #177770,R1
011526  062701  000060                ADD     #60,R1
011532  110137  011566                MOVB    R1,DIG1         ;SAVE IT
011536  010001                        MOV     R0,R1
011540  006101                        ROL     R1
011542  006101                        ROL     R1
011544  000301                        SWAB    R1
011546  042701  177770                BIC     #177770,R1
011552  062701  000060                ADD     #60,R1
011556  110137  011564                MOVB    R1,DIG0         ;SAVE MSD
011562  000207                        RTS     PC              ;EXIT

011564  000000                DIG0:   0
011566  000000                DIG1:   0
011570  000000                DIG2:   0

                                        ;SUBROUTINE FOR THE KEYBOARD CHARACTER TEST
011572  103403                TSTROW: BCS     BYPASS          ;BYPASS HEADER MESSAGE IF FLAG IS SET
011574  004537  011454                JSR     R5,AMSG         ;DISPLAY HEADER
011600  015071                        MKBA
011602  000241                BYPASS: CLC                     ;CLEAR FLAG
011604  004737  013612        1$:     CALL    GETCHR          ;GET A CHARACTER
011610  000775                        BR      1$              ;WAIT FOR INPUT
011612  004737  012010                CALL    PARITY          ;ADJUST CHAR IF PARITY
011616  120037  012000                CMPB    R0,100$         ;CMP EXPCT AND RCVD
011622  001003                        BNE     2$              ;TYPE ERROR INFO
011624  005722                        TST     (R2)+
011626  100366                        BPL     1$              ;LOOP TILL DONE
011630  000207                        RETURN                  ;EXIT TEST

                                        ;COME HERE IF EXPECTED NOT EQUAL TO RECVD
                                        ;CONVERT RESULTS TO OCTAL FOR TYPEOUT

011632  010037  001126        2$:     MOV     R0,$BDDAT       ;LOAD BAD CHARACTER
011636  004737  011474                JSR     PC,OCTAL        ;CONVERT TO OCTAL
011642  113737  011564  016616        MOVB    DIG0,MKBQB      ;LOAD OCTAL #
011650  113737  011566  016617        MOVB    DIG1,MKBQB+1
011656  113737  011570  016620        MOVB    DIG2,MKBQB+2
011664  042700  177600                BIC     #177600,R0
011670  120027  000040                CMPB    R0,#40          ;TEST IF PRINTABLE
011674  101002                        BHI     10$             ;BR IF PRINTABLE
011676  112700  000056                MOVB    #56,R0          ;CONVERT TO A "*" CHARACTER
011702  110037  016612        10$:    MOVB    R0,MKBQ2        ;SAVE CHAR
011706  011200                        MOV     (R2),R0         ;GET GOOD CHAR
011710  053700  012006                BIS     MASK2,R0
011714  010037  001124                MOV     R0,$GDDAT       ;LOAD GOOD CHARACTER
```

```
 2499  011720  004737  011474                      JSR     PC,OCTAL        ;CONVERT IT
 2500  011724  113737  011564  016577              MOVB    DIG0,MKBQA      ;LOAD DIGIT
 2501  011732  113737  011566  016600              MOVB    DIG1,MKBQA+1
 2502  011740  113737  011570  016601              MOVB    DIG2,MKBQA+2
 2503  011746  042700  177600                      BIC     #177600,R0
 2504  011752  110037  016573                      MOVB    R0,MKBQ1        ;SAVE CHAR
 2505  011756  023737  001276  001144              CMP     VTIS,$TKS       ;TEST IF ON CTY
 2506  011764  001403                              BEQ     3$              ;BR IF YES
 2507  011766  004537  011454                      JSR     R5,AMSG         ;DISPLAY ERROR  MESSAGE
 2508  011772  016532                              MKBQ
 2509  011774  104004                  3$:         ERROR   4               ;CHARACTER RECVD NOT EQUAL TO EXPECTED
 2510  011776  000702                              BR      1$              ;BRANCH BACK AND TEST THE CHARACTER AGAIN
 2511  012000  000000                  100$:       0
 2512  012002  000000                  101$:       0
 2513  012004  177600                  MASK1:      177600
 2514  012006  000000                  MASK2:      0
 2515
 2516                                  ;CALLING SEQUENCE:      CALL    PARITY
 2517                                  ;ENTRY CONDITIONS:      (R0)=RECEIVED CHARACTER
 2518                                  ;                       (R2)=EXPECTED CHARACTER
 2519                                  ;EXIT CONDITION:        (100$)=EXPECTED CHARACTER ADJUSTED FOR PARITY
 2520                                  ;TEST TO ADD PARITY TO EXPCTD CHAR
 2521                                  ;       SW02=1          ENABLE PARITY BIT TEST
 2522                                  ;       SW00-01=00      EVEN PARITY CHECK
 2523                                  ;       SW00-01=01      ODD PARITY CHECK
 2524                                  ;       SW00-01=10      ALWAYS A 0
 2525                                  ;       SW00-01=11      ALWAYS A 1
 2526  012010  012737  177600  012004  PARITY: MOV     #177600,MASK1
 2527  012016  005037  012006                  CLR     MASK2
 2528  012022  032777  000004  167110          BIT     #BIT2,@SWR      ;TEST SWR
 2529  012030  001416                          BEQ     4$              ;DO NOT TEST PARITY BIT
 2530  012032  042737  000200  012004          BIC     #BIT7,MASK1     ;ENABLE PARITY BIT
 2531  012040  032777  000002  167072          BIT     #BIT1,@SWR      ;TEST IF FORCED PARITY
 2532  012046  001417                          BEQ     5$              ;BR IF NOT FORCED PARITY BIT
 2533  012050  032777  000001  167062          BIT     #BIT0,@SWR      ;TEST FOR EVEN/ODD PARITY
 2534  012056  001403                          BEQ     4$              ;BR IF ALLWAYS OFF
 2535  012060  052737  000200  012006          BIS     #BIT7,MASK2     ;SET BIT 7
 2536  012066  111237  012000          4$:     MOVB    (R2),100$       ;GET EXPECTED
 2537  012072  053737  012006  012000          BIS     MASK2,100$      ;SET BIT 7 IF EXPECTED
 2538  012100  043700  012004                  BIC     MASK1,R0        ;MASK VALUE READ
 2539  012104  000207                          RETURN                  ;EXIT ROUTINE
 2540  012106  005037  012002          5$:     CLR     101$            ;CLEAR TEMP
 2541  012112  111237  012000                  MOVB    (R2),100$       ;CLEAR CHAR SAVE
 2542  012116  006037  012000          20$:    ROR     100$            ;ROTATE CHAR
 2543  012122  103002                          BCC     21$             ;BR IF NO CARRY
 2544  012124  005237  012002                  INC     101$            ;UPDATE CNT
 2545  012130  105737  012000          21$:    TSTB    100$            ;DONE ?
 2546  012134  001370                          BNE     20$             ;BR IF NOT
 2547  012136  032777  000001  166774          BIT     #BIT0,@SWR      ;TEST EVEN/ODD
 2548  012144  001407                          BEQ     23$             ;BR IF OPER. SAYS EVEN
 2549  012146  006037  012002                  ROR     101$            ;
 2550  012152  103403                          BCS     22$             ;BR IF ODD ALREADY
 2551  012154  052737  000200  012006          BIS     #BIT7,MASK2     ;SET PARITY BIT
 2552  012162  000741                  22$:    BR      4$              ;BR TO TEST CHAR
 2553  012164  006037  012002          23$:    ROR     101$            ;
 2554  012170  103003                          BCC     24$             ;BR IF EVEN ALREADY
```

```
2555  012172  052737  000200  012006          BIS     #BIT7,MASK2
2556  012200  000732                  24$:    BR      4$              ;BR TO TEST CHAR
2557
2558                                  ;ROUTINE TO TEST THE KEYPAD INPUTS.
2559                                  ;STORES THE RECEIVED DATA IN "HOLD" AND THE EXPECTED DATA IN
2560                                  ;"GOOD". INVOKES TSTDAT TO COMPARE THE KEY VALUES.
2561  012202                          TSTKPS:
2562  012202  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2563  012204  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2564  012206  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2565  012210  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2566  012212  012703  026304          70$:    MOV     #HOLD,R3        ;INITIALIZE POINTER TO RCVD DATA
2567  012216  010237  026344                  MOV     R2,RETRY        ;SAVE TABLE POINTER FOR EXPECTED DATA
2568  012222  004737  013612          71$:    CALL    GETCHR          ;GET INPUT-RETURN WITH RCVD IN R0
2569  012226  000775                          BR      71$             ;LOOP IF NONE
2570  012230  042700  177600                  BIC     #177600,R0      ;STRIP PARITY BIT FROM CHARACTER IF ANY
2571  012234  120027  000033                  CMPB    R0,#33          ;IS FIRST CHARACTER 33?
2572  012240  001402                          BEQ     40$             ;BRANCH IF RCVD IS NOT SINGLE-CHARACTER
2573                                                                  ;/SEQUENCE
2574                                  ;PROCESSING FOR SINGLE-CHARACTER RCVD DATA
2575  012242  110023                          MOVB    R0,(R3)+        ;STORE CHARACTER IN HOLD
2576  012244  000422                          BR      DELIM           ;BRANCH TO PUT ON DELIMITER
2577                                  ;PROCESSING IF MORE THAN A SINGLE-CHARACTER RCVD SEQUENCE
2578  012246  110023                  40$:    MOVB    R0,(R3)+        ;STORE 1ST CHARACTER IN HOLD
2579  012250  004737  013612          41$:    CALL    GETCHR          ;GET 2ND CHARACTER
2580  012254  000775                          BR      41$             ;LOOP IF NONE
2581  012256  042700  177600                  BIC     #177600,R0      ;STRIP PARITY BIT FROM CHARACTER IF ANY
2582  012262  120027  000077                  CMPB    R0,#77          ;IS INPUT A THREE-CHARACTER SEQUENCE?
2583  012266  001402                          BEQ     42$             ;BRANCH FOR A THREE-CHARACTER SEQUENCE
2584                                  ;PROCESSING FOR TWO-CHARACTER RCVD DATA
2585  012270  110023                          MOVB    R0,(R3)+        ;STORE 2ND CHARACTER IN HOLD
2586  012272  000407                          BR      DELIM           ;BRANCH TO PUT ON DELIMITER
2587                                  ;PROCESSING FOR THREE-CHARACTER RCVD DATA
2588  012274  110023                  42$:    MOVB    R0,(R3)+        ;STORE 2ND CHARACTER IN HOLD
2589  012276  004737  013612          43$:    CALL    GETCHR          ;GET 3RD CHARACTER
2590  012302  000775                          BR      43$             ;LOOP IF NONE
2591  012304  042700  177600                  BIC     #177600,R0      ;STRIP PARITY BIT FROM CHARACTER IF ANY
2592  012310  110023                          MOVB    R0,(R3)+        ;STORE 3RD CHARACTER IN HOLD
2593  012312  112713  000377          DELIM:  MOVB    #377,(R3)       ;PUT ON DELIMITER
2594                                  ;PROCESSING TO STORE EXPECTED DATA
2595  012316  012701  026324                  MOV     #GOOD,R1        ;INITIALIZE POINTER TO EXPCTD DATA
2596  012322  112221                  72$:    MOVB    (R2)+,(R1)+     ;PUT EXPCTD DATA IN GOOD
2597  012324  121227  000377                  CMPB    (R2),#377       ;END OF KEY DATA?
2598  012330  001374                          BNE     72$             ;GET REST OF KEY DATA
2599  012332  112211                          MOVB    (R2)+,(R1)      ;PUT ON DELIMITER
2600  012334  004737  012360                  CALL    TSTDAT          ;COMPARE THE KEY VALUES
2601  012340  105712                          TSTB    (R2)            ;END OF TABLE?
2602  012342  001401                          BEQ     73$             ;EXIT TEST
2603  012344  000722                          BR      70$             ;GET NEXT KEY
2604  012346                          73$:
2605  012346  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2606  012350  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
2607  012352  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2608  012354  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2609  012356  000207                          RETURN
2610
```

```
                                ;ROUTINE TO COMPARE EXPECTED AND RECEIVED KEY VALUES FOR THE KEYPAD TESTS.
                                ;THE MNEMONIC OF THE EXPECTED KEY IS DISPLAYED.
                                ;ENTER WITH RCVD DATA IN ''HOLD'' STORAGE AREA AND EXPCTD DATA IN
                                ;''GOOD'' STORAGE AREA.
012360  012703  026304        TSTDAT: MOV     #HOLD,R3        ;SET DATA POINTER FOR RCVD DATA
012364  012701  026324                MOV     #GOOD,R1        ;SET DATA POINTER FOR EXPCTD DATA
012370  010305                        MOV     R3,R5           ;POINTER FOR RCVD DATA
012372  010104                        MOV     R1,R4           ;POINTER FOR EXPCTD DATA
012374  121127  000377        75$:    CMPB    (R1),#377       ;DELIMITER YET?
012400  001001                        BNE     CONT            ;BRANCH TO CONTINUE COMPARISON IF
                                                              ;/DELIMITER IS NOT DETECTED
012402  000207                        RETURN                  ;EXIT ROUTINE
012404  122123                CONT:   CMPB    (R1)+,(R3)+     ;COMPR EXPCTD AND RCVD
012406  001772                        BEQ     75$
                                ;CODE SEQUENCE ENTERED FOR ERROR CONDITION
012410  013702  026344                MOV     RETRY,R2        ;SAVE TABLE PTR FOR EXPCTD CHARS
012414  005237  010676                INC     IGNORE
012420  004537  011454                JSR     R5,AMSG         ;REPORT ERROR
012424  022766                        ERCHR                   ;MSG. ''KEY CODE ERROR''
012426  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012430  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012432  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
012434  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
012436  012701  026324                MOV     #GOOD,R1        ;INITIALIZE PTR TO EXPCTD DATA
012442  121127  000033                CMPB    (R1),#33        ;DOES 1ST CHAR DENOTE A
                                                              ;/MULTI-CHAR SEQUENCE?
012446  001414                        BEQ     PROC1           ;BRANCH TO PROCESS MULTI-CHAR SEQUENCE IF EQUAL
                                ;PROCESSING FOR ONE-CHAR SEQUENCE
012450  012702  022524                MOV     #KPTB0,R2       ;PTR TO TABLE OF POSSIBLE CHARS
012454  012703  177777                MOV     #-1,R3          ;INITIALIZE TABLE INDEX
012460  005203                LOOP1:  INC     R3              ;STEP INDEX THRU TABLE
012462  121122                        CMPB    (R1),(R2)+      ;COMPARE CHAR WITH TABLE
                                                              ;/OF POSSIBLE VALUES
012464  001375                        BNE     LOOP1           ;BRANCH TO COMPARE WITH NEXT CHAR IN
                                                              ;/TABLE IF NOT EQUAL
012466  012704  022560                MOV     #TB1MNE,R4      ;INITIALIZE MNEMONIC TABLE PTR
012472  004737  013050                JSR     PC,INSMNE       ;INSERT MNEMONIC CHAR INTO MSG
012476  000434                        BR      MNEMSG          ;BRANCH TO DISPLAY EXPCTD MNENOMIC KEY
012500  105721                PROC1:  TSTB    (R1)+           ;INCREMENT PTR TO ACCESS 2ND CHAR
012502  121127  000077                CMPB    (R1),#77        ;DOES THE 2ND CHAR DENOTE A
                                                              ;/THREE-CHAR ESCAPE SEQUENCE?
012506  001414                        BEQ     PROC2           ;BRANCH FOR THREE-CHAR ESCAPE SEQUENCE
                                ;PROCESSING FOR TWO-CHAR ESCAPE SEQUENCE
012510  012702  022640                MOV     #AKPTB2,R2      ;INITIALIZE PTR TO EXPCTD CHAR
012514  012703  177777                MOV     #-1,R3          ;INITIALIZE TABLE INDEX
012520  005203                LOOP2:  INC     R3              ;STEP INDEX THRU TABLE
012522  121122                        CMPB    (R1),(R2)+      ;COMPARE CHAR WITH TABLE EXPCTD CHAR
012524  001375                        BNE     LOOP2           ;BRANCH IF NOT EQUAL
012526  012704  022650                MOV     #TB2MNE,R4      ;INITIALIZE MNEMONIC TABLE PTR
012532  004737  013050                JSR     PC,INSMNE       ;INSERT MNEMONIC CHAR INTO MSG
012536  000414                        BR      MNEMSG          ;BRANCH TO DISPLAY EXPCTD MNEMONIC KEY
                                ;PROCESSING FOR THREE-CHAR ESCAPE SEQUENCE
012540  105721                PROC2:  TSTB    (R1)+           ;INCREMENT PTR TO ACCESS 3RD CHAR
012542  012702  022542                MOV     #AKPTB1,R2      ;INITIALIZE PTR TO EXPCTD CHAR
012546  012703  177777                MOV     #-1,R3          ;INITIALIZE TABLE INDEX
012552  005203                LOOP3:  INC     R3              ;STEP INDEX THRU TABLE
```

```
2667  012554  121122                  CMPB    (R1),(R2)+      ;COMPARE RCVD CHAR WITH TABLE EXPCTD CHAR
2668  012556  001375                  BNE     LOOP3           ;CONTINUE SEARCH IF NOT EQUAL
2669  012560  012704  022560          MOV     #TB1MNE,R4      ;INITIALIZE MNEMONIC TABLE PTR
2670  012564  004737  013050          JSR     PC,INSMNE       ;INSERT MNEMONIC CHAR INTO MSG
2671  012570  004537  011454  MNEMSG: JSR     R5,AMSG         ;DISPLAY MSG
2672  012574  023011                  EXPDKY                  ;'EXPECTED KEY MNEMONIC=''
2673  012576  005237  010676          INC     IGNORE
2674  012602  004537  011454          JSR     R5,AMSG
2675  012606  022704                  SHBE                    ;MSG. 'EXPECTED KEY CODE=''
2676  012610  012704  026324          MOV     #GOOD,R4        ;INITIALIZE POINTER TO 'EXPECTED KEY CODE'' DATA
2677  012614  004737  013140          CALL    OUTPUT          ;DISPLAY 'EXPECTED KEY CODE'' DATA
2678  012620  005237  010676          INC     IGNORE
2679  012624  004537  011454          JSR     R5,AMSG
2680  012630  022746                  AWAS                    ;MSG. 'BAD KEY CODE=''
2681  012632  010504                  MOV     R5,R4           ;MOVE RCVD PTR TO #HOLD INTO R4 FOR DISPLAY
2682  012634  004737  013140          CALL    OUTPUT          ;RCVD CHARS POINTED TO #HOLD IN R4
2683                          ;CODE SEQUENCE ENTERED TO IDENTIFY THE BAD KEY MNEMONIC
2684  012640  012701  026304          MOV     #HOLD,R1        ;SET DATA PTR TO BAD RCVD DATA
2685                          ;INSPECT 1ST CHAR OF BAD RCVD DATA
2686  012644  121127  000033          CMPB    (R1),#33        ;IS 1ST CHAR 33?
2687  012650  001417                  BEQ     MULCHR          ;IF 1ST CHAR IS 33, BRANCH FOR MULTIPLE
2688                                                          ;/CHAR PROCESSING
2689                          ;PROCESSING TO DISPLAY BAD KEY MNEMONIC FOR SINGLE-CHAR SEQUENCE
2690  012652  012702  022524          MOV     #KPTB0,R2       ;INITIALIZE TABLE PTR
2691  012656  012703  177777          MOV     #-1,R3          ;INITIALIZE TABLE INDEX
2692  012662  121227  000000  LOOP4:  CMPB    (R2),#0         ;AT END OF TABLE?
2693  012666  001456                  BEQ     NOTFND          ;BRANCH TO REPORT THAT KEY CAN NOT
2694                                                          ;/BE IDENTIFIED
2695  012670  005203                  INC     R3              ;STEP INDEX THRU TABLE
2696  012672  121122                  CMPB    (R1),(R2)+      ;COMPARE RCVD CHAR WITH TABLE VALUES
2697  012674  001372                  BNE     LOOP4           ;CONTINUE SEARCH IF CHAR NOT FOUND
2698  012676  012704  022560          MOV     #TB1MNE,R4      ;INITIALIZE MNEMONIC TABLE PTR
2699  012702  004737  013104          JSR     PC,INSBAD       ;INSERT MNEMONIC CHAR INTO MSG
2700  012706  000442                  BR      BADMNE          ;BRANCH TO DISPLAY BAD MNEMONIC KEY
2701                          ;PROCESSING TO DETERMINE WHETHER A TWO- OR THREE-CHAR SEQUENCE
2702  012710  105721  MULCHR: TSTB    (R1)+                   ;INCREMENT DATA PTR TO ACCESS
2703                                                          ;/2ND RCVD CHAR
2704  012712  121127  000077          CMPB    (R1),#77        ;IS 2ND CHAR 77?
2705  012716  001417                  BEQ     MORE            ;BRANCH TO GET 3RD CHAR IF 77
2706                          ;PROCESSING TO DISPLAY BAD KEY MNEMONIC FOR TWO-CHAR SEQUENCE
2707  012720  012702  022640          MOV     #AKPTB2,R2      ;INITIALIZE TABLE PTR
2708  012724  012703  177777          MOV     #-1,R3          ;INITIALIZE TABLE INDEX
2709  012730  121227  000000  LOOP5:  CMPB    (R2),#0         ;AT END OF TABLE?
2710  012734  001433                  BEQ     NOTFND          ;BRANCH TO REPORT THAT KEY CAN NOT
2711                                                          ;/BE IDENTIFIED
2712  012736  005203                  INC     R3              ;STEP INDEX THRU TABLE
2713  012740  121122                  CMPB    (R1),(R2)+      ;COMPARE RCVD CHAR WITH TABLE VALUES
2714  012742  001372                  BNE     LOOP5           ;CONTINUE SEARCH IF CHAR NOT FOUND
2715  012744  012704  022650          MOV     #TB2MNE,R4      ;INITIALIZE MNEMONIC TABLE PTR
2716  012750  004737  013104          JSR     PC,INSBAD       ;INSERT MNEMONIC CHAR INTO MSG
2717  012754  000417                  BR      BADMNE          ;BRANCH TO DISPLAY BAD MNEMONIC KEY
2718                          ;PROCESSING TO DISPLAY BAD KEY MNEMONIC FOR THREE-CHAR SEQUENCE
2719  012756  105721  MORE:   TSTB    (R1)+                   ;INCREMENT DATA PTR TO ACCESS 3RD CHAR
2720  012760  012702  022542          MOV     #AKPTB1,R2      ;INITIALIZE TABLE PTR
2721  012764  012703  177777          MOV     #-1,R3          ;INITIALIZE TABLE INDEX
2722  012770  121227  000000  LOOP6:  CMPB    (R2),#0         ;AT END OF TABLE?
```

```
2723  012774  001413                    BEQ     NOTFND          ;BRANCH TO REPORT THAT KEY CAN NOT
2724                                                              ;/BE IDENTIFIED
2725  012776  005203                    INC     R3              ;STEP INDEX THRU TABLE
2726  013000  121122                    CMPB    (R1),(R2)+      ;COMPARE RCVD CHAR WITH TABLE VALUES
2727  013002  001372                    BNE     LOOP6           ;CONTINUE SEARCH IF CHAR NOT FOUND
2728  013004  012704  022560            MOV     #TB1MNE,R4      ;INITIALIZE MNEMONIC TABLE PTR
2729  013010  004737  013104            JSR     PC,INSBAD       ;INSERT MNEMONIC CHAR INTO MSG
2730  013014  004537  011454  BADMNE:   JSR     R5,AMSG         ;DISPLAY MESSAGE
2731  013020  023046                    BADKEY                  ;'BAD KEY MNEMONIC=''
2732  013022  000403                    BR      74$             ;BRANCH TO EXIT ROUTINE
2733  013024  004537  011454  NOTFND:   JSR     R5,AMSG         ;DISPLAY MESSAGE
2734  013030  023076                    UNLKEY                  ;'KEY CAN NOT BE IDENTIFIED''
2735  013032                    74$:
2736  013032  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
2737  013034  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
2738  013036  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
2739  013040  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
2740  013042  105777  166232            TSTB    @VTIB           ;CLEAR CONDITION CODES
2741  013046  000207                    RETURN                  ;RETURN TO RETRIEVE NEXT INPUT FROM KEYPAD
2742                              ;++
2743                              ;SUBROUTINE TO INSERT MNEMONIC INTO MESSAGE TO IDENTIFY THE EXPECTED KEY
2744                              ;       CALLING SEQUENCE:       JSR     PC,INSMNE
2745                              ;       ENTRY CONDITIONS:       (R3)=TABLE INDEX
2746                              ;                               (R4)=TABLE POINTER
2747                              ;       EXIT CONDITIONS:        MNEMONIC CHAR INSERTED INTO MESSAGE
2748                              ;--
2749  013050  006303            INSMNE: ASL     R3              ;MULTIPLY TABLE INDEX BY 4 TO GET ADDRESS
2750  013052  006303                    ASL     R3              ;/DISPLACEMENT OF TABLE ENTRY
2751  013054  060304                    ADD     R3,R4           ;ADD ADJ INDEX TO GET ADDRESS OF KEY CHAR
2752  013056  012703  023011            MOV     #EXPDKY,R3      ;INITIALIZE MSG ADDRESS PTR
2753  013062  112463  000030            MOVB    (R4)+,30(R3)    ;LOAD CHARS FOR EXPCTD KEY MNEMONIC INTO MSG
2754  013066  112463  000031            MOVB    (R4)+,31(R3)    ;
2755  013072  112463  000032            MOVB    (R4)+,32(R3)    ;
2756  013076  112463  000033            MOVB    (R4)+,33(R3)    ;
2757  013102  000207                    RTS     PC              ;TRANSFER PROGRAM CONTROL BACK TO MAIN
2758                              ;SUBROUTINE TO INSERT MNEMONIC INTO MESSAGE TO IDENTIFY THE BAD KEY
2759  013104  006303            INSBAD: ASL     R3              ;MULTIPLY TABLE INDEX BY 4 TO GET ADDRESS
2760  013106  006303                    ASL     R3              ;/DISPLACEMENT OF TABLE ENTRY
2761  013110  060304                    ADD     R3,R4           ;ADD ADJ INDEX TO GET ADDRESS OF KEY CHAR
2762  013112  012703  023046            MOV     #BADKEY,R3      ;INITIALIZE MSG ADDRESS PTR
2763  013116  112463  000023            MOVB    (R4)+,23(R3)    ;LOAD CHARS FOR BAD KEY MNEMONIC INTO MSG
2764  013122  112463  000024            MOVB    (R4)+,24(R3)    ;
2765  013126  112463  000025            MOVB    (R4)+,25(R3)    ;
2766  013132  112463  000026            MOVB    (R4)+,26(R3)    ;
2767  013136  000207                    RTS     PC              ;TRANSFER PROGRAM CONTROL BACK TO MAIN
2768
2769
2770                              ;ROUTINE TO OUTPUT GOOD AND BAD KEYPAD OCTAL VALUES ON SCREEN
2771                              ;ENTER WITH ADDRESS CONTAINING CHARACTER IN R4
2772  013140                    OUTPUT:
2773  013140  010146                    MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2774  013142  005001                    CLR     R1
2775  013144  112761  000040  022731  77$:     MOVB    #40,OUT(R1)     ;FILL AREA WITH SPACES
2776  013152  005201                    INC     R1
2777  013154  120127  000013                  CMPB    R1,#13          ;FULL YET
2778  013160  001371                    BNE     77$
```

```
2779  013162  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2780  013164  112400                          MOVB    (R4)+,R0        ;SETUP KEY VALUE TO CONVERT
2781  013166  004737  011474                  CALL    OCTAL           ;DO IT
2782  013172  113737  011564  022731          MOVB    DIG0,OUT
2783  013200  113737  011566  022732          MOVB    DIG1,OUT+1
2784  013206  113737  011570  022733          MOVB    DIG2,OUT+2
2785  013214  121427  000377                  CMPB    (R4),#377       ;DELIMITER
2786  013220  001433                          BEQ     76$             ;EXIT
2787  013222  112400                          MOVB    (R4)+,R0        ;SETUP 2ND VALUE OF KEY
2788  013224  004737  011474                  CALL    OCTAL           ;DO IT
2789  013230  113737  011564  022735          MOVB    DIG0,OUT+4
2790  013236  113737  011566  022736          MOVB    DIG1,OUT+5
2791  013244  113737  011570  022737          MOVB    DIG2,OUT+6
2792  013252  121427  000377                  CMPB    (R4),#377       ;DELIMITER ?
2793  013256  001414                          BEQ     76$             ;EXIT
2794  013260  112400                          MOVB    (R4)+,R0        ;SETUP 3RD VALUE OF KEY
2795  013262  004737  011474                  CALL    OCTAL
2796  013266  113737  011564  022741          MOVB    DIG0,OUT+10
2797  013274  113737  011566  022742          MOVB    DIG1,OUT+11
2798  013302  113737  011570  022743          MOVB    DIG2,OUT+12
2799  013310  005237  010676          76$:    INC     IGNORE
2800  013314  004537  011454                  JSR     R5,AMSG         ;OUTPUT VALUES ON SCREEN
2801  013320  022731                          OUT
2802  013322  000207                          RETURN
2803
2804
2805                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2806
2807                                  ;;**********************************************************************
2808                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2809                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2810                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2811                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2812                                  ;*REPLACED WITH SPACES.
2813                                  ;*CALL:
2814                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
2815                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
2816
2817  013324                          $TYPDS:
2818  013324  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2819  013326  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2820  013330  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2821  013332  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2822  013334  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2823  013336  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
2824  013342  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
2825  013346  100004                          BPL     1$              ;;BR IF INPUT IS POS.
2826  013350  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
2827  013352  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
2828  013360  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
2829  013362  012703  013540                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
2830  013366  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
2831  013372  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
2832  013374  016001  013530                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
2833  013400  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
2834  013402  002402                          BLT     4$              ;;BR IF DONE
```

```
2835  013404  005202                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2836  013406  000774                        BR      3$
2837  013410  060105                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
2838  013412  005702                        TST     R2              ;;CHECK IF BCD DIGIT=0
2839  013414  001002                        BNE     5$              ;;FALL THROUGH IF 0
2840  013416  105716                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
2841  013420  100407                        BMI     7$              ;;BR IF YES
2842  013422  106316                5$:     ASLB    (SP)            ;;MSD?
2843  013424  103003                        BCC     6$              ;;BR IF NO
2844  013426  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2845  013434  052702  000060        6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2846  013440  052702  000040        7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2847  013444  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2848  013446  005720                        TST     (R0)+           ;;JUST INCREMENTING
2849  013450  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
2850  013454  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
2851  013456  003002                        BGT     8$              ;;GO TO EXIT
2852  013460  010502                        MOV     R5,R2           ;;GET THE LSD
2853  013462  000764                        BR      6$              ;;GO CHANGE TO ASCII
2854  013464  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2855  013466  100003                        BPL     9$              ;;BR IF NO
2856  013470  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2857  013476  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
2858  013500  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
2859  013502  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2860  013504  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2861  013506  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2862  013510  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2863  013512  104401  013540                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2864  013516  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2865  013524  012616                        MOV     (SP)+,(SP)
2866  013526  000002                        RTI                     ;;RETURN TO USER
2867  013530  023420                $DTBL:  10000.
2868  013532  001750                        1000.
2869  013534  000144                        100.
2870  013536  000012                        10.
2871  013540  000004                $DBLK:  .BLKW   4
2872
2873                                ;SUBROUTINE TO FILL THE SCREEN WITH A CHARACTER
2874
2875  013550  012701  000105        FILLWC: MOV     #'E,R1          ;LOAD CHARACTER BYTE
2876  013554  004737  010144                JSR     PC,FILBUF       ;LOAD THE LINE WITH CHAR
2877  013560  013737  001266  013610        MOV     VH0,10$         ;LOAD COUNT
2878  013566  012737  000001  010664  1$:   MOV     #1,XOFFOK       ;INSURE XOFF/XON CONTROL.
2879  013574  004737  010246                JSR     PC,XPRNT        ;DISPLAY THE LINE
2880  013600  005337  013610                DEC     10$
2881  013604  001370                        BNE     1$              ;LOOP UNTIL DONE
2882  013606  000207                        RTS     PC              ;EXIT
2883  013610  000000                10$:    0
2884
2885                                ;SUBROUTINE TO GET A CHARACTER FROM THE VT50 AND VT52
2886                                ;CALLED AFTER CHAR. HAS BEEN ENTERED INTO INPUT BUFFER
2887                                ;RETURN IS WITH CHAR. IN R0
2888
2889  013612  013737  001254  013664  GETCHR: MOV     TIME0,TIME1     ;LOAD TIME COUNTER
2890  013620  005037  013666                CLR     TIME2
```

```
2891
2892  013624  105777  165446        1$:     TSTB    @VTIS           ;TEST INPUT STATUS
2893  013630  100005                        BPL     2$              ;BR IF CLEARED
2894  013632  017700  165442                MOV     @VTIB,R0        ;READ A CHAR
2895  013636  062716  000002                ADD     #2,(SP)         ;UPDATE RETURN
2896  013642  000207                        RTS     PC              ;EXIT
2897
2898  013644  005337  013666        2$:     DEC     TIME2           ;DELAY
2899  013650  001365                        BNE     1$
2900  013652  005337  013664                DEC     TIME1           ;FINISHED ?
2901  013656  100362                        BPL     1$              ;LOOP TILL TIME EXPIRED
2902  013660  104002                        ERROR   2               ;NO INPUT FLAG FROM DEVICE
2903  013662  000207                        RTS     PC              ;EXIT
2904
2905  013664  000000        TIME1:  0
2906  013666  000000        TIME2:  0
2907
2908                        ;MOVE TO THE OUTPUT BUFFER
2909                        ;CALLED WITH SOURCE ADDRESS OF DATA IN R5
2910                        ;RETURN WITH BUFFER LOADED WITH DATA
2911
2912  013670  012500        MTOB:   MOV     (R5)+,R0        ;LOAD DEST.
2913  013672  012701  026346        MOV     #BUFFER,R1      ;LOAD R1
2914  013676  112021        1$:     MOVB    (R0)+,(R1)+     ;LOAD BYTE
2915  013700  100376                BPL     1$              ;OUTPUT BUFFER LOADED WHEN ''377'' CHAR.
2916                                                        ;/IS ENCOUNTERED
2917  013702  000205                RTS     R5              ;EXIT
2918
```

```
2919
2920                                    ;ASCII MESSAGES
2921
013704 006415 041412 053132 TITLE:  .ASCIZ  <15><15><12>%CZVTCFO VT50A,B,H/52 ACCPT%<15><12><377>
013745    015 005012 043012 M91:    .ASCIZ  <15><12><12><12>/FULL SCREEN OF THE CHARACTER E/<15><12><377>
014013    015 005012 042012 M92:    .ASCIZ  <15><12><12><12>/DATA PATH TEST/<15><12><377>
014041    015 005012 051412 M93:    .ASCIZ  <15><12><12><12>/SINGLE CHARACTER PER LINE/<15><377>
014101    015 005012 051012 M94:    .ASCIZ  <15><12><12><12>/ROTATING PATTERN/<15><12><377>
014131    015 005012 041412 M96:    .ASCIZ  <15><12><12><12>/CURSOR MOTION TEST/<15><12><377>
014163    015 005012 052012 M97:    .ASCIZ  <15><12><12><12>/TAB, BACKSPACE AND BELL TEST/<15><12><12><377>
014230 005015 005012 051105 M910:   .ASCIZ  <15><12><12><12>/ERASE LINE TEST/<15><12><377>
014257    015 005012 042412 M911:   .ASCIZ  <15><12><12><12>/ERASE SCREEN TEST/<15><12><377>
014310 005015 005012 044526 M912:   .ASCIZ  <15><12><12><12>/VIDEO COUPLING TEST/<15><12><377>
014343    033 015510 006512 M914:   .ASCIZ  <33><110><33><112><15><12><12>/TERMINAL IDENTIFICATION TEST/<15><12><377
014412 005015 047503 044520 M920:   .ASCIZ  <15><12>/COPIER PERIMETER/<15><12><377>
014440 005015 044504 041523 M921:   .ASCIZ  <15><12>/DISCLAIMER STATEMENT/<15><12><377>
014472 005015 047510 042114 M922:   .ASCIZ  <15><12>/HOLD SCREEN MODE TEST/<15><12><377>
014525    015 043412 040522 M9221:  .ASCIZ  <15><12>/GRAPHICS MODE AND REV. LINE FEED TEST/<15><12><377>
014600 057433 044033 045033 M923:   .ASCIZ  <33><137><33><110><33><112>/AUTO COPY MODE TEST/<15><12><377>
014635    033 015510 050112 MPTCNT: .ASCIZ  <33><110><33><112>/PRINTER CONTROLLER MODE ENTERED/<15><12><377>
014704 044033 045033 051120 MPTSCN: .ASCIZ  <33><110><33><112>/PRINT A SCREEN OF E'S/<15><12><377>
014741    110 045033 052501 M923A:  .ASCIZ  <110><33><112>/AUTO PRINT MODE TEST/<15><12><377>
014774 005015 005012 042513 MKB:    .ASCII  <15><12><12><12>/KEYBOARD CHARACTER TEST/<15><12><12>
015032 042522 042514 051501 MKB1:   .ASCIZ  /RELEASE THE ''CAPS LOCK'' KEY/<15><12><377>
015071    015 046012 043105 MKBA:   .ASCIZ  <15><12>/LEFT TO RIGHT IN A ROW, DEPRESS ONE KEY AT A TIME/<15><12><377>
015160 005015 052123 051101 MKBB:   .ASCII  <15><12>/STARTING WITH THE TOP ROW EXCEPT THE THIRD/
015234 043040 047522 020115         .ASCIZ  / FROM RIGHT END AND LAST KEYS/<15><12><377>
015275    015 005012 052123 MKBB2:  .ASCIZ  <15><12><12>/STARTING WITH THE TOP ROW EXCEPT THE LAST KEY/<15><12><377>
015361    015 005012 052123 MKBC:   .ASCIZ  <15><12><12>/START WITH THE SECOND ROW/<15><12><377>
015421    015 005012 052123 MKBD:   .ASCII  <15><12><12>/START WITH THE THIRD ROW EXCEPT THE CTRL/
015474 005015 047101 020104         .ASCIZ  <15><12>/AND THE 'BLANK'' KEYS/<15><12><377>
015526 005015 051412 040524 MKBD2:  .ASCIZ  <15><12><12>/START WITH THE THIRD ROW ,BEGIN ROW WITH ''A'' KEY/<15><12><3
015615    015 005012 052123 MKBE:   .ASCII  <15><12><12>/START WITH THE FOURTH ROW EXCEPT THE SCROLL,LEFT-SHIFT,/
015707    015 051012 043511         .ASCIZ  <15><12>/RIGHT-SHIFT, REPEAT AND COPY KEYS/<15><12><377>
015756 005015 042012 050105 MKBF:   .ASCIZ  <15><12><12>/DEPRESS THE SPACE BAR/<15><12><12><377>
016013    015 047012 053517 MKBG:   .ASCII  <15><12>/NOW HOLD DOWN THE 'LEFT-SHIFT'' KEY/<15><12><377>
016062 005015 047012 053517 MKBGA:  .ASCII  <15><12><12>/NOW HOLD DOWN THE 'RIGHT-SHIFT'' KEY/<15><12><377>
016133    015 005012 047516 MKBH:   .ASCII  <15><12><12>/NOW HOLD DOWN THE ''CTRL'' KEY/<15><12><377>
016175    015 051412 040524 MKBI:   .ASCII  <15><12>/START WITH THE TOP ROW/<15><12><377>
016230 005015 052123 051101 MKBJ:   .ASCII  <15><12>/START WITH THE 2ND ROW/<15><12><377>
016263    015 051412 040524 MKBK:   .ASCII  <15><12>/START  WITH THE 3RD ROW/<15><12><377>
016317    015 051412 040524 MKBL:   .ASCII  <15><12>/START WITH THE 4TH ROW/<15><12><377>
016352 005015 052123 051101 MKBM:   .ASCII  <15><12>/START WITH THE LAST ROW/<15><12><377>
016406 005015 005012 052126 MKBN:   .ASCII  <15><12><12><12>%VT50H/VT52 KEYPAD TEST IN NORMAL MODE%<15><12><12><377>
016463    015 005012 053012 MKB52:  .ASCII  <15><12><12><12>/VT52 ALTERNATE KEYPAD MODE TEST/<15><12><12><377>
016532 005015 044103 051101 MKBQ:   .ASCII  <15><12>/CHARACTER WAS IN ERROR/<15><12>
016564 047507 042117 036440         .ASCII  /GOOD = /
016573    040    040    075 MKBQ1:  .BYTE   40,40,75,40
016577    040    040    040 MKBQA:  .BYTE   40,40,40,40,40
016604 040502 020107 020075         .ASCII  /BAD = /
016612    040    040    075 MKBQ2:  .BYTE   40,40,75,40
016616    040    040    040 MKBQB:  .BYTE   40,40,40,15,12,377,0
016625    015 005012 005012 MKBR:   .ASCIZ  <15><12><12><12><12>/KEYBOARD CHARACTER TEST COMPLETE/<15><12><377>
016676 005015 047111 040526 INVLID: .ASCIZ  <15><12>/INVALID BUSS ADDRESS-PLEASE RETRY/<15><12>
016744 005015 042513 041131 MKE:    .ASCII  <15><12>/KEYBOARD ASCII AND OCTAL LOOP/<15><12>
017005    015    012         .BYTE   15,12
```

```
017007  127  042510  020116          .ASCII  /WHEN A KEY IS DEPRESSED, THE ASCII CHARACTER AND/
017067  015  012                     .BYTE   15,12
017071  124  042510  052040          .ASCIZ  /THE THREE DIGIT OCTAL CODE WILL BE ECHOED/
017143  015  012  377                .BYTE   15,12,377,0
017147  015  041412  040510  MKEA:   .ASCII  <15><12>/CHAR = /
017160  040  040  040       MKEA1:  .BYTE   40,40,40,75,40
017165  040  040  040       MKEB:   .BYTE   40,40,40,15,12,377,0
017174  005015  042513  041131  MKEH:   .ASCIZ  <15><12>/KEYBOARD ECHO LOOP/<15><12>
017223  015  012  012         CRLFB:  .BYTE   15,12,12,12,12,12,12,12,12
017234  012  012  012         CRLFA:  .BYTE   12,12,12,12,12,377,0
                              ;HOME AND ERASE SCREEN OPCODE
017243  033  110  033         HOMERS: .BYTE   33,110,33,112,377,0
017251  015  052012  040040  PATH:   .ASCII  <15><12>/T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@/
017323  100  040100  040100          .ASCII  /@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T/
017373  015  052012  042440          .ASCII  <15><12>/T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE/
017445  105  042505  042505          .ASCII  /EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T/
017515  377  000  000                .BYTE   377,0,0
017520  005015  042524  052123  VT50A:  .ASCII  <15><12>/TESTING VT50A (12 LINES-NO COPIER)/
017564  015  012  012               .BYTE   15,12,12,377,0,0
017572  005015  042524  052123  VT50B:  .ASCII  <15><12>/TESTING VT50B (12 LINES-COPIER)/
017633  015  012  012               .BYTE   15,12,12,377,0,0
017641  015  052012  051505  VT55:   .ASCII  <15><12>/TESTING VT55 (24 LINES-COPIER, VT50 BASED)/
017715  015  012  012               .BYTE   15,12,12,377,0,0
017723  015  052012  051505  VT50H:  .ASCII  <15><12>/TESTING VT50H (12 LINES-COPIER-D.C.A.)/
017773  015  012  012               .BYTE   15,12,12,377,0,0
020001  015  052012  051505  VT52K:  .ASCII  <15><12>/TESTING VT52 (24 LINES-NO COPIER OR PRINTER)/
020057  015  012  012               .BYTE   15,12,12,377,0,0
020065  015  052012  051505  VT52L:  .ASCII  <15><12>/TESTING VT52 (24 LINES,COPIER-NO PRINTER)/
020140  015  012  012               .BYTE   15,12,12,377,0,0
020146  005015  042524  052123  VT52M:  .ASCII  <15><12>/TESTING VT52 (24 LINES,PRINTER-NO COPIER)/
020221  015  012  012               .BYTE   15,12,12,377,0,0
020227  015  052012  051505  VT55E:  .ASCII  <15><12>/TESTING VT55 (24 LINES, COPIER, VT52 BASED)/
020304  015  012  012               .BYTE   15,12,12,377,0,0
020312  033  132  377       RFI:    .BYTE   33,132,377,0,0
020317  033  015534  006537  MQ0:    .ASCIZ  <33><134><33><137><15><12><12>/LOOP ON TEST PATTERN LETTER (A THRU Z) ?
020404  056033  057433  005015  MQ1:    .ASCIZ  <33><134><33><137><15><12><12>/START AT TEST PATTERN LETTER (A THRU Z) ?
020472  005015  052012  054522  MQ2:    .ASCIZ  <15><12><12>/TRY AGAIN/<15><12><377>
020512  033  135  012           GOPRNT: .BYTE   33,135,12,12,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0377,0,0
020540  055433  005015  042524  GOHDSC: .ASCIZ  <33><133><15><12>/TESTING SILO REG/<377>
020566  056033  005015  044523  GODSHS: .ASCIZ  <33><134><15><12>/SILO TEST DONE/<15><12><377>
020614  033  136  377           GOAPMD: .BYTE   33,136,377,0,0
020621  033  137  015           GONAPM: .BYTE   33,137,15,12,12,12,0,0,0,0,0,0,0,0,0,0,0,0,377,0,0,0,0
020650  044033  045033  044504  M98:    .ASCIZ  <33><110><33><112>/DIRECT CURSOR ADDRESSING TEST/<377>
020713  015  005012  044506  WHAT0:  .ASCIZ  <15><12><12>/FIRST LINE DEVICE ADDRESS ? = /<200>
020756  005015  046012  051501  WHAT1:  .ASCIZ  <15><12><12>/LAST LINE DEVICE ADDRESS (CR IF NONE) ? = /<200>
021035  015  020012  020040  PASHED: .ASCIZ  <15><12>/          PASS #    # OF ERRORS   /
021102  051105  047522  020122  EM1:    .ASCIZ  /ERROR FLAG SET ON TRANSMITTER STATUS/
021147  116  020117  047111  EM2:    .ASCIZ  /NO INPUT FLAG DETECTED/
021176  047111  047503  051122  EM3:    .ASCIZ  /INCORRECT I.D. CODE/
021222  047125  054105  042520  EM4:    .ASCIZ  /UNEXPECTED OR INCORRECT INPUT CHAR/
021265  111  053116  046101      EM5:    .ASCIZ  /INVALID BUSS ADDRESS, TRY AGAIN/
021325  105  051122  041520      DH1:    .ASCIZ  /ERRPC    VTNOW    TSTNUM/
021354  051105  050122  020103  DH3:    .ASCIZ  /ERRPC    VTNOW    1ST WD   2ND WD   3RD WD/
021423  105  051122  041520      DH4:    .ASCIZ  /ERRPC    VTNOW    TSTNUM    EXPCT    RECV/
        021472
                                        .EVEN
021472  001116  001246  001252  DT1:    $ERRPC,VTNOW,TSTNUM,0
```

```
021502  001116  001246  001330  DT3:    $ERRPC,VTNOW,SAVE4,SAVE2,SAVE3,0
021516  001116  001246  001252  DT4:    $ERRPC,VTNOW,TSTNUM,$GDDAT,$BDDAT,0

                                        .SBTTL  KEYBOARD CHARACTER CODE TABLES

                                ;THE ACTUAL KEYBOARD LAYOUT IS REQUIRED

021532  021566  021622  021660  V50RW:  ROW1,ROW2,ROW3,ROW4,ROW1S,ROW1S,ROW1C
021550  021736  021774  022032  V52RW:  ROW12,ROW22,ROW32,ROW42,ROW12S,ROW12S,ROW12C


                                ;VT50 KEYBOARD EQUIVALENCIES(UPPER CASE CHAR.)

021566  000033  000061  000062  ROW1:   .WORD   33,61,62,63,64,65,66,67,70,71,60,55,75,100010
021622  000011  000121  000127  ROW2:   .WORD   11,121,127,105,122,124,131,125,111,117,120,133,134,12,100177
021660  000101  000123  000104  ROW3:   .WORD   101,123,104,106,107,110,112,113,114,73,47,100015
021710  000132  000130  000103  ROW4:   .WORD   132,130,103,126,102,116,115,54,56,100057
021734  100040                  ROW5:   .WORD   100040

                                ;VT52 KEYBOARD EQUIVALENCES(LOWER CASE CHAR.)

021736  000033  000061  000062  ROW12:  .WORD   33,61,62,63,64,65,66,67,70,71,60,55,75,140,100010
021774  000011  000161  000167  ROW22:  .WORD   11,161,167,145,162,164,171,165,151,157,160,133,134,12,100177
022032  000141  000163  000144  ROW32:  .WORD   141,163,144,146,147,150,152,153,154,73,47,173,100015
022064  000172  000170  000143  ROW42:  .WORD   172,170,143,166,142,156,155,54,56,100057

                                ;SHIFTED ROW CODES

022110  000033  000041  000100  ROW1S:  .WORD   33,41,100,43,44,45,136,46,52,50,51,137,53,100010
022144  000033  000041  000100  ROW12S: .WORD   33,41,100,43,44,45,136,46,52,50,51,137,53,176,100010

                                ;CONTROL ROW CODES

022202  000033  000021  000022  ROW1C:  .WORD   33,21,22,23,24,25,26,27,30,31,20,15,35,100010
022236  000033  000021  000022  ROW12C: .WORD   33,21,22,23,24,25,26,27,30,31,20,15,35,0,100010

                                        ;VT52   ESCAPE SEQUENCES

022274      033     111     015  REVLF:  .BYTE   33,111,015,377,0         ;REVERSE LINE FEED.

022301      033     075     377  ENAKP:  .BYTE   33,075,377,0,0           ;ENABLE ALTERNATE KEYPAD MODE.
022306      033     076     377  EXAKP:  .BYTE   33,076,377,0,0           ;EXIT ALTERNATE KEYPAD MODE

022313      033     106     000  ENGRAF: .BYTE   33,106,0,377,0           ;ENTER GRAPHICS MODE.
022320      033     107     000  EXGRAF: .BYTE   33,107,0,377,0           ;EXIT GRAPHICS MODE.

022325      033     127     377  ENPNTM: .BYTE   33,127,377,0,0           ;ENABLE PRINTER CONTROLLER MODE.
022332      033     130     377  EXPNTM: .BYTE   33,130,377,0,0           ;DISABLE PRINTER CONTROLLER MODE.


                                ;VT50H/VT52 KEYPAD CODES IN NORMAL MODE
022340      033     120     377  KPTAB:  .BYTE   33,120,-1,33,121,-1,33,122,-1,33,101,-1
022354      067     377     070          .BYTE   67,-1,70,-1,71,-1,33,102,-1
022365      064     377     065          .BYTE   64,-1,65,-1,66,-1,33,103,-1
022376      061     377     062          .BYTE   61,-1,62,-1,63,-1,33,104,-1
022407      060     377     056          .BYTE   60,-1,56,-1,15,-1,0
```

```
                                        ;VT52 KEYPAD CODES IN ALTERNATE MODE
022416      033      120      377  AKPTAB: .BYTE   33,120,-1,33,121,-1,33,122,-1,33,101,-1
022432      033      077      167          .BYTE   33,77,167,-1,33,77,170,-1,33,77,171,-1,33,102,-1
022451      033      077      164          .BYTE   33,77,164,-1,33,77,165,-1,33,77,166,-1,33,103,-1
022470      033      077      161          .BYTE   33,77,161,-1,33,77,162,-1,33,77,163,-1,33,104,-1
022507      033      077      160          .BYTE   33,77,160,-1,33,77,156,-1,33,77,115,-1,0
                                        ;KEYPAD CODES FOR ONE-CHAR SEQUENCES IN NORMAL MODE
                                        .EVEN
022524      060      061      062  KPTB0:  .BYTE   60,61,62,63,64,65,66,67,70,71,56,15,0
                                        ;ALTERNATE KEYPAD MODE TABLES TO IDENTIFY MNEMONICS OF KEYPAD CHARACTERS
          022542
022542      160      161      162  AKPTB1: .BYTE   160,161,162,163,164,165,166,167,170,171,156,115,0
          022560
                                        .EVEN
022560  020060   020040          TB1MNE: .ASCII  /0   /
022564  020061   020040                  .ASCII  /1   /
022570  020062   020040                  .ASCII  /2   /
022574  020063   020040                  .ASCII  /3   /
022600  020064   020040                  .ASCII  /4   /
022604  020065   020040                  .ASCII  /5   /
022610  020066   020040                  .ASCII  /6   /
022614  020067   020040                  .ASCII  /7   /
022620  020070   020040                  .ASCII  /8   /
022624  020071   020040                  .ASCII  /9   /
022630  020056   020040                  .ASCII  /.   /
022634  047105   051124                  .ASCII  /ENTR/
022640      101      102      103  AKPTB2: .BYTE   101,102,103,104,120,121,122
          022650
                                        .EVEN
022650  050125   020040          TB2MNE: .ASCII  /UP  /
022654  047504   047127                  .ASCII  /DOWN/
022660  043522   052110                  .ASCII  /RGHT/
022664  042514   052106                  .ASCII  /LEFT/
022670  020114   046102                  .ASCII  /L BL/
022674  020103   046102                  .ASCII  /C BL/
022700  020122   046102                  .ASCII  /R BL/
022704  005015   054105   042520  SHBE:   .ASCII  <15><12>/EXPECTED KEY CODE=/<377>
022731     040   020040   020040  OUT:    .ASCII  /              /<377>
022746  005015   040502   020104  AWAS:   .ASCII  <15><12>/BAD KEY CODE=/<377>
022766  005015   005012   042513  ERCHR:  .ASCII  <15><12><12><12>/KEY CODE ERROR/<377>
023011     015   042412   050130  EXPDKY: .ASCII <15><12>/EXPECTED KEY MNEMONIC=     /<377>
023046  005015   040502   020104  BADKEY: .ASCII <15><12>/BAD KEY MNEMONIC=     /<377>
023076  005015   042513   020131  UNLKEY: .ASCII <15><12>/KEY CAN NOT BE IDENTIFIED/<15><12><377>
023134  052123   051101   020124  INSTR:  .ASCII  /START TOP ROW LEFT TO RIGHT/<15><12>
023171     120   047522   042503          .ASCII  /PROCEED TO BOTTOM ROW/<15><12><377>

023221     015   005012   044124  MTEXT0: .ASCIZ  <15><12><12>/THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR
023325     015   047412   020116  MTEXT1: .ASCIZ  <15><12>/ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION/<
023426  005015   043117   042040  MTEXT2: .ASCIZ  <15><12>/OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT/
023530  005015   051501   046440  MTEXT3: .ASCIZ  <15><12>/AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC./<377>

023613     015   005012   044124  MTEXT4: .ASCIZ  <15><12><12>/THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHO
023715     015   047012   052117  MTEXT5: .ASCIZ  <15><12>/NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL/<
024016  005015   050505   044525  MTEXT6: .ASCIZ  <15><12>/EQUIPMENT CORPORATION./<377>
024050  005015   042012   041505  MTEXT7: .ASCIZ  <15><12><12>/DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
024150  005015   052111   020123  MTEXT8: .ASCII  <15><12>/ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC./
024241     015      012      377          .BYTE   15,12,377,0
(1)                                     .SBTTL  TTY INPUT ROUTINE
```

```
 2922
 2923                                   ;;*****************************************************************
 2924                                   .ENABL  LSB
 2925
 2926                                   .DSABL  LSB
 2927
 2928
 2929                                   ;;*****************************************************************
 2930                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 2931                                   ;*CALL:
 2932                                   ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 2933                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 2934                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
 2935                                   ;
 2936
 2937  024246  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 2938  024250  016666  000004  000002           MOV     4(SP),2(SP)     ;;SAVE THE PS
 2939  024256  105777  154662           1$:     TSTB    @$TKS           ;;WAIT FOR
 2940  024262  100375                           BPL     1$              ;;A CHARACTER
 2941  024264  117766  154656  000004           MOVB    @$TKB,4(SP)     ;;READ THE TTY
 2942  024272  042766  177600  000004           BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 2943  024300  026627  000004  000023           CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 2944  024306  001013                           BNE     3$              ;;BRANCH IF NO
 2945  024310  105777  154630           2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 2946  024314  100375                           BPL     2$              ;;LOOP UNTIL ITS THERE
 2947  024316  117746  154624                   MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 2948  024322  042716  177600                   BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 2949  024326  022627  000021                   CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 2950  024332  001366                           BNE     2$              ;;IF NOT DISCARD IT
 2951  024334  000750                           BR      1$              ;;YES, RESUME
 2952  024336  026627  000004  000140   3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 2953  024344  002407                           BLT     4$              ;;BRANCH IF YES
 2954  024346  026627  000004  000175           CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 2955  024354  003003                           BGT     4$              ;;BRANCH IF YES
 2956  024356  042766  000040  000004           BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 2957  024364  000002                   4$:     RTI                     ;;GO BACK TO USER
 2958                                   ;;*****************************************************************
 2959                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 2960                                   ;*CALL:
 2961                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 2962                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 2963                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 2964
 2965  024366  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 2966  024370  012703  024474           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 2967  024374  022703  024504           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 2968  024400  101405                           BLOS    4$              ;;BR IF YES
 2969  024402  104406                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 2970  024404  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
 2971  024406  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 2972  024412  001003                           BNE     3$              ;;SKIP IF NOT
 2973  024414  104401  001166           4$:     TYPE    ,$QUES          ;;TYPE A '?'
 2974  024420  000763                           BR      1$              ;;CLEAR THE BUFFER AND LOOP
 2975  024422  111337  024472           3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 2976  024426  104401  024472                   TYPE    ,9$
 2977  024432  122723  000015                   CMPB    #15,(R3)+       ;;CHECK FOR RETURN
```

```
2978  024436  001356                        BNE     2$              ;;LOOP IF NOT RETURN
2979  024440  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2980  024444  104401  001170                TYPE    ,$LF            ;;TYPE A LINE FEED
2981  024450  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2982  024452  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2983  024454  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2984  024462  012766  024474  000004        MOV     #$TTYIN,4(SP)
2985  024470  000002                        RTI                     ;;RETURN
2986  024472     000                  9$:    .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2987  024473     000                        .BYTE   0               ;;TERMINATOR
2988  024474  000010              $TTYIN:   .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2989  024504  052536  005015     000  $CNTLU:  .ASCIZ  /^U/<15><12>  ;;CONTROL "U"
2990  024511     136  006507  000012  $CNTLG:  .ASCIZ  /^G/<15><12>  ;;CONTROL "G"
2991  024516  005015  053523  020122  $MSWR:   .ASCIZ  <15><12>/SWR = /
2992  024524  020075     000
2993  024527     040  047040  053505  $MNEW:   .ASCIZ  /  NEW = /
2994  024534  036440  000040
2995                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
2996
2997                                  ;;**********************************************************************
2998                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2999                                  ;*CHANGE IT TO BINARY.
3000                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
3001                                  ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
3002                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
3003                                  ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
3004                                  ;*CALL:
3005                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
3006                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
3007                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
3008
3009  024540  011646              $RDOCT:   MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
3010  024542  016666  000004  000002        MOV     4(SP),2(SP)     ;;INPUT NUMBER
3011  024550  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3012  024552  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3013  024554  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3014  024556  104407                  1$:   RDLIN                   ;;READ AN ASCIZ LINE
3015  024560  012600                        MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
3016  024562  010037  024666                MOV     R0,5$           ;;AND SAVE IT
3017  024566  005001                        CLR     R1              ;;CLEAR DATA WORD
3018  024570  005002                        CLR     R2
3019  024572  112046                  2$:   MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
3020  024574  001420                        BEQ     3$              ;;IF ZERO GET OUT
3021  024576  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
3022  024602  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
3023  024604  122716  000067                CMPB    #'7,(SP)
3024  024610  002423                        BLT     4$
3025  024612  006301                        ASL     R1              ;;*2
3026  024614  006102                        ROL     R2
3027  024616  006301                        ASL     R1              ;;*4
3028  024620  006102                        ROL     R2
3029  024622  006301                        ASL     R1              ;;*8
3030  024624  006102                        ROL     R2
3031  024626  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
3032  024632  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
3033  024634  000756                        BR      2$              ;;LOOP
```

```
 3034  024636  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
 3035  024640  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
 3036  024644  010237  024676                  MOV     R2,$HIOCT
 3037  024650  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 3038  024652  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 3039  024654  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 3040  024656  000002                          RTI                     ;;RETURN
 3041  024660  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
 3042  024662  105010                          CLRB    (R0)            ;;SET A TERMINATOR
 3043  024664  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
 3044  024666  000000                  5$:     .WORD   0
 3045  024670  104401  001166                  TYPE    ,$QUES          ;;''?'' 'CR'' & 'LF''
 3046  024674  000730                          BR      1$              ;;TRY AGAIN
 3047  024676  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 3048  024700  105777  154234  MSCOPE: TSTB    @SWR            ;TEST BIT 7
 3049  024704  100003                          BPL     $SCOPE          ;NOT SET
 3050  024706  005737  010660                  TST     LOOP            ;TEST LOOP
 3051  024712  001041                          BNE     $OVER           ;SET LOOP ON TEST
 3052
 3053                                  .SBTTL  SCOPE HANDLER ROUTINE
 3054
 3055                                  ;;**********************************************************************
 3056                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 3057                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 3058                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 3059                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 3060                                  ;*SW14=1         LOOP ON TEST
 3061                                  ;*SW08=1         LOOP ON TEST IN SWR<7:0>
 3062                                  ;*CALL
 3063                                  ;*      SCOPE           ;;SCOPE=IOT
 3064
 3065  024714                  $SCOPE:
 3066  024714  032777  040000  154216  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 3067  024722  001035                          BNE     $OVER           ;;YES IF SW14=1
 3068                                  ;#####START OF CODE FOR THE XOR TESTER#####
 3069  024724  000416          $XTSTR: BR      6$              ;;IF RUNNING ON THE 'XOR'' TESTER CHANGE
 3070                                                          ;;THIS INSTRUCTION TO A 'NOP'' (NOP=240)
 3071  024726  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 3072  024732  012737  024752  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 3073  024740  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
 3074  024744  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 3075  024750  000414                          BR      $SVLAD          ;;GO TO THE NEXT TEST
 3076  024752  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 3077  024754  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 3078  024760  000416                          BR      $OVER           ;;LOOP ON THE PRESENT TEST
 3079  024762                  6$:;#####END OF CODE FOR THE XOR TESTER#####
 3080  024762  032777  000400  154150          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
 3081  024770  001404                          BEQ     $SVLAD          ;;BR IF NO
 3082  024772  127737  154142  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
 3083  025000  001406                          BEQ     $OVER           ;;BR IF YES
 3084  025002  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 3085  025006  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 3086  025012  105037  001103                  CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 3087  025016  013777  001102  154116  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 3088  025024  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 3089  025030  000002                          RTI                     ;;FIXES PS
```

```
3090
3091                                    .SBTTL  ERROR HANDLER ROUTINE
3092
3093                                    ;;**********************************************************************
3094                                    ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3095                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3096                                    ;*AND GO TO $ERRTYP ON ERROR
3097                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3098                                    ;*SW15=1         HALT ON ERROR
3099                                    ;*SW13=1         INHIBIT ERROR TYPEOUTS
3100                                    ;*CALL
3101                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3102
3103  025032                            $ERROR:
3104  025032  113737  001102  001252            MOVB    $TSTNM,TSTNUM
3105  025040  105237  001103           7$:      INCB    $ERFLG          ;;SET THE ERROR FLAG
3106  025044  001775                            BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
3107  025046  013777  001102  154066            MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
3108  025054  005237  001112                    INC     $ERTTL          ;;INC THE ERROR COUNT
3109  025060  011637  001116                    MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
3110  025064  162737  000002  001116            SUB     #2,$ERRPC
3111  025072  117737  154020  001114            MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
3112  025100  032777  020000  154032            BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
3113  025106  001004                            BNE     20$             ;;SKIP TYPEOUTS
3114  025110  004737  025144                    JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
3115  025114  104401  001167                    TYPE    ,$CRLF
3116  025120                            20$:
3117  025120  005777  154014           2$:      TST     @SWR            ;;HALT ON ERROR
3118  025124  100001                            BPL     3$              ;;SKIP IF CONTINUE
3119  025126  000000                            HALT                    ;;HALT ON ERROR!
3120  025130                            3$:
3121  025130  022737  007014  000042            CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
3122  025136  001001                            BNE     6$              ;;BRANCH IF NO
3123  025140  000000                            HALT                    ;;YES
3124  025142                            6$:
3125  025142  000002                            RTI                     ;;RETURN
3126
3127                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
3128
3129                                    ;;**********************************************************************
3130                                    ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3131                                    ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
3132                                    ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
3133
3134  025144                            $ERRTYP:
3135  025144  104401  001167                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3136  025150  010046                            MOV     R0,-(SP)        ;;SAVE R0
3137  025152  005000                            CLR     R0              ;;PICKUP THE ITEM INDEX
3138  025154  153700  001114                    BISB    @#$ITEMB,R0
3139  025160  001004                            BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
3140                                                                    ;;TYPE THE PC OF THE ERROR
3141  025162  013746  001116                    MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
3142                                                                    ;;ERROR ADDRESS
3143  025166  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3144  025170  000426                            BR      6$              ;;GET OUT
3145  025172  005300                    1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
```

```
 3146  025174  006300                    ASL     R0              ;;      WORK FOR THE ERROR TABLE
 3147  025176  006300                    ASL     R0
 3148  025200  006300                    ASL     R0
 3149  025202  062700  001172            ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 3150  025206  012037  025216            MOV     (R0)+,2$        ;;PICKUP 'ERROR MESSAGE' POINTER
 3151  025212  001404                    BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 3152  025214  104401                    TYPE                    ;;TYPE THE 'ERROR MESSAGE'
 3153  025216  000000            2$:     .WORD   0               ;;'ERROR MESSAGE' POINTER GOES HERE
 3154  025220  104401  001167            TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
 3155  025224  012037  025234    3$:     MOV     (R0)+,4$        ;;PICKUP 'DATA HEADER' POINTER
 3156  025230  001404                    BEQ     5$              ;;SKIP TYPEOUT IF 0
 3157  025232  104401                    TYPE                    ;;TYPE THE 'DATA HEADER'
 3158  025234  000000            4$:     .WORD   0               ;;'DATA HEADER' POINTER GOES HERE
 3159  025236  104401  001167            TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
 3160  025242  011000            5$:     MOV     (R0),R0         ;;PICKUP 'DATA TABLE' POINTER
 3161  025244  001004                    BNE     7$              ;;GO TYPE THE DATA
 3162  025246  012600            6$:     MOV     (SP)+,R0        ;;RESTORE R0
 3163  025250  104401  001167            TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
 3164  025254  000207                    RTS     PC              ;;RETURN
 3165  025256                    7$:
 3166  025256  013046                    MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
 3167  025260  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 3168  025262  005710                    TST     (R0)            ;;IS THERE ANOTHER NUMBER?
 3169  025264  001770                    BEQ     6$              ;;BR IF NO
 3170  025266  104401  025274            TYPE    ,8$             ;;TYPE TWO(2) SPACES
 3171  025272  000771                    BR      7$              ;;LOOP
 3172  025274  020040     000    8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
 3173          025300                    .EVEN
 3174
 3175                                    .SBTTL  TYPE ROUTINE
 3176
 3177                                    ;;**************************************************************************
 3178                                    ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 3179                                    ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 3180                                    ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 3181                                    ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 3182                                    ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 3183                                    ;*
 3184                                    ;*CALL:
 3185                                    ;*1) USING A TRAP INSTRUCTION
 3186                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 3187                                    ;*OR
 3188                                    ;*      TYPE
 3189                                    ;*      MESADR
 3190                                    ;*
 3191
 3192  025300  105737  001157    $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 3193  025304  100002                    BPL     1$              ;;BR IF YES
 3194  025306  000000                    HALT                    ;;HALT HERE IF NO TERMINAL
 3195  025310  000407                    BR      3$              ;;LEAVE
 3196  025312  010046            1$:     MOV     R0,-(SP)        ;;SAVE R0
 3197  025314  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 3198  025320  112046            2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 3199  025322  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 3200  025324  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 3201  025326  012600            60$:    MOV     (SP)+,R0        ;;RESTORE R0
```

```
3202  025330  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
3203  025334  000002                          RTI                     ;;RETURN
3204  025336  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
3205  025342  001430                          BEQ     8$
3206  025344  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
3207  025350  001006                          BNE     5$
3208  025352  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
3209  025354  104401                          TYPE                    ;;TYPE A CR AND LF
3210  025356  001167                          $CRLF
3211  025360  105037  025514                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
3212  025364  000755                          BR      2$              ;;GET NEXT CHARACTER
3213  025366  004737  025450          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
3214  025372  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3215  025376  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
3216  025400  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3217                                                                  ;;AND THE NULL CHAR.
3218  025404  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3219  025410  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3220  025412  004737  025450                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
3221  025416  105337  025514                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
3222  025422  000770                          BR      7$              ;;LOOP
3223
3224                                  ;HORIZONTAL TAB PROCESSOR
3225
3226  025424  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3227  025430  004737  025450          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3228  025434  132737  000007  025514          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3229  025442  001372                          BNE     9$              ;;TAB STOP
3230  025444  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
3231  025446  000724                          BR      2$              ;;GET NEXT CHARACTER
3232  025450  105777  153474          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3233  025454  100375                          BPL     $TYPEC
3234  025456  116677  000002  153466          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3235  025464  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3236  025472  001003                          BNE     1$              ;;BRANCH IF NO
3237  025474  105037  025514                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3238  025500  000406                          BR      $TYPEX          ;;EXIT
3239  025502  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3240  025510  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
3241  025512  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
3242  025514  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
3243  025516  000207          $TYPEX: RTS     PC
3244
3245
3246                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3247
3248                                  ;;**********************************************************************
3249                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3250                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3251                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3252                                  ;*CALL:
3253                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3254                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
3255                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3256                                  ;*      .BYTE   M               ;;M=1 OR 0
3257                                  ;*                                      ;;1=TYPE LEADING ZEROS
```

```
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
025520  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
025524  116637  000001  025743         MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
025532  112637  025745                 MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
025536  062716  000002                 ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
025542  000406                         BR      $TYPON
025544  112737  000001  025743 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
025552  112737  000006  025745         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
025560  112737  000005  025742 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
025566  010346                         MOV     R3,-(SP)        ;;SAVE R3
025570  010446                         MOV     R4,-(SP)        ;;SAVE R4
025572  010546                         MOV     R5,-(SP)        ;;SAVE R5
025574  113704  025745                 MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
025600  005404                         NEG     R4
025602  062704  000006                 ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
025606  110437  025744                 MOVB    R4,$OMODE       ;;SAVE IT FOR USE
025612  113704  025743                 MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
025616  016605  000012                 MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
025622  005003                         CLR     R3              ;;CLEAR THE OUTPUT WORD
025624  006105                 1$:     ROL     R5              ;;ROTATE MSB INTO "C"
025626  000404                         BR      3$              ;;GO DO MSB
025630  006105                 2$:     ROL     R5              ;;FORM THIS DIGIT
025632  006105                         ROL     R5
025634  006105                         ROL     R5
025636  010503                         MOV     R5,R3
025640  006103                 3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
025642  105337  025744                 DECB    $OMODE          ;;TYPE THIS DIGIT?
025646  100016                         BPL     7$              ;;BR IF NO
025650  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
025654  001002                         BNE     4$              ;;TEST FOR 0
025656  005704                         TST     R4              ;;SUPPRESS THIS 0?
025660  001403                         BEQ     5$              ;;BR IF YES
025662  005204                 4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
025664  052703  000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
025670  052703  000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
025674  110337  025740                 MOVB    R3,8$           ;;SAVE FOR TYPING
025700  104401  025740                 TYPE    ,8$             ;;GO TYPE THIS DIGIT
025704  105337  025742         7$:     DECB    $OCNT           ;;COUNT BY 1
025710  003347                         BGT     2$              ;;BR IF MORE TO DO
025712  002402                         BLT     6$              ;;BR IF DONE
025714  005204                         INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
025716  000744                         BR      2$              ;;GO DO THE LAST DIGIT
025720  012605                 6$:     MOV     (SP)+,R5        ;;RESTORE R5
025722  012604                         MOV     (SP)+,R4        ;;RESTORE R4
```

```
3314  025724  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3315  025726  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3316  025734  012616                          MOV     (SP)+,(SP)
3317  025736  000002                          RTI                     ;;RETURN
3318  025740     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3319  025741     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3320  025742     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3321  025743     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3322  025744  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3323                                  .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
3324
3325                                  ;;**************************************************************************
3326                                  ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
3327                                  ;*WITH A RANGE OF 0 TO 2(+33)-1.
3328                                  ;*CALL:
3329                                  ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
3330                                  ;*      RETURN                  ;;RETURN HERE THE RANDOM
3331                                  ;*                              ;;NUMBER WILL BE IN
3332                                  ;*                              ;;$HINUM,$LONUM
3333
3334  025746                          $RAND:
3335  025746  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3336  025750  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3337  025752  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3338  025754  013700  026046                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
3339  025760  013701  026044                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
3340  025764  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
3341  025770  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
3342  025772  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
3343  025774  005202                          INC     R2              ;;CHECK FOR DONE
3344  025776  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
3345  026000  063700  026046                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
3346  026004  005501                          ADC     R1              ;;PROPOGATE CARRY
3347  026006  063701  026044                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
3348  026012  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
3349  026016  005501                          ADC     R1              ;;PROPOGATE CARRY
3350  026020  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
3351  026024  010037  026046                  MOV     R0,$LONUM       ;;SAVE R0
3352  026030  010137  026044                  MOV     R1,$HINUM       ;;SAVE R1
3353  026034  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
3354  026036  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3355  026040  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3356  026042  000207                          RTS     PC              ;;RETURN
3357  026044  176543          $HINUM: .WORD   176543
3358  026046  123456          $LONUM: .WORD   123456
3359                                  .SBTTL  TRAP DECODER
3360
3361                                  ;;**************************************************************************
3362                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
3363                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3364                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3365                                  ;*GO TO THAT ROUTINE.
3366
3367  026050  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
3368  026052  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
3369  026056  005740                  TST     -(R0)           ;;BACKUP BY 2
```

```
3370  026060  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
3371  026062  006300                        ASL     R0              ;;POSITION FOR INDEXING
3372  026064  016000  026104                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
3373  026070  000200                        RTS     R0              ;;GO TO ROUTINE
3374
3375
3376                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
3377
3378  026072  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
3379  026074  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
3380  026102  000002                        RTI                     ;;RESTORE THE PSW
3381
3382                                .SBTTL  TRAP TABLE
3383
3384                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3385                                ;*BY THE ''TRAP'' INSTRUCTION.
3386
3387                                ;       ROUTINE
3388                                ;       -------
3389  026104  026072                $TRPAD: .WORD   $TRAP2
3390  026106  025300                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
3391  026110  025544                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3392  026112  025520                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3393  026114  025560                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
3394  026116  013324                        $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
3395
3396
3397  026120  024246                        $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
3398  026122  024366                        $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
3399  026124  024540                        $RDOCT  ;;CALL=RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
3400                                .SBTTL  POWER DOWN AND UP ROUTINES
3401
3402                                ;;**************************************************************************
3403                                ;POWER DOWN ROUTINE
3404  026126  012737  026266  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
3405  026134  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
3406  026142  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3407  026144  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3408  026146  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3409  026150  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3410  026152  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
3411  026154  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3412  026156  017746  152756                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STA K
3413  026162  010637  026272                MOV     SP,$SAVR6       ;;SAVE SP
3414  026166  012737  026200  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
3415  026174  000000                        HALT
3416  026176  000776                        BR      .-2             ;;HANG UP
3417
3418                                ;;**************************************************************************
3419                                ;POWER UP ROUTINE
3420  026200  012737  026266  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
3421  026206  013706  026272                MOV     $SAVR6,SP       ;;GET SP
3422  026212  005037  026272                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
3423  026216  005237  026272        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
3424  026222  001375                        BNE     1$              ;;CF  WORD
3425  026224  012677  152710                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
```

```
3426  026230  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
3427  026232  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
3428  026234  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
3429  026236  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
3430  026240  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3431  026242  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3432  026244  012737  026126  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
3433  026252  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
3434  026260  104401                          TYPE                    ;REPORT THE POWER FAILURE
3435  026262  026274                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
3436  026264  000002                          RTI
3437  026266  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
3438  026270  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
3439  026272  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
3440  026274  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER'
3441  026302  000122
3442                                          .EVEN
3443  026304  000010                  HOLD:   .BLKW   10
3444  026324  000010                  GOOD.   .BLKW   10
3445  026344  000000                  RETRY:  .WORD   0
3446  026346  000000                  BUFFFR: 0
3447          000001                          .END
```

```
A       007432  2030#
ADELAY  011402  1009  1908  2418#
AKPTAB  022416  2134  2921#
AKPTB1  022542  2664  2720  2921#
AKPTB2  022640  2654  2707  2921#
AMSG    011454   990   992  1031  1108  1119  1140  1164  1188  1276  1280  1287  1374  1418
                1451  1454  1476  1484  1568  1570  1577  1582  1595  1598  1617  1619  1621
                1626  1631  1650  1652  1664  1672  1677  1686  1699  1707  1718  1723  1726
                1734  1746  1751  1754  1762  1779  1788  1794  1801  1813  1815  1826  1836
                1842  1852  1854  1866  1951  1981  2030  2035  2038  2043  2050  2053  2058
                2063  2070  2075  2078  2081  2086  2091  2094  2097  2101  2106  2109  2117
                2119  2122  2128  2130  2132  2136  2140  2148  2287  2290  2436#  2471  2507
                2628  2671  2674  2679  2730  2733  2800
ANESC   010674  2201*  2203*  2207  2283#
AWAS    022746  2680  2921#
AXOFF   010666  1566*  1594*  2231*  2280#
B       007474  2042#
BADKEY  023046  2731  2762  2921#
BADMNE  013014  2700  2717  2730#
BEGIN   001334   757   887#
BEGIN1  001410   759   898#
BEGIN2  001420   760   900#
BEGIN3  001400   761   896#
BFCHAR  007336  1968  2006#
BFCHR   007276  1962  1968  1987#
BIT0  = 000001   721#  2533  2547
BIT00 = 000001   711#   721
BIT01 = 000002   710#   720
BIT02 = 000004   709#   719
BIT03 = 000010   708#   718
BIT04 = 000020   707#   717
BIT05 = 000040   706#   716
BIT06 = 000100   705#   715
BIT07 = 000200   704#   714
BIT08 = 000400   703#   713  3080
BIT09 = 001000   702#   712  1613  2024  2033  2048  2073  2089  2104
BIT1  = 000002   720#  2531
BIT10 = 002000   701#  1644  1810
BIT11 = 004000   700#  1048  1642
BIT12 = 010000   699#  2115  2404
BIT13 = 020000   698#  1239  1252  1471  3112
BIT14 = 040000   697#  1639  3066
BIT15 = 100000   696#  1560
BIT2  = 000004   719#  2528
BIT3  = 000010   718#
BIT4  = 000020   717#
BIT5  = 000040   716#
BIT6  = 000100   715#
BIT7  = 000200   714#  2530  2535  2551  2555
BIT8  = 000400   713#
BIT9  = 001000   712#
BPTVEC= 000014   728#
BUFFER  026346  1193  1232  1290  1354  1383  1428  1478*  1479*  1486*  1487*  1496  1503*  1514
                1533*  1534*  1535*  2162  2174  2191  2341  2361  2377  2913  3446#
BYPASS  011602  2470  2473#
C       007510  2047#
```

```
CONT    012404          2620    2623#
COPTST  005362          1615    1634#
CR    = 000015           636#   3235    3245
CRLF  = 000200           637#   3206    3245
CRLFA   017234          1599    1724    1752    2921#
CRLFB   017223          1277    1281    1569    1571    2921#
D       007600          2070#
DCATST  004462          1480    1488    1492#
DDISP = 177570           643#    791     926
DELAY   011274          1057    1113    1135    1152    1158    1177    1183    1227    1275    1348    1379    1412    1422
                        1445    1459    1481    1489    1607    1633    1660    1666    1679    1701    1728    1756    1781
                        1803    1828    1845    1862    1868    2392#
DELIM   012312          2576    2586    2593#
DH1     021325           827     833    2921#
DH3     021354           839    2921#
DH4     021423           845    2921#
DIG0    011564          1956    2462*   2465#   2488    2500    2782    2789    2796
DIG1    011566          1957    2455*   2466#   2489    2501    2783    2790    2797
DIG2    011570          1958    2448*   2467#   2490    2502    2784    2791    2798
DISPCH  006616          1792    1871#
DISPLA  001142           791#    926*    934*   3087*   3107*
DISPRE  000174           754#    934
DSPCH   011006          2303    2307    2308    2312#
DSWR  = 177570           642#    790     925
DTPSR   011166          1123    1689    2361#
DTPSRA  011214          2368#   2371
DTPSRB  011172          1127    1692    2362#
DT1     021472           828     834    2921#
DT3     021502           840    2921#
DT4     021516           846    2921#
E       007650          2086#
EMTVEC= 000030           731#    913*    914*
EM1     021102           826    2921#
EM2     021147           832    2921#
EM3     021176           838    2921#
EM4     021222           844    2921#
EM5     021265           850    2921#
ENAKP   022301          2133    2921#
ENGRAF  022313          1622    2921#
ENPNTM  022325          1816    2921#
ERCHR   022766          2629    2921#
ERRVEC= 000004           724#    923     924*    935*   3071    3072*   3074*   3077*
EXAKP   022306          2123    2137    2921#
EXGRAF  022320          1632    2921#
EXPDKY  023011          2672    2752    2921#
EXPNTM  022332          1827    2921#
F       007720          2101#
FILBFA  010164          2165#   2167
FILBFB  010150          1339    2162#
FILBUF  010144          1146    1657    1712    1765    1775    1859    2161#   2876
FILLWC  013550          1111    1377    1420    1674    1839    2875#
FINDOT  010706          2253    2289#   2293
FINDTA  010716          2249*   2267*   2291#
FIRLST  011072          1770    2341#
FIRST   001242           854#    951*    952     954     968
FNDA    010700          2287#   2296    2298    2304
```

```
GBBUF   011232            1624    2377#
GENER   004546            1505#   1509    1511    1532    1539
GENRX   004632            1516    1520#   1524    1526
GETCHR  013612             995     998    1001    1953    2151    2292    2474*   2568*   2579*   2589*   2889#
GIN     001426             895     897     899     901#
GINA    001434             891     902#
GNS   = ****** U           753    3390    3391    3392    3393    3394    3397    3398    3399
GOAPMD  020614            1653    1855    2921#
GODSHS  020566            1596    2921#
GOHDSC  020540            1583    2921#
GOTAPM  020621            1665    1867    2921#
GOOD    026324            2595    2616    2634    2676    3444#
GOPRNT  020512            1678    1700    1719    1727    1747    1755    1780    1802    1843    2921#
GRPHST  005266            1614    1616#
HOLD    026304            2566    2615    2684    3443#
HOMERS  017243            1618    2921#
HT    = 000011             634#   3204    3245
IGNORE  010676             937*   1980*   2218    2252*   2255    2273*   2284#   2627*   2673*   2678*   2799*
INSBAD  013104            2699    2716    2729    2759#
INSMNE  013050            2647    2660    2670    2749#
INSTR   023134            2120    2131    2921#
INVLID  016676             966    2921#
IOTVEC= 000020             729#    911*    912*    939*
KPTAB   022340            2124    2921#
KPTBO   022524            2639    2690    2921#
KRBDON  010074            2116    2127    2140#
KRBECH  010110             896    2147#
KRBECO  007110             900    1950#
KRBTST  007404             898    2024#   2142
LAST    001244             856#    955*    959*    960    1895    1897
LASTCH  001310             875#   1047*   1050*   1155    1180    1721    1749    1824    2180
LASTLN  001262             864#   1036*   1041*   1510
LBMT    002774            1193#
LBMT1   003144            1211    1226#
LCM     003154            1231#
LF    = 000012             635#   3239    3245
LIC     010200            1170    1739    1819    2174#
LICA    010216            2178#   2184
LICB    010234            2181    2183#
LODERL  004016            1383#
LODERS  004174            1426#
LOOP    010660             938*   2248*   2266*   2277#   3050
LOOP1   012460            2641#   2644
LOOP2   012520            2656#   2658
LOOP3   012552            2666#   2668
LOOP4   012662            2692#   2697
LOOP5   012730            2709#   2714
LOOP6   012770            2722#   2727
LRL     003436            1290#
LTAB    003706            1306    1310    1314    1318    1322    1326    1330    1353#
MANFU   001360             762     892#
MASK1   012004            2513#   2526*   2530*   2538
MASK2   012006            2497    2514#   2527*   2535*   2537    2551*   2555*
MKB     014774            2031    2921#
MKBA    015071            2472    2921#
MKBB    015160            2039    2079    2095    2110    2921#
```

MKBB2	015275	2036	2076	2092	2107	2921#		
MKBC	015361	2044	2921#					
MKBD	015421	2054	2921#					
MKBD2	015526	2051	2921#					
MKBE	015615	2059	2921#					
MKBF	015756	2064	2921#					
MKBG	016013	2071	2921#					
MKBGA	016062	2087	2921#					
MKBH	016133	2102	2921#					
MKBI	016175	2921#						
MKBJ	016230	2921#						
MKBK	016263	2921#						
MKBL	016317	2921#						
MKBM	016352	2921#						
MKBN	016406	2118	2921#					
MKBQ	016532	2508	2921#					
MKBQA	016577	2500*	2501*	2502*	2921#			
MKBQB	016616	2488*	2489*	2490*	2921#			
MKBQ1	016573	2504*	2921#					
MKBQ2	016612	2495*	2921#					
MKBR	016625	2141	2921#					
MKB1	015032	2082	2098	2921#				
MKB52	016463	2129	2921#					
MKE	016744	1952	2921#					
MKEA	017147	1982	2921#					
MKEA1	017160	1969*	1970*	1975*	1976*	1978*	1979*	2921#
MKEB	017165	1956*	1957*	1958*	2921#			
MKEH	017174	2149	2921#					
MNEMSG	012570	2648	2661	2671#				
MORE	012756	2705	2719#					
MOVDN1	003110	1197	1214#	1217				
MOVDWN	003104	1204	1213#					
MOVRIG	003126	1198	1200	1201	1205	1207	1208	1220#
MPTCNT	014635	1814	2921#					
MPTSCN	014704	1837	2921#					
MQ0	020317	2249	2921#					
MQ1	020404	2267	2291	2921#				
MQ2	020472	2288	2921#					
MSCOPE	024700	939	3048#					
MSGTND	005076	1499	1560#					
MSGTXT	004756	1497	1546#					
MSGTYP	002436	1029	1091#					
MTEXT0	023221	1795	1871	2921#				
MTEXT1	023325	1872	2921#					
MTEXT2	023426	1873	2921#					
MTEXT3	023530	1874	2921#					
MTEXT4	023613	1875	2921#					
MTEXT5	023715	1876	2921#					
MTEXT6	024016	1877	2921#					
MTEXT7	024050	1878	2921#					
MTEXT8	024150	1879	2921#					
MTOB	013670	2437	2912#					
MULCHR	012710	2687	2702#					
M91	013745	1109	1673	2921#				
M910	014230	1375	2921#					
M911	014257	1419	2921#					

```
M912     014310            1452    2921#
M914     014343             991    2921#
M92      014013            1120    1687    2921#
M920     014412            1763    2921#
M921     014440            1789    2921#
M922     014472            1578    2921#
M9221    014525            1620    2921#
M923     014600            1651    2921#
M923A    014741            1853    2921#
M93      014041            1141    1708    2921#
M94      014101            1165    1735    2921#
M96      014131            1189    2921#
M97      014163            1288    2921#
M98      020650            1477    1485    2921#
NOTFND   013024            2693    2710    2723    2733#
NOWEOP   007050            1933    1938#
OCTAL    011474            1955    2445#   2487    2499    2781*   2788*   2795*
OUT      022731            2775*   2782*   2783*   2784*   2789*   2790*   2791*   2796*   2797*   2798*   2801    2921#
OUTPUT   013140            2677*   2682*   2772#
OVRAL    004754            1494*   1538*   1545#
PARITY   012010            2476*   2526#
PASHED   021035            1904    2921#
PATH     017251            1455    2921#
PIRQ   = 177772             641#
PIRQVE = 000240             735#
PNTWID   001316             878#   1820
PRNTST   006342            1641    1804#
PROC1    012500            2637    2649#
PROC2    012540            2652    2663#
PRTCNT   001274             869#    890*    893*    1646    1648*   1938    1940*
PR0    = 000000             658#
PR1    = 000040             659#
PR2    = 000100             660#
PR3    = 000140             661#
PR4    = 000200             662#
PR5    = 000240             663#
PR6    = 000300             664#
PR7    = 000340             665#
PS     = 177776             638#    639
PSW    = 177776             639#
PTCT     001250             858#   1648
PWRVEC = 000024             730#    917*    918*    3404*   3405*   3414*   3420*   3432*   3433*
RBEGIN   001364             758     893#
RDCHR  = 104406            2969    3397#
RDLIN  = 104407            3014    3398#
RDOCT  = 104410             950     958    3399#
RESVEC = 000010             725#
RETRY    026344            2567*   2626    3445#
REVLF    022274            1627    2921#
RFI      020312             993    2921#
ROW1     021566            2921#
ROW1C    022202            2921#
ROW1S    022110            2921#
ROW12    021736            2921#
ROW12C   022236            2921#
ROW12S   022144            2921#
```

```
ROW2     021622            2921#
ROW22    021774            2921#
ROW3     021660            2921#
ROW32    022032            2921#
ROW4     021710            2921#
ROW42    022064            2921#
ROW5     021734            2062    2921#
RSTRT    002014             970#   1901
SAVE1    001322             880#   1353*   1358
SAVE2    001324             881#   1000*   1005*   1012    2921
SAVE3    001326             882#   1003*   1006*   1018    2921
SAVE4    001330             883#    888*    901*    940     997*   1004*   1010    2921
SBEGIN   002006             941     945     963     968#   1945
SET      004752            1495*   1501*   1514*   1517*   1529*   1530    1544#
SHBE     022704            2675    2921#
STACK =  001100             629#    909    1950    2029    2147    2289
STCHAR   001306             874#   1142*   1145    1154*   1155    1166*   1169    1179*   1180    1709*   1711    1720*   1721
                           1736*   1738    1748*   1749    1817*   1818    1823*   1824
STKLMT=  177774             640#
SUBTST   001256             862#   2392    2418
SWR      001140             790#    907     925*    927     933*    944    1642    1644    2237    2404    2528    2531    2533
                           2547    3048    3066    3080    3082    3112    3117    3412    3425*
SWREG    000176             755#    933
SW0    = 000001             693#
SW00   = 000001             683#    693
SW01   = 000002             682#    692
SW02   = 000004             681#    691
SW03   = 000010             680#    690
SW04   = 000020             679#    689
SW05   = 000040             678#    688
SW06   = 000100             677#    687
SW07   = 000200             676#    686
SW08   = 000400             675#    685
SW09   = 001000             674#    684
SW1    = 000002             692#
SW10   = 002000             673#
SW11   = 004000             672#
SW12   = 010000             671#
SW13   = 020000             670#
SW14   = 040000             669#
SW15   = 100000             668#
SW2    = 000004             691#
SW3    = 000010             690#
SW4    = 000020             689#
SW5    = 000040             688#
SW6    = 000100             687#
SW7    = 000200             686#
SW8    = 000400             685#
SW9    = 001000             684#
TBITVE=  000014             726#
TB1MNE   022560            2646    2669    2698    2728    2921#
TB2MNE   022650            2659    2715    2921#
TEMP     001312             876#   1121*   1132*   1453*   1457*   1688*   1696*   1768*   1769*   1772*
TEMP0    001314             877#   1144*   1150*   1153*   1168*   1175*   1178*   1623*   1629*   1655*   1661*   1710*   1714*
                           1716*   1737*   1742*   1744*   1857*   1863*
TIME0    001254             861#   2209    2889
```

```
TIME1   013664          2209* 2215* 2889* 2900* 2905#
TIME2   013666          2210* 2213* 2890* 2898* 2906#
TITLE   013704           943  2921#
TKVEC = 000060           733#
TOTALC  001264           865# 1034* 1039* 1494
TPVEC = 000064           734#
TRAPVE= 000034           732#  915*  916*
TRTVEC= 000014           727#
TSTDAT  012360          2600* 2615#
TSTKPS  012202          2125* 2135* 2561#
TSTNUM  001252           859# 2921  3104*
TSTROW  011572          2040  2045  2055  2060  2066  2080  2096  2111  2470#
TST1    002042           983# 2312
TST10   003776          1349  1373# 2319
TST11   004154          1417# 2320
TST12   004302          1450# 2321
TST13   004340          1464# 2322
TST14   005100          1473  1565# 2323
TST15   005250          1574  1576  1612# 2324
TST16   005362          1638# 2325
TST17   005532          1671# 2326
TST2    002466           887   894  1059  1106# 1900  1941  2313
TST20   005566          1635# 2327
TST21   005656          1706# 2328
TST22   006010          1733# 2329
TST23   006144          1761# 2330
TST24   006266          1786# 2331
TST25   006344          1809# 2332
TST26   006456          1833# 2333
TST27   006512          1850# 2334
TST3    002506          1118# 2314
TST30   007106          1949# 2335
TST31   007402          2022# 2336
TST32   010106          2146# 2337
TST4    002562          1139# 2315
TST5    002662          1163# 2316
TST6    002764          1187# 2317
TST7    003426          1279  1286# 2318
TYPDS = 104405          1922  1943  3394#
TYPE  = 104401           942   948   956   965  1904  1920  1923  1944  2863  2973  2976  2980  3043
                        3045  3116  3135  3152  3154  3157  3159  3163  3170  3209  3306  3390# 3434
TYPEPT  002404          1018  1021  1028  1075#
TYPOC = 104402          3143  3167  3391#
TYPON = 104404          3393#
TYPOS = 104403          3392#
UNLKEY  023076          2734  2921#
VH0     001266           866# 1035* 1040* 1043  1144  1153  1168  1178  1386  1432  1495  1655  1710
                        1716  1737  1744  1857  2877
VH1     001270           867# 1043* 1044* 1045  1121  1195  1213  1231  1251  1453  1688  1768  2348
VH2     001272           868# 1045* 1046*
VTIB    001300           871# 2220  2740  2894
VTIS    001276           870#  970  2205  2211  2505  2892
VTNOW   001246           857#  968*  971  1897  1899* 2921
VTOB    001304           873# 2153* 2204*
VTOS    001302           872# 2154  2192  2194
VT5XX   001260           863# 1028* 1037  1048  1239  1252  1278  1393  1471  1482  1575  1613  1639
```

```
                    1810    2024    2033    2048    2073    2089    2104    2115    2126
VT50A   017520      1032    1091    2921#
VT50B   017572      1092    2921#
VT50H   017723      1094    2921#
VT52K   020001      1095    2921#
VT52L   020065      1096    2921#
VT52M   020146      1097    2921#
VT55    017641      1093    2921#
VT55E   020227      1098    2921#
V50RW   021532      2028    2921#
V52RW   021550      2026    2921#
WFTEST  001332       884#    889*    892*    1007    1474    1573    1906    2394
WHAT0   020713       949    2921#
WHAT1   020756       957    2921#
WHERE   002402       887*    894*    896*    898*    900*    1058    1059#   1900*   1941*
WIDTH   001320       879#    1171    1335    1340    1426    1740    2161    2347    2364
XADDS   004750      1527*    1534    1543#
XOFF  = 000023       863#    2226
XOFFBR  010662      1581*    1593*    2232    2250*    2274*    2278#
XOFFOK  010664      1580*    1658*    1663*    1676*    1698*    1717*    1725*    1745*    1753*    1778*    1800*    1821*    1841*
                    1860*    1865*    2229    2251*    2272*    2279#    2878*
XOFFRC  010670      1567*    1585    2228*    2281#
XON   - 000021       863#    2222
XONRC   010672      1592*    1601    2224*    2282#
XPRNT   010246      1131    1148    1173    1226    1274    1304    1308    1312    1316    1320    1324    1328    1332
                    1347    1410    1444    1537    1625    1659    1695    1713    1741    1766    1767    1771    1776
                    1777    1822    1861    2191#    2439    2879
XPRNTA  010252      2192#    2195    2206    2208    2225    2233    2262
XPRNTB  010270      2193    2197#
YADDS   004746      1512*    1533    1542#
$AUTOB  001134       787#
$BDADR  001122       782#
$BDDAT  001126       784#    2220*    2221*    2222    2226    2240    2243    2486*    2921
$CHARC  025514      3211*    3221*    3228    3237*    3242#
$CKSWR= ****** U    3397
$CMTAG  001100       770#     904     905     913     919     920
$CM1  = 000002       802#    803#    804#
$CM2  = 000004       802#    803#    804#
$CM3  = 000002       800#     802
$CNTLG  024511      2990#
$CNTLU  024504      2989#
$CRLF   001167       805#    2989    3048    3115    3126    3135    3154    3159    3163    3210    3245
$DBLK   013540      2829    2863    2871#
$DOAGN  007024      1916    1925    1931#
$DTBL   013530      2832    2867#
$ENDAD  007014      1927#    3121
$ENDCT  006762       919    1918#
$ENDMG  007033      1920    1935#
$ENULL  007030      1923    1934#    1944
$EOP    006644      1812    1870    1893#
$EOPCT  006754       919*    1915#    1919
$ERFLG  001103       773#    3058    3086*    3090    3105*    3126
$ERMAX  001115       779#    3090
$ERROR  025032       913    3103#
$FRRPC  001116       780#    2921    3109*    3110*    3111    3126    3141
$ERRTB  001172       822#    3149
```

```
$ERRTY  025144            3114    3134#
$ERTTL  001112             777#   1942    3108*   3126
$FILLC  001156             798#   3214    3245
$FILLS  001155             797#   3245
$GDADR  001120             781#
$GDDAT  001124             783#   2235*   2239*   2240    2242*   2243    2498*   2921
$GET42  007004            1924#
$GTSWR= ****** U          3396
$HD   = 000000             626
$HINUM  026044            1493*   1506    1521    3339    3347    3352*   3357#
$HIOCT  024676            3036*   3047#
$ICNT   001104             774#
$ILLUP  026266            3404    3420    3437#
$INTAG  001135             788#
$ITEMB  001114             778#   3111*   3126    3138
$LF     001170             806#   2980    2989    3048    3126    3245
$LONUM  026046            1492*   3338    3345    3351*   3358#
$LPADR  001106             775#    920*   2307*   3085*   3088    3090
$LPERR  001110             776#
$MAIL = ****** U           937    3085    3117    3198
$MNEW   024527            2993#
$MSWR   024516            2991#
$NULL   001154             796#   3216    3245
$NWTST= 000001             980#   1103#   1115#   1136#   1160#   1184#   1283#   1370#   1414#   1447#   1461#   1562#   1609#
                          1635#   1668#   1682#   1703#   1730#   1758#   1783#   1806#   1830#   1847#   1946#   2019#   2143#
$OCNT   025742            3278*   3307*   3320#
$OMODE  025744            3273*   3277*   3282    3285*   3296*   3322#
$OVER   025016            3051    3067    3078    3083    3087#
$PASS   001100             771#   1902    1912*   1913*   1921    1934
$POWER  026274            3435    3440#
$PWRDN  026126             917    3404#   3432
$PWRMG  026262            3435#
$PWRUP  026200            3414    3420#
$QUES   001166             804#   2973    2989    3045    3048    3126    3245
$RAND   025746            1505    1520    3334#
$RDCHR  024246            2937#   3397
$RDDEC= ****** U          3400
$RDLIN  024366            2965#   3398
$RDOCT  024540            3009#   3399
$RDSZ = 000010            2958#
$REGAD  001160             800#
$REG0   001162             802#
$REG1   001164             803#
$RTNAD  007026            1933#
$R2A  = ****** U          3400
$SAVRE= ****** U          3400
$SAVR6  026272            3413*   3421    3422*   3423*   3439#
$SCOPE  024714             911    3049    3065#
$SETUP= 000037             824#    910     911     913     915     917     919     920    1911    2926    2995    3066    3104
                          3120    3121
$STUP = 177777             824#
$SVLAD  025002            3075    3081    3084#
$SWR  = 160400             616#    626     741     742     743     744     745     746     804     920     921     984    1107
                          1119    1140    1164    1188    1287    1374    1418    1451    1465    1566    1613    1639    1672
                          1686    1707    1734    1762    1787    1810    1834    1851    1890    1912    1926    1932    1934
                          1950    2023    2147    3059    3060    3061    3066    3078    3080    3081    3084    3086    3090
```

```
                    3097    3098    3099    3100    3108    3112    3117    3120    3126    3436
$SWRMK= 000000       746     747    3061    3062    3082
$TKB    001146      793#    2924    2941    2947
$TKS    001144      792#    2505    2924    2939    2945
$TN   = 000033      616#     626     980     984#   1103    1107#   1115    1119#   1136    1140#   1160    1164#   1184
                   1188#    1279    1283    1287#   1349    1370    1374#   1414    1418#   1447    1451#   1461    1465#
                   1562     1566#   1574    1576    1609    1613#   1635    1639#   1668    1672#   1682    1686#   1703
                   1707#    1730    1734#   1758    1762#   1783    1787#   1806    1810#   1830    1834#   1847    1851#
                   1946     1950#   2019    2023#   2143    2147#
$TPB    001152      795#    3234*   3245
$TPFLG  001157      799#    3192    3245
$TPS    001150      794#    3232    3245
$TRAP   026050       915    3367#
$TRAP2  026072     3378#    3389
$TRP  = 000011     3382#    3391#   3392#   3393#   3394#   3395#   3397    3398#   3399#   3400#
$TRPAD  026104     3372     3389#
$TSTNM  001102      772#    1911*   2301*   3058    3082    3084*   3087    3090    3104    3107    3126
$TTYIN  024474     2966     2967    2984    2988#
$TYPBN= ****** U   3395
$TYPDS  013324     2817#    3394
$TYPE   025300     3192#    3382    3390
$TYPEC  025450     3213     3220    3227    3232#   3233
$TYPEX  025516     3238     3240    3243#
$TYPOC  025544     3276#    3391
$TYPON  025560     3275     3278#   3393
$TYPOS  025520     3271#    3392
$XTSTR  024724     3069#
$$GET4= 000000     1926#
$OFILL  025743     3272*    3276*   3286    3321#
$40CAT= ****** U   3066     3114
.     = 026350      749#     753#    769#    807     908     920    1934    1938   2871#   2921#   2924    2988#   2989
                   2995     3048    3090    3126   3173#    3245    3416    3438   3443#   3444#
```

```
ANYMOR  1883#  1894
COMMEN     1#   736#
ENDCOM     1#   736#
ERROR    630#  1023   1054   1588   1604   2196   2257   2509   2902
ESCAPE     1#   736#
GETPRI     1#   736#
GETSWR     1#   736#
MULT       1#   736#
NEWTST     1#   736#    980   1103   1115   1136   1160   1184   1283   1370   1414   1447   1461   1562   1609
        1634   1668   1682   1703   1730   1758   1783   1806   1830   1847   1946   2019   2143
POP        1#   736#   2604   2735   2779   2858   3037   3353   3425   3426
PUSH       1#   736#   2561   2630   2772   2817   3011   3334   3406   3412
REPORT     1#   736#
SCOPE    631#   983   1106   1118   1139   1163   1187   1286   1373   1417   1450   1464   1565   1612   1638
        1671   1685   1706   1733   1761   1786   1809   1833   1850   1894   1949   2022   2146
SETPRI     1#   736#
SETTRA  3382#  3391   3392   3393   3394   3397   3398   3399
SETUP      1#   736#    903
SKIP       1#   736#    996    999   1002   1051   1211   1279   1349   1472   1483   1490   1574   1576   1640
        1954   2241
SLASH      1#   736#
SPACE    736#
STARS      1#   736#    765    807    980    982   1103   1105   1115   1117   1136   1138   1160   1162   1184
        1186   1283   1285   1370   1372   1414   1416   1447   1449   1461   1463   1562   1564   1609   1611
        1635   1637   1668   1670   1682   1684   1703   1705   1730   1732   1758   1760   1783   1785   1806
        1808   1830   1832   1847   1849   1886   1946   1948   2019   2021   2143   2145   2807   2923   2929
        2958   2997   3055   3093   3129   3177   3248   3325   3361   3402   3418
SWRSU      1#   736#   921#
TRMTRP  3382#
TYPBIN     1#   736#
TYPDEC     1#   736#   1921
TYPNAM     1#   736#
TYPNUM     1#   736#
TYPOCS     1#   736#
TYPOCT     1#   736#   3141   3165
TYPTXT     1#   736#
$$CMRE   763#   802    803
$$CMTM   763#
$$ESCA     1#   736#
$$NEWT     1#   736#    980   1103   1115   1136   1160   1184   1283   1370   1414   1447   1461   1562   1609
        1635   1668   1682   1703   1730   1758   1783   1806   1830   1847   1946   2019   2143
$$SET   3382#  3391   3392   3393   3394   3397   3398   3399
$$SKIP     1#   736#   1279   1349   1574   1576
.EQUAT     1#   616#    626
.HEADE     1#   616#
.KT11      1#
.SETUP     1#   616#    824
.SWRHI     1#   616#    737
.SWRLO   747#
.$ACTI     1#
.$APTB     1#
.$APTH     1#
.$APTY     1#
.$ASTA     1#
.$CATC     1#   616#    747
.$CMTA     1#   616#    763
```

```
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#    616#   1884
.$ERRO      1#    616#   3091
.$ERRT      1#    616#   3127
.$MULT      1#
.$PARM    616#
.$POWE      1#    616#   3400
.$RAND      1#    616#   3323
.$RDDE      1#
.$RDOC      1#    616#   2995
.$READ      1#    616#   2921
.$R2AZ      1#
.$SAVE      1#    616#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#    616#   3053
.$SIZE      1#
.$SPAC    616#
.$SUPR      1#
.$SWDO    616#
.$TRAP      1#    616#   3359
.$TYPB      1#
.$TYPD      1#    616#   2805
.$TYPE      1#    616#   3175
.$TYPO      1#    616#   3246
.$40CA      1#
.1170       1#

. ABS.  026350      000

 ERRORS DETECTED:  0

 CZVTCF.BIN,CZVTCF.LST/CRF/SOL/NL:TOC=CZVTCF.SML,CZVTCF.P11
 RUN-TIME:  43 57 3 SECONDS
 RUN-TIME RATIO: 608/104=5.8
 CORE USED:  33K  (65 PAGES)
```