

15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

1.0 ABSTRACT

THE PROGRAM WILL RUN ON NON-SWITCH REGISTER CPU TYPES

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

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

ONLY ONE VT55 IS TESTED AT ONE TIME.

2.0 REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

THIS PROGRAM USES 4K OF MEMORY.

3.0 LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

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
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

5.0 OPERATING PROCEDURE

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

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

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

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

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

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

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

6.0 ERRORS

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

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

F01

MAINDEC-11-DZVTD-8
DZVTDB.SAC

MACY11 27(732) 25-SEP-76 10:10 PAGE 6

182
183

EXPT - EXPECTED INPUT CHARACTER
RCVD - RECEIVED INPUT CHARACTER

119
118
117
116
115
114
113
112
111
110
109
108
107
106
105
104
103
102
101
100
99
98
97
96
95
94
93
92
91
90
89
88
87
86
85
84
83
82
81
80
79
78
77
76
75
74
73
72
71
70
69
68
67
66
65
64
63
62
61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1

7.0 MISCELLANEOUS

1. THE OPERATOR SHOULD SET SW 15 AND 13 IF THE VT55 UNDER TEST IS THE CONSOLE TTY.
2. ONLY ONE VT55 CAN BE TESTED AT ONE TIME.
3. EXECUTION TIME
EXECUTION TIME WILL VARY WITH THE "BAUD" RATE AND IF A "COPIER" IS CONNECTED.
4. DEVICE ADDRESS PROGRAM LOCATIONS
THE LOCATION "SBASE" CONTAINS THE FIRST DL11 ADDRESS IF SEVERAL VT-55'S ARE BEING TESTED. THE DEFAULT IS THE CONSOLE ADDRESS (177560). THE LOCATION "LAST" CONTAINS THE LAST DL11 ADDRESS IF SEVERAL VT-55'S ARE BEING TESTED. LOCATION VTNOW CONTAINS THE CURRENT DL11 BASE ADDRESS.

NOTE: IF THESE LOCATIONS ARE CHANGED, THE OPERATOR MUST START THE TEST AGAIN AT LOC. 200. THE PROGRAM WILL USE THE BASE ADDRESS TO UPDATE THE ACTUAL PROGRAM VALUES.
5. PROGRAM IS CHAINABLE UNDER XODP/ACT-11. APT HOOKS HAVE BEEN PROVIDED BUT NOT TESTED.

8.0 RESTRICTIONS

NONE.

22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

9.0 PROGRAM DESCRIPTION

9.1 GROWING HORIZONTAL LINE

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

9.2 GROWING VERTICAL LINE

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

9.3 STEPPING HORIZONTAL LINE FOR GRAPH 0

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

9.4 STEPPING HORIZONTAL LINE FOR GRAPH 1

SAME AS 9.3 EXCEPT GRAPH 1 IS ENABLED.

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
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324

9.5 DIFFERENT DATA ON GRAPH 0 AND 1

GRAPH 0:

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

GRAPH 1:

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

9.6 DISPLAY A STEPPING HISTOGRAM LINE ON GRAPH 0

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

9.7 DISPLAY A STEPPING HISTOGRAM LINE ON GRAPH 1

SAME AS 9.6 EXCEPT GRAPH 1 IS ENABLED.

9.8 HISTOGRAM ON GRAPH 0 AND 1

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

9.9 CURSORS ON GRAPH 0

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

9.10 CURSORS ON GRAPH 1

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

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
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426

9.11 STARTING COORDINATE ON GRAPH 0

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

9.12 STARTING COORDINATE ON GRAPH 1

SAME AS 9.11 EXCEPT GRAPH 1 IS ENABLED.

9.13 VT55 ADJUSTMENT PATTERN

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

```

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

```

427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007

000000
000040
000100
000140
000200
000240
000300
000340

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004

.SBTTL BASIC DEFINITIONS
:
: *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ::BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ::BASIC DEFINITION OF SCOPE CALL
:
: *MISCELLANEOUS DEFINITIONS
HT= 11 ::CODE FOR HORIZONTAL TAB
LF= 12 ::CODE FOR LINE FEED
CR= 15 ::CODE FOR CARRIAGE RETURN
CRLF= 200 ::CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ::PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLM= 177774 ::STACK LIMIT REGISTER
PIRQ= 177772 ::PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ::HARDWARE SWITCH REGISTER
DDISP= 177570 ::HARDWARE DISPLAY REGISTER
:
: *GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0 ::GENERAL REGISTER
R1= %1 ::GENERAL REGISTER
R2= %2 ::GENERAL REGISTER
R3= %3 ::GENERAL REGISTER
R4= %4 ::GENERAL REGISTER
R5= %5 ::GENERAL REGISTER
R6= %6 ::GENERAL REGISTER
R7= %7 ::GENERAL REGISTER
.EQUIV R6,SP ::STACK POINTER
.EQUIV R7,PC ::PROGRAM COUNTER
:
: *PRIORITY LEVEL DEFINITIONS
PR0= 0 ::PRIORITY LEVEL 0
PR1= 40 ::PRIORITY LEVEL 1
PR2= 100 ::PRIORITY LEVEL 2
PR3= 140 ::PRIORITY LEVEL 3
PR4= 200 ::PRIORITY LEVEL 4
PR5= 240 ::PRIORITY LEVEL 5
PR6= 300 ::PRIORITY LEVEL 6
PR7= 340 ::PRIORITY LEVEL 7
:
: *"SWITCH REGISTER" SWITCH DEFINITIONS
SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4

483 000002
484 000001
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
530
531
532
533
534
535
536
537

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

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
BPTVEC= 14 : BREAKPOINT TRAP (BPT)
IOTVEC= 20 : INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24 : POWER FAIL
EMTVEC= 30 : EMULATOR TRAP (EMT) **ERROR**
TRAPVEC= 34 : "TRAP" TRAP
TKVEC= 60 : TTY KEYBOARD VECTOR
TPVEC= 64 : TTY PRINTER VECTOR
PIRQVEC= 240 : PROGRAM INTERRUPT REQUEST VECTOR
ABASE= 176500

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
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593

```

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;* SWITCH USE
;* -----
;* 15 HALT ON ERROR
;* 14 LOOP ON TEST
;* 13 INHIBIT ERROR TIMEOUTS
;* 12 INHIBIT SUB-TEST DELAY'S
;* 10 ENABLE "SAVE COPIER PAPER MODE"
;* 9 LOOP ON ERROR
;* 8 LOOP ON TEST IN SWR<7:0>
.SBTTL TRAP CATCHER
      =0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY-LOADED VECTORS
      =174
DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
      JMP @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
      JMP RBEGIN ;;JUMP TO RESTART ADDRESS
      JMP BEGIN2 ;;JUMP TO ADJUSTMENT PATTERN

.SBTTL ACT11 HOOKS
;*****
;HOOKS REQUIRED BY ACT11
      $SVPC=.; ;SAVE PC
      =46
SENDAD ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
      =52
      .WORD 0 ;;2)SET LOC.52 TO ZERO
      =$SVPC
      ;; RESTORE PC
      =1000
.SBTTL APT PARAMETER BLOCK
;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
      .SX=.; ;SAVE CURRENT LOCATION
      =24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200 ;;FOR APT START UP
      =44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR ;;POINT TO APT HEADER BLOCK
      =.SX ;;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-POP11 DIAGNOSTIC
;INTERFACE SPEC.
$APTHD:
$SHIBTS: .WORD 0 ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL ;; ADDRESS OF APT MAILBOX (BITS 0-15)
$STSTM: .WORD 300 ;; RUN TIM OF LONGEST TEST

```

594 001006 000300
595 001010 000300
596 001012 000031

SPASTM: .WORD 300 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
SUNITM: .WORD 300 ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD SETEND-SMAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652

001100
001100
001100
001102
001103
001104
001106
001110
001112
001114
001115
001116
001120
001122
001124
001126
001130
001132
001134
001135
001136
001140
001140
001144
001144
001146
001150
001150
001154
001155
001156
001157
001160

001162
001164
001166
001170
001171
001172

001174
001174
001176
001200
001202
001204

.SBTTL COMMON TAGS

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

SCNTAG: .=1100

STSTNM: .WORD 0
SERFLG: .BYTE 0000
SICNT: .WORD 0000
SLPADR: .WORD 0000
SLPERR: .WORD 0000
SERTTL: .WORD 0000
SITEMB: .BYTE 0000
SERMAX: .BYTE 0000
SERRPC: .WORD 0000
SGADR: .WORD 0000
SBDADR: .WORD 0000
SGOAT: .WORD 0000
SBDAT: .WORD 0000
SAUTOB: .BYTE 0000
SINTAG: .BYTE 0000
SMR: .WORD DSMR
DISPLAY: .WORD DDISP
STKS: 177560
STKB: 177562
STPS: 177564
STPB: 177566
SMULL: .BYTE 0
SFILLS: .BYTE 2
SFILLC: .BYTE 12
STPFLG: .BYTE 0
SREGAD: .WORD 0

SREGO: .WORD 0
SREG1: .WORD 0
SESCAPE: 0
SQUES: .ASCII /?/
SCRLF: .ASCII <15>
SLF: .ASCII <12>

:: START OF COMMON TAGS

:: 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 (SREGO) WAS OBTAINED
:: CONTAINS ((SREGAD)+0)
:: CONTAINS ((SREGAD)+2)
:: ESCAPE ON ERROR ADDRESS
:: QUESTION MARK
:: CARRIAGE RETURN
:: LINE FEED

.SBTTL APT MAILBOX-ETABLE

: EVEN
SMAIL: .WORD
SMSGTY: .WORD AMSGTY :: APT MAILBOX
SFATAL: .WORD AFATAL :: MESSAGE TYPE CODE
STESTN: .WORD ATESTN :: FATAL ERROR NUMBER
SPASS: .WORD APASS :: TEST NUMBER
SDEVCT: .WORD ADEVCT :: PASS COUNT
:: DEVICE COUNT

653	001206	000000	SUNIT: .WORD	AUNIT	:: I/O UNIT NUMBER
654	001210	000000	SMSGAD: .WORD	AMSGAD	:: MESSAGE ADDRESS
655	001212	000000	SMSGLG: .WORD	AMSGLG	:: MESSAGE LENGTH
656	001214		SETABLE:		:: APT ENVIRONMENT TABLE
657	001214	000	SENV: .BYTE	AENV	:: ENVIRONMENT BYTE
658	001215	000	SENVH: .BYTE	AENVH	:: ENVIRONMENT MODE BITS
659	001216	000000	SSWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
660	001220	000000	SUSR: .WORD	AUSR	:: USER SWITCHES
661	001222	000000	SCPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
662			::		BITS 15-11=CPU TYPE
663			::		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
664			::		11/70=06, PD0=07, 0=10
665			::		BIT 10=REAL TIME CLOCK
666			::		BIT 9=FLOATING POINT PROCESSOR
667			::		BIT 8=MEMORY MANAGEMENT
668	001224	000	SHANS1: .BYTE	AHANS1	:: HIGH ADDRESS, H.S. BYTE
669	001225	000	SHTYP1: .BYTE	ANTYP1	:: MEM. TYPE, BLK#1
670			::		MEM. TYPE BYTE -- (HIGH BYTE)
671			::		900 NSEC CORE=001
672			::		300 NSEC BIPOLAR=002
673			::		500 NSEC MOS=003
674	001226	000000	SHADR1: .WORD	AHADR1	:: HIGH ADDRESS, BLK#1
675			::		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
676	001230	000	SHANS2: .BYTE	AHANS2	:: HIGH ADDRESS, H.S. BYTE
677	001231	000	SHTYP2: .BYTE	ANTYP2	:: MEM. TYPE, BLK#2
678	001232	000000	SHADR2: .WORD	AHADR2	:: MEM. LAST ADDRESS, BLK#2
679	001234	000	SHANS3: .BYTE	AHANS3	:: HIGH ADDRESS, H.S. BYTE
680	001235	000	SHTYP3: .BYTE	ANTYP3	:: MEM. TYPE, BLK#3
681	001236	000000	SHADR3: .WORD	AHADR3	:: MEM. LAST ADDRESS, BLK#3
682	001240	000	SHANS4: .BYTE	AHANS4	:: HIGH ADDRESS, H.S. BYTE
683	001241	000	SHTYP4: .BYTE	ANTYP4	:: MEM. TYPE, BLK#4
684	001242	000000	SHADR4: .WORD	AHADR4	:: MEM. LAST ADDRESS, BLK#4
685	001244	000000	SVECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
686	001246	000000	SVECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
687	001250	176500	SBASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
688	001252	000000	SDEVH: .WORD	ADEVH	:: DEVICE MAP
689	001254	000000	SCON1: .WORD	ACON1	:: CONTROLLER DESCRIPTION WORD#1
690	001256		SETEND:		
691			.NEXT		

692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726

.SBTTL ERROR POINTER TABLE

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

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

SERRTB:

;ITEM 1
EM4 :INCORRECT OR UNEXPECTED INPUT CHARACTER
DH4 :ERRPC VTNOW TSTNUM EXPT RCVD
DT4 :SERRPC VTNOW TSTNUM SGO DAT SB DAT
0

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

;ITEM 3
EM3 :INVALID BUS ADDRESS, TRY AGAIN
0
0
0

001256

001256 011612
001260 011745
001262 012700
001264 000000

001266 011563
001270 011714
001272 012670
001274 000000

001276 011655
001300 000000
001302 000000
001304 000000

727
 728
 729
 730
 731
 732
 733
 734
 735
 736
 737
 738
 739
 740
 741
 742
 743
 744
 745
 746
 747
 748
 749
 750
 751
 752
 753

;VT-55 EQUALITIES

000354	MAXHOZ=236.	;MAX. HORIZ LINE COUNT
000400	MAXHZC=256.	
001000	MAXVRT=512.	;MAX. VERTICAL LINE COUNT
002000	ADDLIN=BIT10	
000033	ESC= 33	
000100	LNO= 100	;NOP
000101	LDE0= 101	;LOAD ENABLE REG. 0
000111	LDE1= 111	;LOAD ENABLE REG. 1
000102	LDG0= 102	;LOAD GRAPH 0
000112	LDG1= 112	;LOAD GRAPH 1
000103	LDC0= 103	;LOAD CURSOR ON GRAPH 0
000113	LDC1= 113	;LOAD CURSOR ON GRAPH 1
000104	LHV0= 104	;LOAD HORIZONTAL LINE
000114	LHV1= 114	;LOAD VERTICAL LINE
000110	LSC= 110	;LOAD STARTING COORDINATE
000135	COPY= 135	;COPY SCREEN COMMAND

754											
755	001306	177560			FIRST:	177560					; FIRST DEVICE ADDRESS OF SEQUENTIAL DL-11-A/B TYPE DEVIC
756											; DEFAULT TO THE CONSOLE ADDRESS
757	001310	000000			LAST:	0					; LAST DEVICE ADDRESS OF DL-11-A/B TYPE
758	001312	177560			VTNOW:	177560					; CURRENT DEVICE BUSS ADDRESS
759	001314	000000			TSTNUM:	0					; ERROR PATTERN
760											
761	001316	000100			TIMED:	100					; CHARACTER FLAG TIMEOUT CONSTANT
762	001320	000012			SUBTST:	10.					; SUBTEST DELAY CONSTANT
763	001322	000000			MFTEST:	0					
764	001324	177560			VTIS:	177560					; DEVICE ADDRESSES
765	001326	177562			VTIB:	177562					; IN DATA
766	001330	177564			VTOS:	177564					; OUT STAT
767	001332	177566			VTOB:	177566					; OUT DATA
768	001334	000000			SAVE4:	0					
769											
770	001336	022626			BUSSTR:	CMP (SP)+, (SP)+					; POP STACK
771	001340	104003				ERROR 3					; INVALID BUS ADDRESS
772	001342	000240				NOP					
773	001344	000240				NOP					
774											
775	001346	012737	002064	002062	BEGIN:	MOV #ST1, WHERE					; STARTING ACCEPTANCE TEST ADDRESS
776	001354	005037	001322			CLR MFTEST					
777	001360	005037	001334			CLR SAVE4					
778	001364	012737	001336	000004		MOV #BUSSTR, #ERRVEC					
779	001372	012737	000340	000006		MOV #340, #ERRVEC+2					
780	001400	000412				BR GINA					
781	001402	012737	002064	002062	RBEGIN:	MOV #ST1, WHERE					
782	001410	000403				BR GIN					
783	001412	012737	000001	001322	BEGIN2:	MOV #1, MFTEST					
784	001420	012737	000001	001334	GIN:	MOV #1, SAVE4					
785	001426	000005			GINA:	RESET					

```

786 .SBTTL INITIALIZE THE COMMON TAGS
787 ;;CLEAR THE COMMON TAGS (SCHTAG) AREA
788 001430 012706 001100 MOV #SCHTAG,R6 ;;FIRST LOCATION TO BE CLEARED
789 001434 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
790 001436 022706 001140 CMP #SMR,R6 ;;DONE?
791 001442 001374 BNE -6 ;;LOOP BACK IF NO
792 001444 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
793 ;;INITIALIZE A FEW VECTORS
794 001450 012737 013644 000020 MOV #SSCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
795 001456 012737 000340 000022 MOV #340,#IOTVEC+2 ;;LEVEL 7
796 001464 012737 014306 000030 MOV #SEERROR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
797 001472 012737 000340 000032 MOV #340,#EMTVEC+2 ;;LEVEL 7
798 001500 012737 015344 000034 MOV #STRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
799 001506 012737 000340 000036 MOV #340,#TRAPVEC+2 ;;LEVEL 7
800 001514 012737 015410 000024 MOV #SPWRDN,#PMRVEC ;;POWER FAILURE VECTOR
801 001522 012737 000340 000026 MOV #340,#PMRVEC+2 ;;LEVEL 7
802 001530 013737 006756 006750 MOV SENDCT,SEOPCT ;;SETUP END-OF-PROGRAM COUNTER
803 001536 005037 001166 CLR SESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
804 001542 112737 000001 001115 MOV#B #1,SEMAX ;;ALLOW ONE ERROR PER TEST
805 001550 012737 001550 001106 MOV #.,SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
806 001556 012737 001556 001110 MOV #.,SLPEAR ;;SETUP THE ERROR LOOP ADDRESS
807 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
808 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
809 001564 013746 000004 MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
810 001570 012737 001624 000004 MOV #645,#ERRVEC ;;SET UP ERROR VECTOR
811 001576 012737 177570 001140 MOV #DSMR,SMR ;;SETUP FOR A HARDWARE SWICH REGISTER
812 001604 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
813 001612 022777 177777 177320 CMP #1,#SMR ;;TRY TO REFERENCE HARDWARE SMR
814 001620 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
815 ;;AND THE HARDWARE SMR IS NOT = -1
816 001622 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
817 001624 012716 001632 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
818 001630 000002 RTI
819 001632 012737 000176 001140 65$: MOV #SMREG,SMR ;;POINT TO SOFTWARE SMR
820 001640 012737 000174 001142 MOV #DISPREG,DISPLAY
821 001646 012637 000004 66$: MOV (SP)+,#ERRVEC ;;RESTORE ERROR VECTOR
822
823 001652 005037 001202 CLR SPASS ;;CLEAR PASS COUNT
824 001656 132737 000200 001215 BITB #APTSIZE,SENVH ;;TEST USER SIZE UNDER APT
825 001664 071403 BEQ 67$ ;;YES,USE NON-APT SWITCH
826 001666 012737 001216 001140 MOV #SSMREG,SMR ;;NO,USE APT SWITCH REGISTER
827 001674 67$:
828 001674 013737 001250 001306 MOV #BASE,FIRST ;;LOAD "APT" 'S BASE ADDRESS
829 001702 005037 010544 CLR LOOP
830 001706 012737 013630 000020 MOV #MSCOPE,#IOTVEC
831 001714 005737 001334 TITST: TST SAVE4 ;;TEST FLAG
832 001720 001002 BNE 11$ ;;BR IF NON-ZERO
833 001722 104400 TYPE
834 001724 011220 TITLE

```

MAINDEC-11-DZVTD-B MACY11 27(732) 25-SEP-76 10:10 PAGE 22
 DZVTDB.SRC INITIALIZE THE COMMON TAGS

```

835
836
837 001726 022737 000176 001140 ;NOW TEST IF RUNNING ON NON-SMR CPU
838 001734 001403 11S:    CMP    @SMREG,SMR    ;TEST IF NON-SWITCH REGISTER
839 001736 105777 177176    BEQ    25          ;BR IF YES - AND ASK ADDRESS
840 001742 100022    TSTB   @SMR
841 001744 104400    BPL    RSTRTA    ;BR IF CLEARED
842 001746 011466    2S:    TYPE
843 001750 104410    MHATO
844 001752 012637 001306    RDOCT
845 001756 022737 160000 001306    MOV    (SP)+,FIRST ;SAVE THE ADDRESS
846 001764 101360    CMP    @160000,FIRST
847 001766 005777 177314    BHI    11S      ;BR IF INVALID
848 001772 005037 001310    TST   @FIRST   ;TEST IF VALID
849 001776 012737 000006 000004    CLR   LAST
850 002004 005037 000006    MOV   @6,@#4
851 002010 013737 001306 001312 RSTRTA: CLR   @#6
852 002016 012700 001324    RSTRT: MOV   FIRST,VTNOW ;LOAD INITIAL DEVICE ADDRESS
853 002022 013720 001312    MOV   @VTIS,RO   ;LOAD POINTER
854 002026 013710 001312    MOV   VTNOW,(RO)+ ;LOAD INPUT STAT
855 002032 062720 000002    MOV   VTNOW,(RO)
856 002036 013710 001312    ADD   @2,(RO)+   ;LOAD INPUT BUFFER
857 002042 062720 000004    MOV   VTNOW,(RO)
858 002046 013710 001312    ADD   @4,(RO)+   ;LOAD OUTPUT STAT
859 002052 062720 000006    MOV   VTNOW,(RO)
860 002056 000177 000006    ADD   @6,(RO)+   ;LOAD OUTPUT BUFFER
861 002062 002064 000000    JMP   @WHERE     ;JUMP TO STARTING ADDRESS
862
WHERE: TST1

```



```

945                                     ::*****
946                                     ::TEST 2           B      DISPLAY A GROWING VERTICAL LINES
947                                     ::*****
948 002520 000004 011070 1ST2:  SCOPE
949 002522 004537 011070      JSR      RS,AMSG      ;DISPLAY HEADER
950 002526 012045      DAVL
951
952 002530 012700 015562      MOV      #BUFFER,RO      ;LOAD THE STARTING ADDRESS
953 002534 112720 000033      MOVB    #ESC,(RO)+      ;LOAD "ESC" CODE
954 002540 112720 000061      MOVB    #61,(RO)+      ;LOAD "01" ENTER CODE
955 002544 112720 000101      MOVB    #LDE0,(RO)+    ;LOAD ENABLE 0
956 002550 112720 000040      MOVB    #BITS,(RO)+    ;DISABLE DISPLAY
957 002554 112720 000111      MOVB    #LDE1,(RO)+    ;LOAD ENABLE 1
958 002560 112720 000060      MOVB    #BITS!BIT4,(RO)+ ;CLEAR GRAPH, LILES, AND CURSORS
959 002564 112720 000101      MOVB    #LDE0,(RO)+    ;LOAD ENABLE 0
960 002570 112720 000041      MOVB    #BITS!BIT0,(RO)+ ;LOAD DISPLAY ENABLE
961 002574 112720 000111      MOVB    #LDE1,(RO)+    ;LOAD ENABLE 1
962 002600 112720 000042      MOVB    #BITS!BIT1,(RO)+ ;LOAD DISPLAY VERTICAL LINE
963 002604 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP
964 002610 112720 000114      MOVB    #LHV1,(RO)+    ;LOAD VERTICAL LINE
965 002614 012737 002777 0G7772      MOV      #ADDLIN!MAXVRT-1,BASE ;LOAD THE STARTING DATA VALUE
966 002622 004737 007776      JSR      PC,SHUFF      ;SHUFFEL THE DATA INTO VT-55 FORMAT
967 002626 010120      MOV      R1,(RO)+      ;SAVE THE LSB MSB BYTE
968 002630 005337 007772      DEC      BASE          ;UPDATE THE DATA
969 002634 022737 002000 007772      CMP      #ADDLIN,BASE  ;COMPARE TO LAST DATA LINE
970 002642 001367      BNE      IS           ;LOOP UNTIL DONE
971 002644 112720 000033      MOVB    #ESC,(RO)+      ;LOAD "ESC" CODE
972 002650 112720 000135      MOVB    #COPY,(RO)+    ;LOAD COPY SCREEN
973 002654 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
974 002660 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
975 002664 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
976 002670 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
977 002674 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
978 002700 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
979 002704 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
980 002710 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
981 002714 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
982 002720 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
983 002724 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
984 002730 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
985 002734 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
986 002740 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
987 002744 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
988 002750 112720 000100      MOVB    #LNO,(RO)+     ;LOAD NOP TO LOAD SILO
989 002754 105020      CLRB    (RO)+         ;LOAD TERM
990 002756 004737 010074      JSR      PC,XPRNT
991 002762 004737 010742      JSR      PC,DELAY

```

992	002766	012700	015562		MOV	#BUFFER,RO	
993	002772	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP
994	002775	112720	000114		MOVB	#LHV1,(RO)+	:LOAD VERTIACL LINE MODE AGAIN
995							
996	003002	012737	001000	007772	MOV	#MAXVRT,BASE	:LOAD STARTING DATA VALUE TO REMOVE THE LINE
997	003010	004737	007776	25:	JSR	PC,SHUFF	:SHUFFEL THE DATA INTO VT-55 FORMAT
998	003014	010120			MOV	R1,(RO)+	:SAVE THE LSB MSB BYTE
999	003016	005337	007772		DEC	BASE	:UPDATE THE DATA
1000	003022	001372			BNE	25	:BR TOLL DONE
1001							
1002	003024	112720	000033		MOVB	#ESC,(RO)+	:LOAD "ESC" CODE
1003	003030	112720	000135		MOVB	#COPY,(RO)+	:LOAD COPY SCREEN
1004	003034	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1005	003040	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1006	003044	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1007	003050	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1008	003054	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1009	003060	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1010	003064	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1011	003070	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1012	003074	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1013	003100	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1014	003104	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1015	003110	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1016	003114	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1017	003120	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1018	003124	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1019	003130	112720	000100		MOVB	#LNO,(RO)+	:LOAD NOP TO LOAD SILO
1020	003134	105020			CLRB	(RO)+	:LOAD TERMINATOR
1021	003136	004737	010074		JSR	PC,XPRNT	:DISPLAY
1022	003142	004737	010742		JSR	PC,DELAY	
1023							

```

1024                                     ;;*****
1025                                     ;;TEST 3      C      GRAPH 0 DISPLAY A STEPPING HORIZONTAL LINE
1026                                     ;;*****
1027 003146 000004                                †TST3: SCOPE
1028 003150 004537 011070                        JSR      R5,AMSG                                ;DISPLAY HEADER
1029 003154 012076                                SHLO
1030 003156 012700 015562                        MOV      #BUFFER,R0                            ;LOAD OUTPUT BUFFER POINTER
1031 003162 112720 000033                        MOV      #ESC,(R0)+                          ;ENTER VT-55 MODE
1032 003166 112720 000061                        MOV      #61,(R0)+
1033 003172 112720 000101                        MOV      #LDE0,(R0)+                          ;LOAD ENABLE 0
1034 003176 112720 000040                        MOV      #BITS,(R0)+                          ;DISABLE DISPLAY
1035 003202 112720 000111                        MOV      #LDE1,(R0)+                          ;LOAD ENABLE 1
1036 003206 112720 000060                        MOV      #BITS!BIT4,(R0)+                    ;CLEAR GRAPH, LILES, AND CURSORS
1037 003212 105020                                CLR      (R0)+                                ;LOAD TERMINATOR
1038 003214 004737 010074                        JSR      PC,XPRNT                             ;EXECUTE
1039
1040 003220 012737 000200 003400                2S:     MOV      #BIT7,100S                       ;LOAD STARTING BASE LINE
1041 003226 012700 015562                        MOV      #BUFFER,R0                            ;LOAD OUTPUT BUFFER POINTER
1042 003232 112720 000033                        MOV      #ESC,(R0)+                          ;ENTER VT-55 FORMAT
1043 003236 112720 000061                        MOV      #61,(R0)+                          ;LOAD ENTER '01' CODE
1044 003242 112720 000101                        MOV      #LDE0,(R0)+                          ;LOAD ENABLE 0
1045 003246 112720 000040                        MOV      #BITS,(R0)+                          ;DISABLE DISPLAY
1046 003252 112720 000111                        MOV      #LDE1,(R0)+                          ;LOAD ENABLE 1
1047 003256 112720 000060                        MOV      #BITS!BIT4,(R0)+                    ;CLEAR GRAPH, LINES AND CURSORS
1048
1049 003262 112720 000101                        MOV      #LDE0,(R0)+                          ;LOAD ENABLE 0
1050 003266 112720 000043                        MOV      #BITS!BIT1!BIT0,(R0)+              ;LOAD DISPLAY ENABLE AND GRAPH 0 ON
1051 003272 112720 000100                        MOV      #LNO,(R0)+                          ;NOP
1052 003276 112720 000110                        MOV      #LSC,(R0)+                          ;LOAD STARTING COORD.
1053 003302 012737 000000 007772                MOV      #0,BASE                             ;GET BASE LINE
1054 003310 004737 007776                        JSR      PC,SHUFF                             ;CONVERT
1055 003314 010120                                MOV      R1,(R0)+                             ;SAVE COORD.
1056 003316 112720 000100                        MOV      #LNO,(R0)+                          ;LOAD NOP
1057 003322 112720 000102                        MOV      #LDG0,(R0)+                          ;LOAD "LOAD GRAPH"
1058 003326 013737 003400 007772                MOV      100S,BASE                            ;LOAD THE STARTING DATA VALUE
1059 003334 004737 007776                        JSR      PC,SHUFF                             ;SHUFFEL THE DATA INTO VT-55 FORMAT
1060 003340 012737 001000 003402                1S:     MOV      #MAXVRT,101S                       ;LOAD COUNTER
1061 003346 010120                                MOV      R1,(R0)+                             ;SAVE THE LSB MSB BYTE
1062 003350 005337 003402                        DEC      101S                                ;DONE FULL GRAPH
1063 003354 001374
1064
1065
1066 003356 105020                                CLR      (R0)+                                ;LOAD TERMINATOR
1067 003360 004737 010074                        JSR      PC,XPRNT                             ;DISPLAY
1068 003364 006037 003400                        ROR      100S                                ;CHANGE DATA VALUE
1069 003370 001316                                BNE     2S
1070 003372 004737 010742                        JSR      PC,DELAY
1071 003376 000402                                BR      TST4                                ;;NEXT TEST
1072
1073 003400 000000                                100S:  0
1074 003402 000000                                101S:  0

```

```

1075
1076
1077
1078 003404 000004
1079 003406 004537 011070
1080 003412 012154
1081 003414 012700 015562
1082 003420 112720 000033
1083 003424 112720 000061
1084 003430 112720 000101
1085 003434 112720 000040
1086 003440 112720 000111
1087 003444 112720 000060
1088 003450 105020
1089 003452 004737 010074
1090
1091 003456 012737 000200 003636
1092 003464 012700 015562 25:
1093 003470 112720 000033
1094 003474 112720 000061
1095 003500 112720 000101
1096 003504 112720 000040
1097 003510 112720 000111
1098 003514 112720 000060
1099 003520 112720 000101
1100 003524 112720 000045
1101 003530 112720 000100
1102 003534 112720 000110
1103 003540 012737 000000 007772
1104 003546 004737 007776
1105 003552 010120
1106 003554 112720 000100
1107 003560 112720 000112
1108 003564 013737 003636 007772
1109 003572 004737 007776
1110 003576 012737 001000 003640
1111 003604 010120 15:
1112 003606 005337 003640
1113 003612 001374
1114
1115
1116 003614 105020
1117 003616 004737 010074
1118 003622 006037 003636
1119 003626 001316
1120 003630 004737 010742
1121 003634 000402
1122
1123 003636 000000 1005: 0
1124 003640 000000 1015: 0

```

```

*****
:TEST 4 D GRAPH 1 DISPLAY A STEPPING HORIZONTAL LINE
*****
TST4: SCOPE
JSR RS,AMSG ;DISPLAY HEADER
SHL 1
MOV @BUFFER,RO ;LOAD OUTPUT BUFFER POINTER
MOV @ESC,(RO)+ ;ENTER VT-55 MODE
MOV @61,(RO)+
MOV @LDED,(RO)+ ;LOAD ENABLE 0
MOV @BITS,(RO)+ ;DISABLE DISPLAY
MOV @LDE1,(RO)+ ;LOAD ENABLE 1
MOV @BITS!BIT4,(RO)+ ;CLEAR GRAPH, LILES, AND CURSORS
CLRB (RO)+ ;LOAD TERMINATOR
JSR PC,XPRNT ;EXECUTE

MOV @BIT7,1005 ;LOAD STARTING BASE LINE
MOV @BUFFER,RO ;LOAD OUTPUT BUFFER POINTER
MOV @ESC,(RO)+ ;ENTER VT55 MODE
MOV @61,(RO)+ ;LOAD ENTER "01" CODE
MOV @LDED,(RO)+ ;LOAD ENABLE 0
MOV @BITS,(RO)+ ;DISABLE DISPLAY
MOV @LDE1,(RO)+ ;LOAD ENAB:E 1
MOV @BITS!BIT4,(RO)+ ;CLEAR GRAPH,LINES MAD CURSORS
MOV @LDED,(RO)+ ;LOAD ENABLE 0
MOV @BITS!BIT2!BIT0,(RO)+ ;LOAD DISPLAY ENABLE AND GRAPH 1 ON
MOV @LNO,(RO)+ ;NOP
MOV @LSC,(RO)+ ;LOAD STARTING COORD.
MOV @0,BASE ;GET BASE LINE
JSR PC,SHUFF ;CONVERT
MOV @R1,(RO)+ ;SAVE COORD.
MOV @LNO,(RO)+ ;LOAD NOP
MOV @LDG1,(RO)+ ;LOAD "LOAD GRAPH"
MOV @1005,BASE ;LOAD THE STARTING DATA VALUE
JSR PC,SHUFF ;SHUFFEL THE DATA INTO VT-55 FORMAT
MOV @MAXVRT,1015 ;LOAD COUNTER
MOV @R1,(RO)+ ;SAVE THE LSB MSB BYTE
DEC @1015 ;DONE FULL GRAPH
BNE @15

CLRB (RO)+ ;LOAD TERMINATOR
JSR PC,XPRNT ;DISPLAY
ROR @1005
BNE @25 ;NO
JSR PC,DELAY
BR TST5 ;;NEXT TEST

1005: 0
1015: 0

```

```

1125
1126
1127
1128 003642 000004
1129 003644 004537 011070
1130 003650 012232
1131
1132 003652 012700 015562
1133 003656 112720 000033
1134 003662 112720 000061
1135 003668 112720 000101
1136 003672 112720 000040
1137 003676 112720 000111
1138 003702 112720 000060
1139 003706 105020
1140 003710 004737 010074
1141
1142 003714 004537 007242
1143 003720 002 102
1144 003722 000000
1145 003724 004737 010074
1146
1147 003730 004537 007242
1148 003734 006 112
1149 003736 100354
1150
1151 003740 112720 000033
1152 003744 112720 000135
1153 003750 112720 000100
1154 003754 112720 000100
1155 003760 112720 000100
1156 003764 112720 000100
1157 003770 112720 000100
1158 003774 112720 000100
1159 004000 112720 000100
1160 004004 112720 000100
1161 004010 112720 000100
1162 004014 112720 000100
1163 004020 112720 000100
1164 004024 112720 000100
1165 004030 112720 000100
1166 004034 112720 000100
1167 004040 112720 000100
1168 004044 112720 000100
1169 004050 105020
1170 004052 004737 010074
1171 004056 004737 010742

*****
:TEST 5 E GRAPH 0 AND 1
*****
TSTS: SCOPE
      JSR RS,AMSG ;DISPLAY HEADER
      GR0A1
      MOV #BUFFER,RO ;LOAD OUTPUT BUFFER POINTER
      MOVB #ESC,(RO)+ ;ENTER VTSS MODE
      MOVB #B1,(RO)+ ;ENTER "01" CODE
      MOVB #LDE0,(RO)+ ;LOAD ENABLE 0
      MOVB #BITS,(RO)+ ;DISABLE DISPLAY
      MOVB #LDE1,(RO)+ ;LOAD ENABLE 1
      MOVB #BITS!BIT4,(RO)+ ;CLEAR GRAPH, LINES AND CURSORS
      CLRB (RO)+ ;LOAD TERMINATOR
      JSR PC,XPRNT ;EXECUTE

      JSR RS,UPDN ;LOAD DATA PATTERN
      .BYTE BIT1,LG0 ;GRAPH 0 INC. PAT.
      .WORD 0
      JSR PC,XPRNT

      JSR RS,UPDN ;LOAD DATA PATTERN
      .BYTE BIT2:BIT1,LG1 ;GRAPH 0 DEC. PAT.
      .WORD BIT15:MAXH0Z ;

      MOVB #ESC,(RO)+ ;LOAD "ESC" CODE
      MOVB #COPY,(RO)+ ;LOAD COPY SCREEN
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
      CLRB (RO)+ ;LOAD TERMINATOR
      JSR PC,XPRNT ;EXECUTE
      JSR PC,DELAY

```

```

1172
1173
1174
1175 004063 000004
1176 004064 004537 011070
1177 004070 012276
1178 004072 012700 015562
1179 004076 112720 000033
1180 004102 112720 000061
1181 004106 112720 000101
1182 004112 112720 000040
1183 004116 112720 000111
1184 004122 112720 000060
1185 004126 105020
1186 004130 004737 010074
1187
1188 004134 012737 000200 004274
1189
1190 004142 012700 015562
1191 004146 112720 000033
1192 004152 112720 000061
1193 004156 112720 000101
1194 004162 112720 000053
1195 004166 112720 000100
1196 004172 112720 000110
1197 004176 012737 000000 007772
1198 004204 004737 007776
1199 004210 010120
1200 004212 112720 000100
1201 004216 112720 000102
1202 004222 013737 004274 007772
1203 004230 004737 007776
1204 004234 012737 001000 004276
1205 004242 010120
1206 004244 005337 004276
1207 004250 001374
1208
1209
1210 004252 105020
1211 004254 004737 010074
1212 004260 006037 004274
1213 004264 001326
1214 004266 004737 010742
1215 004272 000402
1216
1217 004274 000000
1218 004276 000000

*****
:TEST 6 F GRAPH 0 DISPLAY A STEPPING HISTOGRAM LINE
*****
TST6: SCOPE
      JSR RS,AMSG ;DISPLAY HEADER
      SHGLD
      MOV #BUFFER,RO ;LOAD OUTPUT BUFFER POINTER
      MOVB #ESC,(RO)+ ;ENTER VT55 MODE
      MOVB #61,(RO)+ ;ENTER "01" CODE
      MOVB #LDE0,(RO)+ ;LOAD ENABLE 0
      MOVB #BITS,(RO)+ ;DISABLE DISPLAY
      MOVB #LDE1,(RO)+ ;LOAD ENABLE 1
      MOVB #BITS!BIT4,(RO)+ ;CLEAR GRAPH, LINES AND CURSORS
      CLRB (RO)+ ;LOAD TERMINATOR
      JSR PC,XPRNT ;EXECUTE

      MOV #BIT7,100S ;LOAD STARTING BASE LINE

2S: MOV #BUFFER,RO ;LOAD THE STARTING ADDRESS
     MOVB #ESC,(RO)+ ;LOAD "ESC" CODE
     MOVB #61,(RO)+ ;LOAD "01" ENTER CODE
     MOVB #LDE0,(RO)+ ;LOAD ENABLE 0
     MOVB #BITS!BIT3!BIT1!BIT0,(RO)+ ;LOAD DISP. ENABLE , GRAPH 0, HISTO 1 ON
     MOVB #LNO,(RO)+ ;LOAD NOP
     MOVB #LSC,(RO)+ ;LOAD STARTING COORD.
     MOV #0,BASE ;GET BASE LINE
     JSR PC,SHUFF ;CONVERT
     MOV R1,(RO)+ ;SAVE COORD.
     MOVB #LNO,(RO)+ ;LOAD NOP
     MOVB #LDC0,(RO)+ ;LOAD "LOAD GRAPH"
     MOV #100S,BASE ;LOAD THE STARTING DATA VALUE
     JSR PC,SHUFF ;SHUFFLE THE DATA INTO VT-55 FORMAT
     MOV #0,COUNTER ;LOAD COUNTER
     MOVB #0,R1 ;SAVE THE LSB MSB BYTE
     DEC #101S ;DONE FULL GRAPH
     BNE #1S

1S: CLRB (RO)+ ;LOAD TERMINATOR
     JSR PC,XPRNT ;DISPLAY
     ROR #100S ;CHANGE DATA VALUE
     BNE #2S ;NO
     JSR PC,DELAY
     BR TST7 ;;NEXT TEST

100S: 0
101S: 0

```

E03

MAINDEC-11-DZVTD-8
DZVTD8.SRC T7

MACY11 27(732) 25-SEP-76 10:10 PAGE 31
G GRAPH 1 DISPLAY A STEPPING HISTOGRAM LINE

```

1219
1220
1221
1222 004300 000004
1223 004302 004537 011070
1224 004306 012347
1225 004310 012700 015562
1226 004314 112720 000033
1227 004320 112720 000061
1228 004324 112720 000101
1229 004330 112720 000040
1230 004334 112720 000111
1231 004340 112720 000060
1232 004344 105020
1233 004346 004737 010074
1234
1235 004352 012737 000200 004512
1236
1237 004360 012700 015562
1238 004364 112720 000033
1239 004370 112720 000061
1240 004374 112720 000101
1241 004400 112720 000065
1242 004404 112720 000100
1243 004410 112720 000110
1244 004414 012737 000000 007772
1245 004422 004737 007776
1246 004426 010120
1247 004430 112720 000100
1248 004434 112720 000112
1249 004440 013737 004512 007772
1250 004446 004737 007776
1251 004450 012737 001000 004514
1252 004460 010120
1253 004464 005337 004514
1254 004468 001374
1255
1256 004470 105020
1257 004472 004737 010074
1258 004476 006037 004512
1259 004502 001326
1260 004504 004737 010742
1261 004510 000402
1262
1263
1264 004512 000000
1265 004514 000000
1266

```

```

*****
:TEST 7 G GRAPH 1 DISPLAY A STEPPING HISTOGRAM LINE
*****
TST7: SCOPE
JSR RS,AMSG ;DISPLAY HEADER
SHGL1
MOV #BUFFER,RO ;LOAD OUTPUT BUFFER POINTER
MOVB #ESC,(RO)+ ;ENTER VT55 MODE
MOVB #61,(RO)+ ;ENTER "01" CODE
MOVB #LDED,(RO)+ ;LOAD ENABLE 0
MOVB #BITS,(RO)+ ;DISABLE DISPLAY
MOVB #LDE1,(RO)+ ;LOAD ENALBE 1
MOVB #BITS!BIT4,(RO)+ ;CLEAR GRAPH LINES AND CURSRS
CLRB (RO)+ ;LOAD TERMINATOR
JSR PC,XPRNT ;EXECUTRE
MOV #BIT7,100S ;LOAD STARTING BASE LINE
2S: MOV #BUFFER,RO ;LOAD THE STARTING ADDRESS
MOVB #ESC,(RO)+ ;LOAD "ESC" CODE
MOVB #61,(RO)+ ;LOAD "01" ENTER CODE
MOVB #LDED,(RO)+ ;LOAD ENABLE 0
MOVB #BITS!BIT4!BIT2!BIT0,(RO)+ ;LOAD DISPLAY ENABLE AND GRAPH 1 ON
MOVB #LNO,(RO)+ ;LOAD NOP
MOVB #LSC,(RO)+ ;LOAD STARTING COORD.
MOV #0,BASE ;GET BASE LINE
JSR PC,SHUFF ;CONVERT
MOV R1,(RO)+ ;SAVE COORD.
MOVB #LNO,(RO)+ ;LOAD NOP
MOVB #LDC1,(RO)+ ;LOAD "LOAD GRAPH"
MOV 100S,BASE ;LOAD THE STARTING DATA VALUE
JSR PC,SHUFF ;SHUFFEL THE DATA INTO VT-55 FORMAT
MOV #MAXVRT,101S ;LOAD COUNTER
1S: MOV R1,(RO)+ ;SAVE THE LSB MSB BYTE
DEC 101S ;DONE FULL GRAPH
BNE 1S
CLRB (RO)+ ;LOAD TERMINATOR
JSR PC,XPRNT ;DISPLAY
ROR 100S ;NO
BNE 2S
JSR PC,DELAY
BR TST10 ;;NEXT TEST
100S: 0
101S: 0

```

F03

MAINDEC-11-DZVTD-B
DZVTDB.SRC T7

MACY11 27(732) 25-SEP-76 10:10 PAGE 32
G GRAPH 1 DISPLAY A STEPPING HISTOGRAM LINE

1267					
1268					
1269					
1270					
1271	004516	000004			
1272	004520	004537	011070		
1273	004524	012420			
1274	004528	012700	015562		
1275	004532	112720	000033		
1276	004536	112720	000061		
1277	004540	112720	000101		
1278	004544	112720	000040		
1279	004548	112720	000111		
1280	004552	112720	000060		
1281	004556	105020			
1282	004560	004737	010074		
1283					
1284	004570	004537	007242		
1285	004574	012	102		
1286	004578	000000			
1287	004600	004737	010074		
1288					
1289	004604	004537	007242		
1290	004610	036	112		
1291	004612	100354			
1292					
1293	004614	112720	000033		
1294	004620	112720	000135		
1295	004624	112720	000100		
1296	004630	112720	000100		
1297	004634	112720	000100		
1298	004640	112720	000100		
1299	004644	112720	000100		
1300	004650	112720	000100		
1301	004654	112720	000100		
1302	004660	112720	000100		
1303	004664	112720	000100		
1304	004670	112720	000100		
1305	004674	112720	000100		
1306	004700	112720	000100		
1307	004704	112720	000100		
1308	004710	112720	000100		
1309	004714	112720	000100		
1310	004720	112720	000100		
1311	004724	105020			
1312	004728	004737	010074		
1313	004732	004737	010742		
1314					

```

*****
:TEST 10      H      HISTOGRAM ON GRAPH 0 AND 1
*****
TST10: SCOPE
        JSR      RS,AMSG          ;DISPLAY HEADER
        HGOA1
        MOV      @BUFFER,RO      ;LOAD OUTPUT BUFFER POINTER
        MOVB    @ESC,(RO)+      ;LOAD VT55 MODE
        MOVB    @61,(RO)+      ;ENTER "01" CODE
        MOVB    @LDED,(RO)+     ;LOAD ENABLE 0
        MOVB    @BITS,(RO)+     ;DISABLE DISPLAY
        MOVB    @LDE1,(RO)+    ;LOAD ENABLE 1
        MOVB    @BITS!BIT4,(RO)+ ;CLEAR GRAPH, LINES NAD CURSRS
        CLRB    (RO)+          ;LOAD TERMINATOR
        JSR      PC,XPRNT       ;EXECUTE

        JSR      RS,UPDOWN      ;LOAD DATA PATTERN
        .BYTE   BIT3!BIT1,LDC0  ;GRAPH 0 INC. PAT.
        .WORD   0
        JSR      PC,XPRNT

        JSR      RS,UPDOWN      ;LOAD DATA PATTERN
        .BYTE   BIT4!BIT3!BIT2!BIT1,LDC1 ;GRAPH 1
        .WORD   BIT15!MAXH0Z

        MOVB    @ESC,(RO)+      ;LOAD "ESC" CODE
        MOVB    @COPY,(RO)+     ;LOAD COPY SCREEN
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        MOVB    @LNO,(RO)+     ;LOAD NOP TO LOAD SILO
        CLRB    (RO)+          ;TERMINATOR
        JSR      PC,XPRNT       ;EXECUTE
        JSR      PC,DELAY

```



```

1315
1316
1317
1318 004736 000004
1319 004740 004537 011070
1320 004744 012461
1321 004746 012700 015562
1322 004752 112720 000033
1323 004756 112720 000061
1324 004762 112720 000101
1325 004766 112720 000040
1326 004772 112720 000111
1327 004776 112720 000060
1328 005002 105020
1329 005004 004737 010074
1330
1331 005010 004537 007242
1332 005014 002 102
1333 005016 000000
1334 005020 004737 010074
1335
1336 005024 004537 007574
1337 005030 002 103
1338 005032 002000
1339
1340 005034 112720 000033
1341 005040 112720 000135
1342 005044 112720 000100
1343 005050 112720 000100
1344 005054 112720 000100
1345 005060 112720 000100
1346 005064 112720 000100
1347 005070 112720 000100
1348 005074 112720 000100
1349 005100 112720 000100
1350 005104 112720 000100
1351 005110 112720 000100
1352 005114 112720 000100
1353 005120 112720 000100
1354 005124 112720 000100
1355 005130 112720 000100
1356 005134 112720 000100
1357 005140 112720 000100
1358 005144 105020
1359 005146 004737 010074
1360
1361 005152 004537 007574
1362 005156 002 103
1363 005160 100000
1364 005162 004737 010074
1365 005166 004737 010742
1366

```

```

*****
*TEST 11 I CURSORS ON GRAPH 0
*****
*ST11: SCOPE
JSR RS,AMSG ;DISPLAY HEADER
CURGRD
MOV #BUFFER,RO ;LOAD OUPUT BUFFER POINTER
MOVB #ESC,(RO)+ ;LOAD VISS MODE
MOVB #51,(RO)+ ;ENTER "01" CODE
MOVB #LDE0,(RO)+ ;LOAD ENABLE 0
MOVB #BITS,(RO)+ ;DISABLE DISPLAY
MOVB #LDE1,(RO)+ ;LOAD ENABLE 1
MOVB #BITS!BIT4,(RO)+ ;CLEAR GRAPH, LINES MAD CURSRS
CLRB (RO)+ ;LOAD TERMINATOR
JSR PC,XPRNT ;EXECUTE

JSR RS,UPDMN ;LOAD DATA PATTERN
.BYTE BIT1,LDCO ;GRAPH 0
.WORD 0 ;DATA TO BE LOADED
JSR PC,XPRNT ;EXECUTE IT

JSR RS,CURSOR ;ENABLE CURSORS
.BYTE BIT1,LDCO ;SC = 0 GRAPH 0
.WORD ADDLIN

MOVB #ESC,(RO)+ ;LOAD "ESC" CODE
MOVB #COPY,(RO)+ ;LOAD COPY SCREEN
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
MOVB #LNO,(RO)+ ;LOAD NOP TO LOAD SILO
CLRB (RO)+ ;LOAD TERMINATOR
JSR PC,XPRNT ;EXECUTE

JSR RS,CURSOR ;REMOVE CURSORS
.BYTE BIT1,LDCO ;ON GRAPH 0
.WORD BIT15
JSR PC,XPRNT ;EXECUTE
JSR PC,DELAY

```

1367
1368
1369
1370
1371 005172 000004
1372 005174 004537 011070
1373 005200 012512
1374 005202 012700 015562
1375 005206 112720 000033
1376 005212 112720 000061
1377 005216 112720 000101
1378 005222 112720 000040
1379 005226 112720 000111
1380 005232 112720 000060
1381 005236 105020
1382 005240 004737 010074
1383
1384 005244 004537 007242
1385 005250 004 112
1386 005252 100354
1387 005254 004737 010074
1388
1389 005260 004537 007574
1390 005264 004 113
1391 005266 002000
1392
1393 005270 112720 000033
1394 005274 112720 000135
1395 005300 112720 000100
1396 005304 112720 000100
1397 005310 112720 000100
1398 005314 112720 000100
1399 005320 112720 000100
1400 005324 112720 000100
1401 005330 112720 000100
1402 005334 112720 000100
1403 005340 112720 000100
1404 005344 112720 000100
1405 005350 112720 000100
1406 005354 112720 000100
1407 005360 112720 000100
1408 005364 112720 000100
1409 005370 112720 000100
1410 005374 112720 000100
1411 005400 105020
1412 005402 004737 010074
1413 005406 004537 007574
1414 005412 004 113
1415 005414 100000
1416 005416 004737 010074
1417 005422 004737 010742

: TEST 12 J CURSORS ON GRAPH 1

```

TST12: SCOPE
      JSR   RS,AMSG                ;DISPLAY HEADER
      CURGR1
      MOV   #BUFFER,RO            ;LOAD OUTPUT BUFFER POINTER
      MOVB #ESC,(RO)+             ;ENTER VT-55 MODE
      MOVB #61,(RO)+
      MOVB #LDED,(RO)+           ;LOAD ENABLE 0
      MOVB #BITS,(RO)+          ;DISABLE DISPLAY
      MOVB #LDE1,(RO)+          ;LOAD ENABLE 1
      MOVB #BITS!BIT4,(RO)+     ;CLEAR GRAPH, LILES, AND CURSORS
      CLRB (RO)+                 ;LOAD TERMINATOR
      JSR   PC,XPRNT              ;EXECUTE

      JSR   RS,UPDN               ;LOAD DATA PATTERN
      .BYTE BIT2,LDC1            ;GRAPH 1 DECREMENTING PAT.
      .WORD BIT15!MAXHOZ        ;MAX #
      JSR   PC,XPRNT              ;EXECUTE

      JSR   RS,CURSOR            ;ENABLE CURSORS
      .BYTE BIT2,LDC1
      .WORD ADDLIN

      MOVB #ESC,(RO)+           ;LOAD "ESC" CODE
      MOVB #COPY,(RO)+         ;LOAD COPY SCREEN
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      MOVB #LNO,(RO)+          ;LOAD NOP TO LOAD SILO
      CLRB (RO)+                 ;LOAD TERMINATOR
      JSR   PC,XPRNT              ;EXECUTE
      JSR   RS,CURSOR            ;REMOVE CURSOR ON GRAPH 1
      .BYTE BIT2,LDC1
      .WORD BIT15
      JSR   PC,XPRNT              ;EXECUTE
      JSR   PC,DELAY

```



```

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

```

MAINDEC-11-DZVTD-B
DZVTD8.SRC T14

MACY11 27(732) 25-SEP-76 10:10 PAGE 37
L TEST STARTING COORDINATE ON GRAPH 1

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

```

1544
1545
1546
1547
1548 006222 000004
1549
1550
1551 006224 012702 000014
1552 006230 012700 015562
1553 006234 112720 000033
1554 006240 112720 000061
1555 006244 112720 000101
1556 006250 112720 000040
1557 006254 112720 000111
1558 006260 112720 000060
1559 006264 112720 000033
1560 006270 112720 000062
1561 006274 105020
1562 006276 004737 010074
1563 006302 012700 015562
1564 006306 112720 000015
1565 006312 112720 000012
1566 006316 112720 000015
1567 006322 112720 000012
1568 006326 012701 000050
1569 006332 012720 044110
1570 006336 005301
1571 006340 001374
1572 006342 105020
1573 006344 004737 010074
1574 006350 005302
1575 006352 001374
1576
1577
1578
1579 006354 012700 015562
1580 006360 112720 000033
1581 006364 112720 000061
1582 006370 112720 000101
1583 006374 112720 000041
1584 006400 112720 000111
1585 006404 112720 000043
1586 006410 112720 000100
1587 006414 112720 000104
1588 006420 012737 002002 007772
1589 006426 004737 007776
1590 006432 010120
1591 006434 062737 000024 007772
1592 006442 022737 002340 007772
1593 006450 100366
1594 006452 105020
1595 006454 004737 010074

```

```

*****
: *TEST 15 M VT55 ADJUSTMENT PATTERN
*****

```

```

;TEST15: SCOPE
;FILL THE SCREEN WITH THE H CHARACTER

```

```

MOV #12, R2 ;LOAD COUNT
MOV #BUFFER, R0 ;LOAD BUFFER POINTER
MOVB #ESC, (R0)+ ;ENTER VT55 FORMAT
MOVB #61, (R0)+ ;LOAD ENTER "01" CODE
MOVB #LDE0, (R0)+ ;LOAD ENABLE 0
MOVB #BITS, (R0)+ ;DISABLE DISPLAY
MOVB #LDE1, (R0)+ ;LOAD ENABLE 1
MOVB #BITS:BIT4, (R0)+ ;CLEAR GRAPH, LILES, AND CURSORS
MOVB #ESC, (R0)+ ;ENTER VT55 FORMAT
MOVB #62, (R0)+ ;EXIT "02" CODE
CLRB (R0)+ ;LOAD TERMINATOR
JSR PC, XPRNT ;EXECUTE
MOV #BUFFER, R0 ;LOAD BUFFER POINTER
MOVB #15, (R0)+ ;LOAD CR
MOVB #12, (R0)+ ;LOAD LF
MOVB #15, (R0)+ ;LOAD CR
MOVB #12, (R0)+ ;LOAD LF
MOV #40, R1 ;LOAD SCREEN WIDTH
15: MOV #44110, (R0)+ ;LOAD ASCII H
DEC R1 ;FINISHED ?
BNE 15 ;BR IF NOT
CLRB (R0)+ ;LOAD TERM.
25: JSR PC, XPRNT ;XFER TO SCREEN
DEC R2 ;FINISHED ALL LINES
BNE 25 ;BR IF NOT

```

```

;NOW INSTALL THE HORIZONTAL LINES

```

```

MOV #BUFFER, R0 ;LOAD BUFFER POINTER
MOVB #33, (R0)+ ;ENABLE CHART MODE
MOVB #61, (R0)+
MOVB #LDE0, (R0)+ ;LOAD ENABLE 0
MOVB #BITS:BIT0, (R0)+ ;LOAD DISPLAY ENABLE
MOVB #LDE1, (R0)+ ;LOAD ENABLE 1
MOVB #BITS:BIT1:BIT0, (R0)+ ;ENABLE HORIZ.+ VERT LINES
MOVB #LNO, (R0)+
MOVB #LHVD, (R0)+ ;LOAD HORIZ INST
35: MOV #ADDLIN+2, BASE ;LOAD BASE LINE VALUE
JSR PC, SHUFF ;SHUFFEL THE DATA
MOV R1, (R0)+ ;SAVE THE DATA
ADD #24, BASE ;UPDATE BASE LINE VALUE
CMP #ADDLIN+340, BASE ;TEST FOR GREATER THAN VALID
BPL 35 ;BR IF OK
CLRB (R0)+ ;LOAD TERM
JSR PC, XPRNT ;XMIT TO THE SCREEN

```



```

1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647 006674
1648 006674 000004
1649 006676 005737 001310
1650 006702 001411
1651 006704 023737 001310 001312
1652 006712 001405
1653 006714 062737 000010 001312
1654 006722 000137 002016
1655 006726 000240
1656 006730 005037 001102
1657 006734 005237 001202
1658 006740 042737 100000 001202
1659 006746 005327
1660 006750 000001
1661 006752 003015
1662 006754 012737
1663 006756 000001
1664 006760 006750
1665 006762 104400 007015
1666 006766 013700 000042
1667 006772 001405
1668 006774 000005
1669 006776 004710
1670 007000 000240
1671 007002 000240
1672 007004 000240
1673 007006
1674 007006 000137
1675 007010 002010
1676 007012 377 377 000
1677 007015 015 042412 042116
1678 007022 050040 051501 000123
1679

```

.SBTTL END OF PASS ROUTINE

```

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

```

```

SEOP:
SCOPE
TST LAST ;TEST IF MORE
BEQ IS ;BR IF NONE
CMP LAST,VTNOW ;IS THIS THE LAST ONE
BEQ IS ;BR IF YES
ADD #10,VTNOW
JMP RSTRT ;TEST NEXT ONE

IS:
NOP
CLR $TSTNM ;;ZERO THE TEST NUMBER
INC $PASS ;;INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;;LOOP?

SEOPCT: .WORD 1
BGT $DOAGN ;;YES
MOV (PC)+,2(PC)+ ;;RESTORE COUNTER

SENDCT: .WORD 1
TYPE SENDMG ;;TYPE "END PASS"
MOV #42,RD ;;GET MONITOR ADDRESS
BEQ $DOAGN ;;BRANCH IF NO MONITOR
RESET ;;CLEAR THE WORLD
SENDAD: JSR PC,(RD) ;;GO TO MONITOR
NOP ;;SAVE ROOM
NOP ;;FOR
NOP ;;ACT11

$DOAGN: JMP 2(PC)+ ;;RETURN

SRTNAD: .WORD RSTRTA
SENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
SENDMG: .ASCIZ <15><12>/END PASS/

```


1680	007030	000	003	004	.BYTE	000,003,004,004,005
1681	007033	004	005			
1682	007035	006	007	010	.BYTE	006,007,010,012,014,015,017,021,024,027,032,035,040
1683	007040	012	014	015		
1684	007043	017	021	024		
1685	007046	027	032	035		
1686	007051	040				
1687	007052	043	047	053	.BYTE	043,047,053,056,062,066,073,077,103,110,115,121,126,133
1688	007055	056	062	066		
1689	007060	073	077	103		
1690	007063	110	115	121		
1691	007066	126	133			
1692	007070	137	144	151	.BYTE	137,144,151,156,162,167,174,201,206,213,217,223,230,235
1693	007073	156	162	167		
1694	007076	174	201	206		
1695	007101	213	217	223		
1696	007104	230	235			
1697	007106	241	245	251	.BYTE	241,245,251,255,261,265,271,274,277,303,305,310,313
1698	007111	255	261	265		
1699	007114	271	274	277		
1700	007117	303	305	310		
1701	007122	313				
1702	007123	316	317	321	.BYTE	316,317,321,323,324,326,327,330,331,331,331,331
1703	007126	323	324	326		
1704	007131	327	330	331		
1705	007134	331	331	331		
1706	007137	331				
1707	007140	331	327	326	.BYTE	330,327,326,325,323,322,320,317,314,312,307
1708	007143	331	323	322		
1709	007146	331	317	314		
1710	007151	312	307			
1711	007153	304	301	275	.BYTE	304,301,275,272,267,263,257,253,247,243,237,232,225
1712	007156	272	267	263		
1713	007161	257	253	247		
1714	007164	243	237	232		
1715	007167	225				
1716	007170	221	214	207	.BYTE	221,214,207,203,176,171,164,157,153,145,140
1717	007173	203	176	171		
1718	007176	164	157	153		
1719	007201	145	140			
1720	007203	134	127	121	.BYTE	134,127,121,115,111,104,077,073,066,063,057,053,047
1721	007206	115	111	104		
1722	007211	077	073	066		
1723	007214	063	057	053		
1724	007217	047				
1725	007220	043	040	034	.BYTE	043,040,034,031,027,023,021,017,015,013,011,010,007
1726	007223	031	027	023		
1727	007226	021	017	015		
1728	007231	013	011	010		
1729	007234	007				
1730	007236	006	005	004	.BYTE	006,005,004,003
1731	007240	003				
1732	007241	000				
1733						
1734						
1735						

SINEND: .BYTE 0
.EVEN
;UP-DOWN SUBROUTINE

1736	007242	012537	007432	UPDN:	MOV	(R5)+,11\$:GET GRAPH BIT AND INC/DEC WORD	
1737	007246	012537	007434		MOV	(R5)+,12\$:GET STARTING CORD.	
1738	007252	012700	015562		MOV	#BUFFER,R0	:LOAD THE POINTER	
1739	007256	112720	000033		MOVB	#ESC,(R0)+	:LOAD 'ESCAPE'	
1740	007262	112720	000061		MOVB	#61,(R0)+	:ENABLE GRAPHIC MODE	
1741	007266	112720	000101		MOVB	#LOEO,(R0)+	:LOAD ENABLE 0	
1742	007272	112710	000041		MOVB	#BITS!BIT0,(R0)	:ENABLE DISPLAY	
1743	007276	153720	007432		BISB	11\$,(R0)+	:LOAD GRAPH ENABLE BIT	
1744	007302	012737	000001	007430	MOV	#1,10\$		
1745	007310	112720	000100		MOVB	#LNO,(R0)+	:LOAD NOP	
1746	007314	112720	000110		MOVB	#LSC,(R0)+	:LOAD STARTING CORD.	
1747	007320	005037	007772		CLR	BASE	:LOAD START. CORD.	
1748	007324	004737	007776		JSR	PC,SHUFF		
1749	007330	010120			MOV	R1,(R0)+	:LOAD INTO BUFFER	
1750	007332	112720	000100		MOVB	#LNO,(R0)+	:LOAD NOP	
1751	007336	113720	007433		MOVB	11\$+1,(R0)+	:LOAD LOAD GRAPH X	
1752	007342	013737	007434	007772	4\$:	MOV	12\$,BASE	:LOAD DATA
1753	007350	004737	007776	1\$:	JSR	PC,SHUFF	:SHUFFEL DATA	
1754	007354	010120			MOV	R1,(R0)+	:LOAD DATA	
1755	007356	005737	007434		TST	12\$:TEST FOR UP OR DOWN DATA	
1756	007362	100007			BPL	2\$:BR IF INC.	
1757	007364	005337	007772		DEC	BASE	:DEC. DATA	
1758	007370	042737	177400	007772	BIC	#177400,BASE		
1759	007376	001407			BZQ	5\$		
1760	007400	000767			BR	1\$		
1761	007402	005237	007772	2\$:	INC	BASE	:CHANGE DATA	
1762	007406	022737	000354	007772	3\$:	CMF	#MAXHOZ,BASE	:TEST FOR LAST
1763	007414	001355			BNE	1\$:BR IF NOT	
1764	007416	005337	007430	5\$:	DEC	10\$		
1765	007422	100347			BPL	4\$:BR IF NOT	
1766	007424	105010			CLRB	(R0)	:LOAD TERMINATOR	
1767	007426	000205			RTS	R5	:EXIT	
1768	007430	000000		10\$:	0			
1769	007432	000000		11\$:	0			
1770	007434	000000		12\$:	0			

```

1771 ;STARTING COORDINATE SUBROUTINE
1772
1773 007436 012537 007570 STCORD: MOV (RS)+,11$
1774 007442 012537 007572 MOV (RS)+,12$
1775 007448 012537 007566 MOV (RS)+,10$
1776 007452 010046 MOV RO,-(SP)
1777 007454 010146 MOV R1,-(SP)
1778 007456 012700 015562 MOV @BUFFER,RO
1779 007460 112720 000033 MOVB @ESC,(RO)+
1780 007466 112720 000061 MOVB @61,(RO)+
1781 007472 112720 000101 MOVB @LDED,(RO)+
1782 007476 112710 000041 MOVB @BITS!BIT0,(RO)
1783 007502 153720 007570 BISB 11$,(RO)+
1784 007506 112720 000100 MOVB @LNO,(RO)+
1785 007512 112720 000110 MOVB @LSC,(RO)+
1786 007516 013737 007566 007772 MOV 10$,BASE
1787 007524 004737 007776 JSR PC,SHUFF
1788 007530 010120 MOV R1,(RO)+
1789 007532 112720 000100 MOVB @LNO,(RO)+
1790 007536 113720 007571 MOVB 11$+1,(RO)+
1791 007542 013737 007572 007772 4$: MOV 12$,BASE
1792 007550 004737 007776 1$: JSR PC,SHUFF
1793 007554 010120 MOV R1,(RO)+
1794 007556 105010 CLRB (RO)
1795 007560 012601 MOV (SP)+,R1
1796 007562 012600 MOV (SP)+,RO
1797 007564 000205 RTS RS
1798 007566 000000
1799 007570 000000
1800 007572 000000

```

```

::GET GRAPH BIT AND INC/DEC WORD
::GET STARTING CORD.
::GET ARG. WORD
::SAVE RO
::SAVE R1
::LOAD THE POINTER
::LOAD 'ESCAPE'
::ENABLE GRAPHIC MODE
::LOAD ENABLE 0
::ENABLE DISPLAY
::LOAD GRAPH ENABLE BIT
::LOAD NOP
::LOAD STARTING CORD.
::LOAD START. CORD.

::LOAD INTO BUFFER
::LOAD NOP
::LOAD LOAD GRAPH X
::LOAD DATA
::SHUFFEL DATA
::LOAD DATA
::LOAD TERMINATOR

;EXIT

```

```

1801
1802
1803
1804 007574 012537 007766
1805 007600 012537 007770
1806 007604 012700 015562
1807 007610 112720 000033
1808 007614 112720 000061
1809 007620 112720 000101
1810 007624 112710 000041
1811 007630 153720 007766
1812 007634 112720 000111
1813 007640 013701 007766
1814 007644 006301
1815 007646 112710 000040
1816 007652 150120
1817 007654 112720 000100
1818 007660 112720 000110
1819 007664 005037 007772
1820 007670 004737 007776
1821 007674 010120
1822 007676 112720 000100
1823 007702 113720 007767
1824 007706 013737 007770 007772
1825 007714 004737 007776
1826 007720 010120
1827 007722 005737 007770
1828 007726 100007
1829 007730 005337 007772
1830 007734 042737 177000 007772
1831 007742 001407
1832 007744 000763
1833 007746 005237 007772
1834 007752 022737 003000 007772
1835 007760 001355
1836 007762 105010
1837 007764 000205
1838 007766 000000
1839 007770 000000
1840 007772 000000
1841 007774 000000
1842

```

;CURSOR SUBROUTINE

```

CURSOR: MOV (RS)+,11$ ;GET ARG. WORD
MOV (RS)+,12$ ;GET ARG. WORD
MOV @BUFFER,RO ;LOAD THE POINTER
MOVB @ESC,(RO)+ ;LOAD 'ESCAPE'
MOVB @51,(RO)+ ;ENABLE GRAPHIC MODE
MOVB @LED,(RO)+ ;LOAD ENABLE 0
MOVB @BITS!@BIT0,(RO) ;ENABLE DISPLAY
BISB 11$,(RO)+ ;LOAD GRAPH ENABLE BIT
MOVB @LDE1,(RO)+ ;LOAD ENABLE
MOV 11$,R1
ASL R1
MOVB @BITS,(RO)
BISB R1,(RO)+
MOVB @LNO,(RO)+ ;LOAD NOP
MOVB @LSC,(RO)+ ;LOAD STARTING CORD.
CLR BASE ;LOAD START. CORD.
JSR PC,SHUFF
MOV R1,(RO)+ ;LOAD INTO BUFFER
MOVB @LNO,(RO)+ ;LOAD NOP
MOVB 11$+1,(RO)+ ;LOAD "LOAD CURSOR ON GRAPH X"
MOV 12$,BASE ;LOAD CURSOR POSITION DATA
JSR PC,SHUFF ;SHUFFLE DATA
MOV R1,(RO)+ ;LOAD DATA
TST 12$ ;TEST FOR UP OR DOWN DATA
BPL 25$ ;BR IF INC.
DEC BASE
BIC @177000,BASE
BEQ 45$
BR 15$
25: INC BASE ;CHANGE DATA
35: CMP @ADOLIN!MAXVRT,BASE ;TEST FOR LAST
BNE 15$ ;BR IF NOT
45: CLRB (RO) ;LOAD TERMINATOR
RTS RS ;EXIT
11$: 0
12$: 0
BASE: 0
BASE1: 0

```

```

1843                                     ;SUBROUTINE TO SHUFFEL DATA INTO VT-55 DATA BYTE FORMAT
1844
1845 007776 013702 007772 SHUFF: MOV     BASE,R2           ;LOAD VALUE TO BE SHUFFELED
1846 010002 010237 007774     MOV     R2,BASE1
1847 010006 042737 177740 007774     BIC     #177740,BASE1
1848 010014 010201         MOV     R2,R1
1849 010016 006001         ROR     R1
1850 010020 006001         ROR     R1
1851 010022 006001         ROR     R1
1852 010024 006001         ROR     R1
1853 010026 006001         ROR     R1
1854 010030 110137 007775     MOVB    R1,BASE1+1       ;RELOAD R1
1855 010034 042737 170340 007774     BIC     #170340,BASE1   ;MASK
1856 010042 052737 020040 007774     BIS     #20040,BASE1    ;CONVERT TO ASCII
1857 010050 032737 002000 007772     BIT     #BIT10,BASE
1858 010056 001403         BEQ     IS              ;BR IF NOT SET
1859 010060 052737 010000 007774     BIS     #BIT12,BASE1    ;SET BIT
1860 010066 013701 007774     MOV     BASE1,R1
1861 010072 000207         RTS                    ;EXIT
    
```

```

1862
1863
1864 010074 010046
1865 010076 010146
1866 010100 005037 010546
1867 010104 005037 010552
1868 010110 005037 010550
1869 010114 012700 015562
1870 010120 105777 171204
1871 010124 100375
1872 010126 104405
1873 010130 000240
1874 010132 000240
1875 010134 000240
1876 010136 000240
1877 010140 000240
1878 010142 112001
1879 010144 001562
1880 010146 122701 000033
1881 010152 001031
1882 010154 122710 000135
1883 010160 001023
1884 010162 032777 004000 170750
1885 010170 001402
1886 010172 105720
1887 010174 000762
1888 010176 032777 002000 170734
1889 010204 001407
1890 010206 013737 001202 010542
1891 010214 042737 177770 010542
1892 010222 001363
1893 010224 105237 010550
1894 010230 005237 010546
1895 010234 000402
1896 010236 005037 010546
1897 010242 110177 171064
1898 010246 105777 171052
1899 010252 100322
1900 010254 005737 010546
1901 010260 001317
1902 010262 005037 001124
1903 010266 000420
1904

```

;DISPLAY SUBROUTINE

```

XPRNT:  MOV      RD,-(SP)
        MOV      R1,-(SP)
        CLR      ANESC           ;HOUSEKEEP
        CLR      TERM
        CLR      NOEXIT
        MOV      @BUFFER,R0     ;SETUP BUFFER POINTER
15:      TSTB     @VTOS         ;TEST READY
        BPL      IS
        CKSMR                    ;TEST IF "CTRL G" ON CONSOLE TTY
        NOP
        NOP
        NOP
        NOP
        NOP
25:      MOVB     (R0)+,R1       ;GET A CHAR.
        BEQ      16S           ;BR IF TERM
        CMPB     @33,R1        ;TEST FOR ESC
        BNE      45            ;BR IF NOT
        CMPB     @COPY,(R0)    ;TEST IF NEXT CHAR IS A "COPY"
        BNE      35            ;BR IF NOT
        BIT      @BIT11,@SMR   ;TEST IF "INHIBIT" COPIER TESTING
        BEQ      13S          ;BR IF NOT
21S:     TSTB     (R0)+         ;YES - ADJUST POINTER
        BR      2S
13S:     BIT      @BIT10,@SMR   ;TEST IF "SAVE PAPER MODE" IS ENABLED
        BEQ      20S          ;BR IF NOT
        MOV      @PASS,30S     ;GET PASS COUNT
        BIC      @177770,30S   ;MASK OF BITS
        BNE      21S          ;BR IF NOT 1 OR 8 PASSES
        SET     "DO NOT EXIT UNTILL X-ON"
35:      INC      ANESC         ;SET SOFT FLAG
        BR      5S
45:      CLR      ANESC         ;CLEAR SOFT FLAG
55:      MOVB     R1,@VTOS     ;LOAD CHAR
        TSTB     @VTIS        ;TEST INPUT FLAG
        BPL      IS           ;BR IF CLEARED
        TST      ANESC        ;TEST IF "ESC" WAS JUST SENT
        BNE      IS
        CLR      @GODAT       ;CLEAR EXPECTED DATA
        BR      10S

```

```

1905 ;WAIT FOR A KEYBOARD FLAG - ERROR REPORT IF NONE
1906
1907 010270 013737 001316 011162 6S: MOV TIME0,TIME1
1908 010276 005037 011164 CLR TIME2 ;LOAD DELAY
1909 010302 105777 171016 7S: TSTB 2VTIS ;TEST IF INPUT FLAG
1910 010306 100410 BMI 10S ;BR IF SET
1911 010310 005337 011164 DEC TIME2 ;DELAY
1912 010314 001372 BNE 7S
1913 010316 104405 CKSMR ;TEST FOR "CTRG G" ON CONSOLE TTY
1914 010320 005337 011162 DEC TIME1 ;DELAY
1915 010324 001366 BNE 7S
1916 010326 000457 BR 14S ;REPORT ERROR
1917
1918 ;INFUT FLAG SET - FIND OUT WHAT CHARACTER IT WAS
1919
1920 010330 017737 170772 001126 10S: MOV 2VTIB,SBDDAT ;READ CHAR
1921 010336 042737 177600 001126 BIC 8177600,SBDDAT ;MASK
1922 010344 022737 000021 001126 CMP 8XON,SBDDAT ;TEST FOR X ON
1923 010352 001006 BNE 11S
1924 010354 005037 010550 CLR NOEXIT
1925 010360 005737 010552 TST TERM ;TEST IF TERMINATOR WAS SET
1926 010364 001655 BEQ 1S ;BR IF NOT
1927 010366 000462 BR 17S ;BR IF ONLY ONE
1928 010370 022737 000023 001126 11S: CMP 8XOFF,SBDDAT ;TEST FOR X OFF
1929 010376 001006 BNE 12S ;BR IF NOT
1930 010400 105237 010550 INCB NOEXIT ;SET "NO EXIT UNTIL X-ON" RCVD
1931 010404 012737 000021 001124 MOV 8XON,SGDDAT ;LOAD EXPECTED VALUE
1932 010412 000726 BR 6S ;WAIT FOR X-ON
1933 010414 005037 001124 12S: CLR SGDDAT ;LOAD EXPECTED
1934 010420 105777 170514 TSTB 2SMR ;TEST SMR
1935 010424 100020 BPL 14S ;BR IF CLEARED
1936 010426 022737 000057 001126 CMP 8'/,SBDDAT ;COMPARE
1937 010434 001417 BEQ 15S ;BR IF EQUAL
1938 010436 022737 000134 001126 CMP 8'\,SBDDAT ;COMPARE
1939 010444 001010 BNE 14S ;BR IF NOT
1940 010446 012737 000001 010544 MOV 81,LOOP ;SET SOFT FLAG
1941 010454 012737 011277 010572 MOV 8M0,FINDTA ;SETUP MESSAGE
1942 010462 000137 010562 JMP FINDOT
1943
1944 010466 104001 14S: ERROR 1 ;UNEXPECTED OR INCORRECT INPUT FLAG
1945 010470 000137 010120 JMP 1S
1946
1947 010474 005037 010544 15S: CLR LOOP
1948 010500 012737 011362 010572 MOV 8M1,FINDTA ;SETUP MESSAGE
1949 010506 000137 010562 JMP FINDOT
1950 010512 012737 000001 010552 16S: MOV 81,TERM ;SET "TERMINATOR HAS BEEN FOUND" FLAG
1951 010520 012737 000021 001124 MOV 8XON,SGDDAT ;LOAD EXPECTED
1952 010526 105737 010550 TSTB NOEXIT ;TEST IF ALLOWED TO LEAVE ROUTINE ?
1953 010532 001256 BNE 6S ;BR IF NOT ALLOWED
1954 010534 012601 17S: MOV (SP)+,R1
1955 010536 012600 MOV (SP)+,R0
1956 010540 000207 RTS ;EXIT
1957
1958 010542 000000 30S: 0
1959 010544 000000 LOOP: 0
1960 010546 000000 MESC: 0

```

1961	010550	000000			NOEXIT: 0
1962	010552	000000			TERM: 0
1963					
1964	010554	004537	011070		FNDA: JSR RS,AMSG
1965	010560	011446			M02 ;ERROR ASK AGAIN
1966	010562	012706	001100		FINDOT: MOV #STACK,SP
1967	010566	004537	011070		JSR RS,AMSG
1968	010572	011362			FINDTA: M01
1969	010574	004737	011110		JSR PC,GETCHR
1970	010600	000770			BR FINDOT
1971					
1972	010602	105777	170522		IS: TSTB @VTOS
1973	010606	100375			BPL IS
1974	010610	110077	170516		MOVB RO,@VTOS
1975	010614	042700	177640		BIC #177640,RO ;MASK
1976	010620	122700	000101		CHPB #'A,RO ;TEST FOR NUMBER
1977	010624	101353			BHI FNDA
1978	010626	122700	000116		CHPB #'N,RO ;TEST FOR OTHERS
1979	010632	103750			BLO FNDA
1980	010634	042700	177740		BIC #177740,RO ;MAKE 0-37
1981	010640	005300			DEC RO
1982	010642	110037	001102		MOVB RO,\$YSTNM ;LOAD THAT TEST #
1983	010646	006300			ASL RO
1984	010650	005760	010676		TST DSPCH(RO) ;TEST IF VALID
1985	010654	001737			BEQ FNDA ;BR IF NOT
1986	010656	016037	010676	001106	MOV DSPCH(RO),SLPADR ;LOAD RETURN ADDRESS
1987	010664	016037	010676	001110	MOV DSPCH(RO),SLPERR ;LOAD ERROR LOOP ADDRESS
1988	010672	000170	010676		JMP @DSPCH(RO) ;GO TO THAT TEST
1989					
1990	010676	002066			DSPCH: TST1+2
1991	010700	002522			TST2+2
1992	010702	003150			TST3+2
1993	010704	003406			TST4+2
1994	010706	003644			TST5+2
1995	010710	004064			TST6+2
1996	010712	004302			TST7+2
1997	010714	004520			TST10+2
1998	010716	004740			TST11+2
1999	010720	005174			TST12+2
2000	010722	005430			TST13+2
2001	010724	005726			TST14+2
2002	010726	006224			TST15+2
2003	010730	000000			0
2004	010732	000000			0
2005	010734	000000			0
2006	010736	000000			0
2007	010740	000000			0


```

2008                                     ;PROGRAM DELAY ROUTINE
2009
2010 010742 013737 001320 011020 DELAY: MOV      SUBTST,10$      ;LOAD COUNT
2011 010750 005037 011022          CLR      11$
2012 010754 005737 001322          TST     WFTST          ;TEST IF H.F. MODE
2013 010760 001404          BEQ     2$            ;BR IF NOT
2014 010762 006237 011020          ASR     10$          ;CHANGE DELAY TIMER
2015 010766 006237 011020          ASR     10$
2016 010772 032777 010000 170140 2$: BIT     @BIT12,25MR    ;TEST SR
2017 011000 001006          BNE     3$            ;BR IF SET
2018 011002 005337 011022          DEC     11$          ;DELAY
2019 011006 001371          BNE     2$
2020 011010 005337 011020          DEC     10$
2021 011014 100366          BPL     2$            ;DELAY
2022 011016 000207          RTS     PC            ;EXIT
2023
2024 011020 000002          10$:   2
2025 011022 000000          11$:   0
2026
2027 011024 013737 001320 011064 ADELAY: MOV     SUBTST,10$
2028 011032 005037 011066          CLR     11$
2029 011036 006237 011064          ASR     10$
2030 011042 006237 011064          ASR     10$
2031 011046 005337 011066          2$:   DEC     11$
2032 011052 001375          BNE     2$
2033 011054 005337 011064          DEC     10$
2034 011060 100372          BPL     2$
2035 011062 000207          RTS     PC
2036 011064 000000          10$:   0
2037 011066 000000          11$:   0

```

```

2038
2039 ;HEADER SUBROUTINE FOR VT-50
2040
2041 011070 012537 011100      MSG:  MOV    (R5)+,10$      ;GET POINTER
2042 011074 004537 011166      JSR    R5,MT0B          ;MOVE TO BUFFER
2043 011100 000000
2044 011102 004737 010074      10$:  0
2045 011106 000205              JSR    PC,XPRNT        ;DISPLAY IT
2046
2047
2048 ;SUBROUTINE TO GET A CHARACTER FROM THE VT50
2049
2050 011110 013737 001316 011162 GETCHR: MOV    TIME0,TIME1      ;LOAD TIME COUNTER
2051 011116 005037 011164      CLR    TIME2
2052
2053 011122 105777 170176      1$:   TSTB   2VTIS          ;TEST INPUT STATUS
2054 011126 100005              BPL    2$                ;BR IF CLEARED
2055 011130 017700 170172      MOV    2VTIB,R0         ;READ A CHAR
2056 011134 062716 000002      ADD    2,(SP)          ;UPDATE RETURN
2057 011140 000207              RTS    PC                ;EXIT
2058
2059 011142 005337 011164      2$:   DEC    TIME2          ;DELAY
2060 011146 071365              BNE    1$
2061 011150 075337 011162      DEC    TIME1           ;FINISHED ?
2062 011154 100362              BPL    1$               ;LOOP TILL TIME EXPIRED
2063 011156 104002              ERROR  2                ;NO INPUT FLAG FROM DEVICE
2064 011160 000207              RTS    PC                ;EXIT
2065
2066 011162 000000      TIME1: 0
2067 011164 000000      TIME2: 0
2068
2069 ;MOVE TO THE OUTPUT BUFFER
2070
2071 011166 012500 015562      MT0B:  MOV    (R5)+,R0      ;LOAD DEST.
2072 011170 012701 015562      MOV    #BUFFER,R1        ;LOAD R1
2073 011174 112021 015562      1$:   MOVB   (R0)+,(R1)+    ;LOAD BYTE
2074 011176 001376 015562      BNE    1$                ;BR UNTIL DONE
2075 011200 000205              RTS    R5                ;EXIT
2076
2077 .SBTTL  ASCII MESSAGES

```

2078
2079
2080
2081 011202 031033 005015 050042
2082 011210 053517 051105 006442
2083 011216 000012
2084 011220 031033 006415 046412
2085 011226 044501 042116 041505
2086 011234 030455 026461 055104
2087 011242 052126 026504 020102
2088 011250 052126 032465 040440
2089 011256 041503 050105 040524
2090 011264 041516 020105 042524
2091 011272 052123 005015 000
2092 011277 033 015462 015510
2093 011304 046112 047517 020120
2094 011312 047117 052040 051505
2095 011320 020124 040520 052124
2096 011326 051105 020116 042514
2097 011334 052124 051105 024040
2098 011342 020101 044124 052522
2099 011350 046440 020051 020077
2100 011356 020075 000040
2101 011362 031033 044033 045033
2102 011370 052123 051101 020124
2103 011376 052101 052040 051505
2104 011404 020124 040520 052124
2105 011412 051105 020116 042514
2106 011420 052124 051105 024040
2107 011426 020101 044124 052522
2108 011434 046440 020051 020077
2109 011442 020075 000040
2110 011446 005015 052012 054522
2111 011454 040440 040507 047111
2112 011462 006440 000012
2113 011466 005015 042012 053105
2114 011474 041511 020105 042101
2115 011502 051104 051505 020123
2116 011510 020077 020075 000200
2117 011516 051105 047522 020122
2118 011524 046106 043501 051440
2119 011532 052105 047440 020116
2120 011540 051124 047101 046523
2121 011546 052111 042524 020122
2122 011554 052123 052101 051525
2123 011562 000
2124 011563 116 020117 047111
2125 011570 052520 020124 046106
2126 011576 043501 042040 052105
2127 011604 041505 042524 000104
2128 011612 047125 054105 042520
2129 011620 052103 042105 047440
2130 011626 020122 047111 047503
2131 011634 051122 041505 020124
2132 011642 047111 052520 020124
2133 011650 044103 051101 000

;ASCII MESSAGES

PWRMSG: .ASCIZ <33><62><15><12>/"POWER"/<15><12>

TITLE: .ASCIZ <33><62><15><15><12>/MAINDEC-11-DZVTD-B VT55 ACCEPTANCE TEST/<15><12>

MQ0: .ASCIZ <33><62><33><110><33><112>/LOOP ON TEST PATTERN LETTER (A THRU M) ? = /

MQ1: .ASCIZ <33><62><33><110><33><112>/START AT TEST PATTERN LETTER (A THRU M) ? =

MQ2: .ASCIZ <15><12><12>/TRY AGAIN /<15><12>

WHAT0: .ASCIZ <15><12><12>/DEVICE ADDRESS ? = /<.200>

EM1: .ASCIZ /ERROR FLAG SET ON TRANSMITTER STATUS/

EM2: .ASCIZ /NO INPUT FLAG DETECTED/

EM4: .ASCIZ /UNEXPECTED OR INCORRECT INPUT CHAR/

2134	011655	111	053116	046101	EM3:	.ASCIZ /INVALID BUS ADDRESS, TRY AGAIN/
2135	011662	042111	041040	051525		
2136	011670	040440	042104	042522		
2137	011676	051523	020054	051124		
2138	011704	020131	043501	044501		
2139	011712	000116				
2140	011714	051105	050122	020103	DH1:	.ASCIZ /ERRPC VTNOW TSTNUM/
2141	011722	020040	053040	047124		
2142	011730	053517	020040	020040		
2143	011736	051524	047124	046525		
2144	011744	000				
2145	011745	105	051122	041520	DH4:	.ASCIZ /ERRPC VTNOW TSTNUM EXPCT RECV/
2146	011752	020040	053040	047124		
2147	011760	053517	020040	052040		
2148	011766	052123	052516	020115		
2149	011774	020040	054105	041520		
2150	012002	020124	020040	042522		
2151	012010	053103	000			
2152	012013	033	015462	015510	DAHL:	.ASCIZ <33><62><33><110><33><112>/GROWING HORIZ. LINE/
2153	012020	043512	047522	044527		
2154	012026	043516	044040	051117		
2155	012034	055111	020056	044514		
2156	012042	042516	000			
2157	012045	033	015462	015510	DAVL:	.ASCIZ <33><62><33><110><33><112>/GROWING VERT. LINE/
2158	012052	043512	047522	044527		
2159	012060	043516	053040	051105		
2160	012066	027124	046040	047111		
2161	012074	000105				
2162	012076	033	062	033	SHLO:	.BYTE 33,62,33,110,33,112
2163	012101	110	033	112		
2164	012104	044504	050123	040514		.ASCIZ /DISPLAY STEPPING HORIZ. LINE ON GRAPH 0/
2165	012112	020131	052123	050105		
2166	012120	044520	043516	044040		
2167	012126	051117	055111	020056		
2168	012134	044514	042516	047440		
2169	012142	020116	051107	050101		
2170	012150	020110	000060			
2171	012154	033	062	033	SHL1:	.BYTE 33,62,33,110,33,112
2172	012157	110	033	112		
2173	012162	044504	050123	040514		.ASCIZ /DISPLAY STEPPING HORIZ. LINE ON GRAPH 1/
2174	012170	020131	052123	050105		
2175	012176	044520	043516	044040		
2176	012204	051117	055111	020056		
2177	012212	044514	042516	047440		
2178	012220	020116	051107	050101		
2179	012226	020110	000061			
2180	012232	033	062	033	GROA1:	.BYTE 33,62,33,110,33,112
2181	012235	110	033	112		
2182	012240	044504	050123	040514		.ASCIZ /DISPLAY DATA ON GRAPH 0 AND 1/
2183	012246	020131	040504	040524		
2184	012254	047440	020116	051107		
2185	012262	050101	020110	020060		
2186	012270	047101	020104	000061		
2187	012276	033	062	033	SHGLO:	.BYTE 33,62,33,110,33,112
2188	012301	110	033	112		
2189	012304	052123	050105	044520		.ASCIZ /STEPPING HISTOGRAM LINE ON GRAPH 0/

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

.SBTTL TTY INPUT ROUTINE

ENABL LSB

SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
WHEN OPERATING IN TTY FLAG MODE.

2246									
2247									
2248									
2249									
2250									
2251									
2252									
2253									
2254									
2255									
2256	012714	022737	000176	001140	SCKSMR:	CMP	#SMREG,SMR		:: IS THE SOFT-SMR SELECTED?
2257	012722	001074				BNE	155		:: BRANCH IF NO
2258	012724	105777	166214			TSTB	2STKS		:: CHAR THERE?
2259	012730	100071				BPL	155		:: IF NO, DON'T WAIT AROUND
2260	012732	117746	166210			MOVB	2STKB,-(SP)		:: SAVE THE CHAR
2261	012736	042716	177600			BIC	#1C177,(SP)		:: STRIP-OFF THE ASCII
2262	012742	022726	000007			CMP	#7,(SP)+		:: IS IT A CONTROL G?
2263	012746	001062				BNE	155		:: NO, RETURN TO USER
2264	012750	123727	001134	000001		CHPB	\$AUTOB,#1		:: ARE WE RUNNING IN AUTO-MODE?
2265	012756	001456				BEQ	155		:: BRANCH IF YES
2266									
2267	012760	104400	013441			TYPE	,SCNTLG		:: ECHO THE CONTROL-G (?G)
2268	012764	104400	013446		SGTSMR:	TYPE	,SMSMR		:: TYPE CURRENT CONTENTS
2269	012770	013746	000175			MOV	SMREG,-(SP)		:: SAVE SMREG FOR TYPEOUT
2270	012774	104401				TYPOC			:: GO TYPE--OCTAL ASCII(ALL DIGITS)
2271	012776	104400	013457			TYPE	,SMNEW		:: PROMPT FOR NEW SMR
2272	013002	005046			19S:	CLR	-(SP)		:: CLEAR COUNTER
2273	013004	005046				CLR	-(SP)		:: THE NEW SMR
2274	013006	105777	166132		7S:	TSTB	2STKS		:: CHAR THERE?
2275	013012	100375				BPL	7S		:: IF NOT TRY AGAIN
2276									
2277	013014	117746	166126			MOVB	2STKB,-(SP)		:: PICK UP CHAR
2278	013020	042716	177600			BIC	#1C177,(SP)		:: MAKE IT 7-BIT ASCII
2279									
2280									
2281									
2282	013024	021627	000025		9S:	CMP	(SP),#25		:: IS IT A CONTROL-U?
2283	013030	001005				BNE	105		:: BRANCH IF NOT
2284	013032	104400	013434			TYPE	,SCNTLU		:: YES, ECHO CONTROL-U (?U)
2285	013036	062706	000006		20S:	ADD	#6,SP		:: IGNORE PREVIOUS INPUT
2286	013042	000757				BR	19S		:: LET'S TRY IT AGAIN
2287									
2288									
2289	013044	021627	000015		10S:	CMP	(SP),#15		:: IS IT A <CR>?
2290	013050	001022				BNE	165		:: BRANCH IF NO
2291	013052	005766	000004			TST	4(SP)		:: YES, IS IT THE FIRST CHAR?
2292	013056	001403				BEQ	115		:: BRANCH IF YES
2293	013060	016677	000002	166052		MOV	2(SP),2SMR		:: SAVE NEW SMR
2294	013066	062706	000006		11S:	ADD	#6,SP		:: CLEAR UP STACK
2295	013072	104400	001171		14S:	TYPE	,SCRLF		:: ECHO <CR> AND <LF>
2296	013076	123727	001135	000001		CHPB	\$INTAG,#1		:: RE-ENABLE TTY KBD INTERRUPTS?
2297	013104	001003				BNE	155		:: BRANCH IF NOT
2298	013106	012777	000100	166030		MOV	#100,2STKS		:: RE-ENABLE TTY KBD INTERRUPTS
2299	013114	000002			15S:	RTI			:: RETURN
2300	013116	004737	015046		16S:	JSR	PC,STPEC		:: ECHO CHAR
2301	013122	021627	000060			CMP	(SP),#60		:: CHAR < 0?

```

2302 013126 002420
2303 013130 021627 000067
2304 013134 003015
2305 013136 042726 000060
2306 013140 005766 000002
2307 013144 001403
2308 013150 006316
2309 013150 006316
2310 013154 006316
2311 013156 005266 000002
2312 013162 056616 177776
2313 013166 000707
2314 013170 104400 001170
2315 013174 000720

```

```

BLT 185
CMP (SP),#67
BGT 185
BIC #60,(SP)+
TST 2(SP)
BEQ 175
ASL (SP)
ASL (SP)
ASL (SP)
175: INC 2(SP)
BIS -2(SP),(SP)
BR 75
185: TYPE #SQUES
BR 205
.DSABL LSB

```

```

::BRANCH IF YES
::CHAR > 7?
::BRANCH IF YES
::STRIP-OFF ASCII
::IS THIS THE FIRST CHAR
::BRANCH IF YES
::NO, SHIFT PRESENT
::CHAR OVER TO MAKE
::ROOM FOR NEW ONE.
::KEEP COUNT OF CHAR
::SET IN NEW CHAR
::GET THE NEXT ONE
::TYPE ?<CR><LF>
::SIMULATE CONTROL-U

```

THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

#CALL:

```

# RDCHR
# RETURN HERE
#

```

```

::INPUT A SINGLE CHARACTER FROM THE TTY
::CHARACTER IS ON THE STACK
::WITH PARITY BIT STRIPPED OFF

```

```

2327 013176 011646
2328 013200 016666 000004 000002
2329 013206 105777 165732
2330 013212 100375
2331 013214 117766 165726 000004
2332 013222 042766 177600 000004
2333 013230 026627 000004 000023
2334 013236 001013
2335 013240 105777 165700
2336 013244 100375
2337 013246 117746 165674
2338 013250 042716 177600
2339 013256 022627 000021
2340 013260 001366
2341 013264 000750
2342 013266 026627 000004 000140
2343 013274 002407
2344 013276 026627 000004 000175
2345 013304 003003
2346 013306 042766 000040 000004
2347 013314 000002

```

```

SRDCHR: MOV (SP),-(SP)
MOV 4(SP),2(SP)
15: TSTB #2TKS
BPL 15
MOVB #2TKB,4(SP)
BIC #C<177>,4(SP)
CMP 4(SP),#23
BNE 35
25: TSTB #2TKS
BPL 25
MOVB #2TKB,-(SP)
BIC #C<177>,(SP)
CMP (SP)+,#21
BNE 25
BR 15
35: CMP 4(SP),#140
BLT 45
CMP 4(SP),#175
BGT 45
BIC #40,4(SP)
45: RTI

```

```

::PUSH DOWN THE PC
::SAVE THE PS
::WAIT FOR
::A CHARACTER
::READ THE TTY
::GET RID OF JUNK IF ANY
::IS IT A CONTROL-5?
::BRANCH IF NO
::WAIT FOR A CHARACTER
::LOOP UNTIL ITS THERE
::GET CHARACTER
::MAKE IT 7-BIT ASCII
::IS IT A CONTROL-0?
::IF NOT DISCARD IT
::YES, RESUME
::IS IT UPPER CASE?
::BRANCH IF YES
::IS IT A SPECIAL CHAR?
::BRANCH IF YES
::MAKE IT UPPER CASE
::GO BACK TO USER

```

THIS ROUTINE WILL INPUT A STRING FROM THE TTY

#CALL:

```

# RDLIN
# RETURN HERE
#

```

```

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

```

```

2355 013316 010346
2356 013320 012703 013424
2357 013324 022703 013434

```

```

SRDLIN: MOV R3,-(SP)
15: MOV #TTYIN,R3
25: CMP #TTYIN+8.,R3

```

```

::SAVE R3
::GET ADDRESS
::BUFFER FULL?

```

2358	013330	101405				BLOS	4S		::: BR IF YES
2359	013332	104406				ROCHR			::: GO READ ONE CHARACTER FROM THE TTY
2360	013334	112613				MOVB	(SP)+, (R3)		::: GET CHARACTER
2361	013336	122713	000177		10S:	CMPB	#177, (R3)		::: IS IT A RUBOUT
2362	013340	001003				BNE	3S		::: SKIP IF NOT
2363	013344	104400	001170		4S:	TYPE	SQUES		::: TYPE A '?'
2364	013350	000763				BR	1S		::: CLEAR THE BUFFER AND LOOP
2365	013352	111337	013422		3S:	MOVB	(R3), 9S		::: ECHO THE CHARACTER
2366	013356	104400	013422			TYPE	9S		
2367	013362	122723	000015			CMPB	#15, (R3)+		::: CHECK FOR RETURN
2368	013366	001356				BNE	2S		::: LOOP IF NOT RETURN
2369	013370	105063	177777			CLRB	-1(R3)		::: CLEAR RETURN (THE 15)
2370	013374	104400	001172			TYPE	SLF		::: TYPE A LINE FEED
2371	013400	012603				MOV	(SP)+, R3		::: RESTORE R3
2372	013402	011646				MOV	(SP) -, (SP)		::: ADJUST THE STACK AND PUT ADDRESS OF THE
2373	013404	016666	000004	000002		MOV	4(SP), 2(SP)		::: FIRST ASCII CHARACTER ON IT
2374	013412	012766	013424	000004		MOV	#STTYIN, 4(SP)		
2375	013420	000002				RTI			::: RETURN
2376	013422	000			9S:	.BYTE	0		::: STORAGE FOR ASCII CHAR. TO TYPE
2377	013423	000				.BYTE	0		::: TERMINATOR
2378	013424	000010			\$TTYIN:	.BLKB	8.		::: RESERVE 8 BYTES FOR TTY INPUT
2379	013434	052536	005015	000	\$CNTLU:	.ASCIZ	/U/<15><12>		::: CONTROL "U"
2380	013441	136	006507	000012	\$CNTLG:	.ASCIZ	/G/<15><12>		::: CONTROL "G"
2381	013446	005015	053523	020122	\$MSMR:	.ASCIZ	<15><12>/SMR = /		
2382	013454	020075	000						
2383	013457	040	047040	053505	\$MNEW:	.ASCIZ	/ NEW = /		
2384	013464	036440	000040						

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

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

013470 011646
013472 016666 000004 000002
013500 010046
013502 010146
013504 010246
013506 104407
013510 012600
013512 010037 013616
013516 005001
013520 005002
013524 112046
013528 001420
013532 122716 000060
013536 003026 000067
013540 122716
013544 002423
013548 006301
013552 006102
013556 006301
013560 006102
013564 006301
013568 006102
013572 042716 177770
013576 062601
013580 000730
013584 005726 000012
013588 010166 013626
013592 010237
013596 012602
013600 012601
013604 012600
013608 000002
013612 005726
013616 105010
013620 104400
013624 000000
013628 104400 001170
013632 000730
013636 000000

SRDOCT: MOV (SP), -(SP)
MOV 4(SP), 2(SP)
MOV R0, -(SP)
MOV R1, -(SP)
MOV R2, -(SP)
15: ROL IN
MOV (SP)+, R0
MOV R0, SS
CLR R1
CLR R2
25: MOVB (R0)+, -(SP)
BEQ 35
CMPB #'0', (SP)
BGT 45
CMPB #'7', (SP)
BLT 45
ASL R1
ROL R2
ASL R1
ROL R2
ASL R1
ROL R2
ASL R1
ROL R2
BIC #'C7', (SP)
ADD (SP)+, R1
BR 25
35: TST (SP)+
MOV R1, 12(SP)
MOV R2, SHIOCT
MOV (SP)+, R2
MOV (SP)+, R1
MOV (SP)+, R0
RTI
45: TST (SP)+
CLRB (R0)
TYPE
55: .WORD 0
TYPE 'QUES'
BR 15
SHIOCT: .WORD 0

PROVIDE SPACE FOR THE
INPUT NUMBER
PUSH R0 ON STACK
PUSH R1 ON STACK
PUSH R2 ON STACK
READ AN ASCII LINE
GET ADDRESS OF 1ST CHARACTER
AND SAVE IT
CLEAR DATA WORD
PICKUP THIS CHARACTER
IF ZERO GET OUT
MAKE SURE THIS CHARACTER
IS AN OCTAL DIGIT
#2
#4
#8
STRIP THE ASCII JUNK
ADD IN THIS DIGIT
LOOP
CLEAN TERMINATOR FROM STACK
SAVE THE RESULT
POP STACK INTO R2
POP STACK INTO R1
POP STACK INTO R0
RETURN
CLEAN PARTIAL FROM STACK
SET A TERMINATOR
TYPE UP THRU THE BAD CHAR.
"?" "CR" & "LF"
TRY AGAIN
HIGH ORDER BITS GO HERE

01338
01339
01340
01341
01342
01343
01344
01345
01346
01347
01348
01349
01350
01351
01352
01353
01354
01355
01356
01357
01358
01359
01360
01361
01362
01363
01364
01365
01366
01367
01368
01369
01370
01371
01372
01373
01374
01375
01376
01377
01378
01379
01380
01381
01382
01383
01384
01385
01386
01387
01388
01389
01390
01391

013630 105777 165304
013634 100003
013636 005737 010544
013642 001070

013644
013644 104405
013646 104405
013650 032777 040000 165262
013656 001062

013660 000416

013662 013746 000004
013666 012737 013706 000004
013674 005737 177060
013700 012637 000004
013704 000431
013706 022626
013710 012637 000004
013714 000417
013716
013716 032777 000400 165214
013724 001404
013726 127737 165206 001102
013734 001433
013736 105737 001103
013742 001412
013744 032777 001000 165166
013752 001404
013754 013737 001110 001106
013762 000420
013764 105037 001103
013770 105237 001102
013774 113737 001102 001200
014002 011637 001106
014006 011637 001110
014012 005037 001166
014016 112737 000001 001115
014024 013777 001102 165110
014032 013716 001106
014036 000002

```
MSCOPE: TSTB 2SMR ;TEST BIT 7
          BPL 5SCOPE ;NOT SET
          TST LOOP ;TEST LOOP
          BNE SOVER ;SET LOOP ON TEST
.SBTTL SCOPE HANDLER ROUTINE

;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER(STSTN) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SM14=1 LOOP ON TEST
;SM09=1 LOOP ON ERROR
;SM08=1 LOOP ON TEST IN SMR<7:0>
;CALL
;* SCOPE ;;SCOPE=IOT

SSCOPE: CKSMR ;;TEST FOR CHANGE IN SOFT-SMR
         CKSMR
15: BIT 8BIT14,2SMR ;;LOOP ON PRESENT TEST?
     BNE SOVER ;;YES IF SM14=1
;*****START OF CODE FOR THE XOR TESTER*****
SXTSTR: BR 6S
         MOV 2ERRVEC, -(SP) ;;IF RUNNING ON THE "XOR" TESTER CHANGE
         MOV 5S, 2ERRVEC ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
         TST 28177060 ;;SAVE THE CONTENTS OF THE ERROR VECTOR
         MOV (SP)+, 2ERRVEC ;;SET FOR TIMEOUT
         BR 5SVLAD ;;TIME OUT ON XOR?
5S: CMP (SP)+, (SP)+ ;;RESTORE THE ERROR VECTOR
     MOV (SP)+, 2ERRVEC ;;GO TO THE NEXT TEST
     BR 7S ;;CLEAR THE STACK AFTER A TIME OUT
7S: MOV (SP)+, 2ERRVEC ;;RESTORE THE ERROR VECTOR
     BR 6S ;;LOOP ON THE PRESENT TEST
6S: ;*****END OF CODE FOR THE XOR TESTER*****
     BIT 8BIT08,2SMR ;;LOOP ON SPEC. TEST?
     BEQ 2S ;;BR IF NO
     CMPB 2SMR, 3TSTN ;;ON THE RIGHT TEST? SMR<7:0>
     BEQ SOVER ;;BR IF YES
2S: TSTB SERFLG ;;HAS AN ERROR OCCURRED?
     BEQ 5SVLAD ;;BR IF NO
     BIT 8BIT09,2SMR ;;LOOP ON ERROR?
     BEQ 4S ;;BR IF NO
7S: MOV 5LPERR, 5LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
     BR SOVER
4S: CLRB SERFLG ;;ZERO THE ERROR FLAG
5SVLAD: INCB STSTN ;;COUNT TEST NUMBERS
         MOVB STSTN, STSTN ;;SET TEST NUMBER IN APT MAILBOX
         MOV (SP), 5LPADR ;;SAVE SCOPE LOOP ADDRESS
         MOV (SP), 5LPERR ;;SAVE ERROR LOOP ADDRESS
         CLR 5ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
         MOVB 81, 5SERMAX ;;ONLY ALLOW ONE(I) ERROR ON NEXT TEST
SOVER: MOV STSTN, 2DISPLAY ;;DISPLAY TEST NUMBER
        MOV 5LPADR, (SP) ;;FUDGE RETURN ADDRESS
        RTI ;;FIXES PS
```

.SBTTL APT COMMUNICATIONS ROUTINE

014040
014046
014054
014056
014064
014064
014066
014070
014074
014076
014104
014106
014114
014116
014122
014130
014134
014136
014142
014144
014146
014152
014154
014160
014166
014170
014176
014204
014210
014214
014216
014216
014222
014224
014230
014232
014236
014240
014246
014254
014260
014264
014270
014274
014276
014300
014302
014303
014304
014306
000200
000001
000100

112737
112737
000403
112737
010046
010146
105737
001450
122737
001031
132737
001425
017600
062766
005737
001375
010037
105720
001376
163700
006200
010037
012737
000413
017637
062766
177776
004737
000000
105737
001416
005737
001413
005737
001375
017637
062766
005237
105037
105037
105037
012601
012600
000207
000
000
000
014306
000200
000001
000100

SATY1: MOV 81,SFFLG ;:TO REPORT FATAL ERROR
SATY3: MOV 81,SFLG ;:TO TYPE A MESSAGE
BR SATYC
SATY4: MOV 81,SFFLG ;:TO ONLY REPORT FATAL ERROR
SATYC: MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
TSTB SFLG ;:SHOULD TYPE A MESSAGE?
BEQ 55 ;:IF NOT: BR
CMPB 8APTENV,SENV ;:OPERATING UNDER APT?
BNE 35 ;:IF NOT: BR
BITB 8APTPOOL,SENVH ;:SHOULD SPOOL MESSAGES?
BEQ 35 ;:IF NOT: BR
MOV 24(SP),R0 ;:GET MESSAGE ADDR.
ADD 82,4(SP) ;:BUMP RETURN ADDR.
15: TST SMSGTYPE ;:SEE IF DONE W/ LAST XMISSION?
BNE 15 ;:IF NOT: WAIT
MOV R0,SMSGAD ;:PUT ADDR IN MAILBOX
25: TSTB (R0)+ ;:FIND END OF MESSAGE
BNE 25
SUB SMSGAD,R0 ;:SUB START OF MESSAGE
ASR R0 ;:GET MESSAGE LNTH IN WORDS
MOV R0,SMSGLT ;:PUT LENGTH IN MAILBOX
MOV 84,SMSGTYPE ;:TELL APT TO TAKE MSG.
BR 55
35: MOV 24(SP),45 ;:PUT MSG ADDR IN JSR LINKAGE
ADD 82,4(SP) ;:BUMP RETURN ADDRESS
MOV 177776,-(SP) ;:PUSH 177776 ON STACK
JSR PC,STYPE ;:CALL TYPE MACRO
45: .WORD 0
55:
105: TSTB SFFLG ;:SHOULD REPORT FATAL ERROR?
BEQ 125 ;:IF NOT: BR
TST SENV ;:RUNNING UNDER APT?
BEQ 125 ;:IF NOT: BR
115: TST SMSGTYPE ;:FINISHED LAST MESSAGE?
BNE 115 ;:IF NOT: WAIT
MOV 24(SP),SFATAL ;:GET ERROR #
ADD 82,4(SP) ;:BUMP RETURN ADDR.
INC SMSGTYPE ;:TELL APT TO TAKE ERROR
125: CLRB SFFLG ;:CLEAR FATAL FLAG
CLRB SFLG ;:CLEAR LOG FLAG
CLRB SMLG ;:CLEAR MESSAGE FLAG
MOV (SP)+,R1 ;:POP STACK INTO R1
MOV (SP)+,R0 ;:POP STACK INTO R0
RTS PC ;:RETURN
SMLG: .BYTE 0 ;:MESSG. FLAG
SFLG: .BYTE 0 ;:LOG FLAG
SFFLG: .BYTE 0 ;:FATAL FLAG
EVEN
APTSIZE=200
APTENV=001
APTPOOL=100

H05

MAINDEC-11-DZVTD-B MACY11 27(732) 25-SEP-76 10:10 PAGE 60
DZVTDB.SAC APT COMMUNICATIONS ROUTINE

2548

000040

APTCSUP=040

.SBTTL ERROR HANDLER ROUTINE

```

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

```

25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
2600

```

SERROR:
014306      104405      CKSMR      ;;TEST FOR CHANGE IN SOFT-SWR
014306      113737      001102    001314    MOV      STSTNM,TSTNUM
014310      105237      001103    7S:      INCB     SERFLG      ;;SET THE ERROR FLAG
014316      001775      BEQ      7S      ;;DON'T LET THE FLAG GO TO ZERO
014322      013777      001102    164610    MOV      STSTNM,DISPDISP      ;;DISPLAY TEST NUMBER AND ERROR FLAG
014324      005237      001112    INC      SERRTL      ;;INC THE ERROR COUNT
014332      011637      001116    MOV      (SP),SERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
014336      162737      000002    001116    SUB      #2,SERRPC
014342      117737      164542    001114    MOV      2SERRPC,SITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
014350      032777      020000    164554    BIT      #BIT13,SWR      ;;SKIP TYPEOUT IF SET
014356      001004      BNE      20S      ;;SKIP TYPEOUTS
014364      004737      014500    JSR      PC,SERRTYP      ;;GO TO USER ERROR ROUTINE
014372      104400      001171    TYPE     ,SCLF
014376      :22737      000001    001214    20S:    CNPB     #APTENV,SENV      ;;RUNNING IN APT MODE
014376      001007      BNE      2S      ;;NO SKIP APT ERROR REPORT
014404      113737      001114    014420    MOV      SITEMB,21S      ;;SET ITEM NUMBER AS ERROR NUMBER
014414      004737      014056    JSR      PC,SATY4      ;;REPORT FATAL ERROR TO APT
014420      000      21S:    .BYTE   0
014421      000      .BYTE   0
014422      000777      BR      22S      ;;APT ERROR LOOP
014424      005777      164510    2S:    TST     2SWR      ;;HALT ON ERROR
014430      100002      BPL     3S      ;;SKIP IF CONTINUE
014432      000000      HALT    ;;HALT ON ERROR!
014434      104405      CKSMR   ;;TEST FOR CHANGE IN SOFT-SWR
014436      032777      001000    164474    3S:    BIT     #BIT09,2SWR      ;;LOOP ON ERROR SWITCH SET?
014444      001402      BEQ     4S      ;;BR IF NO
014446      013716      001110    MOV     $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
014452      005737      001166    4S:    TST     $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
014456      001402      BEQ     5S      ;;BR IF NONE
014460      013716      001166    MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
014464      022737      006776    000042    5S:    CMP     #SENDAD,2#42    ;;ACT-11 AUTO-ACCEPT?
014464      001001      BNE     6S      ;;BRANCH IF NO
014472      000000      HALT    ;;YES
014476      000002      6S:    RTI      ;;RETURN

```

```

2601
2602
2603
2604
2605
2606
2607
2608
2609
2610 014500
2611 014500 104400 001171
2612 014504 010046
2613 014506 005000
2614 014510 153700 001114
2615 014514 001004
2616 014516 013746 001116
2617
2618 014522 104401
2619 014524 000426
2620 014526 005300
2621 014530 006300
2622 014532 006300
2623 014534 006300
2624 014536 062700 001256
2625 014542 012037 014552
2626 014546 001404
2627 014550 104400
2628 014552 000000
2629 014554 104400 001171
2630 014560 012037 014570
2631 014564 001404
2632 014566 104400
2633 014570 000000
2634 014572 104400 001171
2635 014576 011000
2636 014600 001004
2637 014602 012600
2638 014604 104400 001171
2639 014610 000207
2640 014612
2641 014612 013046
2642 014614 104401
2643 014616 005710
2644 014620 001770
2645 014622 104400 014630
2646 014626 000771
2647 014630 020040 000
2648 014634

```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

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

```

SERRTYP:

```

      TYPE      SCRLF      : "CARRIAGE RETURN" & "LINE FEED"
      MOV      RD,-(SP)    : SAVE RD
      CLR      RD          : PICKUP THE ITEM INDEX
      BISB     @#SITEMB,RD
      BNE     IS          : IF ITEM NUMBER IS ZERO, JUST
                          : TYPE THE PC OF THE ERROR
      MOV      SERRPC,-(SP) : SAVE SERRPC FOR TYPEOUT
                          : ERROR ADDRESS
                          : GO TYPE--OCTAL ASCII(ALL DIGITS)
                          : GET OUT
      TYPCC   6S          : ADJUST THE INDEX SO THAT IT WILL
                          : WORK FOR THE ERROR TABLE
1S:   BR      6S
      DEC     RD
      ASL    RD
      ASL    RD
      ASL    RD
      ADD    #SERRTB,RD   : FORM TABLE POINTER
      MOV    (RD)+,2S    : PICKUP "ERROR MESSAGE" POINTER
      BEQ   3S          : SKIP TYPEOUT IF NO POINTER
      TYPE  "ERROR MESSAGE"
                          : "ERROR MESSAGE" POINTER GOES HERE
2S:   .WORD  0          : "CARRIAGE RETURN" & "LINE FEED"
      TYPE  SCRLF      : PICKUP "DATA HEADER" POINTER
3S:   MOV    (RD)+,4S   : SKIP TYPEOUT IF 0
      BEQ   5S          : TYPE THE "DATA HEADER"
      TYPE  "DATA HEADER"
                          : "DATA HEADER" POINTER GOES HERE
4S:   .WORD  0          : "CARRIAGE RETURN" & "LINE FEED"
      TYPE  SCRLF      : PICKUP "DATA TABLE" POINTER
5S:   MOV    (RD),RD    : GO TYPE THE DATA
      BNE   7S          : RESTORE RD
6S:   MOV    (SP)+,RD   : "CARRIAGE RETURN" & "LINE FEED"
      TYPE  SCRLF      : RETURN
      RTS   PC
7S:   MOV    @ (RD)+,-(SP) : SAVE @ (RD)+ FOR TYPEOUT
      TYPCC 6S          : GO TYPE--OCTAL ASCII(ALL DIGITS)
      TST   (RD)        : IS THERE ANOTHER NUMBER?
      BEQ   8S          : BR IF NO
      TYPE  " "         : TYPE TWO(2) SPACES
      BR   7S          : LOOP
8S:   .ASCIZ / /
      .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: SNULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
:NOTE2: SFILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
:NOTE3: SFILLC 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
*

2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704

014634 105737 001157
014640 100002
014642 000000
014644 000430
014646 010046
014650 017600 000002
014654 122737 000001 001214
014662 001011
014664 132737 000100 001215
014672 001405
014674 010037 014704
014700 004737 014046
014704 000000
014706 132737 000040 001215
014714 001003
014716 112046
014720 001005
014722 005725
014724 012600
014726 062716 000002
014732 010002
014734 122716 000011
014740 001430
014742 122716 000200
014746 001006
014750 005726
014752 104400
014754 001171
014756 105037 015112
014762 000755
014764 004737 015046
014770 123726 001156
014774 001350
014776 013746 001154
015002 105366 000001
015006 002770
015010 004737 015046

STYPE: TSTB STPFLG
BPL 1S
HALT
BR 3S
1S: MOV RD, -(SP)
MOV 32(SP), RD
CMPB #APTENV, SENV
BNE 62S
BITB #APTPOOL, SENVM
BEQ 62S
MOV RD, 61S
JSR PC, SATY3
61S: .WORD 0
62S: BITB #APTCSUP, SENVM
BNE 60S
2S: MOVB (RD)+, -(SP)
BNE 4S
TST (SP)+
60S: MOV (SP)+, RD
3S: ADD #2, (SP)
RTI
4S: CMPB #HT, (SP)
BEQ 8S
CMPB #CRLF, (SP)
BNE 5S
TST (SP)+
TYPE
SCRLF
CLRB SCHARCNT
BR 2S
5S: JSR PC, STYPEC
6S: CMPB SFILLC, (SP)+
BNE 2S
MOV SNULL, -(SP)
7S: DECB 1(SP)
BLT 6S
JSR PC, STYPEC

;; IS THERE A TERMINAL?
;; BR IF YES
;; HALT HERE IF NO TERMINAL
;; LEAVE
;; SAVE RD
;; GET ADDRESS OF ASCIZ STRING
;; RUNNING IN APT MODE
;; NO, GO CHECK FOR APT CONSOLE
;; SPOOL MESSAGE TO APT
;; NO, GO CHECK FOR CONSOLE
;; SETUP MESSAGE ADDRESS FOR APT
;; SPOOL MESSAGE TO APT
;; MESSAGE ADDRESS
;; APT CONSOLE SUPPRESSED
;; YES, SKIP TYPE OUT
;; PUSH CHARACTER TO BE TYPED ONTO STACK
;; BR IF IT ISN'T THE TERMINATOR
;; IF TERMINATOR POP IT OFF THE STACK
;; RESTORE RD
;; ADJUST RETURN PC
;; RETURN
;; BRANCH IF <HT>
;; BRANCH IF NOT <CRLF>
;; POP <CR><LF> EQUIV
;; TYPE A CR AND LF
;; CLEAR CHARACTER COUNT
;; GET NEXT CHARACTER
;; GO TYPE THIS CHARACTER
;; IS IT TIME FOR FILLER CHARS.?
;; IF NO GO GET NEXT CHAR.
;; GET # OF FILLER CHARS. NEEDED
;; AND THE NULL CHAR.
;; DOES A NULL NEED TO BE TYPED?
;; BR IF NO--GO POP THE NULL OFF OF STACK
;; GO TYPE A NULL

```

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


2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784

.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.
#STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
#CALL:
#       MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
#       TYPOS    ;;CALL FOR TYPEOUT
#       .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
#       .BYTE   M              ;;M=1 OR 0
#                               ;;1=TYPE LEADING ZEROS
#                               ;;0=SUPPRESS LEADING ZEROS
#STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
#STYPOS OR STYPOC
#CALL:
#       MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
#       TYPON    ;;CALL FOR TYPEOUT
#STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
#CALL:
#       MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
#       TYPOC    ;;CALL FOR TYPEOUT
2755 015116 017646 000000 015341 STYPOS: MOV      2(SP),-(SP)      ;; PICKUP THE MODE
2756 015122 116637 000001 015341 MOVVB   1(SP),SOFILL    ;; LOAD ZERO FILL SWITCH
2757 015130 112637 015343 MOVVB   (SP)+,SOMODE+1  ;; NUMBER OF DIGITS TO TYPE
2758 015134 062716 000002 ADD      #2,(SP)      ;; ADJUST RETURN ADDRESS
2759 015140 000406 BR      STYPON
2760 015142 112737 000001 015341 STYPOC: MOVVB  #1,SOFILL    ;; SET THE ZERO FILL SWITCH
2761 015150 112737 000006 015343 MOVVB   #6,SOMODE+1  ;; SET FOR SIX(6) DIGITS
2762 015156 112737 000005 015340 STYPON: MOVVB  #5,SOCNT  ;; SET THE ITERATION COUNT
2763 015164 010346 MOV      R3,-(SP)    ;; SAVE R3
2764 015166 010446 MOV      R4,-(SP)    ;; SAVE R4
2765 015170 010546 MOV      R5,-(SP)    ;; SAVE R5
2766 015172 113704 015343 MOVVB   SOMODE+1,R4  ;; GET THE NUMBER OF DIGITS TO TYPE
2767 015176 005404 NEG      R4
2768 015200 062704 000006 ADD      #6,R4      ;; SUBTRACT IT FOR MAX. ALLOWED
2769 015204 110437 015342 MOVVB   R4,SOMODE    ;; SAVE IT FOR USE
2770 015210 113704 015341 MOVVB   SOFILL,R4    ;; GET THE ZERO FILL SWITCH
2771 015214 016605 000012 MOV      12(SP),R5   ;; PICKUP THE INPUT NUMBER
2772 015220 005003 CLR      R3          ;; CLEAR THE OUTPUT WORD
2773 015222 006105 15:  ROL     R5          ;; ROTATE MSB INTO "C"
2774 015224 000404 BR      35
2775 015226 006105 25:  ROL     R5          ;; GO DO MSB
2776 015230 006105 ROL     R5          ;; FORM THIS DIGIT
2777 015232 006105 ROL     R5
2778 015234 010503 MOV      R5,R3
2779 015236 006103 35:  ROL     R3          ;; GET LSB OF THIS DIGIT
2780 015240 105337 015342 DECB   SOMODE        ;; TYPE THIS DIGIT?
2781 015244 100016 BPL     75           ;; BR IF NO
2782 015246 042703 177770 BIC     #177770,R3  ;; GET RID OF JUNK
2783 015252 001002 BNE     45           ;; TEST FOR 0
2784 015254 005704 TST     R4          ;; SUPPRESS THIS 0?

```

2785 015256 001403
2786 015260 005204
2787 015262 052703 000060
2788 015266 052703 000040
2789 015272 110337 015336
2790 015276 104400 015336
2791 015302 105337 015340
2792 015306 003347
2793 015310 002402
2794 015312 005204
2795 015314 000744
2796 015316 012605
2797 015320 012604
2798 015322 012603
2799 015324 016666 000002 000004
2800 015332 012616
2801 015334 000002
2802 015336 000
2803 015337 000
2804 015340 000
2805 015341 000
2806 015342 000000

4S: BEQ 5S
INC R4
BIS #'0,R3
5S: BIS #' ,R3
MOV R3,6S
TYPE 8S
7S: DECB \$OCNT
BGT 2S
BLT 6S
INC R4
BR 2S
6S: MOV (SP)+,R5
MOV (SP)+,R4
MOV (SP)+,R3
MOV 2(SP),4(SP)
MOV (SP)+,(SP)
RTI
8S: .BYTE 0
.BYTE 0
\$OCNT: .BYTE 0
\$OFILL: .BYTE 0
\$OMODE: .WORD 0
.SBTTL TRAP DECODER

::BR IF YES
::DON'T SUPPRESS ANYMORE 0'S
::MAKE THIS DIGIT ASCII
::MAKE ASCII IF NOT ALREADY
::SAVE FOR TYPING
::GO TYPE THIS DIGIT
::COUNT BY 1
::BR IF MORE TO DO
::BR IF DONE
::INSURE LAST DIGIT ISN'T A BLANK
::GO DO THE LAST DIGIT
::RESTORE R5
::RESTORE R4
::RESTORE R3
::SET THE STACK FOR RETURNING
::RETURN
::STORAGE FOR ASCII DIGIT
::TERMINATOR FOR TYPE ROUTINE
::OCTAL DIGIT COUNTER
::ZERO FILL SWITCH
::NUMBER OF DIGITS TO TYPE

2808
2809
2810
2811
2812
2813
2814
2815 015344 010046
2816 015346 016600 000002
2817 015352 005740
2818 015354 111000
2819 015356 006300
2820 015360 016000 015366
2821 015364 000200

::*****
::*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.

\$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 \$TRPAD(R0),R0 ;INDEX TO TABLE
RTS R0 ;GO TO ROUTINE

.SBTTL TRAP TABLE

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

2822
2823
2824
2825
2826
2827
2828
2829
2830 015366
2831 015366 014634
2832 015370 015142
2833 015372 015116
2834 015374 015156
2835
2836 015376 012764
2837
2838 015400 012714
2839 015402 013176
2840 015404 013316

ROUTINE

\$TRPAD: \$TYPE ;:CALL=TYPE TRAP+0(104400) TTY TYPEOUT ROUTINE
\$TYPOC ;:CALL=TYPOC TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS ;:CALL=TYPOS TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON ;:CALL=TYPON TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
\$GTSWR ;:CALL=GTSWR TRAP+4(104404) GET SOFT-SWR SETTING
\$CKSWR ;:CALL=CKSWR TRAP+5(104405) TEST FOR CHANGE IN SOFT-SWR
\$RDCHR ;:CALL=RDCHR TRAP+6(104406) TTY TYPEIN CHARACTER ROUTINE
\$RDLIN ;:CALL=RDLIN TRAP+7(104407) TTY TYPEIN STRING ROUTINE

```

2841 015406 013470
2842
2843
2844
2845
2846 015410 012737 015554 000024
2847 015416 012737 000340 000026
2848 015424 010046
2849 015426 010146
2850 015430 010246
2851 015432 010346
2852 015434 010446
2853 015436 010546
2854 015440 017746 163474
2855 015444 010637 015560
2856 015450 012737 015462 000024
2857 015456 000000
2858 015460 000776
2859
2860
2861
2862 015462 012737 015554 000024
2863 015470 013706 015560
2864 015474 005037 015560
2865 015500 005237 015560
2866 015504 001375
2867 015506 012677 163426
2868 015512 012605
2869 015514 012604
2870 015516 012603
2871 015520 012602
2872 015522 012601
2873 015524 012600
2874 015526 012737 015410 000024
2875 015534 012737 000340 000026
2876 015542 104400
2877 015544 011202
2878 015546 012716
2879 015550 001402
2880 015552 000002
2881 015554 000000
2882 015556 000776
2883 015560 000000
2884 015562 000000
2885 000001

```

```

SRDOCT ;;CALL=RD OCT TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
.SBRTL POWER DOWN AND UP ROUTINES

```

```

*****
:POWER DOWN ROUTINE

```

```

$PWRDN: MOV $SILLUP,2#PWRVEC ;;SET FOR FAST UP
MOV $340,2#PWRVEC+2 ;;PRIO:7
MOV RO,-(SP) PUSH RO ON STACK
MOV R1,-(SP) PUSH R1 ON STACK
MOV R2,-(SP) PUSH R2 ON STACK
MOV R3,-(SP) PUSH R3 ON STACK
MOV R4,-(SP) PUSH R4 ON STACK
MOV R5,-(SP) PUSH R5 ON STACK
MOV $SMR,-(SP) PUSH $SMR ON STACK
MOV SP,$SAVR6 SAVE SP
MOV $PWRUP,2#PWRVEC ;;SET UP VECTOR
HALT
BR -2 ;;HANG UP

```

```

*****
:POWER UP ROUTINE

```

```

$PWRUP: MOV $SILLUP,2#PWRVEC ;;SET FOR FAST DOWN
MOV $SAVR6,SP GET SP
CLR $SAVR6 WAIT LOOP FOR THE TTY
IS: INC $SAVR6 WAIT FOR THE INC
BNE IS OF WORD
MOV (SP)+,$SMR POP STACK INTO $SMR
MOV (SP)+,R5 POP STACK INTO R5
MOV (SP)+,R4 POP STACK INTO R4
MOV (SP)+,R3 POP STACK INTO R3
MOV (SP)+,R2 POP STACK INTO R2
MOV (SP)+,R1 POP STACK INTO R1
MOV (SP)+,R0 POP STACK INTO R0
MOV $PWRDN,2#PWRVEC ;;SET UP THE POWER DOWN VECTOR
MOV $340,2#PWRVEC+2 ;;PRIO:7
TYPE PWRMSG ;;REPORT THE POWER FAILURE
MOV (PC)+,(SP) PWRMSG ;;POWER FAIL MESSAGE POINTER
$PWRAD: .WORD RBEGIN ;;RESTART AT RBEGIN
SILLUP: RTI ;;RESTART ADDRESS
BR -2
$SAVR6: 0
BUFFER: 0
.END

```

```

;; THE POWER UP SEQUENCE WAS STARTED
;; BEFORE THE POWER DOWN WAS COMPLETE
;; PUT THE SP HERE

```


E06

DH1	011714	717	2140#													
DH4	011745	711	2145#													
DISPLA	001142	625#	812#	820*	2489*	2568*										
DISPRE	000174	556#	820													
DSPCH	010576	1984	1986	1987	1988	1990#										
DSMR =	177570	443#	624	811												
DT1	012670	718	2241#													
DT4	012700	712	2243#													
ENTVEC=	000030	532#	79#	797*												
EM1	011516	2117#														
EM2	011563	716	2124#													
EM3	011655	722	2134#													
EM4	011612	710	2128#													
ERRVEC=	000004	525#	778#	779*	809	810*	821*	2463	2464*	2466*	2469*					
ESC =	000033	735#	871	889	923	953	971	1002	1031	1042	1082	1093	1133	1151		
		1179	1191	1226	1238	1275	1293	1322	1340	1375	1393	1425	1459	1488		
		1522	1553	1559	1614	1739	1779	1807								
FINDOT	010562	1942	1949	1966#	1970											
FINDTA	010572	1941#	1948#	1968#												
FIRST	001306	755#	828#	844*	845	847	851									
FNDR	010554	1964#	1977	1979	1985											
GETCHR	011110	1969	2050#													
GIN	001420	782	784#													
GINA	001426	780	785#													
GNS =	##### U	555	2831	2832	2833	2834	2836	2838	2839	2840	2841					
GROAL	012232	1130	2180#													
GTSMR =	104404	2836#														
HGORAL	012420	1273	2203#													
HT =	000011	435#	2688	2729												
IOTVEC=	000020	530#	794#	795*	830*											
LAST	001310	757#	848#	1649	1651											
LDCO =	000103	745#	1337	1362												
LDC1 =	000113	746#	1390	1414												
LDED =	000101	739#	873	877	955	959	1033	1044	1049	1084	1095	1099	1135	1181		
		1193	1228	1240	1277	1324	1377	1427	1490	1555	1582	1741	1781	1809		
LDE1 =	000111	740#	875	879	957	961	1035	1046	1086	1097	1137	1183	1230	1279		
		1326	1379	1429	1492	1557	1584	1812	1086	1097	1137	1183	1230	1279		
LDG0 =	000102	742#	1057	1143	1201	1285	1332	1435	1449							
LDG1 =	000112	743#	1107	1148	1248	1290	1385	1498	1512							
LF =	000012	436#	2723	2729												
LHVD =	000104	748#	882	914	1587											
LHV1 =	000114	749#	964	994	1604											
LNO =	000100	737#	881	891	892	893	894	899	899	900	901	901	901			
		902	903	904	905	906	907	908	909	910	911	912	913	914	915	916
		932	933	934	935	936	937	938	939	940	941	942	943	944	945	946
		976	977	978	979	980	981	982	983	984	985	986	987	988	989	990
		993	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017
		1016	1017	1018	1019	1051	1052	1053	1054	1055	1056	1057	1058	1059	1060	1061
		1158	1159	1160	1161	1162	1163	1164	1165	1166	1167	1168	1169	1170	1171	1172
		1242	1247	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260
		1306	1307	1308	1309	1310	1311	1312	1313	1314	1315	1316	1317	1318	1319	1320
		1350	1351	1352	1353	1354	1355	1356	1357	1358	1359	1360	1361	1362	1363	1364
		1400	1401	1402	1403	1404	1405	1406	1407	1408	1409	1410	1411	1412	1413	1414
		1463	1464	1465	1466	1467	1468	1469	1470	1471	1472	1473	1474	1475	1476	1477
		1476	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535	1536	1537
		1536	1537	1538	1539	1586	1603	1616	1617	1618	1619	1620	1621	1622		

	1623	1624	1625	1626	1627	1628	1629	1630	1631	1745	1750	1784	1789
LOOP = 010544	181	1822											
LSC = 000110	731	1940	1947	1959	2440								
MAXMOZ = 000354	729	1052	1102	1196	1243	1746	1785	1818					
MAXM2C = 000400	730	687	920	1149	1291	1386	1436	1499	1762				
MAXVRT = 001000	731		996	1060	1110	1204	1251	1440	1503	1834			
NO0 = 011277	1941	2092											
NO1 = 011362	1948	1968	2101										
NO2 = 011446	1965	2110											
NSCOPE = 013630	830	2438											
NT08 = 011166	2042	2071											
NOEXIT = 010550	1868	1893	1924	1930	1952	1961							
PC = 000007	456	884	908	909	917	942	943	966	990	991	997	1021	1022
	1038	1054	1059	1067	1070	1089	1104	1109	1117	1120	1140	1145	1170
	1171	1186	1198	1203	1211	1214	1233	1245	1250	1258	1261	1282	1287
	1312	1313	1329	1334	1359	1364	1365	1382	1387	1412	1416	1417	1432
	1437	1453	1478	1479	1495	1500	1516	1541	1542	1562	1573	1589	1595
	1606	1612	1633	1635	1659	1662	1669	1674	1748	1753	1787	1792	1820
	1825	1861	1956	1969	2022	2035	2044	2057	2064	2300	2523	2540	2575
	2581	2639	2678	2697	2704	2711	2725	2727	2878				
PIR0 = 177772	442												
PIR0VE = 000240	536												
PR0 = 000000	459												
PR1 = 000040	460												
PR2 = 000100	461												
PR3 = 000140	462												
PR4 = 000200	463												
PR5 = 000240	464												
PR6 = 000300	465												
PR7 = 000340	466												
PS = 177776	439	440											
PSN = 177776	440												
PRM5C = 011202	2081	2877											
PRVVC = 000024	531	800	801	2846	2847	2856	2862	2874	2875				
RBEGIN = 001402	2350	781	2879										
ROCHR = 104406	2359	2839											
ROLIN = 104407	2404	2840											
ROOCT = 104410	843	2841											
RESVEC = 000010	526												
RSTR = 002016	853	1654											
RSTRTA = 002010	840	851	1675										
RO = 000000	447	853	854	855	856	857	858	859	860	870	871	872	873
	874	875	876	877	878	879	880	881	882	885	889	890	891
	892	893	894	895	896	897	898	899	900	901	902	903	904
	905	906	907	912	913	914	918	923	924	925	926	927	928
	929	930	931	932	933	934	935	936	937	938	939	940	941
	952	953	954	955	956	957	958	959	960	961	962	963	964
	967	971	972	973	974	975	976	977	978	979	980	981	982
	983	984	985	986	987	988	989	992	993	994	998	1002	1003
	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013	1014	1015	1016
	1017	1018	1019	1020	1030	1031	1032	1033	1034	1035	1036	1037	1041
	1042	1043	1044	1045	1046	1047	1049	1050	1051	1052	1055	1056	1057
	1061	1066	1081	1082	1083	1084	1085	1086	1087	1088	1092	1093	1094
	1095	1096	1097	1098	1099	1100	1101	1102	1105	1106	1107	1111	1116
	1132	1133	1134	1135	1136	1137	1138	1139	1151	1152	1153	1154	1155

		1156#	1157#	1158#	1159#	1160#	1161#	1162#	1163#	1164#	1165#	1166#	1167#	1168#
		1169#	1178#	1179#	1180#	1181#	1182#	1183#	1184#	1185#	1190#	1191#	1192#	1193#
		1194#	1195#	1196#	1199#	1200#	1201#	1205#	1210#	1225#	1226#	1227#	1228#	1229#
		1230#	1231#	1232#	1237#	1238#	1239#	1240#	1241#	1242#	1243#	1246#	1247#	1248#
		1252#	1257#	1274#	1275#	1276#	1277#	1278#	1279#	1280#	1281#	1293#	1294#	1295#
		1296#	1297#	1298#	1299#	1300#	1301#	1302#	1303#	1304#	1305#	1306#	1307#	1308#
		1309#	1310#	1311#	1321#	1322#	1323#	1324#	1325#	1326#	1327#	1328#	1340#	1341#
		1342#	1343#	1344#	1345#	1346#	1347#	1348#	1349#	1350#	1351#	1352#	1353#	1354#
		1355#	1356#	1357#	1358#	1374#	1375#	1376#	1377#	1378#	1379#	1380#	1381#	1393#
		1394#	1395#	1396#	1397#	1398#	1399#	1400#	1401#	1402#	1403#	1404#	1405#	1406#
		1407#	1408#	1409#	1410#	1411#	1424#	1425#	1426#	1427#	1428#	1429#	1430#	1431#
		1441#	1443#	1444#	1458#	1459#	1460#	1461#	1462#	1463#	1464#	1465#	1466#	1467#
		1468#	1469#	1470#	1471#	1472#	1473#	1474#	1475#	1476#	1477#	1487#	1488#	1489#
		1490#	1491#	1492#	1493#	1494#	1504#	1506#	1507#	1521#	1522#	1523#	1524#	1525#
		1526#	1527#	1528#	1529#	1530#	1531#	1532#	1533#	1534#	1535#	1536#	1537#	1538#
		1539#	1540#	1552#	1553#	1554#	1555#	1556#	1557#	1558#	1559#	1560#	1561#	1563#
		1564#	1565#	1566#	1567#	1569#	1572#	1579#	1580#	1581#	1582#	1583#	1584#	1585#
		1586#	1587#	1590#	1594#	1600#	1601#	1602#	1603#	1604#	1607#	1613#	1614#	1615#
		1616#	1617#	1618#	1619#	1620#	1621#	1622#	1623#	1624#	1625#	1626#	1627#	1628#
		1629#	1630#	1631#	1632#	1666#	1669#	1738#	1739#	1740#	1741#	1742#	1743#	1745#
		1746#	1749#	1750#	1751#	1754#	1766#	1776#	1778#	1779#	1780#	1781#	1782#	1783#
		1784#	1785#	1788#	1789#	1790#	1793#	1794#	1796#	1806#	1807#	1808#	1809#	1810#
		1811#	1812#	1815#	1816#	1817#	1818#	1821#	1822#	1823#	1826#	1836#	1864#	1869#
		1878#	1882#	1886#	1955#	1974#	1975#	1976#	1978#	1980#	1981#	1982#	1983#	1984#
		1986#	1987#	1988#	2055#	2071#	2073#	2401#	2405#	2406#	2409#	2429#	2432#	2500#
		2508#	2512#	2513#	2515#	2516#	2517#	2539#	2611#	2612#	2613#	2620#	2621#	2622#
		2623#	2624#	2625#	2630#	2635#	2637#	2641#	2643#	2671#	2672#	2677#	2682#	2685#
		2815#	2816#	2817#	2818#	2819#	2820#	2821#	2848#	2873#	1199#	1205#	1246#	1252#
R1	=%000001	448#	885#	918#	967#	998#	1055#	1061#	1105#	1111#	1199#	1205#	1246#	1252#
		1440#	1446#	1454#	1503#	1509#	1517#	1568#	1570#	1590#	1607#	1613#	1749#	1754#
		1777#	1786#	1793#	1795#	1813#	1814#	1816#	1821#	1826#	1848#	1849#	1850#	1851#
		1852#	1853#	1854#	1860#	1865#	1878#	1880#	1897#	1954#	2072#	2073#	2402#	2407#
R2	=%000002	2415#	2417#	2419#	2422#	2425#	2428#	2501#	2538#	2849#	2872#	2073#	2402#	2407#
		449#	1551#	1574#	1845#	1846#	1848#	2403#	2408#	2416#	2418#	2420#	2426#	2427#
R3	=%000003	2850#	2871#	1439#	1456#	1502#	1519#	2355#	2356#	2357#	2360#	2361#	2365#	2367#
		450#	1439#	1456#	1502#	1519#	2355#	2356#	2357#	2360#	2361#	2365#	2367#	2369#
R4	=%000004	2371#	2763#	2772#	2778#	2779#	2782#	2787#	2788#	2789#	2798#	2851#	2870#	2871#
R5	=%000005	451#	2764#	2766#	2767#	2768#	2769#	2770#	2784#	2786#	2794#	2797#	2852#	2869#
		452#	867#	949#	1028#	1079#	1129#	1142#	1147#	1176#	1223#	1272#	1284#	1289#
		1319#	1331#	1336#	1361#	1372#	1384#	1389#	1413#	1422#	1434#	1448#	1485#	1497#
		1511#	1736#	1737#	1767#	1773#	1774#	1775#	1797#	1804#	1805#	1837#	1964#	1967#
		2041#	2042#	2045#	2071#	2075#	2765#	2771#	2773#	2775#	2776#	2777#	2778#	2796#
		2853#	2868#											
R6	=%000006	453#	455#	788#	789#	790#								
R7	=%000007	454#	456#											
SAVE4	001334	768#	777#	784#	831#									
SCORD0	012543	1423#	2222#											
SCORD1	012615	1486#	2231#											
SHALO	012276	1177#	2187#											
SHGL1	012347	1224#	2195#											
SHLO	012076	1029#	2162#											
SHL1	012154	1080#	2171#											
SHUFF	007776	884#	917#	956#	997#	1054#	1059#	1104#	1109#	1198#	1203#	1245#	1250#	1589#
		1606#	1612#	1748#	1753#	1787#	1792#	1820#	1825#	1845#				
SINEND	007241	1441#	1443#	1504#	1506#	1732#								
SP	=%00000E	455#	770#	792#	809#	817#	821#	844#	1776#	1777#	1795#	1796#	1864#	1865#

	1954	1955	1956*	2056*	2260*	2261*	2262	2269*	2272*	2273*	2277*	2278*	2282
	2285*	2289	2291	2293	2294*	2301	2303	2305*	2306*	2308*	2309*	2310*	2311*
	2312*	2327*	2328*	2331*	2332*	2333	2337*	2338*	2339	2342	2344	2346*	2355*
	2360	2371	2372*	2373*	2374*	2399*	2400*	2401*	2402*	2403*	2405	2409*	2411*
	2413	2421*	2422	2424	2425*	2427	2428	2429	2431*	2432*	2436	2438	2439
	2465	2486	2490*	2500*	2501*	2508	2509*	2520	2521*	2522*	2523*	2533*	2534*
	2539	2570	2591*	2594*	2611*	2616*	2637	2641*	2671*	2672*	2682*	2684	2685*
	2686*	2688	2690	2692	2698	2700*	2702*	2710*	2714	2718	2719	2723	2735*
	2756	2757	2758*	2763*	2764*	2765*	2771	2796	2797	2798	2799*	2800*	2815*
	2816	2848*	2849*	2850*	2851*	2852*	2853*	2854*	2855	2863*	2867	2868	2869
	2870	2871	2872	2873	2878*								
	430*	792	1966										
STACK =	001100												
STCORD	007436												
STKLMT=	177774												
SUBTST	001320												
SWR	001140												
	441*												
	762*	2010	2027										
	624*	790	811*	813	819*	826*	837	839	1884	1888	1934	2016	2256
	2293*	2438	2458	2472	2474	2478	2573	2585	2589	2854	2867*		
SMREG	000176	819	837	2256	2269								
SNO	000001												
SNO0	000001	494											
SNO1	000002	484*	494										
SNO2	000004	483*	493										
SNO3	000010	482*	492										
SNO4	000020	481*	491										
SNO5	000040	480*	490										
SNO6	000100	479*	489										
SNO7	000100	478*	488										
SNO8	000200	477*	487										
SNO9	000400	476*	486										
SNO9	001000	475*	485										
SNO1	000002	493*											
SNO10	002000	474*											
SNO11	004000	473*											
SNO12	010000	472*											
SNO13	020000	471*											
SNO14	040000	470*											
SNO15	100000	469*											
SNO15	000004	492*											
SNO15	000010	491*											
SNO15	000020	490*											
SNO15	000040	489*											
SNO15	000100	488*											
SNO15	000200	487*											
SNO15	000400	486*											
SNO15	001000	485*											
TBITVE=	000014	527*											
TERM	010552	1867*	1925	1950*	1962*								
TIMED	001316	761*	1907	2050									
TIME1	011162	1907*	1914*	2050*	2061*	2066*							
TIME2	011164	1908*	1911*	2051*	2059*	2067*							
TITLE	011220	834	2084*										
TITST	001714	831*											
TKVEC =	000060	534*											
TPVEC =	000064	535*											
TRAPVE=	000034	533*	798*	799*									
TRTVEC=	000014	528*											
TSTNUM	001314	759*	2241	2243	2565*								

SBOAGN	007006	1661	1667	1673#					
SENDAD	006776	570	1669#	2596					
SENDCT	006756	802	1663#						
SENDMG	007015	1665	1677#						
SENULL	007012	1676#							
SENV	001214	657#	2504	2528	2578	2673			
SENVN	001215	658#	824	2506	2675	2680			
SEOP	006674	1647#							
SEOPCT	006750	802#	1660#	1664					
SEFLG	001103	607#	2447	2476	2482#	2492	2566#	2601	
SERMAX	001115	613#	804#	2488#	2492				
SERORR	01430#	79#	2563#						
SERRPC	001116	614#	2241	2243	2570#	2571#	2572	2601	2616
SERRTB	001256	706#	2624						
SERTY	014500	2575	2609#						
SERTTL	001112	611#	2569#	2601					
SESCAP	001166	638#	803#	2487#	2592	2594	2601		
SETABL	001214	656#							
SETEND	001256	596	690#						
SFATAL	001176	649#	2532#						
SFFLG	014304	2495#	2498#	2526	2535#	2543#			
SFILLC	001156	632#	2698	2729					
SFILLS	001155	631#	2729						
SGADR	001120	615#							
SGDAT	001124	617#	1902#	1931#	1933#	1951#	2243		
SGET42	006766	1666#							
SGTSMR	012764	2268#	2836						
SHD	= 000300	427							
SHIBTS	001000	591#							
SHIOCT	013626	2426#	2437#						
SICNT	001104	608#							
SILLUP	015554	2846	2862	2881#					
SINTAG	001135	622#	2296	2385					
SITEMB	001114	612#	2572#	2580	2601	2613			
SLF	001172	641#	2370	2379	2438	2601	2729		
SLFLG	014303	2536#	2542#						
SLPADR	001106	609#	805#	1986#	2480#	2485#	2490	2492	2492
SLPERR	001110	610#	806#	1987#	2480	2486#	2492	2591	
SMAOR1	001226	674#							
SMAOR2	001232	678#							
SMAOR3	001236	681#							
SMAOR4	001242	684#							
SMAIL	001174	592	596	647#	823	2484	2578	2673	
SMAMS1	001224	668#							
SMAMS2	001230	676#							
SMAMS3	001234	679#							
SMAMS4	001240	682#							
SMBADR	001002	592#							
SNFLG	014302	2496#	2502	2537#	2541#				
SMNEJ	013457	2271	2383#						
SMSCAD	001210	654#	2512#	2515					
SMSCGC	001212	655#	2517#						
SMSCGY	001174	648#	2510	2518#	2530	2534#			
SMSMR	013446	2268	2381#						
SMTYP1	001225	669#							
SMTYP2	001231	677#							

.SDB20	18		
.SDIV	18		
.SEOP	18	4178	1637
.SERRO	18	4178	2500
.SERRT	18	4178	2602
.SMULT	18		
.SPARM	4178		
.SPOWE	18	4178	2842
.SRAND	18		
.SRDDE	18		
.SRDOC	18	4178	2385
.SREAD	18	4178	2246
.SR2AZ	18		
.SSAVE	18	4178	
.SSB2D	18		
.SSB20	18		
.SSCOP	18	4178	2442
.SSIZE	18		
.SSPAC	4178		
.SSLPR	18		
.SSMCO	4178		
.STRAP	18	4178	2807
.STYPB	18		
.STYPD	18		
.STYPE	18	4178	2650
.STYPO	18	4178	2730
.S4OCA	18		
.1170	18		

MAINDEC-11-DZVTD-B MACY11 27(732) 25-SEP-76 10:10 PAGE 83
DZVTD8.SRC CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ADD	856 2758	858 2768	860	1591	1608	1653	2056	2285	2294	2422	2509	2521	2533	2624	2686
ASL	1814	1983	2308	2309	2310	2415	2417	2419	2621	2622	2623	2819			
ASR	2014	2015	2029	2030	2516										
BEQ	825 1985	838 2013	1445 2265	1508 2292	1650 2307	1652 2410	1667 2473	1759 2475	1831 2477	1858 2479	1879 2503	1885 2507	1889 2527	1926 2529	1937 2567
BGT	2590	2593	2626	2631	2644	2676	2689	2724	2785						
BHI	1661	2304	2345	2412	2792										
BIC	846 1658	1977 1758	1830	1847	1855	1891	1921	1975	1980	2261	2278	2305	2332	2338	2346
BIS	2421	2782													
BISB	1856	1859	2312	2787	2788										
BIT	1743	1783	1811	1816	2613										
BITB	1857	1884	1888	2016	2458	2472	2478	2573	2589						
BLO	824	2506	2675	2680	2712										
BLOS	1979														
BLT	2358	2343	2414	2703	2793										
BMI	2302														
BNE	1910														
	791	814	832	888	921	970	1000	1063	1069	1113	1119	1207	1213	1254	1260
	1457	1520	1571	1575	1763	1835	1881	1883	1892	1901	1912	1915	1923	1929	1939
	1953	2017	2019	2032	2060	2074	2257	2263	2283	2290	2297	2334	2340	2362	2368
	2441	2459	2505	2511	2514	2531	2574	2579	2597	2614	2636	2674	2681	2683	2691
BPL	2699	2713	2720	2783	2866										
	840	1593	1610	1756	1765	1828	1871	1899	1935	1973	2021	2034	2054	2062	2259
BR	2275	2330	2336	2439	2586	2668	2717	2781							
	780	782	816	1071	1121	1215	1262	1442	1455	1505	1518	1760	1832	1887	1895
	1903	1916	1927	1932	1970	2286	2313	2315	2341	2364	2423	2436	2461	2467	2470
	2481	2497	2519	2584	2619	2646	2670	2696	2706	2715	2722	2759	2774	2795	2858
CLR	2882														
	776	777	789	803	823	829	848	850	1656	1747	1819	1866	1867	1868	1896
	1902	1908	1924	1933	1947	2011	2028	2051	2272	2273	2407	2408	2487	2612	2772
CLRB	2854														
	907	941	989	1020	1037	1066	1088	1116	1139	1169	1185	1210	1232	1257	1281
	1311	1328	1358	1381	1411	1431	1477	1494	1540	1561	1572	1594	1632	1766	1794
CMP	1836	2369	2432	2482	2535	2536	2537	2695	2721						
	770	790	813	837	845	887	920	969	1592	1609	1651	1762	1834	1922	1928
CMPB	1936	1938	2256	2262	2282	2289	2301	2303	2333	2339	2342	2344	2357	2468	2596
	1880	1882	1976	1978	2264	2296	2361	2367	2411	2413	2474	2504	2578	2673	2688
DEC	2690	2698	2719	2723											
	968	999	1062	1112	1206	1253	1454	1456	1517	1519	1570	1574	1659	1757	1764
DECB	1829	1911	1914	1981	2018	2020	2031	2033	2059	2061	2620				
EHT	2702	2705	2780	2791											
HALT	431														
INC	555	2587	2598	2669	2857	2881									
INCB	886	919	1657	1761	1833	1894	2311	2534	2569	2786	2794	2865			
IOT	1893	1930	2483	2566	2725										
JMP	432														
JSR	559	560	561	861	1654	1674	1942	1945	1949	1988					
	867	884	908	909	917	942	943	949	966	990					
	1038	1054	1059	1067	1070	1079	1089	1104	1109	1117	1120	1129	1021	1022	1028
	1147	1170	1171	1176	1186	1198	1203	1211	1214	1223	1233	1245	1140	1142	1145
	1272	1282	1284	1287	1289	1312	1313	1319	1329	1331	1334	1336	1250	1258	1261
	1365	1372	1382	1384	1387	1389	1412	1413	1416	1417	1422	1432	1359	1361	1364
	1453	1478	1479	1485	1495	1497	1500	1511	1516	1541	1542	1562	1434	1437	1448
	1606	1612	1633	1635	1669	1748	1753	1787	1792	1820	1825	1964	1573	1589	1595
													1967	1969	2042

.IIF	417	422	427	542	543	544	546	548	549	555	642	646	793	796	802
	803	805	806	1641	1642	1656	1657	1676	1679	2249	2270	2371	2379	2385	2438
	2448	2449	2450	2451	2452	2456	2483	2489	2492	2556	2557	2558	2559	2564	2588
.IRP	2596	2601	2617	2642	2729	2831	2832	2833	2834	2836	2838	2839	2840	2841	
	708	863	945	1024	1075	1125	1172	1219	1268	1315	1368	1418	1481	1545	1648
.LIST	2401	2427	2457	2500	2501	2522	2538	2539	2565	2848	2854	2867	2868		
	1	417	537	548	555	634	636	637	638	643	646	708	763	807	863
	867	945	949	1024	1028	1075	1079	1125	1129	1172	1176	1219	1223	1268	1272
	1315	1319	1368	1372	1418	1422	1481	1485	1545	1549	1656	1668	2348	2451	2596
	2823	2831	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842		
.MACRO	1	549	597	754	823	1544	2823								
.PCALL	417	537	643	807											
.NEXT	691														
.NLIST	1	417	537	548	555	634	636	637	638	643	646	708	763	807	863
	867	945	949	1024	1028	1075	1079	1125	1129	1172	1176	1219	1223	1268	1272
	1315	1319	1368	1372	1418	1422	1481	1485	1545	1549	1656	1668	2348	2451	2596
	2823	2831	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842		
.PAGE	597	692													
.REN	1														
.REPT	555	636	891	925	973	1004	1153	1295	1342	1395	1461	1524	1616		
.SBTTL	427	538	549	558	564	575	597	643	692	786	863	945	1024	1075	1125
	1172	1219	1268	1315	1368	1418	1481	1545	1637	2077	2246	2385	2442	2492	2550
	2602	2650	2730	2807	2823	2842									
.TITLE	417														
.WORD	555	556	557	572	591	592	593	594	595	596	605	608	609	610	611
	614	615	616	617	618	619	620	623	624	625	634	636	637	648	649
	650	651	652	653	654	655	659	660	661	674	678	681	684	685	686
	687	688	689	1144	1149	1286	1291	1333	1338	1363	1386	1391	1415	1436	1450
	1451	1499	1513	1514	1660	1663	1675	2434	2437	2524	2628	2633	2679	2726	2806
	2877	2879													

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

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

