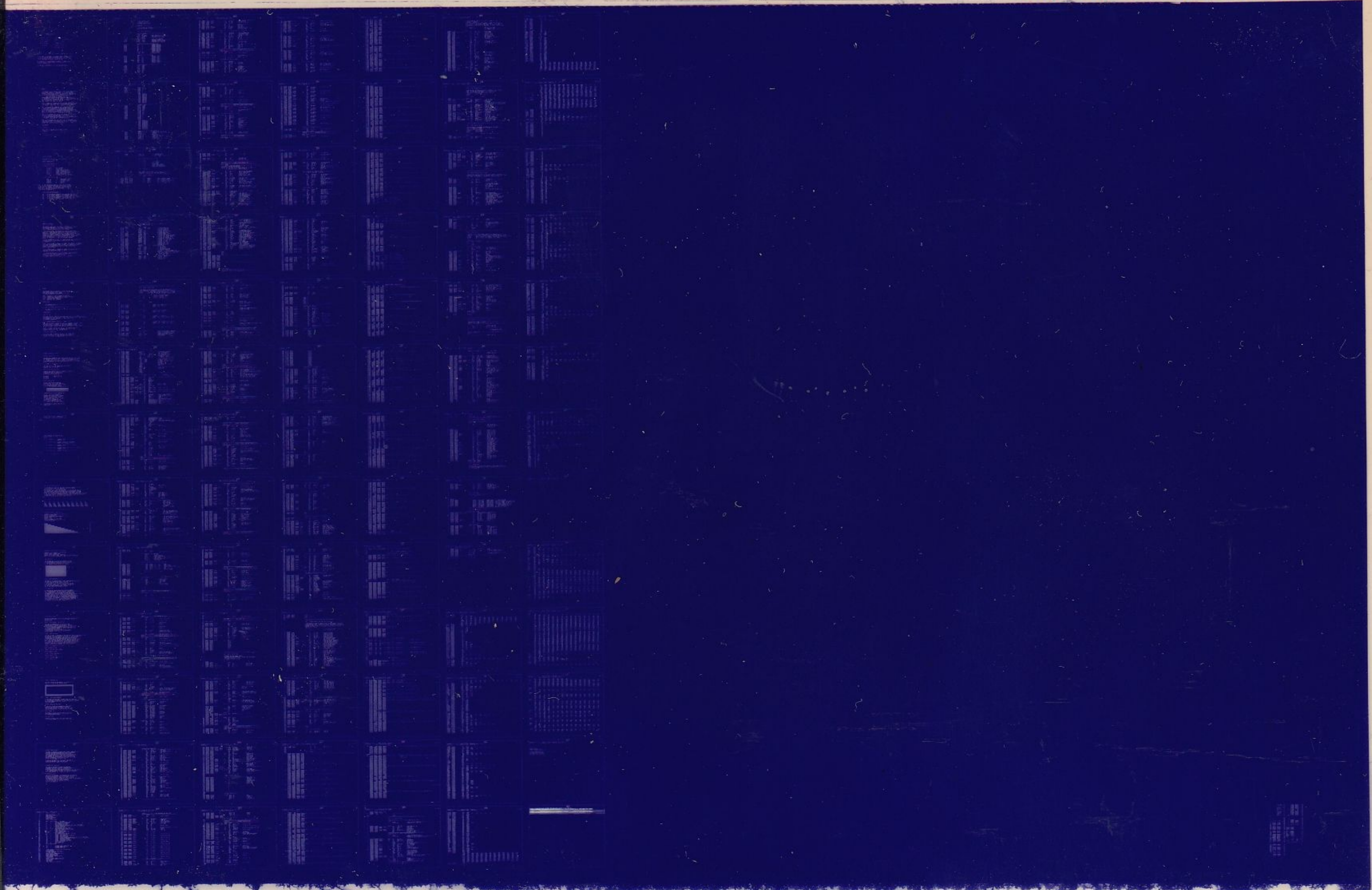


VT50A,B,H

ACCEPTANCE TEST
MD-11-DZVTC-C

EP-DZVTC-C-DL-A
COPYRIGHT 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZVTC-C
PRODUCT NAME: VT50A, B. H. 52 ACCEPTANCE TEST
DATE CREATED: MAY 21, 1975
DATE REVISED: OCTOBER, 1975
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: P NELSON/R SHOOP

COPYRIGHT (C) 1974, 1975 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

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 DL-11 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 SW 00-01.

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 EQUIVALENT (IE. 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 DL-11A-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 A 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 VT-50 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 SW 07 AT THE BEGINNING OF THE PROGRAM. WHEN SW 07 IS A ZERO, THE PROGRAM IS UNDER SWITCH REGISTER CONTROL FOR TEST PATTERN SELECTION. IF 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 DL-11'S TO WHICH VT-50'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 "\" RESPECTFULLY.

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

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

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

6. 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 DL-11 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 SW 15 AND 13 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' ON THE CONSOLE 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 VT-50'S ARE BEING TESTED. THE DEFAULT IS THE CONSOLE ADDRESS <177560>
THE LOCATION "LAST" CONTAINS THE LAST DL11 ADDRESS IF SEVERAL VT-50'S ARE BEING TESTED. LOCAT. 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 SW 10 = 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)

9. PROGRAM DESCRIPTION <SCREEN>

9.1 A TERMINAL IDENTIFICATION TEST

THIS TEST WILL INTERROGATE THE VT50/52 UNDER TEST AS TO IT'S TYPE. THE RESPONSE RECIEVED IS USED TO DETERMINE THE MODES TO BE TESTED (IE. COPIER, 12/24 LINES, D.C.A <DIRECT CURSOR ADDRESSING> ETC.) IF AN UNDEFINED OR AN INCORRECT RESPONSE IS RECIEVED, THE PROGRAM ASSUMES VT50A MODEL (IE. 12 LINES, NO COPIER, NO D.C.A., NO EXTRA KEYPAD).

9.2 B FULL SCREEN OF THE LETTER E

THIS TEST WILL FILL THE SCREEN WITH THE LETTER E. THIS TEST WILL LOAD THE SCREEN RAM WITH A SINGLE CHARACTER IN ALL LOCATIONS

9.3 C DATA PATH AND RAM NOISE TEST

THIS TEST WILL PRODUCE TWO FULL SCREENS OF A WORST CASE NOISE PATTERN FOR THE SCREEN RAM AND THE DATA PATH. ALTERNATING LINES OF THE FOLLOWING PATTERNS SHOULD BE DISPLAYED.

```
*U*U*U*U*U      CODES 52 AND 125
?/?/?/?/?/?/?  CODES 77 AND 100
*U*U*U*U*U
?/?/?/?/?/?/?
```

9.4 D SIMPLE CHARACTER SET

ONE FULL LINE OF EACH CHARACTER (CODES 40 THRU 137) OF THE CHARACTER SET. THIS WILL ALSO TEST THAT ALL WORDS IN THE SCREEN RAM CAN BE LOADED. THIS WILL ALSO EXERCISE THE SCROLLING FUNCTION.

```
IE: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
```

9.5 E INCREMENTING AND SLIDING CHARACTER SET

ONE FULL LINE OF AN INCREMENTING CHARACTER SET ACROSS THE FULL SCREEN STARTING WITH THE CODE 40 (SPACE) THE NEXT LINE SHOULD BEGIN WITH THE "!" CHARACTER (CODE 41) ETC. CONTINUE THE PATTERN BY INCREMENTING THE FIRST COLUMN CHARACTER UNTIL THE FULL CHARACTER SET HAS BEEN EXHAUSTED.

THIS PATTERN WILL VERIFY THAT THE FULL CHARACTER SET CAN BE DISPLAYED IN EACH SCREEN WORD.

9.6 F CURSOR MOTION

H01

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

BEFORE:

```
-----  
+5                                     +  
+                                     +  
+                                     +  
+                                     +  
+                                     +  
+           2                         !   +  
+                                     +  
+                                     +  
+                                     +  
+                                     +  
+                                     +  
+           3                         4+  +  
-----
```

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 LEFT (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:

```
-----  
+X+                                     +  
+                                     +  
+                                     +  
+                                     +  
+                                     +  
+           X                         X   +  
+                                     +  
+                                     +  
+                                     +  
+           X                         X   +  
-----
```


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' CHAR. THE BELL WILL RING AT THE BEGINNING OF EACH LINE. THE SECOND SECTION TESTS 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          + (REF.LINE USING SPACES)
+I          I          I          I          I          I          I          I          I          I          +
+IA         IA         IA         IA         IA         IA         IA         IA         IA         IA         +
+IAA        IAA        IAA        IAA        IAA        IAA        IAA        IAA        IAA        IAA        +
+IAAA       IAAA       IAAA       IAAA       IAAA       IAAA       IAAA       IAAA       IAAA       IAAA       +
+IAAAA      IAAAA      IAAAA      IAAAA      IAAAA      IAAAA      IAAAA      IAAAA      IAAAA      IAAAA      +
+IAAAAA     IAAAAA     IAAAAA     IAAAAA     IAAAAA     IAAAAA     IAAAAA     IAAAAA     IAAAAA     IAAAAA     +
+IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    IAAAAAA    +
```

XX

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           XO (BLANK LINE)
EEEEEEEEEEEEEE  X6
EEEEEEEEEEEEEEEE X12
EEEEEEEEEEEEEEEE X18
EEEEEEEEEEEEEEEE X24
EEEEEEEEEEEEEEEE X30
EEEEEEEEEEEEEEEE X36
EEEEEEEEEEEEEEEE X42
EEEEEEEEEEEEEEEE X48
EEEEEEEEEEEEEEEE X54
EEEEEEEEEEEEEEEE X60
EEEEEEEEEEEEEEEE X66
```

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 EEEEEEE T  

T ~~~~~~ T  

T EEEEEEE T  

T ~~~~~~ T  

T EEEEEEE T  

T ~~~~~~ T  

T EEEEEEE T  

T ~~~~~~ T  

T EEEEEEE T  

T ~~~~~~ T  

T EEEEEEE T  

T ~~~~~~ T  

T EEEEEEE 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 ATTEMPT 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 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 IS USED TO DETERMINE 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

```

EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEE
EEE
EEE
EEE
EEE
EEE
EEE
EEE
EEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE

```

9.24 T COPIER - DISCLAIMER STATEMENT

IN THIS TEST THE DISCLAIMER 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 APPEARING ON SCREEN.

9.26 V PRINT SCREEN TEST

THE SCREEN IS FILLED WITH 24 LINES OF 'E'S. THE PRINT SCREEN ESCAPE SEQUENCE IS THE 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.

9.30 X 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, SW 0 AND 1 ARE USED TO INFORM THE PROGRAM OF THE EXPECTED PARITY. AN INCORRECT SWITCH SETTING WILL RESULT IN AN ERROR.

9.31 Y 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 (IE. DE=DELETE, BL=BELL, CL=CURSOR LEFT ETC.)

9.32 Z KEYBOARD ECHO LOOP

WHEN A KEY IS DEPRESSED, THE CHARACTER 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.

12		BASIC DEFINITIONS
14		OPERATIONAL SWITCH SETTINGS
15		TRAP CATCHER
(1)		STARTING ADDRESS(ES)
21		COMMON TAGS
(1)		ERROR POINTER TABLE
151		
152	SWC-4	TEST LETTER TEST NAME
154	T1	A TERMINAL IDENTIFICATION TEST
263	T2	B FULL SCREEN OF A CHARACTER
272	T3	C DATA TRANSFER PATH TEST
290	T4	D SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
311	T5	E ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
332	T6	F CURSOR MOTION TEST
428	T7	G TAB, BACKSPACE AND BELL TEST
512	T10	H ERASE FROM CURSOR TO END OF LINE
553	T11	I ERASE FROM CURSOR TO END OF SCREEN
583	T12	J VIDEO COUPLING TEST
594	T13	K DIRECT CURSOR ADDRESS TEST
680	T14	L HOLD SCREEN TEST
724	T15	M TEST GRAPHICS MODE AND REV. LINE FEED
746	T16	N COPIER - AUTO COPY TEST
776	T17	O COPIER - FULL SCREEN OF A CHARACTER
787	T20	P COPIER - DATA PATH TEST
805	T21	Q COPIER - SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
829	T22	R COPIER - ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
854	T23	S COPIER - PERIMETER PATTERN
876	T24	T COPIER - DISCLAIMER STATEMENT
896	T25	U PRINTER CONTROLLER MODE TEST
917	T26	V PRINT SCREEN TEST
931	T27	W AUTO PRINT TEST
983		END OF PASS ROUTINE
993		
995	T30	X KEYBOARD OCTAL VALUE LOOP
1062	T31	Y KEYBOARD CHARACTER TEST
1211	T32	Z KEYBOARD ECHO LOOP
1223		
1616		CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1659		ASCII MESSAGES
1773		KEYBOARD CHARACTER CODE TABLES
1841		TTY INPUT ROUTINE
1842		READ AN OCTAL NUMBER FROM THE TTY
1848		SCOPE HANDLER ROUTINE
1850		ERROR HANDLER ROUTINE
1852		ERROR MESSAGE TYPEOUT ROUTINE
1854		TYPE ROUTINE
1856		BINARY TO OCTAL (ASCII) AND TYPE
1857		RANDOM NUMBER GENERATOR ROUTINE
1858		TRAP DECODER
(3)		TRAP TABLE
1859		POWER DOWN AND UP ROUTINES

(1) 000100
(1) 000040
(1) 000020
(1) 000010
(1) 000004
(1) 000002
(1) 000001

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)

(1) 100000
(1) 040000
(1) 020000
(1) 010000
(1) 004000
(1) 002000
(1) 001000
(1) 000400
(1) 000200
(1) 000100
(1) 000040
(1) 000020
(1) 000010
(1) 000004
(1) 000002
(1) 000001

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

(1) 000004
(1) 000010
(1) 000014
(1) 000014
(1) 000014
(1) 000020
(1) 000024
(1) 000030
(1) 000034

ERRVEC= 4 : TIME OUT AND OTHER ERRORS
RESVEC= 10 : RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC= 14 : "T" BIT
TRTVEC= 14 : TRACE TRAP
BPTVEC= 14 : BREAKPOINT TRAP (BPT)
IOTVEC= 20 : INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24 : POWER FAIL
EMTVEC= 30 : EMULATOR TRAP (EMT) **ERROR**
TRAPVEC= 34 : "TRAP" TRAP


```

(1) 000060 TKVEC= 60 ::TTY KEYBOARD VECTOR
(1) 000064 TPVEC= 64 ::TTY PRINTER VECTOR
(1) 000240 PIRQVEC=240 ::PROGRAM INTERRUPT REQUEST VECTOR

```

```

13
14
(1) .SBTTL OPERATIONAL SWITCH SETTINGS
(1) :*
(1) :* SWITCH USE
(1) :* -----
(1) :* 15 HALT ON ERROR
(1) :* 14 LOOP ON TEST
(1) :* 13 INHIBIT ERROR TYPEOUTS
(1) :* 12 INHIBIT SUB-TEST DELAY'S
(1) :* 10 ENABLE SAVE COPIER PAPER MODE
(1) :* 8 LOOP ON TEST IN SWR<7:0>

```

```

15
(1) .SBTTL TRAP CATCHER
(1)
(1) 000000
(1) .=0
(1) :*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2.HALT"
(1) :*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1) :*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

```

```

(1) .SBTTL STARTING ADDRESS(ES)
(1) .=200
(1) 000200 000137 001336 JMP Q#BEGIN ::JUMP TO STARTING ADDRESS OF PROGRAM
16 000204 000137 001402 JMP RBEGIN ::JUMP TO RESTART ADDRESS
17 000210 000137 001426 JMP BEGIN1 ::JUMP TO KEYBOARD CHARACTER TEST
18 000214 000137 001436 JMP BEGIN2 ::JUMP TO CHAR OCTAL VALUE LOOP
19 000220 000137 001416 JMP BEGIN3 ::JUMP TO ASCII ECHO LOOP
20 000224 000137 001376 JMP MANFL ::JUMP TO W.F. SPECIAL TEST PARAM.

```


(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61

001170

001170 017664
001172 020107
001174 020254
001176 000000

001200 017731
001202 020107
001204 020254
001206 000000

001210 017760
001212 020136
001214 020264
001216 000000

001220 020004
001222 020205
001224 020300
001226 000000

001230 020047
001232 000000
001234 000000
001236 000000
001240 177560

001242 000000
001244 177560
001246 100011
001250 000000

001252 000300

```

*****
.SBTTL  ERROR POINTER TABLE

:*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
:*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
:*LOCATION SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
:*NOTE1:      IF SITEMB IS 0 THE ONLY PERTINENT DATA IS ($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

$ERRTB:

:ITEM  1
        EM1            ;;ERROR FLAG ON HOST TRANSMIT STATUS
        DH1            ;;ERRPC VTNOW  TSTNUM
        DT1            ;;$ERRPC VTNOW  TSTNUM
        0

:ITEM  2
        EM2            ;;NO HOST INPUT FLAG RECEIVED
        DH1            ;;ERRPC VTNOW  TSTNUM
        DT1            ;;$ERRPC VTNOW  TSTNUM
        0

:ITEM  3
        EM3            ;;INCORRECT I.D. RESPONSE
        DH3            ;;ERRPC VTNOW  1ST WD  2ND WD  3RD WD
        DT3            ;;$ERRPC VTNOW  SAVE4  SAVE2  SAVE3
        0

:ITEM  4
        EM4            ;;INCORRECT OR UNEXPECTED INPUT CHARACTER
        DH4            ;;ERRPC VTNOW  TSTNUM  EXPT  RCVD
        DT4            ;;$ERRPC VTNOW  TSTNUM  $GDDAT $BDDAT
        0

:ITEM  5
        EM5            ;;INVALID BUSS ADDRESS. TRY AGAIN
        0

FIRST:  177560          ;;FIRST DEVICE ADDRESS OF SEQUENTIAL DL-11-A B TYPE DEVICE
                     ;;DEFAULT TO THE CONSOLE ADDRESS
LAST:    0              ;;LAST DEVICE ADDRESS OF DL-11-A/B TYPE
VTNOW:   177560        ;;CURRENT DEVICE BUSS ADDRESS
PTCT:    100011        ;;COPIER PAPER SAVE COUNT (LOW BYTE)
TSTNUM:  0              ;;ERROR PATTERN

TIMED:   300           ;;CHARACTER FLAG TIMEOUT CONSTANT

```

63	001254	000005			SUBTST: 5		:SUBTEST DELAY CONSTANT
64	001256	000000			VT5XX: 0		: I.D. AND CHARASTICS
65	001260	000053			LASTLN: 53	:OR 67	:LAST VALID LINE # +40
66	001262	001700			TOTALC: 960.	:OR 1920.	:TOTAL CHARACTER COUNT
67	001264	000014			VHO: 12.	:24.	:VERTICAL LINE COUNT
68	001266	000006			VH1: 6.	:12.	:1/2 VERTICAL LINE COUNT
69	001270	000003			VH2: 3.	:6.	:1/4 VERTICAL LINE COUNT
70	001272	000000			PRTCNT: 0		
71	001274	177560			VTIS: 177560		:DEVICE ADDRESSES
72	001276	177562			VTIB: 177562		:IN DATA
73	001300	177564			VTOS: 177564		:OUT STAT
74	001302	177566			VTOB: 177566		:OUT DATA
75	001304	000000			STCHAR: 0		:TEMP REG'S
76	001306	000140			LASTCH: 140	;OR 200	:FIRST NON-VALID CHARACTER
77	001310	000000			TEMP: 0		:TEMP REG'S
78	001312	000000			TEMPO: 0		
79	001314	000204			PNTWID: 132.		:COLUMN COUNT FOR LINE PRINTER.
80	001316	000120			WIDTH: 80.		:COLUMN WIDTH
81	001320	000000			SAVE1: 0		:TEMP REG'S
82	001322	000000			SAVE2: 0		
83	001324	000000			SAVE3: 0		
84	001326	000000			SAVE4: 0		
85	001330	000000			WFTTEST: 0		:NON-ZERO IF SA = 224
86							
87	001332	022626			BUSSTR: CMP	(SP)+, (SP)+	:POP STACK
88	001334	104005			ERROR	5	:INVALID BUSS ADDRESS
89							
90	001336	012737	001760	001756	BEGIN: MOV	#TST1,WHERE	:STARTING ACCEPTANCE TEST ADDRESS
91	001344	005037	001326		CLR	SAVE4	
92	001350	005037	001330		CLR	WFTTEST	
93	001354	012737	001332	000004	MOV	#BUSSTR, @#4	
94	001362	012737	000340	000006	MOV	#340, @#6	
95	001370	005037	001272		CLR	PRTCNT	
96	001374	000426			BR	GINA	
97	001376	005237	001330		MANFU: INC	WFTTEST	:SET W.F. PARAM.
98	001402	005037	001272		RBEGIN: CLR	PRTCNT	:RESTART ADDRESS
99	001406	012737	001760	001756	MOV	#TST1,WHERE	
100	001414	000413			BR	GIN	
101	001416	012737	010172	001756	BEGIN3: MOV	#KRBECH,WHERE	:START AT ECHO LOOP
102	001424	000407			BR	GIN	
103	001426	012737	007316	001756	BEGIN1: MOV	#KRBTST,WHERE	:STARTING CHARACTER TEST ADDRESS
104	001434	000403			BR	GIN	
105	001436	012737	007022	001756	BEGIN2: MOV	#KRBECH,WHERE	:STARTING CHARACTER LOOP ADDRESS
106	001444	012737	000001	001326	GIN: MOV	#1,SAVE4	
107	001452	000005			GINA: RESET		
108	001454	012737	000340	177776	MOV	#340, @#PS	:LOCK OUT ALL INTERRUPTS
109	(1) 001462	012706	001104		MOV	#SCMTAG, R6	:FIRST LOCATION TO BE CLEARED
110	(1) 001466	005026			CLR	(R6)+	:CLEAR MEMORY LOCATION
111	(1) 001470	022706	001132		CMP	#BDDAT, R6	:DONE?
112	(1) 001474	001374			BNE	.-6	:LOOP BACK IF NO
113	(1) 001476	012706	001100		MOV	#STACK, SP	:SETUP THE STACK POINTER
114	(1) 001502	012737	022724	000020	MOV	#SCOPE, @#IOTVEC	:IOT VECTOR FOR SCOPE ROUTINE
115	(1) 001510	012737	000340	000022	MOV	#340, @#IOTVEC+2	:LEVEL 7
116	(1) 001516	012737	023042	000030	MOV	#ERROR, @#EMTVEC	:EMT VECTOR FOR ERROR ROUTINE
117	(1) 001524	012737	000340	000032	MOV	#340, @#EMTVEC+2	:LEVEL 7
118	(1) 001532	012737	024110	000034	MOV	#STRAP, @#TRAPVEC	:TRAP VECTOR FOR TRAP CALLS

```

(1) 001540 012737 000340 000036      MOV      #340, @#TRAPVEC+2; LEVEL 7
(1) 001546 012737 024152 000024      MOV      #SPWADN, @#PWRVEC ;: POWER FAILURE VECTOR
(1) 001554 012737 000340 000026      MOV      #340, @#PWRVEC+2 ;: LEVEL 7
(1) 001562 013737 006674 006666      MOV      SENDCT, @#EOPCT ;: SETUP END-OF-PROGRAM COUNTER
(1) 001570 012737 001570 001112      MOV      #, @#SLPADR. ;: INITIALIZE THE LOOP ADDRESS FOR SCOPE
113 001576 005037 010760      CLR      IGNORE
114 001602 005037 010742      CLR      LOOP
115 001606 012737 022710 000020      MOV      #MSCOPE, @#ICTVEC
116 001614 005737 001326      TST      SAVE4 ;: TEST FLAG
117 001620 001036      BNE      SBEGIN ;: BR IF NON-ZERO
118 001622 104400      TYPE
119 001624 012630      TITLE
120 001626 105777 177246      TSTB     @#SWR
121 001632 000031      BPL      SBEGIN ;: BR IF CLEARED
122 001634 104400      115:    TYPE
123 001636 017475      WHATD ;: FIND OUT THE DEVICE ADDRESS
124 001640 104407      RDOCT
125 001642 012637 001240      MOV      (SP)+, FIRST ;: SAVE THE ADDRESS
126 001646 022737 160000 001240      CMP      #160000, FIRST
127 001654 101367      BHI     115 ;: BR IF INVALID
128 001656 005777 177356      TST      @#FIRST ;: TEST IF VALID
129 001662 005037 001242      CLR      LAST
130 001666 104400      TYPE
131 001670 017540      WHAT1 ;: FIND OUT THE LAST ADDRESS
132 001672 104407      RDOCT
133 001674 012637 001242      MOV      (SP)+, LAST ;: SAVE LAST ADDRESS
134 001700 005777 177336      TST      @#LAST ;: TEST IF VALID
135 001704 012737 000006 000004      MOV      #5, @#4
136 001712 005037 000006      CLR      @#6
137 001716 013737 001240 001244 SBEGIN: MOV      FIRST, VTNOW ;: LOAD INITIAL DEVICE ADDRESS
138
139 001724 012700 001274      RSTRT: MOV      #VTIS, RO ;: LOAD POINTER
140 001730 013701 001244      MOV      VTNOW, R1 ;: LOAD INPUT STAT
141 001734 010120      MOV      R1, (RO)+
142 001736 005721      TST      (R1)+
143 001740 010120      MOV      R1, (RO)+
144 001742 005721      TST      (R1)+
145 001744 010120      MOV      R1, (RO)+
146 001746 005721      TST      (R1)+
147 001750 010110      MOV      R1, (RO)
148 001752 000177 000000      JMP      @#WHERE ;: JUMP TO STARTING ADDRESS
149 001756 001760      WHERE: TST1
154
(3)
(3)
(2) 001760 000004      *TEST 1 A TERMINAL IDENTIFICATION TEST
155 001762 004537 011534      TST1: SCOPE
156 001766 013342      JSR      R5, @#MSG ;: DISPLAY HEADER
157 001770 004537 011534      M914
158 001774 017070      JSR      R5, @#MSG ;: SEND REQUEST FOR IDENTIFICATION
159
160 001776 004737 012536      RFI
161 002002 000537      JSR      PC, @#GETCHR ;: GET A CHARACTER
162 002004 010037 001326      BR      Z$ ;: BR BACK IF NO INPUT
163 002010 004737 012536      MOV      RD, @#SAVE4 ;: SAVE RESPONSE
164 002014 000532      JSR      PC, @#GETCHR ;: GET A CHAR.
BR      Z$ ;: BR IF NO INPUT

```

MAINDEC-11-DZVTC-C
DZVTC.P13 T1MACY11 27(732) 24-AUG-76 14:41 PAGE 1-7
A TERMINAL IDENTIFICATION TEST

SEQ 0021

```

165 002016 010037 001322      MOV      RO,SAVE2
166 002022 004737 012536      JSR      PC,GETCHR      ;GET A CHAR.
167 002026 000525      BR       2$             ;;BR IF NO INPUT
168 002030 010037 001324      MOV      RO,SAVE3
169 002034 042737 177600 001326      BIC      #177600,SAVE4   ;MASK BIT 7
170 002042 042737 177600 001322      BIC      #177600,SAVE2
171 002050 042737 177600 001324      BIC      #177600,SAVE3
172 002056 005737 001330      TST      WFTST
173 002062 001402      BEQ      10$
174 002064 004737 011462      JSR      PC,ADELAY      ;EXTRA DELAY
175 002070 122737 000733 001326 10$:  CMPB    #33,SAVE4      ;TEST FIRST CHAR.
176 002076 001015      BNE     1$             ;BR TO ERROR
177 002100 122737 000057 001322      CMPB    #'/,SAVE2     ;TEST SECOND CHAR.
178 002106 001011      BNE     1$             ;BR TO ERROR
179
180 ;NOW DETERMINE WHICH VTSXX AND ITS CHARASTICS
181
182 002110 005000      CLR      RO             ;CLEAR INITIAL POINTER
183 002112 126037 002316 001324 3$:  CMPB    TYPEPT(RO),SAVE3 ;TEST I.D. TO KNOWN VALLE
184 002120 001406      BEQ     4$             ;BR IF CORRECT
185 002122 005720      TST     (RO)+          ;BUMP THE INDEX
186 002124 105760 002316      TSTB    TYPEPT(RO)     ;CHECK IF MORE VALID I.D.'S
187 002130 001370      BNE     3$            ;BR IF MORE
188 002132 104003      1$:  ERROR  3             ;INCORRECT I.D. CODE
189 002134 005000      11$:  CLR      RO         ;DEFAULT TO VTSOA MODE
190
191 ;HAVE NOW FOUND THE I.D. - REPORT TO CONSOLE
192
193 002136 016037 002316 001256 4$:  MOV     TYPEPT(RO),VTSXX ;SAVE I.D. AND CHARASTICS
194 002144 016037 002350 002160      MOV     MSGTYP(RO),5$   ;SAVE ASCII MESSAGE POINTER
195 002152 001403      BEQ     13$            ;BR IF ZERO MESSAGE POINTER
196 002154 004537 011534      JSR     R5,AMSG        ;TELL THE UUT
197 002160 016376      5$:  VTSOA ;POINTER TO VTSXX MESSAGE
198
199 002162 012737 001700 001262 13$: MOV     #960.,TOTALC    ;LOAD TOTAL CHARACTER COUNT
200 002170 012737 000014 001264      MOV     #12.,VHO       ;LOAD MAX VERTICAL LINE COUNT
201 002176 012737 000053 001260      MOV     #53,LASTLN    ;LOAD LAST LINE VALUE +40 FOR DCA TESTING
202 002204 005737 001256      TST     VTSXX         ;TEST IF 24 LINES AVAIL
203 002210 100007      BPL     6$            ;BR IF NOT
204 002212 006337 001262      ASL     TOTALC        ;ADJUST CHARACTER COUNT
205 002216 006337 001264      ASL     VHO           ;ADJUST LINE COUNT
206 002222 062737 000014 001260      ADD     #12.,LASTLN   ;ADJUST VALID LINE # +40
207
208 002230 013737 001264 001266 6$:  MOV     VHO,VH1       ;LOAD OTHER LINE COUNTS
209 002236 006237 001266      ASR     VH1
210 002242 013737 001266 001270      MOV     VH1,VH2
211 002250 006237 001270      ASR     VH2
212 002254 012737 000140 001306      MOV     #140,LASTCH   ;LOAD FIRST NON-VALID CHARACTER
213 002262 032737 004000 001256      BIT     #BIT11,VTSXX  ;TEST IF UPPER-LOWER CASE TERM.
214 002270 001403      BEQ     7$            ;BR IF NOT
215 002272 062737 000040 001306      ADD     #40,LASTCH    ;UPDATE TO ALLOW UP-LW CASE CHARACTER SET
216 002300      7$:  BR       12$
(1) 002300 000403 ;;BR AND START TESTING
217
218 002302 104002      2$:  ERROR  2             ;NO RESPONSE FROM UUT ADTER ASKING FOR IDENTIFY
219 002304 000713      BR     11$

```

220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272

002306 000240
002310 004737 011354
002314 000431

002316 000101
002320 040102
002322 140103
002324 070110
002326 135113
002330 175114
002332 137115
002334 000000
002336 000000
002340 000000
002342 000000
002344 000000
002346 000000

002350 016376
002352 016450
002354 016517
002356 016565
002360 015643
002362 016726
002364 017007
002366 000000
002370 000000
002372 000000
002374 000000
002376 000000

002400 000004
002402 004537 011534
002406 012731

002410 004737 012474
002414 004737 011354

NUP
125: JSR PC DELAY
BR TST2 ;;BR AND START TESTING

;I.D. VALUES AND CHARACTERISTICS
BIT15 = 1 24 LINES
BIT14 = 1 COPIER CONNECTED
BIT13 = 1 DIRECT CURSOR ADDRESSING (ESC Y) + "ESC-B" + "ESC-D"
BIT12 = 1 VT50H KEYPAD
BIT11 = 1 UPPER AND LOWER CASE CHARACTERS
BIT10 = 1 PRINTER CONNECTED
BIT09 = 1 VT52X MODEL

;LOW BYTE CONTAINS THE I.D. FOR EACH KNOWN VT5??
TYPEPT: .WORD 000101 ;I.D. = 101 ;VT50A
.WORD 040102 ;I.D. = 102 ;VT50B COPIER
.WORD 140103 ;I.D. = 103 ;VT55 24. LINES, COPIER
.WORD 070110 ;I.D. = 110 ;VT50H COPIER DCA VT50H KEYPAD
.WORD 135113 ;I.D. = 113 ;VT52
.WORD 175114 ;I.D. = 114 ;VT52 WITH COPIER
.WORD 137115 ;I.D. = 115 ;VT52 WITH PRINTER.
0
0
0
0
0
0
0
0
0
0

;ASCII MESSAGE POINTERS
MSGTYP: VT50A ;VT50A NO COPIER
VT50B ;VT50B COPIER
VT55 ;VT55 COPIER
VT50H ;VT50H COPIER
VT52K ;VT52
VT52L ;VT52 WITH COPIER
VT52M ;VT52 WITH PRINTER
0
0
0
0
0

;*****
;*TEST 2 B FULL SCREEN OF A CHARACTER
;*****
TST2: SCOPE

JSR R5,AMSG
M91

JSR PC,FILLWC ;FILL SCREEN WITH A 'E'S

JSR PC,DELAY

;*****

```

(3)          : *TEST 3          C          DATA TRANSFER PATH TEST
(3)          : *****
(2) 002420 000004          †TST3: SCOPE
273 002422 004537 011534          JSR      R5,AMSG          ;DISPLAY HEADING
274 002426 012777          M92
275 002430 C13737 001266 001310          MOV      VH1,TEMP          ;SET-UP A COUNTER
276
277 002436 004537 011246          JSR      R5,DTPSR          ;SET-UP BUFFER
278 002442 000077          77          ;OCTAL '?'
279 002444 000100          100         ;OCTAL 'a'
280
281 002446 004537 011252          JSR      R5,DTPSRB        ;SET-UP BUFFER
282 002452 000125          125         ;OCTAL 'U'
283 002454 000052          52          ;OCTAL '*'
284
285 002456 004737 010330          1$:      JSR      PC,XPRNT          ;DISPLAY THIS LINE
286 002462 005337 001310          DEC      TEMP          ;COMPLETED FULL COUNT?
287 002466 001373          BNE      1$          ;BRANCH IF NOT COMPLETED
288
289 002470 004737 011354          JSR      PC,DELAY          ;TEST DELAY SWITCH
290          : *****
(3)          : *TEST 4          D          SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
(3)          : *****
(2) 002474 000004          †TST4: SCOPE
291 002476 004537 011534          JSR      R5,AMSG          ;DISPLAY HEADING
292 002502 013026          M93
293 002504 012737 000040 001304          MOV      #40,STCHAR          ;SET-UP STARTING CHARACTER
294
295 002512 013737 001264 001312          MOV      VHO,TEMPO          ;LOAD COUNT
296 002520 013701 001304          1$:      MOV      STCHAR,R1          ;LOAD R1= TO CHARACTER
297 C02524 004737 010226          JSR      PC,FILBUF          ;LOAD A BUFFER WITH THAT CHARACTER
298
299 002530 004737 010330          JSR      PC,XPRNT          ;DISPLAY A FULL LINE FROM THE BUFFER
300
301 002534 005337 001312          DEC      TEMPO          ;DONE ?
302 002540 001005          BNE      2$          ;FINISHED
303 002542 004737 011354          JSR      PC,DELAY
304 002546 013737 001264 001312          MOV      VHO,TEMPO
305 002554 005237 001304          2$:      INC      STCHAR          ;UPDATE THE CHARACTER
306 002560 023737 001306 001304          CMP      LASTCH,STCHAR          ;TEST FOR FINAL CHARACTER
307 002566 001354          BNE      1$          ;BRANCH IF NOT COMPLETED
308
309 002570 004737 011354          JSR      PC,DELAY          ;TEST DELAY SWITCH
310
311          : *****
(3)          : *TEST 5          E          ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
(3)          : *****
(2) 002574 000004          †TST5: SCOPE
312 002576 004537 011534          JSR      R5,AMSG          ;DISPLAY HEADING
313 002602 013070          M94
314 002604 012737 000040 001304          MOV      #40,STCHAR          ;SET-UP STARTING CHARACTER
315
316 002612 013737 001264 001312          1$:      MOV      VHO,TEMPO          ;LOAD TEMP
317 002620 013701 001304          MOV      STCHAR,R1          ;LOAD R1=TO CHARACTER
318 002624 004537 010262          JSR      R5,LIC          ;LOAD A BUFFER STARTING WITH
319 002630 001316          WIDTH          ; THAT CHARACTER AND WIDTH <BYTE>

```



```

320
321 002632 004737 010330      JSR      PC,XPRNT      ;DISPLAY A FULL LINE FROM THE BUFFER
322
323 002636 005337 001312      DEC      TEMPO        ;DONE ?
324 002642 001005                BNE      Z$           ;BR IF YES
325 002644 004737 011354      JSR      PC,DELAY
326 002650 013737 001264 001312  MOV      VHI,TEMPO
327 002656 005237 001304 2$:   INC      STCHAR      ;UPDATE THE STARTING CHARACTER
328 002662 023737 001306 001304  CMP      LASTCH,STCHAR ;TEST FOR FINAL CHARACTER
329 002670 001353                BNE      IS           ;BRANCH IF NOT COMPLETED
330
331 002672 004737 011354      JSR      PC,DELAY      ;TEST DELAY SWITCH
332
333 (3)  ;*****
334 (3)  ;*TEST 6 F CURSOR MOTION TEST
335 (3)  ;*****
336 (2)  ;S*6: SCOPE
337 002676 000004                JSR      RS,AMSG      ;DISPLAY HEADING
338 002700 004537 011534      JSR      M96
339 002704 013121
340
341 ;ROUTINE TO LOAD REFERENCE LINES FOR CURSOR MOTION TEST
342
343 LBMT:  MOV      #BUFFER,RO ;LOAD POINTER
344      MOV      #65,(RO)+ ;LOAD #5
345      MOV      VHI,R1 ;LOAD R1
346      DEC      R1
347      JSR      PC,MOVDN1 ;MOVE CURSOR DOWN
348      JSR      PC,MOVRIG ;MOVE RIGHT
349      MOV      #62,(RO)+ ;LOAD #2
350      JSR      PC,MOVRIG ;MOVE RIGHT
351      JSR      PC,MOVRIG ;MOVE RIGHT
352      MOV      #40,(RO)+ ;LOAD #1
353      MOV      #61,(RO)+ ;LOAD #1
354      JSR      PC,MOVDN1 ;MOVE DOWN
355      JSR      PC,MOVRIG ;MOVE RIGHT
356      MOV      #63,(RO)+ ;LOAD #3
357      JSR      PC,MOVRIG ;MOVE RIGHT
358      MOV      #64,(RO)+ ;LOAD #4
359      MOV      #377,(RO) ;TERM
360      BR      LBMT1 ;;BR TO NEXT PART
361
362 MOVDN: MOV      VHI,R1
363 MOVDN1: MOV      #17,(RO)+ ;LOAD CR
364          MOV      #12,(RO)+ ;LOAD LF
365          DEC      R1
366          BNE      MOVDN1 ;LOOP UNTIL DONE
367          RTS      PC ;EXIT
368
369 MOVRIG: MOV      #20,R1 ;LOAD R1
370 IS:     MOV      #40,(RO)+ ;LOAD SPACES
371          DEC      R1
372          BPL      IS ;LOOP UNTIL DONE
373          RTS      PC ;EXIT
374
375 LBMT1: JSR      PC,XPRNT ;DISPLAY THIS LINE
376          JSR      PC,DELAY ;TEST DELAY SWITCH

```

423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528

```

;CURSOR MOTION SUBROUTINE
;IF VT50H TYPE - USE "ESC-D" FOR CURSOR LEFT AND USE "ESC-B" FOR CURSOR DOWN
LCM: MOV VHI,R1 ;LOAD COUNT
MOV #BUFFER,R0 ;LOAD BUFFER POINTER
1$: MOVB #33,(R0)+ ;LOAD 'ESC'
MOVB #101,(R0)+ ;LOAD 'A' CURSOR UP
DEC R1
BNE 1$ ;LOOP UNTIL DONE
MOVB #130,(R0)+ ;LOAD 'X'
MOV #4,R1 ;LOAD COUNT
BIT #BIT13,VT5XX ;TEST IF VT50H TYPE
BEQ 20$ ;BR IF NOT
2$: MOVB #33,(R0)+ ;LOAD 'ESC'
MOVB #104,(R0)+ ;LOAD 'CURSOR LEFT'
DEC R1
BPL 2$ ;LOOP UNTIL DONE
BR 21$
20$: MOVB #10,(R0)+ ;LOAD "BACKSPACE"
DEC R1 ;DONE ALL ?
BPL 20$ ;BR IF NOT
21$: MOVB #130,(R0)+ ;LOAD 'X'
MOVB #10,(R0)+ ;LOAD BACKSPACE
MOV VHI,R1 ;LOAD COUNT
BIT #BIT13,VT5XX ;TEST IF VT50H TYPE
BEQ 30$ ;BR IF NOT
3$: MOVB #33,(R0)+ ;LOAD 'ESC' CURSOR DOWN
MOVB #102,(R0)+
DEC R1
BNE 3$ ;LOOP UNTIL DONE
BR 31$
30$: MOVB #12,(R0)+ ;LOAD CURSOR DOWN (LF)
DEC R1 ;DONE ?
BNE 30$ ;BR IF NOT
31$: MOVB #130,(R0)+ ;LOAD 'X'
MOVB #10,(R0)+ ;LOAD BACKSPACE
MOV #4,R1 ;LOAD COUNT
4$: MOVB #33,(R0)+ ;LOAD 'ESC'
MOVB #103,(R0)+ ;LOAD 'C' CURSOR RIGHT
DEC R1
BPL 4$ ;LOOP UNTIL DONE
MOVB #130,(R0)+ ;LOAD 'X'
MOVB #33,(R0)+ ;LOAD 'ESC'
MOVB #110,(R0)+ ;LOAD 'H' CURSOR HOME
MOVB #377,(R0)+
JSR PC,XPRNT ;DISPLAY THIS LINE
JSR PC,DELAY ;DELAY
JSR R5,AMSG
CRLF
TST VT5XX ;TEST IF 24 LINES
BPL TST7 ;BR IF 12 LINES
JSR R5,AMSG ;SCROLL MORE LINES
CRLF

```

001256

001256

```

(3) ;*TEST 7 G TAB, BACKSPACE AND BELL TEST
(3) ;*****
(2) 003340 000004 TST7: SCOPE
429 003342 004537 011534 JSR R5,AMSG ;DISPLAY HEADING
430 003246 013154 M97
431
432 003350 012700 024312 LRL: MOV #BUFFER,R0 ;LOAD BUFFER POINTER
433 003354 112720 000007 MOVB #7,(R0)+ ;LOAD 'BELL'
434 003360 012701 000011 MOV #9,R1 ;LOAD LINE COUNT
435 003364 012702 000006 1$: MOV #6,R2 ;LOAD COLUMN COUNT
436 003370 112720 000111 MOVB #11,(R0)+ ;LOAD COLUMN MARK
437 003374 112720 000040 2$: MOVB #40,(R0)+ ;LOAD SPACE
438 003400 005302 DEC R2 ;
439 003402 100374 BPL 2$ ;BR UNTIL DONE
440 003404 005301 DEC R1 ;
441 003406 100366 BPL 1$ ;BR UNTIL FINISHED ALL TAB STOPS
442 003410 112720 000015 MOVB #15,(R0)+ ;LOAD CR
443 003414 112720 000012 MOVB #12,(R0)+ ;LOAD LF
444 003420 112720 000377 MOVB #37,(R0)+ ;LOAD TERM
445
446 003424 004737 010330 JSR PC,XPRNT
447
448 003430 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #1
449 003434 000000 0 JSR 0
450 003436 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
451
452 003442 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #2
453 003446 000001 1 JSR 1
454 003450 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
455
456 003454 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #3
457 003460 000002 2 JSR 2
458 003462 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
459
460 003466 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #4
461 003472 000003 3 JSR 3
462 003474 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
463
464 003500 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #5
465 003504 000004 4 JSR 4
466 003506 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
467
468 003512 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #6
469 003516 000005 5 JSR 5
470 003520 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
471
472 003524 004537 003620 JSR R5,LTAB ;LOAD TAB LINE #7
473 003530 000006 6 JSR 6
474 003532 004737 010330 JSR PC,XPRNT ;DISPLAY THIS LINE
475 ;NOW EXECUTE THE BACKSPACE SECTION
476
477 003536 013702 001316 MOV WIDTH,R2 ;LOAD WIDTH COUNT
478 003542 005302 DEC R2 ;ADJUST WIDTH
479 003544 005302 DEC R2 ; BY 2
480 003546 012701 000040 MOV #40,R1 ;LOAD "SPACE" INTO THE LINE
481 003552 004737 010232 JSR PC,FILBFB ;LOAD LINE

```

```

003556 013701 001316
003552 112720 000130
003566 112720 000010
003572 112720 000010
003576 005301
003600 001370
003602 112720 000377
003606 004737 010330
003612 004737 011354
003616 000434

```

```

3$: MOV WIDTH,R1 ;LOAD # OF CHARACTER POSITIONS
MOVB #'X,(R0)+ ;LOAD ASCII "X"
MOVB #'0,(R0)+ ;LOAD BACKSPACE CODE
MOVB #'0,(R0)+
DEC R1 ;DONE ALL POSITIONS ?
BNE 3$ ;BR IF NOT
MOVB #'377,(R0)+ ;LOAD TERM.
JSR PC,XPRNT ;EXECUTE ASCII CODE
JSR PC,DELAY ;TEST DELAY SWITCH
BR TST10 ;;BR TO NEXT TEST

```

;SUBROUTINE TO LOAD THE TAB TEST INTO THE BUFFER

```

003620 012537 001320
003624 012702 024312
003630 012701 000011
003634 112722 000007
003640 112722 000111
003644 013700 001320
003650 001404
003652 112722 000101
003656 005300
003662 112722 000011
003666 005301
003670 100363
003672 112722 000015
003676 112722 000012
003702 112722 000377
003706 000205

```

```

LTAB: MOV (R5)+,SAVE1 ;LOAD # OF CHARACTERS
MOV #BUFFER,R2 ;LOAD BUFFER POINTER
MOV #9,R1 ;LOAD WIDTH COUNTER
MOVB #'7,(R2)+ ;LOAD BELL AT START OF LINE
3$: MOVB #'11,(R2)+ ;LOAD 'I'
MOV SAVE1,R0 ;GET THE NO. OF THE CHAR
BEQ 1$ ;BR IF 0
2$: MOVB #'01,(R2)+ ;LOAD THE 'A' CHAR.
DEC R0
BNE 2$ ;LOOP UNTIL DONE
1$: MOVB #'1,(R2)+ ;LOAD 'TAB' CHAR
DEC R1
BPL 3$ ;BR TO NEXT TAB COLUMN CHAR
MOVB #'15,(R2)+ ;LOAD 'CR'
MOVB #'12,(R2)+ ;LOAD 'LF'
MOVB #'377,(R2)+ ;LOAD TERM
RTS R5 ;EXIT

```

```

*****
;TEST 10 H ERASE FROM CURSOR TO END OF LINE
*****

```

```

003710 000004
003712 004537 011534
003716 013222

```

```

TST10: SCOPE
JSR RS,AMSG ;DISPLAY HEADING
M910

```

```

003720 004737 012474
003724 004737 011354

```

```

JSR PC,FILLWC ;FILL BUFFER WITH A CHAR
;DISPLAY THIS LINE
JSR PC,DELAY ;DELAY

```

;LOAD ERASE LINE TEST INTO BUFFER

```

003730 012700 024312
003734 112720 000033
003740 112720 000110
003744 013737 001264 004046
003752 005337 004046
003756 112720 000033
003762 112720 000113
003766 005337 004046
003772 100427
003774 012737 000006 004050
004002 005737 001256
004006 100005
004010 006237 004050

```

```

LODEFL: MOV #BUFFER,R0
MOVB #'33,(R0)+ ;LOAD ESC
MOVB #'H,(R0)+ ;LOAD HOME
MOV VHO,2$ ;LOAD COUNT
DEC 2$
3$: MOVB #'33,(R0)+ ;LOAD ESC
MOVB #'K,(R0)+ ;LOAD ERASE LINE CHAR
DEC 2$ ;FINISHED ?
BMI 1$ ;BR WHEN DONE
MOV #6,10$ ;LOAD COUNT
TST VT5XX ;TEST IF 12 LINES
BPL 4$ ;BR IF 12 LINES
ASR 10$ ;ADJUST HORIZ. COUNT

```

```

535 004014 000240      NOP
536 004016 000240      NOP
537 004020 000240      NOP
538 004022 112720 000033 4S:  MOVB  #33,(RO)+
539 004026 112720 000103      MOVB  #'C,(RO)+      ;CURSOR RIGHT
540 004032 005337 004050      DEC   10$
541 004036 001371      SNE   4$              ;LOOP
542 004040 112720 000012      MOVB  #12,(RO)+      ;CURSOR DOWN
543 004044 000744      BR    3$
544 004046 000000      2S:  0
545 004050 000000      10$: 0
546 004052 112720 000377 1S:  MOVB  #377,(RO)+    ;LOAD TERM

004056 004737 010330      JSR   PC,XPRNT        ;DISPLAY THIS TEST
004062 004737 011354      JSR   PC,DELAY        ;TEST DELAY SWITCH

:*****
:*TEST 11      I      ERASE FROM CURSOR TO END OF SCREEN
:*****
†ST11: SCOPE
004066 000004      JSR   RS,AMSG         ;DISPLAY HEADING
004070 004537 011534      M911
004074 013253      JSR   PC,FILLWC       ;FILL BUFFER WITH A CHAR
004076 004737 012474      JSR   PC,DELAY        ;DISPLAY THIS LINE
                                ;DELAY
004102 004737 011354      JSR   PC,DELAY        ;LOAD ERASE SCREEN TEST INTO BUFFER

004106 013737 001316 004202 LODERS: MOV   WIDTH,2$      ;LOAD COUNT
004114 006237 004202      ASR   2$
004120 012700 024312      MOV   #BUFFER,RO
004124 112720 000010 1S:  MOVB  #10,(RO)+      ;LOAD CURSOR LEFT
004130 005337 004202      DEC   2$              ;DONE ?
004134 100373      BPL   1$              ;BR UNTIL DONE
004136 013737 001264 004202 MOV   VHD,2$          ;LOAD COUNT
004144 112720 000033 4S:  MOVB  #33,(RO)+      ;LOAD ESC
004150 112720 000112      MOVB  #'J,(RO)+      ;LOAD ERASE SCREEN
004154 005337 004202      DEC   2$              ;DONE ?
004160 100405      BMI   3$              ;BR WHEN DONE
004162 112720 000033      MOVB  #33,(RO)+      ;LOAD ESC
004166 112720 000101      MOVB  #'A,(RO)+      ;LOAD CURSOR UP
004172 000764      BR    4$              ;LOOP
004174 112720 000377 3S:  MOVB  #377,(RO)+    ;LOAD TERM
004200 000401      BR    5$
004202 000000      2S:  0

004204 004737 010330 5S:  JSR   PC,XPRNT        ;DISPLAY THIS TEST
004210 004737 011354      JSR   PC,DELAY        ;TEST DELAY SWITCH

:*****
:*TEST 12      J      VIDEO COUPLING TEST
:*****
†ST12: SCOPE
004214 000004      JSR   RS,AMSG         ;DISPLAY HEADER
004216 004537 011534

```

```

585 004222 013306          MY12
586 004224 013737 001266 001310  MOV    VHI TEMP      ;LOAD COUNTER
587 004232 004537 011534 1$:    JSR    RS,AMSG
588 004236 016127          PATH
589
590 004240 005337 001310          DEC    TEMP        ;FINISHED COUNT ?
591 004244 001372          BNE    1$         ;BR UNTIL DONE
592 004246 004737 011354  JSR    PC,DELAY   ;TEST DELAY SWITCH
593
594
595
596
597
598
599
600
601 004254 032737 020000 001256  BIT    #BIT13,VTSXX ;TEST IF DCA TYPE TERMINAL
602 004262 001002          BNE    3$         ;BR IF CURSOR ADDRESSING
603 004264 000137 005012 1$:    JMP    TST14     ;BYPASS THIS TEST
604 004270 005737 001330 3$:    TST    WFTST    ;TEST IF IN W.F. MODE
605 004274 001373          BNE    1$         ;BYPASS IF W.F. MODE
606 004276 004537 011534  JSR    RS,AMSG   ;ERASE SCREEN AND IDENTIFY TEST
607 004302 017427          M98
608 004304 112737 000033 024312  MOVB  #33,BUFFER  ;LOAD "ESC" CODE
609 004312 112737 000131 024313  MOVB  #'Y,BUFFER+1 ;LOAD ASCII "Y" (D.C.A. ENABLE)
610 004320 004737 004374          JSR    PC,DCATST ;RUN TEST WITH ESC Y SEQUENCE
611 004324 004737 011354          JSR    PC,DELAY  ;DELAY TESTING
612 004330 122737 000110 001256  CMPB  #'H,VTSXX  ;TEST IF VTSOH TYPE DISPLAY
613 004336 001352          BNE    1$         ;BR IF NOT VTSOH TYPE
614 004340 004537 011534  JSR    RS,AMSG   ;ERASE SCREEN AND IDENTIFY TEST
615 004344 017427          M98
616 004346 112737 000000 024312  MOVB  #0,BUFFER  ;LOAD "SPACE" AS FIRST VALUE
617 004354 112737 000016 024313  MOVB  #16,BUFFER+1 ;LOAD CODE-16 SEQUENCE
618 004362 004737 004374          JSR    PC,DCATST ;RUN TEST WITH CODE 16 SEQUENCE
619 004366 004737 011354          JSR    PC,DELAY
620 004372 000734          BR     1$         ;BR TO NEXT TEST
621
622 004374 012737 123456 024106 DCATST: MOV    #123456,$LONUM ;PRIME RANDOM NUMBER GENERATOR
623 004402 012737 176543 024104  MOV    #176543,$HINUM
624 004410 013737 001262 004666  MOV    TOTALC,OVRL ;LOAD CHAR COUNT
625 004416 013737 001264 004664  MOV    VHD,SET   ;LOAD LINE COUNT
626 004424 012700 024332          MOV    #BUFFER+20,R0 ;LOAD DESTINATION BUFFER
627 004430 012701 004670 2$:    MOV    #MSGTXT,R1 ;LOAD MESSAGE POINTER
628 004434 012120 1$:    MOV    (R1)+,(R0)+ ;LOAD 2 CHARACTERS
629 004436 022701 005010  CMP    #MSGTND,R1 ;TEST FOR LAST CHAR
630 004442 001374          BNE    1$         ;BR UNTIL DONE 1 LINE
631 004444 005337 004664          DEC    SET       ;FINISHED ALL LINES
632 004450 001367          BNE    2$         ;BR IF NOT
633 004452 012737 177777 024316  MOV    #-1,BUFFER+4 ;LOAD MESSAGE TERMINATOR
634
635 004460 004737 023760  GENER: JSR    %7,$RAND ;GENERATE RANDOM NUMBER
636 004464 013700 024106  MOV    $LONUM,R0 ;GET RANDOM NUMBER
637 004470 042700 177700  BIC    #177700,%0 ;RANDOM NO. MUST BE TWO DIGITS

```

```

*****
*TEST 13      K      DIRECT CURSOR ADDRESS TEST
*****
†ST13: SCOPE
:  RANDOM NUMBERS ARE GENERATED AND USED AS "X" AND "Y" COORDINATES
:  ADDRESSING A 960 CHARACTER PRINTOUT.
:  VERIFICATION OF DISPLAY IS PERFORMED VISUALLY BY THE USER
:  EXECUTE 1ST TIME USING "ESC-Y" SEQUENCE AND IF I.D. IS AN "H"
:  EXECUTE 2ND TIME USING "CODE 16" SEQUENCE

```

638	004474	020027	000037				CMP	%0,#37	:NO, MUST BE LESS THAN 40
639	004500	101767					BLOS	GENER	:LOWER, REGENERATION
640	004502	020037	001260				CMP	%0,LASTLN	:NO, MUST NOT BE GREATER THAN 5+ OR 67
641	004506	101364					BHI	GENER	:GREATER, REGENERATION
642	004510	010037	004660				MOV	%0,YADDS	:STORE RANDOM Y COORDINATE
643	004514	010001					MOV	%0,%1	:COPY DATA
644	004516	012737	024332	004664			MOV	#BUFFER+20,SET	:LOAD BASE POINTER
645	004524	162701	000040				SUB	#40,%1	:MINIMUM Y INDEX
646	004530	001405					BEQ	GENRX	:RESULT, MINIMUM Y COORDINATE
647	004532	062737	000120	004664	1\$:		ADD	#80.,SET	:SETUP Y INDEX LOCATION FOR PRINTOUT
648	004540	005301					DEC	%1	
649	004542	001373					BNE	1\$:Y COORDINATE IS SET
650	004544	004737	023760		GENRX:		JSR	%7,\$RAND	:GENERATE RANDOM NUMBER
651	004550	013700	024106				MOV	\$LONUM,RO	:GET A RANDOM NUMBER
652	004554	042700	177600				BIC	#177600,%0	:RANDOM NO. MAY BE LESS THAN 200
653	004560	020027	000037				CMP	%0,#37	:MUST NOT BE LESS THAN 40
654	004564	101767					BLOS	GENRX	:LOWER, REGENERATION
655	004566	020027	000157				CMP	%0,#157	:MUST NOT BE GREATER THAN 157
656	004572	101364					BHI	GENRX	:GREATER, REGENERATION
657	004574	010037	004662				MOV	%0,XADDS	:STORE RANDOM X COORDINATE
658	004600	162700	000040				SUB	#40,%0	:SETUP MINIMUM X INDEX
659	004604	060037	004664				ADD	%0,SET	:SETUP X COOR, FOR PNTOUT.
660	004610	013701	004664				MOV	SET,%1	:SETUP CHECK
661	004614	105711					TSTB	(1)	:HAS CURRENT CHAR, ALREADY BEEN USED?
662	004616	100720					BMI	GENER	:YES, REGENERATE
663	004620	113737	004660	024314			MOV#B	YADDS,BUFFER+2	:LOAD Y COORDINATE
664	004626	113737	004662	024315			MOV#B	XADDS,BUFFER+3	:LOAD X COORDINATE
665	004634	111137	024316				MOV#B	(1) BUFFER+4	:LOAD CHARACTER TO BE PRINTED
666	004640	152711	000200				BISB	#200,(1)	:INDICATE USE OF CURSOR POSITION
667	004644	004737	010330				JSR	PC,XPRNT	:EXECUTE AND PRINT CHARACTER
668	004650	005337	004666				DEC	OVRAL	:MAXIMUM NO. OF COORDINATES
669	004654	001301					BNE	GENER	:BR BACK UNTIL DONE
670	004656	000207					RTS	PC	:EXIT

671									
672	004660	000000				YADDS:	0		
673	004662	000000				XADDS:	0		
674	004664	000000				SET:	0		
675	004666	000000				OVRAL:	0		
676	004670	052126	030065	050055		MSGTXT:	.ASCII \VT50-PLUS-DIRECT-CURSOR-ADDRESSING-TEST\		
	004676	052514	026523	044504					
	004704	042522	052103	041455					
	004712	051125	047523	026522					
	004720	042101	051104	051505					
	004726	044523	043516	052055					
	004734	051505		124					
677	004737	055	044504	044507			.ASCII \-DIGITAL-EQUIPMENT-CORP.-MAYNARD-MA.-VT50\		
	004744	040524	026514	050505					
	004752	044525	046520	047105					
	004760	026524	047503	050122					
	004766	026456	040515	047131					
	004774	051101	026504	040515					
	005002	026456	052126	030065					

678 005010 100000
679
680
(3)

MSGTND: BIT15
:ONLY 12 LINE TERMINALS WILL RUN THIS TEST
:*****
:*TEST 14 L HOLD SCREEN TEST

```

(3)
(2) 005012 000004
681 005014 005037 010750
682 005020 005037 010752
683 005024 004537 011534
684 005030 016101
685 005032 004537 011534
686 005036 016101
687
688 005040 005737 001330
689 005044 001046
690 005046 005737 001256
691 005052 100443
692 005054 004537 011534
693 005060 013473
694
695 005062 012737 000001 010746
696 005070 012737 000001 010744
697 005076 004537 011534
698 005102 017317
699
700 005104 005737 010752
701 005110 001001
702
703 005112 104002
704
705
706
707 005114 005037 010754
708 005120 005037 010744
709 005124 012737 000001 010750
710 005132 004537 011534
711 005136 017345
712
713 005140 004537 011534
714 005144 016112
715
716 005146 005737 010754
717 005152 001001
718
719 005154 104002
720
721
722 005156 004737 011354
723
724
(3)
(3)
(2) 005162 000004
725 005164 032737 001000 001256
726 005172 001002
727 005174 000137 005274
728 005200 012704 000120
729 005204 004537 011534
730 005210 016121
731 005212 004537 011534

```

```

:*****
TST14: SCOPE
      CLR      AXOFF
      CLR      XOFFRC      ;CLEAR SOFT FLAG
      JSR      R5,AMSG     ;DISPLAY
      CRLF
      JSR      R5,AMSG     ;CR-LF
      CRLF
      TST      WFTST       ;TEST IF IN W.F. MODE
      BNE      TST15      ;BR IF W.F. MODE
      TST      VTSXX      ;TEST IF 12 LINE
      SMI      TST15      ;BR IF NOT 12 LINE
      JSR      R5,AMSG     ;DISPLAY MESSAGE
      M922
      MOV      #1,XOFFOK
      MOV      #1,XOFFBR   ;ENABLE XOFF
      JSR      R5,AMSG     ;TRY TO SCROLL THE SCREEN
      GOHDSO             ;ENABLE HOLD SCREEN
      TST      XOFFRC     ;TEST IF XOFF SENSED
      BNE      1$        ;BR IF SENSED
      ERROR    2         ;ENABLE HOLD SCREEN MODE FAILED TO
                        ;INHIBIT THE SCREEN FROM SCROLLING
                        ;BY SENDING "X-OFF"
      1$:  CLR      XONRC
                        ;CLEAR SOFT FLAG
      CLR      XOFFBR
      MOV      #1,AXOFF
      JSR      R5,AMSG
      GODSHS             ;DISABLE HOLD SCREEN MODE
      JSR      R5,AMSG
      CRLFA             ;TRY SCROLLING THE SCREEN
      TST      XONRC     ;TEST SOFT FLAG (X-ON)
      BNE      2$        ;BR IF SENSED
      ERROR    2         ;DISABLE HOLD SCREEN MODE FAILED TO ENABLE
                        ;THE SCREEN TO SCROLL BY SENDING AN "X-ON"
      2$:  JSR      PC,DELAY ;PROGRAM DELAY
:*****
;*TEST 15 M TEST GRAPHICS MODE AND REV. LINE FEED
:*****
TST15: SCOPE
      BIT      #BIT09,VTSXX ;IS UNIT VT52?
      BNE      GRPHST     ;YES-TEST GRAPHICS AND REV. LINE FEED
      JMP      COPTST     ;NO-GO CHECK FOR COPIER
      GRPHST: MOV      #80,R4
      1$:  JSR      R5,AMSG ;HOME THE CURSOR AND CLEAR THE SCREEN.
      HOMERS
      JSR      R5,AMSG     ;DISPLAY TEST MESSAGE

```



```

732 005216 013530 M9221
733 005220 004537 011534 JSR R5,AMSG ;ENTER GRAPHICS MODE.
734 005224 021075 ENGRAF
735 005226 012737 000045 001312 MOV #37,TEMPO ;SET UP TO XMIT 80 LINES
736 005234 004737 011312 2$: JSR PC,GBBUF ;LOAD BUFFER WITH GRAPHICS
737 005240 004737 010330 JSR PC,XPRNT ;DISPLAY 1 LINE
738 005244 004537 011534 JSR R5,AMSG ;ISSUE REV. LINE FEED AND
739 005250 021056 REVLf ;A CARRIAGE RETURN
740 005252 005304 DEC R4 ;DECREMENT BUFFER COUNT
741 005254 005337 001312 3$: DEC TEMPO ;DONE?
742 005260 001365 BNE 2$ ;NO-LOOP
743 005262 004537 011534 JSR R5,AMSG ;YES-EXIT GRAPHICS MODE
744 005266 021102 EXGRAF
745 005270 004737 011354 JSR PC,DELAY ;AND GO CHECK DELAY
746 005274
COPTST:
;*****
;*TEST 16 N COPIER - AUTO COPY TEST
;*****
†ST16: SCOPE
747 005274 000004 BIT #BIT14,VTSXX ;TEST IF COPIER IS AVAILABLE
748 005304 001002 BNE 2$ ;BR IF AVAILABLE
749 005306 000137 006254 3$: JMP PRNTST ;NOT AVAILABLE SO BYPASS COPIER TESTS
750 005312 032777 004000 173560 2$: BIT #BIT11,2SWR ;TEST IF INHIBIT COPIER TEST SWITCH IS SET
751 005320 001372 BNE 3$ ;BR IF SET
752 005322 032777 002000 173550 BIT #BIT10,2SWR ;TEST IF PAPER SAVE SWITCH IS SET
753 005330 001406 BEQ 6$ ;BR IF NOT SET AND START COPIER TESTING
754 005332 105737 001272 TSTB PRTCNT ;TEST IF TIME TO TEST COPIER
755 005336 001363 BNE 3$ ;NOT TIME TO RUN THE COPIER
756 005340 013737 001246 001272 MOV PTCT,PRTCNT ;RELOAD COUNTER AND TEST COPIER
757
758 005346 004537 011534 6$: JSR R5,AMSG
759 005352 013605 M923 ;DISPLAY HEADER
760 005354 004537 011534 JSR R5,AMSG ;ENABLE AUTO-COPY
761 005360 017373 GOAPMD
762
763 005362 013737 001264 001312 MOV VHD,TEMPO ;LOAD A EXECUTION COUNT
764 005370 012701 000101 MOV #101,R1 ;LOAD A CHARACTER
765 005374 004737 010226 JSR PC,FILBUF ;LOAD CHARACTER INTO BUFFER
766 005400 012737 000001 010746 1$: MOV #1,XOFFOK
767 005406 004737 010330 JSR PC,XPRNT ;DISPLAY IT
768 005412 004737 011354 JSR PC,DELAY ;PROGRAM DELAY
769 005416 005337 001312 DEC TEMPO ;FINISHED ?
770 005422 100366 BPL 1$ ;BR IF NOT
771 005424 012737 000001 010746 MOV #1,XOFFOK
772 005432 004537 011534 JSR R5,AMSG ;DISABLE AUTO-COPY
773 005436 017400 GONAPM
774 005440 004737 011354 JSR PC,DELAY ;ANOTHER DELAY
775
;*****
;*TEST 17 0 COPIER - FULL SCREEN OF A CHARACTER
;*****
†ST17: SCOPE
777 005444 000004 3$: JSR R5,AMSG ;DISPLAY HEADER
778 005452 012731 M91
779 005454 004737 012474 JSR PC,FILLWC ;FILL BUFFER WITH CHAR
780 ;DISPLAY IT

```

```

781 005460 012737 000001 010746
782 005466 004537 011534
783 005472 017271
784 005474 004737 011354
785
786
787
(3)
(3)
(2) 005500 000004
788 005502 004537 011534
789 005506 012777
790 005510 013737 001266 001310
791 005516 004537 011246
792 005522 000077
793 005524 000100
794 005526 004537 011252
795 005532 000125
796 005534 000052
797 005536 004737 010330
798 005542 005337 001310
799 005546 001373
800 005550 012737 000001 010746
801 005556 004537 011534
802 005562 017271
803 005564 004737 011354
804
805
(3)
(3)
(2) 005570 000004
806 005572 004537 011534
807 005576 013026
808 005600 012737 000040 001304
809 005606 013737 001264 001312
810 005614 013701 001304
811 005620 004737 010226
812 005624 004737 010330
813 005630 005337 001312
814 005634 001011
815 005636 013737 001264 001312
816 005644 012737 000001 010746
817 005652 004537 011534
818 005656 017271
819 005660 005237 001304
820 005664 023737 001306 001304
821 005672 001350
822 005674 004537 011534
823 005700 016110
824 005702 012737 000001 010746
825 005710 004537 011534
826 005714 017271
827 005716 004737 011354
828
829
(3)

```

```

MOV #1,XOFFOK ;ALLOW XOFF
JSR RS,AMSG ;COPY INST
GOPRNT
JSR PC,DELAY

:*****
;*TEST 20 P COPIER - DATA PATH TEST
:*****
†ST20: SCOPE
JSR RS,AMSG ;DISPLAY HEADING
M92
MOV VHI,TEMP ;SET-UP A COUNTER
JSR RS,DTPSR ;SET-UP BUFFER
77 ;OCTAL '?'
100 ;OCTAL '0'
JSR RS,DTPSRB ;SET-UP BUFFER
125 ;OCTAL 'U'
52 ;OCTAL '*'
1$: JSR PC,XPRNT ;DISPLAY THIS LINE
DEC TEMP ;COMPLETED FULL COUNT?
BNE 1$ ;BRANCH IF NOT COMPLETED
MOV #1,XOFFOK
JSR RS,AMSG
GOPRNT ;COPY INST
JSR PC,DELAY ;TEST DELAY SWITCH

:*****
;*TEST 21 Q COPIER - SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
:*****
†ST21: SCOPE
JSR RS,AMSG ;DISPLAY HEADING
M93
MOV #40,STCHAR ;SET-UP STARTING CHARACTER
MOV VHO,TEMPO
1$: MOV STCHAR,R1 ;LOAD R1= TO CHARACTER
JSR PC,FILBUF ;LOAD A BUFFER WITH THAT CHARACTER
JSR PC,XPRNT ;DISPLAY A FULL LINE FROM THE BUFFER
DEC TEMPO ;DONE
BNE 3$
MOV VHO,TEMPO
MOV #1,XOFFOK ;ALLOW XOFF
JSR RS,AMSG
GOPRNT ;COPY INST
3$: INC STCHAR ;UPDATE THE CHARACTER
CMP LASTCH,STCHAR ;TEST FOR FINAL CHARACTER
BNE 1$ ;BRANCH IF NOT COMPLETED
JSR RS,AMSG ;CRLF
CRLFA-2
MOV #1,XOFFOK ;ENABLE XOFF
JSR RS,AMSG ;COPY INST
GOPRNT
JSR PC,DELAY ;TEST DELAY SWITCH

:*****
;*TEST 22 R COPIER - ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
:*****

```

```

(3)
(2) 005722 000004
830 005724 004537 011534
831 005730 013070
832 005732 012737 000040 001304
833 005740 013737 001264 001312
934 005746 013701 001304
835 005752 004537 010262
836 005756 001316
937 005760 004737 010330
838 005764 005337 001312
839 005770 001011
840 005772 013737 001264 001312
841 006000 012737 000001 010746
842 006006 004537 011534
843 006012 017271
844 006014 005237 001304
845 006020 023737 001306 001304
846 006026 001347
847 006030 004537 011534
848 006034 016110
849 006036 012737 000001 010746
850 006044 004537 011534
851 006050 017271
852 006052 004737 011354

*****
†ST22: SCOPE
      JSR      R5,AMSG      ;DISPLAY HEADING
      M94
      MOV      #40,STCHAR   ;SET-UP STARTING CHARACTER
      MOV      VHO,TEMPO
1$:   MOV      STCHAR,R1    ;LOAD R1=TO CHARACTER
      JSR      R5,LIC      ;LOAD A BUFFER STARTING WITH
      WIDTH    ;THAT CHARACTER AND WIDTH (BYTE)
      JSR      PC,XPRNT    ;DISPLAY A FULL LINE FROM THE BUFFER
      DEC      TEMPO      ;DONE ?
      BNE     3$
      MOV      VHO,TEMPO
      MOV      #1,XOFFOK
      JSR      R5,AMSG
      GPRNT    ;COPY INST
3$:   INC      STCHAR      ;UPDATE THE STARTING CHARACTER
      CMP      LASTCH,STCHAR ;TEST FOR FINAL CHARACTER
      BNE     1$          ;BRANCH IF NOT COMPLETED
      JSR      R5,AMSG    ;CRLF
      CRLFA-2
      MOV      #1,XOFFOK   ;ENABLE XOFF
      JSR      R5,AMSG    ;COPY INST
      GPRNT
      JSR      PC,DELAY    ;TEST DELAY SWITCH

```

```

*****
(3)
(2)
(2)
(2) 006056 000004
855 006060 004537 011534
856 006064 013407
857 006066 012701 000105
858 006072 004737 010226
859 006076 004737 010330
860 006102 004737 010330
861 006106 013737 001266 001310
862 006114 062737 000002 001310
863 006122 004737 011152
864 006126 004737 010330
865 006132 005337 001310
866 006136 001371
867 006140 012701 000105
868 006144 004737 010226
869 006150 004737 010330
870 006154 004737 010330
871 006160 012737 000001 010746
872 006166 004537 011534
873 006172 017271
874 006174 004737 011354
875
876
(3)
(3)
(2) 006200 000004
877

```

```

*****
; *TEST 23 S COPIER - PERIMETER PATTERN
*****
†ST23: SCOPE
      JSR      R5,AMSG      ;DISPLAY HEADER
      M920
      MOV      #'E,R1      ;LOAD STARTING CHARACTER
      JSR      PC,FILBUF   ;FILL THE BUFFER
      JSR      PC,XPRNT    ;DISPLAY A LINE
      JSR      PC,XPRNT    ;DISPLAY A LINE
      MOV      VHI,TEMP    ;LOAD VERT COUNT
      ADD     #2,TEMP      ;UPDATE BY 2
1$:   JSR      PC,FIRLST   ;FILL FIRST AND LAST
      JSR      PC,XPRNT    ;DISPLAY A LINE
      DEC     TEMP        ;DONE ?
      BNE     1$
      MOV      #'E,R1      ;LOAD STARTING CHAR
      JSR      PC,FILBUF   ;LOAD LINE WITH E'S
      JSR      PC,XPRNT    ;DISPLAY A LINE
      JSR      PC,XPRNT
      MOV      #1,XOFFOK
      JSR      R5,AMSG
      GPRNT    ;COPY SCREEN
      JSR      PC,DELAY

```

```

*****
(3)
(3)
(2) 006200 000004
877
*****
; *TEST 24 T COPIER - DISCLAIMER STATEMENT
*****
†ST24: SCOPE

```

```

878 006202 004537 011534 JSR R5,AMSG ;DISPLAY HEADER
879 006206 013437 M921
880
891 006210 005004 CLR R4
882 006212 016437 006530 006226 1S: MOV DISPCH(R4),10S ;LOAD A POINTER
883 006220 001405 BEQ 2S ;BR IF DONE
884 006222 004537 011534 JSR R5,AMSG ;DISPLAY COPYRIGHT
895 006226 021352 10S: MTEXT0
885 006230 005724 TST (R4)+ ;UPDATE POINTER
897 006232 000767 BR 1S
888
889
890 006234 012737 000001 010746 2S: MOV #1,XOFFOK ;ENABLE XOFF
891 006242 004537 011534 JSR R5,AMSG ;COPY SCREEN
892 006246 017271 GOPRNT
893 006250 004737 011354 JSR PC,DELAY
894 006254 000240 PRNTST: NOP ;TRY PRINTER TESTS
895
896 ;*****
(3) ;*TEST 25 U PRINTER CONTROLLER MODE TEST
(3) ;*****
(2) †ST25: SCOPE
897 006256 000004 BIT #BIT10,VT5XX ;IS UNIT EQUIPPED WITH PRINTER.
899 006266 001002 BNE 1S ;YES-TEST IT
899 005270 000137 006556 JMP $EOP ;NO-EXIT TEST
900 006274 004537 011534 1S: JSR R5,AMSG ;DISPLAY HEADER
901 006300 013644 MPTCNT
902 006302 004537 011534 JSR R5,AMSG ;ENABLE PRINTER CONTROLLER MODE.
903 006306 021107 ENPNTM
904 006310 012737 000040 001304 2S: MOV #40,STCHAR ;LOAD INITIAL CHAR.
905 006316 013701 001304 MOV STCHAR,R1
906 006322 004537 010262 JSR R5,LIC ;BUILD A 132 COL. LINE.
907 006326 001314 PNTWID
908 006330 012737 000001 010746 MOV #1,XOFFOK ;ALLOW XOFF/XON PROTOCOL
909 006336 004737 010330 JSR PC,XPRNT ;DISPLAY IT.
910 006342 005237 001304 INC STCHAR ;INCREMENT 1ST CHAR.
911 006346 023737 001304 001306 CMP STCHAR,LASTCH ;ISSUED 92 LINES?
912 006354 001360 BNE 2S ;NO-LOOP
913 006356 004537 011534 JSR R5,AMSG ;YES-DISABLE PRINTER
914 006362 021114 EXPNTM ;CONTROLLER MODE.
915 006364 004737 011354 JSR PC,DELAY ;TEST DELAY SWITCH AND EXIT.
916
917 ;*****
(3) ;*TEST 26 V PRINT SCREEN TEST
(3) ;*****
(2) †ST26: SCOPE
918
919
920 006372 004537 011534 JSR R5,AMSG ;DISPLAY PRINT SCREEN MESSAGE
921 006376 013713 MPTSCN
922
923 006400 004737 012474 JSR PC,FILLWC ;FILL THE SCREEN WITH E
924
925 006404 012737 000001 010746 MOV #1,XOFFOK ;ALLOW XOFF/XON PROTOCOL
926 006412 004537 011534 JSR R5,AMSG ;PRINT THEM
927 006416 017271 GOPRNT

```

```

928
929 006420 004737 011354          JSR    PC,DELAY          ;TEST DELAY SWITCH.
930
931          ;*****
(3)          ;*TEST 27          W          AUTO PRINT TEST
(3)          ;*****
(2) 006424 000004          TST27: SCOPE
932
933 006426 004537 011534          JSR    R5,AMSG
934 006432 013750          M923A          ;DISPLAY HEADER
935 006434 004537 011534          JSR    R5,AMSG          ;ENABLE AUTO-PRINT
936 006440 017373          GOAPMD
937
938 006442 013737 001264 001312          MOV    VHD,TEMPO          ;LOAD A EXECUTION COUNT
939 006450 012701 000101          MOV    #101,R1          ;LOAD A CHARACTER
940 006454 004737 010226          JSR    PC,FILBUF          ;LOAD CHARACTER INTO BUFFER
941 006460 012737 000001 010746 1$:          MOV    #1,XOFFOK
942 006466 004737 010330          JSR    PC,XPRNT          ;DISPLAY IT
943 006472 004737 011354          JSR    PC,DELAY          ;PROGRAM DELAY
944 006476 005337 001312          DEC    TEMPO          ;FINISHED ?
945 006502 100366          BPL    1$          ;BR IF NOT
946 006504 012737 000001 010746          MOV    #1,XOFFOK
947 006512 004537 011534          JSR    R5,AMSG          ;DISABLE AUTO-PRINT
948 006516 017400          GONAPM
949 006520 004737 011354          JSR    PC,DELAY          ;ANOTHER DELAY
950
951 006524 000137 006556          JMP    $EOP          ;END OF PASS
952          DISPCH: MTEXT0
953          MTEXT1
954          MTEXT2
955          MTEXT3
956          MTEXT4
957          MTEXT5
958          MTEXT6
959          MTEXT7
960          MTEXT8
961          0
962          0
963
983          ;*****
(1)          .SBTTL  END OF PASS ROUTINE
(1)
(1)          ;*INCREMENT THE PASS NUMBER ($PASS)
(1)          ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
(1)          ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
(1)          ;*IF THERES A MONITOR GO TO IT
(1)          ;*IF THERE ISN'T JUMP TO NOWEOP
(1)
(1) 006556          $EOP:
(3) 006556 000004          SCOPE
(3) 006560 005737 001242          TST    LAST          ;TEST IF MORE
(3) 006564 001414          BEQ    1$          ;BR IF NONE
(3) 006566 023737 001242 001244          CMP    LAST,VTNOW          ;IS THIS THE LAST ONE
(3) 006574 001410          BEQ    1$          ;BR IF YES
(3) 006576 062737 000010 001244          ADD    #10,VTNOW

```

```

(3) 006604 012737 001760 001756      MOV    #TST1,WHERE
(3) 006612 000137 001724              JMP    RSTR1          ;TEST NEXT ONE
(3) 006616 005737 001104      1S:   TST    $PASS          ;TEST IF FIRST PASS
(3) 006622 001002              BNE    2S             ;BR IF NOT
(3) 006624 104400 017617      TYPE  .PASHED        ;TYPE EOP HEADER
(3) 006630 000005      2S:   RESET
(3) 006632 005737 001330      TST    WFTEST        ;TEST IF W.F. MODE
(3) 006636 001402              BEQ    3S             ;BR IF NOT
(3) 006640 004737 011462      JSR    PC,ADELAY     ;EXTRA DELAY
(3) 006644 000240      3S:   NOP

(1) 006646 005037 001106      CLR    $STNM         ;;ZERO THE TEST NUMBER
(1) 006652 005237 001104      INC    $PASS         ;;INCREMENT THE PASS NUMBER
(1) 006656 042737 100000 001104      BIC    #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
(1) 006664 005327              DEC    (PC)+         ;;LOOP?
(1) 006666 000001      $EOPCT: .WORD 1
(1) 006670 003922      BGT    $DOAGN        ;;YES
(1) 006672 012737      MOV    (PC)+,2(PC)+ ;RESTORE COUNTER
(1) 006674 000001      $ENDCT: .WORD 1
(1) 006676 006666      $EOPCT
(1) 006700 104400 006742      TYPE  $ENDMG         ;;TYPE "END PASS #"
(2) 006704 013746 001104      MOV    $PASS,-(SP)   ;;SAVE $PASS FOR TYPEOUT
(2) 006710 104404      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
(1) 006712 104400 006757      TYPE  , $ENULL       ;;TYPE A NULL CHARACTER
(1) 006716      $GET42:
(1) 006716 013700 000042      MOV    2#42,RO       ;;GET MONITOR ADDRESS
(1) 006722 001405      BEQ    $DOAGN        ;;BRANCH IF NO MONITOR
(1) 006724 000005      RESET               ;;CLEAR THE WORLD
(1) 006726 004710      $ENDAD: JSR    PC,(RO) ;GO TO MONITOR
(1) 006730 000240      NOP                ;;SAVE ROOM
(1) 006732 000240      NOP                ;;FOR
(1) 006734 000240      NOP                ;;ACT11
(1) 006736      $DOAGN:
(1) 006736 000137 006762      JMP    2#NOWEOP      ;;RETURN
(1) 006742 005015 047105 020104 $ENDMG: .ASCIZ <15><12>/END PASS #/
(1) 006750 040520 051523 021440
(1) 006756 000
(1) 006757 377 377 000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
984 006762 005737 001272      NOWEOP: TST    PRTCNT
985 006766 100002      BPL    11S
986 006770 105337 001272      DECB  PRTCNT
987 006774 012737 001760 001756 11S:   MOV    #TST1,WHERE
988 007002 013746 001116      MOV    $ERTTL,-(SP)
989 007006 104404      TYPDS
990 007010 104400 006757      TYPE  , $ENULL
991 007014 000137 001716      JMP    $BEGIN

;*****
;*TEST 30 X KEYBOARD OCTAL VALUE LOOP
;*****
(3) 007020 000004      $TST30: SCOPE
996 007022 012706 001100      KRBECC: MOV    #STACK,SP
997 007026 004537 011534      JSR    R5,AMSG      ;DISPLAY HEADER
998 007032 015620
999 007034 004737 012536      1S:   JSR    PC,GETCHR    ;GET CHAR
1000 007040 000775      BR     1S           ;;BR BACK IF NO INPUT

```

```

1001 007042 004737 011554 JSR PC,OCTAL ;CONVERT RO TO OCTAL
1002 007046 113737 011644 016043 MOVB DIG0,MKEB ;LOAD DIGIT
1003 007054 113737 011646 016044 MOVB DIG1,MKEB+1 ;LOAD DIGIT
1004 007062 113737 011650 016045 MOVB DIG2,MKEB+2 ;LOAD DIGIT
1005 007070 042700 177600 BIC #177600,RO
1006 007074 001001 BNE 10$
1007 007076 005200 INC RO
1008 007100 012701 007210 10$: MOV #BFCHR,R1 ;LOAD POINTER
1009 007104 121100 5$: CMPB (R1),RO ;TEST IF = TO VALUE IN TABLE ?
1010 007106 001403 BEQ 3$ ;BR IF FOUND
1011 007110 005721 TST (R1)+ ;MOVE POINTER
1012 007112 001374 BNE 5$ ;BR IF MORE
1013 007114 000407 BR 2$ ;BR IF NOT IN LIST
1014 007116 062701 000040 3$: ADD #BFCHAR-BFCHR,R1 ;UPDATE POINTER
1015 007122 112137 016036 MOVB (R1)+,MKEA1 ;LOAD 1ST CHAR
1016 007126 112137 016037 MOVB (R1)+,MKEA1+1 ;LOAD 2ND
1017 007132 000420 BR 4$
1018 007134 120027 000040 2$: CMPB RO,#40 ;TEST IF LESS THAN 40
1019 007140 101010 BHI 6$ ;BR IF ABOVE
1020 007142 062700 000100 ADD #100,RO ;MAKE PRINTABLE
1021 007146 110037 016037 MOVB RO,MKEA1+1 ;SAVE CHAR
1022 007152 112737 000136 016036 MOVB #136,MKEA1 ;ADD A 't' BEFORE CHARACTER
1023 007160 000405 BR 4$
1024 007162 112737 000040 016037 6$: MOVB #40,MKEA1+1 ;LOAD SPACE
1025 007170 110037 016036 MOVB RO,MKEA1 ;LOAD CHARACTER
1026 007174 005237 010760 4$: INC IGNORE ;IGNORE DOUBLE CHARACTER FLAG
1027 007200 004537 011534 JSR R5,AMSG ;DISPLAY MESSAGE
1028 007204 016025 MKEA
1029 007206 000712 BR 1$ ;LOOP BACK

```

:TABLE OF DEFINED CHARACTERS

```

1031
1032
1033 007210 000007 BFCHR: 7 ;BELL CODE
1034 007212 000010 10 ;CURSOR LEFT CODE
1035 007214 000011 11 ;TAB CODE
1036 007216 000012 12 ;LINE FEED CODE
1037 007220 000015 15 ;CARRIAGE RETURN CODE
1038 007222 000033 33 ;ESCAPE CODE
1039 007224 000040 40 ;SPACE CODE
1040 007226 000177 177 ;DELETE CODE
1041 007230 000000 0
1042 007232 000000 0
1043 007234 000000 0
1044 007236 000000 0
1045 007240 000000 0
1046 007242 000000 0
1047 007244 000000 0
1048 007246 000000 0
1049

```

;DEFINED CHARACTER EQUIL

```

1050
1051
1052 007250 046102 BFCHAR: .ASCII /BL/ ;BELL
1053 007252 046103 .ASCII /CL/ ;CURSOR LEFT
1054 007254 052110 .ASCII /HT/ ;H TAB
1055 007256 043114 .ASCII /LF/ ;LINE FEED
1056 007260 051103 .ASCII /CR/ ;CARRIAGE RETURN

```

1057	007262	051505			.ASCII /ES/	;ESCAPE
1058	007264	050123			.ASCII /SP/	;SPACE
1059	007266	042504			.ASCII /DE/	;DELETE
1060	007270	000000	000000	000000	0,0,0,0,0,0,0,0,0,0	
	007276	000000	000000	000000		
	007304	000000	000000	000000		
	007312	000000				

```

1061 .EVEN
1062 ;*****
(3) ;*TEST 31 Y KEYBOARD CHARACTER TEST
(3) ;*****
(2) 007314 000004
1063 007316 032737 001000 001256 KRBTST: BIT #BIT09,VT5XX ;IS UNIT A VT52?
1064 007324 001403 BEQ 1$
1065 007326 012703 020332 MOV #V52RW,R3 ;YES-SET UP FOR LOWER CASE CHAR.
1066 007332 000402 BR 2$
1067 007334 012703 020314 1$: MOV #V50RW,R3 ;NO-SET UP FOR UPPER CASE CHAR.
1068 007340 012706 001100 2$: MOV #STACK,SP
1069 007344 004537 011534 A: JSR R5,AMSG ;DISPLAY HEADER
1070 007350 014005 MKB
1071 007352 012302 MOV (R3)+,R2 ;LOAD ROW #
1072 007354 032737 001000 001256 BIT #BIT09,VT5XX ;UNIT A VT52?
1073 007362 001404 BEQ 1$ ;NO-USE UPPER CASE KEYBOARD.
1074 007364 004537 011534 JSR R5,AMSG ;ISSUE VT52 ROW1 MESSAGE.
1075 007370 014300 MKBB2
1076 007372 000403 BR 2$
1077 007374 004537 011534 1$: JSR R5,AMSG ;TOP ROW
1078 007400 014163 MKBB ;CHECK THE ROW
1079 007402 004737 011652 2$: JSR PC,TSTROW
1080
1081 007406 012302 B: MOV (R3)+,R2 ;LOAD ROW #
1082 007410 004537 011534 JSR R5,AMSG ;2ND ROW
1083 007414 014363 MKBC
1084 007416 004737 011652 JSR PC,TSTROW ;CHECK 2ND ROW
1085
1086 007422 012302 C: MOV (R3)+,R2 ;LOAD ROW #
1087 007424 032737 001000 001256 BIT #BIT09,VT5XX ;UNIT A VT52?
1088 007432 001404 BEQ 1$ ;NO-USE UPPER CASE KEYBOARD.
1089 007434 004537 011534 JSR R5,AMSG ;ISSUE VT52 ROW 3 MESSAGE.
1090 007440 014527 MKBD2
1091 007442 000403 BR 2$
1092 007444 004537 011534 1$: JSR R5,AMSG
1093 007450 014422 MKBD
1094 007452 004737 011652 2$: JSR PC,TSTROW ;CHECK ROW 3
1095
1096 007456 012302 MOV (R3)+,R2 ;LOAD ROW #
1097 007460 004537 011534 JSR R5,AMSG
1098 007464 014615 MKBE
1099 007466 004737 011652 JSR PC,TSTROW ;CHECK ROW 4
1100
1101 007472 012702 020516 MOV #ROW5,R2
1102 007476 004537 011534 JSR R5,AMSG
1103 007502 014735 MKBF
1104 007504 004737 011652 JSR PC,TSTROW ;CHECK ROW 5
1105
1106 ;TEST THE "LEFT"-SHIFT KEY

```



```

110 007510 004537 011534 D: JSR R5,AMSG ;DEPRESS THE "LEFT-SHIFT" KEY
111 007514 014773 MKBG
112 007516 012302 MOV (R3)+,R2 ;LOAD ROW 1 SHIFTED TABLE
113 007520 032737 001000 001256 BIT #BIT09,VT5XX ;UNIT A VT52?
114 007526 001404 BEQ 1$ ;NO-USE UPPER CASE KEYBOARD.
115 007530 004537 011534 JSR R5,AMSG ;ISSUE VT52 ROW1 MESSAGE.
116 007534 014300 MKBB2
117 007536 000403 BR 2$
118 007540 004537 011534 1$: JSR R5,AMSG ;TEST ROW 1
119 007544 014163 MKBB
120 007546 004737 011652 2$: JSR PC,TSTROW ;TEST THE ROW
121 007552 004537 011534 JSR R5,AMSG ;RELEASE THE SHIFT KEY.
122 007556 014041 MKB1
;TEST THE "RIGHT-SHIFT" KEY
123
124 007560 004537 011534 E: JSR R5,AMSG ;SET THE "RIGHT-SHIFT" KEY
125 007564 015042 MKBGA
126 007566 012302 MOV (R3)+,R2 ;LOAD TABLE POINTER
127 007570 032737 001000 001256 BIT #BIT09,VT5XX ;UNIT A VT52?
128 007576 001404 BEQ 1$ ;NO-USE UPPER CASE KEYBOARD.
129 007600 004537 011534 JSR R5,AMSG ;ISSUE VT52 ROW1 MESSAGE.
130 007604 014300 MKBB2
131 007606 000403 BR 2$
132 007610 004537 011534 1$: JSR R5,AMSG
133 007614 014163 MKBB
134 007616 004737 011652 2$: JSR PC,TSTROW ;TEST THE ROW AGAIN WITH THE RIGHT-SHIFT SET
135 007622 004537 011534 JSR R5,AMSG
136 007626 014041 MKB1 ;RELEASE SKIFT KEY
;TEST THE CONTROL MODE
137
138
139 007630 004537 011534 F: JSR R5,AMSG ;SET CTRL
140 007634 015112 MKBH
141 007636 012302 MOV (R3)+,R2 ;LOAD ROW 1 CTRL TABLE
142 007640 032737 001000 001256 BIT #BIT09,VT5XX ;UNIT A VT52?
143 007646 001404 BEQ 1$ ;NO-USE UPPER CASE KEYBOARD.
144 007650 004537 011534 JSR R5,AMSG ;ISSUE VT52 ROW1 MESSAGE.
145 007654 014300 MKBB2
146 007656 000403 BR 2$
147 007660 004537 011534 1$: JSR R5,AMSG
148 007664 014163 MKBB
149 007666 004737 011652 2$: JSR PC,TSTROW ;TEST THE ROW
150
;TEST THE VT50H KEYPAD
151
152
153 007672 032737 010000 001256 BIT #BIT12,VT5XX ;TEST IF VT50H EXTRA KEYPAD
154 007700 001526 BEQ KRBDON ;:BR IF NOT DETECTED
155 007702 004537 011534 JSR R5,AMSG ;TELL OPERATOR THE TEST NAME
156 007706 015364 MKBN
157 007710 012702 021122 MOV #ROW6,R2 ;LOAD ROW POINTER VALUES
158 007714 004537 011534 JSR R5,AMSG
159 007720 015153 MKB1 ;SET TOP ROW KEYPAD
160 007722 004737 011652 JSR PC,TSTROW ;TEST THAT ROW
161 007726 012702 021142 MOV #ROW7,R2
162 007732 004537 011534 JSR R5,AMSG ;2ND ROW KEYPAD TEST

```

```

1163 007736 015206 MKBJ
1164 007740 004737 JSR PC,TSTROW ;TEST 2ND KEYPAD ROW
1165 007744 012702 MOV #ROW8,R2
1166 007750 004537 JSR RS,AMSG
1167 007754 015241 MKBK
1168 007756 004737 JSR PC,TSTROW ;TEST 3RD KEYPAD ROW
1169 007762 012702 MOV #ROW9,R2
1170 007766 004537 JSR RS,AMSG ;TELL ABOUT 4TH ROW
1171 007772 015275 MKBL
1172 007774 004737 JSR PC,TSTROW ;TEST 4TH ROW
1173 010000 012702 MOV #ROW10,R2
1174 010004 004537 JSR RS,AMSG
1175 010010 015330 MKBM
1176 010012 004737 JSR PC,TSTROW ;TEST 5TH ROW
1177 010016 032737 BIT #1000,VT5XX ;IS UNIT A VT52?
1178 010024 001454 BEQ KRBDON ;NO-EXIT
1179 010026 004537 JSR RS,AMSG ;YES-TELL OPERATOR ALT. KEYPAD TEST.
1180 010032 015413 MKB52
1181 010034 004537 JSR RS,AMSG ;PUT THE UNIT IN ALT. KEYPAD MODE.
1182 010040 021063 ENAKP
1183 010042 012702 MOV #ROW6A,R2 ;LOAD ROW POINTER VALUES
1184 010046 004537 JSR RS,AMSG
1185 010052 015153 MKBI ;SET TOP ROW KEYPAD
1186 010054 004737 JSR PC,TSTROW ;TEST THAT ROW
1187 010060 012702 MOV #ROW7A,R2
1188 010064 004537 JSR RS,AMSG ;2ND ROW KEYPAD TEST
1189 010070 015206 MKBJ
1190 010072 004737 JSR PC,TSTROW ;TEST 2ND KEYPAD ROW
1191 010076 012702 MOV #ROW8A,R2
1192 010102 004537 JSR RS,AMSG
1193 010106 015241 MKBK
1194 010110 004737 JSR PC,TSTROW ;TEST 3RD KEYPAD ROW
1195 010114 012702 MOV #ROW9A,R2
1196 010120 004537 JSR RS,AMSG ;TELL ABOUT 4TH ROW
1197 010124 015275 MKBL
1198 010126 004737 JSR PC,TSTROW ;TEST 4TH ROW
1199 010132 012702 MOV #ROW10A,R2
1200 010136 004537 JSR RS,AMSG
1201 010142 015330 MKBM
1202 010144 004737 JSR PC,TSTROW ;TEST 5TH ROW
1203 010150 004537 JSR RS,AMSG ;EXIT ALT. KEYPAD MODE.
1204 010154 021070 EXAKP
1205
1206 ;COMPLETION OF KEYBOARD-KEYPAD TEST
1207
1208 010156 004537 011534 KRBDON: JSR RS,AMSG ;END OF KEYBOARD TEST
1209 010162 015552 MKBR
1210 010164 000137 007316 JMP KRBTST ;LOOP
1211
1212 (3)
1213 (3)
1214 (2) 010170 000004
1215 010172 012706 001100 KRBECH: MOV #STACK,SP
010176 004537 011534 JSR RS,AMSG ;DISPLAY HEADER
010202 016052 MKEH

```

```

1216 010204 004737 012536 1$: JSR PC,GETCHR ;GET CHARACTER
1217 010210 000775 BR 1$
1218 010212 110077 171064 MOVB R0,DVTOB ;LOAD THE CHARACTER
1219 010216 105777 171056 2$: TSTB DVTOS ;WAIT FOR DONE
1220 010222 100375 BPL 2$
1221 010224 000767 BR 1$ ;LOOP BACK
1222
1223 ;LOAD A SINGLE CHARACTER ACROSS THE SCREEN WIDTH
1224
1225
1226
1227
1228
1229 010226 013702 001316 FILBUF: MOV WIDTH,R2 ;LOAD WIDTH VALUE
1230 010232 012700 024312 FILBFB: MOV #BUFFER,R0 ;SET-UP BUFFER POINTER
1231 010236 112720 000015 MOVB #15,(R0)+ ;LOAD 'CR'
1232 010242 112720 000012 MOVB #12,(R0)+ ;LOAD 'FL'
1233 010246 110120 FILBFA: MOVB R1,(R0)+ ;SAVE THE CHARACTER IN THE BUFFER
1234 010250 005302 DEC R2 ;FINISHED?
1235 010252 001375 BNE FILBFA ;BRANCH IF NOT COMPLETED
1236 010254 112710 000377 MOVB #377,(R0) ;LOAD TERM.
1237 010260 000207 RTS PC ;EXIT
1238
1239 ;LOAD A INCREMENTING CHARACTER ACROSS THE SCREEN WIDTH
1240 ;ONLY 40 THRU 177 ARE LEGAL CHARACTERS
1241
1242
1243 010262 012700 024312 LIC: MOV #BUFFER,R0 ;SET-UP BUFFER POINTER
1244 010266 013502 MOV 2(5)+,R2 ;SET-UP WIDTH
1245 010270 112720 000015 MOVB #15,(R0)+ ;LOAD 'CR'
1246 010274 112720 000012 MOVB #12,(R0)+ ;LOAD 'LF'
1247 010300 110120 LICA: MOVB R1,(R0)+ ;SAVE A CHARACTER IN THE BUFFER
1248 010302 005201 INC R1 ;UPDATE THE CHARACTER
1249 010304 023701 001306 CMP LASTCH,R1 ;TEST FOR
1250 010310 001002 BNE LICB ;BRANCH IF NOT
1251 010312 012701 000040 MOV #40,R1 ;MAKE A LEGAL CHARACTER
1252 010316 005302 LICB: DEC R2 ;DECREMENT COUNT
1253 010320 001367 BNE LICA ;BRANCH IF NOT COMPLETED
1254 010322 112710 000377 MOVB #377,(R0) ;LOAD TERM
1255 010326 000205 RTS R5 ;EXIT
1256
1257 ;DISPLAY SUBROUTINE
1258 010330 012700 024312 XPRNT: MOV #BUFFER,R0 ;SETUP BUFFER POINTER
1259 010334 105777 170740 XPRNTA: TSTB DVTOS ;TEST READY
1260 010340 100404 BMI XPRNTB ;BRANCH IF SET
1261 010342 005777 170732 TST DVTOS ;TEST ERROR
1262 010346 100372 BPL XPRNTA ;BRANCH IF RESET
1263 010350 104001 ERROR 1 ;ERROR FLAG SET ON TRANSMITTER STATUS
1264 010352 112001 XPRNTB: MOVB (0)+,R1
1265 010354 100563 BMI 1$ ;BR IF MINUS
1266 010356 122701 000033 5$: CMPB #33,R1 ;TEST FOR ESC
1267 010362 001003 BNE 3$ ;BR IF NOT
1268 010364 005237 010756 INC ANESC ;SET SOFT FLAG
1269 010370 000402 BR 4$
1270 010372 005037 010756 3$: CLR ANESC ;CLEAR SOFT FLAG
1271 010376 110177 170700 4$: MOVB R1,DVTOB ;LOAD CHAR
1272 010402 105777 170666 TSTB DVTIS ;TEST INPUT FLAG
1273 010406 100352 BPL XPRNTA ;BR IF CLEARED
1274 010410 005737 010756 TST ANESC ;TEST IF ESC

```

```

1275 010414 001347          BNE      XPRNTA
1276 010416 013737 001252 012610 52$: MOV      TIME0,TIME1
1277 010424 005037 012612          CLR      TIME2          ;LOAD DELAY
1278 010430 105777 170640          TSTB     QVTIS          ;TEST IF INPUT FLAG
1279 010434 100407          BMI      53$          ;BR IF SET
1280 010436 005337 012612          DEC      TIME2          ;DELAY
1281 010442 001372          SNE      2$
1282 010444 005337 012610          DEC      TIME1          ;DELAY
1283 010450 001367          BNE      2$
1284 010452 000440          BR       60$          ;REPORT ERROR
1285 010454 005737 010760 53$: TST      IGNORE          ;IGNORE KEYBOARD CHARACTERS ?
1286 010460 001104          BNE      15$          ;BR IF YES
1287 010452 017737 170610 001132 MOV      QVTIB,$BDDAT  ;READ CHAR
1288 010470 042737 177600 001132 BIC      #177600,$BDDAT ;MASK
1289 010476 022737 000021 001132 CMP      #XON,$BDDAT  ;TEST FOR X ON
1290 010504 001003          BNE      50$
1291 010506 005237 010754          INC      XONRC          ;SET FLAG
1292 010512 000710          BR       XPRNTA          ;START AGAIN
1293 010514 022737 000023 001132 50$: CMP      #XOFF,$BDDAT  ;TEST FOR X OFF
1294 010522 001020          BNE      51$          ;BR IF NOT
1295 010524 005237 010752          INC      XOFFRC
1296 010530 005737 010746          TST      XOFFOK          ;XOFF ENABLED ?
1297 010534 001730          BEQ      52$
1298 010536 012737 000001 010750 MOV      #1,AXOFF          ;SET X OFF SOFT FLAG
1299 010544 005737 010744          TST      XOFFBR          ;TEST
1300 010550 001271          BNE      XPRNTA          ;BR BACK
1301 010552 000721          BR       52$
1302 010554 012737 000000 001130 60$: MOV      #0,$GDDAT
1303 010562 000437          BR       13$
1304 010564 105777 170310          TSTB     QSWR          ;TEST SWR
1305 010570 100371          BPL      50$          ;BR IF CLEARED
1306 010572 012737 000057 001130 MOV      #'/$GDDAT      ;LOAD EXPECTED
1307 010600 023737 001130 001132 CMP      $GDDAT,$BDDAT  ;COMPARE
1308 010606 001437          BEQ      14$          ;BR IF EQUAL
1309 010610 012737 000134 001130 MOV      #'\'$GDDAT     ;LOAD EXPECTED
1310 010616 023737 001130 001132 CMP      $GDDAT,$BDDAT  ;COMPARE
1311 010624 001016          BNE      13$          ;BR IF NOT
1312
1313          ;"\" OR LOOP EXIT
1314
1315 010626 012737 000001 010742 MOV      #1,LOOP          ;SET SOFT FLAG
1316 010634 012737 017075 011000 MOV      #MQ0,FINDTA     ;SETUP MESSAGE
1317 010642 005037 010744 16$: CLR      XOFFBR
1318 010646 005037 010746          CLR      XOFFOK
1319 010652 005037 010760          CLR      IGNORE
1320 010656 000137 010770          JMP      FINDOT
1321
1322 010662 005737 010760 13$: TST      IGNORE          ;IGNORE INPUT CHARACTER
1323 010666 001001          BNE      15$
1324 010670 104004          ERROR    4          ;UNEXPECTED OR INCORRECT INPUT FLAG
1325 010672 000240 15$: NOP
1326 010674 000240          NOP
1327 010676 000240          NOP
1328 010700 000240          NOP
1329 010702 000137 010334          JMP      XPRNTA
1330

```

```

1331          ;"/" OR STANDARD EXIT
1332
1333 010706 005037 010742          14: CLR LOOP
1334 010712 012737 017162 011000 14: MOV #MQ1,FINDTA ;SETUP MESSAGE
1335 010720 000137 010642          14: JMP 16$
1336
1337          ;NORMAL EXIT
1338
1339 010724 005037 010746          15: CLR XOFFOK
1340 010730 005037 010760          15: CLR IGNORE
1341 010734 005037 010744          15: CLR XOFFBR
1342 010740 000207          15: RTS PC ;EXIT
1343
1344 010742 000000          LOOP: 0
1345 010744 000000          XOFFBR: 0
1346 010746 000000          XOFFOK: 0
1347 010750 000000          AXOFF: 0
1348 010752 000000          XOFFRC: 0
1349 010754 000000          XONRC: 0
1350 010756 000000          ANESC: 0
1351 010760 000000          IGNORE: 0 ;WHEN SET IGNORE KEYBOARD FLAGS
1352
1353          ;DETERMINE THE TEST TO GO TO
1354 010762 004537 011534          FNDA: JSR RS,AMSG
1355 010766 017250          FNDA: MQ2 ;ERROR ASK AGAIN
1356 010770 012706 001100          FINDOT: MOV #STACK,SP
1357 010774 004537 011534          FINDOT: JSR RS,AMSG
1358 011000 017162          FINDTA: MQ1
1359 011002 004737 012536          FINDTA: JSR PC,GETCHR
1360 011006 000770          FINDTA: BR FINDOT
1361 011010 042700 100600          FINDTA: BIC #100600,RO ;MASK
1362 011014 122700 000101          FINDTA: CMPB #'A,RO ;TEST FOR NUMBER
1363 011020 101360          FINDTA: BHI FNDA
1364 011022 122700 000132          FINDTA: CMPB #'Z,RO ;TEST FOR OTHERS
1365 011026 103755          FINDTA: BLO FNDA
1366 011030 042700 177740          FINDTA: BIC #177740,RO ;MAKE 0-32
1367 011034 005300          FINDTA: DEC RO
1368 011036 110037 001106          FINDTA: MOVB RO,$STNM ;LOAD THAT TEST #
1369 011042 006300          FINDTA: ASL RO
1370 011044 005760 011070          FINDTA: TST DSPCH(RO) ;TEST IF VALID
1371 011050 001744          FINDTA: BEQ FNDA ;BR IF NOT
1372 011052 000240          FINDTA: NOP
1373 011054 000240          FINDTA: NOP
1374 011056 016037 011070 001112          FINDTA: MOV DSPCH(RO),SLPADR ;LOAD LOOP ADDRESS
1375 011064 000170 011070          FINDTA: JMP @DSPCH(RO) ;GO TO THAT TEST
1376
1377          ;SUBTEST DISPATCH TABLE
1378
1379 011070 001762          DSPCH: TST1+2
1380 011072 002402          DSPCH: TST2+2
1381 011074 002422          DSPCH: TST3+2
1382 011076 002476          DSPCH: TST4+2
1383 011100 002576          DSPCH: TST5+2
1384 011102 002700          DSPCH: TST6+2
1385 011104 003342          DSPCH: TST7+2
1386 011106 003712          DSPCH: TST10+2

```

1387 011110 004070
1388 011112 004216
1389 011114 004254
1390 011116 005014
1391 011120 005164
1392 011122 005276
1393 011124 005446
1394 011126 005502
1395 011130 005572
1396 011132 005724
1397 011134 006060
1398 011136 006202
1399 011140 006260
1400 011142 006372
1401 011144 006426
1402 011146 007022
1403 011150 007316

TST11+2
TST12+2
TST13+2
TST14+2
TST15+2
TST16+2
TST17+2
TST20+2
TST21+2
TST22+2
TST23+2
TST24+2
TST25+2
TST26+2
TST27+2
TST30+2
TST31+2

;SUBROUTINE TO LOAD COPIER TEST

1404
1405
1406 011152 012700 024312
1407 011156 112720 000015
1408 011162 112720 000012
1409 011166 112720 000105
1410 011172 112720 000105
1411 011176 112720 000105
1412 011202 013702 001316
1413 011206 163702 001266
1414 011212 005302
1415 011214 112720 000040
1416 011220 005302
1417 011222 100374
1418 011224 112720 000105
1419 011230 112720 000105
1420 011234 112720 000105
1421 011240 112710 000377
1422 011244 000207

FIRLST: MOV #BUFFER, R0
MOV #15, (R0)+ ;LOAD CR
MOV #12, (R0)+ ;LOAD LF
MOV #E, (R0)+
MOV #E, (R0)+
MOV #E, (R0)+
MOV WIDTH, R2 ;LOAD WIDTH
SUB VH1, R2
DEC R2
1\$: MOV #40, (R0)+ ;LOAD A SPACE
DEC R2 ;DONE ?
BPL 1\$;BR UNTIL DONE
MCVB #E, (R0)+
MOV #E, (R0)+
MOV #E, (R0)+ ;LOAD 80 TH
MOV #377, (R0) ;LOAD TERM
RTS PC ;EXIT

;SUBROUTINE FOR THE DATA PATH TEST

1423
1424
1425
1426 011246 012700 024312
1427 011252 012501
1428 011254 012502
1429 011256 013703 001316
1430 011262 006203
1431 011264 112720 000015
1432 011270 112720 000012
1433 011274 110120
1434 011276 110220
1435 011300 005303
1436 011302 100374
1437 011304 112710 000377
1438 011310 000205
1439
1440
1441

DTPSR: MOV #BUFFER, R0
DTPSRB: MOV (5)+, R1 ;GET FIRST CHARACTER
MOV (5)+, R2 ;GET SECOND CHARACTER
MOV WIDTH, R3 ;SET THE WIDTH
ASR R3 ;DIVIDE BY 2
MOV #15, (R0)+ ;LOAD 'CR'
MOV #12, (R0)+ ;LOAD 'LF'
DTPSRA: MOV R1, (0)+
MOV R2, (0)+
DEC R3
BPL DTPSRA
MOV #377, (R0) ;LOAD TERM
RTS R5

;SUBROUTINE TO LOAD BUFFER WITH GRAPHICS CHARACTERS

1442 011312 012700 024312

GBBUF: MOV #BUFFER, R0 ;LOAD BUFFER ADDRESS

```

1443 011316 012701 000136      MOV      #136,R1      ;LOAD INITIAL CHAR.
1444 011322 010402      MOV      R4,R2      ;LOAD BUFFER COUNT
1445 011324 110120      1$:      MOVB     R1,(R0)+   ;INSERT A CHAR. IN THE BUFFER
1446 011326 005201      INC      R1          ;INCREMENT CHAR.
1447 011330 122701 000177      CMPB     #177,R1     ;AT END OF GRAPHICS STRING?
1448 011334 001002      BNE      2$         ;NO
1449 011336 012701 000136      MOV      #136,R1     ;YES-RESET IT TO 1ST GRAPH. CHAR.
1450 011342 005302      2$:      DEC      R2          ;DECREMENT BUFFER COUNT.
1451 011344 001367      BNE      1$         ;NOT AT END-LOOP
1452 011346 112710 000377      MOVB     #377,(R0)   ;END OF BUFFER-INSERT TERMINATOR
1453 011352 000207      RTS      PC         ;AND EXIT.

```

:PROGRAM DELAY ROUTINE

```

1454
1455
1456
1457 011354 013737 001254 011456 DELAY:  MOV      SUBTST,10$   ;LOAD COUNT
1458 011362 005037 011460      CLR      11$        ;
1459 011366 005737 001330      TST      WFTST      ;TEST IF W.F. MODE
1460 011372 001413      BEQ      2$         ;BR IF NOT
1461 011374 006237 011456      ASR      10$        ;CHANGE DELAY TIMER
1462 011400 006237 011456      ASR      10$
1463 011404 006237 011456      ASR      10$
1464 011410 000240      NOP
1465 011412 000240      NOP
1466 011414 000240      NOP
1467 011416 000240      NOP
1468 011420 000240      NOP
1469 011422 032777 010000 167450 2$:  BIT      #BIT12,2SWR ;TEST SR
1470 011430 001006      BNE      3$         ;BR IF SET
1471 011432 005337 011460      DEC      11$        ;DELAY
1472 011436 001371      BNE      2$
1473 011440 005337 011456      DEC      10$
1474 011444 100366      BPL      2$         ;DELAY
1475 011446 000240      3$:      NOP
1476 011450 000240      NOP
1477 011452 000240      NOP
1478 011454 000207      RTS      PC         ;EXIT
1479

```

```

1480 011456 000002      10$:    2
1481 011460 000000      11$:    0
1482
1483 011462 013737 001254 011530 ADELAY: MOV      SUBTST,10$
1484 011470 005037 011532      CLR      11$
1485 011474 006237 011530      ASR      10$
1486 011500 006237 011530      ASR      10$
1487 011504 005337 011532      2$:      DEC      11$
1488 011510 001375      BNE      2$
1489 011512 005337 011530      DEC      10$
1490 011516 100372      BPL      2$
1491 011520 000240      NOP
1492 011522 000240      NOP
1493 011524 000240      NOP
1494 011526 000207      RTS      PC
1495 011530 000000      10$:    0
1496 011532 000000      11$:    0
1497
1498

```

```

1499          :HEADER SUBROUTINE FOR VT-50
1500
1501 011534 012537 011544  AMSG:  MOV    (R5)+,10$    ;GET POINTER
1502 011540 004537 012614  1$:   JSR    RS,MT0B    ;MOVE TO BUFFER
1503 011544 000000                10$:   0
1504 011546 004737 010330  11$:  JSR    PC,XPRNT    ;DISPLAY IT
1505 011552 000205                11$:  RTS     RS         ;EXIT
1506
1507
1508          :OCTAL - 3 BIT CONVERSION
1509
1510 011554 010001                OCTAL: MOV    R0,R1      ;LOAD R1
1511 011556 042701 177770        BIC    #177770,R1    ;MASK
1512 011562 062701 000060        ADD    #60,R1
1513 011566 110137 011650        MOVB   R1,DIG2      ;SAVE LSD
1514 011572 010001                MOV    R0,R1
1515 011574 006001                ROR    R1
1516 011576 006001                ROR    R1
1517 011600 006001                ROR    R1
1518 011602 042701 177770        BIC    #177770,R1
1519 011606 062701 000060        ADD    #60,R1
1520 011612 110137 011646        MOVB   R1,DIG1      ;SAVE IT
1521 011616 010001                MOV    R0,R1
1522 011620 006101                ROL    R1
1523 011622 006101                ROL    R1
1524 011624 000301                SWAB   R1
1525 011626 042701 177770        BIC    #177770,R1
1526 011632 062701 000060        ADD    #60,R1
1527 011636 110137 011644        MOVB   R1,DIG0      ;SAVE MSD
1528 011642 000207                RTS     PC         ;EXIT
1529
1530 011644 000000                DIG0:  0
1531 011646 000000                DIG1:  0
1532 011650 000000                DIG2:  0
1533
1534          :SUBROUTINE FOR THE KEYBOARD CHARACTER TEST
1535
1536 011652 004537 011534  TSTROW: JSR   R5,AMSG    ;DISPLAY HEADER
1537 011656 014074                MKBA
1538
1539 011660 004737 012536  1$:   JSR    PC,GETCHR    ;GET CHAR
1540 011664 000775                BR     1$          ;:BR BACK IF NO INPUT
1541 011666 012737 177600 012244  MOV    #177600,MASK1
1542 011674 005037 012246        CLR    MASK2
1543 011700 032777 000004 167172  BIT    #BIT2,@SWR    ;TEST SWR
1544 011706 001416                BEQ    4$          ;DO NOT TEST PARITY BIT
1545 011710 042737 000200 012244  BIC    #BIT7,MASK1  ;ENABLE PARITY BIT
1546 011716 032777 000002 167154  BIT    #BIT1,@SWR    ;TEST IF FORCED PARITY
1547 011724 001424                BEQ    5$          ;BR IF NOT FORCED PARITY BIT
1548 011726 032777 000001 167144  BIT    #BIT0,@SWR    ;TEST FOR EVEN/ODD PARITY
1549 011734 001403                BEQ    4$          ;BR IF ALWAYS OFF
1550 011736 052737 000200 012246  BIS    #BIT7,MASK2  ;SET BIT 7
1551 011744 011237 012240        4$:   MOV    (R2),100$    ;GET EXPECTED
1552 011750 053737 012246 012240  BIS    MASK2,100$   ;SET BIT 7 IF EXPECTED
1553 011756 043700 012244        BIC    MASK1,R0     ;MASK VALUE READ
1554 011762 120037 012240        CMPB   R0,100$     ;COMPARE CHARS

```



```

1555 011766 001041          BNE      2$          ;BR IF NOT EQUAL
1556 011770 005722          TST      (R2)+       ;BUMP R2
1557 011772 100332          BPL      1$          ;LOOP TILL DONE
1558 011774 000207          RTS      PC          ;EXIT
1559
1560                               ;COME HERE ONLY IF TESTING "PARITY" OPTION
1561
1562 011776 005037 012242 5$:   CLR      101$       ;CLEAR TEMP
1563 012002 011237 012240      MOV      (R2),100$   ;CLEAR CHAR SAVE
1564 012006 006037 012240 20$:  ROR      100$       ;ROTATE CHAR
1565 012012 103002          BCC      21$       ;BR IF NO CARRY
1566 012014 005237 012242      INC      101$       ;UPDATE CNT
1567 012020 105737 012240 21$:  TSTB    100$       ;DONE ?
1568 012024 001370          BNE      20$       ;BR IF NOT
1569 012026 032777 000001 167044 BIT      #BIT0,2SWR ;TEST EVEN/ODD
1570 012034 001407          BEQ      23$       ;BR IF OPER. SAYS EVEN
1571 012036 006037 012242      ROR      101$
1572 012042 103403          BCS      22$       ;BR IF ODD ALREADY
1573 012044 052737 000200 012246 BIS      #BIT7,MASK2 ;SET PARITY BIT
1574 012052 000734          BR       4$        ;BR TO TEST CHAR
1575 012054 006037 012242 22$:  ROR      101$
1576 012060 103003          BCC      24$       ;BR IF EVEN ALREADY
1577 012062 052737 000200 012246 BIS      #BIT7,MASK2
1578 012070 000725          BR       4$        ;BR TO TEST CHAR
1579
1580                               ;COME HERE IF EXPECTED NOT EQUAL TO RECVD
1581                               ;CONVERT RESULTS TO OCTAL FOR TYPEOUT
1582
1583 012072 010037 001132 2$:   MOV      R0,$BDDAT  ;LOAD BAD CHARACTER
1584 012076 004737 011554      JSR      PC,OCTAL  ;CONVERT TO OCTAL
1585 012102 113737 011644 015543  MOVB    DIG0,MKBQB  ;LOAD OCTAL *
1586 012110 113737 011646 015544  MOVB    DIG1,MKBQB+1
1587 012116 113737 011650 015545  MOVB    DIG2,MKBQB+2
1588 012124 042700 177600      BIC      #177600,R0
1589 012130 120027 000040      CMPB    R0,#40     ;TEST IF PRINTABLE
1590 012134 101002          BHI      10$       ;BR IF PRINTABLE
1591 012136 112700 000056      MOVB    #56,R0     ;CONVERT TO A "*" CHARACTER
1592 012142 110037 015537 10$:  MOVB    R0,MKBQ2   ;SAVE CHAR
1593 012146 011200          MOV      (R2),R0   ;GET GOOD CHAR
1594 012150 053700 012246      BIS      MASK2,R0
1595 012154 010037 001130      MOV      R0,$GDDAT ;LOAD GOOD CHARACTER
1596 012160 004737 011554      JSR      PC,OCTAL  ;CONVERT IT
1597 012164 113737 011644 015524  MOVB    DIG0,MKBQA  ;LOAD DIGIT
1598 012172 113737 011646 015525  MOVB    DIG1,MKBQA+1
1599 012200 113737 011650 015526  MOVB    DIG2,MKBQA+2
1600 012206 042700 177600      BIC      #177600,R0
1601 012212 110037 015520      MOVB    R0,MKBQ1   ;SAVE CHAR
1602
1603 012216 023737 001274 001142  CMP      VTIS,$TKS ;TEST IF ON CTY
1604 012224 001403          BEQ      3$        ;BR IF YES
1605
1606 012226 004537 011534      JSR      R5,AMSG   ;DISPLAY ERROR MESSAGE
1607 012232 015457          MKBQ
1608 012234 104004          ERROR 4           ;CHARACTER RECVD NOT EQUAL TO EXPECTED
1609 012236 000610          BR      1$        ;BR BACK AND TEST THE CHARACTER AGAIN
1610

```

1611 012240 000000
1612 012242 000000
1613 012244 177600
1614 012246 000000
1615
1616

100\$: 0
101\$: 0
MASK1: 177600
MASK2: 0

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;*REPLACED WITH SPACES.

;*CALL:
;* MOV NUM,-(SP) ;;PUT THE BINARY NUMBER ON THE STACK
;* TYPDS ;;GO TO THE ROUTINE

\$TYPDS:

(2) 012250
(3) 012250 010046
(3) 012252 010146
(3) 012254 010246
(3) 012256 010346
(3) 012260 010546
(1) 012262 012746 020200
(1) 012266 016605 000020
(1) 012272 100004
(1) 012274 005405
(1) 012276 112766 000055 000001
(1) 012304 005000
(1) 012306 012703 012464
(1) 012312 112723 000040
(1) 012316 005002
(1) 012320 016001 012454
(1) 012324 160105
(1) 012326 002402
(1) 012330 005202
(1) 012332 000774
(1) 012334 060105
(1) 012336 005702
(1) 012340 001002
(1) 012342 105716
(1) 012344 100407
(1) 012346 106316
(1) 012350 103003
(1) 012352 116663 000001 177777
(1) 012360 052702 000060
(1) 012364 052702 000040
(1) 012370 110223
(1) 012372 005720
(1) 012374 020027 000010
(1) 012400 002746
(1) 012402 003002
(1) 012404 010502
(1) 012406 000764
(1) 012410 105726

MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK
MOV #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
MOV 20(SP),R5 ;;GET THE INPUT NUMBER
BPL 1\$;;BR IF INPUT IS POS.
NEG R5 ;;MAKE THE BINARY NUMBER POS.
MOVB #'-,1(SP) ;;MAKE THE ASCII NUMBER NEG.
1\$: CLR R0 ;;ZERO THE CONSTANTS INDEX
MOV #5DBLK,R3 ;;SETUP THE OUTPUT POINTER
MOVB #' ,(R3)+ ;;SET THE FIRST CHARACTER TO A BLANK
2\$: CLR R2 ;;CLEAR THE BCD NUMBER
MOV \$DTBL(R0),R1 ;;GET THE CONSTANT
3\$: SUB R1,R5 ;;FORM THIS BCD DIGIT
BLT 4\$;;BR IF DONE
INC R2 ;;INCREASE THE BCD DIGIT BY 1
BR 3\$
4\$: ADD R1,R5 ;;ADD BACK THE CONSTANT
TST R2 ;;CHECK IF BCD DIGIT=0
BNE 5\$;;FALL THROUGH IF 0
TSTB (SP) ;;STILL DOING LEADING 0'S?
BMI 7\$;;BR IF YES
5\$: ASLB (SP) ;;MSD?
BCC 6\$;;BR IF NO
MOVB 1(SP),-1(R3) ;;YES--SET THE SIGN
6\$: BIS #'0,R2 ;;MAKE THE BCD DIGIT ASCII
7\$: BIS #' ,R2 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB R2,(R3)+ ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST (R0)+ ;;JUST INCREMENTING
CMP R0,#10 ;;CHECK THE TABLE INDEX
BLT 2\$;;GO DO THE NEXT DIGIT
BGT 8\$;;GO TO EXIT
MOV R5,R2 ;;GET THE LSD
BR 6\$;;GO CHANGE TO ASCII
8\$: TSTB (SP)+ ;;WAS THE LSD THE FIRST NON-ZERO?

```

(1) 012412 100003          BPL      9$          ;;BR IF NO
(1) 012414 116663 177777 177776  MOVB    -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
(1) 012422 105013          CLR      (R3)          ;;SET THE TERMINATOR
(3) 012424 012605          MOV     (SP)+,R5       ;;POP STACK INTO R5
(3) 012426 012603          MOV     (SP)+,R3       ;;POP STACK INTO R3
(3) 012430 012602          MOV     (SP)+,R2       ;;POP STACK INTO R2
(3) 012432 012601          MOV     (SP)+,R1       ;;POP STACK INTO R1
(3) 012434 012600          MOV     (SP)+,R0       ;;POP STACK INTO R0
(1) 012436 104400 012464  TYPE     $DBLK          ;;NOW TYPE THE NUMBER
(1) 012442 016666 000002 000004  MOV     2(SP),4(SP)    ;;ADJUST THE STACK
(1) 012450 012616          MOV     (SP)+,(SP)
(1) 012452 000002          RTI                      ;;RETURN TO USER
(1) 012454 023420          $DTBL: 10000.
(1) 012456 001750          1000.
(1) 012460 000144          100.
(1) 012462 000012          10.
(1) 012464 000004          $DBLK: .BLKW 4

1617
1618          ;;SUBROUTINE TO FILL THE SCREEN WITH AN CHARACTER
1619
1620 012474 012701 000105  FILLWC: MOV     #'E,R1      ;LOAD CHARACTER BYTE
1621 012500 004737 010226  JSR     PC,FILBUF        ;LOAD THE LINE WITH CHAR
1622 012504 013737 001264 012534  MOV     VHD,10$         ;LOAD COUNT
1623 012512 012737 000001 010746  1$:    MOV     #1,XOFFOK   ;INSURE XOFF/XON CONTROL.
1624 012520 004737 010330  JSR     PC,XPRNT        ;DISPLAY THE LINE
1625 012524 005337 012534  DEC     10$
1626 012530 001370          BNE     1$              ;LOOP UNTIL DONE
1627 012532 000207          RTS     PC              ;EXIT
1628 012534 000000          10$:    0
1629
1630          ;SUBROUTINE TO GET A CHARACTER FROM THE VTSO
1631
1632 012536 013737 001252 012610  GETCHR: MOV     TIME1,TIME1 ;LOAD TIME COUNTER
1633 012544 005037 012612          CLR     TIME2
1634
1635 012550 105777 166520          1$:    TSTB    @VTIS        ;TEST INPUT STATUS
1636 012554 100005          BPL     2$              ;BR IF CLEARED
1637 012556 017700 166514          MOV     @VTIB,R0       ;READ A CHAR
1638 012562 062716 000002          ADD     #2,(SP)        ;UPDATE RETURN
1639 012566 000207          RTS     PC              ;EXIT
1640
1641 012570 005337 012612          2$:    DEC     TIME2        ;DELAY
1642 012574 001365          BNE     1$
1643 012576 005337 012610          DEC     TIME1
1644 012602 100362          BPL     1$              ;FINISHED ?
1645 012604 104002          ERROR  2$              ;LOOP TILL TIME EXPIRED
1646 012606 000207          RTS     PC              ;NO INPUT FLAG FROM DEVICE
1647
1648 012610 000000          TIME1: 0
1649 012612 000000          TIME2: 0
1650
1651          ;MOVE TO THE OUTPUT BUFFER
1652
1653 012614 012500          MTOB:  MOV     (R5)+,R0   ;LOAD DEST.
1654 012616 012701 024312          MOV     #BUFFER,R1     ;LOAD R1
1655 012622 112021          1$:    MOV     (R0)+,(R1)+  ;LOAD BYTE

```

```

1656 012624 100376          BPL      1S      :BR UNTIL DONE
1657 012626 000205          RTS      RS      :EXIT
1658
1659          .SBTTL  ASCII MESSAGES
1660
1661          :ASCII MESSAGES
1662
1663 012630 006415 046412 044501 TITLE: .ASCIZ <15><15><12>\MAINDEC-11-DZVTC-C VT50A, B, VT50H AND VT52 ACCEPTANCE TEST
      012636 042116 041505 030455
      012644 026461 055104 052126
      012652 026503 020103 052126
      012660 030065 026101 041040
      012666 020054 052126 030065
      012674 020110 047101 020104
      012702 052126 031065 040440
      012710 041503 050105 040524
      012716 041516 020105 042524
      012724 052123 005015      000
1664 012731      015 020012 043040 M91: .ASCIZ <15><12>/ FULL SCREEN OF THE CHARACTER E/<15><12><377>
      012736 046125 020114 041523
      012744 042522 047105 047440
      012752 020106 044124 020105
      012760 044103 051101 041501
      012766 042524 020122 006505
      012774 177412      000
1665 012777      015 020012 042040 M92: .ASCIZ <15><12>/ DATA PATH TEST /<15><12><377>
      013004 052101 020101 040520
      013012 044124 052040 051505
      013020 020124 005015 000377
1666 013026 005015 020040 044523 M93: .ASCIZ <15><12>/ SINGLE CHARACTER PER LINE /<15><12><377>
      013034 043516 042514 041440
      013042 040510 040522 052103
      013050 051105 050040 051105
      013056 046040 047111 020105
      013064 005015 000377
1667 013070 005015 020040 047522 M94: .ASCIZ <15><12>/ ROTATING PATTERN /<15><12><377>
      013076 040524 044524 043516
      013104 050040 052101 042524
      013112 047122 006440 177412
      013120      000
1668 013121      015 020012 041440 M96: .ASCIZ <15><12>/ CURSOR MOTION TEST /<15><12><377>
      013126 051125 047523 020122
      013134 047515 044524 047117
      013142 052040 051505 020124
      013150 005015 000377
1669 013154 005015 020040 040524 M97: .ASCIZ <15><12>/ TAB, BACKSPACE AND BELL TEST /<15><12><377>
      013162 026102 041040 041501
      013170 051513 040520 042503
      013176 040440 042116 041040
      013204 046105 020114 042524
      013212 052123 020040 005015
      013220 000377
1670 013222 005015 020040 051105 M910: .ASCIZ <15><12>/ ERASE LINE TEST /<15><12><377>
      013230 051501 020105 044514
      013236 042516 052040 051505
      013244 020124 006440 177412

```

1671	013252	000				
	013253	015	020012	042440	M911:	.ASCIZ <15><12>/ ERASE SCREEN TEST /<15><12><377>
	013260	040522	042523	051440		
	013266	051103	042505	020116		
	013274	042524	052123	020040		
	013302	005015	000377			
1672	013306	005015	020040	044526	M912:	.ASCIZ <15><12>/ VIDEO COUPLING TEST /<15><12><377>
	013314	042504	020117	047503		
	013322	050125	044514	043516		
	013330	052040	051505	020124		
	013336	005015	000377			
1673	013342	044033	045033	005015	M914:	.ASCIZ <33><110><33><112><15><12>/ TERMINAL IDENTIFIER TEST /<15><12><377>
	013350	020040	042524	046522		
	013356	047111	046101	044440		
	013364	042504	052116	043111		
	013372	042511	020122	042524		
	013400	052123	006440	177412		
	013406	000				
1674	013407	015	020012	041440	M920:	.ASCIZ <15><12>/ COPIER PERIMETER/<15><12><377>
	013414	050117	042511	020122		
	013422	042520	044522	042515		
	013430	042524	006522	177412		
	013436	000				
1675	013437	015	020012	042040	M921:	.ASCIZ <15><12>/ DISCLAIMER STATEMENT/<15><12><377>
	013444	051511	046103	044501		
	013452	042515	020122	052123		
	013460	052101	046505	047105		
	013466	006524	177412	000		
1676	013473	015	020012	044040	M922:	.ASCIZ <15><12>/ HOLD SCREEN MODE TEST/<15><12><377>
	013500	046117	020104	041523		
	013506	042522	047105	046440		
	013514	042117	020105	042524		
	013522	052123	005015	000377		
1677	013530	005015	020040	051107	M9221:	.ASCIZ <15><12>/ GRAPHICS MODE AND REV. LINE FEED TEST/<15><12><377>
	013536	050101	044510	051503		
	013544	046440	042117	020105		
	013552	047101	020104	042522		
	013560	027126	046040	047111		
	013566	020105	042506	042105		
	013574	052040	051505	006524		
	013602	177412	000			
1678	013605	033	015537	015510	M923:	.ASCIZ <33><137><33><110><33><112>/ AUTO COPY MODE TEST/<15><12><377>
	013612	020112	040440	052125		
	013620	020117	047503	054520		
	013626	046440	042117	020105		
	013634	042524	052123	005015		
	013642	000377				
1679	013644	044033	045033	051120	MPTCNT:	.ASCIZ <33><110><33><112>/PRINTER CONTROLLER MODE ENTERED/<15><12><377>
	013652	047111	042524	020122		
	013660	047503	052116	047522		
	013666	046114	051105	046440		
	013674	042117	020105	047105		
	013702	042524	042522	006504		
	013710	177412	000			
1680	013713	033	015510	050112	MPTSCN:	.ASCIZ <33><110><33><112>/PRINT A SCREEN OF E'S/<15><12><377>
	013720	044522	052116	040440		

ASCII MESSAGES

1591	0113126 0113136 0113146 0113156 0113166 0113176 0113186 0113196 0114005 0114015 0114025 0114034 0114041 0114054 0114062 0114070 0114102 0114110 0114116 0114124 0114132 0114140 0114146 0114154 0114162 0114163 0114170 0114176 0114204 0114212 0114220 0114226 0114234 0114237 0114244 0114252 0114260 0114266 0114274 0114300 0114306 0114314 0114322 0114330 0114336 0114344 0114352 0114360 0114363 0114370 0114376 0114404 0114412	051440 020116 051447 015510 052125 047111 042504 006524 015 041131 041440 052123 051505 122 042523 021040 021124 005015 052040 044107 040440 020054 051505 020105 052101 046511 000 015 052122 044527 042510 051040 041530 044124 051111 040 051040 042440 042116 046040 020124 005015 000377 005015 044524 052111 020110 047524 020127 042503 042510 020124 177412 015 052122 020110 042523 051040	051103 024311 005015 020112 020117 020124 052040 177412 020012 040517 040510 051105 006524 046105 052040 044123 045440 000377 042514 020117 020124 051040 042504 020123 042513 040440 006505 051412 047111 044124 052040 053517 050105 020105 104 051106 043511 042116 046040 042513 000377 052123 043516 020110 047524 020127 052120 046040 042513 000 051412 053440 044124 047503 053517	042505 042440 000377 040440 051120 047515 051505 000 042513 042122 040522 052040 012 040505 042510 043111 054505 052106 044527 047111 053517 051120 047117 020131 052040 177412 040524 020107 052040 050117 042440 020124 044124 046517 052110 040440 051501 051531 051101 053440 044124 020120 054105 052040 051501 006531 040524 052111 020105 042116 005015	M923A: .ASCIZ <110><33><112>/ AUTO PRINT MODE TEST/<15><12><377> MK8: .ASCII <15><12>/ KEYBOARD CHARACTER TEST/<15><12> MK81: .ASCIZ /RELEASE THE "SHIFT" KEY/<15><12><377> MK8A: .ASCIZ <15><12>/LEFT TO RIGHT IN A ROW, DEPRESS ONE KEY AT A TIME/<15><12><377> MK8B: .ASCII <15><12>/STARTING WITH THE TOP ROW EXCEPT THE THIRD/ MK8B2: .ASCIZ <15><12>/STARTING WITH THE TOP ROW EXCEPT THE LAST KEY/<15><12><377> MK8C: .ASCIZ <15><12>/START WITH THE SECOND ROW/<15><12><377>
------	---	--	--	--	--

02V*00.P13 ASCII MESSAGES

1699	014420	000377		
	014422	005015	052123	051101
	014424	020124	044527	044124
	014426	052040	042510	052040
	014428	044510	042122	051040
	014430	053517	042440	044530
	014432	050105	020124	044124
	014434	020105	052103	046122
1690	014436	005015	040440	042116
	014438	052040	042510	021040
	014440	046102	047101	021113
	014442	177412	054505	006523
	014444	000	000	
1691	014446	015	051412	040524
	014448	052122	053440	052111
	014450	020110	044124	020105
	014452	044124	051111	020104
	014454	047522	020127	041054
	014456	043505	047111	051040
	014458	053517	053440	052111
	014460	020110	040442	020042
	014462	042513	006531	177412
	014464	000		
1692	014466	015	051412	040524
	014468	052122	053440	052111
	014470	020110	044124	020105
	014472	047506	051125	044124
	014474	051040	053517	042440
	014476	041530	050105	020124
	014478	044124	020105	041523
	014480	047522	046114	
1693	014482	005015	051454	044510
	014484	052106	020054	042522
	014486	042520	052101	040440
	014488	042116	040440	052125
	014490	026517	051120	047111
	014492	006524	177412	000
1694	014494	015	051412	040524
	014496	052122	053440	052111
	014498	020110	044124	020105
	014500	044506	052106	020110
	014502	047522	006527	177412
	014504	000		
1695	014506	015	047012	053517
	015000	044040	046117	020104
	015006	047504	047127	052040
	015014	042510	021040	042514
	015022	052106	051455	044510
	015030	052106	020042	042513
	015036	006531	177412	
1696	015042	005015	047516	020127
	015050	047510	042114	042040
	015056	053517	020116	044124
	015064	020105	051042	043511
	015072	052110	051455	044510
	015100	052106	020042	042513

MKBD: .ASCII <15><12> START WITH THE THIRD ROW EXCEPT THE CTRL/

.ASCIZ <15><12> AND THE "BLANK" KEYS/<15><12><377>

MKBD2: .ASCIZ <15><12> START WITH THE THIRD ROW .BEGIN ROW WITH "A" KEY/<15><12><377>

MKBE: .ASCII <15><12> START WITH THE FOURTH ROW EXCEPT THE SCROLL/

.ASCIZ <15><12> ,SHIFT, REPEAT AND AUTO-PRINT/<15><12><377>

MKBF: .ASCIZ <15><12> START WITH THE FIFTH ROW/<15><12><377>

MKBG: .ASCII <15><12> NOW HOLD DOWN THE "LEFT-SHIFT" KEY/<15><12><377>

MKBGA: .ASCII <15><12> NOW HOLD DOWN THE "RIGHT-SHIFT" KEY/<15><12><377>

1697	015106 015112 015120 015126 015134 015142 015150	006531 005015 047510 053517 020105 021114 005015	177412 047516 042114 020116 041442 045440 377	020127 042040 044124 051124 054505	MKBH: .ASCII <15><12>/NOW HOLD DOWN THE "CTRL" KEY/<15><12><377>
1698	015158 015160 015168 015174 015202 015208	052015 052122 020110 047524 006527 005015	051412 053440 044124 020120 177412 052123	040524 052111 020105 047522	MKBI: .ASCII <15><12>/START WITH THE TOP ROW/<15><12><377>
1699	015214 015222 015230 015236 015241 015246	020124 052040 042116 005015 051412 052122	044527 042510 051040 377 040524 053440	051101 044124 031040 053517	MKBJ: .ASCII <15><12>/START WITH THE 2ND ROW/<15><12><377>
1700	015254 015262 015270 015275 015302 015310	044124 031440 053517 051412 052122 020110	052040 042122 005015 377 040524 053440	044527 042510 051040	MKBK: .ASCII <15><12>/START WITH THE 3RD ROW/<15><12><377>
1701	015316 015324 015330 015336 015344 015352	052064 006527 005015 051412 052122 020110	020110 177412 052123 044527 042510 020124	047522 046040 047522	MKBL: .ASCII <15><12>/START WITH THE 4TH ROW/<15><12><377>
1702	015360 015364 015372 015400 015406	006527 005015 044060 040520 052123	177412 053012 045440 020104 005015	377 032524 054505 042524	MKBM: .ASCII <15><12>/START WITH THE LAST ROW/<15><12><377>
1703	015406 015413 015420 015426 015434 015442	006527 005015 020062 047122 042513 046440	177412 053012 046101 052101 050131 042117	377 032524 042524 020105 042101 020105	MKBN: .ASCII <15><12><12>/VT50M KEYPAD TEST/<15><12><377>
1704	015450 015456 015457 015464 015472	042524 377 015 040522 053440	052123 005015 053012 052103 051501	005015 377 032524 042524 020105	MKB52: .ASCII <15><12>/VT52 ALTERNATE KEYPAD MODE TEST/<15><12><377>
1705	015472 015500 015506 015511 015516	042524 377 015 040522 053440	052123 005015 053012 041412 052103	005015 377 032524 040510 051105	MKBG: .ASCII <15><12>/CHARACTER WAS IN ERROR/<15><12>
1706	015516 015520 015523 015524 015527	020075 040 040 040 040	047517 040 040 040 040	020104 075 040 040 040	.ASCII /GOOD = /
1707	015527 015531 015536	040 102 040	040 042101 036440	040 040 036440	MKBQ1: .BYTE 40,40,75,40
1708					MKBQA: .BYTE 40,40,40,40,40
1709					.ASCII /BAD = /

1710	015537	040	040	075	MKBQ2: .BYTE	40,40,75,40
1711	015542	040	040	040	MKBQ8: .BYTE	40,40,40,15,12,377,0
	015543	040	040	040		
	015546	015	012	377		
	015551	000				
1712	015552	005015	042513	041131	MKBR: .ASCIZ	<15><12>/KEYBOARD CHARACTER TEST COMPLETE/<15><12><377>
	015560	040517	042122	041440		
	015566	040510	040522	052103		
	015574	051105	052040	051505		
	015602	020124	047503	050115		
	015610	042514	042524	005015		
	015616	000377				
1713						
1714	015620	005015	045440	054505	MKE: .ASCII	<15><12>/KEYBOARD ASCII AND OCTAL LOOP/<15><12>
	015626	047502	051101	020104		
	015634	051501	044503	020111		
	015642	047101	020104	041517		
	015650	040524	020114	047514		
	015656	050117	005015			
1715	015662	015	012		.BYTE	15,12
1716	015664	044127	047105	040440	.ASCII	/WHEN A KEY IS DEPRESSED, THE ASCII CHARACTER AND/
	015672	045440	054505	044440		
	015700	020123	042504	051120		
	015706	051505	042523	026104		
	015714	052040	042510	040440		
	015722	041523	044511	041440		
	015730	040510	040522	052103		
	015736	051105	040440	042116		
1717	015744	015	012		.BYTE	15,12
1718	015746	052040	042510	052040	.ASCIZ	/THE THREE DIGIT OCTAL CODE WILL BE ECHOED/
	015754	051110	042505	042040		
	015762	043511	052111	047440		
	015770	052103	046101	041440		
	015776	042117	020105	044527		
	016004	046114	041040	020105		
	016012	041505	047510	042105		
	016020	000				
1719	016021	015	012	377	.BYTE	15,12,377,0
	016024	000				
1720	016025	015	041412	040510	MKEA: .ASCII	<15><12>/CHAR = /
	016032	020122	020075			
1721	016036	040	040	040	MKEA1: .BYTE	40,40,40,75,40
	016041	075	040			
1722	016043	040	040	040	MKEB: .BYTE	40,40,40,15,12,377,0
	016046	015	012	377		
	016051	000				
1723	016052	005015	042513	041131	MKEH: .ASCIZ	<15><12>/KEYBOARD ECHO LOOP/<15><12>
	016060	040517	042122	042440		
	016066	044103	020117	047514		
	016074	050117	005015	000		
1724	016101	015	012	012	CRLF: .BYTE	15,12,12,12,12,12,12,12
	016104	012	012	012		
	016107	012	012	012		
1725	016112	012	012	012	CRLFA: .BYTE	12,12,12,12,12,377,0
	016115	012	012	377		
	016120	000				

1726
1727 016121 033 110 033
016124 112 377 000
1729 016127 015 052012 040040
016134 040100 040100 040100
016142 040100 040100 040100
016150 040100 040100 040100
016156 040100 040100 040100
016164 040100 040100 040100
016172 040100 040100 040100
016200 100
1729 016201 100 040100 040100
016206 040100 040100 040100
016214 040100 040100 040100
016222 040100 040100 040100
016230 040100 040100 040100
016236 040100 040100 040100
016244 040100 020100 124
1730 016251 015 052012 042440
016256 042505 042505 042505
016264 042505 042505 042505
016272 042505 042505 042505
016300 042505 042505 042505
016306 042505 042505 042505
016314 042505 042505 042505
016322 105
1731 016323 105 042505 042505
016330 042505 042505 042505
016336 042505 042505 042505
016344 042505 042505 042505
016352 042505 042505 042505
016360 042505 042505 042505
016366 042505 020105 124
1732 016373 377 000 000
1733 016376 005015 042524 052123
016404 047111 020107 052126
016412 030065 020101 030450
016420 020062 044514 042516
016426 026523 047516 041440
016434 050117 042511 024522
1734 016442 015 012 012
016445 377 000 000
1735 016450 005015 042524 052123
016456 047111 020107 052126
016464 030065 020102 030450
016472 020062 044514 042516
016500 026523 047503 044520
016506 051105 051
1736 016511 015 012 012
016514 377 000 000
1737 016517 015 052012 051505
016524 044524 043516 053040
016532 032524 020065 031050
016540 020064 044514 042516
016546 026523 047503 044520
016554 051105 051

:HOME AND ERASE SCREEN OPCODE
HOMERS: .BYTE 33,110,33,112,377,0
PATH: .ASCII <15><12>/T ~~~~~/

.ASCII /~~~~~ T/

.ASCII <15><12>/T EEE/

.ASCII /EE T/

.BYTE 377,0,0
VT50A: .ASCII <15><12>/TESTING VT50A (12 LINES-NO COPIER)/

.BYTE 15,12,12,377,0,0
VT50B: .ASCII <15><12>/TESTING VT50B (12 LINES-COPIER)/

.BYTE 15,12,12,377,0,0
VT55: .ASCII <15><12>/TESTING VT55 (24 LINES-COPIER)/

1738	016557	015	012	012	.BYTE	15,12,12,377,0,0
	016562	377	000	000		
1739	016565	015	052012	051505	VT50H:	.ASCII <15><12>/TESTING VT50H (12 LINES-COPIER-D.C.A.)/
	016572	044524	043516	053040		
	016600	032524	044060	024040		
	016606	031061	046040	047111		
	016614	051505	041455	050117		
	016622	042511	026522	027104		
	016630	027103	027101	051		
1740	016635	015	012	012	.BYTE	15,12,12,377,0,0
	016640	377	000	000		
1741	016643	015	052012	051505	VT52K:	.ASCII <15><12>/TESTING VT52 (24 LINES-NO COPIER OR PRINTER/
	016650	044524	043516	053040		
	016656	032524	020062	031050		
	016664	020064	044514	042516		
	016672	026523	047516	041440		
	016700	050117	042511	020122		
	016706	051117	050040	044522		
	016714	052116	051105			
1742	016720	015	012	012	.BYTE	15,12,12,377,0,0
	016723	377	000	000		
1743	016726	005015	042524	052123	VT52L:	.ASCII <15><12>/TESTING VT52 (24 LINES,COPIER-NO PRINTER)/
	016734	047111	020107	052126		
	016742	031065	024040	032062		
	016750	046040	047111	051505		
	016756	041454	050117	042511		
	016764	026522	047516	050040		
	016772	044522	052116	051105		
	017000	051				
1744	017001	015	012	012	.BYTE	15,12,12,377,0,0
	017004	377	000	000		
1745	017007	015	052012	051505	VT52M:	.ASCII <15><12>/TESTING VT52 (24 LINES,PRINTER-NO COPIER)/
	017014	044524	043516	053040		
	017022	032524	020062	031050		
	017030	020064	044514	042516		
	017036	026123	051120	047111		
	017044	042524	026522	047516		
	017052	041440	050117	042511		
	017060	024522				
1746	017062	015	012	012	.BYTE	15,12,12,377,0,0
	017065	377	000	000		
1747	017070	033	132	377	RFI:	.BYTE 33,132,377,0,0
	017073	000	000			
1748	017075	033	015534	006537	MQ0:	.ASCIZ <33><134><33><137><15><12><12>/LOOP ON TEST PATTERN LETTER (A THRU Z) ?
	017102	005012	047514	050117		
	017110	047440	020116	042524		
	017116	052123	050040	052101		
	017124	042524	047122	046040		
	017132	052105	042524	020122		
	017140	040450	052040	051110		
	017146	020125	024532	037440		
	017154	036440	020040	000377		
1749	017162	056033	057433	005015	MQ1:	.ASCIZ <33><134><33><137><15><12><12>/START AT TEST PATTERN LETTER (A THRU Z) ?
	017170	051412	040524	052122		
	017176	040440	020124	042524		
	017204	052123	050040	052101		

H05

	017212	042524	047122	046040	
	017220	052105	042524	020122	
	017226	040450	052040	051110	
	017234	020125	024532	037440	
	017242	036440	020040	000377	
1750	017250	005015	052012	054522	MQ2: .ASCIZ <15><12><12>/TRY AGAIN /<15><12><377>
	017256	040440	040507	047111	
	017264	006440	177412	000	
1751	017271	033	135	012	GOPRNT: .BYTE 33,135,12,12,0,0,0,0,0,0,0,0,0,0,0,0,0377,0,0
	017274	012	000	000	
	017277	000	000	000	
	017302	000	000	000	
	017305	000	000	000	
	017310	000	000	000	
	017313	000	377	000	
	017316	000			
1752	017317	033	006533	052012	GOMDSC: .ASCIZ <33><133><15><12>/TESTING SILO REG/<377>
	017324	051505	044524	043516	
	017332	051440	046111	020117	
	017340	042522	177507	000	
1753	017345	033	006534	051412	GODSHS: .ASCIZ <33><134><15><12>/SILO TEST DONE/<15><12><377>
	017352	046111	020117	042524	
	017360	052123	042040	047117	
	017366	006505	177412	000	
1754	017373	033	136	377	GOAPMD: .BYTE 33,136,377,0,0
	017376	000	000		
1755	017400	033	137	015	GONAPM: .BYTE 33,137,15,12,12,12,0,0,0,0,0,0,0,0,0,0,0.377,0,0,0,0
	017403	012	012	012	
	017406	000	000	000	
	017411	000	000	000	
	017414	000	000	000	
	017417	000	000	000	
	017422	377	000	000	
	017425	000	000		
1756	017427	033	015510	020112	M98: .ASCIZ <33><110><33><112>/ DIRECT CURSOR ADDRESSING TEST/<377>
	017434	020040	044504	042522	
	017442	052103	041440	051125	
	017450	047523	020122	042101	
	017456	051104	051505	044523	
	017464	043516	052040	051505	
	017472	177524	000		
1757	017475	015	005012	044506	WHAT0: .ASCIZ <15><12><12>/FIRST LINE DEVICE ADDRESS ? = /<200>
	017502	051522	020124	044514	
	017510	042516	042040	053105	
	017516	041511	020105	042101	
	017524	051104	051505	020123	
	017532	020077	020075	000200	
1758	017540	005015	046012	051501	WHAT1: .ASCIZ <15><12><12>/LAST LINE DEVICE ADDRESS (CR IF NONE) ? = /<200>
	017546	020124	044514	042516	
	017554	042040	053105	041511	
	017562	020105	042101	051104	
	017570	051505	020123	041450	
	017576	020122	043111	047040	
	017604	047117	024505	037440	
	017612	036440	100040	000	
1759	017617	015	020012	020040	PASHED: .ASCIZ <15><12>/ PASS * * OF ERRORS /

	017624	020040	020040	020040		
	017632	050040	051501	020123		
	017640	020043	020040	021440		
	017646	047440	020106	051105		
	017654	047522	051522	020040		
	017662	000040				
1760	017664	051105	047522	020122	EM1:	.ASCIZ /ERROR FLAG SET ON TRANSMITTER STATUS/
	017672	046106	043501	051440		
	017700	052105	047440	020116		
	017706	051124	047101	046523		
	017714	052111	042524	020122		
	017722	052123	052101	051525		
	017730	000				
1761	017731	116	020117	047111	EM2:	.ASCIZ /NO INPUT FLAG DETECTED/
	017736	052520	020124	046106		
	017744	043501	042040	052105		
	017752	041505	042524	000104		
1762	017760	047111	047503	051122	EM3:	.ASCIZ /INCORRECT I.D. CODE/
	017766	041505	020124	027111		
	017774	027104	041440	042117		
	020002	000105				
1763	020004	047125	054105	042520	EM4:	.ASCIZ /UNEXPECTED OR INCORRECT INPUT CHAR/
	020012	052103	042105	047440		
	020020	020122	047111	047503		
	020026	051122	041505	020124		
	020034	047111	052520	020124		
	020042	044103	051101	000		
1764	020047	111	053116	046101	EM5:	.ASCIZ /INVALID BUSS ADDRESS, TRY AGAIN/
	020054	042111	041040	051525		
	020062	020123	042101	051104		
	020070	051505	026123	052040		
	020076	054522	040440	040507		
	020104	047111	000			
1765	020107	105	051122	041520	DH1:	.ASCIZ /ERRPC VTNOW TSTNUM/
	020114	020040	053040	047124		
	020122	053517	020040	052040		
1766	020130	052123	052516	000115	DH3:	.ASCIZ /ERRPC VTNOW 1ST WD 2ND WD 3RD WD/
	020136	051105	050122	020103		
	020144	020040	052126	047516		
	020152	020127	020040	051461		
	020160	020124	042127	020040		
	020166	047062	020104	042127		
	020174	020040	051063	020104		
	020202	042127	000			
1767	020205	105	051122	041520	DH4:	.ASCIZ /ERRPC VTNOW TSTNUM EXPCT RECV/
	020212	020040	053040	047124		
	020220	053517	020040	052040		
	020226	052123	052516	020115		
	020234	020040	054105	041520		
	020242	020124	020040	042522		
	020250	053103	000			
1768		020254				.EVEN
1769	020254	001122	001244	001250	DT1:	SERRPC,VTNOW,TSTNUM,0
	020262	000000				
1770	020264	001122	001244	001326	DT3:	SERRPC,VTNOW,SAVE4,SAVE2,SAVE3,0
	020272	001322	001324	000000		

```

1771 020300 001122 001244 001250 DT4: $ERRPC,VTNOW,TSTNUM,$GDDAT,$BDDAT,0
      020306 001130 001132 000000
1772
1773
1774
1775
1776
1777 020314 020350 020404 020442 VSORW: ROW1,ROW2,ROW3,ROW4,ROW15,ROW15,ROW1C
      020322 020472 020672 020672
      020330 020764
1778 020332 020520 020556 020614 VS2RW: ROW12,ROW22,ROW32,ROW42,ROW125,ROW125,ROW12C
      020340 020646 020726 020726
      020346 021020
1779
1780
1781 020350 000033 000061 000062 ROW1: .WORD 33,61,62,63,64,65,66,67,70,71,60,55,75,100010
      020356 000063 000064 000065
      020364 000066 000067 000070
      020372 000071 000060 000055
      020400 000075 100010
1782 020404 000011 000121 000127 ROW2: .WORD 11,121,127,105,122,124,131,125,111,117,120,133,134,12,100177
      020412 000105 000122 000124
      020420 000131 000125 000111
      020426 000117 000120 000133
      020434 000134 000012 100177
1783 020442 000101 000123 000104 ROW3: .WORD 101,123,104,106,107,110,112,113,114,73,47,100015
      020450 000106 000107 000110
      020456 000112 000113 000114
      020464 000073 000047 100015
1784 020472 000132 000130 000103 ROW4: .WORD 132,130,103,126,102,116,115,54,56,100057
      020500 000126 000102 000116
      020506 000115 000054 000056
      020514 100057
1785 020516 100040 ROW5: .WORD 100040
1786 ;VT52 KEYBOARD EQUIVALENCES(LOWER CASE CHAR.)
1787
1788 020520 000033 000061 000062 ROW12: .WORD 33,61,62,63,64,65,66,67,70,71,60,55,75,140,100010
      020526 000063 000064 000065
      020534 000066 000067 000070
      020542 000071 000060 000055
      020550 000075 000140 100010
1789 020556 000011 000161 000167 ROW22: .WORD 11,161,167,145,162,164,171,165,151,157,160,133,134,12,100177
      020564 000145 000162 000164
      020572 000171 000165 000151
      020600 000157 000160 000133
      020606 000134 000012 100177
1790 020614 000141 000163 000144 ROW32: .WORD 141,163,144,146,147,150,152,153,154,73,47,173,100015
      020622 000146 000147 000150
      020630 000152 000153 000154
      020636 000073 000047 000173
      020644 100015
1791 020646 000172 000170 000143 ROW42: .WORD 172,170,143,166,142,156,155,54,56,100057
      020654 000166 000142 000156
      020662 000155 000054 000056
      020670 100057
1792

```

1793
1794
1795

:SHIFTED ROW CODES

020672 000033 000041 000100 ROW1S: .WORD 33,41,100,43,44,45,136,46,52,50,51,137,53,100010
020700 000043 000044 000045
020706 000136 000046 000052
020714 000050 000051 000137

1796

020722 000053 100010
020726 000033 000041 000100 ROW12S: .WORD 33,41,100,43,44,45,136,46,52,50,51,137,53,176,100010
020734 000043 000044 000045
020742 000136 000046 000052
020750 000050 000051 000137
020756 000053 000176 100010

1797
1798
1799
1800

:CONTROL ROW CODES

020764 000033 000021 000022 ROW1C: .WORD 33,21,22,23,24,25,26,27,30,31,20,15,35,100010
020772 000023 000024 000025
021000 000026 000027 000030
021006 000031 000020 000015
021014 000035 100010

1801

021020 000033 000021 000022 ROW12C: .WORD 33,21,22,23,24,25,26,27,30,31,20,15,35,0,100010
021026 000023 000024 000025
021034 000026 000027 000030
021042 000031 000020 000015
021050 000035 000000 100010

1802
1803

:VT52 ESCAPE SEQUENCES

021056 033 111 015 REVLf: .BYTE 33,111,015,377,0 ;REVERSE LINE FEED.
021061 377 000
1805
021063 033 075 377 ENAKP: .BYTE 33,075,377,0,0 ;ENABLE ALTERNATE KEYPAD MODE.
021066 000 000
1807 021070 033 076 377 EXAKP: .BYTE 33,076,377,0,0 ;EXIT ALTERNATE KEYPAD MODE
021073 000 000
1808
021075 033 106 377 ENGRAF: .BYTE 33,106,377,0,0 ;ENTER GRAPHICS MODE.
021100 000 000
1810 021102 033 107 377 EXGRAF: .BYTE 33,107,377,0,0 ;EXIT GRAPHICS MODE.
021105 000 000
1811
021107 033 127 377 ENPNTM: .BYTE 33,127,377,0,0 ;ENABLE PRINTER CONTROLLER MODE.
021112 000 000
1813 021114 033 130 377 EXPNTM: .BYTE 33,130,377,0,0 ;DISABLE PRINTER CONTROLLER MODE.
021117 000 000

1814
1815
1816
1817
1818

:VT50-H KEYPAD CODES

021122 021122 .EVEN
1819 021122 000033 000120 000033 ROW6: .WORD 33,'P',33,'Q',33,'R',33,100101
021130 000121 000033 000122
021136 000033 100101
1820 021142 000067 000070 000071 ROW7: .WORD 67,70,71,33,100102
021150 000033 100102
1821 021154 000064 000065 000066 ROW8: .WORD 64,65,66,33,100103
021152 000033 100103

1822	021166	000061	000062	000063	ROW9:	.WORD	61,62,63,33,100104
	021174	000033	100104				
1823	021200	000060	000056	100015	ROW10:	.WORD	60,56,100015
1824	021206	000033	000120	000033	ROW6A:	.WORD	33,'P',33,'Q',33,'R',33,100101
	021214	000121	000033	000122			
	021222	000033	100101				
1825	021226	000033	000077	000167	ROW7A:	.WORD	33,'?',167,33,'?',170,33,'?',171,33,100102
	021234	000033	000077	000170			
	021242	000033	000077	000171			
	021250	000033	100102				
1826	021254	000033	000077	000164	ROW8A:	.WORD	33,'?',164,33,'?',165,33,'?',166,33,100103
	021252	000033	000077	000165			
	021270	000033	000077	000166			
	021276	000033	100103				
1827	021302	000033	000077	000161	ROW9A:	.WORD	33,'?',161,33,'?',162,33,'?',163,33,100104
	021310	000033	000077	000162			
	021316	000033	000077	000163			
	021324	000033	100104				
1828	021330	000033	000077	000160	ROW10A:	.WORD	33,'?',160,33,'?',156,33,'?',100115
	021336	000033	000077	000156			
	021344	000033	000077	100115			
1829	021352	005015	052012	044510	MTEXT0:	.ASCIZ	<15><12><12>/THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR
	021360	020123	047523	052106			
	021366	040527	042522	044440			
	021374	020123	052506	047122			
	021402	051511	042510	020104			
	021410	047524	050040	051125			
	021416	044103	051501	051105			
	021424	052440	042116	051105			
	021432	040440	046040	041511			
	021440	047105	042523	043040			
	021446	051117	052440	042523			
	021454	000377					
1830	021456	005015	047117	040440	MTEXT1:	.ASCIZ	<15><12>/ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION<
	021464	051440	047111	046107			
	021472	020105	047503	050115			
	021500	052125	051105	051440			
	021506	051531	042524	020115			
	021514	047101	020104	040503			
	021522	020116	042502	041440			
	021530	050117	042511	020104			
	021536	053450	052111	020110			
	021544	047111	046103	051525			
	021552	047511	177516	000			
1831	021557	015	047412	020106	MTEXT2:	.ASCIZ	<15><12>/OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT/
	021564	042504	023503	020123			
	021572	047503	054520	044522			
	021600	044107	020124	047516			
	021606	044524	042503	020051			
	021614	047117	054514	043040			
	021622	051117	052440	042523			
	021630	044440	020116	052523			
	021636	044103	051440	051531			
	021644	042524	026115	042440			
	021652	041530	050105	177524			
	021660	000					

1832	021661	015	040412	020123	MTEXT3: .ASCIZ <15><12>/AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC./<377>
	021666	040515	020131	052117	
	021674	042510	053522	051511	
	021702	020105	042502	050040	
	021710	047522	044526	042504	
	021716	020104	047111	053440	
	021724	044522	044524	043516	
	021732	041040	020131	042504	
	021740	027103	000377		
1833					
1834	021744	005015	052012	042510	MTEXT4: .ASCIZ <15><12><12>/THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHC
	021752	044440	043116	051117	
	021760	040515	044524	047117	
	021766	044440	020116	044124	
	021774	051511	042040	041517	
	022002	046525	047105	020124	
	022010	051511	051440	041125	
	022016	042512	052103	052040	
	022024	020117	044103	047101	
	022032	042507	053440	052111	
	022040	047510	052125	000377	
1835	022046	005015	047516	044524	MTEXT5: .ASCIZ <15><12>/NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL/<
	022054	042503	040440	042116	
	022062	051440	047510	045125	
	022070	020104	047516	020124	
	022076	042502	041440	047117	
	022104	052123	052522	042105	
	022112	040440	020123	020101	
	022120	047503	046515	052111	
	022126	042515	052116	041040	
	022134	020131	044504	044507	
	022142	040524	177514	000	
1836	022147	015	042412	052521	MTEXT6: .ASCIZ <15><12>/EQUIPMENT CORPORATION./<377>
	022154	050111	042515	052116	
	022162	041440	051117	047520	
	022170	040522	044524	047117	
	022176	177456	000		
1837	022201	015	005012	042504	MTEXT7: .ASCIZ <15><12><12>/DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
	022206	020103	051501	052523	
	022214	042515	020123	047516	
	022222	051040	051505	047520	
	022230	051516	041111	046111	
	022236	052111	020131	047506	
	022244	020122	044124	020105	
	022252	051525	020105	051117	
	022260	051040	046105	040511	
	022266	044502	044514	054524	
	022274	047440	177506	000	
1838	022301	015	044412	051524	MTEXT8: .ASCII <15><12>/ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC./
	022306	051440	043117	053524	
	022314	051101	020105	047117	
	022322	042440	052521	050111	
	022330	042515	052116	053440	
	022336	044510	044103	044440	
	022344	020123	047516	020124	
	022352	052523	050120	044514	

1839 022360 042105 041040 020131
022366 042504 027103
022372 015 012 377
022375 000

.BYTE 15,12,377,0

1840
1841

.EVEN

.SBTTL TTY INPUT ROUTINE

;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

;*CALL:

;* RDCHR ;: INPUT A SINGLE CHARACTER FROM THE TTY
;* RETURN HERE ;: CHARACTER IS ON THE STACK

(1) 022376 011646
(1) 022400 016666 000004 000002
(1) 022406 105777 156530
(1) 022412 100375
(1) 022414 117766 156524 000004
(1) 022422 042766 177600 000004
(1) 022430 000002

\$RDCHR: MOV (SP), -(SP) ;: PUSH DOWN THE PC
MOV 4(SP), 2(SP) ;: SAVE THE PS
1\$: TSTB @STKS ;: WAIT FOR
BPL 1\$;: A CHARACTER
MOVB @STKB, 4(SP) ;: READ THE TTY
BIC #1C<177>, 4(SP) ;: GET RID OF JUNK IF ANY
RTI ;: GO BACK TO USER

;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY

;*CALL:

;* RDLIN ;: INPUT A STRING FROM THE TTY
;* RETURN HERE ;: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
;* ;: TERMINATOR WILL BE A BYTE OF ALL 0'S

(1) 022432 010346
(1) 022434 012703 022540
(1) 022440 022703 022550
(1) 022444 101405
(1) 022446 104405
(1) 022450 112613
(1) 022452 122713 000177
(1) 022456 001003
(1) 022460 104400 001164
(1) 022464 000763
(1) 022466 111337 022536
(1) 022472 104400 022536
(1) 022476 122723 000015
(1) 022502 001356
(1) 022504 105063 177777
(1) 022510 104400 001166
(1) 022514 012603
(1) 022516 011646
(1) 022520 016666 000004 000002
(1) 022526 012766 022540 000004
(1) 022534 000002
(1) 022536 000
(1) 022537 000
(1) 022540 000010

\$RDLIN: MOV R3, -(SP) ;: SAVE R3
1\$: MOV #STYIN, R3 ;: GET ADDRESS
2\$: CMP #STYIN+8., R3 ;: BUFFER FULL?
BLOS 4\$;: BR IF YES
RDCHR ;: GO READ ONE CHARACTER FROM THE TTY
MOVB (SP)+, (R3) ;: GET CHARACTER
CMPB #177, (R3) ;: IS IT A RUBOUT
BNE 3\$;: SKIP IF NOT
4\$: TYPE \$QUES ;: TYPE A '?'
BR 1\$;: CLEAR THE BUFFER AND LOOP
3\$: MOVB (R3), 9\$;: ECHO THE CHARACTER
TYPE 9\$
CMPB #15, (R3)+ ;: CHECK FOR RETURN
BNE 2\$;: LOOP IF NOT RETURN
CLRB -1(R3) ;: CLEAR RETURN (THE 15)
TYPE \$LF ;: TYPE A LINE FEED
MOV (SP)+, R3 ;: RESTORE R3
MOV (SP), -(SP) ;: ADJUST THE STACK AND PUT ADDRESS OF THE
MOV 4(SP), 2(SP) ;: FIRST ASCII CHARACTER ON IT
MOV #STYIN, 4(SP)
RTI ;: RETURN
9\$: .BYTE 0 ;: STORAGE FOR ASCII CHAR. TO TYPE
.BYTE 0 ;: TERMINATOR
\$TTYIN: .BLKB 8. ;: RESERVE 8 BYTES FOR TTY INPUT

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

1842
(1)
(1)

```

:*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
:*CHANGE IT TO BINARY.
:*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
:*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
:*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
:*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
:*CALL:

```

```

:*          $ROOCT          :: READ AN OCTAL NUMBER
:*          RETURN HERE    :: LOW ORDER BITS ARE ON TOP OF THE STACK
:*                          :: HIGH ORDER BITS ARE IN $HIOCT

```

```

(1) 022551 011646 000004 000002 $ROOCT: MOV (SP),-(SP) :: PROVIDE SPACE FOR THE
(1) 022552 016666 000004 000002 MOV 4(SP),2(SP) :: INPUT NUMBER
(1) 022553 010046 MOV RO,-(SP) :: PUSH RO ON STACK
(1) 022554 010146 MOV R1,-(SP) :: PUSH R1 ON STACK
(1) 022555 010246 MOV R2,-(SP) :: PUSH R2 ON STACK
(1) 022556 104406 15: RDLIN :: READ AN ASCII LINE
(1) 022557 012600 MOV (SP)+,RO :: GET ADDRESS OF 1ST CHARACTER
(1) 022558 010037 022676 MOV RO,$$ :: AND SAVE IT
(1) 022559 005001 CLR R1 :: CLEAR DATA WORD
(1) 022560 005002 CLR R2
(1) 022561 112046 25: MOV8 (RO)+,-(SP) :: PICKUP THIS CHARACTER
(1) 022562 001420 BEQ 3$ :: IF ZERO GET OUT
(1) 022563 122716 000060 CMPB #'0,(SP) :: MAKE SURE THIS CHARACTER
(1) 022564 003026 BGT 4$ :: IS AN OCTAL DIGIT
(1) 022565 122716 000067 CMPB #'7,(SP)
(1) 022566 002423 BLT 4$
(1) 022567 006301 ASL R1 :: *2
(1) 022568 006102 ROL R2
(1) 022569 006301 ASL R1 :: *4
(1) 022570 006102 ROL R2
(1) 022571 006301 ASL R1 :: *8
(1) 022572 006102 ROL R2
(1) 022573 042716 177770 BIC #'C7,(SP) :: STRIP THE ASCII JUNK
(1) 022574 062601 ADD (SP)+,R1 :: ADD IN THIS DIGIT
(1) 022575 000756 BR 2$ :: LOOP
(1) 022576 005726 35: TST (SP)+ :: CLEAN TERMINATOR FROM STACK
(1) 022577 010166 000012 MOV R1,12(SP) :: SAVE THE RESULT
(1) 022578 010237 022706 MOV R2,$HIOCT
(1) 022579 012602 MOV (SP)+,R2 :: POP STACK INTO R2
(1) 022580 012601 MOV (SP)+,R1 :: POP STACK INTO R1
(1) 022581 012600 MOV (SP)+,RO :: POP STACK INTO RO
(1) 022582 000002 RTI :: RETURN
(1) 022583 005726 45: TST (SP)+ :: CLEAN PARTIAL FROM STACK
(1) 022584 105010 CLRB (RO) :: SET A TERMINATOR
(1) 022585 104400 TYPE :: TYPE UP THRU THE BAD CHAR.
(1) 022586 000000 55: .WORD 0
(1) 022587 104400 001164 TYPE $QUES
(1) 022588 000730 BR 1$ :: "?? "CR" & "LF"
(1) 022589 000000 $HIOCT: .WORD 0 :: TRY AGAIN
(1) 022590 105777 156164 $SCOPE: TSTB $SWR :: HIGH ORDER BITS GO HERE
(1) 022591 100003 BPL $SCOPE :: TEST BIT ?
(1) 022592 005737 010742 TST LOOP :: NOT SET
(1) 022593 001041 BNE $OVER :: TEST LOOP

```

:849

.SBTTL SCOPE HANDLER ROUTINE

.*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
.*AND LOAD THE TEST NUMBER(\$TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
.*AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>
.*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
.*SW14=1 LOOP ON TEST
.*SW09=1 LOOP ON TEST IN SWR<7:0>
.*CALL
.* SCOPE ::SCOPE=IOT

(1) 022724 032777 040000 156146
(1) 022724 001035
(1) 022732
(1) 022734 000416
(1) 022736 013746 000004
(1) 022742 012737 022762 000004
(1) 022750 005737 177060
(1) 022754 012637 000004
(1) 022760 000414
(1) 022762 022626
(1) 022764 012637 000004
(1) 022770 000416
(1) 022772
(1) 022772 032777 000400 156100
(1) 023000 001404
(1) 023002 127737 156072 001106
(1) 023010 001406
(1) 023012 105237 001106
(1) 023016 011637 001112
(1) 023022 105037 001107
(1) 023026 013777 001106 156046
(1) 023034 013716 001112
(1) 023040 000002

\$SCOPE:
15: BIT #BIT14,\$SWR ::LOOP ON PRESENT TEST?
BNE \$OVER ::YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
\$XSTR: BR 65 ::IF RUNNING ON THE "XOR" TESTER CHANGE
::THIS INSTRUCTION TO A "NOP" (NOP=240)
MOV \$ERRVEC, -(SP) ::SAVE THE CONTENTS OF THE ERROR VECTOR
MOV \$5, \$ERRVEC ::SET FOR TIMEOUT
TST \$177060 ::TIME OUT ON XOR?
MOV (SP)+, \$ERRVEC ::RESTORE THE ERROR VECTOR
BR \$SVLAD ::GO TO THE NEXT TEST
55: CMP (SP)+, (SP)+ ::CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+, \$ERRVEC ::RESTORE THE ERROR VECTOR
BR \$OVER ::LOOP ON THE PRESENT TEST
65: *****END OF CODE FOR THE XOR TESTER*****
BIT #BIT08,\$SWR ::LOOP ON SPEC. TEST?
BEQ \$SVLAD ::BR IF NO
CMPB \$SWR,\$TSTNM ::ON THE RIGHT TEST? SWR 7:0
BEQ \$OVER ::BR IF YES
\$SVLAD: INCB \$TSTNM ::COUNT TEST NUMBERS
MOV (SP), \$LPADR ::SAVE SCOPE LOOP ADDRESS
CLRB \$ERFLG ::ZERO THE ERROR FLAG
\$OVER: MOV \$TSTNM,\$DISPLAY ::DISPLAY TEST NUMBER
MOV \$LPADR,(SP) ::FUDGE RETURN ADDRESS
RTI ::FIXES PS

:849
:850

.SBTTL ERROR HANDLER ROUTINE

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

(1) 023042
(2) 023042 113737 001106 001250
(1) 023050 105237 001107
(1) 023054 001775

\$ERROR:
75: MOVB \$TSTNM,\$TSTNUM
INCB \$ERFLG ::SET THE ERROR FLAG
BEQ 75 ::DON'T LET THE FLAG GO TO ZERO

```

(1) 023056 013777 001106 156016
(1) 023064 005237 001116
(1) 023070 011637 001122
(1) 023074 162737 000002 001122
(1) 023102 117737 156014 001120
(1) 023110 032777 020000 155762
(1) 023116 001004
(1) 023120 004737 023154
(1) 023124 104400 001165
(1) 023130
(1) 023130 005777 155744
(1) 023134 100006
(1) 023136 000000
(1) 023140 022737 006726 000042
(1) 023146 001001
(1) 023150 000000
(1) 023152 000002

```

```

MOV $STNM, $DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
INC $ERTTL ;; INC THE ERROR COUNT
MOV (SP), $ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
SUB #2, $ERRPC
MOVB $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13, $SWR ;; SKIP TYPEOUT IF SET
SNE 20$ ;; SKIP TYPEOUTS
JSR PC, $ERRTYP ;; GO TO USER ERROR ROUTINE
TYPE $SCRLF

20$:
TST $SWR ;; HALT ON ERROR
BPL 3$ ;; SKIP IF CONTINUE
HALT ;; HALT ON ERROR!
CMP #SENDAD, $#42 ;; ACT-11 AUTO-ACCEPT?
BNE 3$ ;; BRANCH IF NO
HALT ;; YES

3$:
RTI ;; RETURN

```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

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

```

```

(1) 023154
(1) 023154 104400 001165
(1) 023160 010046
(1) 023162 005000
(1) 023164 153700 001120
(1) 023170 001004
(2) 023172 013746 001122
(2) 023176 104401
(1) 023200 000426
(1) 023202 005300
(1) 023204 006300
(1) 023206 006300
(1) 023210 006300
(1) 023212 062700 001170
(1) 023216 012037 023226
(1) 023222 001404
(1) 023224 104400
(1) 023226 000000
(1) 023230 104400 001165
(1) 023234 012037 023244
(1) 023240 001404
(1) 023242 104400
(1) 023244 000000
(1) 023246 104400 001165
(1) 023252 011000
(1) 023254 001004
(1) 023256 012600

```

```

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

```

```

(1) 023260 104400 001165          TYPE    $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
(1) 023264 000207          RTS      0C        ;; RETURN
(2) 023266 013046          7$:      MOV      2(RO)+, -(SP) ;; SAVE 2(RO)+ FOR TYPEOUT
(2) 023270 104401          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 023272 005710          TST      (RO)      ;; IS THERE ANOTHER NUMBER?
(1) 023274 001770          SEQ      6$        ;; BR IF NO
(1) 023276 104400 023304          TYPE    8$        ;; TYPE TWO(2) SPACES
(1) 023302 000771          BR      7$        ;; LOOP
(1) 023304 020040 000          8$:      .ASCIZ  / /      ;; TWO(2) SPACES
(1) 023310 023310          .EVEN

*****

(1) 023310 023310          .SBTTL  TYPE ROUTINE

(1) 023310 023310          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1) 023310 023310          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1) 023310 023310          ;*NOTE1:           $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1) 023310 023310          ;*NOTE2:           $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1) 023310 023310          ;*NOTE3:           $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1) 023310 023310          ;*
(1) 023310 023310          ;*CALL:
(1) 023310 023310          ;*1) USING A TRAP INSTRUCTION
(1) 023310 023310          ;*   TYPE    ,MESADR      ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1) 023310 023310          ;*OR
(1) 023310 023310          ;*   TYPE    MESADR
(1) 023310 023310          ;*
(1) 023310 023310          ;*2) USING A JSR INSTRUCTION
(1) 023310 023310          ;*   MOV     PS, -(SP)    ;; PUSH PROCESSOR STATUS WORD ON THE STACK
(1) 023310 023310          ;*   JSR    PC, $TYPE    ;; CALL TYPE ROUTINE
(1) 023310 023310          ;*   MESADDR      ;; FIRST ADDRESS OF MESSAGE

(1) 023310 105737 001155          $TYPE:  TSTB   $TPFLG    ;; IS THERE A TERMINAL?
(1) 023314 100002          BPL     1$          ;; BR IF YES
(1) 023316 000000          HALT                    ;; HALT HERE IF NO TERMINAL
(1) 023320 000407          BR      3$          ;; LEAVE
(1) 023322 010046          1$:      MOV     RO, -(SP)    ;; SAVE RO
(1) 023324 017600 000002          MOV     22(SP), RO    ;; GET ADDRESS OF ASCIZ STRING
(1) 023330 112046          2$:      MOVVB  (RO)+, -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 023332 001005          BNE     4$          ;; BR IF IT ISN'T THE TERMINATOR
(1) 023334 005726          TST     (SP)+        ;; IF TERMINATOR POP IT OFF THE STACK
(1) 023336 012600          60$:     MOV     (SP)+, RO  ;; RESTORE RO
(1) 023340 062716 000002          3$:      ADD     #2, (SP)     ;; ADJUST RETURN PC
(1) 023344 000002          RTI                    ;; RETURN
(1) 023346 122716 000011          4$:      CMPB   #THT, (SP)   ;; BRANCH IF <HT>
(1) 023352 001431          BEQ     8$          ;; BRANCH IF NOT <CRLF>
(1) 023354 122716 000200          CMPB   #TCRLF, (SP)
(1) 023360 001007          BNE     5$          ;; POP <CR><LF> EQUIV
(1) 023362 005726          TST     (SP)+        ;; TYPE CR AND LF
(1) 023364 013746 177776          MOV     PS, -(SP)
(1) 023370 004737 023310          JSR    PC, $TYPE
(1) 023374 001165          $CRLF
(1) 023376 000754          BR      2$          ;; GET NEXT CHARACTER
(1) 023400 004737 023462          5$:      JSR    PC, $TYPEC   ;; GO TYPE THIS CHARACTER
    
```

```

(1) 023404 123726 001154 6S:  CMPB  $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
(1) 023410 001347  BNE  2S  ;; IF NO GO GET NEXT CHAR.
(1) 023412 013746 001152  MOV  $NULL,-(SP)  ;; GET # OF FILLER CHARS. NEEDED
(1)  AND  THE NULL CHAR.
(1) 023416 105366 000001 7S:  DECB  1(SP)  ;; DOES A NULL NEED TO BE TYPED?
(1) 023422 002770  BLT  6S  ;; BR IF NO--GO POP THE NULL OFF OF STACK
(1) 023424 004737 023462  JSR  PC,$TYPEC  ;; GO TYPE A NULL
(1) 023430 105337 023526  DECB  $CHARCNT  ;; DO NOT COUNT AS A COUNT
(1) 023434 000770  BR  7S  ;; LOOP

```

:HORIZONTAL TAB PROCESSOR

```

(1) 023436 112716 000040 8S:  MOVB  #40,(SP)  ;; REPLACE TAB WITH SPACE
(1) 023442 004737 023462 9S:  JSR  PC,$TYPEC  ;; TYPE A SPACE
(1) 023446 132737 000007 023526  BITB  #7,$CHARCNT  ;; BRANCH IF NOT AT
(1) 023454 001372  BNE  9S  ;; TAB STOP
(1) 023456 005726  TST  (SP)+  ;; POP SPACE OFF STACK
(1) 023460 000723  BR  2S  ;; GET NEXT CHARACTER
(1) 023462 105777 155460 $TYPEC: TSTB  @STPS  ;; WAIT UNTIL PRINTER IS READY
(1) 023466 100375  BPL  $TYPEC
(1) 023470 116677 000002 155452  MOVB  2(SP),@STPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 023476 122766 000015 000002  CMPB  #15,2(SP)  ;; BRANCH IF
(1) 023504 001003  BNE  1S  ;; NOT <CR>
(1) 023506 105037 023526  CLRB  $CHARCNT
(1) 023512 000406  BR  $TYPEX
(1) 023514 122766 000012 000002 1S:  CMPB  #12,2(SP)  ;; EXIT
(1) 023522 002002  BGE  $TYPEX  ;; BRANCH IF
(1) 023524 105227  INCB  (PC)+  ;; <LF>
(1) 023526 000000  $CHARCNT: .WORD 0  ;; INC SPACE
(1) 023530 000207  $TYPEX: RTS  PC  ;; COUNT
(1)  EQUATES
(1)  THT=11
(1)  TCRLF=200
(1) 000011
(1) 000200

```

1855
1856

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE

```

```

*CALL:
*   MOV    NUM,-(SP)  ;; NUMBER TO BE TYPED
*   TYPOS  ;; CALL FOR TYPEOUT
*   .BYTE  N          ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE  M          ;; M=1 OR 0
*   ;; I=TYPE LEADING ZEROS
*   ;; O=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

```

```

(1) ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1) ;*CALL:
(1) ;* MOV NUM,-(SP) ;:NUMBER TO BE TYPED
(1) ;* TYPOC ;:CALL FOR TYPEOUT
(1) 023532 017646 000000 $TYPOS: MOV 2(SP),-(SP) ;:PICKUP THE MODE
(1) 023536 116637 000001 023755 MOV 1(SP),$OFILL ;:LOAD ZERO FILL SWITCH
(1) 023544 112637 023757 MOV 2(SP)+,$OMODE+1 ;:NUMBER OF DIGITS TO TYPE
(1) 023550 062716 000002 ADD #2,(SP) ;:ADJUST RETURN ADDRESS
(1) 023554 000406 BR $TYPON
(1) 023556 112737 000001 023755 $TYPOC: MOV #1,$OFILL ;:SET THE ZERO FILL SWITCH
(1) 023564 112737 000006 023757 MOV #6,$OMODE+1 ;:SET FOR SIX(6) DIGITS
(1) 023572 112737 000005 023754 $TYPON: MOV #5,$OCNT ;:SET THE ITERATION COUNT
(1) 023600 010346 MOV R3,-(SP) ;:SAVE R3
(1) 023602 010446 MOV R4,-(SP) ;:SAVE R4
(1) 023604 010546 MOV R5,-(SP) ;:SAVE R5
(1) 023606 113704 023757 MOV $OMODE+1,R4 ;:GET THE NUMBER OF DIGITS TO TYPE
(1) 023612 005404 NEG R4
(1) 023614 062704 000006 ADD #6,R4 ;:SUBTRACT IT FOR MAX. ALLOWED
(1) 023620 110437 023756 MOV R4,$CMODE ;:SAVE IT FOR USE
(1) 023624 113704 023755 MOV $OFILL,R4 ;:GET THE ZERO FILL SWITCH
(1) 023630 016605 000012 MOV 12(SP),R5 ;:PICKUP THE INPUT NUMBER
(1) 023634 005003 CLR R3 ;:CLEAR THE OUTPUT WORD
(1) 023636 006105 1$: ROL R5 ;:ROTATE MSB INTO "C"
(1) 023640 000404 BR 3$ ;:GO DO MSB
(1) 023642 006105 2$: ROL R5 ;:FORM THIS DIGIT
(1) 023644 006105 ROL R5
(1) 023646 006105 ROL R5
(1) 023650 010503 MOV R5,R3
(1) 023652 006103 3$: ROL R3 ;:GET LSB OF THIS DIGIT
(1) 023654 105337 023756 DECB $OMODE ;:TYPE THIS DIGIT?
(1) 023660 100016 BPL 7$ ;:BR IF NO
(1) 023662 042703 177770 BIC #177770,R3 ;:GET RID OF JUNK
(1) 023666 001002 BNE 4$ ;:TEST FOR 0
(1) 023670 005704 TST R4 ;:SUPPRESS THIS 0?
(1) 023672 001403 BEQ 5$ ;:BR IF YES
(1) 023674 005204 4$: INC R4 ;:DON'T SUPPRESS ANYMORE 0'S
(1) 023676 052703 000060 BIS #'0,R3 ;:MAKE THIS DIGIT ASCII
(1) 023702 052703 000040 5$: BIS #' ,R3 ;:MAKE ASCII IF NOT ALREADY
(1) 023706 110337 023752 MOV R3,$S ;:SAVE FOR TYPING
(1) 023712 104400 023752 TYPE #8$ ;:GO TYPE THIS DIGIT
(1) 023716 105337 023754 7$: DECB $OCNT ;:COUNT BY 1
(1) 023722 003347 BGT 2$ ;:BR IF MORE TO DO
(1) 023724 002402 BLT 6$ ;:BR IF DONE
(1) 023726 005204 INC R4 ;:INSURE LAST DIGIT ISN'T A BLANK
(1) 023730 000744 BR 2$ ;:GO DO THE LAST DIGIT
(1) 023732 012605 6$: MOV (SP)+,R5 ;:RESTORE R5
(1) 023734 012604 MOV (SP)+,R4 ;:RESTORE R4
(1) 023736 012603 MOV (SP)+,R3 ;:RESTORE R3
(1) 023740 016666 000002 000004 MOV 2(SP),4(SP) ;:SET THE STACK FOR RETURNING
(1) 023746 012616 MOV (SP)+,(SP)
(1) 023750 000002 RTI ;:RETURN
(1) 023752 000 8$: .BYTE 0 ;:STORAGE FOR ASCII DIGIT
(1) 023753 000 .BYTE 0 ;:TERMINATOR FOR TYPE ROUTINE
(1) 023754 000 $OCNT: .BYTE 0 ;:OCTAL DIGIT COUNTER
(1) 023755 000 $OFILL: .BYTE 0 ;:ZERO FILL SWITCH

```



```

(1) 023756 000000
1857 (1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(2) 023760
(3) 023760 010046
(3) 023762 010146
(3) 023764 010246
(3) 023766 010346
(1) 023770 013700 024106
(1) 023774 013701 024104
(1) 024000 012703 177771
(1) 024004 005002
(1) 024006 006300
(1) 024010 006101
(1) 024012 006102
(1) 024014 005203
(1) 024016 001373
(1) 024020 063700 024106
(1) 024024 005501
(1) 024026 063701 024104
(1) 024032 005502
(1) 024034 062700 001057
(1) 024040 005501
(1) 024042 005502
(1) 024044 062701 047401
(1) 024050 005502
(1) 024052 062702 000006
(1) 024056 060200
(1) 024060 005501
(1) 024062 010037 024106
(1) 024066 010137 024104
(3) 024072 012603
(3) 024074 012602
(3) 024076 012601
(3) 024100 012600
(1) 024102 000207
(1) 024104 176543
(1) 024106 123456
1858

```

```

$CMODE: .WORD 0 ;:NUMBER OF DIGITS TO TYPE
;*****
.SBTTL RANDOM NUMBER GENERATOR ROUTINE
;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
;*WITH A RANGE OF 0 TO 2(+33)-1.
;*CALL:
;* JSR PC,$RAND ;:CALL THE ROUTINE
;* RETURN ;:RETURN HERE THE RANDOM
;* ;:NUMBER WILL BE IN
;* ;:$HINUM,$LONUM

$RAND:
MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV $LONUM,R0 ;:SET R0 WITH LOW
MOV $HINUM,R1 ;:SET R1 WITH HIGH
MOV #-7,R3 ;:SET SHIFT COUNT
CLR R2 ;:ZERO R2
1$: ASL R0 ;:SHIFT R0 LEFT AND
ROL R1 ;:ROTATE CARRY INTO R1 AND
ROL R2 ;:ROTATE CARRY INTO R2
INC R3 ;:CHECK FOR DONE
BNE 1$ ;:CONTINUE SHIFT LOOP
ADD $LONUM,R0 ;:ADD NUMBER TO MAKE X 129
ADC R1 ;:PROPOGATE CARRY
ADD $HINUM,R1 ;:ADD NUMBER TO MAKE X 129
ADC R2 ;:PROPOGATE CARRY
ADD #1057,R0 ;:ADD LOW CONSTANT
ADC R1 ;:PROPOGATE CARRY
ADC R2 ;:PROPOGATE CARRY
ADD #47401,R1 ;:ADD HIGH CONSTANT
ADC R2 ;:PROPOGATE CARRY
ADD #6,R2 ;:ADD HIGHEST CONSTART
ADD R2,R0 ;:REPRIME R0 WITH HIGHEST DIGIT
ADC R1 ;:PROPOGATE CARRY
MOV R0,$LONUM ;:SAVE R0
MOV R1,$HINUM ;:SAVE R1
MOV (SP)+,R3 ;:POP STACK INTO R3
MOV (SP)+,R2 ;:POP STACK INTO R2
MOV (SP)+,R1 ;:POP STACK INTO R1
MOV (SP)+,R0 ;:POP STACK INTO R0
RTS PC ;:RETURN

$HINUM: .WORD 176543
$LONUM: .WORD 123456
;*****

```

```

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

```

```

(1) (1) 024110 010046 $TRAP: MOV RO, -(SP) ;; SAVE RO
(1) (1) 024112 016600 000002 MOV 2(SP), RO ;; GET TRAP ADDRESS
(1) (1) 024116 005740 TST -(RO) ;; BACKUP BY 2
(1) (1) 024120 111000 MOV B (RO), RO ;; GET RIGHT BYTE OF TRAP
(1) (1) 024122 006300 ASL RO ;; POSITION FOR INDEXING
(1) (1) 024124 016000 024132 MOV $TRPAD(RO), RO ;; INDEX TO TABLE
(1) (1) 024130 000200 RTS RO ;; GO TO ROUTINE

```

.SBTTL TRAP TABLE

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

ROUTINE

```

(1) (3) 024132 $TRPAD:
(2) (3) 024132 023310 $TYPE ;; CALL=TYPE TRAP+0(104400) TTY TYPEOUT ROUTINE
(3) (3) 024134 023556 $TYPOC ;; CALL=TYPOC TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3) (3) 024136 023532 $TYPOS ;; CALL=TYPOS TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3) (3) 024140 023572 $TYPON ;; CALL=TYPON TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
(3) (3) 024142 012250 $TYPDS ;; CALL=TYPDS TRAP+4(104404) TYPE DECIMAL NUMBER (WITH SIGN)
(3) (3) 024144 022376 $RDCHR ;; CALL=RDCHR TRAP+5(104405) TTY TYPEIN CHARACTER ROUTINE
(3) (3) 024146 022432 $RDLIN ;; CALL=RDLIN TRAP+6(104406) TTY TYPEIN STRING ROUTINE
(3) (3) 024150 022550 $RDOCT ;; CALL=RDOCT TRAP+7(104407) READ AN OCTAL NUMBER FROM TTY

```

.SBTTL POWER DOWN AND UP ROUTINES

:POWER DOWN ROUTINE

```

(1) (1) 024152 012737 024274 000024 $PWRDN: MOV $SILLUP, 2#PWRVEC ;; SET FOR FAST UP
(1) (1) 024160 012737 000340 000026 MOV #340, 2#PWRVEC+2 ;; PWRUP:7
(3) (3) 024166 010046 MOV RO, -(SP) ;; PUSH RO ON STACK
(3) (3) 024170 010146 MOV R1, -(SP) ;; PUSH R1 ON STACK
(3) (3) 024172 010246 MOV R2, -(SP) ;; PUSH R2 ON STACK
(3) (3) 024174 010346 MOV R3, -(SP) ;; PUSH R3 ON STACK
(3) (3) 024176 010446 MOV R4, -(SP) ;; PUSH R4 ON STACK
(3) (3) 024200 010546 MOV R5, -(SP) ;; PUSH R5 ON STACK
(1) (1) 024202 010637 024300 MOV SP, $SAVR6 ;; SAVE SP
(1) (1) 024206 012737 024220 000024 MOV #PWRUP, 2#PWRVEC ;; SET UP VECTOR
(1) (1) 024214 000000 HALT
(1) (1) 024216 000776 BR -2 ;; HANG UP

```

:POWER UP ROUTINE

```

(1) (1) 024220 013706 024300 $PWRUP: MOV $SAVR6, SP ;; GET SP
(1) (1) 024224 005037 024300 CLR $SAVR6 ;; WAIT LOOP FOR THE TTY
(1) (1) 024230 005237 024300 1$: INC $SAVR6 ;; WAIT FOR THE INC
(1) (1) 024234 001375 BNE 1$ ;; OF WORD
(3) (3) 024236 012605 MOV (SP)+, R5 ;; POP STACK INTO R5
(3) (3) 024240 012604 MOV (SP)+, R4 ;; POP STACK INTO R4
(3) (3) 024242 012603 MOV (SP)+, R3 ;; POP STACK INTO R3
(3) (3) 024244 012602 MOV (SP)+, R2 ;; POP STACK INTO R2
(3) (3) 024246 012601 MOV (SP)+, R1 ;; POP STACK INTO R1
(3) (3) 024250 012600 MOV (SP)+, R0 ;; POP STACK INTO R0
(1) (1) 024252 012737 024152 000024 MOV #PWRDN, 2#PWRVEC ;; SET UP THE POWER DOWN VECTOR

```

```

(1) 024260 012737 000340 000026      MOV      #340,2#PWRVEC+2 ;;PRIO:7
(1) 024266 104400                      TYPE                      ;;REPORT THE POWER FAILURE
(1) 024270 024302      $PWRMG: .WORD $POWER      ;;POWER FAIL MESSAGE POINTER
(1) 024272 000002                      RTI
(1) 024274 000000      $ILLUP: HALT                ;; THE POWER UP SEQUENCE WAS STARTED
(1) 024276 000776                      BR      .-2                ;; BEFORE THE POWER DOWN WAS COMPLETE
(1) 024300 000000      $$AVR6: 0                  ;; PUT THE SP HERE
(1) 024302 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
(1) 024310 000122
(1)
1860 024312 000000      BUFFER: 0
1861          000001          .END

```


HOMERS	016121	730	1727*								
IGNORE	010760	113*	1026*	1285	1319*	1322	1340*	1351*			
IOTVEC=	000020	12*	112*	115*							
KRBDON	010156	1154	1178	1208*							
KRBECH	010172	105	1212*								
KRBECD	007022	109	996*								
KRBTST	007316	107	1063*	1210							
LAST	001242	56*	129*	133*	134	983					
LASTCH	001306	80*	212*	215*	306	328	820	845	911	1248	
LASTLN	001260	69*	201*	206*	640						
LBMT	002706	338*									
LBMT1	003056	356	371*								
LCM	003066	376*									
LIC	010262	318	835	906	1242*						
LICA	010300	1246*	1252								
LICB	010316	1249	1251*								
LODERL	003730	522*									
LODERS	004106	562*									
LOOP	010742	114*	1315*	1333*	1344*	1845					
LRL	003350	432*									
LTAB	003620	448	452	456	460	464	468	472	495*		
MANFU	001376	20	101*								
MASK1	012244	1541*	1545*	1553	1613*						
MASK2	012246	1542*	1550*	1552	1573*	1577*	1594	1614*			
MKB	014005	1070	1682*								
MKBA	014074	1537	1684*								
MKBB	014163	1078	1117	1133	1148	1685*					
MKBB2	014300	1075	1114	1130	1145	1687*					
MKBC	014363	1083	1688*								
MKBD	014422	1093	1689*								
MKBD2	014527	1090	1691*								
MKBE	014615	1098	1692*								
MKBF	014735	1103	1694*								
MKBG	014773	1109	1695*								
MKBGA	015042	1125	1696*								
MKBH	015112	1140	1697*								
MKBI	015153	1159	1185	1698*							
MKBJ	015206	1163	1189	1699*							
MKBK	015241	1167	1193	1700*							
MKBL	015275	1171	1197	1701*							
MKBM	015330	1175	1201	1702*							
MKBN	015364	1156	1703*								
MKBQ	015457	1607	1705*								
MKBQA	015524	1597*	1598*	1599*	1708*						
MKBQB	015543	1585*	1586*	1587*	1711*						
MKBQ1	015520	1601*	1707*								
MKBQ2	015537	1592*	1710*								
MKBR	015552	1209	1712*								
MKB1	014041	1120	1136	1683*							
MKB52	015413	1180	1704*								
MKE	015620	998	1714*								
MKEA	016025	1028	1720*								
MKEA1	016036	1015*	1016*	1021*	1022*	1024*	1025*	1721*			
MKEB	016043	1002*	1003*	1004*	1722*						
MKEH	016052	1214	1723*								
MOVDN1	003022	342	359*	362							

MOVDWN	003016	349	358#											
MOVRIG	003040	343	345	346	350	352	353	365#						
MPTCNT	013644	901	1679#											
MPTSCN	013713	921	1680#											
MQ0	017075	1316	1748#											
MQ1	017162	1334	1358	1749#										
MQ2	017250	1355	1750#											
MSCOPE	022710	115	1843#											
MSGTND	005010	629	678#											
MSGTXT	004670	627	676#											
MSGTYP	002350	194	251#											
MTEXT0	021352	885	952	1829#										
MTEXT1	021456	953	1830#											
MTEXT2	021557	954	1831#											
MTEXT3	021661	955	1832#											
MTEXT4	021744	956	1834#											
MTEXT5	022046	957	1835#											
MTEXT6	022147	958	1836#											
MTEXT7	022201	959	1837#											
MTEXT8	022301	960	1838#											
MT08	012614	1502	1653#											
M91	012731	266	778	1664#										
M910	013222	514	1670#											
M911	013253	555	1671#											
M912	013306	585	1672#											
M914	013342	156	1673#											
M92	012777	274	789	1665#										
M920	013407	856	1674#											
M921	013437	879	1675#											
M922	013473	693	1676#											
M9221	013530	732	1677#											
M923	013605	759	1678#											
M923A	013750	934	1681#											
M93	013026	292	807	1666#										
M94	013070	313	831	1667#										
M96	013121	334	1668#											
M97	013154	430	1669#											
M98	017427	607	615	1756#										
NOWEOP	006762	983	984#											
OCTAL	011554	1001	1510#	1584	1596									
OVRAL	004666	624*	668*	675#										
PASHED	017617	983	1759#											
PATH	016127	588	1728#											
PC	=%000007	12#	160*	163*	166*	174*	221*	268*	270*	285*	289*	297*	299*	303*
		309*	321*	325*	331*	342*	343*	345*	346*	349*	350*	352*	353*	363*
		369*	371*	372*	419*	420*	446*	450*	454*	458*	462*	466*	470*	474*
		481*	489*	490*	516*	518*	549*	551*	556*	558*	580*	581*	592*	610*
		611*	618*	619*	667*	670*	722*	736*	737*	745*	765*	767*	768*	774*
		779*	784*	797*	803*	811*	812*	827*	837*	852*	858*	859*	860*	863*
		864*	868*	869*	870*	874*	893*	909*	915*	923*	929*	940*	942*	943*
		949*	983*	999*	1001*	1079*	1084*	1094*	1099*	1104*	1118*	1134*	1149*	1160*
		1164*	1168*	1172*	1176*	1186*	1190*	1194*	1198*	1202*	1216*	1237*	1342*	1359*
		1422*	1453*	1478*	1494*	1504*	1528*	1539*	1558*	1584*	1596*	1621*	1624*	1627*
		1639*	1646*	1850*	1852*	1854*	1857*							

PIRQ = 177772
PIRQVE= 000240

SW12	=	010000	12*
SW13	=	020000	12*
SW14	=	040000	12*
SW15	=	100000	12*
SW2	=	000004	12*
SW3	=	000010	12*
SW4	=	000020	12*
SW5	=	000040	12*
SW6	=	000100	12*
SW7	=	000200	12*
SW8	=	000400	12*
SW9	=	001000	12*
BITVE	=	000014	12*
CRIT	=	000200	1954*
TEMP	=	001310	81*
TEMP0	=	001312	82*
			815*
TH	=	000011	1954*
TIME0	=	001252	61*
TIME1	=	012610	1276*
TIME2	=	012612	1277*
TITLE	=	012630	119*
KVEC	=	000060	12*
TOTALC	=	001262	70*
TPVEC	=	000064	12*
TRAPVE	=	000034	12*
TRTVEC	=	000014	12*
TSTNUM	=	001250	59*
TSTROW	=	011652	1079*
TST1	=	001760	94*
TST10	=	003710	491*
TST11	=	004066	553*
TST12	=	004214	583*
TST13	=	004252	594*
TST14	=	005012	603*
TST15	=	005162	689*
TST16	=	005274	746*
TST17	=	005444	776*
TST2	=	002400	222*
TST20	=	005500	787*
TST21	=	005570	805*
TST22	=	005722	829*
TST23	=	006056	854*
TST24	=	006200	876*
TST25	=	006256	896*
TST26	=	006370	917*
TST27	=	006424	931*
TST3	=	002420	272*
TST30	=	007020	995*
TST31	=	007314	1062*
TST32	=	010170	1211*
TST4	=	002474	290*
TST5	=	002574	311*
TST6	=	002676	332*
TST7	=	003340	424*
			275*
			286*
			586*
			590*
			790*
			798*
			861*
			862*
			865*
			769*
			809*
			813*
			295*
			301*
			304*
			316*
			323*
			326*
			735*
			741*
			763*
			833*
			839*
			840*
			938*
			944*
			1276
			1632
			1632*
			1643*
			1646*
			1282*
			1280*
			1633*
			1641*
			1649*
			1663*
			199*
			204*
			624
			112*
			176*
			1771
			1850*
			1039
			1104
			1118
			1134
			1149
			1160
			1164
			1169
			1172
			1176
			1084
			1094
			1099
			1194
			1198
			1202
			1536*
			983
			987
			1379
			1386
			1387
			1388
			1389
			680*
			1390
			691
			724*
			1391
			1392
			1393
			263*
			1380
			1394
			1395
			1396
			1397
			1398
			1399
			1400
			1401
			1381
			1402
			1403
			1382
			1383
			1384
			428*
			1385

.STYPE	7#	8#	1854
.STYPO	7#	1856	

IOT	12															
JMP	15	16	17	18	19	20	148	603	727	749	899	951	983	991	1210	
JSR	1320	1329	1335	1375												
	155	157	160	163	166	174	196	221	265	268	270	273	277	281	285	
	229	291	297	299	303	309	312	318	321	325	331	333	342	343	345	
	346	349	350	352	353	371	372	419	420	421	425	429	446	448	450	
	452	454	456	458	460	462	464	466	468	470	472	474	481	489	490	
	513	516	518	549	551	554	556	558	580	581	584	587	592	606	610	
	611	614	618	619	635	650	667	682	685	692	697	710	713	722	729	
	731	733	736	737	738	743	745	758	760	765	767	768	772	774	777	
	779	782	784	788	791	794	797	801	803	806	811	812	817	822	825	
	827	830	835	837	842	847	850	852	855	858	859	860	863	864	868	
	869	870	872	874	878	884	891	893	900	902	906	909	913	915	920	
	923	926	929	933	935	940	942	943	947	949	983	997	999	1001	1027	
	1069	1074	1077	1079	1082	1084	1089	1092	1094	1097	1099	1102	1104	1108	1113	
	1116	1118	1119	1124	1129	1132	1134	1135	1139	1144	1147	1149	1155	1158	1160	
	1162	1164	1166	1168	1170	1172	1174	1176	1179	1181	1184	1186	1188	1190	1192	
	1194	1196	1198	1200	1202	1203	1208	1213	1216	1354	1357	1359	1502	1504	1536	
	1539	1584	1596	1606	1621	1624	1850	1854								
MOV	94	97	98	103	105	107	109	110	112	115	125	133	135	137	139	
	140	141	143	145	147	162	165	168	193	194	199	200	201	208	210	
	212	275	293	295	296	304	314	316	317	326	338	339	340	358	365	
	376	377	383	396	409	432	434	435	477	480	482	495	496	497	500	
	522	525	531	562	564	568	586	622	623	624	625	626	627	628	633	
	636	642	643	644	651	657	660	695	696	709	728	735	756	763	764	
	766	771	781	790	800	808	809	810	815	816	824	832	833	834	840	
	841	849	857	861	867	871	882	890	904	905	908	925	938	939	941	
	946	983	987	988	996	1008	1065	1067	1068	1071	1081	1086	1096	1101	1110	
	1126	1141	1157	1161	1165	1169	1173	1183	1187	1191	1195	1199	1212	1229	1230	
	1242	1243	1250	1258	1276	1287	1298	1302	1306	1309	1315	1316	1334	1356	1374	
	1406	1412	1426	1427	1428	1429	1442	1443	1444	1449	1457	1483	1501	1510	1514	
	1521	1541	1551	1563	1583	1593	1595	1616	1620	1622	1623	1632	1637	1653	1654	
	1841	1842	1848	1850	1852	1854	1856	1857	1858	1859						
MOV8	344	347	348	351	354	355	359	360	366	378	379	382	386	387	391	
	394	395	399	400	404	407	408	410	411	414	415	416	417	418	433	
	436	437	442	443	444	483	484	485	488	498	499	502	505	508	509	
	510	523	524	527	528	538	539	542	546	565	569	570	573	574	576	
	608	609	616	617	663	664	665	1002	1003	1004	1015	1016	1021	1022	1024	
	1025	1218	1231	1232	1233	1236	1244	1245	1246	1253	1264	1271	1368	1407	1408	
	1409	1410	1411	1415	1418	1419	1420	1421	1431	1432	1433	1434	1437	1445	1452	
	1513	1520	1527	1585	1586	1587	1591	1592	1597	1598	1599	1601	1616	1655	1841	
	1842	1850	1854	1856	1858											
	1841	1856														
NEG	1616	1856														
NOP	220	535	536	537	894	983	1325	1326	1327	1328	1372	1373	1464	1465	1466	
	1467	1468	1475	1476	1477	1491	1492	1493								
RESET	111	983														
ROL	1522	1523	1842	1856	1857											
ROR	1515	1516	1517	1564	1571	1575										
RTI	1616	1841	1842	1848	1850	1854	1856	1859								
RTS	363	369	511	670	1237	1254	1342	1422	1438	1453	1478	1494	1505	1528	1558	
	1627	1639	1646	1657	1852	1854	1857	1858								
	645	658	1413	1616	1850											
SUB	1524															
SWAB	1858															
TRAP																
TST	116	128	134	142	144	146	172	185	202	423	532	604	688	690	700	
	716	886	983	984	1011	1261	1274	1285	1296	1299	1322	1370	1459	1556	1616	

TSTB	1842	1845	1848	1850	1852	1854	1856	1858							
.ASCII	120	186	661	754	1219	1259	1272	1278	1304	1567	1616	1635	1841	1843	1854
	21	676	677	1052	1053	1054	1055	1056	1057	1058	1059	1682	1685	1689	1692
	1695	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1709	1714	1716
.ASCIZ	1720	1728	1729	1730	1731	1733	1735	1737	1739	1741	1743	1745	1838		
	21	983	1663	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675
	1676	1677	1678	1679	1680	1681	1683	1684	1686	1687	1688	1690	1691	1693	1694
	1712	1718	1723	1748	1749	1750	1752	1753	1756	1757	1758	1759	1760	1761	1762
.BLKB	1763	1764	1765	1766	1767	1829	1830	1831	1832	1834	1835	1836	1837	1852	1859
.BLKW	1841														
.BYTE	1616														
	21	993	1707	1708	1710	1711	1715	1717	1719	1721	1722	1724	1725	1727	1732
	1734	1736	1738	1740	1742	1744	1746	1747	1751	1754	1755	1804	1806	1807	1809
	1810	1812	1813	1839	1841	1856									
.ENABL	4														
.END	1861														
.ENDC	11	12	14	15	21	23	112	154	161	164	167	216	222	263	272
	290	311	332	356	424	428	491	512	553	583	594	602	613	620	680
	689	691	724	746	748	776	787	805	829	854	876	896	917	931	983
	995	1000	1062	1154	1211	1308	1540	1616	1841	1842	1848	1850	1852	1854	1856
	1857	1858	1859												
.EQUIV	12														
.EVEN	1061	1768	1818	1840	1852	1859									
.IF	11	12	14	15	21	23	112	154	161	164	167	216	222	263	272
	290	311	332	356	424	428	491	512	553	583	594	602	613	620	680
	689	691	724	746	748	776	787	805	829	854	876	896	917	931	983
	995	1000	1062	1154	1211	1308	1540	1616	1841	1842	1848	1850	1852	1854	1856
	1857	1858	1859												
.IFF	12	14	21	112	154	161	164	167	216	222	263	272	290	311	332
	356	424	428	491	512	553	583	594	602	613	620	680	689	691	724
	746	748	776	787	805	829	854	876	896	917	931	983	995	1000	1062
	1154	1211	1308	1540	1616	1841	1842	1848	1850	1852	1854	1856	1857	1858	1859
.IFT	1841	1842	1848	1850											
.IFTF	1841	1842	1848	1850											
.IIF	11	14	15	21	112	983	1841	1842	1848	1950	1852	1854	1858		
.IRP	23	154	263	272	290	311	332	428	512	553	583	594	680	724	746
	776	787	805	829	854	876	896	917	931	983	995	1062	1211	1616	1842
	1850	1857	1859												
.LIST	2	10	12	14	15	21	23	67	153	154	263	272	290	311	332
	428	512	553	583	594	680	724	746	776	787	805	829	854	876	996
	917	931	983	994	995	1062	1211	1224	1841	1848	1850	1858			
.MACRO	14	21	964	1858											
.MCALL	7	8	9	12											
.NLIST	1	3	12	14	15	21	23	63	150	154	263	272	290	311	332
	428	512	553	583	594	680	724	746	776	787	905	929	854	876	896
	917	931	983	992	995	1062	1211	1222	1841	1848	1850	1858			
.PAGE	21														
.REPT	15	21													
.SBTTL	12	14	15	21	151	152	154	263	272	290	311	332	428	512	553
	583	594	680	724	746	776	787	805	829	854	876	896	917	931	983
	993	995	1062	1211	1223	1616	1659	1773	1841	1842	1848	1850	1852	1854	1856
	1857	1858	1859												
.TITLE	11														
.WORD	15	21	235	236	237	238	239	240	241	983	1781	1782	1783	1784	1795
	1788	1789	1790	1791	1795	1796	1800	1801	1819	1820	1821	1822	1823	1824	1825
	1826	1827	1828	1842	1852	1854	1856	1857	1859						

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

* DZVTCC/CRF=DZVTCC.P13
RUN-TIME: 42 28 5 SECONDS
RUN-TIME RATIO: 313/77=4.0
CORE USED: 22K (43 PAGES)

