

VT50A/B/H

ACCEPTANCE TEST
MD-11-DZVTC-D

EP-DZVTC D-DL-B
COPYRIGHT 1977
FICHE 1 OF 1

MAR 1977
digital
MADE IN USA

The image shows a grid of 150 small, illegible data tables or charts arranged in 10 columns and 15 rows. Each cell in the grid contains a small table with multiple columns and rows of text, which appears to be test data or performance metrics. The text is too small and faded to be read, but the layout suggests a comprehensive set of test results for the device.

B01

ECF:027GASEQ

00010000

770224

PDP10 411

HDRIDZVTC0SEQ

00010000

770224

105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154

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

155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
1955. 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 A WHICH TEST PATTERN HE/SHE WISHES TO START. THE OPERATOR NOW DEPRESSES THE LETTER WHICH REPRESENTS THE TEST PATTERN TO BE STARTED WITH. REFER TO THE PROGRAM LISTING TABLE OF CONTENTS FOR THE 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.

196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
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

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
- TESTNUM - 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 CONNED THE PROGRAM WILL TYPE 'END PASS' ON THE CONSOLE WHEN A PASS HAS BEEN COM 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. LOCATION VTNOW CONTAINS THE CURRENT DL11 BASE ADDRESS.

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

8.3 COPIER SAVE PAPER SWITCH

IF 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. PCT IS # OF PASSES)

252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307

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 KEYPAC

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: AA
BB
CC
DDDDDDDDDDDDDDDDDDDDJJDDDDDDDDDDDDDDDDDDDD

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.

MAINDEC-11-DZVTC-D MACY11 27(1006) 21-FEB-77 15:32 PAGE 7
DZVTC.F11 21-FEB-77 15:27

308
309
310
311

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

312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400

9.6 F CURSOR MOTION

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

BEFORE:

```

-----
+5                                     +
+                                     +
+                                     +
+                                     +
+                                     +
+           2                         1   +
+                                     +
+                                     +
+                                     +
+                                     +
+           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) FOURTY 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 FOURTY TWO TIMES AND DISPLAY A "X"

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

AFTER:

```

-----
+X+                                     +
+                                     +
+                                     +
+                                     +
+                                     +
+           X                         X   +
+                                     +
+                                     +
+           X                         X   +
-----

```

K01

MAINDEC-11-DZVTC-D MACY11 27(1006) 21-FEB-77 15:32 PAGE 9
DZVTCO.P11 21-FEB-77 15:27

368

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

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 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE T
T @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ T
T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE 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.

179
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
529

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 CORR 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

530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571

9.23 S COPIER - PERIMETER TEST

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

```
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEE                                                                 EEE
EEE                                                                 EEE
EEE                                                                 EEE
EEE                                                                 EEE
EEE                                                                 EEE
EEE                                                                 EEE
EEE                                                                 EEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

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 APPERING ON SCREEN.

9.26 V PRINT SCREEN TEST

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

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

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 VTSO. 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 VTSO/52 IS IN LOCAL MODE (NO UART/SERIAL LINE) THE CHARACTERS SHOULD BE ECHOED CORRECTLY. IF NOT, THE PROBLEM IS IN THE VTSO UNIT. IF CHANGED TO REMOTE MODE AND THE CHARACTERS ARE IN ERROR, THERE IS A UART/SERIAL LINE PROBLEM.


```

612 .TITLE MAINDEC-11-DZVTC-D
613 : *COPYRIGHT (C) 1975
614 : *DIGITAL EQUIPMENT CORP.
615 : *MAYNARD, MASS. 01754
616 : *
617 : *PROGRAM BY MIKE DENSMORE
618 : *
619 : *THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
620 : *PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
621 : *
622 .SBTTL BASIC DEFINITIONS
623
624 : *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
625 301100 STACK= 1100
626 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
627 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
628
629 : *MISCELLANEOUS DEFINITIONS
630 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
631 000012 LF= 12 ;;CODE FOR LINE FEED
632 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
633 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
634 177776 PS= 177776 ;;PROCESSOR STATUS WORD
635 .EQUIV PS,PSW
636 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
637 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
638 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
639 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
640
641 : *GENERAL PURPOSE REGISTER DEFINITIONS
642 000000 R0= %0 ;;GENERAL REGISTER
643 000001 R1= %1 ;;GENERAL REGISTER
644 000002 R2= %2 ;;GENERAL REGISTER
645 000003 R3= %3 ;;GENERAL REGISTER
646 000004 R4= %4 ;;GENERAL REGISTER
647 000005 R5= %5 ;;GENERAL REGISTER
648 000006 R6= %6 ;;GENERAL REGISTER
649 000007 R7= %7 ;;GENERAL REGISTER
650 000006 SP= %6 ;;STACK POINTER
651 000007 PC= %7 ;;PROGRAM COUNTER
652
653 : *PRIORITY LEVEL DEFINITIONS
654 000000 PR0= 0 ;;PRIORITY LEVEL 0
655 000040 PR1= 40 ;;PRIORITY LEVEL 1
656 000100 PR2= 100 ;;PRIORITY LEVEL 2
657 000140 PR3= 140 ;;PRIORITY LEVEL 3
658 000200 PR4= 200 ;;PRIORITY LEVEL 4
659 000240 PR5= 240 ;;PRIORITY LEVEL 5
660 000300 PR6= 300 ;;PRIORITY LEVEL 6
661 000340 PR7= 340 ;;PRIORITY LEVEL 7
662
663 : *"SWITCH REGISTER" SWITCH DEFINITIONS
664 100000 SW15= 100000
665 040000 SW14= 40000
666 020000 SW13= 20000
667 010000 SW12= 10000

```

668 004000
669 002000
670 001000
671 000400
672 000200
673 000100
674 000040
675 000020
676 000010
677 000004
678 000002
679 000001
680
681
682
683
684
685
686
687
688
689
690
691
692 100000
693 040000
694 020000
695 010000
696 004000
697 002000
698 001000
699 000400
700 000200
701 000100
702 000040
703 000020
704 000010
705 000004
706 000002
707 000001
708
709
710
711
712
713
714
715
716
717
718
719
720 000004
721 000010
722 000014
723 000014

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

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

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

.*BASIC "CPU" TRAP VECTOR ADDRESSES

ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC=14 ;: "T" BIT
TRTVEC= 14 ;: TRACE TRAP

```

724      000014      BPTVEC= 14      ;;BREAKPOINT TRAP (BPT)
725      000020      IOTVEC= 20      ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
726      000024      PWRVEC= 24      ;;POWER FAIL
727      000030      EMTVEC= 30      ;;EMULATOR TRAP (EMT) **ERROR**
728      000034      TRAPVEC=34      ;;"TRAP" TRAP
729      000060      TKVEC= 60      ;;TTY KEYBOARD VECTOR
730      000064      TPVEC= 64      ;;TTY PRINTER VECTOR
731      000240      PIRQVEC=240    ;;PROGRAM INTERRUPT REQUEST VECTOR

732
733      .SBTTL  OPERATIONAL SWITCH SETTINGS
734      ;*
735      ;*      SWITCH      USE
736      ;*      -----
737      ;*      15      HALT ON ERROR
738      ;*      14      LOOP ON TEST
739      ;*      13      INHIBIT ERROR TYPEOUTS
740      ;*      12      INHIBIT SUB-TEST DELAY'S
741      ;*      10      ENABLE SAVE COPIER PAPER MODE
742      ;*      8      LOOP ON TEST IN SWR<7:0>
743      .SBTTL  TRAP CATCHER
744
745      000000      .=0
746      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
747      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
748      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
749      000174      .=174
750      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
751      000176      000000      SWREG: .WORD 0      ;;SOFTWARE SWITCH REGISTER
752      .SBTTL  STARTING ADDRESS(ES)
753      000200      000137      001340      JMP      JMBEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
754      000204      000137      001404      JMP      RBEGIN  ;;JUMP TO RESTART ADDRESS
755      000210      000137      001430      JMP      BEGIN1  ;;JUMP TO KEYBOARD CHARACTER TEST
756      000214      000137      001440      JMP      BEGIN2  ;;JUMP TO CHAR OCTAL VALUE LOOP
757      000220      000137      001420      JMP      BEGIN3  ;;JUMP TO ASCII ECHO LOOP
758      000224      000137      001400      JMP      MANFJ   ;;JUMP TO W.F. SPECIAL TEST PARAM.
    
```

.SBTTL COMMON TAGS

759
760
761
762
763
764
765 001100
766 001100
767 001100 000000
768 001102 000
769 001103 000
770 001104 000000
771 001106 000000
772 001110 000000
773 001112 000000
774 001114 000
775 001115 001
776 001116 000000
777 001120 000000
778 001122 000000
779 001124 000000
780 001126 000000
781 001130 000000
782 001132 000000
793 001134 000
794 001135 000
795 001136 000000
786 001140 177570
787 001142 177570
798 001144 177560
789 001146 177562
790 001150 177564
791 001152 177566
792 001154 000
793 001155 002
794 001156 012
795 001157 000
796 001160 000000
797
798 001162 000000
799 001164 000000
800 001156 077
801 001167 015
802 001170 000012
803

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

.=1100

\$CMTAG: .WORD 0
\$PASS: .WORD 0
\$STNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0
\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$REGAD: .WORD 0
\$REG0: .WORD 0
\$REG1: .WORD 0
\$QJES: .ASCII ' '?
\$CRLF: .ASCII '<15>
\$LF: .ASCIIZ '<12>
: START OF COMMON TAGS
: CONTAINS PASS COUNT
: CONTAINS THE TEST NUMBER
: CONTAINS ERROR FLAG
: CONTAINS SUBTEST ITERATION COUNT
: CONTAINS SCOPE LOOP ADDRESS
: CONTAINS SCOPE RETURN FOR ERRORS
: CONTAINS TOTAL ERRORS DETECTED
: CONTAINS ITEM CONTROL BYTE
: CONTAINS MAX. ERRORS PER TEST
: CONTAINS PC OF LAST ERROR INSTRUCTION
: CONTAINS ADDRESS OF 'GOOD' DATA
: CONTAINS ADDRESS OF 'BAD' DATA
: CONTAINS 'GOOD' DATA
: CONTAINS 'BAD' DATA
: RESERVED--NOT TO BE USED
: AUTOMATIC MODE INDICATOR
: INTERRUPT MODE INDICATOR
: ADDRESS OF SWITCH REGISTER
: ADDRESS OF DISPLAY REGISTER
: TTY KBD STATUS
: TTY KBD BUFFER
: TTY PRINTER STATUS REG. ADDRESS
: TTY PRINTER BUFFER REG. ADDRESS
: CONTAINS NULL CHARACTER FOR FILLS
: CONTAINS # OF FILLER CHARACTERS REQUIRED
: INSERT FILL CHARS. AFTER A "LINE FEED"
: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
: CONTAINS THE ADDRESS FROM
: WHICH (\$REG0) WAS OBTAINED
: CONTAINS ((\$REGAD)+0,
: CONTAINS ((\$REGAD)+2,
: QUESTION MARK
: CARRIAGE RETURN
: LINE FEED
:*****

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

804
805
806
807
808
809
810
811
812
813
814
815
816
817
818 001172
819
820
821
822 001172 020045
823 001174 020270
824 001176 020434
825 001200 000000
826
827
828 001202 020112
829 001204 020270
830 001206 020434
831 001210 000000
832
833
834 001212 020141
835 001214 020317
836 001216 020444
837 001220 000000
838
839
840 001222 020165
841 001224 020366
842 001226 020460
843 001230 000000
844
845
846 001232 020230
847 001234 000000
848 001236 000000
849 001240 000000
850 001242 177560
851
852 001244 000000
853 001246 177560
854 001250 100011
855 001252 000000
856
857 001254 000300
858 001256 000000
859 001260 000000

: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 RCVC
DT4 :\$ERRPC VTNOW TSTNUM \$GDDAT \$BDDAT
0

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

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

TIMEO: 300 :CHARACTER FLAG TIMEOUT CONSTANT
SUBTST: 5 :SUBTEST DELAY CONSTANT
VTXX: 0 :I.D. AND CHARASTICS

860	001262	000053			LASTLN: 53	:OR 67	:LAST VALID LINE # +4C
861	001264	001700			TOTALC: 960.	:OR 1920.	:TOTAL CHARACTER COUNT
862	001266	000014			VHO: 12.	:24.	:VERTICAL LINE COUNT
863	001270	000006			VH1: 6.	:12.	:1/2 VERTICAL LINE COUNT
864	001272	000003			VH2: 3.	:6.	:1/4 VERTICAL LINE COUNT
865	001274	000000			PRTCNT: 0		
866	001276	177560			VTIS: 177560		:DEVICE ADDRESSES
867	001300	177562			VTIB: 177562		:IN DATA
868	001302	177564			VTOS: 177564		:OUT STAT
869	001304	177566			VTOB: 177566		:OUT DATA
870	001306	000000			STCHAR: 0		:TEMP REG'S
871	001310	000140			LASTCH: 140	:OR 200	:FIRST NON-VALID CHARACTER
872	001312	000000			TEMP: 0		:TEMP REG'S
873	001314	000000			TEMPO: 0		
874	001316	000204			PNTWID: 132.		:COLUMN COUNT FOR LINE PRINTER.
875	001320	000120			WIDTH: 80.		:COLUMN WIDTH
876	001322	000000			SAVE1: 0		:TEMP REG'S
877	001324	000000			SAVE2: 0		
878	001326	000000			SAVE3: 0		
879	001330	000000			SAVE4: 0		
880	001332	000000			WFTST: 0		:NON-ZERO IF SA = 224
881							
882	001334	022626			BUSSTR: CMP (SP)+, (SP)+		:POP STACK
883	001336	104005			ERROR 5		:INVALID BUSS ADDRESS
884							
885	001340	012737	002042	002040	BEGIN: MOV #TST1, WHERE		:STARTING ACCEPTANCE TEST ADDRESS
886	001346	005037	001330		CLR SAVE4		
887	001352	005037	001332		CLR WFTST		
888	001356	012737	001334	000004	MOV #BUSSTR, @#4		
889	001364	012737	000340	000006	MOV #340, @#6		
890	001372	005037	001274		CLR PRTCNT		
891	001376	000426			BR GINA		
892	001400	005237	001332		MANFU: INC WFTST		:SET W.F. PARAM.
893	001404	005037	001274		RBEGIN: CLR PRTCNT		:RESTART ADDRESS
894	001410	012737	002042	002040	MOV #TST1, WHERE		
895	001416	000413			BR GIN		
896	001420	012737	010254	002040	BEGIN3: MOV #KRBECH, WHERE		:START AT ECHG LOOP
897	001426	000407			BR GIN		
898	001430	012737	007400	002040	BEGIN1: MOV #KRBTST, WHERE		:STARTING CHARACTER TEST ADDRESS
899	001436	000403			BR GIN		
900	001440	012737	007104	002040	BEGIN2: MOV #KRBECH, WHERE		:STARTING CHARACTER LOOP ADDRESS
901	001446	012737	000001	001330	GIN: MOV #1, SAVE4		
902	001454	000005			GINA: RESET		
903					.SBTL INITIALIZE THE COMMON TAGS		
904					::CLEAR THE COMMON TAGS (\$CMTAG) AREA		
905	001456	012706	001100		MOV #CMTAG, R6		:FIRST LOCATION TO BE CLEARED
906	001462	005026			CLR (R6)+		:CLEAR MEMORY LOCATION
907	001464	022706	001140		CMP #SWR, R6 ;:DONE?		
908	001470	001374			BNE -6		:LOOP BACK IF NO
909	001472	012706	001100		MOV #STACK, SP		:SETUP THE STACK POINTER
910					::INITIALIZE A FEW VECTORS		
911	001476	012737	023224	000020	MOV #SSCOPE, @#IOTVEC		:IOT VECTOR FOR SCOPE ROUTINE
912	001504	012737	000340	000022	MOV #340, @#IOTVEC+2		:LEVEL 7
913	001512	012737	023342	000030	MOV #SEERR, @#EMTVEC		:EMT VECTOR FOR ERROR ROUTINE
914	001520	012737	000340	000032	MOV #340, @#EMTVEC+2		:LEVEL 7
915	001526	012737	024360	000034	MOV #STRAP, @#TRAPVEC		:TRAP VECTOR FOR TRAP CALLS

```

916 001534 012737 000340 000036      MOV      #340, @#TRAPVEC+2; LEVEL 7
917 001542 012737 024436 000024      MOV      #SPWRDN, @#PWRVEC ;: POWER FAILURE VECTOR
918 001550 012737 000340 000026      MOV      #340, @#PWRVEC+2 ;: LEVEL 7
919 001556 013737 006756 006750      MOV      $ENDCT, $EOPCT ;: SETUP END-OF-PROGRAM COUNTER
920 001564 012737 001564 001106      MOV      #, $LPADR ;: INITIALIZE THE LOOP ADDRESS FOR SCOPE
921                                     ;: SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
922                                     ;: EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
923 001572 013746 000004                MOV      @#ERRVEC, -(SP) ;: SAVE ERROR VECTOR
924 001576 012737 001632 000004      MOV      #64$, @#ERRVEC ;: SET UP ERROR VECTOR
925 001604 012737 177570 001140      MOV      #DSWR, SWR ;: SETUP FOR A HARDWARE SWITCH REGISTER
926 001612 012737 177570 001142      MOV      #DDISP, DISPLAY ;: AND A HARDWARE DISPLAY REGISTER
927 001620 022777 177777 177312      CMP      #-1, @SWR ;: TRY TO REFERENCE HARDWARE SWR
928 001626 001012                        BNE      66$ ;: BRANCH IF NO TIMEOUT TRAP OCCURRED
929                                     ;: AND THE HARDWARE SWR IS NOT = -1
930 001630 000403                        BR       65$ ;: BRANCH IF NO TIMEOUT
931 001632 012716 001640 64$:      MOV      #65$, (SP) ;: SET UP FOR TRAP RETURN
932 001636 000002                        RTI
933 001640 012737 000176 001140 65$:      MOV      #SWREG, SWR ;: POINT TO SOFTWARE SWR
934 001646 012737 000174 001142      MOV      #DISPREG, DISPLAY
935 001654 012637 000004 66$:      MOV      (SP)+, @#ERRVEC ;: RESTORE ERROR VECTOR
936
937 001660 005037 011042      CLR      IGNORE
938 001664 005037 011024      CLR      LOOP
939 001670 012737 023210 000020      MOV      #MSCOPE, @#IOTVEC
940 001676 005737 001330      TST     SAVE4 ;: TEST FLAG
941 001702 001036      BNE     SBEGIN ;: BR IF NON-ZERO
942 001704 104401      TYPE
943 001706 012712      TITLE
944 001710 105777 177224      TSTB    @SWR
945 001714 100031      BPL     SBEGIN ;: BR IF CLEARED
946 001716 104401 11$:      TYPE
947 001720 017656      WHAT0 ;: FIND OUT THE DEVICE ADDRESS
948 001722 104410      RDOCT
949 001724 012637 001242      MOV     (SP)+, FIRST ;: SAVE THE ADDRESS
950 001730 022737 160000 001242      CMP     #160000, FIRST
951 001736 101367      BHI     11$ ;: BR IF INVALID
952 001740 005777 177276      TST     @FIRST ;: TEST IF VALID
953 001744 005037 001244      CLR     LAST
954 001750 104401      TYPE
955 001752 017721      WHAT1 ;: FIND OUT THE LAST ADDRESS
956 001754 104410      RDOCT
957 001756 012637 001244      MOV     (SP)+, LAST ;: SAVE LAST ADDRESS
958 001762 005777 177256      TST     @LAST ;: TEST IF VALID
959 001766 012737 000006 000004      MOV     #6, @#4
960 001774 005037 000006      CLR     @#6
961 002000 013737 001242 001246 SBEGIN: MOV     FIRST, VTNOW ;: LOAD INITIAL DEVICE ADDRESS
962
963 002006 012700 001276      RSTR1: MOV     #VTIS, RO ;: LOAD POINTER
964 002012 013701 001246      MOV     VTNOW, R1 ;: LOAD INPUT STAT
965 002016 010120      MOV     R1, (RO)+
966 002020 005721      TST     (R1)+
967 002022 010120      MOV     R1, (RO)+
968 002024 005721      TST     (R1)+
969 002026 010120      MOV     R1, (RO)+
970 002030 005721      TST     (R1)+
971 002032 010110      MOV     R1, (RO)
    
```

```

972 002034 000177 000000          JMP      @WHERE          ;JUMP TO STARTING ADDRESS
973 002040 002042          WHERE: TST1
974                                     ;*****
975                                     ;*TEST 1      A      TERMINAL IDENTIFICATION TEST
976                                     ;*****
977 002042 000004          TST1:  SCOPE
978 002044 004537 011616          JSR      R5,AMSG        ;DISPLAY HEADER
979 002050 013424          M914
980 002052 004537 011616          JSR      R5,AMSG        ;SEND REQUEST FOR IDENTIFICATION
981 002056 017251          RFI
982
983 002060 004737 012620          JSR      PC,GETCHR      ;GET A CHARACTER
984 002064 000537          BR      2$             ;BR BACK IF NO INPUT
985 002066 010037 001330          MOV      RO,SAVE4      ;SAVE RESPONSE
986 002072 004737 012620          JSR      PC,GETCHR      ;GET A CHAR.
987 002076 000532          BR      2$             ;BR IF NO INPUT
988 002100 010037 001324          MOV      RO,SAVE2
989 002104 004737 012620          JSR      PC,GETCHR      ;GET A CHAR.
990 002110 000525          BR      2$             ;BR IF NO INPUT
991 002112 010037 001326          MOV      RO,SAVE3
992 002116 042737 177500 001330          BIC      #177600,SAVE4  ;MASK BIT 7
993 002124 042737 177600 001324          BIC      #177600,SAVE2
994 002132 042737 177600 001326          BIC      #177600,SAVE3
995 002140 005737 001332          TST      WFTST
996 002144 001402          BEQ     10$
997 002146 004737 011544          JSR      PC,ADelay      ;EXTRA DELAY
998 002152 122737 000033 001330 10$:  CMPB     #33,SAVE4      ;TEST FIRST CHAR.
999 002160 001015          BNE     1$             ;BR TO ERROR
1000 002162 122737 000057 001324          CMPB     #'/',SAVE2    ;TEST SECOND CHAR.
1001 002170 001011          BNE     1$             ;BR TO ERROR
1002
1003                                     ;NOW DETERMINE WHICH VTSXX AND ITS CHARASTICS
1004
1005 002172 005000          CLR      RO             ;CLEAR INITIAL POINTER
1006 002174 126037 002400 001326 3$:  CMPB     TYPEPT(RO),SAVE3 ;TEST I.D. TO KNOWN VALJE
1007 002202 001406          BEQ     4$             ;BR IF CORRECT
1008 002204 005720          TST     (RO)+          ;BUMP THE INDEX
1009 002206 105760 002400          TSTB    TYPEPT(RO)    ;CHECK IF MORE VALID I.D.'S
1010 002212 001370          BNE     3$             ;BR IF MORE
1011 002214 104003          1$:     ERROR 3        ;INCORRECT I.D. CODE
1012 002216 005000          11$:   CLR      RO     ;DEFAULT TO VT50A MODE
1013
1014                                     ;HAVE NOW FOUND THE I.D. - REPORT TO CONSOLE
1015
1016 002220 016037 002400 001260 4$:  MOV      TYPEPT(RO),VTSXX ;SAVE I.D. AND CHARASTICS
1017 002226 016037 002432 002242          MOV      MSGTYP(RO),5$ ;SAVE ASCII MESSAGE POINTER
1018 002234 001403          BEQ     13$            ;BR IF ZERO MESSAGE POINTER
1019 002236 004537 011616          JSR      R5,AMSG        ;TELL THE UUT
1020 002242 016460          5$:     VT50A          ;POINTER TO VTSXX MESSAGE
1021
1022 002244 012737 001700 001264 13$:  MOV      #960.,TOTALC    ;LOAD TOTAL CHARACTER COUNT
1023 002252 012737 000014 001266          MOV      #12.,VHO       ;LOAD MAX VERTICAL LINE COUNT
1024 002260 012737 000053 001262          MOV      #53,LASTLN     ;LOAD LAST LINE VALJE +40 FOR DCA TESTING
1025 002266 005737 001260          TST     VTSXX          ;TEST IF 24 LINES AVAIL
1026 002272 100007          BPL     6$             ;BR IF NOT
1027 002274 006337 001264          ASL     TOTALC         ;ADJUST CHARACTER COUNT

```



```

1028 002300 006337 001266 ASL VHO ;ADJUST LINE COUNT
1029 002304 062737 000014 001262 ADD #12.,LASTLN ;ADJUST VALID LINE # +40
1030
1031 002312 013737 001266 001270 6$: MOV VHO,VH1 ;LOAD OTHER LINE COUNTS
1032 002320 006237 001270 ASR VH1
1033 002324 013737 001270 001272 MOV VH1,VH2
1034 002332 006237 001272 ASR VH2
1035 002336 012737 000140 001310 MOV #140,LASTCH ;LOAD FIRST NON-VALID CHARACTER
1036 002344 032737 004000 001260 BIT #BIT11,VT5XX ;TEST IF UPPER-LOWER CASE TERM.
1037 002352 001403 BEQ 7$ ;BR IF NOT
1038 002354 062737 000040 001310 ADD #40,LASTCH ;UPDATE TO ALLOW UP-LW CASE CHARACTER SET
1039
1040 002362 000403 7$: BR 12$ ;;BR AND START TESTING
1041
1042 002364 104002 2$: ERROR 2 ;NO RESPONSE FROM UUT ADTER ASKING FOR IDENTIFY
1043 002366 000713 BR 11$
1044 002370 000240 NOP
1045 002372 004737 011436 12$: JSR PC,DELAY
1046 002376 000431 BR TST2 ;;BR AND START TESTING
1047
1048 ;I.D. VALUES AND CHARACTERISTICS
1049 ; BIT15 = 1 24 LINES
1050 ; BIT14 = 1 COPIER CONNECTED
1051 ; BIT13 = 1 DIRECT CURSOR ADDRESSING (ESC Y) + "ESC-B" + "ESC-D"
1052 ; BIT12 = 1 VT50H KEYPAD
1053 ; BIT11 = 1 UPPER AND LOWER CASE CHARACTERS
1054 ; BIT10 = 1 PRINTER CONNECTED
1055 ; BIT09 = 1 VT52X MODEL
1056
1057 ;LOW BYTE CONTAINS THE I.D. FOR EACH KNOWN VT5??
1058
1059 TYPEPT: .WORD 000101 ;I.D. = 101 ;VT50A
1060 .WORD 040102 ;I.D. = 102 ;VT50B COPIER
1061 .WORD 140103 ;I.D. = 103 ;VT55 24. LINES, COPIER(VT50 BASED)
1062 .WORD 070110 ;I.D. = 110 ;VT50H COPIER DCA VT50H KEYPAD
1063 .WORD 135113 ;I.D. = 113 ;VT52
1064 .WORD 175114 ;I.D. = 114 ;VT52 WITH COPIER
1065 .WORD 137115 ;I.D. = 115 ;VT52 WITH PRINTER.
1066 .WORD 175105 ;I.D. = 105 ;VT55,24LINES.WITH COPIER(VT52 BASED)
1067 0
1068 0
1069 0
1070 0
1071 002430 000000 0
1072
1073 ;ASCII MESSAGE POINTERS
1074
1075 MSGTYP: VT50A ;VT50A NO COPIER
1076 002434 016532 VT50B ;VT50B COPIER
1077 002436 016601 VT55 ;VT55 COPIER
1078 002440 016663 VT50H ;VT50H COPIER
1079 002442 016741 VT52K ;VT52
1080 002444 017024 VT52L ;VT52 WITH COPIER
1081 002446 017105 VT52M ;VT52 WITH PRINTER
1082 002450 017166 VT55E
1083 002452 000000 0

```

```

1084 002454 000000 0
1085 002456 000000 0
1086 002460 000000 0
1087 *****
1088 ;*TEST 2 B FULL SCREEN OF A CHARACTER
1089 *****
1090 002462 000004 †ST2: SCOPE
1091
1092 002464 004537 011616 JSR R5,AMSG
1093 002470 013013 M91
1094
1095 002472 004737 012556 JSR PC,FILLWC ;FILL SCREEN WITH A 'E'S
1096
1097 002476 004737 011436 JSR PC,DELAY
1098
1099 *****
1100 ;*TEST 3 C DATA TRANSFER PATH TEST
1101 *****
1102 002502 000004 †ST3: SCOPE
1103 002504 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1104 002510 013061 M92
1105 002512 013737 001270 001312 MOV VH1,TEMP ;SET-UP A COUNTER
1106
1107 002520 004537 011330 JSR R5,DTPSR ;SET-UP BUFFER
1108 002524 000077 77 ;OCTAL '?'
1109 002526 000100 100 ;OCTAL 'a'
1110
1111 002530 004537 011334 JSR R5,DTPSRB ;SET-UP BUFFER
1112 002534 000125 125 ;OCTAL 'U'
1113 002536 000052 52 ;OCTAL '*'
1114
1115 002540 004737 010412 1$: JSR PC,XPRNT ;DISPLAY THIS LINE
1116 002544 005337 001312 DEC TEMP ;COMPLETED FULL COUNT?
1117 002550 001373 BNE 1$ ;BRANCH IF NOT COMPLETED
1118
1119 002552 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH
1120 *****
1121 ;*TEST 4 D SINGLE CHARACTER ACROSS ALL COLS. (ALL CHARACTERS)
1122 *****
1123 002556 000004 †ST4: SCOPE
1124 002560 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1125 002564 013110 M93
1126 002566 012737 000040 001306 MOV #40,STCHAR ;SET-UP STARTING CHARACTER
1127
1128 002574 013737 001266 001314 1$: MOV VH0,TEMPO ;LOAD COUNT
1129 002602 013701 001306 MOV STCHAR,R1 ;LOAD R1= TO CHARACTER
1130 002606 004737 010310 JSR PC,FILBUF ;LOAD A BUFFER WITH THAT CHARACTER
1131
1132 002612 004737 010412 JSR PC,XPRNT ;DISPLAY A FULL LINE FROM THE BUFFER
1133
1134 002616 005337 001314 DEC TEMPO ;DONE ?
1135 002622 001005 BNE 2$ ;FINISHED
1136 002624 004737 011436 JSR PC,DELAY
1137 002630 013737 001266 001314 MOV VH0,TEMPO
1138 002636 005237 001306 2$: INC STCHAR ;UPDATE THE CHARACTER
1139 002642 023737 001310 001306 CMP LASTCH,STCHAR ;TEST FOR FINAL CHARACTER
    
```

```

1140 002650 001354 BNE 1$ ;BRANCH IF NOT COMPLETED
1141
1142 002652 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH
1143
1144 ;*****
1145 ;*TEST 5 E ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
1146 ;*****
1147 002656 000004 STS: SCOPE
1148 002660 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1149 002664 013152 M94
1150 002666 012737 000040 001306 MOV #40,STCHAR ;SET-UP STARTING CHARACTER
1151
1152 002674 013737 001266 001314 MOV VHD,TEMPO ;LOAD TEMP
1153 002702 013701 001306 1$: MOV STCHAR,R1 ;LOAD R1=TO CHARACTER
1154 002706 004537 010344 ; JSR R5,LIC ;LOAD A BUFFER STARTING WITH
1155 002712 001320 ; WIDTH ; THAT CHARACTER AND WIDTH <BYTE>
1156
1157 002714 004737 010412 JSR PC,XPRNT ;DISPLAY A FULL LINE FROM THE BUFFER
1158
1159 002720 005337 001314 DEC TEMPO ;DONE ?
1160 002724 001005 BNE 2$ ;BR IF YES
1161 002726 004737 011436 JSR PC,DELAY
1162 002732 013737 001266 001314 MOV VHD,TEMPO
1163 002740 005237 001306 2$: INC STCHAR ;UPDATE THE STARTING CHARACTER
1164 002744 023737 001310 001306 CMP LASTCH,STCHAR ;TEST FOR FINAL CHARACTER
1165 002752 001353 BNE 1$ ;BRANCH IF NOT COMPLETED
1166
1167 002754 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH
1168
1169 ;*****
1170 ;*TEST 6 F CURSOR MOTION TEST
1171 ;*****
1171 002760 000004 STS6: SCOPE
1172 002762 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1173 002766 013203 M96
1174
1175 ;ROUTINE TO LOAD REFERENCE LINES FOR CURSOR MOTION TEST
1176
1177 002770 012700 024614 LBMT: MOV #BUFFER,RO ;LOAD POINTER
1178 002774 012720 000065 MOV #65,(RO)+ ;LOAD #5
1179 003000 013701 001270 MOV VHI,R1 ;LOAD R1
1180 003004 005301 DEC R1
1181 003006 004737 003104 JSR PC,MOVDN1 ;MOVE CURSOR DOWN
1182 003012 004737 003122 JSR PC,MOVRIG ;MOVE RIGHT
1183 003016 112720 000062 MOVB #62,(RO)+ ;LOAD #2
1184 003022 004737 003122 JSR PC,MOVRIG ;MOVE RIGHT
1185 003026 004737 003122 JSR PC,MOVRIG ;MOVE RIGHT
1186 003032 112720 000040 MOVB #40,(RO)+
1187 003036 112720 000061 MOVB #61,(RO)+ ;LOAD #1
1188 003042 004737 003100 JSR PC,MOVDWN ;MOVE DOWN
1189 003046 004737 003122 JSR PC,MOVRIG ;MOVE RIGHT
1190 003052 112720 000063 MOVB #63,(RO)+ ;LOAD #3
1191 003056 004737 003122 JSR PC,MOVRIG
1192 003062 004737 003122 JSR PC,MOVRIG ;MOVE RIGHT
1193 003066 112720 000064 MOVB #64,(RO)+ ;LOAD #4
1194 003072 112710 000377 MOVB #377,(RO) ;TERM
1195 003076 000420 BR LBMT1 ;BR TO NEXT PART

```

```

1196
1197 003100 013701 001270      MOVDWN: MOV      VH1,R1
1198 003104 112720 000015      MOVDN1: MOVB    #15,(R0)+      ;LOAD CR
1199 003110 112720 000012      MOV      #12,(R0)+      ;LOAD LF
1200 003114 005301      DEC      R1
1201 003116 001372      BNE     MOVDN1      ;LOOP UNTIL DONE
1202 003120 000207      RTS     PC          ;EXIT
1203
1204 003122 012701 000024      MOVRIQ: MOV      #20,R1      ;LOAD R1
1205 003126 112720 000040      1$:     MOVB    #40,(R0)+      ;LOAD SPACES
1206 003132 005301      DEC      R1
1207 003134 100374      BPL     1$          ;LOOP UNTIL DONE
1208 003136 000207      RTS     PC          ;EXIT
1209
1210 003140 004737 010412      LBMT1: JSR     PC,XPRNT      ;DISPLAY THIS LINE
1211 003144 004737 011436      JSR     PC,DELAY      ;TEST DELAY SWITCH
1212
1213      ;CURSOR MOTION SUBROUTINE
1214      ;IF VT50H TYPE - USE "ESC-D" FOR CURSOR LEFT AND USE "ESC-B" FOR CURSOR DOWN
1215 003150 013701 001270      LCM:    MOV      VH1,R1      ;LOAD COUNT
1216 003154 012700 024614      MOV      #BUFFER,R0      ;LOAD BUFFER POINTER
1217 003160 112720 000033      1$:     MOVB    #33,(R0)+      ;LOAD 'ESC'
1218 003164 112720 000101      MOVB    #101,(R0)+      ;LOAD 'A' CURSOR UP
1219 003170 005301      DEC      R1
1220 003172 001372      BNE     1$          ;LOOP UNTIL DONE
1221 003174 112720 000130      MOVB    #130,(R0)+      ;LOAD 'X'
1222 003200 012701 000054      MOV      #44,R1          ;LOAD COUNT
1223 003204 032737 020000 001260      BIT     #BIT13,VT5XX      ;TEST IF VT50H TYPE
1224 003212 001407      BEQ     20$          ;BR IF NOT
1225 003214 112720 000033      2$:     MOVB    #33,(R0)+      ;LOAD 'ESC'
1226 003220 112720 000104      MOVB    #104,(R0)+      ;LOAD 'CURSOR LEFT'
1227 003224 005301      DEC      R1
1228 003226 100372      BPL     2$          ;LOOP UNTIL DONE
1229 003230 000404      BR      21$
1230 003232 112720 000010      20$:    MOVB    #10,(R0)+      ;LOAD "BACKSPACE"
1231 003236 005301      DEC      R1          ;DONE ALL ?
1232 003240 100374      BPL     20$          ;BR IF NOT
1233 003242 112720 000130      21$:    MOVB    #130,(R0)+      ;LOAD 'X'
1234 003246 112720 000010      MOVB    #10,(R0)+      ;LOAD BACKSPACE
1235 003252 013701 001270      MOV      VH1,R1          ;LOAD COUNT
1236 003256 032737 020000 001260      BIT     #BIT13,VT5XX      ;TEST IF VT50H TYPE
1237 003264 001407      BEQ     30$          ;BR IF NOT
1238 003266 112720 000033      3$:     MOVB    #33,(R0)+      ;LOAD 'ESC' CURSOR DOWN
1239 003272 112720 000102      MOVB    #102,(R0)+      ;
1240 003276 005301      DEC      R1
1241 003300 001372      BNE     3$          ;LOOP UNTIL DONE
1242 003302 000404      BR      31$
1243 003304 112720 000012      30$:    MOVB    #12,(R0)+      ;LOAD CURSOR DOWN (LF)
1244 003310 005301      DEC      R1          ;DONE ?
1245 003312 001374      BNE     30$          ;BR IF NOT
1246 003314 112720 000130      31$:    MOVB    #130,(R0)+      ;LOAD 'X'
1247 003320 112720 000010      MOVB    #10,(R0)+      ;LOAD BACKSPACE
1248 003324 012701 000052      MOV      #42,R1          ;LOAD COUNT
1249 003330 112720 000033      4$:     MOVB    #33,(R0)+      ;LOAD 'ESC'
1250 003334 112720 000103      MOVB    #103,(R0)+      ;LOAD 'C' CURSOR RIGHT
1251 003340 005301      DEC      R1
    
```

```

1252 003342 100372          BPL      4$          ;LOOP UNTIL DONE
1253 003344 112720 000130  MOVB    #130,(R0)+ ;LOAD 'X'
1254 003350 112720 000033  MOVB    #33,(R0)+  ;LOAD 'ESC'
1255 003354 112720 000110  MOVB    #110,(R0)+ ;LOAD 'H' CURSOR HOME
1256 003360 112720 000130  MOVB    #130,(R0)+
1257 003364 112720 000377  MOVB    #377,(R0)+
1258 003370 004737 010412  JSR     PC,XPRNT   ;DISPLAY THIS LINE
1259 003374 004737 011436  JSR     PC,DELAY   ;DELAY
1260 003400 004537 011616  JSR     R5,AMSG    ;
1261 003404 016163          CRLFB
1262 003406 005737 001260  TST     VT5XX      ;TEST IF 24 LINES
1263 003412 100003          BPL     TST7       ;BR IF 12 LINES
1264 003414 004537 011616  JSR     R5,AMSG    ;SCROLL MORE LINES
1265 003420 016163          CRLFB
1266
1267
1268 ;*****
1268 ;TEST 7          G          TAB, BACKSPACE AND BELL TEST
1269 ;*****
1270 003422 000004          †ST7:  SCOPE
1271 003424 004537 011616  JSR     R5,AMSG    ;DISPLAY HEADING
1272 003430 013236          M97
1273
1274 003432 012700 024614  LRL:   MOV     #BUFFER,R0 ;LOAD BUFFER POINTER
1275 003436 112720 000007  MOVB    #7,(R0)+  ;LOAD 'BELL'
1276 003442 012701 000011  MOV     #9,R1     ;LOAD LINE COUNT
1277 003446 012702 000006  1$:   MOV     #6,R2     ;LOAD COLUMN COUNT
1278 003452 112720 000111  MOVB    #111,(R0)+ ;LOAD COLUMN MARK
1279 003456 112720 000040  2$:   MOVB    #40,(R0)+ ;LOAD SPACE
1280 003462 005302          DEC     R2
1281 003464 100374          BPL     2$        ;BR UNTIL DONE
1282 003466 005301          DEC     R1
1283 003470 100366          BPL     1$        ;BR UNTIL FINISHED ALL TAB STOPS
1284 003472 112720 000015  MOVB    #15,(R0)+ ;LOAD CR
1285 003476 112720 000012  MOVB    #12,(R0)+ ;LOAD LF
1286 003502 112720 000377  MOVB    #377,(R0)+ ;LOAD TERM
1287
1288 003506 004737 010412  JSR     PC,XPRNT
1289
1290 003512 004537 003702  JSR     R5,LTAB   ;LOAD TAB LINE #1
1291 003516 000000          0
1292 003520 004737 010412  JSR     PC,XPRNT ;DISPLAY THIS LINE
1293
1294 003524 004537 003702  JSR     R5,LTAB   ;LOAD TAB LINE #2
1295 003530 000001          1
1296 003532 004737 010412  JSR     PC,XPRNT ;DISPLAY THIS LINE
1297
1298 003536 004537 003702  JSR     R5,LTAB   ;LOAD TAB LINE #3
1299 003542 000002          2
1300 003544 004737 010412  JSR     PC,XPRNT ;DISPLAY THIS LINE
1301
1302 003550 004537 003702  JSR     R5,LTAB   ;LOAD TAB LINE #4
1303 003554 000003          3
1304 003556 004737 010412  JSR     PC,XPRNT ;DISPLAY THIS LINE
1305
1306 003562 004537 003702  JSR     R5,LTAB   ;LOAD TAB LINE #5
1307 003566 000004          4

```

```

1308 003570 004737 010412      JSR    PC,XPRNT      ;DISPLAY THIS LINE
1309
1310 003574 004537 003702      JSR    S,LTAB       ;LOAD TAB LINE #6
1311 003600 000005
1312 003602 004737 010412      JSR    PC,XPRNT     ;DISPLAY THIS LINE
1313
1314 003606 004537 003702      JSR    R5,LTAB      ;LOAD TAB LINE #7
1315 003612 000006
1316 003614 004737 010412      JSR    PC,XPRNT     ;DISPLAY THIS LINE
1317
1318 ;NOW EXECUTE THE BACKSPACE SECTION
1319 003620 013702 001320      MOV    WIDTH,R2     ;LOAD WIDTH COUNT
1320 003624 005302      DEC    R2           ;ADJUST WIDTH
1321 003626 005302      DEC    R2           ;    BY 2
1322 003630 012701 000040      MOV    #40,R1       ;LOAD "SPACE" INTO THE LINE
1323 003634 004737 010314      JSR    PC,FILBFB    ;LOAD LINE
1324 003640 013701 001320      MOV    WIDTH,R1     ;LOAD # OF CHARACTER POSITIONS
1325 003644 112720 000130      3$:  MOVB  #'X,(R0)+ ;LOAD ASCII "X"
1326 003650 112720 000010      MOVB  #10,(R0)+    ;LOAD BACKSPACE CODE
1327 003654 112720 000010      MOVB  #10,(R0)+
1328 003660 005301      DEC    R1           ;DONE ALL POSITIONS ?
1329 003662 001370      BNE   3$           ;BR IF NOT
1330 003664 112720 000377      MOVB  #377,(R0)+  ;LOAD TERM.
1331 003670 004737 010412      JSR    PC,XPRNT    ;EXECUTE ASCII CODE
1332 003674 004737 011436      JSR    PC,DELAY    ;TEST DELAY SWITCH
1333 003700 000434      BR    TST10       ;;BR TO NEXT TEST
1334
1335 ;SUBROUTINE TO LOAD THE TAB TEST INTO THE BUFFER
1336
1337 003702 012537 001322      LTAB:  MOV    (R5)+,SAVE1 ;LOAD # OF CHARACTERS
1338 003706 012702 024614      MOV    #BUFFER,R2  ;LOAD BUFFER POINTER
1339 003712 012701 000011      MOV    #9,R1       ;LOAD WIDTH COUNTER
1340 003716 112722 000007      MOVB  #7,(R2)+    ;LOAD BELL AT START OF LINE
1341 003722 112722 000111      3$:  MOVB  #11,(R2)+ ;LOAD 'I'
1342 003726 013700 001322      MOV    SAVE1,R0    ;GET THE NO. OF THE CHAR
1343 003732 001404      BEQ   1$           ;BR IF 0
1344 003734 112722 000101      2$:  MOVB  #101,(R2)+ ;LOAD THE 'A' CHAR.
1345 003740 005300      DEC    R0           ;
1346 003742 001374      BNE   2$           ;LOOP UNTIL DONE
1347 003744 112722 000011      1$:  MOVB  #11,(R2)+ ;LOAD 'TAB' CHAR
1348 003750 005301      DEC    R1           ;
1349 003752 100363      BPL   3$           ;BR TO NEXT TAB COLUMN CHAR
1350 003754 112722 000015      MOVB  #15,(R2)+  ;LOAD 'CR'
1351 003760 112722 000012      MOVB  #12,(R2)+  ;LOAD 'LF'
1352 003764 112722 000377      MOVB  #377,(R2)+ ;LOAD TERM
1353 003770 000205      RTS    R5          ;EXIT
1354
1355 ;*****
1356 ;*TEST 10 H ERASE FROM CURSOR TO END OF LINE
1357 ;*****
1357 003772 000004      TST10: SCOPE
1358 003774 004537 011616      JSR    R5,AMSG     ;DISPLAY HEADING
1359 004000 013304      M910
1360
1361 004002 004737 012556      JSR    PC,FILLWC   ;FILL BUFFER WITH A CHAR
1362
1363 004006 004737 011436      JSR    PC,DELAY    ;DISPLAY THIS LINE
                        ;DELAY
    
```

```

1364
1365 ;LOAD ERASE LINE TEST INTO BUFFER
1366
1367 004012 012700 024614 LODERL: MOV #BUFFER,RO
1368 004016 112720 000033 MOV #33,(RO)+ ;LOAD ESC
1369 004022 112720 000110 MOV #H,(RO)+ ;LOAD HOME
1370 004026 013737 001266 004130 MOV VHO,2$ ;LOAD COUNT
1371 004034 005337 004130 DEC 2$
1372 004040 112720 000033 3$: MOV #33,(RO)+ ;LOAD ESC
1373 004044 112720 000113 MOV #K,(RO)+ ;LOAD ERASE LINE CHAR
1374 004050 005337 004130 DEC 2$ ;FINISHED ?
1375 004054 100427 BMI 1$ ;BR WHEN DONE
1376 004056 012737 000006 004132 MOV #6,10$ ;LOAD COUNT
1377 004064 005737 001260 TST VTSXX ;TEST IF 12 LINES
1378 004070 100005 BPL 4$ ;BR IF 12 LINES
1379 004072 006237 004132 ASR 10$ ;ADJUST HORIZ. COUNT
1380 004076 000240 NOP
1381 004100 000240 NOP
1382 004102 000240 NOP
1383 004104 112720 000033 4$: MOV #33,(RO)+
1384 004110 112720 000103 MOV #C,(RO)+ ;CURSOR RIGHT
1385 004114 005337 004132 DEC 10$
1386 004120 001371 BNE 4$ ;LOOP
1387 004122 112720 000012 MOV #12,(RO)+ ;CURSOR DOWN
1388 004126 000744 BR 3$
1389 004130 000000 2$: 0
1390 004132 000000 10$: 0
1391 004134 112720 000377 1$: MOV #37,(RO)+ ;LOAD TERM
1392
1393
1394 004140 004737 010412 JSR PC,XPRNT ;DISPLAY THIS TEST
1395
1396 004144 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH
1397
1398 ;*****
1399 ;*TEST 11 I ERASE FROM CURSOR TO END OF SCREEN
1400 ;*****
1401 004150 000004 TST11: SCOPE
1402 004152 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1403 004156 013335 M911
1404 004160 004737 012556 JSR PC,FILLWC ;FILL BUFFER WITH A CHAR
1405 ;DISPLAY THIS LINE
1406 004164 004737 011436 JSR PC,DELAY ;DELAY
1407
1408 ;LOAD ERASE SCREEN TEST INTO BUFFER
1409
1410 004170 013737 001320 004264 LODERS: MOV WIDTH,2$ ;LOAD COUNT
1411 004176 006237 004264 ASR 2$
1412 004202 012700 024614 MOV #BUFFER,RO
1413 004206 112720 000010 1$: MOV #10,(RO)+ ;LOAD CURSOR LEFT
1414 004212 005337 004264 DEC 2$ ;DONE ?
1415 004216 100373 BPL 1$ ;BR UNTIL DONE
1416 004220 013737 001266 004264 MOV VHO,2$ ;LOAD COUNT
1417 004226 112720 000033 4$: MOV #33,(RO)+ ;LOAD ESC
1418 004232 112720 000112 MOV #J,(RO)+ ;LOAD ERASE SCREEN
1419 004236 005337 004264 DEC 2$ ;DONE ?

```

```

1420 004242 100405          BMI      3$          ;BR WHEN DONE
1421 004244 112720 000033     MOVB    #33,(R0)+   ;LOAD ESC
1422 004250 112720 000101     MOVB    #'A,(R0)+   ;LOAD CURSOR UP
1423 004254 000764          BR      4$          ;LOOP
1424 004256 112720 000377     3$:    MOVB    #377,(R0)+ ;LOAD TERM
1425 004262 000401          BR      5$
1426 004264 000000     2$:    0
1427
1428 004266 004737 010412     5$:    JSR     PC,XPRNT   ;DISPLAY THIS TEST
1429 004272 004737 011436     JSR     PC,DELAY     ;TEST DELAY SWITCH
1430
1431      ;*****
1432      ;*TEST 12      J      VIDEO COUPLING TEST
1433      ;*****
1434 004276 000004     TST12: SCOPE
1435 004300 004537 011616     JSR     R5,AMSG     ;DISPLAY HEADER
1436 004304 013370     M912
1437 004306 013737 001270 001312     MOV     VH1,TEMP    ;LOAD COUNTER
1438 004314 004537 011616     1$:    JSR     R5,AMSG
1439 004320 016211     PATH
1440
1441 004322 005337 001312     DEC     TEMP        ;FINISHED COUNT ?
1442 004326 001372     BNE     1$         ;BR UNTIL DONE
1443 004330 004737 011436     JSR     PC,DELAY     ;TEST DELAY SWITCH
1444
1445      ;*****
1446      ;*TEST 13      K      DIRECT CURSOR ADDRESS TEST
1447      ;*****
1448 004334 000004     TST13: SCOPE
1449      ;
1450      ;RANDOM NUMBERS ARE GENERATED AND USED AS "X" AND "Y" COORDINATES
1451      ;ADDRESSING A 960 CHARACTER PRINTOUT.
1452      ;VERIFICATION OF DISPLAY IS PERFORMED VISUALLY BY THE USER
1453      ;EXECUTE 1ST TIME USING "ESC-Y" SEQUENCE AND IF I.D. IS AN "H"
1454      ;
1455      ;EXECUTE 2ND TIME USING "CODE 16" SEQUENCE
1456
1455 004336 032737 020000 001260     BIT     #BIT13,VT5XX ;TEST IF DCA TYPE TERMINAL
1456 004344 001002     BNE     3$         ;;BR IF CURSOR ADDRESSING
1457 004346 000137 005074     1$:    JMP     TST14      ;BYPASS THIS TEST
1458 004352 005737 001332     3$:    TST     WFTEST   ;TEST IF IN W.F. MODE
1459 004356 001373     BNE     1$         ;BYPASS IF W.F. MODE
1460 004360 004537 011616     JSR     R5,AMSG     ;ERASE SCREEN AND IDENTIFY TEST
1461 004364 017610     M98
1462 004366 112737 000033 024614     MOVB    #33,BUFFER  ;LOAD "ESC" CODE
1463 004374 112737 000131 024615     MOVB    #'Y,BUFFER+1 ;LOAD ASCII "Y" (D.C.A. ENABLE)
1464 004402 004737 004456     JSR     PC,DCATST   ;RUN TEST WITH ESC Y SEQUENCE
1465 004406 004737 011436     JSR     PC,DELAY     ;DELAY TESTING
1466 004412 122737 000110 001260     CMPB    #'H,VT5XX   ;TEST IF VT50H TYPE DISPLAY
1467 004420 001352     BNE     1$         ;;BR IF NOT VT50H TYPE
1468 004422 004537 011616     JSR     R5,AMSG     ;ERASE SCREEN AND IDENTIFY TEST
1469 004426 017610     M98
1470 004430 112737 000000 024614     MOVB    #0,BUFFER   ;LOAD "SPACE" AS FIRST VALJE
1471 004436 112737 000016 024615     MOVB    #16,BUFFER+1 ;LOAD CODE-16 SEQUENCE
1472 004444 004737 004456     JSR     PC,DCATST   ;RUN TEST WITH CODE 16 SEQUENCE
1473 004450 004737 011436     JSR     PC,DELAY
1474 004454 000734     BR      1$         ;:BR TO NEXT TEST
1475

```



```

1476 004456 012737 123456 024356 DCATST: MOV #123456,$LONUM ;PRIME RANDOM NUMBER GENERATOR
1477 004464 012737 176543 024354 MOV #176543,$HINUM
1478 004472 013737 001264 004750 MOV TOTALC,OVRAL ;LOAD CHAR COUNT
1479 004500 013737 001266 004746 MOV VHC,SET ;LOAD LINE COUNT
1480 004506 012700 024634 MOV #BUFFER+20,RO ;LOAD DESTINATION BUFFER
1481 004512 012701 004752 2$: MOV #MSGTXT,R1 ;LOAD MESSAGE POINTER
1482 004516 012120 1$: MOV (R1)+,(RO)+ ;LOAD 2 CHARACTERS
1483 004520 022701 005072 CMP #MSGTND,R1 ;TEST FOR LAST CHAR
1484 004524 001374 BNE 1$ ;BR UNTIL DONE 1 LINE
1485 004525 005337 004746 DEC SET ;FINISHED ALL LINES
1486 004532 001367 BNE 2$ ;BR IF NOT
1487 004534 012737 177777 024620 MOV #-1,BUFFER+4 ;LOAD MESSAGE TERMINATOR
1488
1489 004542 004737 024256 GENER: JSR %7,$RAND ;GENERATE RANDOM NUMBER
1490 004546 013700 024356 MOV $LONUM,RO ;GET RANDOM NUMBER
1491 004552 042700 177700 BIC #177700,%0 ;RANDOM NO. MUST BE TWO DIGITS
1492 004556 020027 000037 CMP %0,#37 ;NO. MUST BE LESS THAN 40
1493 004562 101767 BLOS GENER ;LOWER, REGENERATION
1494 004564 020037 001262 CMP %0,LASTLN ;NO. MUST NOT BE GREATER THAN 54 OR 57
1495 004570 101364 BHI GENER ;GREATER, REGENERATION
1496 004572 010037 004742 MOV %0,YADDS ;STORE RANDOM Y COORDINATE
1497 004576 010001 MOV %0,%1 ;COPY DATA
1498 004600 012737 024634 004746 MOV #BUFFER+20,SET ;LOAD BASE POINTER
1499 004606 162701 000040 SUB #40,%1 ;MINIMUM Y INDEX
1500 004612 001405 BEQ GENRX ;RESULT, MINIMUM Y COORDINATE
1501 004614 062737 000120 004746 1$: ADD #80.,SET ;SETUP Y INDEX LOCATION FOR PRINTOUT
1502 004622 005301 DEC %1
1503 004624 001373 BNE 1$ ;Y COORDINATE IS SET
1504 004626 004737 024256 GENERX: JSR %7,$RAND ;GENERATE RANDOM NUMBER
1505 004632 013700 024356 MOV $LONUM,RO ;GET A RANDOM NUMBER
1506 004636 042700 177600 BIC #177600,%0 ;RANDOM NO. MAY BE LESS THAN 200
1507 004642 020027 000037 CMP %0,#37 ; MUST NOT BE LESS THAN 40
1508 004646 101767 BLOS GENRX ;LOWER, REGENERATION
1509 004650 020027 000157 CMP %0,#157 ; MUST NOT BE GREATER THAN 157
1510 004654 101364 BHI GENRX ;GREATER, REGENERATION
1511 004656 010037 004744 MOV %0,XADDS ;STORE RANDOM X COORDINATE
1512 004662 162700 000040 SUB #40,%0 ;SETUP MINIMUM X INDEX
1513 004666 060037 004746 ADD %0,SET ;SETUP X COOR, FOR PNTOUT.
1514 004672 013701 004746 MOV SET,%1 ;SETUP CHECK
1515 004676 105711 TSTB (1) ;HAS CURRENT CHAR. ALREADY BEEN USED?
1516 004700 100720 BMI GENER ;YES, REGENERATE
1517 004702 113737 004742 024616 MOVB YADDS,BUFFER+2 ;LOAD Y COORDINATE
1518 004710 113737 004744 024617 MOVB XADDS,BUFFER+3 ;LOAD X COORDINATE
1519 004716 111137 024620 MOVB (1),BUFFER+4 ;LOAD CHARACTER TO BE PRINTED
1520 004722 152711 000200 BLSB #200,(1) ;INDICATE USE OF CURSOR POSITION
1521 004726 004737 010412 JSR PC,XPRNT ;EXECUTE AND PRINT CHARACTER
1522 004732 005337 004750 DEC OVRAL ;MAXIMUM NO. OF COORDINATES
1523 004736 001301 BNE GENER ;BR BACK UNTIL DONE
1524 004740 000207 RTS PC ;EXIT
1525
1526 004742 000000 YADDS: 0
1527 004744 000000 XADDS: 0
1528 004746 000000 SET: 0
1529 004750 000000 OVRAL: 0
1530 004752 052126 030065 050055 MSGTXT: .ASCII \VT50-PLUS-DIRECT-CURSOR-ADDRESSING-TEST\
1531 004760 052514 026523 044504

```

```

1532 004766 042522 052103 041455
1533 004774 051125 047523 026522
1534 005002 042101 051104 051505
1535 005010 044523 043516 052055
1536 005016 051505 124
1537 005021 055 044504 044507
1538 005026 040524 026514 050505
1539 005034 044525 046520 047105
1540 005042 026524 047503 050122
1541 005050 026456 040515 047131
1542 005056 051101 026504 040515
1543 005064 026456 052126 030065
1544 005072 100000
1545
1546
1547
1548
1549 005074 000004
1550 005076 005037 011032
1551 005102 005037 011034
1552 005106 004537 011616
1553 005112 016163
1554 005114 004537 011616
1555 005120 016163
1556
1557 005122 005737 001332
1558 005126 001046
1559 005130 005737 001260
1560 005134 100443
1561 005136 004537 011616
1562 005142 013555
1563
1564 005144 012737 000001 011030
1565 005152 012737 000001 011026
1566 005160 004537 011616
1567 005164 017500
1568
1569 005166 005737 011034
1570 005172 001001
1571
1572 005174 104002
1573
1574
1575
1576 005176 005037 011036
1577 005202 005037 011026
1578 005206 012737 000001 011032
1579 005214 004537 011616
1580 005220 017526
1581
1582 005222 004537 011616
1583 005226 016174
1584
1585 005230 005737 011036
1586 005234 001001
1587

```

.ASCII \-DIGITAL-EQUIPMENT-CORP.-MAYNARD-MA.-VT50\

```

MSGTND: BIT15
: ONLY 12 LINE TERMINALS WILL RUN THIS TEST
: *****
: *TEST 14 L HOLD SCREEN TEST
: *****
†ST14: 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 VT5XX ;TEST IF 12 LINE
      BMI 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 IS ;BR IF SENSED
      ERROR 2 ;ENABLE HOLD SCREEN MODE FAILED TO
      ;INHIBIT THE SCREEN FROM SCROLLING
      ;BY SENDING "X-OFF"
      IS: 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 ZS ;BR IF SENSED

```

```

1588 005236 104002          ERROR 2          ;DISABLE HOLD SCREEN MODE FAILED TO ENABLE
1589                                     ;THE SCREEN TO SCROLL BY SENDING AN "X-CN"
1590
1591 005240 004737 011436    2$:   JSR      PC,DELAY          ;PROGRAM DELAY
1592
1593                                     ;*****
1594                                     ;*TEST 15      M      TEST GRAPHICS MODE AND REV. LINE FEED
1595                                     ;*****
1596 005244 000004          TST15: SCOPE
1597 005246 032737 001030 001260    BIT      #BIT09,VT5XX      ;IS UNIT VT52?
1598 005254 001002          BNE      GRPHST           ;YES-TEST GRAPHICS AND REV. LINE FEED
1599 005256 000137 005356    JMP      COPTST          ;NO-GO CHECK FOR COPIER
1600 005262 012704 000120    GRPHST: MOV      #80,R4
1601 005266 004537 011616    1$:   JSR      R5,AMSG      ;HOME THE CURSOR AND CLEAR THE SCREEN.
1602 005272 016203          HOMERS
1603 005274 004537 011616    JSR      R5,AMSG          ;DISPLAY TEST MESSAGE
1604 005300 013612          M9221
1605 005302 004537 011616    JSR      R5,AMSG          ;ENTER GRAPHICS MODE.
1606 005306 021255          ENGRAF
1607 005310 012737 000045 001314    MOV      #37,TEMPO        ;SET UP TO XMIT 80 LINES
1608 005316 004737 011374    2$:   JSR      PC,GABUF      ;LOAD BUFFER WITH GRAPHICS
1609 005322 004737 010412    JSR      PC,XPRNT         ;DISPLAY 1 LINE
1610 005326 004537 011616    JSR      R5,AMSG          ;ISSUE REV. LINE FEED AND
1611 005332 021236          REVLFF                    ;A CARRIAGE RETURN
1612 005334 005304          DEC      R4               ;DECREMENT BUFFER COUNT
1613 005336 005337 001314    3$:   DEC      TEMPO         ;DONE?
1614 005342 001365          BNE      2$              ;NO-LOOP
1615 005344 004537 011616    JSR      R5,AMSG          ;YES-EXIT GRAPHICS MODE
1616 005350 021262          EXGRAF
1617 005352 004737 011436    JSR      PC,DELAY          ;AND GO CHECK DELAY
1618 005356          COPTST:
1619                                     ;*****
1620                                     ;*TEST 16      N      COPIER - AUTO COPY TEST
1621                                     ;*****
1622 005356 000004          TST16: SCOPE
1623 005360 032737 040000 001260    BIT      #BIT14,VT5XX      ;TEST IF COPIER IS AVAILABLE
1624 005366 001002          BNE      2$              ;BR IF AVAILABLE
1625 005370 000137 006336    3$:   JMP      PRNTST          ;NOT AVAILABLE SO BYPASS COPIER TESTS
1626 005374 032777 004000 173536    2$:   BIT      #BIT11,JSWR      ;TEST IF INHIBIT COPIER TEST SWITCH IS SET
1627 005402 001372          BNE      3$              ;BR IF SET
1628 005404 032777 002000 173526    BIT      #BIT10,JSWR      ;TEST IF PAPER SAVE SWITCH IS SET
1629 005412 001406          BEQ      6$              ;BR IF NOT SET AND START COPIER TESTING
1630 005414 105737 001274          TSTB    PRTCNT           ;TEST IF TIME TO TEST COPIER
1631 005420 001363          BNE      3$              ;NOT TIME TO RUN THE COPIER
1632 005422 013737 001250 001274    MOV      PTCT,PRTCNT      ;RELOAD COUNTER AND TEST COPIER
1633
1634 005430 004537 011616    6$:   JSR      R5,AMSG
1635 005434 013667          M923
1636 005436 004537 011616    JSR      R5,AMSG          ;DISPLAY HEADER
1637 005442 017554          GOAPMD                    ;ENABLE AUTO-COPY
1638
1639 005444 013737 001266 001314    MOV      VHO,TEMPO        ;LOAD A EXECUTION COUNT
1640 005452 012701 000101          MOV      #101,R1          ;LOAD A CHARACTER
1641 005456 004737 010310          JSR      PC,FILBUF        ;LOAD CHARACTER INTO BUFFER
1642 005462 012737 000001 011030    1$:   MOV      #1,XOFFOK
1643 005470 004737 010412          JSR      PC,XPRNT         ;DISPLAY IT

```

```

1644 005474 004737 011436 JSR PC,DELAY ;PROGRAM DELAY
1645 005500 005337 001314 DEC TEMPO ;FINISHED ?
1646 005504 100366 BPL 1$ ;BR IF NOT
1647 005506 012737 000001 011030 MOV #1,XOFFOK
1648 005514 004537 011616 JSR R5,AMSG ;DISABLE AUTO-COPY
1649 005520 017561 GONAPM
1650 005522 004737 011436 JSR PC,DELAY ;ANOTHER DELAY
1651
1652
1653 ;:*****
1654 ;*TEST 17 0 COPIER - FULL SCREEN OF A CHARACTER
1655 ;:*****
1655 005526 000004 †ST17: SCOPE
1656 005530 004537 011616 1$: JSR R5,AMSG ;DISPLAY HEADER
1657 005534 013013 M91
1658 005536 004737 012556 JSR PC,FILLWC ;FILL BUFFER WITH CHAR
1659 ;:*****
1660 005542 012737 000001 011030 MOV #1,XOFFOK ;ALLOW XOFF
1661 005550 004537 011616 JSR R5,AMSG ;COPY INST
1662 005554 017452 GOPRNT
1663 005556 004737 011436 JSR PC,DELAY
1664
1665
1666 ;:*****
1667 ;*TEST 20 P COPIER - DATA PATH TEST
1668 ;:*****
1669 005562 000004 †ST20: SCOPE
1670 005564 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1671 005570 013061 M92
1672 005572 013737 001270 001312 MOV VH1,TEMP ;SET-UP A COUNTER
1673 005600 004537 011330 JSR R5,DTPSR ;SET-UP BUFFER
1674 005604 000077 ;OCTAL '2'
1675 005606 000100 100 ;OCTAL '3'
1676 005610 004537 011334 JSR R5,DTPSRB ;SET-UP BUFFER
1677 005614 000125 125 ;OCTAL 'U'
1678 005616 000052 52 ;OCTAL '*'
1679 005620 004737 010412 1$: JSR PC,XPRNT ;DISPLAY THIS LINE
1680 005624 005337 001312 DEC TEMP ;COMPLETED FULL COUNT?
1681 005630 001373 BNE 1$ ;BRANCH IF NOT COMPLETED
1682 005632 012737 000001 011030 MOV #1,XOFFOK
1683 005640 004537 011616 JSR R5,AMSG
1684 005644 017452 GOPRNT ;COPY INST
1685 005646 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH
1686
1687
1688 ;:*****
1689 ;*TEST 21 Q COPIER - SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
1690 ;:*****
1690 005652 000004 †ST21: SCOPE
1691 005654 004537 011616 JSR R5,AMSG ;DISPLAY HEADING
1692 005660 013110 M93
1693 005662 012737 000040 001306 MOV #40,STCHAR ;SET-UP STARTING CHARACTER
1694 005670 013737 001266 001314 MOV VH0,TEMPO
1695 005676 013701 001306 1$: MOV STCHAR,R1 ;LOAD R1= TO CHARACTER
1696 005702 004737 010310 JSR PC,FILBUF ;LOAD A BUFFER WITH THAT CHARACTER
1697 005706 004737 010412 JSR PC,XPRNT ;DISPLAY A FULL LINE FROM THE BUFFER
1698 005712 005337 001314 DEC TEMPO ;DONE
1699 005716 001011 BNE 3$

```

```

1700 005720 013737 001266 001314      MOV      VHD,TEMPO
1701 005726 012737 000001 011030      MOV      #1,XOFFOK      ;ALLOW XOFF
1702 005734 004537 011616      JSR      R5,AMSG
1703 005740 017452      GOPRNT
1704 005742 005237 001306      3$: INC      STCHAR      ;COPY INST
1705 005746 023737 001310 001306      INC      STCHAR      ;UPDATE THE CHARACTER
1706 005754 001350      CMP      LASTCH,STCHAR  ;TEST FOR FINAL CHARACTER
1707 005756 004537 011616      BNE      1$           ;BRANCH IF NOT COMPLETED
1708 005762 016172      JSR      R5,AMSG      ;CRLF
1709 005764 012737 000001 011030      CRLFA-2
1710 005772 004537 011616      MOV      #1,XOFFOK      ;ENABLE XOFF
1711 005776 017452      JSR      R5,AMSG      ;COPY INST
1712 006000 004737 011436      GOPRNT
1713      JSR      PC,DELAY      ;TEST DELAY SWITCH

```

```

;*****
;*TEST 22      R      COPIER - ROTATING CHARACTERS ACROSS ALL COLS. (ALL CHARACTERS)
;*****

```

```

1714
1715
1716
1717 006004 000004      †ST22: SCOPE
1718 006006 004537 011616      JSR      R5,AMSG      ;DISPLAY HEADING
1719 006012 013152      M94
1720 006014 012737 000040 001306      MOV      #40,STCHAR    ;SET-UP STARTING CHARACTER
1721 006022 013737 001266 001314      MOV      VHD,TEMPO
1722 006030 013701 001306      1$: MOV      STCHAR,R1    ;LOAD R1=TO CHARACTER
1723 006034 004537 010344      JSR      R5,LIC      ;LOAD A BUFFER STARTING WITH
1724 006040 001320      WIDTH      ;THAT CHARACTER AND WIDTH (BYTE)
1725 006042 004737 010412      JSR      PC,XPRNT    ;DISPLAY A FULL LINE FROM THE BUFFER
1726 006046 005337 001314      DEC      TEMPO      ;DONE ?
1727 006052 001011      BNE      3$
1728 006054 013737 001266 001314      MOV      VHD,TEMPO
1729 006062 012737 000001 011030      MOV      #1,XOFFOK
1730 006070 004537 011616      JSR      R5,AMSG
1731 006074 017452      GOPRNT
1732 006076 005237 001306      3$: INC      STCHAR    ;COPY INST
1733 006102 023737 001310 001306      INC      STCHAR    ;UPDATE THE STARTING CHARACTER
1734 006110 001347      CMP      LASTCH,STCHAR ;TEST FOR FINAL CHARACTER
1735 006112 004537 011616      BNE      1$         ;BRANCH IF NOT COMPLETED
1736 006116 016172      JSR      R5,AMSG    ;CRLF
1737 006120 012737 000001 011030      CRLFA-2
1738 006126 004537 011616      MOV      #1,XOFFOK    ;ENABLE XOFF
1739 006132 017452      JSR      R5,AMSG    ;COPY INST
1740 006134 004737 011436      GOPRNT
1741      JSR      PC,DELAY  ;TEST DELAY SWITCH

```

```

;*****
;*TEST 23      S      COPIER - PERIMETER PATTERN
;*****

```

```

1742
1743
1744
1745 006140 000004      †ST23: SCOPE
1746 006142 004537 011616      JSR      R5,AMSG      ;DISPLAY HEADER
1747 006146 013471      M920
1748 006150 012701 000105      MOV      #'E,R1      ;LOAD STARTING CHARACTER
1749 006154 004737 010310      JSR      PC,FILBUF   ;FILL THE BUFFER
1750 006160 004737 010412      JSR      PC,XPRNT   ;DISPLAY A LINE
1751 006164 004737 010412      JSR      PC,XPRNT   ;DISPLAY A LINE
1752 006170 013737 001270 001312      MOV      VHI,TEMP    ;LOAD VERT COUNT
1753 006176 062737 000002 001312      ADD      #2,TEMP     ;UPDATE BY 2
1754 006204 004737 011234      1$: JSR      PC,FIRST  ;FILL FIRST AND LAST
1755 006210 004737 010412      JSR      PC,XPRNT   ;DISPLAY A LINE

```

```

1756 006214 005337 001312          DEC      TEMP          ;DONE ?
1757 006220 001371          BNE      1$
1758 006222 012701 000105          MOV      #'E,R1        ;LOAD STARTING CHAR
1759 006226 004737 010310          JSR      PC,FILBUF     ;LOAD LINE WITH E'S
1760 006232 004737 010412          JSR      PC,XPRNT      ;DISPLAY A LINE
1761 006236 004737 010412          JSR      PC,XPRNT
1762 006242 012737 000001 011030      MOV      #1,XOFFOK
1763 006250 004537 011616          JSR      R5,AMSG
1764 006254 017452          GOPRNT
1765 006256 004737 011436          JSR      PC,DELAY     ;COPY SCREEN
1766
1767
1768          ;*****
1768          ;*TEST 24      T      COPIER - DISCLAIMER STATEMENT
1769          ;*****
1770 006262 000004          †ST24: SCOPE
1771
1772 006264 004537 011616          JSR      R5,AMSG     ;DISPLAY HEADER
1773 006270 013521          M921
1774
1775 006272 005004          CLR      R4
1776 006274 016437 006612 006310 1$:  MOV      DISPCH(R4),10$ ;LOAD A POINTER
1777 006302 001405          BEQ      2$          ;BR IF DONE
1778 006304 004537 011616          JSR      R5,AMSG     ;DISPLAY COPYRIGHT
1779 006310 021532          10$:  MTEXT0
1780 006312 005724          TST
1781 006314 000767          BR      (R4)+       ;UPDATE POINTER
1782
1783
1784 006316 012737 000001 011030 2$:  MOV      #1,XOFFOK     ;ENABLE XOFF
1785 006324 004537 011616          JSR      R5,AMSG     ;COPY SCREEN
1786 006330 017452          GOPRNT
1787 006332 004737 011436          JSR      PC,DELAY
1788 006336 000240          PRNTST: NOP        ;TRY PRINTER TESTS
1789
1790
1791          ;*****
1791          ;*TEST 25      U      PRINTER CONTROLLER MODE TEST
1792          ;*****
1793 006340 000004          †ST25: SCOPE
1794 006342 032737 002000 001260          BIT      #BIT10,VTSXX ;IS UNIT EQUIPPED WITH PRINTER.
1795 006350 001002          BNE      1$          ;YES-TEST IT
1796 006352 000137 006640          JMP      $EOP        ;NO-EXIT TEST
1797 006356 004537 011616          1$:  JSR      R5,AMSG     ;DISPLAY HEADER
1798 006362 013726          MPTCNT
1799 006364 004537 011616          JSR      R5,AMSG     ;ENABLE PRINTER CONTROLLER MODE.
1800 006370 021267          ENPNTM
1801 006372 012737 000040 001306          MOV      #40,STCHAR  ;LOAD INITIAL CHAR.
1802 006400 013701 001306          2$:  MOV      STCHAR,R1
1803 006404 004537 010344          JSR      R5,LIC      ;BUILD A 132 COL. LINE.
1804 006410 001316          PNTWID
1805 006412 012737 000001 011030          MOV      #1,XOFFOK   ;ALLOW XOFF/XON PROTOCOL
1806 006420 004737 010412          JSR      PC,XPRNT    ;DISPLAY IT.
1807 006424 005237 001306          INC      STCHAR      ;INCREMENT 1ST CHAR.
1808 006430 023737 001306 001310          CMP      STCHAR,LASTCH ;ISSUED 92 LINES?
1809 006436 001360          BNE      2$          ;NO-LOOP
1810 006440 004537 011616          JSR      R5,AMSG     ;YES-DISABLE PRINTER
1811 006444 021274          EXPNTM             ;CONTROLLER MODE.

```

```

1812 006446 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH AND EXIT.
1813
1814
1815 ::*****
1816 :*TEST 26 V PRINT SCREEN TEST
1817 006452 000004 :*****
1818 †ST26: SCOPE
1819
1820 006454 004537 011616 JSR R5,AMSG ;DISPLAY PRINT SCREEN MESSAGE
1821 006460 013775 MPTSCN
1822
1823 006462 004737 012556 JSR PC,FILLWC ;FILL THE SCREEN WITH E
1824
1825 006466 012737 000001 011030 MOV #1,XOFFOK ;ALLOW XOFF/XON PROTOCOL
1826 006474 004537 011616 JSR R5,AMSG ;PRINT THEM
1827 006500 017452 GOPRNT
1828
1829 006502 004737 011436 JSR PC,DELAY ;TEST DELAY SWITCH.
1830
1831 ::*****
1832 :*TEST 27 W AUTO PRINT TEST
1833 :*****
1834 006506 000004 †ST27: SCOPE
1835
1836 006510 004537 011616 JSR R5,AMSG
1837 006514 014032 M923A ;DISPLAY HEADER
1838 006516 004537 011616 JSR R5,AMSG ;ENABLE AUTO-PRINT
1839 006522 017554 GOAPMC
1840
1841 006524 013737 001266 001314 MOV VHD,TEMPO ;LOAD A EXECUTION COUNT
1842 006532 012701 000101 MOV #101,R1 ;LOAD A CHARACTER
1843 006536 004737 010310 JSR PC,FILBUF ;LOAD CHARACTER INTO BUFFER
1844 006542 012737 000001 011030 1$: MOV #1,XOFFOK
1845 006550 004737 010412 JSR PC,XPRNT ;DISPEAY IT
1846 006554 004737 011436 JSR PC,DELAY ;PROGRAM DELAY
1847 006560 005337 001314 DEC TEMPO ;FINISHED ?
1848 006564 100366 BPL 1$ ;BR IF NOT
1849 006566 012737 000001 011030 MOV #1,XOFFOK
1850 006574 004537 011616 JSR R5,AMSG ;DISABLE AUTO-PRINT
1851 006600 017561 GONAPM
1852 006602 004737 011436 JSR PC,DELAY ;ANOTHER DELAY
1853
1854 006606 000137 006640 JMP $EOP ;END OF PASS
1855 006612 021532 DISPCH: MTEXT0
1856 006614 021636 MTEXT1
1857 006616 021737 MTEXT2
1858 006620 022041 MTEXT3
1859 006622 022124 MTEXT4
1860 006624 022226 MTEXT5
1861 006626 022327 MTEXT6
1862 006630 022361 MTEXT7
1863 006632 022461 MTEXT8
1864 006634 000000 C
1865 006636 000000 O
1866
1867 .SBTTL END OF PASS ROUTINE

```

```

1868
1869
1870
1871
1872
1873
1874
1875
1876 006640
1877 006640 000004
1878 006642 005737 001244
1879 006646 001414
1880 006650 023737 001244 001246
1881 006656 001410
1882 006660 062737 000010 001246
1883 006666 012737 002042 002040
1884 006674 000137 002006
1885 006700 005737 001100
1886 006704 001002
1887 006706 104401 020000
1888 006712 000005
1889 006714 005737 001332
1890 006720 001402
1891 006722 004737 011544
1892 006726 000240
1893
1894 006730 005037 001102
1895 006734 005237 001100
1896 006740 042737 100000 001100
1897 006746 005327
1898 006750 000001
1899 006752 003022
1900 006754 012737
1901 006756 000001
1902 006760 006750
1903 006762 104401 007027
1904 006766 013746 001100
1905 006772 104405
1906 006774 104401 007024
1907 007000 013700 000042
1908 007004 001405
1909 007006 000005
1910 007010 004710
1911 007012 000240
1912 007014 000240
1913 007016 000240
1914 007020
1915 007020 000137
1916 007022 007044
1917 007024 077 377 000
1918 007027 015 042412 042116
1919 007034 050040 051501 020123
1920 007042 000043
1921 007044 005737 001274
1922 007050 100002
1923 007052 105337 001274

```

```

*****
*INCREMENT THE PASS NUMBER ($PASS)
*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
*IF THERES A MONITOR GO TO IT
*IF THERE ISN'T JUMP TO NOWEOP

$EOP:
SCOPE
TST LAST ; TEST IF MORE
BEQ 1$ ; BR IF NONE
CMP LAST,VTNOW ; IS THIS THE LAST ONE
BEQ 1$ ; BR IF YES
ADD #10,VTNOW
MOV #TST1,WHERE
JMP RSTRT ; TEST NEXT ONE
1$: TST $PASS ; TEST IF FIRST PASS
BNE 2$ ; BR IF NOT
TYPE ,PASHED ; TYPE EOP HEADER
2$: RESET
TST WFTST ; TEST IF W.F. MODE
BEQ 3$ ; BR IF NOT
JSR PC,ADELAY ; EXTRA DELAY
3$: NOP

CLR $STNM ; ZERO THE TEST NUMBER
INC $PASS ; INCREMENT THE PASS NUMBER
BIC #100000,$PASS ; DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ; LOOP?
$ECPCT: .WORD 1
BGT $DOAGN ; YES
MOV (PC)+,a(PC)+ ; RESTORE COUNTER
$ENDCT: .WORD 1
TYPE $ENDMG ; TYPE "END PASS #"
MOV $PASS,-(SP) ; SAVE $PASS FOR TYPEOUT
TYPDS ; GO TYPE--DECIMAL ASCII WITH SIGN
TYPE $ENULL ; TYPE A NULL CHARACTER
$GET42: MOV a#42,RO ; GET MONITOR ADDRESS
BEQ $DOAGN ; BRANCH IF NO MONITOR
RESET ; CLEAR THE WORLD
$ENDAD: JSR PC,(RO) ; GO TO MONITOR
NOP ; SAVE ROOM
NOP ; FOR
NOP ; ACT11
$DOAGN: JMP a(PC)+ ; RETURN
$RTNAD: .WORD NOWEOP
$ENULL: .BYTE -1,-1,0 ; NULL CHARACTER STRING
$ENDMG: .ASCIZ <15><12>/END PASS #/

NOWEOP: TST PRTCNT
BPL 11$
DECB PRTCNT

```



```

1924 007056 012737 002042 002040 11$: MOV #TST1,WHERE
1925 007064 013746 001112 001112 MOV $ERTTL,-(SP)
1926 007070 104405 TYPDS
1927 007072 104401 007024 TYPE , $ENULL
1929 007076 000137 002000 JMP $BEGIN
1929 *****
1930 ;*TEST 30 X KEYBOARD OCTAL VALUE LOOP
1931 *****
1932 007102 000004 TST30: SCOPE
1933 007104 012706 001100 KRBECO: MOV #STACK,SP
1934 007110 004537 011616 JSR R5,AMSG ;DISPLAY HEADER
1935 007114 015702 MKE
1936 007116 004737 012620 1$: JSR PC,GETCHR ;GET CHAR
1937 007122 000775 BR 1$ ;:BR BACK IF NO INPUT
1938 007124 004737 011636 JSR PC,OCTAL ;CONVERT RO TO OCTAL
1939 007130 113737 011726 016125 MOVB DIG0,MKEB ;LOAD DIGIT
1940 007136 113737 011730 016126 MOVB DIG1,MKEB+1 ;LOAD DIGIT
1941 007144 113737 011732 016127 MOVB DIG2,MKEB+2 ;LOAD DIGIT
1942 007152 042700 177600 BIC #177600,RO
1943 007156 001001 BNE 10$
1944 007160 005200 INC RO
1945 007162 012701 007272 10$: MOV #BFCHR,R1 ;LOAD POINTER
1946 007166 121100 5$: CMPB (R1),RO ;TEST IF = TO VALUE IN TABLE ?
1947 007170 001403 BEQ 3$ ;BR IF FOUND
1948 007172 005721 TST (R1)+ ;MOVE POINTER
1949 007174 001374 BNE 5$ ;BR IF MORE
1950 007176 000407 BR 2$ ;BR IF NOT IN LIST
1951 007200 062701 000040 3$: ADD #BFCHAR-BFCHR,R1 ;UPDATE POINTER
1952 007204 112137 016120 MOVB (R1)+,MKEA1 ;LOAD 1ST CHAR
1953 007210 112137 016121 MOVB (R1)+,MKEA1+1 ;LOAD 2ND
1954 007214 000420 BR 4$
1955 007216 120027 000040 2$: CMPB RO,#40 ;TEST IF LESS THAN 40
1956 007222 101010 BHI 6$ ;BR IF ABOVE
1957 007224 062700 000100 ADD #100,RO ;MAKE PRINTABLE
1958 007230 110037 016121 MOVB RO,MKEA1+1 ;SAVE CHAR
1959 007234 112737 000136 016120 MOVB #136,MKEA1 ;ADD A 't' BEFORE CHARACTER
1960 007242 000405 BR 4$
1961 007244 112737 000040 016121 6$: MOVB #40,MKEA1+1 ;LOAD SPACE
1962 007252 110037 016120 MOVB RO,MKEA1 ;LOAD CHARACTER
1963 007256 005237 011042 4$: INC IGNORE ;IGNORE DOUBLE CHARACTER FLAG
1964 007262 004537 011616 JSR R5,AMSG ;DISPLAY MESSAGE
1965 007266 016107 MKEA
1966 007270 000712 BR 1$ ;LOOP BACK
1967
1968 ;TABLE OF DEFINED CHARACTERS
1969
1970 007272 000007 BFCHR: 7 ;BELL CODE
1971 007274 000010 10 ;CURSOR LEFT CODE
1972 007276 000011 11 ;TAB CODE
1973 007300 000012 12 ;LINE FEED CODE
1974 007302 000015 15 ;CARRIAGE RETURN CODE
1975 007304 000023 33 ;ESCAPE CODE
1976 007306 000040 40 ;SPACE CODE
1977 007310 000177 177 ;DELETE CODE
1978 007312 000000 0
1979 007314 000000 0

```

1980 007316 000000
1981 007320 000000
1982 007322 000000
1983 007324 000000
1984 007326 000000
1985 007330 000000

0
0
0
0
0
0

;DEFINED CHARACTER EQUIL

1989 007332 046102
1990 007334 046103
1991 007336 052110
1992 007340 043114
1993 007342 051103
1994 007344 051505
1995 007346 050123
1996 007350 042504
1997 007352 000000 000000 000000
1998 007360 000000 000000 000000
1999 007366 000000 000000 000000
2000 007374 000000

BFCHAR: .ASCII /BL/ ;BELL
.ASCII /CL/ ;CURSOR LEFT
.ASCII /HT/ ;H TAB
.ASCII /LF/ ;LINE FEED
.ASCII /CR/ ;CARRIAGE RETURN
.ASCII /ES/ ;ESCAPE
.ASCII /SP/ ;SPACE
.ASCII /DE/ ;DELETE
0,0,0,0,0,0,0,0,0,0

.EVEN

2001
2002
2003
2004
2005 007376 000004
2006 007400 032737 001000 001260
2007 007406 001403
2008 007410 012703 020512
2009 007414 000402
2010 007416 012703 020474
2011 007422 012706 001100
2012 007426 004537 011616
2013 007432 014067
2014 007434 012302
2015 007436 032737 001000 001260
2016 007444 001404
2017 007446 004537 011616
2018 007452 014362
2019 007454 000403
2020 007456 004537 011616
2021 007462 014245
2022 007464 004737 011734
2023
2024 007470 012302
2025 007472 004537 011616
2026 007476 014445
2027 007500 004737 011734
2028
2029 007504 012302
2030 007506 032737 001000 001260
2031 007514 001404
2032 007516 004537 011616
2033 007522 014611
2034 007524 000403
2035 007526 004537 011616

: *TEST 31 Y KEYBOARD CHARACTER TEST

↑ST31: SCOPE
KRBTST: BIT #BIT09, VTSXX ; IS UNIT A VTS2?
1\$
BEQ 1\$
MOV #V52RW, R3 ; YES-SET UP FOR LOWER CASE CHAR.
BR 2\$
1\$: MOV #V50RW, R3 ; NO-SET UP FOR UPPER CASE CHAR.
2\$: MOV #STACK, SP
A: JSR R5, AM5G ; DISPLAY HEADER
MKB
MOV (R3)+, R2 ; LOAD ROW #
BIT #BIT09, VTSXX ; UNIT A VTS2?
BEQ 1\$; NO-USE UPPER CASE KEYBOARD.
JSR R5, AM5G ; ISSUE VTS2 ROW1 MESSAGE.
MKBB2
BR 2\$
1\$: JSR R5, AM5G ; TOP ROW
MKBB ; CHECK THE ROW
2\$: JSR PC, TSTROW
B: MOV (R3)+, R2 ; LOAD ROW #
JSR R5, AM5G ; 2ND ROW
MKBC
JSR PC, TSTROW ; CHECK 2ND ROW
C: MOV (R3)+, R2 ; LOAD ROW #
BIT #BIT09, VTSXX ; UNIT A VTS2?
BEQ 1\$; NO-USE UPPER CASE KEYBOARD.
JSR R5, AM5G ; ISSUE VTS2 ROW 3 MESSAGE.
MKBD2
BR 2\$
1\$: JSR R5, AM5G

MAINDEC-11-DZVTC-D MACY11 27(1006) 21-FEB-77 15:32 PAGE 41
 DZVTC.P11 21-FEB-77 15:27 T31 Y KEYBOARD CHARACTER TEST

```

2036 007532 014504          MKBD
2037 007534 004737 011734 2$: JSR      PC,TSTROW      ;CHECK ROW 3
2038
2039 007540 012302          MOV      (R3)+,R2      ;LOAD ROW #
2040 007542 004537 011616  JSR      R5,AMSG
2041 007546 014677          MKBE
2042 007550 004737 011734  JSR      PC,TSTROW      ;CHECK ROW 4
2043
2044 007554 012702 020676  MOV      *ROW5,R2
2045 007560 004537 011616  JSR      R5,AMSG
2046 007564 015017          MKBF
2047 007566 004737 011734  JSR      PC,TSTROW      ;CHECK ROW 5
2048
2049          ;TEST THE "LEFT"-SHIFT KEY
2050
2051 007572 004537 011616  D:      JSR      R5,AMSG      ;DEPRESS THE "LEFT-SHIFT" KEY
2052 007576 015055          MKBG
2053 007600 012302          MOV      (R3)+,R2      ;LOAD ROW 1 SHIFTED TABLE
2054 007602 032737 001000 001260 BIT      #BIT09,VT5XX    ;UNIT A VT52?
2055 007610 001404          BEQ      1$            ;NO-USE UPPER CASE KEYBOARD.
2056 007612 004537 011616  JSR      R5,AMSG      ;ISSUE VT52 ROW1 MESSAGE.
2057 007616 014362          MKBB2
2058 007620 000403          BR      2$
2059 007622 004537 011616  1$:    JSR      R5,AMSG      ;TEST ROW 1
2060 007626 014245          MKBB
2061 007630 004737 011734  2$:    JSR      PC,TSTROW      ;TEST THE ROW
2062 007634 004537 011616  JSR      R5,AMSG      ;RELEASE THE SHIFT KEY
2063 007640 014123          MKB1
2064
2065          ;TEST THE "RIGHT-SHIFT" KEY
2066
2067 007642 004537 011616  E:      JSR      R5,AMSG      ;SET THE "RIGHT-SHIFT" KEY
2068 007646 015124          MKBGA
2069 007650 012302          MOV      (R3)+,R2      ;LOAD TABLE POINTER
2070 007652 032737 001000 001260 BIT      #BIT09,VT5XX    ;UNIT A VT52?
2071 007660 001404          BEQ      1$            ;NO-USE UPPER CASE KEYBOARD.
2072 007662 004537 011616  JSR      R5,AMSG      ;ISSUE VT52 ROW1 MESSAGE.
2073 007666 014362          MKBB2
2074 007670 000403          BR      2$
2075 007672 004537 011616  1$:    JSR      R5,AMSG
2076 007676 014245          MKBB
2077 007700 004737 011734  2$:    JSR      PC,TSTROW      ;TEST THE ROW AGAIN WITH THE RIGHT-SHIFT SET
2078 007704 004537 011616  JSR      R5,AMSG
2079 007710 014123          MKB1      ;RELEASE SKIFT KEY
2080          ;TEST THE CONTROL MODE
2081
2082 007712 004537 011616  F:      JSR      R5,AMSG      ;SET CTRL
2083 007716 015174          MKBH
2084 007720 012302          MOV      (R3)+,R2      ;LOAD ROW 1 CTRL TABLE
2085 007722 032737 001000 001260 BIT      #BIT09,VT5XX    ;UNIT A VT52?
2086 007730 001404          BEQ      1$            ;NO-USE UPPER CASE KEYBOARD.
2087 007732 004537 011616  JSR      R5,AMSG      ;ISSUE VT52 ROW1 MESSAGE.
2088 007736 014362          MKBB2
2089 007740 000403          BR      2$
2090 007742 004537 011616  1$:    JSR      R5,AMSG
2091 007746 014245          MKBB

```

MAINDFC-11-DZVTC-D MACY11 27(1006) 21-FEB-77 15:32 PAGE 42
 DZVTC.D.P11 21-FEB-77 15:27 T31 Y KEYBOARD CHARACTER TEST

```

2092 007750 004737 011734 2$: JSR PC,TSTROW ;TEST THE ROW
2093
2094 ;TEST THE VT50H KEYPAD
2095
2096 007754 032737 010000 001260 BIT #BIT12,VT5XX ;TEST IF VT50H EXTRA KEYPAD
2097 007762 001526 BEQ KRBDON ;:BR IF NOT DETECTED
2098 007764 004537 011616 JSR R5,AMSG ;TELL OPERATOR THE TEST NAME
2099 007770 015446 MKBN
2100 007772 012702 021302 MOV #ROW6,R2 ;LOAD ROW POINTER VALUES
2101 007776 004537 011616 JSR R5,AMSG
2102 010002 015235 MKBI ;SET TOP ROW KEYPAD
2103 010004 004737 011734 JSR PC,TSTROW ;TEST THAT ROW
2104 010010 012702 021322 MOV #ROW7,R2
2105 010014 004537 011616 JSR R5,AMSG ;2ND ROW KEYPAD TEST
2106 010020 015270 MKBJ
2107 010022 004737 011734 JSR PC,TSTROW ;TEST 2ND KEYPAD ROW
2108 010026 012702 021334 MOV #ROW8,R2
2109 010032 004537 011616 JSR R5,AMSG
2110 010036 015323 MKBK
2111 010040 004737 011734 JSR PC,TSTROW ;TEST 3RD KEYPAD ROW
2112 010044 012702 021346 MOV #ROW9,R2
2113 010050 004537 011616 JSR R5,AMSG ;TELL ABOUT 4TH ROW
2114 010054 015357 MKBL
2115 010056 004737 011734 JSR PC,TSTROW ;TEST 4TH ROW
2116 010062 012702 021360 MOV #ROW10,R2
2117 010066 004537 011616 JSR R5,AMSG
2118 010072 015412 MKBM
2119 010074 004737 011734 JSR PC,TSTROW ;TEST 5TH ROW
2120 010100 032737 001000 001260 BIT #1000,VT5XX ;IS UNIT A VT52?
2121 010106 001454 BEQ KRBDON ;NO-EXIT
2122 010110 004537 011616 JSR R5,AMSG ;YES-TELL OPERATOR ALT. KEYPAD TEST.
2123 010114 015475 MKB52
2124 010116 004537 011616 JSR R5,AMSG ;PUT THE UNIT IN ALT. KEYPAD MODE.
2125 010122 021243 ENAKP
2126 010124 012702 021366 MOV #ROW6A,R2 ;LOAD ROW POINTER VALUES
2127 010130 004537 011616 JSR R5,AMSG
2128 010134 015235 MKBI ;SET TOP ROW KEYPAD
2129 010136 004737 011734 JSR PC,TSTROW ;TEST THAT ROW
2130 010142 012702 021406 MOV #ROW7A,R2
2131 010146 004537 011616 JSR R5,AMSG ;2ND ROW KEYPAD TEST
2132 010152 015270 MKBJ
2133 010154 004737 011734 JSR PC,TSTROW ;TEST 2ND KEYPAD ROW
2134 010160 012702 021434 MOV #ROW8A,R2
2135 010164 004537 011616 JSR R5,AMSG
2136 010170 015323 MKBK
2137 010172 004737 011734 JSR PC,TSTROW ;TEST 3RD KEYPAD ROW
2138 010176 012702 021462 MOV #ROW9A,R2
2139 010202 004537 011616 JSR R5,AMSG ;TELL ABOUT 4TH ROW
2140 010206 015357 MKBL
2141 010210 004737 011734 JSR PC,TSTROW ;TEST 4TH ROW
2142 010214 012702 021510 MOV #ROW10A,R2
2143 010220 004537 011616 JSR R5,AMSG
2144 010224 015412 MKBM
2145 010226 004737 011734 JSR PC,TSTROW ;TEST 5TH ROW
2146 010232 004537 011616 JSR R5,AMSG ;EXIT ALT. KEYPAD MODE.
2147 010236 021250 EXAKP

```

```

2148
2149 ;COMPLETION OF KEYBOARD-KEYPAD TEST
2150
2151 010240 004537 011616 KRBDON: JSR R5,AMSG ;END OF KEYBOARD TEST
2152 010244 015634 MKBR
2153 010246 000137 007400 JMP KRBST ;LOOP
2154 ;*****
2155 ;*TEST 32 Z KEYBOARD ECHO LOOP
2156 ;*****
2157 010252 000004 †ST32: SCOPE
2158 010254 012706 001100 KRBECH: MOV #STACK,SP
2159 010260 004537 011616 JSR R5,AMSG ;DISPLAY HEADER
2160 010264 016134 MKEH
2161
2162 010266 004737 012620 1$: JSR PC,GETCHR ;GET CHARACTER
2163 010272 000775 BR 1$
2164 010274 110077 171004 MOVB R0,AVT09 ;LOAD THE CHARACTER
2165 010300 105777 170776 2$: TSTB AVT09 ;WAIT FOR DONE
2166 010304 100375 BPL 2$
2167 010306 000767 BR 1$ ;LOOP BACK
2168
2169 ;LOAD A SINGLE CHARACTER ACROSS THE SCREEN WIDTH
2170 ;
2171
2172 010310 013702 001320 FILBUF: MOV WIDTH,R2 ;LOAD WIDTH VALUE
2173 010314 012700 024614 FILBFB: MOV #BUFFER,R0 ;SET-UP BUFFER POINTER
2174 010320 112720 000015 MOVB #15,(R0)+ ;LOAD 'CR'
2175 010324 112720 000012 MOVB #12,(R0)+ ;LOAD 'FL'
2176 010330 110120 FILBFA: MOVB R1,(R0)+ ;SAVE THE CHARACTER IN THE BUFFER
2177 010332 005302 DEC R2 ;FINISHED?
2178 010334 001375 BNE FILBFA ;BRANCH IF NOT COMPLETED
2179 010336 112710 000377 MOVB #377,(R0) ;LOAD TERM.
2180 010342 000207 RTS PC ;EXIT
2181
2182 ;LOAD A INCREMENTING CHARACTER ACROSS THE SCREEN WIDTH
2183 ;ONLY 40 THRU 177 ARE LEGAL CHARACTERS
2184
2185 010344 012700 024614 LIC: MOV #BUFFER,R0 ;SET-UP BUFFER POINTER
2186 010350 013502 MOV AV(5)+,R2 ;SET-UP WIDTH
2187 010352 112720 000015 MOVB #15,(R0)+ ;LOAD 'CR'
2188 010356 112720 000012 MOVB #12,(R0)+ ;LOAD 'LF'
2189 010362 110120 LICA: MOVB R1,(R0)+ ;SAVE A CHARACTER IN THE BUFFER
2190 010364 005201 INC R1 ;UPDATE THE CHARACTER
2191 010366 023701 001310 CMP LASTCH,R1 ;TEST FOR
2192 010372 001002 BNE LICB ;BRANCH IF NOT
2193 010374 012701 000040 MOV #40,R1 ;MAKE A LEGAL CHARACTER
2194 010400 005302 LICB: DEC R2 ;DECREMENT COUNT
2195 010402 001367 BNE LICA ;BRANCH IF NOT COMPLETED
2196 010404 112710 000377 MOVB #377,(R0) ;LOAD TERM
2197 010410 000205 RTS R5 ;EXIT
2198
2199 ;DISPLAY SUBROUTINE
2200
2201 010412 012700 024614 XPRNT: MOV #BUFFER,R0 ;SETUP BUFFER POINTER
2202 010416 105777 170663 XPRNTA: TSTB AVT09 ;TEST READY
2203 010422 100404 BMI XPRNTB ;BRANCH IF SET
    
```

2204	010424	005777	170652		TST	QVTOS			:TEST ERROR
2205	010430	100372			BPL	XPRNTA			:BRANCH IF RESET
2206	010432	104001			ERROR	1			:ERROR FLAG SET ON TRANSMITTER STATUS
2207	010434	112001			XPRNTB:	MOVB	(0)+,R1		
2209	010435	100563			BMI	1\$:BR IF MINUS
2209	010440	122701	000033		5\$:	CMPB	#33,R1		:TEST FOR ESC
2210	010444	001003			BNE	3\$:BR IF NOT
2211	010446	005237	011040		INC	ANESC			:SET SOFT FLAG
2212	010452	000402			BR	4\$			
2213	010454	005037	011040		3\$:	CLR	ANESC		:CLEAR SOFT FLAG
2214	010460	110177	170620		4\$:	MOVB	R1,QVTOB		:LOAD CHAR
2215	010464	105777	170606		TSTB	QVTIS			:TEST INPUT FLAG
2216	010470	100352			BPL	XPRNTA			:BR IF CLEARED
2217	010472	005737	011040		TST	ANESC			:TEST IF ESC
2218	010476	001347			BNE	XPRNTA			
2219	010500	013737	001254	012672	52\$:	MOV	TIMED,TIME1		
2220	010506	005037	012674		CLR	TIME2			:LOAD DELAY
2221	010512	105777	170560		2\$:	TSTB	QVTIS		:TEST IF INPUT FLAG
2222	010516	100407			BMI	53\$:BR IF SET
2223	010520	005337	012674		DEC	TIME2			:DELAY
2224	010524	001372			BNE	2\$			
2225	010526	005337	012672		DEC	TIME1			:DELAY
2226	010532	001367			BNE	2\$			
2227	010534	000440			BR	60\$:REPORT ERROR
2228	010536	005737	011042		53\$:	TST	IGNORE		:IGNORE KEYBOARD CHARACTERS ?
2229	010542	001104			BNE	15\$:BR IF YES
2230	010544	017737	170530	001126	MOV	QVTIB,\$BDDAT			:READ CHAR
2231	010552	042737	177600	001126	BIC	#177600,\$BDDAT			:MASK
2232	010560	022737	000021	001126	CMP	#XON,\$BDDAT			:TEST FOR X ON
2233	010566	001003			BNE	50\$			
2234	010570	005237	011036		INC	XONRC			:SET FLAG
2235	010574	000710			BR	XPRNTA			:START AGAIN
2236	010576	022737	000023	001126	50\$:	CMP	#XOFF,\$BDDAT		:TEST FOR X OFF
2237	010604	001020			BNE	51\$:BR IF NOT
2238	010606	005237	011034		INC	XOFFRC			
2239	010612	005737	011030		TST	XOFFOK			:XOFF ENABLED ?
2240	010616	001730			BEQ	52\$			
2241	010620	012737	000001	011032	MOV	#1,AXOFF			:SET X OFF SOFT FLAG
2242	010626	005737	011026		TST	XOFFBR			:TEST
2243	010632	001271			BNE	XPRNTA			:BR BACK
2244	010634	000721			BR	52\$:BR
2245	010636	012737	000000	001124	60\$:	MOV	#0,\$GDDAT		
2246	010644	000437			BR	13\$			
2247	010646	105777	170266		51\$:	TSTB	QSWR		:TEST SWR
2248	010652	100371			BPL	60\$:BR IF CLEARED
2249	010654	012737	000057	001124	MOV	#',,\$GDDAT			:LOAD EXPECTED
2250	010662	023737	001124	001126	CMP	\$GDDAT,\$BDDAT			:COMPARE
2251	010670	001437			BEQ	14\$:BR IF EQUAL
2252	010672	012737	000134	001124	MOV	#\",,\$GDDAT			:LOAD EXPECTED
2253	010700	023737	001124	001126	CMP	\$GDDAT,\$BDDAT			:COMPARE
2254	010706	001016			BNE	13\$:BR IF NOT
2255									
2256									
2257									
2258	010710	012737	000001	011024	MOV	#1,LOOP			:SET SOFT FLAG
2259	010716	012737	017256	011062	MOV	#M00,FINDTA			:SETUP MESSAGE

;"" OR LOOP EXIT

```

2260 010724 005037 011026      16$: CLR      XOFFBR
2261 010730 005037 011030      CLR      XOFFOK
2262 010734 005037 011042      CLR      IGNORE
2263 010740 000137 011052      JMP      FINDOT
2264
2265 010744 005737 011042      13$: TST      IGNORE      :IGNORE INPUT CHARACTER
2266 010750 001001          BNE      15$
2267 010752 104004          ERROR    4      :UNEXPECTED OR INCORRECT INPUT FLAG
2268 010754 000240      15$: NOP
2269 010756 000240      NOP
2270 010760 000240      NOP
2271 010762 000240      NOP
2272 010764 000137 010416      JMP      XPRNTA
2273
2274      :"/" OR STANDARD EXIT
2275
2276 010770 005037 011024      14$: CLR      LOOP
2277 010774 012737 017343 011062      MOV      #MQ1,FINDTA :SETUP MESSAGE
2278 011002 000137 010724      JMP      16$
2279
2280      :NORMAL EXIT
2281
2282 011006 005037 011030      1$: CLR      XOFFOK
2283 011012 005037 011042      CLR      IGNORE
2284 011016 005037 011026      CLR      XOFFBR
2285 011022 000207          RTS      PC      :EXIT
2286
2287 011024 000000      LOOP: 0
2288 011026 000000      XOFFBR: 0
2289 011030 000000      XOFFOK: 0
2290 011032 000000      AXOFF: 0
2291 011034 000000      XOFFRC: 0
2292 011036 000000      XONRC: 0
2293 011040 000000      ANESC: 0
2294 011042 000000      IGNORE: 0      :WHEN SET IGNORE KEYBOARD FLAGS
2295
2296      :DETERMINE THE TEST TO GO TO
2297 011044 004537 011616      FNDA: JSR      R5,AMSG
2298 011050 017431          MQ2
2299 011052 012706 001100      FINDOT: MOV      #STACK,SP :ERROR ASK AGAIN
2300 011056 004537 011616      JSR      R5,AMSG
2301 011062 017343          MQ1
2302 011064 004737 012620      FINDTA: JSR      PC,GETCHR
2303 011070 000770          BR      FINDOT
2304 011072 042700 100600      BIC      #100600,R0 :MASK
2305 011076 122700 000101      CMPB    #'A,R0 :TEST FOR NUMBER
2306 011102 101360          BHI     FNDA
2307 011104 122700 000132      CMPB    #'Z,R0 :TEST FOR OTHERS
2308 011110 103755          BLO     FNDA
2309 011112 042700 177740      BIC      #177740,R0 :MAKE 0-32
2310 011116 005300          DEC     R0
2311 011120 110037 001102      MOVB    R0,$TSTNM :LOAD THAT TEST #
2312 011124 006300          ASL     R0
2313 011126 005760 011152      TST     DSPCH(R0) :TEST IF VALID
2314 011132 001744          BEQ     FNDA :BR IF NOT
2315 011134 000240          NOP

```

```

2316 011136 000240      NOP
2317 011140 016037 011152 001106      MOV      DSPCH(RO), $LPADR      :LOAD LOOP ADDRESS
2318 011146 000170 011152      JMP      @DSPCH(RO)      :GO TO THAT TEST
2319
2320      :SUBTEST DISPATCH TABLE
2321
2322 011152 002044      DSPCH:  TST1+2
2323 011154 002464      TST2+2
2324 011156 002504      TST3+2
2325 011160 002560      TST4+2
2326 011162 002660      TST5+2
2327 011164 002762      TST6+2
2328 011166 003424      TST7+2
2329 011170 003774      TST10+2
2330 011172 004152      TST11+2
2331 011174 004300      TST12+2
2332 011176 004336      TST13+2
2333 011200 005076      TST14+2
2334 011202 005246      TST15+2
2335 011204 005360      TST16+2
2336 011206 005530      TST17+2
2337 011210 005564      TST20+2
2338 011212 005654      TST21+2
2339 011214 006006      TST22+2
2340 011216 006142      TST23+2
2341 011220 006264      TST24+2
2342 011222 006342      TST25+2
2343 011224 006454      TST26+2
2344 011226 006510      TST27+2
2345 011230 007104      TST30+2
2346 011232 007400      TST31+2
2347
2348      :SUBROUTINE TO LOAD COPIER TEST
2349 011234 012700 024614      FIRLST: MOV      #BUFFER, R0
2350 011240 112720 000015      MOVVB   #15, (R0)+      :LOAD CR
2351 011244 112720 000012      MOVVB   #12, (R0)+      :LOAD LF
2352 011250 112720 000105      MOVVB   #'E, (R0)+
2353 011254 112720 000105      MOVVB   #'E, (R0)+
2354 011260 112720 000105      MOVVB   #'E, (R0)+
2355 011264 013702 001320      MOV      WIDTH, R2      :LOAD WIDTH
2356 011270 163702 001270      SUB      VH1, R2
2357 011274 005302      DEC      R2
2358 011276 112720 000040      1$:  MOVVB   #40, (R0)+      :LOAD A SPACE
2359 011302 005302      DEC      R2      :DONE
2360 011304 100374      BPL      1$      :BR UNTIL DONE
2361 011306 112720 000105      MOVVB   #'E, (R0)+
2362 011312 112720 000105      MOVVB   #'E, (R0)+
2363 011316 112720 000105      MOVVB   #'E, (R0)+      :LOAD 80 TH
2364 011322 112710 000377      MOVVB   #377, (R0)      :LOAD TERM
2365 011326 000207      RTS      PC      :EXIT
2366
2367      :SUBROUTINE FOR THE DATA PATH TEST
2368
2369 011330 012700 024614      DTCSR:  MOV      #BUFFER, R0
2370 011334 012501      DTCSR:  MOV      (5)+, R1      :GET FIRST CHARACTER
2371 011336 012502      MOV      (5)+, R2      :GET SECOND CHARACTER

```



```

2372 011340 013703 001320      MOV      WIDTH,R3      ;SET THE WIDTH
2373 011344 006203      ASR      R3            ;DIVIDE BY 2
2374 011346 112720 000015      MOVVB   #15,(R0)+     ;LOAD 'CR'
2375 011352 112720 000012      MOVVB   #12,(R0)+     ;LOAD 'LF'
2376 011356 110120      DTPSRA: MOVVB   R1,(0)+  ;
2377 011360 110220      MOVVB   R2,(0)+
2378 011362 005303      DEC     R3
2379 011364 100374      BPL     DTPSRA
2380 011366 112710 000377      MOVVB   #377,(R0)    ;LOAD TERM
2381 011372 000205      RTS     R5
2382
2383      ;SUBROUTINE TO LOAD BUFFER WITH GRAPHICS CHARACTERS
2384
2385 011374 012700 024614      GBBUF:  MOV     #BUFFER,R0 ;LOAD BUFFER ADDRESS
2386 011400 012701 000136      MOV     #136,R1        ;LOAD INITIAL CHAR.
2387 011404 010402      MOV     R4,R2         ;LOAD BUFFER COUNT
2388 011406 110120      1$:    MOVVB   R1,(R0)+   ;INSERT A CHAR. IN THE BUFFER
2389 011410 005201      INC     R1            ;INCREMENT CHAR.
2390 011412 122701 000177      CMPB   #177,R1        ;AT END OF GRAPHICS STRING?
2391 011416 001002      BNE    2$            ;NO
2392 011420 012701 000136      MOV     #136,R1        ;YES-RESET IT TO 1ST GRAPH. CHAR.
2393 011424 005302      2$:    DEC     R2         ;DECREMENT BUFFER COUNT.
2394 011426 001367      BNE    1$            ;NOT AT END-LOOP
2395 011430 112710 000377      MOVVB   #377,(R0)    ;END OF BUFFER-INSERT TERMINATOR
2396 011434 000207      RTS     PC            ;AND EXIT.
2397
2398      ;PROGRAM DELAY ROUTINE
2399
2400 011436 013737 001256 011540      DELAY:  MOV     SUBTST,10$ ;LOAD COUNT
2401 011444 005037      CLR     11$
2402 011450 005737 001332      TST    WFTST         ;TEST IF W.F. MODE
2403 011454 001413      BEQ    2$            ;BR IF NOT
2404 011456 006237 011540      ASR    10$           ;CHANGE DELAY TIMER
2405 011462 006237 011540      ASR    10$
2406 011466 006237 011540      ASR    10$
2407 011472 000240      NOP
2408 011474 000240      NOP
2409 011476 000240      NOP
2410 011500 000240      NOP
2411 011502 000240      NOP
2412 011504 032777 010000 167426 2$:    BIT    #BIT12,DSWR    ;TEST SR
2413 011512 001006      BNE    3$            ;BR IF SET
2414 011514 005337 011542      DEC    11$           ;DELAY
2415 011520 001371      BNE    2$
2416 011522 005337 011540      DEC    10$
2417 011526 100366      BPL    2$            ;DELAY
2418 011530 000240      3$:    NOP
2419 011532 000240      NOP
2420 011534 000240      NOP
2421 011536 000207      RTS     PC            ;EXIT
2422
2423 011540 000002      10$:   2
2424 011542 000000      11$:   0
2425
2426 011544 013737 001256 011612      ADELAY: MOV     SUBTST,10$
2427 011552 005037      CLR     11$

```

```

2428 011556 006237 011612          ASR      10$
2429 011562 006237 011612          ASR      10$
2430 011566 005337 011614          2$:     DEC      11$
2431 011572 001375                   BNE      2$
2432 011574 005337 011612          DEC      10$
2433 011600 100372                   BPL      2$
2434 011602 000240                   NOP
2435 011604 000240                   NOP
2436 011606 000240                   NOP
2437 011610 000207                   RTS      PC
2438 011612 000000          10$:     0
2439 011614 000000          11$:     0
2440
2441
2442          ;HEADER SUBROUTINE FOR VT-5C
2443
2444 011616 012537 011626          MSG:    MOV      (R5)+,10$      ;GET POINTER
2445 011622 004537 012676          1$:     JSR      R5,MT0B      ;MOVE TO BUFFER
2446 011626 000000          10$:     0
2447 011630 004737 010412          11$:     JSR      PC,XPRNT    ;DISPLAY IT
2448 011634 000205          RTS      R5              ;EXIT
2449
2450          ;OCTAL - 3 BIT CONVERSION
2451
2452
2453 011636 010001          OCTAL:  MOV      R0,R1          ;LOAD R1
2454 011640 042701 177770          BIC      #177770,R1        ;MASK
2455 011644 062701 000060          ADD      #60,R1
2456 011650 110137 011732          MOVVB   R1,DIG2          ;SAVE LSD
2457 011654 010001          MOV      R0,R1
2458 011656 006001          ROR     R1
2459 011660 006001          ROR     R1
2460 011662 006001          ROR     R1
2461 011664 042701 177770          BIC      #177770,R1
2462 011670 062701 000060          ADD      #60,R1
2463 011674 110137 011730          MOVVB   R1,DIG1          ;SAVE IT
2464 011700 010001          MOV      R0,R1
2465 011702 006101          ROL     R1
2466 011704 006101          ROL     R1
2467 011706 000301          SWAB   R1
2468 011710 042701 177770          BIC      #177770,R1
2469 011714 062701 000060          ADD      #60,R1
2470 011720 110137 011726          MOVVB   R1,DIG0          ;SAVE MSD
2471 011724 000207          RTS      PC              ;EXIT
2472
2473 011726 000000          DIG0:   0
2474 011730 000000          DIG1:   0
2475 011732 000000          DIG2:   0
2476
2477          ;SUBROUTINE FOR THE KEYBOARD CHARACTER TEST
2478
2479 011734 004537 011616          TSTROW: JSR      R5,MSG      ;DISPLAY HEADER
2480 011740 014156          MKBA
2481
2482 011742 004737 012620          1$:     JSR      PC,GETCHR    ;GET CHAR
2483 011746 000775          BR      1$              ;:BR BACK IF NO INPUT

```

```

2484 011750 012737 177600 012326      MOV      #177600, MASK1
2485 011756 005037 012330      CLR      MASK2
2486 011762 032777 000004 167150      BIT      #BIT2, QSWR      ; TEST SWR
2487 011770 001416          BEQ      4$              ; DO NOT TEST PARITY BIT
2488 011772 042737 000200 012326      BIC      #BIT7, MASK1     ; ENABLE PARITY BIT
2489 012000 032777 000002 167132      BIT      #BIT1, QSWR     ; TEST IF FORCED PARITY
2490 012006 001424          BEQ      5$              ; BR IF NOT FORCED PARITY BIT
2491 012010 032777 000001 167122      BIT      #BIT0, QSWR     ; TEST FOR EVEN/ODD PARITY
2492 012016 001403          BEQ      4$              ; BR IF ALWAYS OFF
2493 012020 052737 000200 012330      BIS      #BIT7, MASK2     ; SET BIT 7
2494 012026 011237 012322 4$:      MOV      (R2), 100$      ; GET EXPECTED
2495 012032 053737 012330 012322      BIS      MASK2, 100$     ; SET BIT 7 IF EXPECTED
2496 012040 043700 012326          BIC      MASK1, R0       ; MASK VALUE READ
2497 012044 120037 012322          CMPB    R0, 100$       ; COMPARE CHARS
2498 012050 001041          BNE      2$              ; BR IF NOT EQUAL
2499 012052 005722          TST     (R2)+           ; BUMP R2
2500 012054 100332          BPL     1$              ; LOOP TILL DONE
2501 012056 000207          RTS     PC              ; EXIT
2502
2503          ; COME HERE ONLY IF TESTING "PARITY" OPTION
2504
2505 012060 005037 012324 5$:      CLR      101$           ; CLEAR TEMP
2506 012064 011237 012322          MOV     (R2), 100$     ; CLEAR CHAR SAVE
2507 012070 006037 012322 20$:     ROR     100$           ; ROTATE CHAR
2508 012074 103002          BCC     21$           ; BR IF NO CARRY
2509 012076 005237 012324          INC     101$           ; UPDATE CNT
2510 012102 105737 012322 21$:     TSTB   100$           ; DONE ?
2511 012106 001370          BNE     20$           ; BR IF NOT
2512 012110 032777 000001 167022      BIT     #BIT0, QSWR     ; TEST EVEN/ODD
2513 012116 001407          BEQ     23$           ; BR IF OPER. SAYS EVEN
2514 012120 006037 012324          ROR     101$           ;
2515 012124 103403          BCS     22$           ; BR IF ODD ALREADY
2516 012126 052737 000200 012330      BIS     #BIT7, MASK2     ; SET PARITY BIT
2517 012134 000734          BR     4$              ; BR TO TEST CHAR
2518 012136 006037 012324 22$:     ROR     101$           ;
2519 012142 103003          BCC     24$           ; BR IF EVEN ALREADY
2520 012144 052737 000200 012330      BIS     #BIT7, MASK2     ;
2521 012152 000725          BR     4$              ; BR TO TEST CHAR
2522
2523          ; COME HERE IF EXPECTED NOT EQUAL TO RECVD
2524          ; CONVERT RESULTS TO OCTAL FOR TYPEOUT
2525
2526 012154 010037 001126 2$:      MOV     R0, $BDDAT     ; LOAD BAD CHARACTER
2527 012160 004737 011636          JSR    PC, OCTAL       ; CONVERT TO OCTAL
2528 012164 113737 011726 015625      MOVB   DIG0, MKBQB     ; LOAD OCTAL #
2529 012172 113737 011730 015626      MOVB   DIG1, MKBQB+1
2530 012200 113737 011732 015627      MOVB   DIG2, MKBQB+2
2531 012206 042700 177600          BIC    #177600, R0
2532 012212 120027 000040          CMPB   R0, #40         ; TEST IF PRINTABLE
2533 012216 101002          BHI    10$            ; BR IF PRINTABLE
2534 012220 112700 000056          MOVB   #56, R0         ; CONVERT TO A "*" CHARACTER
2535 012224 110037 015621 10$:     MOVB   R0, MKBQ2       ; SAVE CHAR
2536 012230 011200          MOV    (R2), R0       ; GET GOOD CHAR
2537 012232 053700 012330          BIS    MASK2, R0
2538 012236 010037 001124          MOV    R0, $GDDAT     ; LOAD GOOD CHARACTER
2539 012242 004737 011636          JSR    PC, OCTAL       ; CONVERT IT

```

```

2540 012246 113737 011726 015606      MOVB  DIG0,MKBQA      ;LOAD DIGIT
2541 012254 113737 011730 015607      MOVB  DIG1,MKBQA+1
2542 012262 113737 011732 015610      MOVB  DIG2,MKBQA+2
2543 012270 042700 177600      BIC   #177600,R0
2544 012274 110037 015602      MOVB  R0,MKBQ1      ;SAVE CHAR
2545
2546 012300 023737 001276 001144      CMP   VTIS,$TKS     ;TEST IF ON CTY
2547 012306 001403                      BEQ   3$            ;BR IF YES
2548
2549 012310 004537 011616                      JSR   R5,AMSG      ;DISPLAY ERROR MESSAGE
2550 012314 015541                      MKBQ
2551 012316 104004                      3$:  ERRCR 4        ;CHARACTER RECVD NOT EQUAL TO EXPECTED
2552 012320 000610                      BR    1$            ;BR BACK AND TEST THE CHARACTER AGAIN
2553
2554 012322 000000      100$: 0
2555 012324 000000      101$: 0
2556 012326 177600      MASK1: 177600
2557 012330 000000      MASK2: 0
2558
2559                      .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2560
2561                      ;*****
2562                      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2563                      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2564                      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2565                      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2566                      ;*REPLACED WITH SPACES.
2567                      ;*CALL:
2568                      ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
2569                      ;*      TYPDS                      ;;GO TO THE ROUTINE
2570
2571                      $TYPDS:
2572                      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
2573                      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
2574                      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
2575                      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
2576                      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
2577                      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
2578                      MOV      20(SP),R5    ;;GET THE INPUT NUMBER
2579                      BPL      1$           ;;BR IF INPUT IS POS.
2580                      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
2581                      MOVB     #'-',1(SP)    ;;MAKE THE ASCII NUMBER NEG.
2582                      CLR      R0           ;;ZERO THE CONSTANTS INDEX
2583                      MOV      #SDBLK,R3    ;;SETUP THE OUTPUT POINTER
2584                      MOVB     #'',(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
2585                      CLR      R2           ;;CLEAR THE BCD NUMBER
2586                      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
2587                      SUB      R1,R5       ;;FORM THIS BCD DIGIT
2588                      BLT      4$           ;;BR IF DONE
2589                      INC      R2         ;;INCREASE THE BCD DIGIT BY 1
2590                      BR      3$
2591                      4$:  ADD      R1,R5    ;;ADD BACK THE CONSTANT
2592                      TST      R2         ;;CHECK IF BCD DIGIT=0
2593                      BNE     5$         ;;FALL THROUGH IF 0
2594                      TSTB    (SP)      ;;STILL DOING LEADING 0'S?
2595                      BMI     7$         ;;BR IF YES

```

```

2596 012430 106316          5$: ASLB (SP)          ;;MSD?
2597 012432 103003          BCC 6$          ;;BR IF NO
2598 012434 116663 000001 177777 MOVB 1(SP),-1(R3) ;;YES--SET THE SIGN
2599 012442 052702 000060     6$: BIS #'0,R2    ;;MAKE THE BCD DIGIT ASCII
2600 012446 052702 000040     7$: BIS #' ,R2    ;;MAKE IT A SPACE IF NOT-ALREADY A DIGIT
2601 012452 110223          MOVB R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2602 012454 005720          TST (R0)+       ;;JUST INCREMENTING
2603 012456 020027 000010     CMP RO,#10      ;;CHECK THE TABLE INDEX
2604 012462 002746          BLT 2$         ;;GO DO THE NEXT DIGIT
2605 012464 003002          BGT 8$         ;;GO TO EXIT
2606 012466 010502          MOV R5,R2      ;;GET THE LSD
2607 012470 000764          BR 6$         ;;GO CHANGE TO ASCII
2608 012472 105726          8$: TSTB (SP)+  ;;WAS THE LSD THE FIRST NON-ZERO?
2609 012474 100003          BPL 9$         ;;BR IF NO
2610 012476 116663 177777 177776 MOVB -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
2611 012504 105013          9$: CLRB (R3)   ;;SET THE TERMINATOR
2612 012506 012605          MOV (SP)+,R5   ;;POP STACK INTO R5
2613 012510 012603          MOV (SP)+,R3   ;;POP STACK INTO R3
2614 012512 012602          MOV (SP)+,R2   ;;POP STACK INTO R2
2615 012514 012601          MOV (SP)+,R1   ;;POP STACK INTO R1
2616 012516 012600          MOV (SP)+,R0   ;;POP STACK INTO R0
2617 012520 104401 012546     TYPE $DBLK     ;;NOW TYPE THE NUMBER
2618 012524 016666 000002 000004 MOV 2(SP),4(SP) ;;ADJUST THE STACK
2619 012532 012616          MOV (SP)+,(SP)
2620 012534 000002          RTI          ;;RETURN TO USER
2621 012536 023420          $DTBL: 10000.
2622 012540 001750          1000.
2623 012542 000144          100.
2624 012544 000012          10.
2625 012546 000004          $DBLK: .BLKW 4
2626
2627 ;SUBROUTINE TO FILL THE SCREEN WITH AN CHARACTER
2628
2629 012556 012701 000105     FILLWC: MOV #'E,R1 ;;LOAD CHARACTER BYTE
2630 012562 004737 010310     JSR PC,FILBUF ;;LOAD THE LINE WITH CHAR
2631 012568 013737 001266 012616 MOV VHO,10$ ;;LOAD COUNT
2632 012574 012737 000001 011030 1$: MOV #1,XOFFOK ;;INSURE XOFF/XON CONTROL.
2633 012602 004737 010412     JSR PC,XPRNT ;;DISPLAY THE LINE
2634 012606 005337 012616     DEC 10$
2635 012612 001370          BNE 1$        ;;LOOP UNTIL DONE
2636 012614 000207          RTS PC       ;;EXIT
2637 012616 000000          10$: 0
2638
2639 ;SUBROUTINE TO GET A CHARACTER FROM THE VT50
2640
2641 012620 013737 001254 012672 GETCHR: MOV TIME0,TIME1 ;;LOAD TIME COUNTER
2642 012626 005037 012674     CLR TIME2
2643
2644 1$: TSTB @VTIS ;;TEST INPUT STATUS
2645 012636 100005          BPL 2$        ;;BR IF CLEARED
2646 012640 017700 166434     MOV @VTIB,R0 ;;READ A CHAR
2647 012644 062716 000002     ADD #2,(SP)  ;;UPDATE RETURN
2648 012650 000207          RTS PC       ;;EXIT
2649
2650 012652 005337 012674     2$: DEC TIME2 ;;DELAY
2651 012656 001365          BNE 1$

```

2652 012660 005337 012672 DEC TIME1 ;FINISHED ?
2653 012664 100362 BPL 1\$;LOOP TILL TIME EXPIRED
2654 012666 104002 ERROR 2 ;NO INPUT FLAG FROM DEVICE
2655 012670 000207 RTS PC ;EXIT

2656
2657 012672 000000 TIME1: 0
2658 012674 000000 TIME2: 0
2659
2660

:MOVE TO THE OUTPUT BUFFER

2661
2662 012676 012500 MTOB: MOV (R5)+,R0 ;LOAD DEST.
2663 012700 012701 024614 MOV #BUFFER,R1 ;LOAD R1
2664 012704 112021 1\$: MOVB (R0)+,(R1)+ ;LOAD BYTE
2665 012706 100376 BPL 1\$;BR UNTIL DONE
2666 012710 000205 RTS R5 ;EXIT
2667

.SBTTL ASCII MESSAGES

2668
2669
2670 :ASCII MESSAGES
2671

2672 012712 006415 046412 044501 TITLE: .ASCIZ <15><15><12>\MAINDEC-11-DZVTC-C VT50A, B, VT50H AND VT52 ACCEPTANCE TEST
2673 012720 042116 041505 030455
2674 012726 026461 055104 052126
2675 012734 026503 020103 052126
2676 012742 030065 026101 041040
2677 012750 020054 052126 030065
2678 012756 020110 047101 020104
2679 012764 052126 031065 040440
2680 012772 041503 050105 040524
2681 013000 041516 020105 042524
2682 013006 052123 005015 000
2683 013013 015 020012 043040 M91: .ASCIZ <15><12>/ FULL SCREEN OF THE CHARACTER E/<15><12><377>
2684 013020 046125 020114 041523
2685 013026 042522 047105 047440
2686 013034 020106 044124 020105
2687 013042 044103 051101 041501
2688 013050 042524 020122 006505
2689 013056 177412 000
2690 013061 015 020012 042040 M92: .ASCIZ <15><12>/ DATA PATH TEST /<15><12><377>
2691 013066 052101 020101 040520
2692 013074 044124 052040 051505
2693 013102 020124 005015 000377
2694 013110 005015 020040 044523 M93: .ASCIZ <15><12>/ SINGLE CHARACTER PER LINE /<15><12><377>
2695 013116 043516 042514 041440
2696 013124 040510 040522 052103
2697 013132 051105 050040 051105
2698 013140 046040 047111 020105
2699 013146 005015 000377
2700 013152 005015 020040 047522 M94: .ASCIZ <15><12>/ ROTATING PATTERN /<15><12><377>
2701 013160 040524 044524 043516
2702 013166 050040 052101 042524
2703 013174 047122 006440 177412
2704 013202 000
2705 013203 015 020012 041440 M96: .ASCIZ <15><12>/ CURSOR MOTION TEST /<15><12><377>
2706 013210 051125 047523 020122
2707 013216 047515 044524 047117

2708	013224	052040	051505	020124		
2709	013232	005015	000377			
2710	013236	005015	020040	040524	M97:	.ASCIZ <15><12>/ TAB, BACKSPACE AND BELL TEST /<15><12><377>
2711	013244	026102	041040	041501		
2712	013252	051513	040520	042503		
2713	013260	040440	042116	041040		
2714	013266	046105	020114	042524		
2715	013274	052123	020040	005015		
2716	013302	000377				
2717	013304	005015	020040	051105	M910:	.ASCIZ <15><12>/ ERASE LINE TEST /<15><12><377>
2718	013312	051501	020105	044514		
2719	013320	042516	052040	051505		
2720	013326	020124	006440	177412		
2721	013334	000				
2722	013335	015	020012	042440	M911:	.ASCIZ <15><12>/ ERASE SCREEN TEST /<15><12><377>
2723	013342	040522	042523	051440		
2724	013350	051103	042505	020116		
2725	013356	042524	052123	020040		
2726	013364	005015	000377			
2727	013370	005015	020040	044526	M912:	.ASCIZ <15><12>/ VIDEO COUPLING TEST /<15><12><377>
2728	013376	042504	020117	047503		
2729	013404	050125	044514	043516		
2730	013412	052040	051505	020124		
2731	013420	005015	000377			
2732	013424	044033	045033	005015	M914:	.ASCIZ <33><110><33><112><15><12>/ TERMINAL IDENTIFIER TEST /<15><12><377>
2733	013432	020040	042524	046522		
2734	013440	047111	046101	044440		
2735	013446	042504	052116	043111		
2736	013454	042511	020122	042524		
2737	013462	052123	006440	177412		
2738	013470	000				
2739	013471	015	020012	041440	M920:	.ASCIZ <15><12>/ COPIER PERIMETER /<15><12><377>
2740	013476	050117	042511	020122		
2741	013504	042520	044522	042515		
2742	013512	042524	006522	177412		
2743	013520	000				
2744	013521	015	020012	042040	M921:	.ASCIZ <15><12>/ DISCLAIMER STATEMENT /<15><12><377>
2745	013526	051511	046103	044501		
2746	013534	042515	020122	052123		
2747	013542	052101	046505	047105		
2748	013550	006524	177412	000		
2749	013555	015	020012	044040	M922:	.ASCIZ <15><12>/ HOLD SCREEN MODE TEST /<15><12><377>
2750	013562	046117	020104	041523		
2751	013570	042522	047105	046440		
2752	013576	042117	020105	042524		
2753	013604	052123	005015	000377		
2754	013612	005015	020040	051107	M9221:	.ASCIZ <15><12>/ GRAPHICS MODE AND REV. LINE FEED TEST /<15><12><377>
2755	013620	050101	044510	051503		
2756	013626	046440	042117	020105		
2757	013634	047101	020104	042522		
2758	013642	027126	046040	047111		
2759	013650	020105	042506	042105		
2760	013656	052040	051505	006524		
2761	013664	177412	000			
2762	013667	033	015537	015510	M923:	.ASCIZ <33><137><33><110><33><112>/ AUTO COPY MODE TEST /<15><12><377>
2763	013674	020112	040440	052125		

2764	013702	020117	047503	054520	
2765	013710	046440	042117	020105	
2766	013716	042524	052123	005015	
2767	013724	000377			
2768	013726	044033	045033	051120	MPTCNT: .ASCIZ <33><110><33><112>/PRINTER CONTROLLER MODE ENTERED/<15><12><377>
2769	013734	047111	042524	020122	
2770	013742	047503	052116	047522	
2771	013750	046114	051105	046440	
2772	013756	042117	020105	047105	
2773	013764	042524	042522	006504	
2774	013772	177412	000		
2775	013775	033	015510	050112	MPTSCN: .ASCIZ <33><110><33><112>/PRINT A SCREEN OF E'S/<15><12><377>
2776	014002	044522	052116	040440	
2777	014010	051440	051103	042505	
2778	014016	020116	043117	042440	
2779	014024	051447	005015	000377	
2780	014032	015510	020112	040440	M923A: .ASCIZ <110><33><112>/ AUTO PRINT MODE TEST/<15><12><377>
2781	014040	052125	020117	051120	
2782	014046	047111	020124	047515	
2783	014054	042504	052040	051505	
2784	014062	006524	177412	000	
2785	014067	015	020012	042513	MKB: .ASCII <15><12>/ KEYBOARD CHARACTER TEST/<15><12>
2786	014074	041131	040517	042122	
2787	014102	041440	040510	040522	
2788	014110	052103	051105	052040	
2789	014116	051505	006524	012	
2790	014123	122	046105	040505	MKB1: .ASCIZ /RELEASE THE "SHIFT" KEY/<15><12><377>
2791	014130	042523	052040	042510	
2792	014136	021040	044123	043111	
2793	014144	021124	045440	054505	
2794	014152	005015	000377		
2795	014156	005015	042514	052106	MKBA: .ASCIZ <15><12>/LEFT TO RIGHT IN A ROW, DEPRESS ONE KEY AT A TIME/<15><12><377>
2796	014164	052040	020117	044522	
2797	014172	044107	020124	047111	
2798	014200	040440	051040	053517	
2799	014206	020054	042504	051120	
2800	014214	051505	020123	047117	
2801	014222	020105	042513	020131	
2802	014230	052101	040440	052040	
2803	014236	046511	006505	177412	
2804	014244	000			
2805	014245	015	051412	040524	MKBB: .ASCII <15><12>/STARTING WITH THE TOP ROW EXCEPT THE THIRD/
2806	014252	052122	047111	020107	
2807	014260	044527	044124	052040	
2808	014266	042510	052040	050117	
2809	014274	051040	053517	042440	
2810	014302	041530	050105	020124	
2811	014310	044124	020105	044124	
2812	014316	051111	104		
2813	014321	040	051106	046517	.ASCIZ / FROM RIGHT END AND LAST KEYS/<15><12><377>
2814	014326	051040	043511	052110	
2815	014334	042440	042116	040440	
2816	014342	042116	046040	051501	
2817	014350	020124	042513	051531	
2818	014356	005015	000377		
2819	014362	005015	052123	051101	MKBB2: .ASCIZ <15><12>/STARTING WITH THE TOP ROW EXCEPT THE LAST KEY/<15><12><377>

2820	014370	044524	043516	053440
2821	014376	052111	020110	044124
2822	014404	020105	047524	020120
2823	014412	047522	020127	054105
2824	014420	042503	052120	052040
2825	014426	042510	046040	051501
2826	014434	020124	042513	006531
2827	014442	177412	000	
2828	014445	015	051412	040524
2829	014452	052122	053440	052111
2830	014460	020110	044124	020105
2831	014466	042523	047503	042116
2832	014474	051040	053517	005015
2833	014502	000377		
2834	014504	005015	052123	051101
2835	014512	020124	044527	044124
2836	014520	052040	042510	052040
2837	014526	044510	042122	051040
2838	014534	053517	042440	041530
2839	014542	050105	020124	044124
2840	014550	020105	052103	046122
2841	014556	005015	040440	042116
2842	014564	052040	042510	021040
2843	014572	046102	047101	021113
2844	014600	045440	054505	006523
2845	014606	177412	000	
2846	014611	015	051412	040524
2847	014616	052122	053440	052111
2848	014624	020110	044124	020105
2849	014632	044124	051111	020104
2850	014640	047522	020127	041054
2851	014646	043505	047111	051040
2852	014654	053517	053440	052111
2853	014662	020110	040442	020042
2854	014670	042513	006531	177412
2855	014676	000		
2856	014677	015	051412	040524
2857	014704	052122	053440	052111
2858	014712	020110	044124	020105
2859	014720	047506	051125	044124
2860	014726	051040	053517	042440
2861	014734	041530	050105	020124
2862	014742	044124	020105	041523
2863	014750	047522	046114	
2864	014754	005015	051454	044510
2865	014762	052106	020054	042522
2866	014770	042520	052101	040440
2867	014776	042116	040440	052125
2868	015004	026517	051120	047111
2869	015012	006524	177412	000
2870	015017	015	051412	040524
2871	015024	052122	053440	052111
2872	015032	020110	044124	020105
2873	015040	044506	052106	020110
2874	015046	047522	006527	177412
2875	015054	000		

MKBC: .ASCIZ <15><12>/START WITH THE SECOND ROW/<15><12><377>

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>

2876	015055	015	047012	053517	MKBG:	.ASCII	<15><12>/NOW HOLD DOWN THE "LEFT-SHIFT" KEY/<15><12><377>
2877	015062	044040	046117	020104			
2878	015070	047504	047127	052040			
2879	015076	042510	021040	042514			
2880	015104	052106	051455	044510			
2881	015112	052106	020042	042513			
2882	015120	006531	177412				
2883	015124	005015	047516	020127	MKBGA:	.ASCII	<15><12>/NOW HOLD DOWN THE "RIGHT-SHIFT" KEY/<15><12><377>
2884	015132	047510	042114	042040			
2885	015140	053517	020116	044124			
2886	015146	020105	051042	043511			
2887	015154	052110	051455	044510			
2888	015162	052106	020042	042513			
2889	015170	006531	177412				
2890	015174	005015	047516	020127	MKBH:	.ASCII	<15><12>/NOW HOLD DOWN THE "CTRL" KEY/<15><12><377>
2891	015202	047510	042114	042040			
2892	015210	053517	020116	044124			
2893	015216	020105	041442	051124			
2894	015224	021114	045440	054505			
2895	015232	005015	377				
2896	015235	015	051412	040524	MKBI:	.ASCII	<15><12>/START WITH THE TOP ROW/<15><12><377>
2897	015242	052122	053440	052111			
2898	015250	020110	044124	020105			
2899	015256	047524	020120	047522			
2900	015264	006527	177412				
2901	015270	005015	052123	051101	MKBJ:	.ASCII	<15><12>/START WITH THE 2ND ROW/<15><12><377>
2902	015276	020124	044527	044124			
2903	015304	052040	042510	031040			
2904	015312	042116	051040	053517			
2905	015320	005015	377				
2906	015323	015	051412	040524	MKBK:	.ASCII	<15><12>/START WITH THE 3RD ROW/<15><12><377>
2907	015330	052122	020040	044527			
2908	015336	044124	052040	042510			
2909	015344	031440	042122	051040			
2910	015352	053517	005015	377			
2911	015357	015	051412	040524	MKBL:	.ASCII	<15><12>/START WITH THE 4TH ROW/<15><12><377>
2912	015364	052122	053440	052111			
2913	015372	020110	044124	020105			
2914	015400	052064	020110	047522			
2915	015406	006527	177412				
2916	015412	005015	052123	051101	MKBM:	.ASCII	<15><12>/START WITH THE LAST ROW/<15><12><377>
2917	015420	020124	044527	044124			
2918	015426	052040	042510	046040			
2919	015434	051501	020124	047522			
2920	015442	006527	177412				
2921	015446	005015	053012	032524	MKBN:	.ASCII	<15><12><12>/VT50H KEYPAD TEST/<15><12><377>
2922	015454	044060	045440	054505			
2923	015462	040520	020104	042524			
2924	015470	052123	005015	377			
2925	015475	015	053012	032524	MK852:	.ASCII	<15><12>/VT52 ALTERNATE KEYPAD MODE TEST/<15><12><377>
2926	015502	020062	046101	042524			
2927	015510	047122	052101	020105			
2928	015516	042513	050131	042101			
2929	015524	046440	042117	020105			
2930	015532	042524	052123	005015			
2931	015540	377					

2932	015541	015	041412	040510	MKBQ:	.ASCII	<15><12>/CHARACTER WAS IN ERROR//15><12>
2933	015546	040522	052103	051105			
2934	015554	053440	051501	044440			
2935	015562	020116	051105	047522			
2936	015570	006522	012				
2937	015573	107	047517	020104		.ASCII	/GOOD = /
2938	015600	020075					
2939	015602	040	040	075	MKBQ1:	.BYTE	40,40,75,40
2940	015605	040					
2941	015606	040	040	040	MKBQA:	.BYTE	40,40,40,40,40
2942	015611	040	040				
2943	015613	102	042101	036440		.ASCII	/BAD = /
2944	015620	040					
2945	015621	040	040	075	MKBQ2:	.BYTE	40,40,75,40
2946	015624	040					
2947	015625	040	040	040	MKBQB:	.BYTE	40,40,40,15,12,377,0
2948	015630	015	012	377			
2949	015633	000					
2950	015634	005015	042513	041131	MKBR:	.ASCIZ	<15><12>/KEYBOARD CHARACTER TEST COMPLETE/<15><12><377>
2951	015642	040517	042122	041440			
2952	015650	040510	040522	052103			
2953	015656	051105	052040	051505			
2954	015664	020124	047503	050115			
2955	015672	042514	042524	005015			
2956	015700	000377					
2957							
2958	015702	005015	045440	054505	MKE:	.ASCII	<15><12>/KEYBOARD ASCII AND OCTAL LOOP/<15><12>
2959	015710	047502	051101	020104			
2960	015716	051501	044503	020111			
2961	015724	047101	020104	041517			
2962	015732	040524	020114	047514			
2963	015740	050117	005015				
2964	015744	015	012			.BYTE	15,12
2965	015746	044127	047105	040440		.ASCII	/WHEN A KEY IS DEPRESSED. THE ASCII CHARACTER AND/
2966	015754	045440	054505	044440			
2967	015762	020123	042504	051120			
2968	015770	051505	042523	026104			
2969	015776	052040	042510	040440			
2970	016004	041523	044511	041440			
2971	016012	040510	040522	052103			
2972	016020	051105	040440	042116			
2973	016026	015	012			.BYTE	15,12
2974	016030	052040	042510	052040		.ASCIZ	/ THE THREE DIGIT OCTAL CODE WILL BE ECHOED/
2975	016036	051110	042505	042040			
2976	016044	043511	052111	047440			
2977	016052	052103	046101	041440			
2978	016060	042117	020105	044527			
2979	016066	046114	041040	020105			
2980	016074	041505	047510	042105			
2981	016102	000					
2982	016103	015	012	377		.BYTE	15,12,377,0
2983	016106	000					
2984	016107	015	041412	040510	MKEA:	.ASCII	<15><12>/CHAR = /
2985	016114	020122	020075				
2986	016120	040	040	040	MKEA1:	.BYTE	40,40,40,75,40
2987	016123	075	040				

H05

MAINDEC-11-DZVTC-D MACY11 27(1006) 21-FEB-77 15:32 PAGE 58
DZVTC.D.P11 21-FEB-77 15:27 ASCII MESSAGES

2988	016125	040	040	040	MKEB:	.BYTE	40,40,40,15,12,377,0
2989	016130	015	012	377			
2990	016133	000					
2991	016134	005015	042513	041131	MKEH:	.ASCIZ	<15><12>/KEYBOARD ECHO LOOP/<15><12>
2992	016142	040517	042122	042440			
2993	016150	044103	020117	047514			
2994	016156	050117	005015	000			
2995	016163	015	012	012	CRLFB:	.BYTE	15,12,12,12,12,12,12,12,12
2996	016166	012	012	012			
2997	016171	012	012	012			
2998	016174	012	012	012	CRLFA:	.BYTE	12,12,12,12,12,377,0
2999	016177	012	012	377			
3000	016202	000					
3001							:HOME AND ERASE SCREEN OPCODE
3002	016203	033	110	033	HOMERS:	.BYTE	33,110,33,112,377,0
3003	016206	112	377	000			
3004	016211	015	052012	040040	PATH:	.ASCII	'15'<12>/T ~~~~~~T'
3005	016216	040100	040100	040100			
3006	016224	040100	040100	040100			
3007	016232	040100	040100	040100			
3008	016240	040100	040100	040100			
3009	016246	040100	040100	040100			
3010	016254	040100	040100	040100			
3011	016262	100					
3012	016263	100	040100	040100	.ASCII	/~~~~~T'	
3013	016270	040100	040100	040100			
3014	016276	040100	040100	040100			
3015	016304	040100	040100	040100			
3016	016312	040100	040100	040100			
3017	016320	040100	040100	040100			
3018	016326	040100	020100	124			
3019	016333	015	052012	042440	.ASCII	<15><12>/T EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE'	
3020	016340	042505	042505	042505			
3021	016346	042505	042505	042505			
3022	016354	042505	042505	042505			
3023	016362	042505	042505	042505			
3024	016370	042505	042505	042505			
3025	016376	042505	042505	042505			
3026	016404	105					
3027	016405	105	042505	042505	.ASCII	/EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE'	
3028	016412	042505	042505	042505			
3029	016420	042505	042505	042505			
3030	016426	042505	042505	042505			
3031	016434	042505	042505	042505			
3032	016442	042505	042505	042505			
3033	016450	042505	020105	124			
3034	016455	377	000	000	VT50A:	.BYTE	377,0,0
3035	016460	005015	042524	052123	.ASCII	<15><12>/TESTING VT50A (12 LINES-COPIER)'	
3036	016466	047111	020107	052126			
3037	016474	030065	020101	030450			
3038	016502	020062	044514	042516			
3039	016510	026523	047516	041440			
3040	016516	050117	042511	024522			
3041	016524	015	012	012	.BYTE	15,12,12,377,0,0	
3042	016527	377	000	000			
3043	016532	005015	042524	052123	VT50B:	.ASCII	<15><12>/TESTING VT50B (12 LINES-COPIER)'

MAINDEC-11-DZVTC-D
DZVTC.P11 21-FEB-77

MACY11 27(1006)
15:27

21-FEB-77 15:32 PAGE 59
ASCII MESSAGES

3044	016540	047111	020107	052126
3045	016546	030065	020102	030450
3046	016554	020062	044514	042516
3047	016562	026523	047503	044520
3048	016570	051105	051	
3049	016573	015	012	012
3050	016576	377	000	000
3051	016601	015	052012	051505
3052	016606	044524	043516	053040
3053	016614	032524	020065	031050
3054	016622	020064	044514	042516
3055	016630	026523	047503	044520
3056	016636	051105	020054	052126
3057	016644	030065	041040	051501
3058	016652	042105	051	
3059	016655	015	012	012
3060	016660	377	000	000
3061	016663	015	052012	051505
3062	016670	044524	043516	053040
3063	016676	032524	044060	024040
3064	016704	031061	046040	047111
3065	016712	051505	041455	050117
3066	016720	042511	026522	027104
3067	016726	027103	027101	051
3068	016733	015	012	012
3069	016736	377	000	000
3070	016741	015	052012	051505
3071	016746	044524	043516	053040
3072	016754	032524	020062	031050
3073	016762	020064	044514	042516
3074	016770	026523	047516	041440
3075	016776	050117	042511	020122
3076	017004	051117	050040	044522
3077	017012	052116	051105	
3078	017016	015	012	012
3079	017021	377	000	000
3080	017024	005015	042524	052123
3081	017032	047111	020107	052126
3082	017040	031065	024040	032062
3083	017046	046040	047111	051505
3084	017054	041454	050117	042511
3085	017062	026522	047516	050040
3086	017070	044522	052116	051105
3087	017076	051		
3088	017077	015	012	012
3089	017102	377	000	000
3090	017105	015	052012	051505
3091	017112	044524	043516	053040
3092	017120	032524	020062	031050
3093	017126	020064	044514	042516
3094	017134	026123	051120	047111
3095	017142	042524	026522	047516
3096	017150	041440	050117	042511
3097	017156	024522		
3098	017160	015	012	012
3099	017163	377	000	000

.BYTE 15,12,12,377,0,0
VT55: .ASCII <15><12>/TESTING VT55 (24 LINES-COPIER, VT50 BASED)/

.BYTE 15,12,12,377,0,0
VT50H: .ASCII <15><12>/TESTING VT50H (12 LINES-COPIER-D.C.A.)

.BYTE 15,12,12,377,0,0
VT52K: .ASCII <15><12>/TESTING VT52 (24 LINES-NO COPIER OR PRINTER)

.BYTE 15,12,12,377,0,0
VT52L: .ASCII <15><12>/TESTING VT52 (24 LINES,COPIER-NO PRINTER)/

.BYTE 15,12,12,377,0,0
VT52M: .ASCII <15><12>/TESTING VT52 (24 LINES,PRINTER-NO COPIER)/

.BYTE 15,12,12,377,0,0

K05

3156	017575	000	000	000	
3157	017600	000	000	000	
3158	017603	377	000	000	
3159	017606	000	000		
3160	017610	044033	045033	020040	M98: .ASCIZ <33><110><33><112>/ DIRECT CURSOR ADDRESSING TEST/'377'
3161	017616	042040	051111	041505	
3162	017624	020124	052503	051522	
3163	017632	051117	040440	042104	
3164	017640	042522	051523	047111	
3165	017646	020107	042524	052123	
3166	017654	000377			
3167	017656	005015	043012	051111	WHAT0: .ASCIZ <15><12><12>/FIRST LINE DEVICE ADDRESS ? = /<200>
3168	017664	052123	046040	047111	
3169	017672	020105	042504	044526	
3170	017700	042503	040440	042104	
3171	017706	042522	051523	037440	
3172	017714	036440	100040	000	
3173	017721	015	005012	040514	WHAT1: .ASCIZ <15><12><12>/LAST LINE DEVICE ADDRESS (CR IF NONE) ? = /<200>
3174	017726	052123	046040	047111	
3175	017734	020105	042504	044526	
3176	017742	042503	040440	042104	
3177	017750	042522	051523	024040	
3178	017756	051103	044440	020106	
3179	017764	047516	042516	020051	
3180	017772	020077	020075	000200	
3181	020000	005015	020040	020040	PASHED: .ASCIZ <15><12>/ PASS # # OF ERRORS /
3182	020006	020040	020040	020040	
3183	020014	040520	051523	021440	
3184	020022	020040	020040	020043	
3185	020030	043117	042440	051122	
3186	020036	051117	020123	020040	
3187	020044	000			
3188	020045	105	051122	051117	EM1: .ASCIZ /ERROR FLAG SET ON TRANSMITTER STATUS/
3189	020052	043040	040514	020107	
3190	020060	042523	020124	047117	
3191	020066	052040	040522	051516	
3192	020074	044515	052124	051105	
3193	020102	051440	040524	052524	
3194	020110	000123			
3195	020112	047516	044440	050116	EM2: .ASCIZ /NO INPUT FLAG DETECTED/
3196	020120	052125	043040	040514	
3197	020126	020107	042504	042524	
3198	020134	052103	042105	000	
3199	020141	111	041516	051117	EM3: .ASCIZ /INCORRECT I.D. CODE/
3200	020146	042522	052103	044440	
3201	020154	042056	020056	047503	
3202	020162	042504	000		
3203	020165	125	042516	050130	EM4: .ASCIZ /UNEXPECTED OR INCORRECT INPUT CHAR/
3204	020172	041505	042524	020104	
3205	020200	051117	044440	041516	
3206	020206	051117	042522	052103	
3207	020214	044440	050116	052125	
3208	020222	041440	040510	000122	
3209	020230	047111	040526	044514	EM5: .ASCIZ /INVALID BUSS ADDRESS, TRY AGAIN/
3210	020236	020104	052502	051523	
3211	020244	040440	042104	042522	

3212	020252	051523	020054	051124						
3213	020260	020131	043501	044501						
3214	020266	000116								
3215	020270	051105	050122	020103	DH1:	.ASCIZ	/ERRPC	VTNOW	TSTNUM/	
3216	020276	020040	052126	047516						
3217	020304	020127	020040	051524						
3218	020312	047124	046525	000						
3219	020317	105	051122	041520	DH3:	.ASCIZ	/ERRPC	VTNOW	1ST WD 2ND WD 3RD WD/	
3220	020324	020040	053040	047124						
3221	020332	053517	020040	030440						
3222	020340	052123	053440	020104						
3223	020346	031040	042116	053440						
3224	020354	020104	031440	042122						
3225	020362	053440	000104							
3226	020366	051105	050122	020103	DH4:	.ASCIZ	/ERRPC	VTNOW	TSTNUM EXPCT RECV/	
3227	020374	020040	052126	047516						
3228	020402	020127	020040	051524						
3229	020410	047124	046525	020040						
3230	020416	042440	050130	052103						
3231	020424	020040	051040	041505						
3232	020432	000126								
3233										
3234	020434	001116	001246	001252	DT1:	.EVEN	\$ERRPC,VTNOW,TSTNUM,0			
3235	020442	000000								
3236	020444	001116	001246	001330	DT3:	\$ERRPC,VTNOW,SAVE4,SAVE2,SAVE3,0				
3237	020452	001324	001326	000000						
3238	020460	001116	001246	001252	DT4:	\$ERRPC,VTNOW,TSTNUM,\$GDDAT,\$BDDAT,0				
3239	020466	001124	001126	000000						
3240										
3241										
3242										
3243										
3244										
3245	020474	020530	020564	020622	V50RW:	ROW1,ROW2,ROW3,ROW4,ROW15,ROW15,ROW1C				
3246	020502	020652	021052	021052						
3247	020510	021144								
3248	020512	020700	020736	020774	V52RW:	ROW12,ROW22,ROW32,ROW42,ROW125,ROW125,ROW12C				
3249	020520	021026	021106	021106						
3250	020526	021200								
3251										
3252										
3253	020530	000033	000061	000062	ROW1:	.WORD	33,61,62,63,64,65,66,67,70,71,60,55,75,10001C			
3254	020536	000063	000064	000065						
3255	020544	000066	000067	000070						
3256	020552	000071	000060	000055						
3257	020560	000075	100010							
3258	020564	000011	000121	000127	ROW2:	.WORD	11,121,127,105,122,124,131,125,111,117,120,133,134,12,100177			
3259	020572	000105	000122	000124						
3260	020600	000131	000125	000111						
3261	020606	000117	000120	000133						
3262	020614	000134	000012	100177						
3263	020622	000101	000123	000104	ROW3:	.WORD	101,123,104,106,107,110,112,113,114,73,47,100015			
3264	020630	000106	000107	000110						
3265	020636	000112	000113	000114						
3266	020644	000073	000047	100015						
3267	020652	000132	000130	000103	ROW4:	.WORD	132,130,103,126,102,116,115,54,56,100057			

3268	020660	000126	000102	000116
3269	020666	000115	000054	000056
3270	020674	100057		
3271	020676	100040		
3272				
3273				
3274	020700	000033	000061	000062
3275	020706	000063	000064	000065
3276	020714	000066	000067	000070
3277	020722	000071	000060	000055
3278	020730	000075	000140	100010
3279	020736	000011	000161	000167
3280	020744	000145	000162	000164
3281	020752	000171	000165	000151
3282	020760	000157	000160	000133
3283	020766	000134	000012	100177
3284	020774	000141	000163	000144
3285	021002	000146	000147	000150
3286	021010	000152	000153	000154
3287	021016	000073	000047	000173
3288	021024	100015		
3289	021026	000172	000170	000143
3290	021034	000166	000142	000156
3291	021042	000155	000054	000056
3292	021050	100057		
3293				
3294				
3295				
3296	021052	000033	000041	000100
3297	021060	000043	000044	000045
3298	021066	000136	000046	000052
3299	021074	000050	000051	000137
3300	021102	000053	100010	
3301	021106	000033	000041	000100
3302	021114	000043	000044	000045
3303	021122	000136	000046	000052
3304	021130	000050	000051	000137
3305	021136	000053	000176	100010
3306				
3307				
3308				
3309	021144	000033	000021	000022
3310	021152	000023	000024	000025
3311	021160	000026	000027	000030
3312	021166	000031	000020	000015
3313	021174	000035	100010	
3314	021200	000033	000021	000022
3315	021206	000023	000024	000025
3316	021214	000026	000027	000030
3317	021222	000031	000020	000015
3318	021230	000035	000000	100010
3319				
3320				
3321	021236	033	111	015
3322	021241	377	000	
3323				

ROWS: .WORD 100040
;VT52 KEYBOARD EQUIVALENCES(LOWER CASE CHAR.)

ROW12: .WORD 33,61,62,63,64,65,66,67,70,71,60,55,75,140,100010

ROW22: .WORD 11,161,167,145,162,164,171,165,151,157,160,133,134,12,100177

ROW32: .WORD 141,163,144,146,147,150,152,153,154,73,47,173,100015

ROW42: .WORD 172,170,143,166,142,156,155,54,56,100057

;SHIFTED ROW CODES

ROW1S: .WORD 33,41,100,43,44,45,136,46,52,50,51,137,53,100010

ROW12S: .WORD 33,41,100,43,44,45,136,46,52,50,51,137,53,176,100010

;CONTRCL ROW CODES

ROW1C: .WORD 33,21,22,23,24,25,26,27,30,31,20,15,35,100010

ROW12C: .WORD 33,21,22,23,24,25,26,27,30,31,20,15,35,0,100010

;VT52 ESCAPE SEQUENCES

REVLf: .BYTE 33,111,0,5,377,0 :REVERSE LINE FEED.

3324	021243	033	075	377	ENAKP: .BYTE	33,075,377,0,0	;ENABLE ALTERNATE KEYPAD MODE.
3325	021246	000	000				
3326	021250	033	076	377	EXAKP: .BYTE	33,076,377,0,0	;EXIT ALTERNATE KEYPAD MODE
3327	021253	000	000				
3329							
3329	021255	033	106	000	ENGRAF: .BYTE	33,106,0,377,0	;ENTER GRAPHICS MODE.
3330	021260	377	000				
3331	021262	033	107	000	EXGRAF: .BYTE	33,107,0,377,0	;EXIT GRAPHICS MODE.
3332	021265	377	000				
3333							
3334	021267	033	127	377	ENPNTM: .BYTE	33,127,377,0,0	;ENABLE PRINTER CONTROLLER MODE.
3335	021272	000	000				
3336	021274	033	130	377	EXPNTM: .BYTE	33,130,377,0,0	;DISABLE PRINTER CONTROLLER MODE.
3337	021277	000	000				

;VTSO-H KEYPAD CODES

3342	021302	000033	000120	000033	ROW6: .EVEN		
3343	021302	000033	000120	000033	ROW6: .WORD	33,'P,33,'Q,33,'R,33,100101	
3344	021310	000121	000033	000122			
3345	021316	000033	100101				
3346	021322	000067	000070	000071	ROW7: .WORD	67,70,71,33,100102	
3347	021330	000033	100102				
3348	021334	000064	000065	000066	ROW8: .WORD	64,65,66,33,100103	
3349	021342	000033	100103				
3350	021346	000061	000062	000063	ROW9: .WORD	61,62,63,33,100104	
3351	021354	000033	100104				
3352	021360	000060	000056	100015	ROW10: .WORD	60,56,100015	
3353	021366	000033	000120	000033	ROW6A: .WORD	33,'P,33,'Q,33,'R,33,100101	
3354	021374	000121	000033	000122			
3355	021402	000033	100101				
3356	021406	000033	000077	000167	ROW7A: .WORD	33,'?',167,33,'?',170,33,'?',171,33,100102	
3357	021414	000033	000077	000170			
3358	021422	000033	000077	000171			
3359	021430	000033	100102				
3360	021434	000033	000077	000164	ROW8A: .WORD	33,'?',164,33,'?',165,33,'?',166,33,100103	
3361	021442	000033	000077	000165			
3362	021450	000033	000077	000166			
3363	021456	000033	100103				
3364	021462	000033	000077	000161	ROW9A: .WORD	33,'?',161,33,'?',162,33,'?',163,33,100104	
3365	021470	000033	000077	000162			
3366	021476	000033	000077	000163			
3367	021504	000033	100104				
3368	021510	000033	000077	000160	ROW10A: .WORD	33,'?',160,33,'?',156,33,'?',100115	
3369	021516	000033	000077	000156			
3370	021524	000033	000077	100115			
3371	021532	005015	052012	044510	MTEXT0: .ASCIZ	<15><12><12>/THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR	
3372	021540	020123	047523	052106			
3373	021546	040527	042522	044440			
3374	021554	020123	052506	047122			
3375	021562	051511	042510	020104			
3376	021570	047524	050040	051125			
3377	021576	044103	051501	051105			
3378	021604	052440	042116	051105			
3379	021612	040440	046040	041511			

3380	021620	047105	042523	043040
3381	021626	051117	052440	042523
3382	021634	000377		
3383	021636	005015	047117	040440
3384	021644	051440	047111	046107
3385	021652	020105	047503	050115
3386	021660	052125	051105	051440
3387	021666	051531	042524	020115
3388	021674	047101	020104	040503
3389	021702	020116	042502	041440
3390	021710	05J117	042511	020104
3391	021716	053450	052111	020110
3392	021724	047111	046103	051525
3393	021732	047511	177516	000
3394	021737	015	047412	020106
3395	021744	042504	023503	020123
3396	021752	047503	054520	044522
3397	021760	044107	020124	047516
3398	021766	044524	042503	020051
3399	021774	047117	054514	043040
3400	022002	051117	052440	042523
3401	022010	044440	020116	052523
3402	022016	044103	051440	051531
3403	022024	042524	026115	042440
3404	022032	041530	050105	177524
3405	022040	000		
3406	022041	015	040412	020123
3407	022046	040515	020131	052117
3408	022054	042510	053522	051511
3409	022062	020105	042502	050040
3410	022070	047522	044526	042504
3411	022076	020104	047111	053440
3412	022104	044522	044524	043516
3413	022112	041040	020131	042504
3414	022120	027103	000377	
3415				
3416	022124	005015	052012	042510
3417	022132	044440	043116	051117
3418	022140	040515	044524	047117
3419	022146	044440	020116	044124
3420	022154	051511	042040	041517
3421	022162	046525	047105	020124
3422	022170	051511	051440	041125
3423	022176	042512	052103	052040
3424	022204	020117	044103	047101
3425	022212	042507	053440	052111
3426	022220	047510	052125	000377
3427	022226	005015	047516	044524
3428	022234	042503	040440	042116
3429	022242	051440	047510	046125
3430	022250	020104	047516	020124
3431	022256	042502	041440	047117
3432	022264	052123	052522	042105
3433	022272	040440	020123	020101
3434	022300	047503	046515	052111
3435	022306	042515	052116	041040

MTEXT1: .ASCIZ <15><12>/ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION/<

MTEXT2: .ASCIZ <15><12>/OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT/

MTEXT3: .ASCIZ <15><12>/AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC./<377>

MTEXT4: .ASCIZ <15><12><12>/THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHO

MTEXT5: .ASCIZ <15><12>/NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL/<

3436	022314	020131	044504	044507
3437	022322	040524	177514	000
3438	022327	015	042412	052521
3439	022334	050111	042515	052116
3440	022342	041440	051117	047520
3441	022350	040522	044524	047117
3442	022356	177456	000	
3443	022361	015	005012	042504
3444	022366	020103	051501	052523
3445	022374	042515	020123	047516
3446	022402	051040	051505	047520
3447	022410	051516	041111	046111
3448	022416	052111	020131	047506
3449	022424	020122	044124	020105
3450	022432	051525	020105	051117
3451	022440	051040	046105	040511
3452	022446	044502	044514	054524
3453	022454	047440	177506	000
3454	022461	015	044412	051524
3455	022466	051440	043117	053524
3456	022474	051101	020105	047117
3457	022502	042440	052521	050111
3458	022510	042515	052116	053440
3459	022516	044510	044103	044440
3460	022524	020123	047516	020124
3461	022532	052523	050120	044514
3462	022540	042105	041040	020131
3463	022546	042504	027103	
3464	022552	015	012	377
3465	022555	000		
3466				
3467				
3468				
3469				
3470				
3471				
3472				
3473				
3474				
3475				
3476				
3477				
3478				
3479				
3480				
3481				
3482				
3483	022556	011646		
3484	022560	016666	000004	000002
3485	022566	105777	156352	
3486	022572	100375		
3487	022574	117766	156346	000004
3488	022602	042756	177600	000004
3489	022610	026627	000004	000023
3490	022616	001013		
3491	022620	105777	156320	

MTEXT6: .ASCIZ <15><12>/EQUIPMENT CORPORATION./<377>

MTEXT7: .ASCIZ <15><12><12>/DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF

MTEXT8: .ASCII <15><12>/ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC./

.BYTE 15,12,377,0

.EVEN
.SBTTL TTY INPUT ROUTINE

.ENABL LSB
.DSABL LSB

;; THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:
* RDCHP ;; INPUT A SINGLE CHARACTER FROM THE TTY
* RETURN HERE ;; CHARACTER IS ON THE STACK
* ;; WITH PARITY BIT STRIPPED OFF

```

$RDCHR: MOV (SP), -(SP) ;; PUSH DOWN THE PC
MOV 4(SP), 2(SP) ;; SAVE THE PS
1$: TST @STKS ;; WAIT FOR
BPL 1$ ;; A CHARACTER
MOV @STKB, 4(SP) ;; READ THE TTY
BIC #1C<177>, 4(SP) ;; GET RID OF JUNK IF ANY
CMP 4(SP), #23 ;; IS IT A CONTROL-S?
BNE 3$ ;; BRANCH IF NO
2$: *STB @STKS ;; WAIT FOR A CHARACTER

```

```

3492 022624 100375          BPL      2$          ;; LOOP UNTIL ITS THERE
3493 022626 117746 156314  MOVB     2$TKB, -(SP) ;; GET CHARACTER
3494 022632 042716 177600  BIC     #1C177, (SP) ;; MAKE IT 7-BIT ASCII
3495 022636 022627 000021  CMP     (SP)+, #21    ;; IS IT A CONTROL-Q?
3496 022642 001366          BNE     2$          ;; IF NOT DISCARD IT
3497 022644 000750          BR      1$          ;; YES, RESUME
3498 022646 026627 000004 000140 3$:    CMP     4(SP), #140  ;; IS IT UPPER CASE?
3499 022654 002407          BLT     4$          ;; BRANCH IF YES
3500 022656 026627 000004 000175  CMP     4(SP), #175  ;; IS IT A SPECIAL CHAR?
3501 022664 003003          BGT     4$          ;; BRANCH IF YES
3502 022666 042766 000040 000004  BIC     #40, 4(SP)   ;; MAKE IT UPPER CASE
3503 022674 000002          4$:    RTI          ;; GO BACK TO USER
3504          ;; *****
3505          ;; *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3506          ;; *CALL:
3507          ;; *      RDLIN          ;; INPUT A STRING FROM THE TTY
3508          ;; *      RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3509          ;; *          ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
3510
3511 022676 010346          $RDLIN: MOV     R3, -(SP) ;; SAVE R3
3512 022700 012703 023004  1$:    MOV     #STTYIN, R3 ;; GET ADDRESS
3513 022704 022703 023014  2$:    CMP     #STTYIN+8., R3 ;; BUFFER FULL?
3514 022710 101405          BLOS    4$          ;; BR IF YES
3515 022712 104406          RDCHR   ;; GO READ ONE CHARACTER FROM THE TTY
3516 022714 112613          MOVB   (SP)+, (R3)  ;; GET CHARACTER
3517 022716 122713 000177  10$:  CMPB   #177, (R3)  ;; IS IT A RUBOUT
3518 022722 001003          BNE     3$          ;; SKIP IF NOT
3519 022724 104401 001166  4$:    TYPE   $QUES   ;; TYPE A '?'
3520 022730 000763          BR      1$          ;; CLEAR THE BUFFER AND LOOP
3521 022732 111337 023002  3$:    MOVB   (R3), 9$  ;; ECHO THE CHARACTER
3522 022736 104401 023002  TYPE    9$
3523 022742 122723 000015  CMPB   #15, (R3)+  ;; CHECK FOR RETURN
3524 022746 001356          BNE     2$          ;; LOOP IF NOT RETURN
3525 022750 105063 177777  CLRB   -1(R3)      ;; CLEAR RETURN (THE 15)
3526 022754 104401 001170  TYPE   $LF         ;; TYPE A LINE FEED
3527 022760 012603          MOV    (SP)+, R3   ;; RESTORE R3
3528 022762 011646          MOV    (SP), -(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
3529 022764 016666 000004 000002  MOV    4(SP), 2(SP) ;; FIRST ASCII CHARACTER ON IT
3530 022772 012766 023004 000004  MOV    #STTYIN, 4(SP)
3531 023000 000002          RTI          ;; RETURN
3532 023002 000          9$:    .BYTE  0          ;; STORAGE FOR ASCII CHAR. TO TYPE
3533 023003 000          .BYTE  0          ;; TERMINATOR
3534 023004 000010          $TTYIN: .BLKB  8.   ;; RESERVE 8 BYTES FOR TTY INPUT
3535 023014 052536 005015 000          $CNTLU: .ASCIZ  /1U<15><12> ;; CONTROL "U"
3536 023021 136 006507 000012  $CNTLG: .ASCIZ  /1G<15><12> ;; CONTROL "G"
3537 023026 005015 053523 020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
3538 023034 020075 000          $MNEW:  .ASCIZ  / NEW = /
3539 023037 040 047040 053505
3540 023044 036440 000040
3541          .SBTTL READ AN OCTAL NUMBER FROM THE TTY
3542
3543          ;; *****
3544          ;; *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
3545          ;; *CHANGE IT TO BINARY.
3546          ;; *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
3547          ;; *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED

```

3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603

023050 011646
023052 016666 000004 000002
023060 010046
023062 010146
023064 010246
023066 104407
023070 012600
023072 010037 023176
023076 005001
023100 005002
023102 112046
023104 001420
023106 122716 000060
023112 003026
023114 122716 000067
023120 002423
023122 006301
023124 006102
023126 006301
023130 006102
023132 006301
023134 006102
023136 042716 177770
023142 062601
023144 000756
023146 005726
023150 010166 000012
023154 010237 023206
023160 012602
023162 012601
023164 012600
023166 000002
023170 005726
023172 105010
023174 104401
023176 000000
023200 104401 001166
023204 000730
023206 000000
023210 105777 155724
023214 100003
023216 005737 011024
023222 001041

```

;* FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
;* THEN BE RE-TYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
;* CALL:
;*   RDOCT          ;; READ AN OCTAL NUMBER
;*   RETURN HERE   ;; LOW ORDER BITS ARE ON TOP OF THE STACK
;*                ;; HIGH ORDER BITS ARE IN $HIOCT

$RDOCT: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR THE
MOV      4(SP), 2(SP)          ;; INPUT NUMBER
MOV      R0, -(SP)             ;; PUSH R0 ON STACK
MOV      R1, -(SP)             ;; PUSH R1 ON STACK
MOV      R2, -(SP)             ;; PUSH R2 ON STACK
1$: RDLIN                    ;; READ AN ASCII LINE
MOV      (SP)+, R0              ;; GET ADDRESS OF 1ST CHARACTER
MOV      R0, 5$                ;; AND SAVE IT
CLR      R1                      ;; CLEAR DATA WORD
CLR      R2
2$: MOVB      (R0)+, -(SP)      ;; PICKUP THIS CHARACTER
BEQ      3$                      ;; IF ZERO GET OUT
CMPB     #'0, (SP)              ;; MAKE SURE THIS CHARACTER
BGT      4$                      ;; IS AN OCTAL DIGIT
CMPB     #'7, (SP)
BLT      4$
ASL      R1                      ;; *2
RC
ASL      R1                      ;; *4
ROL      R2
ASL      R1                      ;; *8
ROL      R2
BIC      #'07, (SP)            ;; STRIP THE ASCII JUNK
ADD      (SP)+, R1              ;; ADD IN THIS DIGIT
BR       2$                      ;; LOOP
3$: TST      (SP)+              ;; CLEAN TERMINATOR FROM STACK
MOV      R1, 12(SP)            ;; SAVE THE RESULT
MOV      R2, $HIOCT
MOV      (SP)+, R2              ;; POP STACK INTO R2
MOV      (SP)+, R1              ;; POP STACK INTO R1
MOV      (SP)+, R0              ;; POP STACK INTO R0
RTI
4$: TST      (SP)+              ;; CLEAN PARTIAL FROM STACK
CLRB     (R0)                   ;; SET A TERMINATOR
TYPE     ;; TYPE UP THRU THE BAD CHAR.
5$: .WORD    0
TYPE     $QUES                  ;; "?" "CR" & "LF"
3R      1$                      ;; TRY AGAIN
$HIOCT: .WORD    0              ;; HIGH ORDER BITS GO HERE
MSCOPE: TSTB     $SWR            ;; TEST BIT 7
BPL      $SCOPE                 ;; NOT SET
TST      LOOP                   ;; TEST LOOP
BNE      $OVER                  ;; SET LOOP ON TEST

```

.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>)

```

3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659

023224
023224 032777 040000 155706
023232 001035
023234 000416
023236 013746 000004
023242 012737 023262 000004
023250 005737 177060
023254 012637 000004
023260 000414
023262 022626
023264 012637 000004
023270 000416
023272
023272 032777 000400 155640
023300 001404
023302 127737 155632 001102
023310 001406
023312 105237 001102
023316 011637 001106
023322 105037 001103
023326 013777 001102 155606
023334 013716 001106
023340 000002

023342
023342 113737 001102 001252
023350 105237 001103
023354 001775
023356 013777 001102 155556
023364 005237 001112
023370 011637 001116
023374 162737 000002 001116
023402 117737 155510 001114
023410 032777 020000 155522
023416 001034

```

;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY(15:08)
;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;*SW14=1      LOOP ON TEST
;*SW08=1      LOOP ON TEST IN SWR(7:0)
;*CALL
;*          SCOPE          ;;SCOPE=IOT

$SCOPE:
1$: BIT      #BIT14,$SWR      ;;LOOP ON PRESENT TEST?
   BNE      $OVER           ;;YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR      6$          ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                               THIS INSTRUCTION TO A "NOP" (NOP=240)
                               SAVE THE CONTENTS OF THE ERROR VECTOR
   MOV      2$ERRVEC, -(SP)  ;;SET FOR TIMEOUT
   MOV      5$, 2$ERRVEC    ;;TIME OUT ON XOR?
   TST      2$177060        ;;RESTORE THE ERROR VECTOR
   MOV      (SP)+, 2$ERRVEC  ;;GO TO THE NEXT TEST
   BR      $SVLAD           ;;CLEAR THE STACK AFTER A TIME OUT
5$: CMP      (SP)+, (SP)+    ;;RESTORE THE ERROR VECTOR
   MOV      (SP)+, 2$ERRVEC  ;;LOOP ON THE PRESENT TEST
   BR      $OVER           ;*****END OF CODE FOR THE XOR TESTER*****
6$: BIT      #BIT08,$SWR    ;;LOOP ON SPEC. TEST?
   BEQ      $SVLAD          ;;BR IF NO
   CMPB     2$SWR,$STNM     ;;ON THE RIGHT TEST? SWR(7:0)
   BEQ      $OVER          ;;BR IF YES
   INCB     $STNM           ;;COUNT TEST NUMBERS
   MOV      (SP), $LPADR    ;;SAVE SCOPE LOOP ADDRESS
   CLRB     $ERFLG         ;;ZERO THE ERROR FLAG
$OVER: MOV     $STNM, 2$DISPLAY ;;DISPLAY TEST NUMBER
   MOV      $LPADR, (SP)   ;;FUDGE RETURN ADDRESS
   RTI                    ;;FIXES PS

```

.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

$ERROR:
7$: MOVB     $STNM, TSTNUM
   INCB     $ERFLG        ;;SET THE ERROR FLAG
   BEQ      7$           ;;DON'T LET THE FLAG GO TO ZERO
   MOV      $STNM, 2$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
   INC      $ERTTL        ;;INC THE ERROR COUNT
   MOV      (SP), $ERRPC  ;;GET ADDRESS OF ERROR INSTRUCTION
   SUB     #2, $ERRPC
   MOVB    2$ERRPC, $ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
   BIT     #BIT13,$SWR    ;;SKIP TYPEOUT IF SET
   BNE     20$           ;;SKIP TYPEOUTS

```

```

3660 023420 004737 023454 JSR PC,SERRTYP ;;GO TO USER ERROR ROUTINE
3661 023424 104401 001167 TYPE ,SCRFL
3662 023430 20$: TST JSWR ;;HALT ON ERROR
3663 023430 005777 155504 2$: BPL 3$ ;;SKIP IF CONTINUE
3664 023434 100001 HALT ;;HALT ON ERROR!
3665 023436 000000
3666 023440 3$: CMP #SENDAD,2*42 ;;ACT-11 AUTO-ACCEPT?
3667 023440 022737 007010 000042 BNE 6$ ;;BRANCH IF NO
3668 023446 001001 HALT ;;YES
3669 023450 000000
3670 023452 6$: RTI ;;RETURN
3671 023452 000002

```

.SBTTL ERROR MESSAGE TIMEOUT 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.

```

```

3680 023454 SERRTYP:
3681 023454 104401 001167 TYPE ,SCRFL ;; "CARRIAGE RETURN" & "LINE FEED"
3682 023460 010046 MOV RO,-(SP) ;; SAVE RO
3683 023462 005000 CLR RO ;; PICKUP THE ITEM INDEX
3684 023464 153700 001114 BISR 2*$ITEMB,RO
3685 023470 001004 BNE 1$ ;; IF ITEM NUMBER IS ZERO, JUST
3686 023472 013746 001116 MOV $ERRPC,-(SP) ;; TYPE THE PC OF THE ERROR
3687 023472 013746 001116 ;; SAVE $ERRPC FOR TIMEOUT
3688 023472 013746 001116 ;; ERROR ADDRESS
3689 023476 104402 TYPCC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3690 023500 000426 BR 6$ ;; GET OUT
3691 023502 005300 1$: DEC RO ;; ADJUST THE INDEX SO THAT IT WILL
3692 023504 006300 ASL RO ;; WORK FOR THE ERROR TABLE
3693 023506 006300 ASL RO
3694 023510 006300 ASL RO
3695 023512 062700 001172 ADD #ERRTB,RO ;; FORM TABLE POINTER
3696 023516 012037 023526 MOV (RO)+,2$ ;; PICKUP "ERROR MESSAGE" POINTER
3697 023522 001404 BEQ 3$ ;; SKIP TIMEOUT IF NO POINTER
3698 023524 104401 TYPE ;; TYPE THE "ERROR MESSAGE"
3699 023526 000000 .WORD 0 ;; "ERROR MESSAGE" POINTER GOES HERE
3700 023530 104401 001167 TYPE ,SCRFL ;; "CARRIAGE RETURN" & "LINE FEED"
3701 023534 012037 023544 3$: MOV (RO)+,4$ ;; PICKUP "DATA HEADER" POINTER
3702 023540 001404 BEQ 5$ ;; SKIP TIMEOUT IF 0
3703 023542 104401 TYPE ;; TYPE THE "DATA HEADER"
3704 023544 000000 .WORD 0 ;; "DATA HEADER" POINTER GOES HERE
3705 023546 104401 001167 TYPE ,SCRFL ;; "CARRIAGE RETURN" & "LINE FEED"
3706 023552 011000 5$: MOV (RO),RO ;; PICKUP "DATA TABLE" POINTER
3707 023554 001004 BNE 7$ ;; GO TYPE THE DATA
3708 023556 012600 6$: MOV (SP)+,RO ;; RESTORE RO
3709 023560 104401 001167 TYPE ,SCRFL ;; "CARRIAGE RETURN" & "LINE FEED"
3710 023564 000207 RTS PC ;; RETURN
3711 023566 7$: MOV 2(RO)+,-(SP) ;; SAVE 2(RO)+ FOR TIMEOUT
3712 023566 013046 TYPCC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3713 023570 104402 TST (RO) ;; IS THERE ANOTHER NUMBER?
3714 023572 005710 BEG 6$ ;; BR IF NO
3715 023574 001770

```



```

3716 023576 104401 023604
3717 023602 000771
3718 023604 020040 000
3719 023610 023610
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738 023610 105737 001157
3739 023614 100002
3740 023616 000000
3741 023620 000407
3742 023622 010046
3743 023624 017600 000002
3744 023630 112046
3745 023632 001005
3746 023634 005726
3747 023636 012600
3748 023640 062716 000002
3749 023644 000002
3750 023646 122716 000011
3751 023652 001430
3752 023654 122716 000200
3753 023660 001006
3754 023662 005726
3755 023664 104401
3756 023666 001167
3757 023670 105037 024024
3758 023674 000755
3759 023676 004737 023760
3760 023702 123726 001156
3761 023706 001350
3762 023710 013746 001154
3763
3764 023714 105366 000001
3765 023720 002770
3766 023722 004737 023760
3767 023726 105337 024024
3768 023732 000770
3769
3770
3771
3772

```

```

TYPE 8$ ::TYPE TWO(2) SPACES
BR 7$ ::LOOP
9$: .ASCIZ / / ::TWO(2) SPACES
.EVEN

.SBTTL TYPE ROUTINE

:*****
:*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
:*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
:*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
:*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
:*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE .MESADR ::MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*
$TYPE: TSTB $TFPLG ::IS THERE A TERMINAL?
SPL 1$ ::BR IF YES
HALT ::HALT HERE IF NO TERMINAL
BR 3$ ::LEAVE
1$: MOV RO, -(SP) ::SAVE RO
MOV 02(SP), RO ::GET ADDRESS OF ASCIZ STRING
2$: MOVB (RO)+, -(SP) ::PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 4$ ::BR IF IT ISN'T THE TERMINATOR
TST (SP)+ ::IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+, RO ::RESTORE RO
3$: ADD #2, (SP) ::ADJUST RETURN PC
RTI ::RETURN
4$: CMPB #HT, (SP) ::BRANCH IF <HT>
BEQ 8$
CMPB #CRLF, (SP) ::BRANCH IF NOT <CRLF>
BNE 5$
TST (SP)+ ::POP <CR><LF> EQUIV
TYPE A CR AND LF
5$: CLRB $CHARCNT ::CLEAR CHARACTER COUNT
BR 2$ ::GET NEXT CHARACTER
5$: JSR PC, $TYPEC ::GO TYPE THIS CHARACTER
6$: CMPB $FILLC, (SP)+ ::IS IT TIME FOR FILLER CHARS.?
BNE 2$ ::IF NO GO GET NEXT CHAR.
MOV $NULL, -(SP) ::GET # OF FILLER CHARS. NEEDED
AND THE NULL CHAR.
7$: DECB 1(SP) ::DOES A NULL NEED TO BE TYPED?
BLT 6$ ::BR IF NO--GO POP THE NULL OFF OF STACK
JSR PC, $TYPEC ::GO TYPE A NULL
DECB $CHARCNT ::DO NOT COUNT AS A COUNT
BR 7$ ::LOOP

```

;HORIZONTAL TAB PROCESSOR

```

3772 023734 112716 000040 98:   MOVB   #' (SP)      ;; REPLACE TAB WITH SPACE
3773 023740 004737 023760 98:   JSR    PC,$TYPEC    ;; TYPE A SPACE
3774 023744 132737 000007 024024 BITB   #7,$CHARCNT   ;; BRANCH IF NOT AT
3775 023752 001372         BNE    98           ;; TAB STOP
3776 023754 005726         TST    (SP)+        ;; POP SPACE OFF STACK
3777 023756 000724         BR     28           ;; GET NEXT CHARACTER
3778 023760 105777 155164 $TYPEC: TSTB   @STPS    ;; WAIT UNTIL PRINTER IS READY
3779 023764 100375         BPL   $TYPEC
3780 023766 116677 000002 155156 MOVB   2(SP),@STPB   ;; LOAD CHAR TO BE TYPED INTO DATA REG.
3781 023774 122765 000015 000002 CMPB   #CR,2(SP)    ;; IS CHARACTER A CARRIAGE RETURN?
3782 024002 001003         BNE    18           ;; BRANCH IF NO
3783 024004 105037 024024 CLRB   $CHARCNT     ;; YES--CLEAR CHARACTER COUNT
3784 024010 000406         BR     $TYPEX      ;; EXIT
3785 024012 122766 000012 000002 18:   CMPB   #LF,2(SP)   ;; IS CHARACTER A LINE FEED?
3786 024020 001402         BEQ   $TYPEX      ;; BRANCH IF YES
3787 024022 105227         INCB  (PC)+        ;; COUNT THE CHARACTER
3788 024024 000000 $CHARCNT: .WORD   0 ;; CHARACTER COUNT STORAGE
3789 024026 000207 $TYPEX: RTS      PC

```

.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.
;;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;;CALL:
;;   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED
;;   TYPOS   N              ;; CALL FOR TYPEOUT
;;   .BYTE  M              ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;;   .BYTE  M              ;; M=1 OR 0
;;                               ;; 1=TYPE LEADING ZEROS
;;                               ;; 0=SUPPRESS LEADING ZEROS
;;$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
;;$TYPOS OR $TYPOC
;;CALL:
;;   MOV     NJM,-(SP)     ;; NUMBER TO BE TYPED
;;   TYPON   N              ;; CALL FOR TYPEOUT
;;$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
;;CALL:
;;   MOV     NUM,-(SP)     ;; NUMBER TO BE TYPED
;;   TYPOC   N              ;; CALL FOR TYPEOUT
3817 024030 017646 000000 $TYPOS: MOV    @ (SP),-(SP)  ;; PICKUP THE MODE
3818 024034 116637 000001 024253 MOVB   1(SP),$OFILL   ;; LOAD ZERO FILL SWITCH
3819 024042 112637 024255 MOVB   (SP)+,$OMODE+1 ;; NUMBER OF DIGITS TO TYPE
3820 024046 062716 000002 ADD    #2,(SP)        ;; ADJUST RETURN ADDRESS
3821 024052 000406 BR     $TYPON
3822 024054 112737 000001 024253 $TYPOC: MOVB   #1,$OFILL   ;; SET THE ZERO FILL SWITCH
3823 024062 112737 000006 024255 MOVB   #6,$OMODE+1   ;; SET FOR SIX(6) DIGITS
3824 024070 112737 000005 024252 $TYPON: MOVB   #5,$OCNT   ;; SET THE ITERATION COUNT
3825 024076 010346 MOV    R3,-(SP)      ;; SAVE R3
3826 024100 010446 MOV    R4,-(SP)      ;; SAVE R4
3827 024102 010546 MOV    R5,-(SP)      ;; SAVE R5

```

```

3828 024104 113704 024255      MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
3829 024110 005404              NEG      R4              ;;
3830 024112 062704 000006      ADD      #6,R4           ;;SUBTRACT IT FOR MAX. ALLCWED
3831 024116 110437 024254      MOVB    R4,$OMODE       ;;SAVE IT FOR USE
3832 024122 113704 024253      MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
3833 024126 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
3834 024132 005003              CLR      R3              ;;CLEAR THE OUTPUT WORD
3835 024134 006105      1$:    ROL      R5        ;;ROTATE MSB INTO "C"
3836 024136 000404              BR       3$              ;;GO DO MSB
3837 024140 006105      2$:    ROL      R5        ;;FORM THIS DIGIT
3838 024142 006105      ROL      R5
3839 024144 006105      ROL      R5
3840 024146 010503              MOV      R5,R3
3841 024150 006103      3$:    ROL      R3        ;;GET LSB OF THIS DIGIT
3842 024152 105337 024254      DECB    $OMODE          ;;TYPE THIS DIGIT?
3843 024156 100016      BPL     7$              ;;BR IF NO
3844 024160 042703 177770      BIC     #177770,R3      ;;GET RID OF JUNK
3845 024164 001002      BNE     4$              ;;TEST FOR 0
3846 024166 005704              TST     R4              ;;SUPPRESS THIS 0?
3847 024170 001403      BEQ     5$              ;;BR IF YES
3848 024172 005204      4$:    INC      R4        ;;DON'T SUPPRESS ANYMORE 0'S
3849 024174 052703 000060      BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3850 024200 052703 000040      5$:    BIS     #' ,R3      ;;MAKE ASCII IF NOT ALREADY
3851 024204 110337 024250      MOVB    R3,$$          ;;SAVE FOR TYPING
3852 024210 104401 024250      TYPE   $$              ;;GO TYPE THIS DIGIT
3853 024214 105337 024252      7$:    DECB    $OCNT      ;;COUNT BY 1
3854 024220 003347      BGT     2$              ;;BR IF MORE TO DO
3855 024222 002402      BLT     6$              ;;BR IF DONE
3856 024224 005204      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3857 024226 000744      BR      2$              ;;GO DO THE LAST DIGIT
3858 024230 012605      E$:    MOV     (SP)+,R5    ;;RESTORE R5
3859 024232 012604      MOV     (SP)+,R4        ;;RESTORE R4
3860 024234 012603      MOV     (SP)+,R3        ;;RESTORE R3
3861 024236 016666 000002 000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3862 024244 012616      MOV     (SP)+,(SP)
3863 024246 000002      RTI                    ;;RETURN
3864 024250      8$:    .BYTE   0          ;;STORAGE FOR ASCII DIGIT
3865 024251      .BYTE   0          ;;TERMINATOR FOR TYPE ROUTINE
3866 024252      .BYTE   0          ;;OCTAL DIGIT COUNTER
3867 024253      $OFILL: .BYTE   0          ;;ZERO FILL SWITCH
3868 024254 000000      $OMODE: .WORD   0          ;;NUMBER OF DIGITS TO TYPE
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880 024256      $RAND:
3881 024256 010046      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3882 024260 010146      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3883 024262 010246      MOV     R2,-(SP)        ;;PUSH R2 ON STACK

;*****
;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
*

```

```

3884 024264 013700 024356
3885 024270 013701 024354
3886 024274 012702 177771
3887 024300 006300
3888 024302 006101
3889 024304 005202
3890 024306 001374
3891 024310 063700 024356
3892 024314 005501
3893 024316 063701 024354
3894 024322 062700 001057
3895 024326 005501
3896 024330 062701 047401
3897 024334 010037 024356
3898 024340 010137 024354
3899 024344 012602
3900 024346 012601
3901 024350 012600
3902 024352 000207
3903 024354 176543
3904 024356 123456

```

```

MOV $LONUM,R0 ;; SET R0 WITH LOW
MOV $SHINUM,R1 ;; SET R1 WITH HIGH
MOV #-7,R2 ;; SET SHIFT COUNT
1$: ASL R0 ;; SHIFT R0 LEFT AND
ROL R1 ;; ROTATE CARRY INTO R1 AND
INC R2 ;; CHECK FOR DONE
BNE 1$ ;; CONTINUE SHIFT LOOP
ADD $LONUM,R0 ;; ADD NUMBER TO MAKE X 129
ADC R1 ;; PROPOGATE CARRY
ADD $SHINUM,R1 ;; ADD NUMBER TO MAKE X 129
ADD #1057,R0 ;; ADD LOW CONSTANT
ADC R1 ;; PROPOGATE CARRY
ADD #47401,R1 ;; ADD HIGH CONSTANT
MOV R0,$LONUM ;; SAVE R0
MOV R1,$SHINUM ;; SAVE R1
MOV (SP)+,R2 ;; POP STACK INTO R2
MOV (SP)+,R1 ;; POP STACK INTO R1
MOV (SP)+,R0 ;; POP STACK INTO R0
RTS PC ;; RETURN
$SHINUM: .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.

```

```

3913 024360 010046
3914 024362 016600 000002
3915 024366 005740
3916 024370 111000
3917 024372 006300
3918 024374 016000 024414
3919 024400 000200

```

```

$TRAP: MOV R0,-(SP) ;; SAVE R0
MOV 2(SP),R0 ;; GET TRAP ADDRESS
TST -(R0) ;; BACKUP BY 2
MOVB (R0),R0 ;; GET RIGHT BYTE OF TRAP
ASL R0 ;; POSITION FOR INDEXING
MOV $TRAPAD(R0),R0 ;; INDEX TO TABLE
RTS R0 ;; GO TO ROUTINE

```

```

;; THIS IS USE TO HANDLE THE "GETPRI" MACRO

```

```

3924 024402 011646
3925 024404 016666 000004 000002
3926 024412 000002

```

```

$TRAP2: MOV (SP),-(SP) ;; MOVE THE PC DOWN
MOV 4(SP),2(SP) ;; MOVE THE PSW DOWN
RTI ;; RESTORE THE PSW

```

```

.SBTTL TRAP TABLE

```

```

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

```

```

3933
3934
3935 024414 024402
3936 024416 023610
3937 024420 024054
3938 024422 024030
3939 024424 024070

```

```

; ROUTINE
; -----
$TRPAD: .WORD $TRAP2
$TYPE ;; CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
$TYPOC ;; CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
$TYPOS ;; CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
$TYPON ;; CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)

```

```

3940 024426 012332          $TYPDS ;;CALL=TYPDS   TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
3941
3942
3943 024430 022556          $RDCHR ;;CALL=RDCHR   TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
3944 024432 022676          $RDLIN ;;CALL=RDLIN   TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
3945 024434 023050          $RDOCT ;;CALL=RDOCT   TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
3946
3947 .SBTTL POWER DOWN AND UP ROUTINES
3948
3949 ::*****
3950 024436 012737 024576 000024 $PWRDN: MOV    $ILLUP,@#PWRVEC ;;SET FOR FAST UP
3951 024444 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
3952 024452 010046          MOV    RD,-(SP) ;;PUSH RD ON STACK
3953 024454 010146          MOV    R1,-(SP) ;;PUSH R1 ON STACK
3954 024456 010246          MOV    R2,-(SP) ;;PUSH R2 ON STACK
3955 024460 010346          MOV    R3,-(SP) ;;PUSH R3 ON STACK
3956 024462 010446          MOV    R4,-(SP) ;;PUSH R4 ON STACK
3957 024464 010546          MOV    R5,-(SP) ;;PUSH R5 ON STACK
3958 024466 017746 154446      MOV    @SWR,-(SP) ;;PUSH @SWR ON STACK
3959 024472 010637 024602      MOV    SP,$SAVR6 ;;SAVE SP
3960 024476 012737 024510 000024      MOV    $PWRUP,@#PWRVEC ;;SET UP VECTOR
3961 024504 000000          HALT
3962 024506 000776          BR     .-2 ;;HANG UP
3963
3964 ::*****
3965 .POWER UP ROUTINE
3966 024510 012737 024576 000024 $PWRUP: MOV    $ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
3967 024516 013706 024602      MOV    $SAVR6,SP ;;GET SP
3968 024522 005037 024602      CLR    $SAVR6 ;;WAIT LOOP FOR THE TTY
3969 024526 005237 024602      i$:   INC    $SAVR6 ;;WAIT FOR THE INC
3970 024532 001375          BNE    i$ ;;OF WORD
3971 024534 012677 154400      MOV    (SP)+,@SWR ;;POP STACK INTO @SWR
3972 024540 012605          MOV    (SP)+,R5 ;;POP STACK INTO R5
3973 024542 012604          MOV    (SP)+,R4 ;;POP STACK INTO R4
3974 024544 012603          MOV    (SP)+,R3 ;;POP STACK INTO R3
3975 024546 012602          MOV    (SP)+,R2 ;;POP STACK INTO R2
3976 024550 012601          MOV    (SP)+,R1 ;;POP STACK INTO R1
3977 024552 012600          MOV    (SP)+,R0 ;;POP STACK INTO R0
3978 024554 012737 024436 000024      MOV    $PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
3979 024562 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
3980 024570 104401          TYPE   $POWER ;;REPORT THE POWER FAILURE
3981 024572 024604          $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
3982 024574 000002          RTI
3983 024576 000000          $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
3984 024600 000776          BR     .-2 ;;BEFORE THE POWER DOWN WAS COMPLETE
3985 024602 000000          $SAVR6: 0 ;;PUT THE SP HERE
3986 024604 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
3987 024612 000122
3988
3989 024614 000000          BJJFER: 0
3990 000001          .END

```


MKEA	016107	1965	2984#					
MKEA1	016120	1952*	1953*	1958*	1959*	1961*	1962*	2986#
MKEB	016125	1939*	1940*	1941*	2988#			
MKEH	016134	2160	2991#					
MOVDN1	003104	1181	1198#	1201				
MOVDWN	003100	1188	1197#					
MOVRIQ	003122	1182	1184	1185	1189	1191	1192	1204#
MPTCNT	013726	1798	2768#					
MPTSCN	013775	1821	2775#					
MQ0	017256	2259	3112#					
MQ1	017343	2277	2301	3121#				
MQ2	017431	2298	3131#					
MSCOPE	023210	939	3594#					
MSGTND	005072	1483	1544#					
MSGTXT	004752	1481	1530#					
MSGTYP	002432	1017	1075#					
MTEXT0	021532	1779	1855	3371#				
MTEXT1	021636	1856	3383#					
MTEXT2	021737	1857	3394#					
MTEXT3	022041	1858	3406#					
MTEXT4	022124	1859	3416#					
MTEXT5	022226	1860	3427#					
MTEXT6	022327	1861	3438#					
MTEXT7	022361	1862	3443#					
MTEXT8	022461	1863	3454#					
MT08	012676	2445	2662#					
M91	013013	1093	1657	2683#				
M910	013304	1359	2717#					
M911	013335	1403	2722#					
M912	013370	1436	2727#					
M914	013424	979	2732#					
M92	013061	1104	1671	2690#				
M920	013471	1747	2739#					
M921	013521	1773	2744#					
M922	013555	1562	2749#					
M9221	013612	1604	2754#					
M923	013667	1635	2762#					
M923A	014032	1837	2780#					
M93	013110	1125	1692	2694#				
M94	013152	1149	1719	2700#				
M96	013203	1173	2705#					
M97	013236	1272	2710#					
M98	017610	1461	1469	3160#				
NOWEOP	007044	1916	1921#					
OCTAL	011636	1938	2453#	2527	2539			
OVRAL	004750	1478*	1522*	1529#				
PASHED	020000	1887	3181#					
PATH	016211	1439	3004#					
PIRQ =	177772	637#						
PIRQVE =	000240	731#						
PNTWID	001316	874#	1804					
PRNTST	006336	1625	1788#					
PRTCNT	001274	865#	890*	893*	1630	1632*	1921	1923*
PRO =	000000	654#						
PR1 =	000040	655#						
PR2 =	000100	656#						

K07

MAINDEC-11-DZVTC-D MACY11 27(1006) 21-FEB-77 15:32 PAGE 99
DZVTCO.P11 21-FEB-77 15:27 CROSS REFERENCE TABLE -- MACRO NAMES

.SRDOC	612#	3541
.SREAD	612#	3467
.SSAVE	612#	
.SSCOP	612#	3599
.SSPAC	612#	
.SSWDC	612#	
.STRAP	612#	3905
.STYPD	612#	2559
.STYPE	612#	3721
.STYPC	612#	3792

. ABS. 024616 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

.DZVTCO.SEG/CRF/SOL/NL:TOC=DSKZ:DZVTCO.P11
RUN-TIME: 15 10 : SECONDS
RUN-TIME RATIO: 282/27=10.2
CORE USED: 25K (49 PAGES)