

VTV30,VT30-H

VTV30J/H VT30H LOGIC
CVVTAA0

AH F654A MC

COPYRIGHT 1979

FICHE 1 OF 1

NOV 1979



MADE IN USA

1
2
3
4
5

000000

.REPT 0

IDENTIFICATION

PRODUCT CODE: AC-F652A-MC
PRODUCT NAME: CVVTAA0 VTV30J/H-VT30H LOGIC
PRODUCT DATE: OCT 1, 1979
MAINTAINER: COMPUTER SPECIAL SYSTEMS
DIGITAL EQUIPMENT CO. LTD.
READING
BERKS. U.K.

7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

COPYRIGHT (C) 1979 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE
USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF
SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE
COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER COPIES
THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE
TO ANY OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE
SOFTWARE IS HEREBY TRANSFERRED.

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

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

VTV30-H/J OR VT30-H LOGIC DIAGNOSTIC

PROGRAM DESCRIPTION

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. ABSTRACT

THIS IS THE FIRST PART OF A TWO PART DIAGNOSTIC FOR THE VTV30-H/J OR VT30-H GRAPHICS DISPLAY CONTROLLER. THIS PART CONTAINS THE LOGIC TESTS TO CHECK OUT AS MUCH OF THE HARDWARE AS POSSIBLE. THE PROGRAM WILL PRINT OUT AN ERROR MESSAGE WHENEVER A FAULT IS FOUND. AFTER SUCCESSFULLY RUNNING THIS PART OF THE DIAGNOSTIC, PART 2 SHOULD BE RUN. FURTHER LOGIC TESTS ARE CARRIED OUT IN PART 2.

2. REQUIREMENTS

2.1 EQUIPMENT

- A. PDP-11 COMPUTER
- B. CONSOLE TELETYPE
- C. VTV30-H/J OR VT30-H
- D. DIAGNOSTIC TAPE AND LISTINGS

2.2 STORAGE

THIS PROGRAM REQUIRES A MINIMUM OF 8K WORDS OF MEMORY.

3. LOADING PROCEDURE

THE PROGRAM IS LOADED USING THE ABSOLUTE BINARY LOADER AND IS IN ABSOLUTE BINARY FORMAT. THE PROGRAM CAN ALSO BE LOADED AND RUN IN THE NORMAL XXDP MANNER.

4. STARTING PROCEDURE

THE PROGRAM HAS A LOAD AND GO FEATURE WHICH AUTOMATICALLY STARTS THE PROGRAM AT ADDRESS 1000 UPON A SUCCESSFUL LOAD.

102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145

5. RESTARTING PROCEDURE

THE PROGRAM HAS A RESTART ADDRESS AT 1200 WHICH ALLOWS THE PROGRAM TO BE RESTARTED WITHOUT HAVING TO RE-ENTER THE BUS AND VECTOR ADDRESSES. IF IT IS NECESSARY TO RESTART THE PROGRAM WITH NEW BUS AND VECTOR ADDRESSES, THE ADDRESSES 1000 OR 200 SHOULD BE USED AS THE RESTART ADDRESS.

6. PROGRAM AND OPERATOR ACTION

THE FOLLOWING OPERATOR REQUESTS ARE MADE BY THE PROGRAM PRIOR TO THE COMMENCEMENT OF THE ACTUAL TESTS:-

TYPE 6 FOR 625-LINES OR 5 FOR 525-LINE DISPLAY
FIRST BUS ADDRESS IS
FIRST VECTOR ADDRESS IS.....
FIRST PRIORITY LEVEL IS

THE OPERATOR SHOULD REPLY TO REQUESTS ABOVE, BY INPUTTING THE CORRECT DATA.

'SELECT DESIRED SWITCH REGISTER SETTINGS.'

'TYPE CNTRL-C TO CONTINUE'

OR

'SWR = 0.'

IN REPLY TO THE REQUEST ABOVE THE OPERATOR SHOULD SELECT DESIRED SWITCH REGISTER OPTIONS AS SET OUT UNDER SWITCH OPTIONS BELOW.

7. SWITCH REGISTER OPTIONS

THIS PROGRAM IS DESIGNED TO RUN EQUALLY EASILY ON PDP-11 PROCESSORS WITH, OR WITHOUT, A HARDWARE SWITCH REGISTER. ON STARTING, A TEST IS DONE TO SEE IF A HARDWARE SWITCH REGISTER IS PRESENT. IF IT IS PRESENT, IT MAY BE USED IN THE NORMAL MANNER.

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
182
183
184
185

THE SWITCH REGISTER SETTINGS ARE:-

SWR15=1 INHIBIT ERROR HALT
SWR14=1 INHIBIT ERROR PRINT-OUT
SWR13=1 FAST ITERATION
SWR12=1 :
SWR11=1 :SCOPE LOOPS, SEE BELOW
SWR10=1 :FOR A DESCRIPTION, AND
SWR09=1 :APPENDIX A FOR EXAMPLES
SWR08=1 :OF THEIR USE
SWR07=1 :
SWR06=1 SELECTED TEST
SWR05=1 :
SWR04=1 :
SWR03=1 :TEST NOS.
SWR02=1 :
SWR01=1 :
SWR00=1 :

THE SETTING OF BITS 7, 8, 9, 10, 11, AND 12 IN THE SWITCH REGISTER ARE USED TO SELECT THE TRAP OPTIONS PRESENT IN THE PROGRAM. THE SELECTION IS MADE IN THE FOLLOWING MANNER:

| BIT(S) SET | TRAP FUNCTION SELECTED |
|--------------|---|
| 7 | TRAP+2 |
| 8 | TRAP+4 |
| 9 | TRAP+10 |
| 10 | TRAP+20 |
| 9 AND 10 | TRAP+30 |
| 11 | TRAP+40 |
| 9 AND 11 | TRAP+50 |
| 10 AND 11 | TRAP+60 |
| 9, 10 AND 11 | TRAP+70 |
| 12 | USES THE SWITCH REGISTER SETTING THAT WAS IN FORCE WHEN THE LAST TRAP INSTRUCTION WAS EXECUTED. |

187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238

IF A HARDWARE SWITCH REGISTER IS NOT PRESENT, THE PROGRAM ASSIGNS A LOCATION IN MEMORY AS A SOFTWARE SWITCH REGISTER, THE OPTIONS REMAINING AS ABOVE. THIS MEANS THAT ALL MODIFICATIONS TO THE SWITCH REGISTER MAY BE MADE USING THE CONSOLE TELETYPE VIA A MONITOR ROUTINE. THIS MONITOR IS CALLED BY TYPING CTRL-G AT THE CONSOLE TELETYPE AND RESPONDS BY PRINTING THE CURRENT CONTENTS OF THE SOFTWARE SWITCH REGISTER, FOLLOWED BY A PROMPT CHARACTER (>). THE OPERATOR SHOULD THEN TYPE IN THE NEW SWITCH REGISTER SETTINGS AS AN OCTAL NUMBER, FOLLOWED BY A CARRIAGE RETURN. TYPING CARRIAGE RETURN ALONE WILL CAUSE THE SETTING TO REMAIN UNCHANGED. THE SWITCH REGISTER IS THEN LOADED WITH THE NEW VALUE AND PROGRAM EXECUTION CONTINUES. IF A SETTING IS ENTERED WHICH INCLUDES THE SELECT TEST BIT (SWR06), THE TEST INDICATED BY SWR 00-05 WILL BE SELECTED IMMEDIATELY. THIS DOES NOT APPLY WHEN DEFAULTING ON AN EXISTING SETTING.

THE SWR MONITOR IS ALSO CALLED AUTOMATICALLY IF AN ERROR IS DETECTED AND SWR BIT 15 IS NOT SET. OCTAL EQUIVALENTS FOR THE SWITCHES ARE AS FOLLOWS:

| | |
|---------|--------|
| SWR15 = | 100000 |
| SWR14 = | 40000 |
| SWR13 = | 20000 |
| SWR12 = | 10000 |
| SWR11 = | 4000 |
| SWR10 = | 2000 |
| SWR09 = | 1000 |
| SWR08 = | 400 |
| SWR07 = | 200 |
| SWR06 = | 100 |
| SWR05 = | 40 |
| SWR04 = | 20 |
| SWR03 = | 10 |
| SWR02 = | 4 |
| SWR01 = | 2 |
| SWR00 = | 1 |

TO SET A COMBINATION OF THESE SWITCHES, SIMPLY ADD TOGETHER THE CORRESPONDING OCTAL NUMBERS AND ENTER THE TOTAL IN RESPONSE TO "SWR= X>". (LEADING ZEROS MAY BE IGNORED).

FOR WORST CASE TESTING, ALL SWITCHES SHOULD BE ZERO. IT IS POSSIBLE, WITH THESE SWITCH REGISTER OPTIONS, TO EXECUTE ONLY A PRE-SELECTED TEST WITH THE FACILITY TO LOOP ON THAT TEST OR TO START THE PROGRAM PASS OR FINISH THE PROGRAM PASS AT ANY PARTICULAR TEST.

240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289

8. ERROR REPORTS

THE FORMAT OF THE ERROR REPORTS IS AS FOLLOWS:-

| | |
|--------------|--------------------|
| E# AABB | AT PC:CCCC |
| GOOD: DDDD | BAD: EEEE |
| STATUS: FFFF | ADDRESS: GGGG |
| DATA: KKKK | CALLED FROM: HHHH |
| | ERROR COUNT = JJJJ |

WHERE:

AA IS THE TEST NUMBER
BB IS THE ERROR NUMBER
CCCC IS THE ADDRESS WHERE THE ERROR REPORT OCCURS.
DDDD IS THE DATA EXPECTED
EEEE IS THE DATA RECEIVED
FFFF GGGG AND KKKK ARE CONTENTS OF REGISTERS.
HHHH IS THE ADDRESS IN THE MAINLINE CODE WHERE THE
SUBROUTINE, WHERE THE ERROR REPORT IS
GENERATED, IS CALLED FROM.
JJJJ IS THE NUMBER OF ERRORS REPORTED TO DATE IN
THIS SECTION.

9. ONLINE MODIFICATIONS

AVAILABLE TO THE USER IS A ROUTINE TO MODIFY PROGRAM
LOCATIONS. IT IS ENTERED BY TYPING CNTRL-O DURING THE
RUNNING OF THE TESTS. ON ENTRY, A PROMPT '\$' IS MADE
FOR THE ADDRESS TO BE MODIFIED. IF NO ADDRESS IS GIVEN,
IT IS ASSUMED THAT NO MODIFICATIONS ARE REQUIRED AND THE
ROUTINE WILL COMPLETE. IF AN ADDRESS IS SPECIFIED, IT
WILL BE CHECKED TO SEE IF IT IS EVEN AND IN EXISTANCE.
ONCE IT HAS BEEN CHECKED, ITS CONTENTS ARE DISPLAYED AND
A PROMPT '/' IS MADE FOR THE NEW CONTENTS. IF NO NEW
VALUE IS GIVEN, THE EXISTING VALUE WILL BE LOADED.
HAVING DEALT WITH THAT ADDRESS, THE ROUTINE WILL THEN
EXAMINE THE TERMINATING CHARACTER TO DETERMINE THE NEXT
OPERATION TO PERFORM.

TYPING <ESC> WILL COMPLETE THE MODIFICATIONS BEING
DONE.
<CR> WILL CAUSE A PROMPT FOR THE NEXT ADDRESS
TO BE MODIFIED.
<LF> WILL TAKE THE NEXT ADDRESS TO BE MODIFIED
AS THE CURRENT ADDRESS+2.

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
325
326
327
328
329
330
331
332
333
334
335
336
337

10. PROGRAM DESCRIPTION

TEST 0

THIS TEST CHECKS ALL THE READ/WRITE BITS IN THE CONTROL AND STATUS REGISTER, FOR CORRECT SETTING AND CLEARING. IT ALSO CHECKS THE READ-ONLY 525-LINE BIT AGAINST THE ANSWER GIVEN TO THE 525/625 LINE QUESTION AT THE START OF THE PROGRAM.

- ERROR 0000 CSR DID NOT INITIALISE ON RESET
- 0001 525-LINE BIT READ INCORRECTLY
- 0002 CSR DATA ERROR WHEN TRYING TO SET BIT
- 0003 CSR DATA ERROR WHEN TRYING TO CLEAR BIT.

TEST 1

THIS TEST CHECKS THAT THE READY AND TIMER INTERRUPTS OPERATE CORRECTLY AND ON THE RIGHT LEVEL. IT DOES NOT CHECK THAT NO INTERRUPT OCCURS WHEN THE READY BIT IS CLEAR (THIS IS DONE IN TEST 5). THE LAST PART OF THIS TEST RINGS THE BELL ON THE CONSOLE DEVICE, TEN TIMES AT ONE SECOND INTERVALS. THE TIMER INTERRUPT IS USED TO SET THE INTERVAL, AND THE OPERATOR SHOULD CHECK THAT THE BELL RINGS ARE EVENLY SPACED OVER ABOUT NINE SECONDS.

- ERROR 1000 READY BIT DID NOT SET AFTER INITIALISE
- 1001 UNEXPECTED INTERRUPT WITH ENABLE CLEAR
- 1003 NO READY INTERRUPT AT DEVICE LEVEL
- 1005 NO TIMER INTERRUPT AT DEVICE LEVEL
- 1006 NO TIMER INTERRUPT WITHIN 40MS OF ENABLE BEING SET
- 1007 NO SECOND TIMER INTERRUPT IN 20MS OF BIT BEING CLEARED
- 1010 TIMER BIT WOULD NOT CLEAR AFTER INTERRUPT.

339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388

TEST 2

THE CORRECT LOADING, READING AND INCREMENTING OF THE CHARACTER STORE ADDRESS REGISTER IS CHECKED IN THIS TEST (NOTE THE ADDRESS WILL INCREMENT WHEN THE DATA BYTE IS LOADED, AND ALSO WHEN THE ADDRESS BYTE IS READ).

- ERROR 2000 CHARACTER STORE ADDRESS REGISTER NOT ZEROED BY INITIALISE
- 2001 CHARACTER STORE ADDRESS REGISTER DATA ERROR
- 2002 CHARACTER STORE ADDRESS INCREMENT ERROR WHEN LOADING AND READING CHARACTER REGISTER IN 8V MODE
- 2003 DITTO IN 6V MODE
- 2004 INVISIBLE BITS OF CHARACTER STORE ADDRESS (ROW IN CHARACTER) WERE NOT SET TO ZERO WHEN ADDRESS WAS LOADED.

TEST 3

THE ENTIRE CHARACTER STORE IS CHECKED IN THIS TEST BY LOADING EVERY LOCATION WITH RANDOM DATA AND THEN READING EACH LOCATION BACK AND CHECKING THE DATA.

- ERROR 3001 WRONG DATA READ BACK FROM CHARACTER STORE
- ADDRESS = CHARACTER NUMBER
STATUS = ROW OF CHARACTER

TEST 4

THIS TEST CHECKS THE CURSOR REGISTER FOR CORRECT LOADING AND READING. IT ALSO CHECKS FOR CORRECT INCREMENT WHEN LOADING THE PICTURE STORE AND THE EFFECT ON THE REGISTER OF LINE FEED, CARRIAGE RETURN AND HOME.

- ERROR 4001 LOAD/READ ERROR WITH CURSOR ADDRESS REGISTER
- 4002 REGISTER NOT CLEARED BY "HOME" COMMAND
- 4003 LOAD/READ ERROR ON X ADDRESS

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
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443

4004 X INCREMENT ERROR ON CHARACTER LOAD TO
PICTURE STORE (NOTE: NO INCREMENT SHOULD
OCCUR WHEN X ADDRESS IS AT MAXIMUM VALUE)

MAXIMUM X IN 6H = 79

MAXIMUM X IN 8H = 63

4005 DITTO ON "SET BLINK OFF INCREMENT"
COMMAND

4006 DITTO ON "SET BLINK ON INCREMENT" COMMAND

4007 X ADDRESS DID NOT ZERO ON "CARRIAGE
RETURN"

4010 Y ADDRESS LOAD/READ ERROR

4011 Y INCREMENT ERROR ON "LINE FEED" (NO
INCREMENT SHOULD OCCUR WHEN Y ADDRESS IS
AT MAXIMUM VALUE)

525L 625L

MAXIMUM Y IN 6V = 39 , 47

MAXIMUM Y IN 8V = 29 , 35

TEST 5

THIS TEST CHECKS EVERY LOCATION IN THE PICTURE STORE FOR
CORRECT LOADING AND READING. IT ALSO CHECKS THE
OPERATION OF PRESET AND THAT THE READY BIT IS CLEAR
DURING THE PRESET OPERATION AND THAT NO READY INTERRUPT
OCCURS WHILST READY IS CLEAR. NOTE THE DISPLAY IS
TURNED ON DURING THIS TEST.

ERROR 5001 PICTURE STORE DATA ERROR

ADDRESS = STORE X,Y ADDRESS OF LOCA-
TION WITH ERROR.

5002 READY BIT DID NOT CLEAR FOR PRESET

5003 NO READY INTERRUPT WITHIN 40MS AFTER
PRESET

5004 READY BIT WAS NOT SET WHEN INTERRUPT
OCCURRED

5005 PICTURE STORE DATA ERROR ATER PRESET

445
446
447
448

5006

PICTURE STORE DATA ERROR AFTER LOADING
PART OF THE PICTURE STORE, USING BLINK
ON/OFF WILL INCREMENT COMMAND (301 OR
300) WHEN BLINK IS REQUIRED.

450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501

APPENDIX A

PDP-11 DIAGNOSTIC LOOPING FACILITIES VIA
SWITCH REGISTER OPTIONS

N.B.

THE INTENTION OF THIS APPENDIX IS TO EXPLAIN, IN GENERAL, THE SCOPE FACILITIES WITHIN PROGRAM CODING. THE USER MUST FIRST EXAMINE THE CODING ABOUT THE AREA HE WISHES TO USE SCOPING FACILITIES, TO ASCERTAIN THAT THE PARTICULAR FACILITY HE REQUIRES IS, IN FACT, AVAILABLE.

PROGRAM LOOPING CONTROL CAN BE SELECTED BY USING SWR 12 - 07. THE PROGRAM HANDLES THIS BY USE OF THE TRAPSV ROUTINE, WHICH IS ENTERED USING THE TRAP INSTRUCTION. BASICALLY, THE ROUTINE CHECKS EQUALITY BETWEEN BITS 05 - 00 OF THE TRAP INSTRUCTION AND SWR 12 - 07.

THERE ARE THREE DISTINCT FUNCTIONS CONTROLLED BY THE TRAP ROUTINE:-

- A) RUN - (TRAP + 2 INSTRUCTION AND SWR 07 SET)
USUALLY USED TO INHIBIT TEST NUMBER PRINTOUT; USEFUL IN THE CASE OF NON-INTERVENTION TESTS. WHEN SWR 07 IS SET, ALL TEST NUMBER MESSAGES ARE SUPPRESSED.
NOTE: IT DOES NOT SUPPRESS ERROR PRINTOUTS.
- B) LOOP ON A SUB-TEST - (TRAP + 4 INSTRUCTION AND SWR 08 SET)
THIS IS GENERALLY USED TO ALLOW THE OPERATOR, BY SETTING SWR 08, TO CONTINUOUSLY LOOP ON ONE LOGICAL TEST OR GROUP OF TESTS.
- C) SCOPE ON A SUB-TEST - (TRAP + 10 - 70 INSTRUCTIONS AND SWR 09 - 11)
THIS ALLOWS THE OPERATOR TO SELECT SEVEN LEVELS OF LOOPING FACILITY WITHIN A SELECTION OR TEST.

IT IS USED TYPICALLY WITHIN A TEST WHERE ONE SUB TEST SETS A FLAG AND THE NEXT ONE CLEARS IT. BY USING SCOPE LEVEL 1 (SWR 09) - TRAP + 10), HE COULD LOOP ON THE FIRST SUB-TEST, SCOPE LEVEL 2 (SWR10) - TRAP + 20), HE COULD LOOP ON THE SECOND SUB-TEST OR SCOPE LEVEL 3 (SWR 10 & 09 - TRAP + 30) TO LOOP CONSTANTLY THROUGH BOTH SUB-TESTS SEQUENTIALLY. THE SCOPE RETURN ADDRESS ALWAYS APPEARS AS THE ARGUMENT OR NEXT WORD AFTER THE TRAP INSTRUCTION.

503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521

TO ALLOW THE SCOPE LEVEL TO BE CHANGED WITHOUT STOPPING THE PROGRAM, E.G. TO CHANGE FROM LEVEL 1 TO 2, WHICH WOULD ALMOST CERTAINLY CAUSE LEVEL 3 OR 0 TO BE SEEN MOMENTARILY), A 'PRESERVE SCOPE' FACILITY IS PROVIDED WITH SWR12. WHEN THIS IS SELECTED, THE PROGRAM NO LONGER INSPECTS SWR 11 - 09 BUT USES THE SETTING MEMORISED FROM BEFORE SWR 12 WAS SELECTED. THE SCOPE LEVEL MAY NOW BE CHANGED WITH NO EFFECT UNTIL SWR 12 IS SET TO 0, WHEN THE NEW SCOPE SETTING APPLIES.

N.B. SETTING SWR 12 SHOULD ONLY BE USED TO PRESERVE AS EXISTING SCOPE LEVEL, AS PREVIOUSLY SET ON THE SWITCH REGISTER.

WHICH SCOPE LEVEL TO SELECT MAY BE DETERMINED THE LISTING; LEVELS 1 THROUGH 7 ARE CALLED BY TRAP + 10 (SWR 09) THROUGH TRAP + 70 (SWR 09, 10 AND 11).

EXAMPLE OF SCOPE LOOP FACILITIES WITHIN DIAGNOSTIC

523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573

```

TEST10: MOV #40,COUNT ; ITERATION COUNT.
T101: BIS #1,STATUS ; SET GO BIT.
      BIT #1,STATUS ; IS GO BIT SET?
      BNE T10SCP ; YES, BRANCH.
      JSR PC,ERROR ; ERROR!!! GO NOT SET.
      10 ; ERROR NUMBER.
T10SCP: TRAP+10 ; SCOPE TEST10 IF LEVEL 1 SELECTED.
      T101 ; RETURN LABEL FOR SCOPE.
TEST11: BIC #1,STATUS ; CLEAR GO BIT
      BIT #1,STATUS ; GO BIT CLEAR ?
      BEQ T11SCP ; GO BIT CLEAR BRANCH.
      JSR PC,ERROR ; NO. GO BIT FAILED TO CLEAR.
      11 ; ERROR!!! NO. 11.
T11SCP: TRAP+30 ; SCOPE TEST 11 IF LEVEL 3 SELECTED.
      TEST11 ; RETURN LABEL FOR SCOPE.
      TRAP+20 ; SCOPE TESTS 10 & 11.
      T101 ; RETURN LABEL.
TEST12: BIS #100,STATUS ; NO SCOPE SELECTED. CARRY ON.
      BIT #100,STATUS ; IS DONE BIT SET ?
      (ETC.)
      .
      .
T12SCR: TRAP+10 ; SCOPE INT. EN. SET, LEVEL 1.
      TEST 12 ; RETURN LABEL.
TEST13: BIC #100,STATUS ; CLEAR INT ENABLE BIT.
      BIT #100,STATUS ; IS BIT CLEAR?
      (ETC.)
      .
      .
T13SCP: TRAP+30 ; SCOPE INT CLEAR LEVEL 3.
      TEST13 ; RETURN LABEL.
      TRAP+20 ; SCOPE INT. SET AND CLEAR.
      TEST12 ; LEVEL 2.
      TRAP+4 ; SCOPE LOOP ON THIS SET OF TESTS.
      T101 ; RETURN LABEL.
      DEC COUNT ; DO 40 TIMES ANYWAY
      BGT T101
TEST14: (ETC.)
      .
      .
      .

```


| | | .ENDR |
|-----|--------|----------------------------|
| | | .TITLE VTV LOGIC TESTS |
| | | .SBTTL GENERAL DEFINITIONS |
| 575 | | |
| 576 | | |
| 577 | | |
| 578 | | |
| 579 | | |
| 580 | 000000 | R0=%00 |
| 581 | 000001 | R1=%01 |
| 582 | 000002 | R2=%02 |
| 583 | 000003 | R3=%03 |
| 584 | 000004 | R4=%04 |
| 585 | 000005 | R5=%05 |
| 586 | 000006 | R6=%06 |
| 587 | 000007 | R7=%07 |
| 588 | 000006 | SP=%06 |
| 589 | 000007 | PC=%07 |
| 590 | 177776 | PSW=177776 |
| 591 | 177570 | HSWR=177570 |
| 592 | | |
| 593 | | |
| 594 | 172340 | PAR0=172340 |
| 595 | 172342 | PAR1=172342 |
| 596 | 172344 | PAR2=172344 |
| 597 | 172346 | PAR3=172346 |
| 598 | 172350 | PAR4=172350 |
| 599 | 172352 | PAR5=172352 |
| 600 | 172354 | PAR6=172354 |
| 601 | 172356 | PAR7=172356 |
| 602 | | |
| 603 | 172300 | PDR0=172300 |
| 604 | 172302 | PDR1=172302 |
| 605 | 172304 | PDR2=172304 |
| 606 | 172306 | PDR3=172306 |
| 607 | 172310 | PDR4=172310 |
| 608 | 172312 | PDR5=172312 |
| 609 | 172314 | PDR6=172314 |
| 610 | 172316 | PDR7=172316 |
| 611 | | |
| 612 | 177572 | SR0=177572 |
| 613 | 177574 | SR1=177574 |
| 614 | 177576 | SR2=177576 |
| 615 | | |
| 616 | 177546 | LKS=177546 |
| 617 | | |
| 618 | 177746 | CA1170=177746 |
| 619 | | |
| 620 | | |
| 621 | 001000 | REPCT1=1000 |
| 622 | 000100 | REPCT2=100 |
| 623 | 000002 | REPCT3=2 |
| 624 | | |
| 625 | | |
| 626 | | |
| 627 | 100000 | G=100000 |
| 628 | 040000 | D=40000 |
| 629 | 020000 | A=20000 |
| 630 | 010000 | S=10000 |

| | | |
|-----|--------|------------|
| 631 | 004000 | C=4000 |
| 632 | | |
| 633 | | |
| 634 | 177564 | TPS=177564 |
| 635 | 177566 | TPB=177566 |
| 636 | 177560 | TKS=177560 |
| 637 | 177562 | TKB=177562 |
| 638 | | |
| 639 | | |
| 640 | | |
| 641 | | |
| 642 | | |
| 643 | | .ENABL ABS |
| 644 | | .ENABL AMA |
| 645 | | |
| 646 | | |
| 647 | | |
| 648 | | |
| 649 | | |

MACROS

651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673

```
.SBTTL MACROS
:
: SET PROCESSOR PRIORITY
:
: .MACRO PSWSET $ARG,?LAB
MOV $ARG,-(SP) ; SET UP NEW PSW AS $ARG
MOV #LAB,-(SP) ; SET RETURN ADDRESS
RTI ; RTI TO SET PRIORITY
LAB: NOP ; RETURN ADDRESS
.ENDM
:
: READ PROCESSOR PRIORITY
:
: .MACRO PSWREA $ARG
EMT ; ISSUE EMT TO READ PSW
MOV FSAVPW,$ARG ; READ PSW IN $ARG
.ENDM
:
```



```

675                                     .SBTTL  INITIALISATION
676                                     .ASECT
677                                     .=0
678                                     .=200
679 000200 000137 001000             JMP    @#START      ; JMP TO START AT 200
680                                     .=1000
681
682
683 001000 012706 001000             START: MOV    #,SP      ; INITIALISE STACK POINTER
684 001004                                     PSWSET #340
685
686 001020 012737 014506 000004       MOV    #SWRSET,4    ; TEST FOR HARDWARE SWR
687 001026 012737 000340 000006       MOV    #340,6      ; TRAPS TO 4 IF IT
688 001034 012737 177570 014220       MOV    #HSWR,SWR   ; DOES NOT EXIST
689 001042 005777 013152              TST    @SWR
690 001046 005037 014246              CLR    SSWR        ; INITIALISE SOFTWARE SWR
691
692
693                                     ;
694                                     ; NOW ESTABLISH WHETHER OR NOT WE ARE RUNNING ON A SINGLE
695                                     ; INTERRUPT LEVEL LSI-11.
696 001052 004737 014442              JSR    PC,SILLSI   ; ESTABLISH PROCESSOR TYPE
697
698 001056 004737 014516              JSR    PC,VECTOR   ; FILL 0-574 WITH HALT TRAPS
699 001062 004737 016632              JSR    PC,TYPOUT
700 001066 012272                    GOMSG              ; DIAGNOSTIC
701 001070 004737 012062              JSR    PC,SET56
702
703
704 001074 004537 017510              JSR    R5,BUSSET   ; SET UP BUS AND VECTOR ADDRESSES
705 001100      007      004
706 001102 014222                    .BYTE  7,4
707 001104      001      001
708 001106 014242                    .WORD  CSR
709 001110 014244                    .BYTE  1,1
710 001112 000000                    .WORD  INTVEC
711                                     .WORD  VECLEV
712                                     .WORD  0
713
714 001114 013737 014226 014232       MOV    CAR,CARX
715 001122 013737 014226 014234       MOV    CAR,CARY
716 001130 005237 014234              INC    CARY
717 001134 013737 014230 014236       MOV    CHSR,CHDR
718 001142 013737 014230 014240       MOV    CHSR,CHAR
719 001150 005237 014240              INC    CHAR
720 001154 000137 001200              JMP    RSTART      ; THEN GOTO THE RESTART ADDRESS
721                                     .=1200
722 001200 012706 001000             RSTART: MOV    #START,SP ; INITIALISE STACK POINTER
723 001204                                     PSWSET #340
724
725 001220 004737 016632              JSR    PC,TYPOUT
726 001224 012736                    WMSG              ; SELECT DESIRED CONSOLE SWITCHES
727 001226 004737 015344             START1: JSR    PC,MONIT ; GO TO SWR MONITOR
728
729 001232 012706 001000             START2: MOV    #START,SP ; INITIALISE STACK POINTER
730 001236                                     PSWSET #340

```



```
731  
732 001252 032777 000100 012740 BIT #100,@SWR ;CHECK FOR PRE-SELECTED TEST  
733 001260 001002 BNE START3  
734  
735 001262 000137 001332 JMP TEST0 ;TO TEST 0  
736 001266 000137 001272 START3: JMP TABLE ;TO LOOK-UP TABLE  
737  
738  
739  
740  
741  
742 ;INTERFACE REGISTERS AND VECTORS  
743  
744  
745
```


747 001272 017700 012722
748 001276 042700 177700
749 001302 022700 000005
750 001306 002734
751
752 001310 006300
753 001312 000170 001316
754
755
756
757
758

TABLE: MOV @SWR,RO ;GET SELECTED TEST NO
BIC #177700,RO
CMP #5,RO ;CHECK FOR VALID TEST NO
BLT RSTART
ASL RO
JMP @TABLE1(RO) ;JUMP TO PRE-SELECTED TEST

759 001316 001332
760 001320 002776
761 001322 004710
762 001324 005734
763 001326 006260
764 001330 007676

TABLE1: TEST0
TEST1
TEST2
TEST3
TEST4
TEST5


```

766          .SBTTL  TESTO
767          ;*****
768          ;THIS TEST CHECKS ALL THE READ/WRITE
769          ;BITS IN THE CSR FOR SETTING & CLEARING
770          ;*****
771
772 001332 012737 000000 014216 TESTO: MOV #0,TESTNO ;SET UP TEST NO.
773 001340 104402          TRAP+2
774 001342 001350          T0000
775 001344 004737 014416          JSR PC,TESTR ;OUTPUT TEST NO
776
777 001350 005037 014346          T0000: CLR ERRDIS
778 001354 012704 000100          MOV #REPC2,R4
779 001360 004737 015104          JSR PC,FASTSW
780 001364 010437 014250          MOV R4,REPCNT ;SET UP TEST ITERATION COUNT
781
782 001370 000005          T0001: RESET
783 001372          PSWSET #200 ;TURN OFF INTERRUPTS
784 001406 017737 012610 014336          MOV @CSR,BAD ;READ CSR
785 001414 042737 002000 014336          BIC #2000,BAD ;CLEAR 525 LINE BIT
786 001422 022737 000200 014336          CMP #200,BAD ;BAD SHOULD NOW EQUAL 200 OCTAL
787 001430 001411          BEQ T0002 ;BRANCH IF IT DOES
788 001432 013737 014222 014344          MOV CSR,ADDRES ;SET UP FOR ERROR REPORT
789 001440 012737 000200 014334          MOV #200,GOOD
790 001446 004737 020544          JSR PC,ERROR ;ERROR0, CSR DID NOT
791 001452 120000          G+A+0 ;INITIALISE ON RESET
792 001454 104440          T0002: TRAP+40 ;SCOPE LOOP ON RESET (SWR BIT11)
793 001456 001370          T0001
794 001460 032777 002000 012534 T0003: BIT #2000,@CSR ;IS 525 LINE BIT SET
795 001466 001004          BNE 1$ ;BRANCH IF SET
796          ;*ENTER THIS CODING IF CSR
797          ;*READS SET TO 625 LINES
798 001470 005737 012156          TST L525 ;TEST TO SEE IF 625 LINES
799          ;WAS INPUT AT START REQUEST
800 001474 100033          BPL T0004 ;O.K. 625 LINE WAS SELECTED
801 001476 000403          BR 2$ ;BRANCH TO ERROR REPORT
802          ;ENTER THIS CODING IF
803          ;525 LINE WAS READ
804 001500 005737 012156          1$: TST L525 ;TEST TO SEE IF 525 LINE SELECTED
805 001504 100427          BMI T0004 ;O.K. 525 LINE WAS SELECTED
806 001506 013737 014222 014344 2$: MOV CSR,ADDRES ;SET UP FOR ERROR REPORT
807 001514 017737 012502 014340          MOV @CSR,DATA
808 001522 005737 012156          TST L525
809 001526 100406          BMI 3$ ;SELECT RIGHT ERROR MESSAGE
810 001530 104402          TRAP+2 ;SUPPRES PRINTOUT (SWR BIT 7 )
811 001532 001556          4$
812 001534 004737 016632          JSR PC,TYPOUT ;ERROR MESSAGE 625 LINE EXPECTED
813 001540 002500          ERMES1
814 001542 000405          BR 4$
815 001544 104402          3$: TRAP+2 ;SUPPRES PRINTOUT (SWR BIT 7 )
816 001546 001556          4$
817 001550 004737 016632          JSR PC,TYPOUT ;ERROR MESSAGE 525 LINE EXPECTED
818 001554 002564          ERMES2
819 001556 004737 020544          4$: JSR PC,ERROR ;ERROR 1 , WRONG STATE OF
820 001562 060001          A+D+1 ;525 LINE BIT.
821 001564 104450          T0004: TRAP+50 ;SCOPE LOOP ON READING CSR
    
```



```

822 001566 001460          T0003          ;(SWR BITS 9 & 11 )
823
824 001570 112737 000060 002706 T0005: MOVB  #'0,SETBIT      ;* SET UP
825 001576 112737 000060 002707      MOVB  #'0,SETBIT+1    ;* FOR ERROR
826 001604 112737 000060 002762      MOVB  #'0,CLRBIT     ;* REPORT
827 001612 112737 000060 002763      MOVB  #'0,CLRBIT+1   ;*
828 001620 012737 000001 002476      MOV   #1,BITS        ;SET TO TEST DISPLAY ON BIT
829 001626 004737 002210          JSR   PC,TSTBIT      ;SUBROUTINE FOR TESTING BIT
830
831
832 001632 112737 000060 002706      MOVB  #'0,SETBIT     ;* SET UP
833 001640 112737 000061 002707      MOVB  #'1,SETBIT+1   ;* FOR ERROR
834 001646 112737 000060 002762      MOVB  #'0,CLRBIT     ;* REPORT
835 001654 112737 000061 002763      MOVB  #'1,CLRBIT+1   ;*
836 001662 012737 000002 002476      MOV   #2,BITS        ;SET TO TEST CURSOR ON BIT
837 001670 004737 002210          JSR   PC,TSTBIT      ;SUBROUTINE FOR TESTING BIT
838
839
840 001674 112737 000060 002706      MOVB  #'0,SETBIT     ;* SET UP
841 001702 112737 000062 002707      MOVB  #'2,SETBIT+1   ;* FOR ERROR
842 001710 112737 000060 002762      MOVB  #'0,CLRBIT     ;* REPORT
843 001716 112737 000062 002763      MOVB  #'2,CLRBIT+1   ;*
844 001724 012737 000004 002476      MOV   #4,BITS        ;SET TO TEST CURSOR INC BIT
845 001732 004737 002210          JSR   PC,TSTBIT      ;SUBROUTINE TO TEST BIT
846
847
848
849 001736 112737 000060 002706      MOVB  #'0,SETBIT     ;* SET UP
850 001744 112737 000066 002707      MOVB  #'6,SETBIT+1   ;* FOR ERROR
851 001752 112737 000060 002762      MOVB  #'0,CLRBIT     ;* REPORT
852 001760 112737 000066 002763      MOVB  #'6,CLRBIT+1   ;*
853 001766 012737 000100 002476      MOV   #100,BITS      ;TEST READY INT. ENAB. BIT
854 001774 004737 002210          JSR   PC,TSTBIT      ;SUBROUTINE TO TEST BIT
855
856
857 002000 112737 000060 002706      MOVB  #'0,SETBIT     ;* SET UP
858 002006 112737 000070 002707      MOVB  #'8,SETBIT+1   ;* FOR ERROR
859 002014 112737 000060 002762      MOVB  #'0,CLRBIT     ;* REPORT
860 002022 112737 000070 002763      MOVB  #'8,CLRBIT+1   ;*
861 002030 012737 000400 002476      MOV   #400,BITS      ;SET TO TEST 6H BIT
862 002036 004737 002210          JSR   PC,TSTBIT      ;SUBROUTINE TO TEST BIT
863
864
865 002042 112737 000060 002706      MOVB  #'0,SETBIT     ;* SET UP
866 002050 112737 000071 002707      MOVB  #'9,SETBIT+1   ;* FOR ERROR
867 002056 112737 000060 002762      MOVB  #'0,CLRBIT     ;* REPORT
868 002064 112737 000071 002763      MOVB  #'9,CLRBIT+1   ;*
869 002072 012737 001000 002476      MOV   #1000,BITS     ;SET TO TEST 6V BIT
870 002100 004737 002210          JSR   PC,TSTBIT      ;SUBROUTINE TO TEST BITS
871
872
873 002104 112737 000061 002706      MOVB  #'1,SETBIT     ;* SET UP
874 002112 112737 000065 002707      MOVB  #'5,SETBIT+1   ;* FOR ERROR
875 002120 112737 000061 002762      MOVB  #'1,CLRBIT     ;* REPORT
876 002126 112737 000065 002763      MOVB  #'5,CLRBIT+1   ;*
877 002134 012737 100000 002476      MOV   #100000,BITS   ;SET UP FOR TESTING TIMER BIT
    
```

```
878 002142 004737 002210 JSR PC,TSTBIT ;SUBROUTINE FOR TESTING BIT
879
880
881
882 002146 104460 TRAP+60 ;SCOPE LOOP ON SETTING AND
883 002150 001570 T0005 ;CLEARING BITS IN CSR
884 ;(SWR BITS 10 &11 )
885
886
887 002152 104404 TRAP+4 ;SET SWR BIT 8 TO
888 002154 001370 T0001 ;LOOP ON TEST
889 002156 005337 014250 DEC REPCNT ;DONE ENOUGH ?
890 002162 001402 BEQ T0007 ;YES
891 002164 000137 001370 JMP T0001 ;NO
892
893 002170 032777 000100 012022 T0007: BIT #100,@SWR ;CHECK FOR PRE-SELECTED TEST
894 002176 001402 BEQ T0008 ;
895 002200 000137 001232 JMP START2
896 002204 000137 002776 T0008: JMP TEST1
897
898
899
```



```

901
902
903
904
905
906
907
908 002210 012737 000200 014334 TSTRIT: MOV #200,GOOD ;*SET UP
909 002216 005737 012156 TST L525 ;*GOOD WITH BITS
910 002222 100003 BPL TSTB1 ;*THAT SHOULD ALWAYS
911 002224 052737 002000 014334 BIS #2000,GOOD ;*BE READ AS 1
912 002232 005077 011764 TSTB1: CLR @CSR ;
913 002236 053777 002476 011756 BIS BITS,@CSR ;TRY TO SET BIT IN CSR
914 002244 017737 011752 014336 MOV @CSR,BAD ;READ CSR
915 002252 053737 002476 014334 BIS BITS,GOOD ;SET UP GOOD DATA
916 002260 023737 014334 014336 CMP GOOD,BAD ;DID CSR CONTAIN EXPECTED DATA
917 002266 001415 BEQ TSTB2 ;BRANCH IF O.K.
918 002270 013737 014222 014344 MOV CSR,ADDRES ;SET UP FOR ERROR REPORT
919 002276 011637 014352 MOV @SP,CALLPC ;
920 002302 104402 TRAP+2 ;INHIBIT PRINTOUT (SWR BIT 7 )
921 002304 002314 1$ ;
922 002306 004737 016632 JSR PC,TYPEOUT ;ERROR WHEN TRYING TO SET
923 002312 002650 ERMES3 ;BIT ** IN CSR
924 002314 004737 020544 1$: JSR PC,ERROR ;ERROR 2
925 002320 124002 C+A+G+2 ;
926 002322 104410 TSTB2: TRAP+10 ;SCOPE LOOP ON SETTING BIT
927 002324 002232 TSTB1 ;(SWR BIT 9 )
928
929 002326 012777 177767 011666 TSTB3: MOV #177767,@CSR ;SET ALL BITS IN CSR
930 002334 012737 141707 014334 MOV #141707,GOOD ;SET UP GOOD
931 002342 005037 002474 CLR RETRY ;CLEAR RETRY COUNT
932 002346 005737 012156 TST L525 ;
933 002352 100003 BPL 1$ ;
934 002354 052737 002000 014334 BIS #2000,GOOD ;
935 002362 043777 002476 011632 1$: BIC BITS,@CSR ;TRY TO CLEAR BIT IN CSR
936 002370 017737 011626 014336 MOV @CSR,BAD ;READ CSR
937 002376 043737 002476 014334 BIC BITS,GOOD ;SET UP GOOD DATA
938 002404 023737 014334 014336 CMP GOOD,BAD ;DID CSR CONTAIN EXPECTED DATA
939 002412 001423 BEQ TSTB4 ;BRANCH IF O.K.
940 002414 005737 002476 TST BITS ;IF WE ARE NOT TESTING TIMER ENABLE
941 002420 100003 BPL 3$ ;THEN THIS IS A LEGITIMATE ERROR
942 002422 005137 002474 COM RETRY ;ELSE ALLOW ONE RETRY
943 002426 001355 BNE 1$ ;BECAUSE THE TIMER MIGHT HAVE FIRED
944 002430 011637 014352 3$: MOV @SP,CALLPC ;SET UP FOR ERROR REPORT
945 002434 013737 014222 014344 MOV CSR,ADDRES ;
946 002442 104402 TRAP+2 ;INHIBIT PRINTOUT (SWR BIT 7 )
947 002444 002454 2$ ;
948 002446 004737 016632 JSR PC,TYPEOUT ;ERROR WHEN TRYING TO CLEAR
949 002452 002722 ERMES4 ;BIT ** IN CSR
950 002454 004737 020544 2$: JSR PC,ERROR ;ERROR 3
951 002460 124003 C+A+G+3 ;
952
953 002462 104420 TSTB4: TRAP+20 ;SCOPE LOOP ON CLEARING BIT
954 002464 002326 TSTB3 ;(SWR BIT 10 )
955 002466 104430 TRAP+30 ;SCOPE LOOP ON SETTING AND CLEARING
956 002470 002232 TSTB1 ;BIT IN CSR (SWR BITS 9 & 10 )

```

```
957 002472 000207          RTS      PC          ;END OF SUBROUTINE.
958
959
960 002474 000000          RETRY: 0
961 002476 000000          BITS:  0
962
963
964 002500 052133 051505 044524  ERMES1: .NLIST BEX
965 002564 052133 051505 044524  ERMES2: .ASCII /[TESTING FOR 625 LINES, CSR BIT 10 READS 525 LINES[@/
966 002650 042533 051122 051117  ERMES3: .ASCII /[TESTING FOR 525 LINES, CSR BIT 10 READS 625 LINES[@/
967                                     .ASCII /[ERROR WHEN TRYING TO SET BIT /
968 002706 000000          SETBIT: .EVEN
969 002710 044440 020116 051503  SETBIT: .WORD 0
970 002722 042533 051122 051117  ERMES4: .ASCII / IN CSR [a/
971                                     .EVEN
972 002762 000000          CLRBIT: .WORD 0
973 002764 044440 020116 051503  CLRBIT: .ASCII / IN CSR [a/
974                                     .EVEN
975                                     .LIST BEX
976
```



```

978          .SBTTL TEST1
979          :*****
980          :THIS TEST CHECKS THE READY AND TIMER
981          :INTERRUPTS. IT DOES NOT CHECK THAT
982          :NO INTERRUPT OCCURES WHEN THE READY
983          :BIT IS CLEAR (THIS IS DONE IN TEST 5).
984          :*****
985
986 002776 012737 000001 014216 TEST1: MOV #1,TESTNO ;SET UP TEST NO.
987 003004 104402          TRAP+2
988 003006 003014          T1000
989 003010 004737 014416          JSR PC,TESTR ;OUTPUT TEST NO
990
991 003014 005037 014346          T1000: CLR ERDIS
992 003020 012704 000100          MOV #REPC2,R4
993 003024 004737 015104          JSR PC,FASTSW
994 003030 010437 014250          MOV R4,REPCNT ;SET UP TEST ITERATION COUNT
995
996 003034 000005          T1001: RESET ;INITALISE HARDWARE
997 003036          PSWSET #200 ;TURN OFF INTERUPTS
998 003052 013700 014242          MOV INTVEC,R0
999 003056 012720 004206          MOV #INTSRV,(R0)+ ;SET UP INTERRUPT VECTOR
1000 003062 012710 000200          MOV #200,(R0)
1001 003066 017737 011130 014336          MOV @CSR,BAD ;READ CSR
1002 003074 032737 000200 014336          BIT #200,BAD ;WAS READY BIT SET
1003 003102 001017          BNE 2$ ;BRANCH IF O.K.
1004 003104 013737 014222 014344          MOV CSR,ADDRES ;* SET UP
1005 003112 012737 000200 014334          MOV #200,GOOD ;* FOR ERROR
1006 003120 005737 012156          TST L525 ;* REPORT
1007 003124 100003          BPL 1$
1008 003126 052737 002000 014334          BIS #2000,GOOD
1009 003134 004737 020544          1$: JSR PC,ERROR ;ERROR 10 , READY BIT NOT SET
1010 003140 120000          A+G+0
1011 003142 104410          2$: TRAP+10 ;SCOPE LOOP ON RESET (SWR BIT 9 )
1012 003144 003034          T1001
1013
1014
1015 003146 005037 004272          T1002: CLR INTFLG ;CLEAR INTERRUPT FLAG
1016 003152 017737 011044 014342          MOV @CSR,STATUS
1017 003160          PSWSET #0 ;TURN ON INTERRUPTS
1018 003174 000240          NOP ;WAIT FOR INTERRUPT
1019 003176 000240          NOP
1020 003200          PSWSET #200 ;TURN OFF INTERUPTS
1021 003214 005737 004272          TST INTFLG ;DID IT INTERRUPT ?
1022 003220 001413          BEQ T1004 ;BRANCH IF NO INTERRUPT OCCURED
1023 003222 013737 014222 014344          MOV CSR,ADDRES ;SET UP FOR ERROR REPORT
1024 003230 104402          TRAP+2 ;INHIBIT ERROR MESSAGE
1025 003232 003242          T1003
1026 003234 004737 016632          JSR PC,TYP0UT ;INTERRUPT WHEN ENABLE CLEAR
1027 003240 004361          ERMS10
1028 003242 004737 020544          T1003: JSR PC,ERROR ;ERROR 1 , UNEXPECTED INTERRUPT
1029 003246 030001          A+S+1
1030 003250 104420          T1004: TRAP+20 ;SCOPE LOOP ON INTERRUPT (SWR BIT 10 )
1031 003252 003146          T1002
1032
1033

```


| | | | | | | | | |
|------|--------|--------|--------|--------|---------|---------|----------------|--------------------------------|
| 1059 | 003370 | 005037 | 004272 | | T1012: | CLR | INTFLG | :CLEAR INTERRUPT FLAG |
| 1060 | 003374 | 052777 | 100000 | 010620 | | BIS | #100000,@CSR | :SER TIMER BIT |
| 1061 | 003402 | 017737 | 010614 | 014342 | | MOV | @CSR,STATUS | : |
| 1062 | 003410 | | | | | PSWSET | #0 | :TURN ON INTERUPTS |
| 1063 | 003424 | 000240 | | | | NOP | | :WAIT FOR INTERRUPT |
| 1064 | 003426 | 000240 | | | | NOP | | : |
| 1065 | 003430 | | | | | PSWSET | #200 | :TURN OFF INTERUPTS |
| 1066 | 003444 | 005737 | 004272 | | | TST | INTFLG | :TEST IF INTERRUPT OCCURED |
| 1067 | 003450 | 001013 | | | | BNE | T1013 | :BRANCH IF INTERRUPT OCCURED |
| 1068 | 003452 | 013737 | 014222 | 014344 | | MOV | CSR,ADDRE | :SET UP FOR ERROR REPORT |
| 1069 | 003460 | 104402 | | | | TRAP+2 | | :INHIBIT ERROR MESSAGE |
| 1070 | 003462 | 003472 | | | | 1\$ | | : (SWR BIT 7) |
| 1071 | 003464 | 004737 | 016632 | | | JSR | PC,TYPOUT | :NO INTERRUPT AT DEVICE LEVEL |
| 1072 | 003470 | 004473 | | | | | ERMS12 | : |
| 1073 | 003472 | 004737 | 020544 | | 1\$: | JSR | PC,ERROR | :ERROR 15 , |
| 1074 | 003476 | 030005 | | | | | S+A+5 | : |
| 1075 | 003500 | 104460 | | | T1013: | TRAP+60 | | :SCOPE LOOP ON TIMER INTERRUPT |
| 1076 | 003502 | 003370 | | | | T1012 | | : (SWR BITS 10 &11) |
| 1077 | | | | | | | | |
| 1078 | | | | | | | | |
| 1079 | | | | | | | | |
| 1080 | | | | | | | | |
| 1081 | | | | | | | | |
| 1082 | | | | | | | | |
| 1083 | | | | | | | | |
| 1084 | | | | | | | | |
| 1085 | 003504 | 005037 | 004202 | | T1014: | CLR | WTC1 | :*SET UP |
| 1086 | 003510 | 012737 | 000002 | 004204 | | MOV | #2,WTCO | :*TIME OUT |
| 1087 | 003516 | 005037 | 004272 | | | CLR | INTFLG | :CLEAR INTERRUPT FLAG |
| 1088 | 003522 | 012777 | 040000 | 010472 | | MOV | #40000,@CSR | :ENABLE TIMER INTERRUPT |
| 1089 | 003530 | | | | | PSWSET | #0 | :TURN ON INTERUPTS |
| 1090 | 003544 | 004737 | 004156 | | T1015: | JSR | PC,WAIT | :WAIT FOR TIME OUT |
| 1091 | 003550 | 103004 | | | | BCC | T1016 | : BRANCH IF TIMED OUT |
| 1092 | 003552 | 005737 | 004272 | | | TST | INTFLG | :TEST FOR INTERRUPT |
| 1093 | 003556 | 001027 | | | | BNE | T1017 | :BRANCH IF INTERRUPT O.K. |
| 1094 | 003560 | 000771 | | | | BR | T1015 | :GO BACK AND WAIT SOME MORE |
| 1095 | 003562 | | | | T1016: | PSWSET | #200 | :TURN OFF INTERUPTS |
| 1096 | 003576 | 017737 | 010420 | 014342 | | MOV | @CSR,STATUS | :SET UP FOR ERROR REPORT |
| 1097 | 003604 | 013737 | 014222 | 014344 | | MOV | CSR,ADDRES | : |
| 1098 | 003612 | 005077 | 010404 | | | CLR | @CSR | :CLEAR CSR |
| 1099 | 003616 | 104402 | | | | TRAP+2 | | :INHIBIT ERRER PRINTOUT |
| 1100 | 003620 | 003630 | | | | 1\$ | | : |
| 1101 | 003622 | 004737 | 016632 | | | JSR | PC,TYPOUT | :NO INTERRUPT BEFORE TIMEOUT |
| 1102 | 003626 | 004531 | | | | | ERMS13 | : |
| 1103 | 003630 | 004737 | 020544 | | 1\$: | JSR | PC,ERROR | :ERROR 16 |
| 1104 | 003634 | 030006 | | | | | S+A+6 | : |
| 1105 | 003636 | 032737 | 100000 | 014342 | T1017: | BIT | #100000,STATUS | :WAS TIMER BIT CLEARED |
| 1106 | 003644 | 001415 | | | | BEQ | T1017A | :YES,SO BRANCH |
| 1107 | 003646 | 013737 | 014222 | 014344 | | MOV | CSR,ADDRES | :NO,SO SET ERROR REPORT |
| 1108 | 003654 | 032777 | 040000 | 010336 | | BIT | #40000,@SWR | :SWR BIT 14 TO |
| 1109 | 003662 | 001003 | | | | BNE | 1\$ | :INHIBIT PRINTOUT |
| 1110 | 003664 | 004737 | 016632 | | | JSR | PC,TYPOUT | :ERROR TIMER BIT WAS NOT |
| 1111 | 003670 | 004602 | | | | | ERMS14 | :CLEARED BY BIC |
| 1112 | 003672 | 004737 | 020544 | | 1\$: | JSR | PC,ERROR | :ERROR 10, |
| 1113 | 003676 | 030010 | | | | | A+S+10 ; | : |
| 1114 | 003700 | 104470 | | | T1017A: | TRAP+70 | | :SCOPE LOOP ON INTERRUPT |

```

1115 003702 003504          T1014          ;(SWR BITS 9 , 10 & 11 )
1116
1117
1118
1119 003704 005337 014250    DEC      REPCNT    ;DONE ENOUGH ?
1120 003710 001402          BEQ      T1018    ;YES
1121 003712 000137 003034    JMP      T1001    ;NO
1122
1123
1124

```


| | | | | | | | | |
|------|--------|--------|--------|--------|---------|---------|---------------|----------------------------------|
| 1126 | 003716 | 013700 | 014242 | | T1018: | MOV | INTVEC,RO | :*SET UP |
| 1127 | 003722 | 012720 | 004230 | | | MOV | #TIME1,(RO)+ | :*INTERRUPT |
| 1128 | 003726 | 012710 | 000340 | | | MOV | #340,(RO) | :*VECTOR |
| 1129 | 003732 | 005737 | 012156 | | | TST | L525 | :TEST FOR 625 OR 525 LINES |
| 1130 | 003736 | 100404 | | | | BMI | 1\$ | :BRANCH IF 525 LINES |
| 1131 | 003740 | 012737 | 000062 | 004270 | | MOV | #50.,SECOND | :SET 50 INTERRUPTS = 1 SECOND |
| 1132 | 003746 | 000403 | | | | BR | 2\$ | : |
| 1133 | 003750 | 012737 | 000074 | 004270 | 1\$: | MOV | #60.,SECOND | :SET 60 INTERRUPTS = 1 SECOND |
| 1134 | 003756 | 004737 | 016632 | | 2\$: | JSR | PC,TYPOUT | :PRINT INSTRUCTIONS |
| 1135 | 003762 | 004276 | | | | | BELLS | :BELLS WILL RING EVERY SECOND |
| 1136 | | | | | | | | :FOR 10 SECONDS |
| 1137 | 003764 | 012737 | 000012 | 004266 | T1019: | MOV | #10.,COUNT2 | : |
| 1138 | 003772 | 013737 | 004270 | 004264 | | MOV | SECOND,COUNT1 | : |
| 1139 | 004000 | 005037 | 004272 | | | CLR | INTFLG | :CLEAR INTERRUPT FLAG |
| 1140 | 004004 | 012777 | 040000 | 010210 | | MOV | #40000,@CSR | :ENABLE TIMER INTERRUPT |
| 1141 | 004012 | 005037 | 004202 | | T1019A: | CLR | WTC1 | :*SET UP |
| 1142 | 004016 | 012737 | 000002 | 004204 | | MOV | #2,WTC0 | :*TIME OUT |
| 1143 | 004024 | | | | | PSWSET | #0 | :TURN ON INTERUPTS |
| 1144 | 004040 | 004737 | 004156 | | T1020: | JSR | PC,WAIT | :WAIT FOR TIMEOUT |
| 1145 | 004044 | 103025 | | | | BCC | T1022 | :TIMED OUT WAITING FOR INTERRUPT |
| 1146 | 004046 | 005737 | 004272 | | | TST | INTFLG | :TEST FOR INTERRUPT |
| 1147 | 004052 | 001001 | | | | BNE | T1021 | :O.K. BRANCH IF INTERRUPT |
| 1148 | 004054 | 000771 | | | | BR | T1020 | :NO INTERRUPT SO WAIT SOME MORE |
| 1149 | 004056 | 005037 | 004272 | | T1021: | CLR | INTFLG | :CLEAR INTERRUPT FLAG |
| 1150 | 004062 | 005737 | 004274 | | | TST | BELLS | :SHOULD WE RING BELL ? |
| 1151 | 004066 | 001764 | | | | BEQ | T1020 | :BRANCH IF NO |
| 1152 | 004070 | 012700 | 000007 | | | MOV | #7,RO | :*RING |
| 1153 | 004074 | 004737 | 016710 | | | JSR | PC,PCHR | :*BELL |
| 1154 | 004100 | 005037 | 004274 | | | CLR | BELLS | :CLEAR BELL FLAG |
| 1155 | 004104 | 005337 | 004266 | | | DEC | COUNT2 | :DECREMENT BELL COUNT |
| 1156 | 004110 | 001340 | | | | BNE | T1019A | :BRANCH IF MORE RINGS REQUIRED |
| 1157 | 004112 | 005077 | 010104 | | | CLR | @CSR | :DONE ENOUGH SO CLEAR CSR |
| 1158 | 004116 | 000403 | | | | BR | T1023 | :COMPLETE BRANCH TO END OF TEST |
| 1159 | | | | | | | | |
| 1160 | 004120 | 004737 | 020544 | | T1022: | JSR | PC,ERROR | :ERROR TIME OUT |
| 1161 | 004124 | 000007 | | | | | 7 | |
| 1162 | | | | | | | | |
| 1163 | 004126 | 104410 | | | T1023: | TRAP+10 | | :LOOP ON BELL |
| 1164 | 004130 | 003764 | | | | T1019 | | |
| 1165 | | | | | | | | |
| 1166 | 004132 | 104404 | | | T10WW: | TRAP+4 | | :SET SWR BIT 8 TO |
| 1167 | 004134 | 003034 | | | | T1001 | | :LOOP ON TEST |
| 1168 | 004136 | 032777 | 000100 | 010054 | | BIT | #100,@SWR | :CHECK FOR PRE-SELECTED TEST |
| 1169 | 004144 | 001402 | | | | BEQ | 1\$ | |
| 1170 | 004146 | 000137 | 001232 | | | JMP | START2 | |
| 1171 | 004152 | 000137 | 004710 | | 1\$: | JMP | TEST2 | : |
| 1172 | | | | | | | | |
| 1173 | | | | | | | | |
| 1174 | | | | | | | | |
| 1175 | 004156 | 005337 | 004202 | | WAIT: | DEC | WTC1 | :DECREMENT TEMPORARY COUNT |
| 1176 | 004162 | 001005 | | | | BNE | 5\$ | :EXIT IF NOT ZERO |
| 1177 | 004164 | 005337 | 004204 | | | DEC | WTC0 | :DECREMENT TIMER COUNT |
| 1178 | 004170 | 001002 | | | | BNE | 5\$ | :EXIT IF NOT ZERO |
| 1179 | 004172 | 000241 | | | | CLC | | :TIMED OUT |
| 1180 | 004174 | 000207 | | | | RTS | PC | :EXIT RETURN |
| 1181 | 004176 | 000261 | | | 5\$: | SEC | | :NOT TIMED OUT EXIT |

```

1182 004200 000207          RTS      PC      ;
1183
1184 004202 000000          WTC1:   0
1185 004204 000000          WTC0:   0
1186
1187
1188 004206 005237 004272    INTSRV: INC    INTFLG      ;SET INTERRUPT FLAG
1189 004212 042777 140100 010002    BIC    #140100,@CSR ;CLEAR INTERRUPT & ENABLE BITS
1190 004220 017737 007776 014342    MOV    @CSR,STATUS ;SAVE CONTENTS OF CSR
1191 004226 000002          RTI
1192
1193 004230 042777 100000 007764    TIME1: BIC    #100000,@CSR ;CLEAR TIMER BIT
1194 004236 005237 004272          INC    INTFLG      ;SET INTERRUPT BIT
1195 004242 005337 004264          DEC    COUNT1
1196 004246 001005          BNE    TIME1A      ;IS 1 SECOND UP
1197 004250 005237 004274          INC    BELLS
1198 004254 013737 004270 004264    MOV    SECOND,COUNT1 ;YES , SO SET BELL
1199 004262 000002          TIME1A: RTI
1200
1201 004264 000000          COUNT1: 0
1202 004266 000000          COUNT2: 0
1203 004270 000000          SECOND: 0
1204 004272 000000          INTFLG: 0
1205 004274 000000          BELLS:  0
1206
1207
1208
1209 004276 041133 046105 020114    BELLMS: .NLIST BEX
1210 004361      133 044440 052116    ERMS10: .ASCII /[BELL SHOULD RING 10 TIMES AT 1 SECOND INTERVALS @/
1211 004427      133 044440 052116    ERMS11: .ASCII /[ INTERRUPT OCCURED WITH ENABLE CLEAR@/
1212 004473      133 047040 020117    ERMS12: .ASCII /[ INTERRUPT OCCURED AT WRONG LEVEL @/
1213 004531      133 047040 020117    ERMS13: .ASCII /[ NO INTERRUPT WHEN EXPECTED @/
1214 004602 020133 044524 042515    ERMS14: .ASCII /[ NO TIMER INTERRUPT IN 20 MILLISECONDS @/
1215 004660 040440 052106 051105    .ASCII /[ TIMER BIT WAS NOT CLEARED BY BIC INSTRUCTION/
1216                                     .ASCII / AFTER TIMER INTERRUPT @/
1217                                     .EVEN
                                     .LIST BEX

```



```

1219 .SBTTL TEST2
1220 ;*****
1221 ;THIS TEST CHECKS THE OPERATION OF
1222 ;THE CHARACTER STORE ADDRESS REGISTER
1223 ;IN LOADING , READING & INCREMENTING.
1224 ;*****
1225
1226 004710 012737 000002 014216 TEST2: MOV #2,TESTNO ;SET UP TEST NO.
1227 004716 104402 TRAP+2
1228 004720 004726 T2000
1229 004722 004737 014416 JSR PC,TESTR ;OUTPUT TEST NO
1230
1231 004726 005037 014346 T2000: CLR ERRDIS
1232 004732 012704 000100 MOV #REPC2,R4
1233 004736 004737 015104 JSR PC,FASTSW
1234 004742 010437 014250 MOV R4,REPCNT ;SET UP TEST ITERATION COUNT
1235
1236 004746 000005 T2001: RESET
1237 004750 117737 007264 014336 MOVB @CHAR,BAD ;READ CHARACTER STORE ADDRESS REGISTER
1238 004756 105037 014337 CLRB BAD+1 ;
1239 004762 005737 014336 TST BAD ;CHECK WHAT WAS READ
1240 004766 001417 BEQ T2002 ;BRANCH IF IT WAS ZERO
1241 004770 013737 014240 014344 MOV CHAR,ADDRES ;SET UP FOR ERROR REPORT
1242 004776 005037 014334 CLR GOOD ;
1243 005002 032777 040000 007210 BIT #40000,@SWR ;SET SWR BIT 14 TO INHIBIT
1244 005010 001003 BNE 1$ ;ERROR PRINTOUT
1245 005012 004737 016632 JSR PC,TYPOUT ;CHARACTER ADDRESS REGISTER ERROR
1246 005016 005662 ERMS20 ;
1247 005020 004737 020544 1$: JSR PC,ERROR ;ERROR 0
1248 005024 120000 A+G+0 ;
1249 005026 104410 T2002: TRAP+10 ;SCOPE LOOP ON RESET
1250 005030 004746 T2001 ;(SWR BIT 9 )
1251
1252 005032 012703 000001 T2003: MOV #1,R3 ;SET R3 FOR FIRST DATA PATTERN
1253 005036 012702 000020 T2003A: MOV #20,R2 ;REPEAT COUNT FOR EACH PATTERN
1254 005042 004737 020246 T2004: JSR PC,GENER ;* SET UP
1255 005046 010037 014334 MOV R0,GOOD ;* DATA PATTERN
1256 005052 105037 014335 CLRB GOOD+1 ;* IN GOOD
1257 005056 113777 014334 007154 T2004A: MOVB GOOD,@CHAR ;LOAD CHARACTER ADDRESS REGISTER
1258 005064 117737 007150 014336 MOVB @CHAR,BAD ;READ IT BACK
1259 005072 105037 014337 CLRB BAD+1 ;
1260 005076 042737 000200 014334 BIC #200,GOOD ;
1261 005104 023737 014334 014336 CMP GOOD,BAD ;WAS IT LOADED CORRECTLY
1262 005112 001415 BEQ T2005 ;BRANCH IF O.K.
1263 005114 013737 014240 014344 MOV CHAR,ADDRES ;SET UP FOR ERROR REPORT
1264 005122 032777 040000 007070 BIT #40000,@SWR ;SET SWR BIT 14 TO INHIBIT
1265 005130 001003 BNE 1$ ;ERROR PRINTOUT
1266 005132 004737 016632 JSR PC,TYPOUT ;CHARACTER ADDRESS REGISTER ERROR
1267 005136 005662 ERMS20 ;
1268 005140 004737 020544 1$: JSR PC,ERROR ;ERROR 1
1269 005144 120001 A+G+1 ;
1270 005146 104420 T2005: TRAP+20 ;SCOPE LOOP ON THIS PATTERN
1271 005150 005056 T2004A ; ( SWR BIT 10 )
1272
1273 005152 104430 TRAP+30 ;SCOPE LOOP ON CHANGING PATTERN
1274 005154 005042 T2004 ; ( SWR BITS 9 & 10 )
    
```

| | | | | | |
|------|--------|--------|-----|--------|--------------------------------------|
| 1275 | | | | | |
| 1276 | 005156 | 005302 | DEG | R2 | ;HAVE WE DONE ENOUGH OF THIS PATTERN |
| 1277 | 005160 | 001330 | BNE | T2004 | ;BRANCH IF NO |
| 1278 | 005162 | 005203 | INC | R3 | ;NEXT PATTERN |
| 1279 | 005164 | 020327 | CMP | R3,#7 | ;HAVE WE DONE ALL PATTERNS. |
| 1280 | 005170 | 100722 | BMI | T2003A | ;BRANCH IF NO |
| 1281 | | | | | |


```

1283
1284
1285
1286
1287
1288
1289
1290 005172 000005          T2006: RESET          ;INITALISE HARDWARE
1291 005174 012701 000004    MOV          #4,R1      ;
1292 005200 012737 000000 014334 T2007: MOV          #0,GOOD    ;SET UP GOOD FOR ERROR REPORT
1293 005206 112777 000000 007022 MOVB         #0,@CHDR    ;LOAD SOMETHING TO CHARACTER STORE
1294                                     ;DATA REGISTER
1295 005214 117737 007020 014336 MOVB         @CHAR,BAD   ;READ CHARACTER ADDRESS REGISTER
1296 005222 105037 014337     CLRB         BAD+1      ;
1297 005226 005737 014336     TST         BAD        ;IT SHOULD READ ZERO
1298 005232 001016          BNE         T2008      ;BRANCH TO ERROR REPORT IF NOT
1299 005234 005301          DEC         R1         ;DO THIS 4 TIMES
1300 005236 001360          BNE         T2007      ;BRANCH IF MORE TO DO
1301 005240 117737 006774 014336 MOVB         @CHAR,BAD   ;READ CHACACTER ADDRESS REGISTER
1302 005246 105037 014337     CLRB         BAD+1      ;
1303 005252 022737 000001 014336 CMP          #1,BAD      ;BAD SHOULD NOW CONTAIN 1
1304 005260 001415          BEQ         T2009      ;BRANCH IF O.K.
1305 005262 012737 000001 014334 MOV          #1,GOOD    ;SET UP FOR ERROR REPORT
1306 005270 032777 040000 006722 T2008: BIT          #40000,@SWR ;SET SWR BIT 14 TO INHIBIT
1307 005276 001003          BNE         1$        ;ERROR PRINTOUT
1308 005300 004737 016632     JSR         PC,TYPOUT   ;CHARACTER STORE ADDRESS REGISTER
1309 005304 005662          ERMS20          ;INCREMENT ERROR
1310 005306 004737 020544     1$: JSR         PC,ERROR ;ERROR 2,ADDRESS SHOULD ONLY
1311 005312 120002          A+G+2          ;AFTER 8 DATA LOADS OR READS
1312 005314 104440          T2009: TRAP+40      ;SCOPE LOOP ON RESET AND LOAD
1313 005316 005172          T2006          ;(SWR BIT 11)
1314
1315
1316
1317 005320 000005          T2010: RESET          ;INITALISE HARDWARE
1318 005322 012777 001000 006672 MOV          #1000,@CSR  ;SET 6V BIT
1319 005330 012701 000003     MOV          #3,R1      ;
1320 005334 012737 000000 014334 T2011: MOV          #0,GOOD    ;SET UP GOOD FOR ERROR REPORT
1321 005342 112777 000000 006666 MOVB         #0,@CHDR    ;LOAD SOMETHING TO CHARACTER STORE
1322                                     ;DATA REGISTER
1323 005350 117737 006664. 014336 MOVB         @CHAR,BAD   ;READ CHARACTER ADDRESS REGISTER
1324 005356 105037 014337     CLRB         BAD+1      ;
1325 005362 005737 014336     TST         BAD        ;IT SHOULD READ ZERO
1326 005366 001014          BNE         T2012      ;BRANCH TO ERROR REPORT IF NOT
1327 005370 005301          DEC         R1         ;DO THIS 4 TIMES
1328 005372 001360          BNE         T2011      ;BRANCH IF MORE TO DO
1329 005374 117737 006640 014336 MOVB         @CHAR,BAD   ;READ CHARACTER ADDRESS REGISTER
1330 005402 022737 000001 014336 CMP          #1,BAD      ;BAD SHOULD NOW CONTAIN 1
1331 005410 001415          BEQ         T2013      ;BRANCH IF O.K.
1332 005412 012737 000001 014334 MOV          #1,GOOD    ;SET UP FOR ERROR REPORT
1333 005420 032777 040000 006572 T2012: BIT          #40000,@SWR ;SET SWR BIT 14 TO INHIBIT
1334 005426 001003          BNE         1$        ;ERROR PRINTOUT
1335 005430 004737 016632     JSR         PC,TYPOUT   ;CHARACTER STORE ADDRESS REGISTER
1336 005434 005662          ERMS20          ;INCREMENT ERROR
1337 005436 004737 020544     1$: JSR         PC,ERROR ;ERROR 3,ADDRESS SHOULD ONLY
1338 005442 120003          A+G+3          ;AFTER 6 DATA LOADS OR READS

```

```

*****
:THIS PART OF THE TEST CHECKS THAT
:CHAR (CHSR TOP BYTE )WILL INCREMENT
:WHEN :-
:      CHDR (CHSR LOW BYTE )IS LOADED
:OR    CHSR IS READ
*****

```



```

1339 005444 104450          T2013: TRAP+50          ; SCOPE LOOP ON RESET AND LOAD
1340 005446 005320          T2010          ; (SWR BIT 9 & 11 )
1341
1342
1343
1344
1345
1346 005450 005077 006546    T2014: CLR      @CSR          ;
1347 005454 112777 000000 006556    MOVB     #0,@CHAR          ; LOAD CHARACTER ADDRESS REGISTER
1348 005462 112777 000000 006546    MOVB     #0,@CHDR          ; DO TWO LOADS TO DATA REGISTER
1349 005470 112777 000000 006540    MOVB     #0,@CHDR          ; TO INCREMENT INVISIBLE BITS
1350 005476 112777 000000 006534    MOVB     #0,@CHAR          ; LOAD ADDRESS REGISTER AGAIN
1351
1352
1353 005504 012701 000004          MOV      #4,R1             ;
1354 005510 012737 000000 014334    T2015: MOV      #0,GOOD          ; SET UP GOOD FOR ERROR REPORT
1355 005516 112777 000000 006512    MOVB     #0,@CHDR          ; LOAD SOMETHING TO CHARACTER STORE
1356
1357 005524 117737 006510 014336    MOVB     @CHAR,BAD          ; DATA REGISTER
1358 005532 105037 014337          CLRB     BAD+1             ; READ CHARACTER ADDRESS REGISTER
1359 005536 005737 014336          TST      BAD               ;
1360 005542 001016          BNE      T2016             ; IT SHOULD READ ZERO
1361 005544 005301          DEC      R1                 ; BRANCH TO ERROR REPORT IF NOT
1362 005546 001360          BNE      T2015             ; DO THIS 4 TIMES
1363 005550 117737 006464 014336    MOVB     @CHAR,BAD          ; BRANCH IF MORE TO DO
1364 005556 105037 014337          CLRB     BAD+1             ; READ CHARACTER ADDRESS REGISTER
1365 005562 022737 000001 014336    CMP      #1,BAD            ;
1366 005570 001415          BEQ      T2017             ; BAD SHOULD NOW CONTAIN 1
1367 005572 012737 000001 014334    MOV      #1,GOOD           ; BRANCH IF O.K.
1368 005600 032777 040000 006412    T2016: BIT     #40000,@SWR          ; SET UP FOR ERROR REPORT
1369 005606 001003          BNE      1$                ; SET SWR BIT 14 TO INHIBIT
1370 005610 004737 016632          JSR      PC,TYP0UT          ; ERROR PRINTOUT
1371 005614 005662          ERMS20          ; CHARACTER STORE ADDRESS REGISTER
1372 005616 004737 020544          1$: JSR      PC,ERROR          ; INCREMENT ERROR
1373 005622 120004          T2017: TRAP+60          ; ERROR 4, ADDRESS SHOULD ONLY
1374 005624 104460          T2014          ; AFTER 8 DATA LOADS OR READS
1375 005626 005450          TRAP+4          ; SCOPE LOOP ON READ AND LOAD
1376 005630 104404          T2003          ; (SWR BIT 10 & 11 )
1377 005632 005032          T2003          ; SET SWR BIT 8 TO
1378
1379
1380 005634 005337 014250          DEC      REPCNT            ; LOOP ON TEST
1381 005640 001402          BEQ      T2018             ; DONE ENOUGH ?
1382 005642 000137 005032          JMP      T2003             ; YES
1383
1384 005646 032777 000100 006344    T2018: BIT     #100,@SWR          ; NO
1385 005654 001427          BEQ      TEST3             ; CHECK FOR PRE-SELECTED TEST
1386 005656 000137 001232          JMP      START2
1387
1388
1389
1390
1391 005662 020133 044103 051101    ERMS20: .NLIST BEX
1392
1393
1393

```



```

1395          .SBTTL TEST3
1396          ;*****
1397          ;THIS TEST CHECKS OUT THE CHARACTER
1398          ;STORE BY LOADING EVERY LOCATION
1399          ;WITH RANDOM DATA AND THEN
1400          ;READING EACH LOCATION BACK AND
1401          ;CHECKING THE DATA.
1402          ;*****
1403
1404
1405 005734 012737 000003 014216 TEST3: MOV #3,TESTNO ;SET UP TEST NO.
1406 005742 104402 TRAP+2
1407 005744 005752 T3000
1408 005746 004737 014416 JSR PC,TESTR ;OUTPUT TEST NO
1409
1410 005752 005037 014346 T3000: CLR ERRDIS
1411 005756 012704 000100 MOV #REPC2,R4
1412 005762 004737 015104 JSR PC,FASTSW
1413 005766 010437 014250 MOV R4,REPCNT ;SET UP TEST ITERATION COUNT
1414
1415 005772 005077 006224 T3001: CLR @CSR ;SET FOR 8X8 MATRIX
1416 005776 105077 006236 CLRB @CHAR ;SET CHARACTER ADDRESS TO 0
1417 006002 013705 012060 MOV BUFF,R5 ;SAVE FOR COMPARE IN BUFF
1418 006006 012701 002000 MOV #2000,R1 ;2000 (8) BYTES
1419 006012 006201 ASR R1 ;TWO PER LOAD
1420 006014 012703 000006 T3002: MOV #6,R3 ;GENERATE RANDOM NUMBER
1421 006020 004737 020246 JSR PC,GENER
1422 006024 004737 006146 JSR PC,LOPROM ;LOAD CHARACTER DATA BYTE
1423 006030 113700 014335 MOVB GOOD+1,R0 ;NEXT BYTE IN R0
1424 006034 004737 006146 JSR PC,LOPROM ;LOAD SECOND BYTE
1425 006040 005301 DEC R1 ;IS THERE MORE ?
1426 006042 003364 BGT T3002 ;BRANCH IF YES
1427 006044 013705 012060 T3003: MOV BUFF,R5 ;NO, SO NOW READ & COMPARE
1428 006050 012701 002000 MOV #2000,R1
1429 006054 105077 006160 CLRB @CHAR ;SET CHARACTER ADDRESS TO 0
1430 006060 004737 006162 T3004: JSR PC,REPROM ;READ A BYTE
1431 006064 000407 BR T3005 ;O.K. ARG
1432 006066 110037 014336 MOVB RO,BAD ;SET UP FOR ERROR MESSAGE
1433 006072 105037 014337 CLRB BAD+1 ;CLEAR OUT RUBBISH
1434 ;ADDRES CONTAINS STORE ADDRESS
1435 ;WRONG DATA IN STORE
1436 006076 004737 020544 JSR PC,ERROR ;ERROR 1,STORE DATA ERROR
1437 006102 130001 G+S+A+1
1438
1439 006104 104410 T3005: TRAP+10 ;SCPOE LOOP ON READING 1ST CHAR.
1440 006106 006044 T3003 ;(SWR BIT 9 )
1441
1442 006110 005301 DEC R1 ;MORE ?
1443 006112 003362 BGT T3004 ;YES
1444 006114 104404 TRAP+4 ;SET SWR BIT 8 TO
1445 006116 005772 T3001 ;LOOP ON TEST
1446
1447
1448 006120 005337 014250 DEC REPCNT ;DONE ENOUGH ?
1449 006124 001402 BEQ T3006 ;YES
1450 006126 000137 005772 JMP T3001 ;NO
    
```

```

1451
1452 006132 032777 000100 006060 T3006: BIT #100,@SWR ;CHECK FOR PRE-SELECTED TEST
1453 006140 001447 BEQ TEST4
1454 006142 000137 001232 JMP START2
1455
1456
1457
1458 006146 042700 177400 LOPROM: BIC #177400,R0 ;CLEAR TOP RUBBISH
1459 006152 110025 MOV R0,(R5)+ ;STORE IN CORE
1460 006154 110077 006056 MOV R0,@CHDR ;& IN DISPLAY CHARACTER STORE
1461 006160 000207 RTS PC ;RETURN
1462
1463 006162 017702 006042 REPRM: MOV @CHSR,R2 ;READ CHARACTER STORE REGISTER
1464 006166 010200 MOV R2,R0 ;*SET UP R0 WITH
1465 006170 042700 177400 BIC #177400,R0 ;*CHARACTER DATA
1466 006174 000302 SWAB R2 ;*SET UP
1467 006176 042702 177400 BIC #177400,R2 ;*ADDRESS WITH
1468 006202 010237 014344 MOV R2,ADDRES ;*CHARACTER ADDRESS
1469 006206 010537 014342 MOV R5,STATUS
1470 006212 163737 012060 014342 SUB BUFF,STATUS
1471 006220 162737 000001 014342 SUB #1,STATUS
1472 006226 042737 177770 014342 BIC #177770,STATUS ;ROW OF CHARACTER
1473 006234 111537 014334 MOVB (R5),GOOD ;SET UP WITH GOOD DATA
1474 006240 042737 177400 014334 BIC #177400,GOOD
1475 006246 120025 CMPB R0,(R5)+ ;CHECK WHAT WAS READ
1476 006250 001402 BEQ REPRM1 ;BRANCH IF O.K.
1477 006252 062716 000002 ADD #2,(SP) ;NO SO JUMP OVER OK RETURN ADD.
1478 006256 000207 REPRM1: RTS PC ;RETURN
1479
1480
    
```



```

1482
1483
1484
1485
1486
1487
1488
1489
1490
1491 006260 012737 000004 014216 TEST4: MOV #4,TESTNO ;SET UP TEST NO.
1492 006266 104402 TRAP+2
1493 006270 006276 T4000
1494 006272 004737 014416 JSR PC,TESTR ;OUTPUT TEST NO
1495
1496 006276 005037 014346 T4000: CLR ERRDIS
1497 006302 012704 000100 MOV #REPC2,R4
1498 006306 004737 015104 JSR PC,FASTSW
1499 006312 010437 014250 MOV R4,REPCNT ;SET UP TEST ITERATION COUNT
1500
1501 006316 000005 T4001: RESET ;INITALISE HARDWARE
1502 006320 012702 000100 T4002A: MOV #100,R2 ;SET REPEAT COUNT
1503 006324 012703 000006 MOV #6,R3 ;*GENERATE RANDOM
1504 006330 004737 020246 T4002: JSR PC,GENER ;*DATA
1505 006334 010077 005666 MOV RO,@CAR ;LOAD CAR
1506 006340 017737 005662 014336 MOV @CAR,BAD ;READ IT BACK
1507 006346 042700 140200 BIC #140200,RO ;CLEAR BITS THAT READ ZERO
1508 006352 020037 014336 CMP RO,BAD ;WAS IT LOADED & READ CORRECTLY
1509 006356 001410 BEQ T4003 ;BRANCH IF O.K.
1510 006360 010037 014334 MOV RO,GOOD ;*SET UP FOR
1511 006364 013737 014226 014344 MOV CAR,ADDRES ;*ERROR REPORT
1512 006372 004737 020544 JSR PC,ERROR ;ERROR 1, CURSOR ADDRESS REGISTER
1513 006376 120001 A+G+1 ;DATA ERROR
1514 006400 104410 T4003: TRAP+10 ;LOOP ON LOAD
1515 006402 006330 T4002 ;(SWR BIT 9 )
1516
1517 006404 012777 000335 005612 MOV #335,@DBUF ;DO HOME CURSOR
1518 006412 017737 005610 014336 MOV @CAR,BAD ;READ CURSOR
1519 006420 005737 014336 TST BAD ;WAS IT CLEARED BY HOME
1520 006424 001410 BEQ T4004 ;BRANCH IF O.K.
1521 006426 005037 014334 CLR GOOD ;SET UP FOR ERROR REPORT
1522 006432 013737 014226 014344 MOV CAR,ADDRES ;
1523 006440 004737 020544 JSR PC,ERROR ;ERROR 2, CAR DID NOT CLEAR
1524 006444 120002 A+G+2 ;ON HOME
1525 006446 104420 T4004: TRAP+20 ;LOOP ON LOAD
1526 006450 006330 T4002 ;(SWR BIT 10 )
1527
1528 006452 005302 DEC R2 ;ANY MORE ?
1529 006454 001325 BNE T4002 ;YES SO BRANCH
    
```

```

1531
1532
1533
1534 006456 013737 014232 014344      MOV    CARX,ADDRES      ;SET UP FOR ANY ERROR REPORT
1535 006464 012777 000404 005530      MOV    #404,@CSR       ;SET FOR 6H & INC
1536 006472 012701 000117                MOV    #79.,R1         ;SET R1 WITH MAX VALUE
1537 006476 110177 005530                T4005: MOVB   R1,@CARX    ;LOAD X ADDRESS
1538 006502 117737 005524 014336      MOVB   @CARX,BAD      ;READ IT BACK
1539 006510 105037 014337                CLRB   BAD+1          ;
1540 006514 010137 014334                MOV    R1,GOOD        ;SET UP GOOD
1541 006520 023737 014334 014336      CMP    GOOD,BAD       ;WAS DATA READ CORRECTLY
1542 006526 001403                BEQ    T4006          ;BRANCH IF O.K.
1543 006530 004737 020544                JSR    PC,ERROR       ;ERROR 3, LOAD/READ ERROR
1544 006534 120003                A+G+3                ;
1545
1546 006536 112777 000000 005460  T4006: MOVB   #0,@DBUF     ;DO A LOAD TO PICTURE STORE
1547 006544 117737 005462 014336      MOVB   @CARX,BAD      ;READ X ADDRESS
1548 006552 105037 014337                CLRB   BAD+1          ;
1549 006556 023727 014334 000117      CMP    GOOD,#79.     ;SHOULD IT HAVE INCREMENTED
1550 006564 001402                BEQ    1$             ;BRANCH IF NO
1551 006566 005237 014334                INC    GOOD           ;YES, SO INC GOOD
1552 006572 023737 014334 014336  1$:  CMP    GOOD,BAD     ;DID CARX INCREMENT CORRECTLY
1553 006600 001403                BEQ    T4007          ;BRANCH IF O.K.
1554 006602 004737 020544                JSR    PC,ERROR       ;ERROR 4, CARX INCREMENT ERROR
1555 006606 120004                A+G+4                ;
1556
1557 006610 112777 000300 005406  T4007: MOVB   #300,@DBUF   ;SET BLINK OFF INCREMENT
1558 006616 117737 005410 014336      MOVB   @CARX,BAD      ;READ X ADDRESS
1559 006624 105037 014337                CLRB   BAD+1          ;
1560 006630 023727 014334 000117      CMP    GOOD,#79.     ;SHOULD IT HAVE INCREMENTED
1561 006636 001402                BEQ    1$             ;BRANCH IF NO
1562 006640 005237 014334                INC    GOOD           ;YES, SO INCREMENT GOOD
1563 006644 023737 014334 014336  1$:  CMP    GOOD,BAD     ;DID CARX INCREMENT CORRECTLY
1564 006652 001403                BEQ    T4008          ;BRANCH IF O.K.
1565 006654 004737 020544                JSR    PC,ERROR       ;ERROR 5, CARX INCREMENT ERROR
1566 006660 120005                A+G+5                ;
1567
1568 006662 112777 000301 005334  T4008: MOVB   #301,@DBUF   ;SET BLINK ON INCREMENT
1569 006670 117737 005336 014336      MOVB   @CARX,BAD      ;READ X ADDRESS
1570 006676 105037 014337                CLRB   BAD+1          ;
1571 006702 023727 014334 000117      CMP    GOOD,#79.     ;SHOULD IT HAVE INCREMENTED
1572 006710 001402                BEQ    1$             ;BRANCH IF NO
1573 006712 005237 014334                INC    GOOD           ;YES, SO INCREMENT GOOD
1574 006716 023737 014334 014336  1$:  CMP    GOOD,BAD     ;DID CARX INCREMENT CORRECTLY
1575 006724 001403                BEQ    T4009          ;BRANCH IF O.K.
1576 006726 004737 020544                JSR    PC,ERROR       ;ERROR 6, CARX INCREMENT ERROR
1577 006732 120006                A+G+6                ;
1578
1579 006734 112777 000315 005262  T4009: MOVB   #315,@DBUF   ;DO A CARAGE RETURN
1580 006742 117737 005264 014336      MOVB   @CARX,BAD      ;READ X ADDRESS
1581 006750 105037 014337                CLRB   BAD+1          ;
1582 006754 005037 014334                CLR    GOOD           ;
1583 006760 023737 014334 014336      CMP    GOOD,BAD      ;DID CARX ZERO ON CR.
1584 006766 001403                BEQ    T4010          ;BRANCH IF O.K.
1585 006770 004737 020544                JSR    PC,ERROR       ;ERROR 7,CARX DID NOT ZERO
1586 006774 120007                A+G+7                ;
    
```



```

1587
1588 006776 104440          T4010: TRAP+40          ;SCOPE LOOP ON LOAD AND INC.
1589 007000 006476          T4005          ;(SWR BIT 11 )
1590 007002 005301          DEC          R1          ;HAVE WE DONE ENOUGH
1591 007004 100234          BPL          T4005      ;BRANCH IF MORE
1592
1593
1594
1595
1596 007006 013737 014232 014344          MOV          CARX,ADDRES ;SET UP FOR ANY ERROR REPORT
1597 007014 012777 000004 005200          MOV          #4,@CSR     ;SET FOR 8H & INC
1598 007022 012701 000077          MOV          #63.,R1    ;SET R1 WITH MAX VALUE
1599 007026 110177 005200          T4015: MOVB       R1,@CARX ;LOAD X ADDRESS
1600 007032 117737 005174 014336          MOVB       @CARX,BAD   ;READ IT BACK
1601 007040 105037 014337          CLRB       BAD+1      ;
1602 007044 010137 014334          MOV          R1,GOOD   ;SET UP GOOD
1603 007050 023737 014334 014336          CMP          GOOD,BAD  ;WAS DATA READ CORRECTLY
1604 007056 001403          BEQ          T4016     ;BRANCH IF O.K.
1605 007060 004737 020544          JSR          PC,ERROR  ;ERROR 3, LOAD/READ ERROR
1606 007064 120003          ;
1607
1608 007066 112777 000000 005130 T4016: MOVB       #0,@DBUF    ;DO A LOAD TO PICTURE STORE
1609 007074 117737 005132 014336          MOVB       @CARX,BAD  ;READ X ADDRESS
1610 007102 105037 014337          CLRB       BAD+1      ;
1611 007106 023727 014334 000077          CMP          GOOD,#63. ;SHOULD IT HAVE INCREMENTED
1612 007114 001402          BEQ          1$        ;BRANCH IF NO
1613 007116 005237 014334          INC          GOOD      ;YES, SO INC GOOD
1614 007122 023737 014334 014336 1$: CMP          GOOD,BAD  ;DID CARX INCREMENT CORRECTLY
1615 007130 001403          BEQ          T4017     ;BRANCH IF O.K.
1616 007132 004737 020544          JSR          PC,ERROR  ;ERROR 4, CARX INCREMENT ERROR
1617 007136 120004          ;
1618
1619 007140 112777 000300 005056 T4017: MOVB       #300,@DBUF  ;SET BLINK OFF INCREMENT
1620 007146 117737 005060 014336          MOVB       @CARX,BAD  ;READ X ADDRESS
1621 007154 105037 014337          CLRB       BAD+1      ;
1622 007160 023727 014334 000077          CMP          GOOD,#63. ;SHOULD IT HAVE INCREMENTED
1623 007166 001402          BEQ          1$        ;BRANCH IF NO
1624 007170 005237 014334          INC          GOOD      ;YES, SO INCREMENT GOOD
1625 007174 023737 014334 014336 1$: CMP          GOOD,BAD  ;DID CARX INCREMENT CORRECTLY
1626 007202 001403          BEQ          T4018     ;BRANCH IF O.K.
1627 007204 004737 020544          JSR          PC,ERROR  ;ERROR 5, CARX INCREMENT ERROR
1628 007210 120005          ;
1629
1630 007212 112777 000301 005004 T4018: MOVB       #301,@DBUF  ;SET BLINK ON INCREMENT
1631 007220 117737 005006 014336          MOVB       @CARX,BAD  ;READ X ADDRESS
1632 007226 105037 014337          CLRB       BAD+1      ;
1633 007232 023727 014334 000077          CMP          GOOD,#63. ;SHOULD IT HAVE INCREMENTED
1634 007240 001402          BEQ          1$        ;BRANCH IF NO
1635 007242 005237 014334          INC          GOOD      ;YES, SO INCREMENT GOOD
1636 007246 023737 014334 014336 1$: CMP          GOOD,BAD  ;DID CARX INCREMENT CORRECTLY
1637 007254 001403          BEQ          T4019     ;BRANCH IF O.K.
1638 007256 004737 020544          JSR          PC,ERROR  ;ERROR 6, CARX INCREMENT ERROR
1639 007262 120006          ;
1640
1641 007264 112777 000315 004732 T4019: MOVB       #315,@DBUF  ;DO A CARAGE RETURN
1642 007272 117737 004734 014336          MOVB       @CARX,BAD  ;READ X ADDRESS
    
```


| | | | | | | | | |
|------|--------|--------|--------|--------|--------|---------|-------------|-------------------------------|
| 1643 | 007300 | 105037 | 014337 | | | CLRB | BAD+1 | : |
| 1644 | 007304 | 005037 | 014334 | | | CLR | GOOD | : |
| 1645 | 007310 | 023737 | 014334 | 014336 | | CMP | GOOD,BAD | : DID CARX ZERO ON CS. |
| 1646 | 007316 | 001403 | | | | BEQ | T4020 | : BRANCH IF O.K. |
| 1647 | 007320 | 004737 | 020544 | | | JSR | PC,ERROR | : ERROR 7,CARX DID NOT ZERO |
| 1648 | 007324 | 120007 | | | | | A+G+7 | : |
| 1649 | | | | | | | | : |
| 1650 | 007326 | 104440 | | | T4020: | TRAP+40 | | : SCOPE LOOP ON LOAD AND INC. |
| 1651 | 007330 | 007026 | | | | T4015 | | : (SWR BIT 11) |
| 1652 | 007332 | 005301 | | | | DEC | R1 | : HAVE WE DONE ENOUGH |
| 1653 | 007334 | 100234 | | | | BPL | T4015 | : BRANCH IF MORE |
| 1654 | 007336 | 013737 | 014234 | 014344 | | MOV | CARY,ADDRES | : SET UP FOR ANY ERROR REPORT |
| 1655 | 007344 | 012777 | 001004 | 004650 | | MOV | #1004,@CSR | : SET 6V & INC BITS |
| 1656 | 007352 | 013701 | 012160 | | | MOV | HALF6V,R1 | : *SET UP |
| 1657 | 007356 | 006301 | | | | ASL | R1 | : *R1 WITH |
| 1658 | 007360 | 005301 | | | | DEC | R1 | : *MAX VALUE OF Y |
| 1659 | 007362 | 010102 | | | | MOV | R1,R2 | : SAVE IN R2 |
| 1660 | | | | | | | | : |
| 1661 | 007364 | 110177 | 004644 | | T4021: | MOVB | R1,@CARY | : LOAD Y ADDRESS |
| 1662 | 007370 | 117737 | 004640 | 014336 | | MOVB | @CARY,BAD | : READ IT BACK |
| 1663 | 007376 | 105037 | 014337 | | | CLRB | BAD+1 | : |
| 1664 | 007402 | 010137 | 014334 | | | MOV | R1,GOOD | : SET UP GOOD |
| 1665 | 007406 | 023737 | 014334 | 014336 | | CMP | GOOD,BAD | : WAS LOAD/READ CORRECT |
| 1666 | 007414 | 001403 | | | | BEQ | 1\$ | : BRANCH IF O.K. |
| 1667 | 007416 | 004737 | 020544 | | | JSR | PC,ERROR | : ERROR 8, LOAD/READ ERROR |
| 1668 | 007422 | 120010 | | | | | A+G+10 | : |
| 1669 | 007424 | 112777 | 000312 | 004572 | 1\$: | MOVB | #312,@DBUF | : DO A LINE FEED |
| 1670 | 007432 | 117737 | 004576 | 014336 | | MOVB | @CARY,BAD | : READ Y |
| 1671 | 007440 | 105037 | 014337 | | | CLRB | BAD+1 | : |
| 1672 | 007444 | 023702 | 014334 | | | CMP | GOOD,R2 | : *SET UP |
| 1673 | 007450 | 001402 | | | | BEQ | 2\$ | : *GOOD WITH |
| 1674 | 007452 | 005237 | 014334 | | | INC | GOOD | : *EXPECTED DATA |
| 1675 | 007456 | 023737 | 014334 | 014336 | 2\$: | CMP | GOOD,BAD | : DID IT INCREMENT CORRECTLY |
| 1676 | 007464 | 001403 | | | | BEQ | T4022 | : BRANCH IF O.K. |
| 1677 | 007466 | 004737 | 020544 | | | JSR | PC,ERROR | : ERROR 9,INC ERROR ON LF |
| 1678 | 007472 | 120011 | | | | | A+G+11 | : |
| 1679 | 007474 | 104460 | | | T4022: | TRAP+60 | | : SCOPE LOOP ON LOAD & INC |
| 1680 | 007476 | 007364 | | | | T4021 | | : (SWR BITS 10 & 11) |
| 1681 | 007500 | 005301 | | | | DEC | R1 | : ANY MORE TO DO |
| 1682 | 007502 | 100330 | | | | BPL | T4021 | : BRANCH IF YES |
| 1683 | | | | | | | | : |
| 1684 | 007504 | 012777 | 000004 | 004510 | | MOV | #4,@CSR | : SET INC BITS |
| 1685 | 007512 | 013701 | 012162 | | | MOV | HALF8V,R1 | : *SET UP |
| 1686 | 007516 | 006301 | | | | ASL | R1 | : *R1 WITH |
| 1687 | 007520 | 005301 | | | | DEC | R1 | : *MAX VALUE OF Y |
| 1688 | 007522 | 010102 | | | | MOV | R1,R2 | : SAVE IN R2 |
| 1689 | | | | | | | | : |
| 1690 | 007524 | 110177 | 004504 | | T4023: | MOVB | R1,@CARY | : LOAD Y ADDRESS |
| 1691 | 007530 | 117737 | 004500 | 014336 | | MOVB | @CARY,BAD | : READ IT BACK |
| 1692 | 007536 | 105037 | 014337 | | | CLRB | BAD+1 | : |
| 1693 | 007542 | 010137 | 014334 | | | MOV | R1,GOOD | : SET UP GOOD |
| 1694 | 007546 | 023737 | 014334 | 014336 | | CMP | GOOD,BAD | : WAS LOAD/READ CORRECT |
| 1695 | 007554 | 001403 | | | | BEQ | 1\$ | : BRANCH IF O.K. |
| 1696 | 007556 | 004737 | 020544 | | | JSR | PC,ERROR | : ERROR 8, LOAD/READ ERROR |
| 1697 | 007562 | 120010 | | | | | A+G+10 | : |
| 1698 | 007564 | 112777 | 000312 | 004432 | 1\$: | MOVB | #312,@DBUF | : DO A LINE FEED |

| | | | | | | | | |
|------|--------|--------|--------|--------|--------|---------|-----------|-----------------------------|
| 1699 | 007572 | 117737 | 004436 | 014336 | | MOVB | @CARY,BAD | :READ Y |
| 1700 | 007600 | 105037 | 014337 | | | CLRB | BAD+1 | : |
| 1701 | 007604 | 023702 | 014334 | | | CMP | GOOD,R2 | :*SET UP |
| 1702 | 007610 | 001402 | | | | BEQ | 2\$ | :*GOOD WITH |
| 1703 | 007612 | 005237 | 014334 | | | INC | GOOD | :*EXPECTED DATA |
| 1704 | 007616 | 023737 | 014334 | 014336 | 2\$: | CMP | GOOD,BAD | :DID IT INCREMENT CORRECTLY |
| 1705 | 007624 | 001403 | | | | BEQ | T4024 | :BRANCH IF O.K. |
| 1706 | 007626 | 004737 | 020544 | | | JSR | PC,ERROR | :ERROR 9,INC ERROR ON LF |
| 1707 | 007632 | 120011 | | | | | A+G+11 | : |
| 1708 | 007634 | 104460 | | | T4024: | TRAP+60 | | :SCOPE LOOP ON LOAD & INC |
| 1709 | 007636 | 007524 | | | | T4023 | | :(SWR BITS 10 & 11) |
| 1710 | 007640 | 005301 | | | | DEC | R1 | :ANY MORE TO DO |
| 1711 | 007642 | 100330 | | | | BPL | T4023 | :BRANCH IF YES |
| 1712 | | | | | | | | : |
| 1713 | 007644 | 104404 | | | | TRAP+4 | | :SET SWR BIT 8 TO |
| 1714 | 007646 | 006320 | | | | T4002A | | :LOOP ON TEST |
| 1715 | 007650 | 005337 | 014250 | | | DEC | REPCNT | :DONE ENOUGH ? |
| 1716 | 007654 | 001402 | | | | BEQ | T40WW | :YES |
| 1717 | 007656 | 000137 | 006320 | | | JMP | T4002A | :NO |
| 1718 | | | | | | | | : |
| 1719 | 007662 | 032777 | 000100 | 004330 | T40WW: | BIT | #100,@SWR | :CHECK FOR PRESELECTED TEST |
| 1720 | 007670 | 001402 | | | | BEQ | TEST5 | : |
| 1721 | 007672 | 000137 | 001232 | | | JMP | START2 | : |

```

1723          .SBTTL TEST5
1724          :*****
1725          :THIS TEST CHECKS EVERY LOCATION
1726          :IN THE PICTURE STORE FOR CORRECT
1727          :LOADING AND READING. IT ALSO
1728          :CHECKS PRESET AND THE OPERATION
1729          :OF THE VARIOUS BLINK COMMANDS.
1730          :*****
1731
1732 007676 012737 000005 014216 TEST5: MOV #5,TESTNO ;SET UP TEST NO.
1733 007704 104402 TRAP+2
1734 007706 007714 T5000
1735 007710 004737 014416 JSR PC,TESTR ;OUTPUT TEST NO
1736
1737 007714 005037 014346 T5000: CLR ERRDIS
1738 007720 012704 000100 MOV #REPC2,R4
1739 007724 004737 015104 JSR PC,FASTSW
1740 007730 010437 014250 MOV R4,REPCNT ;SET UP TEST ITERATION COUNT
1741
1742 007734 112777 000302 004262 MOVB #302,@DBUF ;CLEAR BLINK CONTROL
1743 007742 005037 012056 CLR BLNFLAG ;CLEAR BLINK FLAG
1744 007746 005077 004254 T5001: CLR @CAR ;CLEAR PICTURE STORE ADDRESS
1745 007752 012777 001405 004242 MOV #1405,@CSR ;SET 6V,6H & CURSOR INC BITS
1746 007760 013705 012060 MOV BUFF,R5 ;SAVE FOR COMPARE IN BUFF
1747 007764 013704 012160 MOV HALF6V,R4 ;SET UP R4 WITH HALF NO OF ROWS
1748 007770 012702 000120 T5002: MOV #80.,R2 ;SET UP R2 WITH CHAR PER LINE
1749 007774 012703 000006 T5003: MOV #6,R3 ;*GENERATE RANDOM
1750 010000 004737 020246 JSR PC,GENER ;*DATA PATTERNS
1751 010004 042737 100200 014334 BIC #100200,GOOD ;*
1752 010012 013700 014334 MOV GOOD,R0 ;*
1753
1754 010016 010025 MOV R0,(R5)+ ;SAVE IN BUFFER
1755 010020 004737 012002 JSR PC,SETBLN ;CHECK BLINK FLAG
1756 010024 042700 000100 BIC #100,R0 ;*MAKE CONTENTS OF R0 INTO
1757 010030 052700 000200 BIS #200,R0 ;*COLOUR COMAND
1758 010034 110077 004164 MOVB R0,@DBUF ;LOAD DATA BYTE
1759 010040 000300 SWAB R0 ;GET NEXT BYTE
1760 010042 110077 004156 MOVB R0,@DBUF ;NOW LOAD CHARACTER
1761 010046 005302 DEC R2 ;ANY MORE CHARACTERS IN LINE
1762 010050 001351 BNE T5003 ;BRANCH IF YES
1763 010052 112777 000315 004144 MOVB #315,@DBUF ;DO CARAGE RETURN
1764 010060 112777 000312 004136 MOVB #312,@DBUF ;DO LINE FEED
1765 010066 005304 DEC R4 ;ANY MORE ROWS IN HALF SCREEN
1766 010070 001337 BNE T5002 ;BRANCH IF YES
1767 010072 013705 012060 MOV BUFF,R5 ;NO, SO SET UP FOR NEXT HALF
1768 010076 013704 012160 MOV HALF6V,R4 ;NUMBER OF ROWS
1769 010102 012702 000120 T5004: MOV #80.,R2 ;NUMBER OF CHARACTER$ PER LINE
1770 010106 012500 T5005: MOV (R5)+,R0 ;GET FIRST TWO BYTES
1771 010110 000300 SWAB R0 ;
1772 010112 004737 012002 JSR PC,SETBLN ;CHECK BLINK FLAG
1773 010116 042700 000100 BIC #100,R0 ;*MAKE CONTENTS OF R0 INTO
1774 010122 052700 000200 BIS #200,R0 ;*COLOUR COMAND
1775 010126 110077 004072 MOVB R0,@DBUF ;LOAD COLOUR
1776 010132 000300 SWAB R0 ;GET NEXT BYTE
1777 010134 110077 004064 MOVB R0,@DBUF ;LOAD CHARACTER
1778 010140 005302 DEC R2 ;ANY MORE CHARACTERS IN LINE

```


| | | | | | | | | | |
|------|--------|--------|--------|--------|---------|---------|-------------|----------------|------------------------------------|
| 1779 | 010142 | 001361 | | | | BNE | T5005 | | :BRANCH IF YES |
| 1780 | 010144 | 112777 | 000315 | 004052 | | MOVB | #315,@DBUF | | :DO CARAGE RETURN |
| 1781 | 010152 | 112777 | 000312 | 004044 | | MOVB | #312,@DBUF | | :DO LINE FEED |
| 1782 | 010160 | 005304 | | | | DEC | R4 | | :ANY MORE ROWS |
| 1783 | 010162 | 001347 | | | | BNE | T5004 | | :BRANCH IF YES |
| 1784 | | | | | | | | | :***** |
| 1785 | | | | | | | | | :WE HAVE NOW FINISHED LOADING |
| 1786 | | | | | | | | | :***** |
| 1787 | | | | | | | | | :***** |
| 1788 | | | | | | | | | :NOW START READING TO CHECK STORE |
| 1789 | | | | | | | | | :***** |
| 1790 | 010164 | 112777 | 000335 | 004032 | T5006: | MOVB | #335,@DBUF | | :HOME CURSOR |
| 1791 | 010172 | 042777 | 000004 | 004022 | | BIC | #4,@CSR | | :CLEAR INC BIT |
| 1792 | 010200 | 013705 | 012060 | | | MOV | BUFF,R5 | | :SET UP BUFFER FOR COMPARE |
| 1793 | 010204 | 013704 | 012160 | | | MOV | HALF6V,R4 | | :SET UP NO OF ROWS IN HALF SCREEN |
| 1794 | 010210 | 012702 | 000120 | | T5007: | MOV | #80.,R2 | | :SET CHARACTERS PER ROW |
| 1795 | 010214 | 012500 | | | T5007A: | MOV | (R5)+,R0 | | :*SET UP R0 WITH EXPECTED DATA |
| 1796 | 010216 | 052777 | 000020 | 003776 | T5008: | BIS | #20,@CSR | | :SET READ STORE BIT |
| 1797 | 010224 | 000240 | | | | NOP | | | : |
| 1798 | 010226 | 000240 | | | | NOP | | | : |
| 1799 | 010230 | 000240 | | | | NOP | | | :WAIT |
| 1800 | 010232 | 000240 | | | | NOP | | | : |
| 1801 | 010234 | 000240 | | | | NOP | | | : |
| 1802 | 010236 | 017737 | 003762 | 014336 | | MOV | @DBUF,BAD | | :READ PICTURE STORE DATA |
| 1803 | 010244 | 020037 | 014336 | | | CMP | R0,BAD | | :IS IT CORRECT ? |
| 1804 | 010250 | 001410 | | | | BEQ | T5009 | | :YES, SO BRANCH |
| 1805 | 010252 | 010037 | 014334 | | | MOV | R0,GOOD | | :NO, SO SET UP FOR ERROR REPORT |
| 1806 | 010256 | 017737 | 003744 | 014344 | | MOV | @CAR,ADDRES | | : |
| 1807 | 010264 | 004737 | 020544 | | | JSR | PC,ERROR | | :ERROR 51,PICTURE STORE DATA ERROR |
| 1808 | 010270 | 120001 | | | | | A+G+1 | | : |
| 1809 | | | | | | | | | : |
| 1810 | 010272 | 104410 | | | T5009: | TRAP+10 | | | :SCOPE LOOP READING THIS ADDRESS |
| 1811 | 010274 | 010216 | | | | T5008 | | | :(SWR BIT 9) |
| 1812 | | | | | | | | | : |
| 1813 | 010276 | 105277 | 003730 | | | INCB | @CARX | | :INCREMENT X ADDRESS |
| 1814 | 010302 | 005302 | | | | DEC | R2 | | :HAVE WE READ ALL OF ROW |
| 1815 | 010304 | 001343 | | | | BNE | T5007A | | :NO, SO READ SOME MORE |
| 1816 | 010306 | 105077 | 003720 | | | CLRB | @CARX | | :YES, SO SET UP ADDRESS OF |
| 1817 | 010312 | 105277 | 003716 | | | INCB | @CARY | | :NEXT ROW |
| 1818 | 010316 | 005304 | | | | DEC | R4 | | :ANY MORE ROWS |
| 1819 | 010320 | 001333 | | | | BNE | T5007 | | :YES, SO BRANCH |
| 1820 | | | | | | | | | : |
| 1821 | | | | | | | | | : |
| 1822 | 010322 | 013705 | 012060 | | | MOV | BUFF,R5 | :SET UP BUFFER | : |
| 1823 | 010326 | 013704 | 012160 | | | MOV | HALF6V,R4 | | : |
| 1824 | 010332 | 012702 | 000120 | | T5010: | MOV | #80.,R2 | | : |
| 1825 | 010336 | 012500 | | | T5011: | MOV | (R5)+,R0 | | :GET EXPECTED DATA |
| 1826 | 010340 | 000300 | | | | SWAB | R0 | | : |
| 1827 | 010342 | 052777 | 000020 | 003652 | T5012: | BIS | #20,@CSR | | :SET READ STORE BIT |
| 1828 | 010350 | 017737 | 003650 | 014336 | | MOV | @DBUF,BAD | | :READ PICTURE STORE DATA |
| 1829 | 010356 | 020037 | 014336 | | | CMP | R0,BAD | | :IS IT CORRECT |
| 1830 | 010362 | 001410 | | | | BEQ | T5013 | | :YES, SO BRANCH |
| 1831 | 010364 | 010037 | 014334 | | | MOV | R0,GOOD | | :NO, SO SET UP FOR ERROR REPORT |
| 1832 | 010370 | 017737 | 003632 | 014344 | | MOV | @CAR,ADDRES | | : |
| 1833 | 010376 | 004737 | 020544 | | | JSR | PC,ERROR | | :ERROR 1, PICTURE STORE DATA ERROR |
| 1834 | 010402 | 120001 | | | | | A+G+1 | | : |

1835 010404 104420
1836 010406 010342
1837 010410 005277 003616
1838 010414 005302
1839 010416 001347
1840 010420 105077 003606
1841 010424 105277 003604
1842 010430 005304
1843 010432 001337
1844
1845

T5013: TRAP+20
T5012
INC @CARX
DEC R2
BNE T5011
CLRB @CARX
INCB @CARY
DEC R4
BNE T5010

;LOOP ON THIS ADDRESS
; (SWR BIT 10)
;NEXT ADDRESS
;ANY MORE IN ROW
;YES , SO DO SOME MORE
;NO, SO SET UP NEXT ADDRESS
;
;ANY MORE ROWS ?
;YES , SO DO SOME MORE

| | | | | | | | |
|------|--------|--------|--------|--------|----------------|--------------|---------------------------------|
| 1903 | 010764 | 013737 | 011756 | 014334 | MOV | PRESET,GOOD | :LOAD GOOD WITH EXPECTED DATA |
| 1904 | 010772 | 013704 | 012160 | | MOV | HALF6V,R4 | :*SET UP |
| 1905 | 010776 | 006304 | | | ASL | R4 | :*NUMBER OF ROWS |
| 1906 | 011000 | 012702 | 000120 | | T5018A: MOV | #80.,R2 | :NUMBER OF CHARACTERS PER ROW |
| 1907 | 011004 | 052777 | 000020 | 003210 | T5018B: BIS | #20,@CSR | :SET READ PICTURE STORE BIT |
| 1908 | 011012 | 017737 | 003206 | 014336 | MOV | @DBUF,BAD | :READ DATA BUFFER |
| 1909 | 011020 | 023737 | 014334 | 014336 | CMP | GOOD,BAD | :CHECK DATA |
| 1910 | 011026 | 001406 | | | BEQ | T5019 | :BRANCH IF O.K. |
| 1911 | 011030 | 017737 | 003172 | 014344 | MOV | @CAR,ADDRES | :SET UP FOR ERROR REPORT |
| 1912 | 011036 | 004737 | 020544 | | JSR | PC,ERROR | :PICTURE STORE DATA ERROR |
| 1913 | 011042 | 120005 | | | | A+G+5 | :AFTER PRESET |
| 1914 | 011044 | 104470 | | | T5019: TRAP+70 | | : |
| 1915 | 011046 | 011000 | | | T5018A | | : |
| 1916 | 011050 | 105277 | 003156 | | INCB | @CARX | :INCREMENT X ADDRESS |
| 1917 | 011054 | 005302 | | | DEC | R2 | :WAS IT MAX |
| 1918 | 011056 | 001352 | | | BNE | T5018B | :BRANCH IF NO. |
| 1919 | 011060 | 105077 | 003146 | | CLRB | @CARX | :YES, SO SET X=0 |
| 1920 | 011064 | 105277 | 003144 | | INCB | @CARY | :INCREMENT Y ADDRESS |
| 1921 | 011070 | 005304 | | | DEC | R4 | :WAS IT MAX |
| 1922 | 011072 | 001342 | | | BNE | T5018A | :BRANCH IF NO |
| 1923 | | | | | | | :ALL CHECKED. |
| 1924 | | | | | | | :***** |
| 1925 | | | | | | | :THE NEXT PART OF THIS TESTS |
| 1926 | | | | | | | :LOADING PART OF THE PICTURE |
| 1927 | | | | | | | :STORE WITH RANDOM DATA |
| 1928 | | | | | | | :WHEN BLINK IS REQUIRED THE |
| 1929 | | | | | | | :BLINK ON OR OFF WITH INCREMENT |
| 1930 | | | | | | | :COMMAND IS USED. THE CONTENTS |
| 1931 | | | | | | | :OF THE PICTURE STORE ARE THEN |
| 1932 | | | | | | | :READ BACK TO CHECK THAT IT |
| 1933 | | | | | | | :LOADED CORRECTLY. |
| 1934 | 011074 | 005037 | 012056 | | CLR | BLNFLG | :CLEAR BLINK FLAG |
| 1935 | 011100 | 012777 | 000302 | 003116 | MOV | #302,@DBUF | :CLEAR BLINK CONTROL |
| 1936 | 011106 | 005077 | 003114 | | T5020: CLR | @CAR | :CLEAR CURSOR ADDRESS |
| 1937 | 011112 | 052777 | 000004 | 003102 | BIS | #4,@CSR | :SET INCREMENT BIT |
| 1938 | 011120 | 113777 | 012160 | 003106 | MOVB | HALF6V,@CARY | :*SET UP |
| 1939 | 011126 | 106277 | 003102 | | ASRB | @CARY | :*Y ADDRESS |
| 1940 | 011132 | 013705 | 012060 | | MOV | BUFF,R5 | :SAVE FOR COMPARE IN BUFF |
| 1941 | 011136 | 013704 | 012160 | | MOV | HALF6V,R4 | :SET UP R4 WITH HALF ROWS |


```

1943 011142 012702 000120      T5020A: MOV      #80.,R2      ;CHARACTERS PER LINE
1944 011146 012703 000006      MOV      #6,R3        ;*GENERATE
1945 011152 004737 020246      T5020B: JSR      PC,GENER ;*RANDOM
1946 011156 042737 100200      BIC      #100200,GOOD ;*DATA
1947 011164 013700 014334      MOV      GOOD,R0      ;*PATTERNS
1948 011170 004737 011634      JSR      PC,STBINC    ;SUBROUTINE TO SET BLINK INC
1949 011174 000137 011250      JMP      T5021        ;EXIT IF END
1950 011200 010025      MOV      R0,(R5)+     ;SAVE IN BUFFER
1951 011202 042700 000100      BIC      #100,R0      ;*MAKE CONTENTS OF R0
1952 011206 052700 000200      BIS      #200,R0      ;*INTO COLOUR COMMAND
1953 011212 110077 003006      MOVVB   R0,@DBUF     ;LOAD COLOUR
1954 011216 000300      SWAB    R0           ;GET NEXT BYTE
1955 011220 110077 003000      MOVVB   R0,@DBUF     ;NOW LOAD CHARACTER
1956 011224 005302      DEC     R2           ;ANY MORE CHARACTERS IN LINE
1957 011226 001351      BNE     T5020B       ;BRANCH IF YES
1958 011230 112777 000315      MOVVB   #315,@DBUF   ;DO CR
1959 011236 112777 000312      MOVVB   #312,@DBUF   ;DO LF
1960 011244 005304      DEC     R4           ;ANY MORE ROWS
1961 011246 001335      BNE     T5020A       ;BRANCH IF YES
1962
1963      ;*****
1964      ;DONE THE LOADING
1965      ;*****
1966      ;NOW START CHECKING
1967      ;*****
1967 011250 005077 002752      T5021: CLR      @CAR      ;SET ADDRESS TO ZERO
1968 011254 042777 000004      BIC     #4,@CSR      ;CLEAR INC BIT
1969 011262 013737 011756      MOV     PRESET,GOOD  ;EXPECTED DATA
1970 011270 013700 012160      MOV     HALF6V,R0    ;
1971 011274 006200      ASR     R0           ;NUMBER OF ROWS
1972 011276 012701 000120      T5022: MOV     #80.,R1  ;CHARACTERS PER ROW
1973 011302 052777 000020      T5023: BIS     #20,@CSR  ;READ STORE
1974 011310 017737 002710      MOV     @DBUF,BAD    ;
1975 011316 023737 014334      CMP     GOOD,BAD     ;CHECK DATA
1976 011324 001406      BEQ     T5024        ;BRANCH IF O.K.
1977 011326 017737 002674      MOV     @CAR,ADDRES  ;SET UP FOR ERROR REPORT
1978 011334 004737 020544      JSR     PC,ERROR     ;ERROR 6, STORE DATA ERROR
1979 011340 120006      A+G+6      ;
1980
1981 011342 005277 002664      T5024: INC     @CARX     ;INCREMENT X ADDRESS
1982 011346 005301      DEC     R1           ;ANY MORE CHARACTERS IN ROW
1983 011350 001354      BNE     T5023        ;BRANCH IF YES
1984 011352 105077 002654      CLRB   @CARX        ;SET X TO ZERO
1985 011356 105277 002652      INCB   @CARY        ;INC Y ADDRESS
1986 011362 005300      DEC     R0           ;ANY MORE ROWS
1987 011364 001344      BNE     T5022        ;BRANCH IF YES
1988
1989 011366 013705 012060      MOV     BUFF,R5      ;START OF BUFFER
1990 011372 013700 012160      MOV     HALF6V,R0    ;NUMBER OF ROWS
1991 011376 012701 000120      T5025: MOV     #80.,R1  ;CHERACTERS PER ROW
1992 011402 012537 014334      T5026: MOV     (R5)+,GOOD ;GET EXPECTED DATA
1993 011406 052777 000020      BIS     #20,@CSR    ;READ STORE
1994 011414 017737 002604      MOV     @DBUF,BAD    ;
1995 011422 023737 014334      CMP     GOOD,BAD     ;CHECK DATA
1996 011430 001406      BEQ     T5027        ;BRANCH IF O.K.
1997 011432 017737 002570      MOV     @CAR,ADDRES  ;SET UP FOR ERROR REPORT
1998 011440 004737 020544      JSR     PC,ERROR     ;STORE DATA ERROR
    
```

```

1999 011444 120006          G+A+6          ;
2000
2001 011446 105277 002560    T5027:  INCB   @CARX      ;NEXT X ADDRESS
2002 011452 005301          DEC     R1           ;IS IT END OF ROW
2003 011454 001352          BNE    T5026        ;BRANCH IF MORE
2004 011456 105077 002550    CLRB   @CARX      ;SET X TO ZERO
2005 011462 105277 002546    INCB   @CARY      ;INCREMENT Y ADDRESS
2006 011466 005300          DEC     R0           ;HAVE WE FINISHED
2007 011470 001342          BNE    T5025        ;BRANCH IF NO
2008
2009 011472 013737 011756 014334    MOV    PRESET,GOOD ;EXPECTED DATA
2010 011500 013700 012160    MOV    HALF6V,R0   ;
2011 011504 006200          ASR    R0           ;SET NUMBER OF ROWS
2012 011506 012701 000120    T5028: MOV    #80.,R1 ;CHARACTERS PER ROW
2013 011512 052777 000020 002502  T5029: BIS    #20,@CSR ;READ STORE
2014 011520 017737 002500 014336    MOV    @DBUF,BAD   ;
2015 011526 023737 014334 014336    CMP    GOOD,BAD    ;CHECK DATA
2016 011534 001406          BEQ    T5030        ;BRANCH IF O.K.
2017 011536 017737 002464 014344    MOV    @CAR,ADDRES ;SET UP FOR ERROR REPORT
2018 011544 004737 020544    JSR    PC,ERROR    ;STORE DATA ERROR
2019 011550 120006          G+A+6          ;
2020 011552 005277 002454    T5030: INC    @CARX   ;INCREMENT X ADDRESS
2021 011556 005301          DEC    R1           ;HAVE WE FINISHED ROW
2022 011560 001354          BNE    T5029        ;BRANCH IF MORE
2023 011562 105077 002444    CLRB   @CARX      ;ZERO X ADDRESS
2024 011566 105277 002442    INCB   @CARY      ;INCREMENT Y ADDRESS
2025 011572 005300          DEC    R0           ;ANY MORE ROWS
2026 011574 001344          BNE    T5028        ;BRANCH IF YES
2027
2028 011576 104404          TRAP+4          ;SET SWR BIT 8 TO
2029 011600 007746          T5001          ;LOOP ON TEST
2030 011602 005337 014250    DEC    REPCNT      ;DONE ENOUGH ?
2031 011606 001402          BEQ    T5014        ;YES
2032 011610 000137 007746    JMP    T5001        ;NO
2033
2034 011614 032777 000100 002376  T5014: BIT    #100,@SWR ;CHECK FOR PRE-SELECTED TEST
2035 011622 001402          BEQ    T50WY        ;
2036 011624 000137 001232    JMP    START2      ;
2037
2038
2039 011630 000137 014664          T50WY: JMP    ENDIT    ;SIGNAL END OF PASS
2040
2041 011634 032700 000100    STBINC: BIT    #100,R0 ;SHOULD BLINK BE ON
2042 011640 001013          BNE    BINC1        ;BRANCH IF YES
2043 011642 005737 012056    TST    BLNFLG      ;CHECK BLINK FLAG
2044 011646 001440          BEQ    BINC2        ;BRANCH IF CLEAR
2045 011650 112777 000300 002346    MOVB   #300,@DBUF  ;CLEAR BLINC INC COMMAND
2046 011656 013725 011756    MOV    PRESET,(R5)+ ;SAVE PICTURE CONTENTS
2047 011662 005037 012056    CLR    BLNFLG      ;CLEAR FLAG
2048 011666 000413          BR     BINC3        ;
2049 011670 005737 012056    BINC1: TST    BLNFLG ;CHECK BLINK FLAG
2050 011674 001025          BNE    BINC2        ;BRANCH IF SET
2051 011676 112777 000301 002320    MOVB   #301,@DBUF  ;SET BLINK INC COMMAND
2052 011704 013725 011756    MOV    PRESET,(R5)+ ;SAVE PICTURE CONTENTS
2053 011710 012737 000001 012056    MOV    #1,BLNFLG  ;SET FLAG
2054 011716 005302          BINC3: DEC    R2    ;
    
```



```

2055 011720 001013          BNE      BINC2          ;BRANCH IF MORE CHARACTERS
2056 011722 112777 000315 002274      MOVB     #315,@DBUF     ;DO CR
2057 011730 112777 000312 002266      MOVB     #312,@DBUF     ;DO LF
2058 011736 012702 000120          MOV      #80.,R2       ;NUMBER OF CHARACTERS IN ROW
2059 011742 005304          DEC      R4             ;ANY MORE ROWS
2060 011744 001001          BNE      BINC2          ;BRANCH IF YES
2061 011746 000207          RTS      PC             ;END RETURN
2062 011750 062716 000004      BINC2:  ADD     #4,(SP)  ;CHANGE RETURN ADDRESS
2063 011754 000207          RTS      PC             ;MORE RETURN
2064
2065 011756 000000          PRESET: 0
2066
2067 011760 005237 004272      RDYINT: INC     INTFLG   ;SET FLAG
2068 011764 017737 002232 014342      MOV      @CSR,STATUS   ;SAVE CSR
2069 011772 042777 000100 002222      BIC      #100,@CSR     ;CLEAR ENABLE
2070 012000 000002          RTI
2071
2072
2073
2074
2075
2076 012002 032700 000100      SETBLN: BIT     #100,R0 ;DO WE WANT BLINK ON
2077 012006 001011          BNE      BLN1          ;BRANCH IF YES
2078 012010 005737 012056          TST     BLNFLG        ;NO, SO IS FLAG OFF
2079 012014 001417          BEQ     BLN2          ;BRANCH IF FLAG IS OFF
2080 012016 012777 000302 002200      MOV      #302,@DBUF   ;NO, SO CLEAR BLINK CONTROL
2081 012024 005037 012056          CLR     BLNFLG        ;AND CLEAR FLAG
2082 012030 000411          BR      BLN2          ;ALL DONE
2083 012032 005737 012056      BLN1:  TST     BLNFLG   ;WE WANT BLINK ON
2084 012036 001006          BNE      BLN2          ;BLINK ALREADY ON SO O.K.
2085 012040 012777 000303 002156      MOV      #303,@DBUF   ;BLINK WAS NOT ON SO SET IT
2086 012046 012737 000001 012056      MOV      #1,BLNFLG    ;SET FLAG
2087 012054 000207          BLN2:  RTS      PC
2088
2089
2090 012056 000000          BLNFLG: 0
2091 012060 021104          BUFF:  BUFF1
    
```

```

2093          .SBTTL  UTILITY ROUTINES
2094
2095
2096 012062 004737 016632      SET56: JSR      PC,TYPOUT      ;ASK 525 625
2097 012066 012164              ASK56
2098 012070 004737 016104      JSR      PC,READ        ;READ
2099 012074 020027 000065      CMP      RO,#'5         ;WAS IT 5
2100 012100 001012              BNE      SET56A         ;BRANCH IF NO
2101 012102 012737 177777 012156  MOV      #177777,L525   ;YES, SO SET FLAG
2102 012110 012737 000017 012162  MOV      #15.,HALF8V    ;
2103 012116 012737 000024 012160  MOV      #20.,HALF6V    ;
2104 012124 000207              RTS      PC              ;DONE
2105 012126 020027 000066      SET56A: CMP      RO,#'6   ;WAS IT 6
2106 012132 001353              BNE      SET56          ;BRANCH IF NO AND ASK AGAIN
2107 012134 005037 012156      CLR      L525           ;CLEAR FLAG
2108 012140 012737 000022 012162  MOV      #18.,HALF8V    ;
2109 012146 012737 000030 012160  MOV      #24.,HALF6V    ;
2110 012154 000207              RTS      PC              ;DONE
2111
2112 012156 000000      L525:  0
2113 012160 000030      HALF6V: 24.
2114 012162 000022      HALF8V: 18.
2115
2116
2117
2118
2119          .SBTTL  ASCII STRINGS
2120
2121          .NLIST  BEX
2122
2123 012164 020133 051511 052040  ASK56: .ASCII  /[ IS THE CONTROLLER JUMPERED FOR 525 OR 626/
2124 012237          040 044514 042516  .ASCII  / LINES,[ TYPE 5 OR 6.....@/
2125 012272 053133 053124 030063  GOMSG: .ASCII  ![VTV30-H/J OR VT30-H DIAGNOSTIC AND!
2126 012335          040 054105 051105  .ASCII  / EXERCISER/
2127 012347          133 040520 052122  .ASCII  /[PART1-----LOGIC TESTS/
2128 012375          133 052123 051101  .ASCII  /[START ADDRESS IS 1000, OR 200/
2129 012433          133 042522 052123  .ASCII  /[RESTART ADDRESS IS 1200/
2130 012463          133 047531 020125  .ASCII  /[YOU ARE STRONGLY ADVISED TO READ THE/
2131 012530 047533 052120 047511  .ASCII  /[OPTION DESCRIPTION BEFORE PROCEEDING,/
2132 012576 040533 054516 043040  .ASCII  /[ANY FURTHER./
2133 012613          133 047506 020122  .ASCII  /[FOR DETAILS OF BUS AND VECTOR ADDRESSES/
2134 012663          133 046120 040505  .ASCII  /[PLEASE REFER TO THE OPTION DESCRIPTION/
2135 012732 055533 040133          .ASCII  /[[[ @/
2136 012736 051533 046105 041505  WMSG:  .ASCII  /[SELECT DESIRED SWITCH REGISTER OPTIONS[ @/
2137 013007          133 042524 052123  TESMSG: .ASCII  /[TEST NO @/
2138 013020 042533 042116 047440  PASMSG: .ASCII  /[END OF PASS @/
2139 013036 052133 040522 020120  ILVMSG: .ASCII  /[TRAP TO ILLEGAL VECTOR @/
2140 013070 043133 047522 020115  FRMMSG: .ASCII  /[FROM ADDRESS @/
2141 013110 043133 051117 052040  FIRVMS: .ASCII  /[FOR THE FIRST VECTOR GROUP @/
2142 013144 043133 051117 052040  NXTVMS: .ASCII  /[FOR THE NEXT VECTOR GROUP @/
2143 013177          133 047516 020116  NOMEMA: .ASCII  /[NON EXISTANT MEMORY HAS NOT BEEN FOUND @/
2144
2145 013247          133 054524 042520  REDMES: .ASCII  /[TYPE CNTRL-C TO CONTINUE @/
2146 013302 051533 051127 036440  SWRMSG: .ASCII  /[SWR = @/
2147 013314 022133 020040          100  MODADM: .ASCII  /[$ @/
2148 013321          040 020057          100  MODSPA: .ASCII  ? / @?
  
```


| | | | | | | | |
|------|--------|--------|--------|--------|---------|--------|--------------------------------------|
| 2149 | 013325 | 133 | 020052 | 040040 | MODPRM: | .ASCII | /[* @/ |
| 2150 | 013332 | 043133 | 051111 | 052123 | BAMSG: | .ASCII | /[FIRST BUS ADDRESS IS@/ |
| 2151 | 013366 | 043133 | 051111 | 052123 | VAMSG: | .ASCII | /[FIRST VECTOR ADDRESS IS@/ |
| 2152 | 013425 | 133 | 047111 | 040526 | ODAMSG: | .ASCII | /[INVALID ADDRESS @/ |
| 2153 | 013450 | 040533 | 042104 | 042522 | OVAMSG: | .ASCII | /[ADDRESS EXCEEDS 772 @/ |
| 2154 | 013477 | 133 | 044506 | 051522 | PRMSG: | .ASCII | /[FIRST PRIORITY LEVEL IS@/ |
| 2155 | 013536 | 047133 | 054105 | 020124 | NPRMSG: | .ASCII | /[NEXT PRIORITY LEVEL IS@/ |
| 2156 | 013575 | 133 | 047516 | 020116 | NXMSG: | .ASCII | /[NON EXISTANT ADDRESS @/ |
| 2157 | 013625 | 133 | 047111 | 040526 | BADPRI: | .ASCII | /[INVALID PRIORITY, PLEASE RE-ENTER/ |
| 2158 | 013667 | 133 | 044124 | 020105 | | .ASCII | /[THE PRIORITY.....@/ |
| 2159 | 013712 | 042133 | 043105 | 052501 | NODEFM: | .ASCII | /[DEFAULT SETTINGS ARE NOT ALLOWED/ |
| 2160 | 013753 | 133 | 046120 | 040505 | | .ASCII | /[PLEASE RE-ENTER THE VALUE@/ |
| 2161 | 014014 | 042533 | 040043 | | EMSG1: | .ASCII | /[E#@/ |
| 2162 | 014020 | 020040 | 052101 | 050040 | EMSG2: | .ASCII | / AT PC @/ |
| 2163 | 014032 | 043533 | 047517 | 020104 | EMSG3: | .ASCII | /[GOOD : @/ |
| 2164 | 014043 | 040 | 020040 | 040502 | EMSG4: | .ASCII | / BAD :@/ |
| 2165 | 014054 | 042133 | 052101 | 020101 | EMSG5: | .ASCII | /[DATA : @/ |
| 2166 | 014065 | 040 | 020040 | 042101 | EMSG6: | .ASCII | / ADDRESS :@/ |
| 2167 | 014103 | 133 | 052123 | 052101 | EMSG7: | .ASCII | /[STATUS :@/ |
| 2168 | 014116 | 041533 | 046101 | 042514 | EMSG8: | .ASCII | /[CALLED FROM :@/ |
| 2169 | 014135 | 040 | 020040 | 051105 | EMSG9: | .ASCII | / ERROR COUNT = @/ |
| 2170 | | 014160 | | | | .EVEN | |
| 2171 | 014160 | 000000 | 000000 | 000000 | OCTNUM: | .WORD | 0,0,0 |
| 2172 | 014166 | 100 | 000 | | | .BYTE | 100,0 |
| 2173 | 014170 | 040502 | 042523 | 030061 | BASE11: | .ASCII | /BASE10@/ |
| 2174 | | 014200 | | | | .EVEN | |
| 2175 | 014200 | 000000 | 000000 | 000000 | DECMSG: | .WORD | 0,0,0 |
| 2176 | 014206 | 000000 | 000000 | 000000 | OCTMSG: | .WORD | 0,0,0,0 |
| 2177 | | | | | | .EVEN | |
| 2178 | | | | | | .LIST | BEX |

```
2180 .SBTTL PROGRAM VARIABLES
2181
2182 014216 000000 TESTNO: 0
2183 014220 177570 SWR: HSWR
2184 014222 164000 CSR: 164000
2185 014224 164002 DBUF: 164002
2186 014226 164004 CAR: 164004
2187 014230 164006 CHSR: 164006
2188 014232 164004 CARX: 164004
2189 014234 164005 CARY: 164005
2190 014236 164006 CHDR: 164006
2191 014240 164007 CHAR: 164007
2192 014242 000170 INTVEC: 170
2193 014244 000004 VECLEV: 4
2194 014246 000000 SSWR: 0
2195 014250 000000 REPCNT: 0
2196 014252 000137 001000 JM600: JMP @#START
2197 014256 000001 FSTCNT: 1
2198 014260 000000 TRPARG: 0
2199 014262 000000 TRPSEL: 0
2200 014264 000000 TRPMEM: 0
2201 014266 000000 SAVPC: 0
2202 014270 000000 SAVPC1: 0
2203 014272 000000 SAVSTA: 0
2204 014274 000000 TYPOTA: 0
2205 014276 000000 RAND: 0
2206 014300 000000 TYPD1: 0
2207 014302 000000 MODADR: 0
2208 014304 000000 MODSAV: 0
2209 014306 000000 MASK: 0
2210 014310 000000 BASADD: 0
2211 014312 000000 TRPERR: 0
2212 014314 000000 CHWORD: 0
2213 014316 000000 ERR: 0
2214 014320 000000 PARITY: 0
2215 014322 000000 BCCHAR: 0
2216 014324 052525 RANDN: 52525
2217 014326 000001 RANSEL: 1
2218 014330 000006 RANMTA: 6
2219 014332 052525 RANST: 52525
2220 014334 000000 GOOD: 0
2221 014336 000000 BAD: 0
2222 014340 000000 DATA: 0
2223 014342 000000 STATUS: 0
2224 014344 000000 ADDRES: 0
2225 014346 000000 ERRDIS: 0
2226 014350 000000 ERRARG: 0
2227 014352 000000 CALLPC: 0
2228 014354 000000 TRXVAD: 0
2229 014356 000000 TRXPAR: 0
2230 014360 000000 TRXEXM: 0
2231 014362 000000 NXMADR: 0
2232 014364 000000 SAVEXM: 0
2233 014366 000000 SAVPAR: 0
2234 014370 000000 SAVLT4: 0
2235 014372 000000 SAVLT6: 0
```


| | | | |
|------|--------|--------|-----------|
| 2236 | 014374 | 000000 | CNVFLG: 0 |
| 2237 | 014376 | 000000 | STRADD: 0 |
| 2238 | 014400 | 000000 | STRLEN: 0 |
| 2239 | 014402 | 000000 | LOWCHR: 0 |
| 2240 | 014404 | 000000 | UPPCHR: 0 |
| 2241 | 014406 | 000000 | RUBFLG: 0 |
| 2242 | 014410 | 000000 | RANDC: 0 |
| 2243 | 014412 | 000000 | FSAVPW: 0 |
| 2244 | 014414 | 000000 | BCOUNT: 0 |

2246
2247
2248
2249
2250
2251f
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300

014416 004737 016632
014422 013007
014424 013700 014216
014430 004737 016722
014434 004737 016576
014440 000207

```

.SBTTL PRINT TEST NUMBER

DESCRIPTION:
    ROUTINE TO PRINT THE TEST NUMBER IN OCTAL

CALLING SEQUENCE:
    JSR    PC,TESTR

INPUT PARAMETERS:
    TESTNO CONTAINS THE OCTAL TEST NUMBER TO BE
    PRINTED

IMPLICIT INPUT PARAMETERS:
    THE LABEL TESMSG IS THE START ADDRESS OF AN
    ASCII STRING 'TEST NO'

OUTPUT PARAMETERS:
    R0 WILL BE CORRUPTED

IMPLICIT OUTPUT PARAMETERS:
    THE MESSAGE 'TEST NO N' WILL BE PRINTED ON THE
    CONSOLE TERMINAL

COMPLETION CODES:
    NONE

POSSIBLE ERROR CODES:
    NONE

TESTR:  JSR    PC,TYPEOUT
        TESMSG ;TEST NO
        MOV    TESTNO,R0
        JSR    PC,PROCT ;PRINT OCTAL TEST NO
        JSR    PC,CRLF
        RTS    PC ;EXIT

```


2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354

.SBTTL 'SILLSI' SUBROUTINE

DESCRIPTION:

ROUTINE TO ESTABLISH WHETHER OR NOT THE
DIAGNOSTIC IS RUNNING ON A PROCESSOR
WHICH POSSESSES ONLY ONE INTERRUPT BUS
PRIORITY LEVEL.

CALLING SEQUENCE:

JSR PC,SILLSI

INPUT PARAMTERS:

NONE

IMPLICIT INPUT PARAMETERS:

NCNE

OUTPUT PARAMETERS:

THE VARIABLE 'LSIFLG' WILL BE SET UP TO
REFLECT WHETHER OR NOT THE PROCESSOR HAS
A SINGLE INTERRUPT PRIORITY LEVEL.

LSIFLG = 0 => MULTIPLE INT. PRIORITIES
<> 0 => SINGLE INT. PRIORITY

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

2355 014442 005037 014504
2356 014446 012737 014472 000004
2357 014454 012737 000340 000006

SILLSI: CLR LSIFLG ;SET UP FLAG FOR NON-LSI
MOV #18,4 ;INSTALL TRAP THRU 4 VECTOR
MOV #340,6 ;AND CORRESPONDING PRIORITY

```
2358 ;  
2359 014462 005737 177776 ; TST PSW ;TRY ADDRESSING THE PSW --  
2360 014466 000240 ; NOP ;IF NOT THERE, TRAP THRU 4  
2361 014470 000404 ; BR 2$ ;WILL OCCUR, ELSE BRANCH.  
2362 ;  
2363 014472 022626 1$: CMP (SP)+,(SP)+ ;PERFORM A DUMMY RTI  
2364 014474 012737 177777 014504 ;MOV #-1,LSIFLG ;PROCESSOR HAS ONE INT.LEVEL  
2365 ;  
2366 014502 000207 2$: RTS PC ;RETURN TO MAINLINE, CALL.  
2367 ;  
2368 ;  
2369 ;  
2370 014504 000000 LSIFLG: 0 ;0 => MULTIPLE INT.PRIORITIES  
2371 ; ;-1=> SINGLE INT. PRIORITY  
2372 ;  
2373 ;  
2374 ;
```


2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429

.SBTTL NON EXISTANT SWR TRAP

DESCRIPTION:

THE TRAP WHEN TESTING FOR THE HARDWARE
SWITCH REGISTER WILL OCCUR HERE
THE LOACTION SWR WILL BE SET TO CONTAIN THE
ADDRESS OF THE SOFTWARE SWITCH REGISTER SSWR

ENTRY POINT

SWRSET

INPUT PARAMETERS:

OCCURS IF A HARDWARE SWITCH REGISTER IN NOT
PRESENT

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

THE LOCATION SWR WILL BE SET TO CONTAIN SSWR

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

AN RTI IS PERFORMED

POSSIBLE ERROR CODES:

NONE

014506 012737 014246 014220 SWRSET: MOV #SSWR,SWR
014514 000002 RTI

2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486

.SBTTL SET UP ILLEGAL VECTOR TRAPS

DESCRIPTION:

ROUTINE TO SET CATCHES FOR TRAPS TO ILLEGAL VECTORS IN THE RANGE 0 TO 772, DURING THE RUNNING OF THE TESTS.

THE CATCH IS TO FORCE THE EXECUTION OF AN IOT TRAP.

THE VECTOR 14 (ODT VECTOR) IS LEFT FREE, 34 (TRAP VECTOR) IS SET TO THE ADDRESS TRAPSV TO SERVICE THE TRAP INSTRUCTION. THE VECTOR 20 (IOT) IS SET TO ILLVEC TO SERVICE ILLEGAL VECTOR TRAPS, AND THE ADDRESSES 200 AND 202 ARE SET WITH A JUMP TO START, THUS ALLOWING THE BE BE RESTARTED FROM ADDRESS 200. LOCATIONS 30 AND 32 AND SET TO CATCH EMT CALLS AND HENCE READ THE PROCESSOR PRIORITY.

LOCATION 40 IS LEFT FREE TO CONTAIN THE LOAD MEDIUM INDICATORS, LOCATION IS LEFT FREE TO CONTAIN THE XXDP RETURN ADDRESS (IF PRESENT). LOCATION 46 IS SET TO CONTAIN A POINTER TO THE XXDP RETURN ADDRESS AND LOCATION 52 IS SET TO ZERO.

CALLING SEQUENCE:

JSR PC,VECTOR

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

R0 AND R1 WILL BE CORRUPTED

ADDRESSES 0 THRU TO 774 WILL BE SET WITH APPROPRIATE VALUES

IMPLICIT OUTPUT PARAMETERS:

NONE


```

2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501 014516 005000          VECTOR: CLR      R0          ;FILL 0-572 WITH IOT TRAPS
2502 014520 012701 000002    MOV      #2,R1
2503 014524 020027 000014    FILL:  CMP      R0,#14      ;ODT TRAP?
2504 014530 001002          BNE      1$
2505 014532 022020          CMP      (R0)+,(R0)+      ;YES BUMP R0
2506 014534 000410          BR       FILL1
2507 014536 020027 000040    1$:  CMP      R0,#40        ;XXDP RETURN ADDRESS
2508 014542 001002          BNE      2$
2509 014544 022020          CMP      (R0)+,(R0)+      ;YES BUMP R0
2510 014546 000403          BR       FILL1
2511 014550 010120          2$:  MOV      R1,(R0)+        ;'.+2'
2512 014552 012720 000004    MOV      #4,(R0)+        ;'IOT'
2513 014556 062701 000004    FILL1: ADD      #4,R1
2514 014562 020027 000774    CMP      R0,#774
2515 014566 002756          BLT     FILL
2516 014570 012737 015124 000034 FILL2: MOV      #TRAPSV,34      ;TRAP (LOOP CONTROL)
2517 014576 012737 000340 000036    MOV      #340,36
2518 014604 012737 014754 000030    MOV      #FADR,30        ; PLUG EMT FOR READING
2519 014612 012737 000340 000032    MOV      #340,32        ; THE PROCESSOR STATUS
2520 014620 012737 015024 000020    MOV      #ILLVEC,20     ;PLUG 20 FOR IOTS
2521 014626 012737 000340 000022    MOV      #340,22
2522 014634 013737 014252 000200    MOV      JM600,200       ;SET UP JMP START IN LOC 200
2523 014642 013737 014254 000202    MOV      JM600+2,202
2524 014650 012737 014740 000046    MOV      #$ENDAD,46     ;POINT TO RETURN TO XXDP
2525 014656 005037 000052    CLR     52              ;CLEAR 52
2526 014662 000207          RTS     PC              ;THEN EXIT
  
```

2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583

.SBTTL XXDP END OF PASS HOOKS

DESCRIPTION:

ROUTINE TO SIGNIFY END OF PASS, AND IF THE PROGRAM HAS BEEN LOADED USING AN XXDP MONITOR A CALL WILL BE MADE BACK TO THE MONITOR. THE LOCATIONS USED BY XXDP ARE 40, AND 41 FOR THE LOAD MEDIUM AND 42, 43 FOR THE RETURN ADDRESS.
IF A PRESELECTED TEST IS IN OPERATION THE PROGRAM WILL GO AND SELECT THAT TEST.

ENTRY POINT:

ENDIT

INPUT PARAMETERS:

LOCATION 42/43 CONTAINS THE XXDP RETURN ADDRESS

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

IF LOCATIONS 42/43 ARE NON ZERO THEY ARE ASSUMED TO CONTAIN THE XXDP MONITOR RETURN ADDRESS

POSSIBLE ERROR CODES:

NONE


```

2584
2585
2586 014664 012706 001000      ENDIT:  MOV    #START,SP      ; RESET THE STACK
2587 014670                                PSWSET  #340                ; AND PROCESSOR PRIORITY
2588 014704 032777 000100 177306  BIT    #100,@SWR          ; IS THERE A PRESELECT ON
2589 014712 001402                                BEQ    1$                  ; NO
2590 014714 000137 001232      JMP    START2              ; YES GO SELECT THE TEST
2591
2592                                ; NO PRESELECT ON SO SIGNAL END OF PASS
2593
2594 014720 004737 015632      1$:   JSR    PC,TYPOUT        ; END OF PASS
2595 014724 013020                                PASMMSG
2596 014726 013700 000042      MOV    @#42,R0            ; GET RETURN ADDRESS TO XXDP
2597 014732 001002                                BNE    $ENDAD              ; IF IT IS ZERO THERE IS NO
2598 014734 000137 001232      JMP    START2              ; MONITOR SO RESTART DIAG
2599
2600
2601
2602 014740 004710      $ENDAD: JSR    PC,(R0)        ; CALLED VIA XXDP SO RETURN
2603 014742 000240                                NOP                          ; THERE
2604 014744 000240                                NOP
2605 014746 000240                                NOP
2606 014750 000137 001232      JMP    START2              ; ALLOW A SLIGHT PAUSE
2607                                ; THEN RESTART THE DIAGNOSTIC
2608
2609

```

2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651

.SBTTL READ PROCESSOR PRIORITY

DESCRIPTION:

THIS IS THE EMT HANDLER AND IS USED TO READ
THE PROCESSOR PRIORITY OFF THE STACK AND RETURNING
IT IN FSAVPW

CALLING SEQUENCE:

CALLED BY ISSUING AN EMT

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

THE LOCATIONS 30 AND 32 MUST HAVE BEEN SET
UP TO POINT TO THIS ROUTINE

OUTPUT PARAMETERS:

THE CONTENTS OF THE PROCESSOR PRIORITY
ARE RETURNED IN THE LOCATION FSAVPW

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

014754 016637 000002 014412 FADR: MOV 2(SP),FSAVPW ; READ PRIORITY
014762 000002 RTI ; RETURN TO CALLER

;

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
2705
2706
2707
2708

.SBTTL RING TTY BELL

DESCRIPTION:

ROUTINE TO RING THE BELL ON THE CONSOLE
 TERMINAL, IF BIT 5 IS SET IN THE SWITCH
 REGISTER.

CALLING SEQUENCE:

JSR PC,BELL

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

THE SWITCH REGISTER MUST HAVE BEEN
 SET UP

OUTPUT PARAMETERS:

THE TELETYPE BELL WILL BE RUNG IF
 APPROPRIATE

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

| | | | | | | | |
|--------|--------|--------|--------|--------|------|-----------|-------------------------|
| 014764 | 032777 | 000040 | 177226 | BELL: | BIT | #40,@SWR | |
| 014772 | 001401 | | | | BEQ | BELL1 | |
| 014774 | 000207 | | | | RTS | PC | |
| 014776 | 004737 | 015074 | | BELL1: | JSR | PC,TPREDY | ;WAIT FOR PRINTER READY |
| 015002 | 112737 | 000007 | 177566 | | MOVB | #7,TPB | |
| 015010 | 004737 | 015074 | | | JSR | PC,TPREDY | ;WAIT FOR PRINTER READY |

| | | | | | | | |
|------|--------|--------|--------|--------|------|--------|-------------|
| 2709 | 015014 | 112737 | 000000 | 177566 | MOVB | #0,TPB | ;PRINT NULL |
| 2710 | 015022 | 000207 | | | RTS | PC | ;GO OUT |

2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
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

.SBTTL ILLEGAL VECTOR TRAP CATCH

DESCRIPTION:

TRAPS TO ILLEGAL VECTORS WILL BE REPORTED HERE. THE VECTOR TO WHICH THE TRAP OCCURRED WILL BE PRINTED AS WELL AS THE ADDRESS IN THE MAIN LINE CODE FROM WHICH THE TRAP OCCURRED. A PROGRAM RESTART IS THEN PERFORMED, UNLESS A NEW TEST HAS BEEN SELECTED WHILE RUNNING UNDER A SOFTWARE SWITCH REGISTER.

ENTRY POINT

ILLVEC

INPUT PARAMETERS:

ENTRY IS CAUSED BY AN ILLEGAL VECTOR TRAP

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

THE VECTOR AND MAINLINE ADDRESS WILL BE PRINTED

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

A PROGRAM RESTART OR A NEW TEST SELECTION .

POSSIBLE ERROR CODES:

NONE

2768
2769
2770
2771
2772
2773
2774 015024 004737 016632
2775 015030 013036
2776 015032 012600
2777 015034 162700 000004
2778 015040 004737 016722
2779 015044 004737 016632
2780 015050 013070
2781 015052 005726
2782 015054 012600
2783 015056 004737 016722
2784 015062 005726
2785 015064 004737 015344
2786 015070 000137 001200

```
;  
:  
: A TRAP TO AN UNRECOGNISED VECTOR WILL  
: BE REPORTED FROM HERE.  
:  
ILLVEC: JSR PC, TYP0UT  
ILVMSG  
MOV (SP)+, R0  
SUB #4, R0  
JSR PC, PROCT ;PRINT VECTOR  
JSR PC, TYP0UT  
FRMMSG ;PRINT MAINLINE ADDRES  
TST (SP)+  
MOV (SP)+, R0  
JSR PC, PROCT  
TST (SP)+  
JSR PC, MONIT  
JMP RSTART
```


2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837

.SBTTL WAIT FOR PRINTER READY

DESCRIPTION:

ROUTINE TO WAIT UNTIL THE PRINTER
ON THE CONSOLE TERMINAL IS READY,
IE: IT IS READY TO PRINT THE NEXT
CHARACTER.

CALLING SEQUENCE:

JSR PC,TPREDY

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

015074 105737 177564
015100 100375
015102 000207

TPREDY: TSTB TPS ;ROUTINE TO WAIT FOR PRINTER READY
BPL TPREDY
RTS PC

2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894

.SBTTL SET ITERATION COUNT

DESCRIPTION:

-- ROUTINE TO SET UP THE TEST ITERATION
COUNT. A PROPOSED COUNT IS SET IN R4
THEN UNLESS BIT 13 IN THE SWITCH REGISTER
IS SET, THE SAME VALUE IS RETURNED.
IF BIT 13 IS SET THEN FAST ITERATION IS
ASSUMED AND A VALUE OF 1 IS RETURNED IN
R4.

CALLING SEQUENCE:

JSR PC,FASTSW

INPUT PARAMETERS:

R4 CONTAINS THE PROPOSED ITERATION
COUNT

IMPLICIT INPUT PARAMETERS:

SETTING BIT 13 IN THE SWR WILL INDICATE
FAST ITERATION, AND A SINGLE PASS
WILL BE REQUESTED

OUTPUT PARAMETERS:

R4 WILL CONTAIN THE ACTUAL ITERATION
COUNT

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

015104 032777 020000 177106 FASTSW: BIT #20000,@SWR

| | | | | | |
|------|--------|--------|--------|----------|-----------|
| 2895 | 015112 | 001001 | | BNE | 1\$ |
| 2896 | 015114 | 000207 | | RTS | PC |
| 2897 | 015116 | 013704 | 014256 | 1\$: MOV | FSTCNT,R4 |
| 2898 | 015122 | 000207 | | RTS | PC |

2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955

.SBTTL TRAP SERVICE ROUTINE

DESCRIPTION:

TRAP HANDLING ROUTINE. THE TRAP HANDLER IS ENTERED UPON THE EXECUTION OF ANY TRAP INSTRUCTION. IT COMPARES THE LOWER BYTE OF THE TRAP INSTRUCTION WITH THE CONTENTS ON THE SWITCH REGISTER, AND IF A MATCH IS FOUND TAKES THE CONTENTS OF THE ADDRESS FOLLOWING THE TRAP INSTRUCTION AS THE RETURN ADDRESS.

THE EXPECTED FORMAT IS:

TRAP+N
ADDR

WHERE ADDR IS THE ADDRESS TO PROCEED TO IF N MATCHES THE SWITCH REGISTER SETTINGS. THE TRAP ARGUMENT IS RELATED TO THE SWITCH REGISTER SETTINGS THUS:

| TRAP ARG | SWR SETTING |
|----------|-------------|
| 2 | 200 |
| 4 | 400 |
| 10 | 1000 |
| 20 | 2000 |
| 30 | 3000 |
| 40 | 4000 |
| 50 | 5000 |
| 60 | 6000 |
| 70 | 7000 |

THE SETTING OF SWR BIT 12, WILL FORCE THE TRAP HANDLER TO USE THE SWR SETTINGS THAT WERE IN FORCE WHEN THE LAST TRAP INSTRUCTION WAS EXECUTED.

IF A CNTRL-G IS OUTSTANDING ON THE CONSOLE TERMINAL WHEN THE TRAP WAS EXECUTED, THEN MONIT IS CALLED. IF CNTRL-O WAS OUTSTANDING THEN MODIFY IS CALLED

ENTRY POINT

TRAPSV

INPUT PARAMETERS:


```

2956                                     :           ON EXECUTION OF ANY TRAP INSTRUCTION
2957                                     :
2958                                     :
2959                                     :
2960                                     :           IMPLICIT INPUT PARAMETERS:
2961                                     :
2962                                     :           THE SWR HAS BEEN SET UP
2963                                     :
2964                                     :
2965                                     :           OUTPUT PARAMETERS:
2966                                     :
2967                                     :           EXIT TO THE CONTENTS OF THE ADDRESS FOLLOWING
2968                                     :           THE TRAP INSTRUCTION IF A MATCH WAS FOUND, ELSE THE
2969                                     :           PROGRAM WILL CONTINUE FROM THE ADDRESS AFTER THAT.
2970                                     :
2971                                     :
2972                                     :           IMPLICIT OUTPUT PARAMETERS:
2973                                     :
2974                                     :           NONE
2975                                     :
2976                                     :
2977                                     :           COMPLETION CODES:
2978                                     :
2979                                     :           NONE
2980                                     :
2981                                     :
2982                                     :           POSSIBLE ERROR CODES:
2983                                     :
2984                                     :           NONE
2985                                     :
2986                                     :
2987                                     :
2988 015124 004737 015536 TRAPSV: JSR    PC,SAVREG    ;SAVE REGS
2989 015130 011600          MOV    (SP),R0
2990 015132 016037 177776 014260 MOV    -2(R0),TRPARG ;GET TRAP CALL
2991 015140 105737 177560          TSTB  TKS          ;LOOK FOR MONITOR CALL
2992 015144 100015          BPL   3$
2993 015146 004737 016104          JSR   PC,READ
2994 015152 120027 000017          CMPB RO,#17      ;IS IT CNTRL-O ?
2995 015156 001003          BNE  1$          ;NO
2996 015160 004737 017162          JSR   PC,MODIFY  ;YES CALL MODIFIER ROUTINE
2997 015164 000405          BR   3$
2998 015166 120027 000007          1$: CMPB  RO,#7      ;IS IT CTRL-G?
2999 015172 001002          BNE  3$
3000 015174 004737 015344          JSR   PC,MONIT   ;YES, GO TO SWR MONITOR
3001 015200 017737 177014 014262 3$: MOV    @SWR,TRPSEL
3002 015206 132737 000020 014263 BITB  #20,TRPSEL+1 ;IS IT PRESERVE SCOPE
3003 015214 001012          BNE  TRPSCP      ;YES
3004 015216 013737 014262 014264 MOV    TRPSEL,TRPMEM ;NO SO SAVE SWITCH SETTING
3005 015224 042737 170777 014264 BIC   #170777,TRPMEM ;GET IT SO WE CAN COMPARE
3006 015232 132737 000016 014263 BITB  #16,TRPSEL+1 ;ANY SCOPE LEVELS SET
3007 015240 001412          BEQ  TRPLP      ;NO
3008 015242 013700 014260 TRPSCP: MOV   TRPARG,R0 ;YES
3009 015246 006000          ROR  R0         ;GET TO POSITION FOR COMPARE
3010 015250 006000          ROR  R0         ;FOR 10 & ABOVE
3011 015252 000300          SWAB R0        ;ONLY FOR SCOPE BITS(9-11)

```

| | | | | | | | |
|------|--------|--------|--------|---------|------|-------------|---------------------------|
| 3012 | 015254 | 042700 | 170777 | | BIC | #170777,RO | :ARGUMENT SCOPE BITS |
| 3013 | 015260 | 020037 | 014264 | | CMP | RO,TRPMEM | :AS SELECTED ? |
| 3014 | 015264 | 001422 | | | BEQ | TRPBAK | :YES, GO BACK |
| 3015 | 015266 | 013700 | 014262 | TRPLP: | MOV | TRPSEL,RO | :NO, TEST LOOP |
| 3016 | 015272 | 006100 | | | ROL | RO | :FOR BITS 7&8 |
| 3017 | 015274 | 006100 | | | ROL | RO | |
| 3018 | 015276 | 000300 | | | SWAB | RO | |
| 3019 | 015300 | 142700 | 000371 | | BICB | #371,RO | :CUT OUT SW06 |
| 3020 | 015304 | 142737 | 000370 | 014260 | BICB | #370,TRPARG | :ONLY SWITCHES 7 OR 8 NOW |
| 3021 | 015312 | 130037 | 014260 | | BITB | RO,TRPARG | :ANY SELECTED ? |
| 3022 | 015316 | 001005 | | | BNE | TRPBAK | :YES |
| 3023 | 015320 | 004737 | 015634 | | JSR | PC,RSTREG | |
| 3024 | 015324 | 062716 | 000002 | | ADD | #2,(SP) | :NO SCOPE OR LOOP,SO RUN |
| 3025 | 015330 | 000002 | | | RTI | | :PAST ARGUMENT AND RETURN |
| 3026 | 015332 | 004737 | 015634 | TRPBAK: | JSR | PC,RSTREG | :RESTORE REGS |
| 3027 | 015336 | 017616 | 000000 | | MOV | @(SP),(SP) | :RETURN ADDRESS TO STACK |
| 3028 | 015342 | 000002 | | | RTI | | :EXIT, LOOPING |

3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085

.SBTTL SWITCH REGISTER MONITOR

DESCRIPTION:

SWITCH REGISTER MONITOR. CALLED BY AN ERROR WITH SWR BIT 15 CLEAR, OR BY TYPING CTRL-G ON THE CONSOLE TELETYPE. IF USING HARDWARE SWR, SIMPLY ASKS FOR CTRL-C TO CONTINUE. OTHERWISE, IT PRINTS THE CURRENT CONTENTS OF THE SOFTWARE SWITCH REGISTER, FOLLOWED BY A PROMPT (>). THE NEW SWITCH REGISTER SETTINGS SHOULD THEN BE ENTERED AS AN OCTAL NUMBER, TERMINATED BY CARRIAGE RETURN. TYPING CARRIAGE RETURN ALONE WILL CAUSE THE SWITCH REGISTER CONTENTS TO REMAIN UNCHANGED. IF THE SWITCH REGISTER IS UPDATED TO SELECT A TEST (BIT 6 SET) THE NEW TEST WILL BE ENTERED IMMEDIATELY.

CALLING SEQUENCE:

JSR PC,MONIT

INPUT PARAMETERS:

BY TYPING CNTRL-G DURING THE RUNNING OF THE TESTS.

IMPLICIT INPUT PARAMETERS:

THE SOFTWARE SWITCH REGISTER, IF BEING USED MUST HAVE BEEN SET UP.

OUTPUT PARAMETERS:

IF RUNNING UNDER SOFTWARE SWITCH REGISTER MODE A NEW SETTING OF THE SWR COULD HAVE BEEN SET UP.

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

```

3086
3087
3088
3089 015344 010046
3090 015346 023727 014220 177570
3091 015354 001430
3092 015356 004737 016632
3093 015362 013302
3094 015364 013700 014246
3095 015370 004737 016722
3096 015374 012700 000076
3097 015400 004737 016710
3098 015404 004737 015732
3099 015410 005737 014276
3100 015414 001413
3101 015416 010037 014246
3102 015422 032737 000100 014246
3103 015430 001405
3104 015432 000137 001232
3105 015436 004737 015454
3106 015442 000775
3107 015444 004737 016576
3108 015450 012600
3109 015452 000207

```

```

:
:
:
MONIT: MOV RO,-(SP) ;SAVE RO
CMP SWR,#HSWR ;HARDWARE SWR?
BEQ MONITA ;IF YES, GO TO END
JSR PC,TYPOUT ;SWR=
SWRMSG
MOV SSWR,RO
JSR PC,PROCT
MOV #76,RO
JSR PC,PCHR ;PRINT '>'
JSR PC,OCTIN ;GET NEW SETTING
TST RAND ;ANY INPUT?
BEQ MONITX
MOV RO,SSWR ;YES UPDATE SSWR
BIT #100,SSWR ;TEST SELECTED?
BEQ MONITX
JMP START2 ;YES GO DO IT
MONITA: JSR PC,TYPCTC ;CTRL-C TO CONTINUE
BR MONITA
MONITX: JSR PC,CRLF
MOV (SP)+,RO ;RESTORE RO
RTS PC

```


3111
 3112
 3113
 3114
 3115
 3116
 3117
 3118
 3119
 3120
 3121
 3122
 3123
 3124
 3125
 3126
 3127
 3128
 3129
 3130
 3131
 3132
 3133
 3134
 3135
 3136
 3137
 3138
 3139
 3140
 3141
 3142
 3143
 3144
 3145
 3146
 3147
 3148
 3149
 3150
 3151
 3152
 3153
 3154
 3155
 3156
 3157
 3158
 3159
 3160
 3161
 3162
 3163
 3164
 3165
 3166

.SBTTL WAIT FOR CNTRL-C

DESCRIPTION:

ROUTINE TO WAIT FOR THE USER TO TYPE
 CNTRL-C ON THE CONSOLE. IF CNTRL-O IS HIT MODIFY
 IS CALLED, AND IF CNTRL-G IS HIT MONIT IS CALLED.

IF NONE OF THESE ARE HIT, THE ROUTINE WILL RETURN
 TO THE NEXT LOCATION AFTER THE CALL. IF CNTRL-C WAS
 HIT THE ROUTINE WILL RETURN TO THE NEXT LOCATION+2.

CALLING SEQUENCE:

JSR PC,TYPCTC

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

INPUT IS REQUESTED FROM THE CONSOLE

OUTPUT PARAMETERS:

RO IS CORRUPTED

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

A RETURN TO THE FIRST OR SECOND LOCATION AFTER
 THE CALL IS PERFORMED.

POSSIBLE ERROR CODES:

NONE

| | | | |
|---------|--------|-----------|--------------------------------|
| TYPCTC: | JSR | PC,TYPOUT | ;PRINTS TYPE CTRL/C WHEN READY |
| | REDMES | | |
| QEXIT: | JSR | PC,READ | ;CTRL/C ENTERED |
| | CMF | RO,#3 | |
| | BNE | QEXIT2 | ;NO |

| | | |
|--------|--------|--------|
| 015454 | 004737 | 016632 |
| 015460 | 013247 | |
| 015462 | 004737 | 016104 |
| 015466 | 020027 | 000003 |
| 015472 | 001005 | |

3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236

.SBTTL SAVE REGISTERS

DESCRIPTION:

ROUTINE TO SAVE ALL THE GENERAL PURPOSE
REGISTERS ON THE STACK, AND LEAVE THE ADDRESS OF THE
CALLING ROUTINE ON THE STACK. THE ROUTINE WILL RUN AT
PRIORITY 7 TO AVOID ANY INTERRUPTS

CALLING SEQUENCE:

JSR PC,SAVREG

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

REGISTERS 0 THRU 5 ARE SAVED ON THE STACK
AND THE RETURN ADDRESS OF THE CALLING ROUTINE IS
SET AS THE LAST ENTRY ON THE STACK

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

SAVREG: PSWREA SAVSTA
PSWSET #340
MOV (SP)+,SAVPC ;SAVE PC FOR RETURN FROM THIS
MOV (SP)+,SAVPC1 ; ROUTINE

015536
015546
015562 012637 014266
015566 012637 014270

| | | | | | | |
|------|--------|--------|--------|--------|--------------|-------------------|
| 3237 | 015572 | 010546 | | MOV | R5,-(SP) | |
| 3238 | 015574 | 010446 | | MOV | R4,-(SP) | |
| 3239 | 015576 | 010346 | | MOV | R3,-(SP) | |
| 3240 | 015600 | 010246 | | MOV | R2,-(SP) | |
| 3241 | 015602 | 010146 | | MOV | R1,-(SP) | |
| 3242 | 015604 | 010046 | | MOV | R0,-(SP) | |
| 3243 | 015606 | 013746 | 014270 | MOV | SAVPC1,-(SP) | |
| 3244 | 015612 | 013746 | 014266 | MOV | SAVPC,-(SP) | ;PUT PC READY FOR |
| 3245 | 015616 | | | PSWSET | SAVSTA | |
| 3246 | 015632 | 000207 | | RTS | PC | ;RETURN |
| 3247 | | | | | | |
| 3248 | | | | | | |
| 3249 | | | | | | |

3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306

015634
015644
015660 012637 014266
015664 012637 014270
015670 012600
015672 012601
015674 012602
015676 012603
015700 012604
015702 012605

.SBTTL RESTORE REGISTERS

DESCRIPTION:
RESTORE TO RESTORE THE GENERAL PURPOSE
REGISTERS. THE STACK IS LEFT IN THE SAME STATE AS IT
WAS WHEN SAVREG WAS CALLED.

CALLING SEQUENCE:
JSR PC,RSTREG

INPUT PARAMETERS:
NONE

IMPLICIT INPUT PARAMETERS:
NONE

OUTPUT PARAMETERS:
R0 THRU R5 RESTORED

IMPLICIT OUTPUT PARAMETERS:
NONE

COMPLETION CODES:
NONE

POSSIBLE ERROR CODES:
NONE

RSTREG: PSWREA SAVSTA
PSWSET #340
MOV (SP)+,SAVPC
MOV (SP)+,SAVPC1
MOV (SP)+,R0
MOV (SP)+,R1
MOV (SP)+,R2
MOV (SP)+,R3
MOV (SP)+,R4
MOV (SP)+,R5

| | | | | | | |
|------|--------|--------|--------|--------|--------------|-------------------|
| 3307 | 015704 | 013746 | 014270 | MOV | SAVPC1,-(SP) | |
| 3308 | 015710 | 013746 | 014266 | MOV | SAVPC,-(SP) | ;PUT PC READY FOR |
| 3309 | 015714 | | | PSWSET | SAVSTA | |
| 3310 | 015730 | 000207 | | RTS | PC | ;RETURN |

3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367

.SBTTL ENTER AN OCTAL NUMBER

DESCRIPTION:

ROUTINE TO ENTER AN OCTAL NUMBER ON THE CONSOLE.
THE NUMBER ENTERED IS RETURNED IN R0, AND THE VALUE RAND
IS SET TO NON ZERO IF ANY CHARACTERS WERE ENTERED.
TYPING CARRIAGE RETURN, LINE FEED OR ESCAPE WILL
TERMINATE THE LINE BEING ENTERED.
RUBOUT WILL DELETE THE LAST CHARACTER ENTERED,
CNTRL-U WILL RUBOUT THE WHOLE LINE, AND CNTRL-R
WILL TYPE OUT THE CHARACTERS SO FAR ENTERED.
RAND WILL BE SET TO NON ZERO IF ANY CHARACTERS
WERE HIT AND RANDC WILL CONTAIN THE TERMINATING
CHARACTER.

A '!' WILL BE PRINTED IF AN OVERFLOW CONDITION IS
DETECTED AND A '?' IF AN INVALID CHARACTER WAS
ENTERED.

CALLING SEQUENCE:

JSR PC,OCTIN

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

R0 CONTAINS THE NUMBER ENTERED.
RAND=0 IF NO CHARACTERS WERE ENTERED.

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

```

3368
3369
3370
3371 015732 004737 015536
3372 015736 005000
3373 015740 005037 014276
3374 015744 012737 014206 014376
3375 015752 012737 000006 014400
3376 015760 012737 000060 014402
3377 015766 012737 000067 014404
3378 015774 005037 014374
3379 016000 004737 016130
3380 016004 013701 014400
3381 016010 001420
3382 016012 012702 014206
3383 016016 112203
3384 016020 162703 000060
3385 016024 000241
3386 016026 006300
3387 016030 103417
3388 016032 006300
3389 016034 103415
3390 016036 006300
3391 016040 050300
3392 016042 005237 014276
3393 016046 005301
3394 016050 001362
3395 016052 010037 014274
3396 016056 004737 015634
3397 016062 013700 014274
3398 016066 000207
3399 016070 012700 000041
3400 016074 004737 016710
3401 016100 000137 015736
3402

```

```

:
:
:
OCTIN: JSR PC, SAVREG ; SAVE THE REGISTERS*
TYPOTB: CLR RO ; CLEAR RO
CLR RAND ; CLEAR FLAG WORD
MOV #OCTMSG, STRADD ; SET START ADDRESS OF STRING
MOV #6, STRLEN ; AND ITS SIZE
MOV #60, LOWCHR ; MINIMUM CHAR
MOV #67, UPPCHR ; MAXIMUM CHAR
CLR CNVFLG ; DON'T CONVERT TO UPPER CASE
JSR PC, GETSTR ; GET A STRING
MOV STRLEN, R1 ; ZERO LENGTH ?
BEQ TYPOTD ; YES
MOV #OCTMSG, R2 ; NO GET START ADDRESS OF STRING
TYPOTC: MOVB (R2)+, R3 ; GET A CHARACTER
SUB #60, R3 ; PUT IN RANGE
ASL RO ; SHIFT OUT RESULT
BCS TYPOTE ; C BIT SET ERROR
ASL RO ; TIMES 4
BCS TYPOTE ; C BIT ERROR
ASL RO ; TIMES 8
BIS R3, RO ; SET NEW BITS IN
INC RAND ; SET GOOD
DEC R1
BNE TYPOTC ; LOOP IF MORE TO COME
TYPOTD: MOV RO, TYPOTA ; SAVE FINAL NUMBER
JSR PC, RSTREG ; RESTORE REGISTERS
MOV TYPOTA, RO ; GET RESULT
RTS PC
TYPOTE: MOV #41, RO ; OVERFLOW
JSR PC, PCHR ; PRINT A !
JMP TYPOTB ; THEN GO AGAIN
:

```


3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455

.SBTTL READ A SINGLE CHARACTER

DESCRIPTION:

ROUTINE TO READ A SINGLE CHARACTER
FROM THE CONSOLE. THE CHARACTER IS RETURN IN RO
AND HAS THE 200 BIT STRIPPED OFF.

CALLING SEQUENCE:

JSR PC,READ

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

RO CONTAINS THE CHARACTER READ IN.

IMPLICIT OUTPUT PARAMETERS:

THE CHARACTER IS ECHOED ON THE TERMINAL

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

READ: -TSTB TKS
BPL READ
MOV TKB,RO
BIC #177600,RO
JSR PC,PCHR
RTS PC

016104 105737 177560
016110 100375
016112 013700 177562
016116 042700 177600
016122 004737 016710
016126 000207

3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512

.SBTTL ENTERING A CHARACTER STRING

DESCRIPTION:

ROUTINE TO ENTER A STRING OF CHARACTERS ON THE CONSOLE TERMINAL. VARIOUS CONTROL CODES ARE USED TO CONTROL HOW THE CHARACTERS ARE INTERPTED:

ESCAPE, CARRIAGE RETURN AND LINE FEED ARE USED TO TERMINATE THE STRING. RUBOUT WILL DELETE THE LAST CHARACTER ENTERED, CNTRL-U WILL DELETE THE ALL THE CHARACTERS ENTERED, AND CNTRL-R WILL ECHO THOSE CHARACTERS ALREADY ENTERED.

ON ENTRY THE FOLLOWING POINTERS ARE USED:
STRADD TO INDICATE THE START OF THE CHARACTER STRING
STRLEN TO INDICATE ITS SIZE
CVNFLG SET TO CONVERT LOWER CASE TO UPPER CASE
UPPCHR TO INDICATE THE HIGHEST CHARACTER CODE
LOWCHR TO INDICATE THE LOWEST CHARACTER CODE

ON EXIT THE LOCATION STRLEN WILL INDICATE THE NUMBER OF CHARACTERS ENTERED AND RANDC WILL CONTAIN THE TERMINATING CHARACTER

IF AN INVALID CHARACTER WAS HIT A '?' IS PRINTED

CALLING SEQUENCE:

JSR PC,GETSTR

INPUT PARAMETERS:

STRADD THE START ADDRESS OF THE STRING
STRLEN THE NUMBER OF CHARACTERS TO READ
UPPCHR THE HIGHEST CHARACTER CODE ALLOWED
LOWCHR THE LOWEST CHARACTER CODE ALLOWED
CNVFLG TO INDICATE LOWER TO UPPER CASE CONVERSION

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

STRLEN THE NUMBERS OF CHARACTERS READ
RANDC THE TERMINATING CHARACTER

3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533 016130 004737 015536
3534 016134 005037 014406
3535 016140 013702 014376
3536 016144 005001
3537 016146 105737 177560
3538 016152 100375
3539 016154 013700 177562
3540 016160 042700 177600
3541
3542
3543
3544 016164 020027 000015
3545 016170 001002
3546 016172 000137 016554
3547 016176 020027 000012
3548 016202 001002
3549 016204 000137 016554
3550 016210 020027 000033
3551 016214 001002
3552 016216 000137 016554
3553 016222 020027 000177
3554 016226 001002
3555 016230 000137 016402
3556 016234 020027 000022
3557 016240 001002
3558 016242 000137 016500
3559 016246 020027 000025
3560 016252 001002
3561 016254 000137 016450
3562
3563
3564
3565
3566
3567 016260 020027 000140
3568 016264 002405

```

: IMPLICIT OUTPUT PARAMETERS:
:
:     NONE
:
: COMPELETION CODES:
:
:     NONE
:
: POSSIBLE ERROR CONDITIONS:
:
:     A '?' IS PRINTED IF AN INVALID CHARACTER IS ENTERED
:
:
: GETSTR: JSR      PC,SAVREG      ; SAVE REGISTERS
: GETCH1: CLR      RUBFLG        ; CLEAR RUBOUT FLAG
:         MOV      STRADD,R2     ; GET STARTING POINTER
:         CLR      R1           ; SET INTITAL LENGTH
: GETCH2: TSTB    TKS           ; ANY TTY INPUT
:         BPL      GETCH2       ; NO WAIT FOR IT
:         MOV      TKB,R0       ; GET CHARACTER
:         BIC      #177600,R0   ; CLEAR RUBBISH
:
: CHECK FOR CONTROL CODES
:
:         CMP      R0,#15      ; WAS IT <CR> ?
:         BNE      1$         ; NO
:         JMP      CHRFIN
: 1$:      CMP      R0,#12     ; WAS IT <LF>
:         BNE      3$         ; NO
:         JMP      CHRFIN
: 3$:      CMP      R0,#33     ; WAS IT <ESC>
:         BNE      5$         ; NO
:         JMP      CHRFIN
: 5$:      CMP      R0,#177    ; RUBOUT ?
:         BNE      7$         ; NO
:         JMP      RUBCHR     ; YES
: 7$:      CMP      R0,#22     ; CNTRL-R
:         BNE      9$         ; NO
:         JMP      LINECH    ; YES EXCO LINE
: 9$:      CMP      R0,#25     ; CNTRL-U
:         BNE      GETCH3    ; NO
:         JMP      LINDEL    ; YES DELETE LINE
:
: IT WAS NOT A CONTROL CODE CHECK
: FOR LEGALITY ?
:
: GETCH3: CMP      R0,#140    ; IS IT LOWER CASE
:         BLT      3$         ; NO

```

```

3569 016266 005737 014374          TST      CNVFLG      ; YES DO WE CONVERT TO UPPER
3570 016272 001402                   BEQ      3$         ; NO
3571 016274 042700 000040          BIC      #40,R0    ; YES STRIP 40 OFF
3572 016300 020037 014402          3$: CMP      R0,LOWCHR ; IS THE CHAR TOO SMALL
3573 016304 002002                   BGE      5$         ; NO
3574 016306 000137 016534          JMP      ILLCHR    ; YES
3575 016312 020037 014404          5$: CMP      R0,UPPCHR ; IS THE CHAR TOO BIG ?
3576 016316 003402                   BLE      GETCH4    ; NO
3577 016320 000137 016534          JMP      ILLCHR    ; YES ITS ILLEGAL
3578
3579          ; WE HAVE A LEGAL CHARACTER
3580
3581 016324 010003          GETCH4: MOV     R0,R3      ; SAVE CHAR
3582 016326 005737 014406          TST      RUBFLG      ; IS RUBOUT SET
3583 016332 001406          BEQ      3$         ; NO
3584 016334 012700 000057          MOV     #57,R0      ; YES PRINT A '/'
3585 016340 004737 016710          JSR     PC,PCHR     ;
3586 016344 005037 014406          CLR     RUBFLG      ; CLEAR RUBOUT FLAG
3587 016350 005201          3$: INC     R1        ; UPDATE CHAR COUNT
3588 016352 020137 014400          CMP     R1,STRLEN   ; END OF STRING SEEN
3589 016356 003403          BLE     5$         ; NO
3590 016360 005000          CLR     R0         ; YES, FORCE A NULL TERMINATOR
3591 016362 000137 016554          JMP     CHRFIN      ; AND COMPLETE
3592 016366 010300          5$: MOV     R3,R0    ; RESTORE CHAR
3593 016370 004737 016710          JSR     PC,PCHR     ; ECHO IT
3594 016374 110022          MOVB    R0,(R2)+    ; SAVE IT IN BUFFER
3595 016376 000137 016146          JMP     GETCH2     ; NO GET NEXT ONE
3596
3597          ; RUBOUT WAS HIT
3598
3599 016402 005701          RUBCHR: TST     R1      ; ANY CHARACTERS TO RUBOUT
3600 016404 001002          BNE     3$         ; YES
3601 016406 000137 016470          JMP     LINDL1     ; NO
3602 016412 005737 014406          3$: TST     RUBFLG   ; IS RUBOUT SET ALREADY ?
3603 016416 001006          BNE     5$         ; YES
3604 016420 012700 000057          MOV     #57,R0    ; NO PRINT A '/'
3605 016424 004737 016710          JSR     PC,PCHR     ;
3606 016430 005237 014406          INC     RUBFLG     ; SET RUBOUT
3607 016434 114200          5$: MOVB    -(R2),R0 ; GET LAST CHAR ENTERED
3608 016436 004737 016710          JSR     PC,PCHR     ; PRINT IT
3609 016442 005301          DEC     R1        ; REDUCE COUNT
3610 016444 000137 016146          JMP     GETCH2     ; GET ANOTHER CHAR
3611
3612          ; RUBOUT LINE WAS HIT
3613
3614 016450 012700 000136          LINDEL: MOV     #136,R0 ; PRINT A ^
3615 016454 004737 016710          JSR     PC,PCHR     ;
3616 016460 012700 000125          MOV     #125,R0   ; THEN U
3617 016464 004737 016710          JSR     PC,PCHR     ;
3618 016470 004737 016576          LINDL1: JSR     PC,CRLF ; START ON A NEWLINE
3619 016474 000137 016134          JMP     GETCH1     ; A GO BACK TO BEGINNING
3620
3621          ; CNTRL-R WAS HIT
3622
3623 016500 112712 000100          LINECH: MOVB    #'@,(R2) ; PUT IN A TERMINATOR
3624 016504 004737 016576          JSR     PC,CRLF   ; START ON A NEW LINE
  
```



```

3625 016510 013737 014376 016526      MOV    STRADD,3$      ; SET START ADDRESS
3626 016516 005037 014406                CLR    RUBFLG         ; CLEAR RUBOUTS
3627 016522 004737 016632                JSR    PC,TYPOUT     ; PRINT LINE
3628 016526 000000                3$:  0
3629 016530 000137 016146                JMP    GETCH2        ; AND GET ANOTHER CHAR
3630
3631      ; ILLEGAL CHARACTER ENTERED
3632
3633 016534 004737 016710      ILLCHR: JSR    PC,PCHR      ; ECHO IT
3634 016540 012700 000077                MOV    #'?,RO        ; PRINT A ?
3635 016544 004737 016710                JSR    PC,PCHR
3636 016550 000137 016500                JMP    LINECH        ; THE ECHO LINE
3637
3638      ; A TERMINATOR WAS FOUND
3639
3640 016554 010037 014410      CHRFIN: MOV    RO,RANDC    ; SAVE TERMINATOR
3641 016560 163702 014376                SUB    STRADD,R2      ; CALCULATE BYTE COUNT
3642 016564 010237 014400                MOV    R2,STRLN      ; SET IT FOR RETURN
3643 016570 004737 015634                JSR    PC,RSTREG     ; RESTORE REGISTERS
3644 016574 000207                RTS    PC             ; AND EXIT
  
```

3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701

.SBTTL CARRIAGE RETURN LINE FEED

DESCRIPTION:

ROUTINE TO DO A <CR> <LF> ON THE
CONSOLE TERMINAL

CALLING SEQUENCE:

JSR PC,CRLF

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

THE CARRIAGE WILL BE PLACED ON A NEW LINE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

```

CRLF:  MOV    RO,-(SP)
      MOV    #15,RO      ;PRINT A RETURN - LINE FEED
      JSR    PC,PCHR
      CLR    RO          ;DUMMY
      JSR    PC,PCHR
      MOV    #12,RO
      JSR    PC,PCHR
      MOV    (SP)+,RO
      RTS    PC

```


3702

3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759

.SBTTL PRINT AN ASCII MESSAGE

DESCRIPTION:

ROUTINE TO PRINT A STRING OF ASCII
CHARACTERS ON THE CONSOLE TERMINAL. CERTAIN
CHARACTERS WITHIN THE STRING ARE INTERPRETED
AS CONTROL CODES, THESE ARE:

133 (L) WILL GENERATE A <CR>,<LF>
100 (@) WILL SIGNIFY END OF MESSAGE

THE ADDRESS OF THE MESSAGE STRING TO BE PRINTED
WILL BE HELD IN THE LOCATION FOLLOWING THE CALL
TO THE ROUTINE, IE:

JSR PC,TYPOUT
ADDR

CALLING SEQUENCE:

JSR PC,TYPOUT

INPUT PARAMETERS:

THE ADDRESS OF THE MESSAGE STRING FOLLOWS
THE SUBROUTINE CALL.

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

THE SPECIFIED MESSAGE WILL BE PRINTED

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

| | | | | | | | |
|------|--------|--------|--------|---------|------|------------|----------------------------------|
| 3760 | | | | : | | | |
| 3761 | | | | : | | | |
| 3762 | | | | : | | | |
| 3763 | 016632 | 004737 | 015536 | TYPOUT: | JSR | PC, SAVREG | :SAVE REGS |
| 3764 | 016636 | 017601 | 000000 | | MOV | @(SP), R1 | :R1 POINTS AT STRING |
| 3765 | 016642 | 062716 | 000002 | | ADD | #2, (SP) | :JUMPS OVER ARGUMENT |
| 3766 | 016646 | 111100 | | PMSG1: | MOVB | @R1, R0 | :PRINT THE MESSAGE POINTED |
| 3767 | 016650 | 022700 | 000100 | | CMP | #100, R0 | :TO BY R1 UNTIL @, WHICH IS END. |
| 3768 | 016654 | 001412 | | | BEQ | PMSG4 | :L MEANS CR-LF |
| 3769 | 016656 | 022700 | 000133 | | CMP | #133, R0 | |
| 3770 | 016662 | 001003 | | | BNE | PMSG2 | |
| 3771 | 016664 | 004737 | 016576 | | JSR | PC, CRLF | |
| 3772 | 016670 | 000402 | | | BR | PMSG3 | |
| 3773 | 016672 | 004737 | 016710 | PMSG2: | JSR | PC, PCHR | |
| 3774 | 016676 | 005201 | | PMSG3: | INC | R1 | |
| 3775 | 016700 | 000762 | | | BR | PMSG1 | |
| 3776 | 016702 | 004737 | 015634 | PMSG4: | JSR | PC, RSTREG | :RESTORE REGS |
| 3777 | 016706 | 000207 | | | RTS | PC | |

3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824 016710 004737 015074
3825 016714 010037 177566
3826 016720 000207

.SBTTL PRINT A CHARACTER

DESCRIPTION:

ROUTINE TO PRINT A CHARACTER ON THE
CONSOLE. RO CONTAINS THE CHARACTER TO BE PRINTED

CALLING SEQUENCE:

JSR PC,PCHR

INPUT PARAMETERS:

RO CONTAINS THE CHARACTER TO BE PRINTED

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

THE CHARACTER SELECTED WILL BE PRINTED

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

PCHR: JSR PC,TPREDY ;PRINTER READY
MOV RO,TPB
RTS PC

3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883

016722 004737 015536
016726 012701 014160
016732 112721 000040
016736 020127 014166
016742 001373
016744 000241
016746 010002
016750 042702 177770
016754 062702 000060

.SBTTL PRINT AN OCTAL NUMBER

DESCRIPTION:

ROUTINE TO PRINT AN OCTAL NUMBER ON THE CONSOLE TERMINAL. R0 CONTAINS THE BINARY REPRESENTATION ON THE NUMBER THAT IS TO BE PRINTED.

CALLING SEQUENCE:

JSR PC,PROCT

INPUT PARAMETERS:

R0 CONTAINS THE NUMBER THAT IS TO BE PRINTED

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

THE NUMBER SPECIFIED WILL BE PRINTED

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE

PROCT: JSR PC,SAVREG ;SAVE REGS
MOV #OCTNUM,R1 ;POINTER TO MESSAGE
PROCT1: MOVB #40,(R1)+ ;FILL WITH SPACES
CMP R1,#OCTNUM+6 ;ALL DONE
BNE PROCT1 ;NO
PRCT1A: CLC ;CLEAR AT START
PROCT2: MOV R0,R2 ;SAVE CHARA
BIC #177770,R2 ;CLEAR ALL BUT BOTTOM 3 BITS
ADD #60,R2 ;NOW ASCII!

3908
 3909
 3910
 3911
 3912
 3913
 3914
 3915
 3916
 3917
 3918
 3919
 3920
 3921
 3922
 3923
 3924
 3925
 3926
 3927
 3928
 3929
 3930
 3931
 3932
 3933
 3934
 3935
 3936
 3937
 3938
 3939
 3940
 3941
 3942
 3943
 3944
 3945
 3946
 3947
 3948
 3949
 3950
 3951
 3952
 3953
 3954
 3955
 3956
 3957
 3958
 3959
 3960
 3961
 3962
 3963

.SBTTL PRINT A DECIMAL NUMBER

DESCRIPTION:

ROUTINE TO PRINT A SIGNED, OR UNSIGNED
 DECIMAL NUMBER ON THE CONSOLE. RO CONTAINS THE
 BINARY REPRESENTATION ON THE NUMBER TO BE PRINTED

CALLING SEQUENCE:

JSR PC,BASE10 FOR UNSIGNED
 JSR PC,BASM10 FOR SIGNED

INPUT PARAMETERS:

RO CONTAINS THE NUMBER TO BE PRINTED

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

NONE

IMPLICIT OUTPUT PARAMETERS:

THE NUMBER SPECIFIED WILL BE PRINTED

COMPLETION CODES.

NONE

POSSIBLE ERROR CODES:

NONE

ROUTINE TO PRINT A SIGNED DECIMAL NO.
 ENTER WITH IT HELD IN RO
 RO :IS IT -VE
 BASE10 :NO
 RO :YES SO MAKE +VE
 #55,BASE11 :PUT '-' IN MESSAGE
 BAS10A :GO DO IT

BASM10: TST
 BPL BASE10
 NEG RO
 MOVB #55,BASE11
 BR BAS10A

017046 005700
 017050 100005
 017052 005400
 017054 112737 000055 014170
 017062 000403

ROUTINE TO PRINT UNSIGNED DECIMAL NO.

| | | | | | | | | |
|------|--------|--------|--------|--------|--------------|-------------|---------------|------------------------------|
| 3964 | | | | | | | | |
| 3965 | 017064 | 112737 | 000040 | 014170 | BASE10: MOVB | #40, BASE11 | | :PUT SPACE AS 1ST CHARA |
| 3966 | 017072 | 004737 | 015536 | | BAS10A: JSR | PC, SAVREG | | :SAVE REGS |
| 3967 | 017076 | 012703 | 014171 | | | MOV | #BASE11+1, R3 | :REST OF MESSAGE WITH SPACES |
| 3968 | 017102 | 112723 | 000040 | | BASE1A: MOVB | #40, (R3)+ | | |
| 3969 | 017106 | 022703 | 014176 | | | CMP | #BASE11+6, R3 | :ALL DONE |
| 3970 | 017112 | 001373 | | | | BNE | BASE1A | :NO |
| 3971 | 017114 | 005001 | | | BASE1D: CLR | R1 | | :R1 IS RECEIVER |
| 3972 | 017116 | 020027 | 000012 | | BASE1B: CMP | RO, #12 | | :MORE THAN 10 |
| 3973 | 017122 | 103404 | | | | BLO | BASE1C | :YES SO DONE THIS TIME |
| 3974 | 017124 | 162700 | 000012 | | | SUB | #12, RO | :NO SO SUB 10 & ADD 1 TO R1 |
| 3975 | 017130 | 005201 | | | | INC | R1 | |
| 3976 | 017132 | 000771 | | | | BR | BASE1B | :GO DO AGAIN |
| 3977 | 017134 | 062700 | 000060 | | BASE1C: ADD | #60, RO | | :MAKE ASCII & STORE |
| 3978 | 017140 | 110043 | | | | MOVB | RO, -(R3) | |
| 3979 | 017142 | 010100 | | | | MOV | R1, RO | |
| 3980 | 017144 | 001363 | | | | BNE | BASE1D | |
| 3981 | 017146 | 004737 | 015634 | | | JSR | PC, RSTREG | :RESTORE REGS |
| 3982 | 017152 | 004737 | 016632 | | | JSR | PC, TYP0UT | :TYPE MESSAGE |
| 3983 | 017156 | 014170 | | | | BASE11 | | :ADDRESS OF MESSAGE |
| 3984 | 017160 | 000207 | | | RTS | PC | | |

3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041

.SBTTL MODIFY THE PROGRAM

DESCRIPTION:

ROUTINE TO MODIFY A LOCATION IN MEMORY
ENTERED BY TYPING CNTRL-O ON THE CONSOLE
TERMINAL.

PROMPTS (\$) FOR AN ADDRESS TO EXAMINE
AND PRINTS IT IN THE FORM

ADDR CONTENTS /

THEN A NEW VALUE CAN BE ENTERED

VIS:

ADDR CONTENTS / NEW VALUE

THE NEW VALUE CAN BE TERMINATED USING
<CR>, <LF>, OR <ESC>

<LF> TO EXAMINE THE NEXT LOC
<CR> TO SELECT ANOTHER ADDRESS
<ESC> TO EXIT

CALLING SEQUENCE:

JSR PC,MODIFY

INPUT PARAMETERS:

NONE

IMPLICIT INPUT PARAMETERS:

ENTERED BY TYPING CNTRL-O DURING
THE RUNNING OF THE TESTS

OUTPUT PARAMETERS:

NONE

4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097

017162 004737 015536
 017166
 017176
 017212 004737 016632
 017216 013314
 017220 004737 015732
 017224 005737 014276
 017230 001506
 017232 010037 014302
 017236 032737 000001 014302
 017244 001404
 017246 004737 016632
 017252 013425
 017254 000756
 017256 012737 020232 000004
 017264 012737 000340 000006
 017272 005037 014312
 017276 005777 175000
 017302 005737 014312
 017306 001341
 017310 004737 016576
 017314 013700 014302
 017320 004737 016722
 017324 012700 000040
 017330 004737 016710
 017334 017700 174742
 017340 004737 016722
 017344 004737 016632
 017350 013321
 017352 004737 015732
 017356 005737 014276
 017362 001402
 017364 010077 174712
 017370 013700 014410
 017374 001706
 017376 020027 000015
 017402 001703
 017404 020027 000012
 017410 001006

```

:
:      IMPLICIT OUTPUT PARAMETERS:
:
:      THE LOCATIONS SPECIFIED WILL HAVE BEEN
:      MODIFIED
:
:      COMPLETION CODES:
:
:      NONE
:
:      POSSIBLE ERROR CODES:
:
:      NONE
:
:      MODIFY: JSR      PC,SAVREG      ;SAVE REGISTERS
:              PSWREA MODSAV
:              PSWSET #340
:
:      MOD11: JSR      PC,TYPEOUT     ;PROMPT $ FOR AN ADDRESS
:              MODADM                ;TO HAVE A LOOK AT
:              JSR      PC,OCTIN      ;READ REPLY
:              TST      RAND          ;ANYTHING READ ?
:              BEQ      MODXIT        ;NO, SO EXIT
:              MOV      RO,MODADR     ;ELSE SAVE OUR ADDRESS
:      MOD12: BIT      #1,MODADR     ;IS IT EVEN ?
:              BEQ      MODI3        ;YES WE CAN USE IT
:              JSR      PC,TYPEOUT     ;ELSE SAY IT IS AN ODD
:              ODAMSG                ;ADDRESS, AND REPROMPT
:              BR      MOD11
:      MOD13: MOV      #NXMTRP,4      ;PLUG TRAP THRU 4
:              MOV      #340,6       ;FOR NXM TESTS
:              CLR      TRPERR        ;CLEAR NXM FLAG
:              TST      @MODADR       ;TEST OUR ADDRESS
:              TST      TRPERR        ;DOES IT EXIST
:              BNE      MOD11         ;NO TRY AGAIN
:              JSR      PC,CRLF        ;START ON A NEW LINE
:              MOV      MODADR,RO     ;PRINT OUR ADDRESS
:              JSR      PC,PROCT
:              MOV      #40,RO        ;THEN A SPACE
:              JSR      PC,PCHR        ;AS A SEPARATOR
:              MOV      @MODADR,RO    ;THEN PRINT THE CONTENTS
:              JSR      PC,PROCT
:              JSR      PC,TYPEOUT     ;PROMPT FOR THE NEW
:              MODSPA                ;CONTENTS '/'
:              JSR      PC,OCTIN      ;READ REPLY
:              TST      RAND          ;ANYTHING GIVEN
:              BEQ      MODI4        ;NO, DON'T UPDATE
:              MOV      RO,@MODADR     ;ELSE SET NEW VALUE
:      MOD14: MOV      RANDC,RO      ; GET TERMINATOR
:              BEQ      MOD11         ; NULL MEANS <CR>
:              CMP      RO,#15        ;WAS IT <CR> ?
:              BEQ      MOD11         ;YES GET NEXT ADDRESS
:              CMP      RO,#12        ;WAS IT <LF> ?
:              BNE      MOD15         ;NO
    
```


| | | | | | | | |
|------|--------|--------|--------|--------|-------------|-----------|------------------------------|
| 4098 | 017412 | 004737 | 016576 | | JSR | PC,CRLF | : NEWLINE |
| 4099 | 017416 | 062737 | 000002 | 014302 | ADD | #2,MODADR | :UPDATE THE ADDRESS BY 2 |
| 4100 | 017424 | 000704 | | | BR | MOD12 | : THEN TRY THIS ADDRESS |
| 4101 | 017426 | 020027 | 000033 | | MOD15: CMP | R0,#33 | :WAS IT EXIT ? |
| 4102 | 017432 | 001405 | | | BEQ | MODXIT | :YES DISSAPPEAR |
| 4103 | 017434 | 012700 | 000077 | | MOV | #77,R0 | :ELSE GIVE A ? |
| 4104 | 017440 | 004737 | 016710 | | JSR | PC,PCHR | :AS AN ERROR |
| 4105 | 017444 | 000751 | | | BR | MOD14 | :THEN TRY AGAIN |
| 4106 | 017446 | 012737 | 000006 | 000004 | MODXIT: MOV | #6,4 | :PLUG BACK TRAP THRU 4 |
| 4107 | 017454 | 012737 | 000004 | 000006 | MOV | #4,6 | :WITH WHAT IS WAS |
| 4108 | 017462 | 004737 | 016576 | | JSR | PC,CRLF | :PUT OURSELVES ON A NEW LINE |
| 4109 | 017466 | | | | PSWSET | MODSAV | |
| 4110 | 017502 | 004737 | 015634 | | JSR | PC,RSTREG | :RESTORE REGISTERS |
| 4111 | 017506 | 000207 | | | RTS | PC | :AND GO AWAY |
| 4112 | | | | | | | |

4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169

.SBTTL SET BUS AND VECTOR ADDRESSES

DESCRIPTION:

ROUTINE TO SET UP THE BUS AND VECTOR ADDRESSES AND PRIORITY LEVELS OF THE DEVICE UNDER TEST. THE VALIDITY OF THE VARIOUS ADDRESSES ARE CHECKED. THE ARGUMENTS AFTER THE CALL MUST BE SET TO INDICATE THE NUMBER OF BUS AND VECTOR ADDRESSES AND THE NUMBER OF PRIORITIES REQUIRED, AND THE LOCATIONS WHERE THE VARIOUS PARAMTERES ARE TO BE STORED

IT IS CALLED THUS:

JSR R5,BUSSET
.BYTE A,B
.WORD C
.BYTE D,E
.WORD F,G
.BYTE H,J
.WORD K,L

WHERE A IS EITHER 7 IF THERE ARE 4 OR LESS BUS ADDRESSES OR 17 IF THERE ARE 5-8 ADDRESSES, OR 37 IF THERE 9-16, AND SO ON. B IS THE NUMBER OF BUS ADDRESSES. C IS THE ADDRESS INTO WHICH THE FIRST BUS ADDRESS WILL BE LOADED, SUBSEQUENT BUS ADDRESSES WIL BE LOADED INTO THE LOCATION FOLLOWING THE ADDRESS C.HENCE IF THERE A 4 BUS ADDRESSES THE C MUST POINT TO A FOUR WORD BUFFER THAT WILL CONTAIN THE 4 BUS ADDRESSES. D IS THE NUMBER OF VECTOR ADDRESSES E IS THE NUMBER OF PRIORITY LEVELS, AND IF THE VECTOR IS TO BE ON A 10 BYTE BOUNDARY, BIT 7 OF E MUST BE SET AS A FLAG. F IS THE ADDRESS INTO WHICH THE FIRST VECTOR ADDRESS WILL BE LOADED.G IS THE ADDRESS INTO WHICH THE FIRST PRIORITY WILL BE LOADED. H AND J ARE EQUIVALENT TO D AND E INCASE MORE THAN ONE VECTOR GROUP ARE PRESENT. K AND L ARE THE ADDRESSES EQUIVALENT TO F AND G. THE SEQUENCE H,J,K,L CAN BE REPEATED FOR AS MANY VECTOR PAIRS AS REQUIRED. THE SEQUENCE IS TERMINATED BY SETTING H AND J TO ZERO

CALLING SEQUENCE:

JSR R5,BUSSET

INPUT PARAMETERS:

THE ARGUMENTS TO SET UP THE ADDRESSES, VECTORS AND PRIORITIES MUST BE SET UP, AS DESCRIBED ABOVE

IMPLICIT INPUT PARAMETERS:


```

4170      :          NONE
4171      :
4172      :
4173      :          OUTPUT PARAMETERS:
4174      :
4175      :          NONE
4176      :
4177      :
4178      :          IMPLICIT OUTPUT PARAMETERS:
4179      :
4180      :          THE BUS AND VECTOR ADDRESSES WILL
4181      :          BE SET UP AS SPECIFIED BY THE ARGUEMENTS
4182      :
4183      :
4184      :          COMPLETION CODES:
4185      :
4186      :          NONE
4187      :
4188      :
4189      :          POSSIBLE ERROR CODES:
4190      :
4191      :          NONE
4192      :
4193      :
4194      :
4195 017510 004737 015536      :BUSSET: JSR      PC,SAVREG      ;SAVE REGISTER CONTENTS
4196 017514 012737 020232 000004      :      MOV      #NXMTRP,4      ;SET UP MEMORY ERROR TRAP
4197 017522 012737 000340 000006      :      MOV      #340,6
4198      :
4199 017530 012537 014306      :      MOV      (R5)+,MASK      ;GET MASK AND NO OF ADDRESSES
4200 017534 012537 014310      :      MOV      (R5)+,BASADD     ;GET BASE BUS ADDRESS
4201      :
4202 017540 004737 016632      :BUSSE1: JSR      PC,TYPOUT
4203 017544 013332      :      BAMSG
4204 017546 004737 015732      :1$:      JSR      PC,OCTIN      ;FIRST BUS ADDRESS IS....
4205 017552 005737 014276      :      TST      RAND            ;INPUT OCTAL ADDRESS
4206 017556 001004      :      BNE      2$
4207 017560 004737 016632      :      JSR      PC,TYPOUT
4208 017564 013712      :      NODEFM
4209 017566 000767      :      BR       1$
4210 017570 133700 014306      :2$:      BITB   MASK,R0        ;CHECK INPUT
4211 017574 001406      :      BEQ     BUSSE2
4212      :
4213 017576 004737 016632      :      JSR      PC,TYPOUT
4214 017602 013425      :      ODAMSG
4215 017604 004737 016722      :BUSSE1A: JSR      PC,PROCT
4216 017610 000753      :      BR       BUSSE1
4217      :
4218 017612 005037 014312      :BUSSE2: CLR      TRPERR
4219 017616 113701 014307      :      MOVB   MASK+1,R1      ;SET UP COUNT
4220 017622 013702 014310      :      MOV      BASADD,R2     ;SET UP ADDRESS BASE
4221 017626 010022      :BUSSE3: MOV      R0,(R2)+      ;SET UP ADDRESS
4222 017630 005710      :      TST      (R0)          ;CHECK ADDRESS EXISTS
4223 017632 005737 014312      :      TST      TRPERR        ;CHECK NON-EXISTANT MEMORY FLAG
4224 017636 001362      :      BNE     BUSS1A
4225 017640 062700 000002      :      ADD     #2,R0          ;UPDATE TO NEXT ADDRESS

```

| | | | | | | | | | |
|------|--------|--------|---------------|---------|--------|---------------|--|--|---------------------------------|
| 4226 | | | | | | | | | |
| 4227 | 017644 | 005301 | | | DEC | R1 | | | |
| 4228 | 017646 | 001367 | | | BNE | BUSSE3 | | | |
| 4229 | 017650 | 004737 | 016632 | | JSR | PC, TYPOUT | | | ; PROMPT FOR VECTOR GROUP |
| 4230 | 017654 | 013110 | | | FIRVMS | | | | |
| 4231 | | | | | | | | | |
| 4232 | 017656 | 012537 | 014306 | | MOV | (R5)+, MASK | | | ; GET NO OF VECTORS |
| 4233 | 017662 | 012537 | 014310 | BUS3A: | MOV | (R5)+, BASADD | | | ; GET BASE VECTOR ADDRESS |
| 4234 | | | | | | | | | |
| 4235 | 017666 | 004737 | 016632 | BUSSE4: | JSR | PC, TYPOUT | | | |
| 4236 | 017672 | 013366 | | | VAMSG | | | | ; FIRST VECTOR ADDRESS IS.... |
| 4237 | 017674 | 004737 | 015732 | 1\$: | JSR | PC, OCTIN | | | ; INPUT OCTAL ADDRESS |
| 4238 | 017700 | 005737 | 014276 | | TST | RAND | | | |
| 4239 | 017704 | 001004 | | | BNE | 2\$ | | | |
| 4240 | 017706 | 004737 | 016632 | | JSR | PC, TYPOUT | | | |
| 4241 | 017712 | 013712 | | | NODEFM | | | | |
| 4242 | 017714 | 000767 | | | BR | 1\$ | | | |
| 4243 | 017716 | 005737 | 014306 | 2\$: | TST | MASK | | | ; CHECK MASK TYPE |
| 4244 | 017722 | 100007 | | | BPL | BUSSE6 | | | |
| 4245 | | | | | | | | | |
| 4246 | 017724 | 032700 | 000007 | | BIT | #7, R0 | | | ; CHECK ADDRESS |
| 4247 | 017730 | 001404 | | | BEQ | BUSSE6 | | | |
| 4248 | | | | | | | | | |
| 4249 | 017732 | 004737 | 016632 | BUSSE5: | JSR | PC, TYPOUT | | | |
| 4250 | 017736 | 013425 | | | ODAMSG | | | | ; INVALID ADDRESS |
| 4251 | 017740 | 000752 | | | BR | BUSSE4 | | | |
| 4252 | | | | | | | | | |
| 4253 | 017742 | 032700 | 000003 | BUSSE6: | BIT | #3, R0 | | | ; CHECK ADDRESS |
| 4254 | 017746 | 001371 | | | BNE | BUSSE5 | | | |
| 4255 | | | | | | | | | |
| 4256 | 017750 | 022700 | 000774 | | CMP | #774, R0 | | | ; CHECK ADDRESS LESS THAN 772 |
| 4257 | 017754 | 002004 | | | BGE | BUSSE7 | | | |
| 4258 | | | | | | | | | |
| 4259 | 017756 | 004737 | 016632 | | JSR | PC, TYPOUT | | | |
| 4260 | 017762 | 013450 | | | OVAMSG | | | | ; ADDRESS EXCEEDS 772 |
| 4261 | 017764 | 000740 | | | BR | BUSSE4 | | | |
| 4262 | | | | | | | | | |
| 4263 | 017766 | 113701 | 014306 | BUSSE7: | MOVB | MASK, R1 | | | ; SET UP COUNT |
| 4264 | 017772 | 013702 | 014310 | | MOV | BASADD, R2 | | | ; SET UP ADDRESS BASE |
| 4265 | 017776 | 010022 | | BUS10: | MOV | R0, (R2)+ | | | ; SET UP ADDRESS |
| 4266 | 020000 | 022020 | | | CMP | (R0)+, (R0)+ | | | |
| 4267 | 020002 | 005301 | | | DEC | R1 | | | |
| 4268 | 020004 | 001374 | | | BNE | BUS10 | | | |
| 4269 | 020006 | 042737 | 100000 014306 | | BIC | #100000, MASK | | | |
| 4270 | | | | | | | | | |
| 4271 | 020014 | 113701 | 014307 | | MOVB | MASK+1, R1 | | | |
| 4272 | 020020 | 012537 | 014310 | | MOV | (R5)+, BASADD | | | ; GET BASE PRIORITY ADDRESS |
| 4273 | | | | | | | | | |
| 4274 | 020024 | 005737 | 014504 | | TST | LSIFLG | | | ; SINGLE INT. LEVEL PROCESSOR ? |
| 4275 | 020030 | 001403 | | | BEQ | 3\$ | | | ; IF NOT THEN BRANCH AWAY. |
| 4276 | 020032 | 012700 | 000004 | | MOV | #4, R0 | | | ; ELSE SET PRIORITY LEVEL AT 4 |
| 4277 | 020036 | 000416 | | | BR | BUS11B | | | ; THEN GO INSTALL IT. |
| 4278 | | | | | | | | | |
| 4279 | 020040 | 004737 | 016632 | 3\$: | JSR | PC, TYPOUT | | | |
| 4280 | 020044 | 013477 | | | PRMSG | | | | ; FIRST PRIORITY LEVEL IS... |
| 4281 | 020046 | 013702 | 014310 | | MOV | BASADD, R2 | | | |

| | | | | | | | | |
|------|--------|--------|--------|--------|-------------|------------|-----------|-------------------------------|
| 4282 | 020052 | 004737 | 015732 | | BUSS11: JSR | PC,OCTIN | | ;INPUT OCTAL PRIORITY |
| 4283 | 020056 | 005737 | 014276 | | | TST | RAND | |
| 4284 | 020062 | 001004 | | | | BNE | BUS11B | |
| 4285 | 020064 | 004737 | 016632 | | | JSR | PC,TYPOUT | |
| 4286 | 020070 | 013712 | | | | NODEFM | | |
| 4287 | 020072 | 000767 | | | | BR | BUSS11 | |
| 4288 | 020074 | 020027 | 000007 | | BUS11B: CMP | RO,#7 | | ;IS THE PRIORITY LEGAL ? |
| 4289 | 020100 | 101404 | | | | BLOS | 2\$ | ;YES |
| 4290 | 020102 | 004737 | 016632 | | | JSR | PC,TYPOUT | |
| 4291 | 020106 | 013625 | | | | BADPRI | | ;NO, RE-ENTER VALUE |
| 4292 | 020110 | 000760 | | | | BR | BUSS11 | |
| 4293 | 020112 | 042700 | 177770 | | 2\$: | BIC | #-10,RO | ;CLEAR UNWANTED BITS |
| 4294 | 020116 | 000300 | | | | SWAB | RO | |
| 4295 | 020120 | 006200 | | | | ASR | RO | |
| 4296 | 020122 | 006200 | | | | ASR | RO | |
| 4297 | 020124 | 006200 | | | | ASR | RO | |
| 4298 | 020126 | 010022 | | | | MOV | RO,(R2)+ | ;SET UP PRIORITY |
| 4299 | | | | | | | | |
| 4300 | 020130 | 005301 | | | | DEC | R1 | |
| 4301 | 020132 | 001412 | | | | BEQ | BUSS12 | |
| 4302 | | | | | | : | | |
| 4303 | 020134 | 005737 | 014504 | | | TST | LSIFLG | ;SINGLE INT.LEVEL PROCESSOR ? |
| 4304 | 020140 | 001403 | | | | BEQ | 4\$ | ;IF NOT THEN BRANCH AWAY |
| 4305 | 020142 | 012700 | 000004 | | | MOV | #4,RO | ;ELSE SET UP PRIORITY AS 4 |
| 4306 | 020146 | 000752 | | | | BR | BUS11B | ;AND GO INSTALL IT. |
| 4307 | | | | | | : | | |
| 4308 | 020150 | 004737 | 016632 | | 4\$: | JSR | PC,TYPOUT | |
| 4309 | 020154 | 013536 | | | | NPRMSG | | ;NEXT PRIORITY LEVEL IS.... |
| 4310 | 020156 | 000735 | | | | BR | BUSS11 | |
| 4311 | | | | | | | | |
| 4312 | 020160 | 012537 | 014306 | | BUSS12: MOV | (R5)+,MASK | | ; GET NEXT VECTOR PAIR |
| 4313 | 020164 | 001405 | | | | BEQ | BUSS13 | ; ZERO MEANS END |
| 4314 | 020166 | 004737 | 016632 | | | JSR | PC,TYPOUT | |
| 4315 | 020172 | 013144 | | | | NXTVMS | | ; ELSE PROMPT FOR NEXT GROUP |
| 4316 | 020174 | 000137 | 017662 | | | JMP | BUSS3A | |
| 4317 | 020200 | 010537 | 014306 | | BUSS13: MOV | R5,MASK | | |
| 4318 | 020204 | 012737 | 000006 | 000004 | | MOV | #6,4 | |
| 4319 | 020212 | 012737 | 000004 | 000006 | | MOV | #4,6 | |
| 4320 | | | | | | | | |
| 4321 | 020220 | 004737 | 015634 | | | JSR | PC,RSTREG | ;RESTORE REGISTER |
| 4322 | 020224 | 013705 | 014306 | | | MOV | MASK,R5 | |
| 4323 | 020230 | 000205 | | | | RTS | R5 | |
| 4324 | | | | | | | | |
| 4325 | | | | | | | | |
| 4326 | | | | | | | | |
| 4327 | | | | | | | | |
| 4328 | 020232 | 004737 | 016632 | | NXMTRP: JSR | PC,TYPOUT | | |
| 4329 | 020236 | 013575 | | | | NXMSG | | ;NON-EXISTANT ADDRESS |
| 4330 | 020240 | 005237 | 014312 | | | INC | TRPERR | |
| 4331 | 020244 | 000002 | | | | RTI | | |
| 4332 | | | | | | | | |
| 4333 | | | | | | | | |
| 4334 | | | | | | | | |
| 4335 | | | | | | | | |
| 4336 | | | | | | | | |

4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393

.SBTTL NUMBER GENERATOR

DESCRIPTION:

ROUTINE TO GENERATE DATA PATTERNS,
THE TYPE OF PATTERN IS SELECTED BY R3, AND THE
PATTERN GENERATED IS RETURNED IN R0 AND LOCATION
GOOD.

CALLING SEQUENCE:

JSR PC,GENER

INPUT PARAMETERS:

R3 CONTAINS THE PATTERN NUMBER

| | |
|------|---|
| R3=0 | ALL ZEROES |
| 1 | ALL ONES |
| 2 | 010101 ETC BIT PATTERN |
| 3 | 101010 ETC BIT PATTERN |
| 4 | ROTATING 1 IN A ZERO WORD |
| 5 | ROTATING 0 IN AN ALL ONE WORD |
| 6 | PSEUDO RANDOM NUMBER |
| 7 | INCREMENTING DATA PATTERN, GOOD CONTAINS THE VALUE TO BE UPDATED |

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

THE NUMBER GENERATED IS HELD IN
R0 AND GOOD.

IMPLICIT OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

POSSIBLE ERROR CODES:

NONE


```

4394
4395
4396
4397 020246 042703 177770
4398 020252 004737 015536
4399 020256 006303
4400 020260 000173 020264
4401 020264 020304
4402 020266 020310
4403 020270 020316
4404 020272 020324
4405 020274 020332
4406 020276 020342
4407 020300 020400
4408 020302 020520
4409 020304 005000
4410 020306 000507
4411 020310 005000
4412 020312 005100
4413 020314 000504
4414 020316 012700 052525
4415 020322 000501
4416 020324 012700 125252
4417 020330 000476
4418 020332 000241
4419 020334 004737 020354
4420 020340 000472
4421 020342 000241
4422 020344 004737 020354
4423 020350 005100
4424 020352 000465
4425 020354 006037 020376
4426 020360 001003
4427 020362 012737 100000 020376
4428 020370 013700 020376
4429 020374 000207
4430 020376 000001
4431 020400 012737 000005 014326
4432 020406 004737 020420
4433 020412 013700 014324
4434 020416 000443
4435 020420 013702 014324
4436 020424 001002
4437 020426 013702 014332
4438 020432 032737 000777 014326
4439 020440 001003
4440 020442 012737 000001 014326
4441 020450 013703 014326
4442 020454 013702 014324
4443 020460 033702 014330
4444 020464 001405
4445 020466 005102
4446 020470 033702 014330
4447 020474 001401
4448 020476 000402
4449 020500 000241

```

```

:
:
:
GENER:  BIC      #177770,R3
        JSR      PC,SAVREG
        ASL      R3
        JMP      @GENSEL(R3)
GENSEL: GEN0
        GEN1
        GEN52
        GEN25
        GENR1
        GENRO
        GENRAN
        GENINC
GENO:   CLR      RO
        BR       GENEX
GEN1:   CLR      RO
        COM     RO
        BR       GENEX
GEN52:  MOV      #52525,R0
        BR       GENEX
GEN25:  MOV      #125252,R0
        BR       GENEX
GENR1:  CLC
        JSR      PC,GENROT
        BR       GENEX
GENRO:  CLC
        JSR      PC,GENROT
        COM     RO
        BR       GENEX
GENROT: ROR      GENISH
        BNE     GENER1
        MOV     #100000,GENISH
        MOV     GENISH,R0
        RTS
GENISH: 1
GENRAN: MOV      #5,RANSEL
        JSR      PC,RANGEN
        MOV     RANDN,R0
        BR       GENEX
RANGEN: MOV     RANDN,R2
        BNE     RAN1
        MOV     RANST,R2
        BIT     #777,RANSEL
        BNE     RAN2
        MOV     #1,RANSEL
        MOV     RANSEL,R3
        MOV     RANDN,R2
        BIT     RANMTA,R2
        BEQ     RANCLC
        COM     R2
        BIT     RANMTA,R2
        BEQ     RANCLC
        BR       RANSEC
RANCLC: CLC

```

```

:ALL ZERO WORD
:ALL ONE WORD
:52 PATTERN
:25 PATTERN
:ROTATE '1' EACH CALL
:ROTATE '0' EACH CALL
:RANDOM NUMBER
:INCREMENTING COUNT
:0>R0
:NOT0>R0
:5252>R0
:125252>R0
:SHIFT 1 > R0
:SHIFT 0 > R0
:ROTATE 1 PATTERN
:= 0?
:YES, SET MSB
:PUT 1 IN R0
:AND EXIT
:SET SELECT VALUE TO 5
:GENERATE RANDOM NUMBER IN R0
:
:
:
:IS RANDOM = 0
:YES, PUT RANDOM START VALUE IN
:NO;IS RANSEL SELECT VALUE = 0
:NO
:YES: SET RANSEL = 1
:
:GET R2 <0 AND 1>
:

```

| | | | |
|------|--------|--------|--------|
| 4450 | 020502 | 000401 | |
| 4451 | 020504 | 000261 | |
| 4452 | 020506 | 006037 | 014324 |
| 4453 | 020512 | 005303 | |
| 4454 | 020514 | 001357 | |
| 4455 | 020516 | 000207 | |
| 4456 | 020520 | 013700 | 014334 |
| 4457 | 020524 | 005200 | |
| 4458 | 020526 | 010037 | 014334 |
| 4459 | 020532 | 010066 | 000002 |
| 4460 | 020536 | 004737 | 015634 |
| 4461 | 020542 | 000207 | |

| | | |
|---------|-----|-----------|
| | BR | RAN4 |
| RANSEC: | SEC | |
| RAN4: | ROR | RANDN |
| | DEC | R3 |
| | BNE | RAN2+4 |
| RANEX: | RTS | PC |
| GENINC: | MOV | GOOD,RO |
| | INC | RO |
| GENEX: | MOV | RO,GOOD |
| | MOV | RO,2(SP) |
| | JSR | PC,RSTREG |
| | RTS | PC |

:ROTATE C TO B15
:IS THIS NUMBER REQUIRED?
:NO, GET ANOTHER
:YES, EXIT
:INCREMENTS LOC. 'GOOD'

:LOAD RO MAINROUTINE

4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518

.SBTTL PRINT ERROR MESSAGES

DESCRIPTION:

ROUTINE TO PRINT ERROR MESSAGES
IF BIT 14 IN THE SWR IS SET NO MESSAGES WILL BE
PRINTED, IF BIT 15 IS SET THE PROGRAM WILL NO WAIT
AFTER AN ERROR HAS BEEN PRINTED. IT IS CALLED THUS:

JSR PC,ERROR
ARG

WHERE ARG CONTAINS THE ERROR CODE AND IS OF THE FORM
X+N, WHERE N IS THE ERROR NUMBER IN THE RANGE 0-177
AND X IS A COMBINATION OF FLAGS THAT INDICATE WHAT
VALUES ARE TO BE PRINTED. THESE VALUES SHOULD BE LOADED
BEFORE THE ERROR ROUTINE IS CALLED AND ARE DEFINED
AS FOLLOWS:

| FLAG SETTING | NUMBER | LOCATION | MESSAGE |
|--------------|--------|-----------|-------------|
| ----- | ----- | ----- | ----- |
| C | 4000 | CALLPC | CALLED FROM |
| S | 10000 | STATUS | STATUS |
| A | 20000 | ADDRES | ADDRESS |
| D | 40000 | DATA | DATA |
| G | 100000 | GOOD, BAD | GOOD= BAD = |

IN ADDITION THE ERROR NUMBER WILL BE COMBINED WITH
THE TEST NUMBER TO INDICATE IN WHICH TEST THE ERROR
OCCURRED.

AN ERROR COUNT IS MAINTAINED AN ON EACH ERROR THE
COUNT IS UPDATED. IF RUNNING UNDER A SOFTWARE SWITCH
REGISTER, IT IS POSSIBLE TO SELECT NEW OPTIONS

CALLING SEQUENCE:

JSR PC,ERROR

INPUT PARAMETERS:

THE LOACTION FOLLOWING THE CALL
CONTAINS THE ERROR CODE AND FLAG SETTINGS

IMPLICIT INPUT PARAMETERS:

NONE

OUTPUT PARAMETERS:

4519
 4520
 4521
 4522
 4523
 4524
 4525
 4526
 4527
 4528
 4529
 4530
 4531
 4532
 4533
 4534
 4535
 4536
 4537
 4538
 4539
 4540 020544 004737 015536
 4541 020550 005237 014346
 4542 020554 032777 040000 173436
 4543 020562 001133
 4544 020564 017637 000000 014350
 4545 020572 004737 016632
 4546 020576 014014
 4547 020600 013700 014216
 4548 020604 000300
 4549 020606 006300
 4550 020610 153700 014350
 4551 020614 004737 016722
 4552 020620 004737 016632
 4553 020624 014020
 4554 020626 011600
 4555 020630 004737 016722
 4556 020634 132737 000200 014351
 4557 020642 001416
 4558 020644 004737 016632
 4559 020650 014032
 4560 020652 013700 014334
 4561 020656 004737 016722
 4562 020662 004737 016632
 4563 020666 014043
 4564 020670 013700 014336
 4565 020674 004737 016722
 4566 020700 132737 000100 014351
 4567 020706 001407
 4568 020710 004737 016632
 4569 020714 014054
 4570 020716 013700 014340
 4571 020722 004737 016722
 4572 020726 132737 000040 014351
 4573 020734 001407
 4574 020736 004737 016632

.....
 NONE
 IMPLICIT OUTPUT PARAMETERS:
 THE APPROPRAITE ERROR MESSAGE WILL BE PRINTED
 IF PERMITTED.
 COMPLETION CODES:
 NONE
 POSSIBLE ERROR CODES:
 NONE
 15/11/78
 ERROR: JSR PC,SAVREG
 INC ERRDIS ;INC COUNT
 BIT #40000,@SWR ;SUPPRESS PRINTOUT
 BNE ERHALT ;YES
 MOV @(SP),ERRARG ;NO
 JSR PC,TYPOUT ;TYPE MESSAGE
 MSG1
 MOV TESTNO,RO
 SWAB RO
 ASL RO
 BISB ERRARG,RO
 JSR PC,PROCT
 JSR PC,TYPOUT
 MSG2
 MOV (SP),RO
 JSR PC,PROCT ;PRINT 6 DIGITS
 BITB #200,ERRARG+1
 BEQ ERRSV1
 JSR PC,TYPOUT
 MSG3
 MOV GOOD,RO
 JSR PC,PROCT
 JSR PC,TYPOUT
 MSG4
 MOV BAD,RO
 JSR PC,PROCT
 BITB #100,ERRARG+1 ;D SET ?
 BEQ ERRSV2 ;NO
 JSR PC,TYPOUT
 MSG5
 MOV DATA,RO
 JSR PC,PROCT
 BITB #40,ERRARG+1 ;A SET ?
 BEQ ERRSV3 ;NO
 JSR PC,TYPOUT

| | | | | | | | |
|------|--------|--------|--------|--------|--------------|--------------|---------------------------|
| 4575 | 020742 | 014065 | | | EMSG6 | | |
| 4576 | 020744 | 013700 | 014344 | | MOV | ADDRES,RO | |
| 4577 | 020750 | 004737 | 016722 | | JSR | PC,PROCT | |
| 4578 | 020754 | 132737 | 000020 | 014351 | ERRSV3: BITB | #20,ERRARG+1 | :S SET ? |
| 4579 | 020762 | 001407 | | | BEQ | ERRSV4 | :NO |
| 4580 | 020764 | 004737 | 016632 | | JSR | PC,TYPOUT | |
| 4581 | 020770 | 014103 | | | EMSG7 | | |
| 4582 | 020772 | 013700 | 014342 | | MOV | STATUS,RO | |
| 4583 | 020776 | 004737 | 016722 | | JSR | PC,PROCT | |
| 4584 | 021002 | 132737 | 000010 | 014351 | ERRSV4: BITB | #10,ERRARG+1 | :C SET |
| 4585 | 021010 | 001407 | | | BEQ | ERRSV5 | :NO |
| 4586 | 021012 | 004737 | 016632 | | JSR | PC,TYPOUT | |
| 4587 | 021016 | 014116 | | | EMSG8 | | |
| 4588 | 021020 | 013700 | 014352 | | MOV | CALLPC,RO | |
| 4589 | 021024 | 004737 | 016722 | | JSR | PC,PROCT | |
| 4590 | 021030 | 004737 | 016576 | | ERRSV5: JSR | PC,CRLF | |
| 4591 | 021034 | 004737 | 016632 | | JSR | PC,TYPOUT | |
| 4592 | 021040 | 014135 | | | EMSG9 | | |
| 4593 | 021042 | 013700 | 014346 | | MOV | ERRDIS,RO | |
| 4594 | 021046 | 004737 | 017064 | | JSR | PC,BASE10 | |
| 4595 | 021052 | 032777 | 100000 | 173140 | ERHALT: BIT | #100000,@SWR | |
| 4596 | 021060 | 001004 | | | BNE | NOHALT | |
| 4597 | 021062 | 013700 | 014346 | | MOV | ERRDIS,RO | :DISPLAY ERROR COUNT |
| 4598 | 021066 | 004737 | 015344 | | JSR | PC,MONIT | :GO TO SWR |
| 4599 | 021072 | 004737 | 015634 | | NOHALT: JSR | PC,RSTREG | |
| 4600 | 021076 | 062716 | 000002 | | ADD | #2,(SP) | |
| 4601 | 021102 | 000207 | | | RTS | PC | |
| 4602 | 021104 | 000000 | | | BUFF1: 0 | | |
| 4603 | | 001000 | | | .END | START | :PROGRAM END, SELF-START. |

| | | | | |
|----------------|----------------|----------------|---------------|---------------|
| A = 020000 | CHRFIN 016554 | GENER 020246 | MONITA 015436 | RANDC 014410 |
| ADDRES 014344 | CHSR 014230 | GENER1 020370 | MONITX 015444 | RANDN 014324 |
| ASK56 012164 | CHWRD 014314 | GENEX 020526 | NODEFM 013712 | RANEX 020516 |
| BAD 014336 | CLRBIT 002762 | GENINC 020520 | NOHALT 021072 | RANGEN 020420 |
| BADPRI 013625 | CNVFLG 014374 | GENISH 020376 | NOMEMA 013177 | RANMTA 014330 |
| BAMSG 013332 | COUNT1 004264 | GENRAN 020400 | NPRMSG 013536 | RANSEC 020504 |
| BASADD 014310 | COUNT2 004266 | GENROT 020354 | NXMADR 014362 | RANSEL 014326 |
| BASE1A 017102 | CRLF 016576 | GENRO 020342 | NXMSG 013575 | RANST 014332 |
| BASE1B 017116 | CSR 014222 | GENR1 020332 | NXMTRP 020232 | RAN1 020432 |
| BASE1C 017134 | D = 040000 | GENSEL 020264 | NXTVMS 013144 | RAN2 020450 |
| BASE1D 017114 | DATA 014340 | GENO 020304 | OCTIN 015732 | RAN4 020506 |
| BASE10 017064 | DBUF 014224 | GEN1 020310 | OCTMSG 014206 | RDYINT 011760 |
| BASE11 014170 | DECMMSG 014200 | GEN25 020324 | OCTNUM 014160 | READ 016104 |
| BASM10 017046 | EMSG1 014014 | GEN52 020316 | ODAMSG 013425 | REDMES 013247 |
| BAS10A 017072 | EMSG2 014020 | GETCH1 016134 | OVAMSG 013450 | REPCNT 014250 |
| BCCHAR 014322 | EMSG3 014032 | GETCH2 016146 | PARITY 014320 | REPC1= 001000 |
| BCOUNT 014414 | EMSG4 014043 | GETCH3 016260 | PAR0 = 172340 | REPC2= 000100 |
| BELL 014764 | EMSG5 014054 | GETCH4 016324 | PAR1 = 172342 | REPC3= 000002 |
| BELLMS 004276 | EMSG6 014065 | GETSTR 016130 | PAR2 = 172344 | REPRM1 006256 |
| BELLS 004274 | EMSG7 014103 | GOMSG 012272 | PAR3 = 172346 | REPROM 006162 |
| BELL1 014776 | EMSG8 014116 | GOOD 014334 | PAR4 = 172350 | RETRY 002474 |
| BINC1 011670 | EMSG9 014135 | HALF6V 012160 | PAR5 = 172352 | RSTART 001200 |
| BINC2 011750 | ENDIT 014664 | HALF8V 012162 | PAR6 = 172354 | RSTREG 015634 |
| BINC3 011716 | ERHALT 021052 | HSWR = 177570 | PAR7 = 172356 | RUBCHR 016402 |
| BITS 002476 | ERMES1 002500 | ILLCHR 016534 | PASMSG 013020 | RUBFLG 014406 |
| BLNFLG 012056 | ERMES2 002564 | ILLVEC 015024 | PCHR 016710 | R6 = %000006 |
| BLN1 012032 | ERMES3 002650 | ILVMSG 013036 | PDR0 = 172300 | R7 = %000007 |
| BLN2 012054 | ERMES4 002722 | INTFLG 004272 | PDR1 = 172302 | S = 010000 |
| BUFF 012060 | ERMS10 004361 | INTSRV 004206 | PDR2 = 172304 | SAVEXM 014364 |
| BUFF1 021104 | ERMS11 004427 | INTVEC 014242 | PDR3 = 172306 | SAVLT4 014370 |
| BUSSET 017510 | ERMS12 004473 | JM600 014252 | PDR4 = 172310 | SAVLT6 014372 |
| BUSSE1 017540 | ERMS13 004531 | LINDEL 016450 | PDR5 = 172312 | SAVPAR 014366 |
| BUSSE2 017612 | ERMS14 004602 | LINDL1 016470 | PDR6 = 172314 | SAVPC 014266 |
| BUSSE3 017626 | ERMS20 005662 | LINECH 016500 | PDR7 = 172316 | SAVPC1 014270 |
| BUSSE4 017666 | ERR 014316 | LKS = 177546 | PMSG1 016646 | SAVREG 015536 |
| BUSSE5 017732 | ERRARG 014350 | LOPPROM 006146 | PMSG2 016672 | SAVSTA 014272 |
| BUSSE6 017742 | ERRDIS 014346 | LOWCHR 014402 | PMSG3 016676 | SECOND 004270 |
| BUSSE7 017766 | ERROR 020544 | LSIFLG 014504 | PMSG4 016702 | SETBIT 002706 |
| BUSS1A 017604 | ERRSV1 020700 | L525 012156 | PRCT1A 016744 | SETBLN 012002 |
| BUSS10 017776 | ERRSV2 020726 | MASK 014306 | PRESET 011756 | SET56 012062 |
| BUSS11 020052 | ERRSV3 020754 | MODADM 013314 | PRMSG 013477 | SET56A 012126 |
| BUSS12 020160 | ERRSV4 021002 | MODADR 014302 | PRNT3 017014 | SILLS1 014442 |
| BUSS13 020200 | ERRSV5 021030 | MODIFY 017162 | PROCT 016722 | SRO = 177572 |
| BUSS3A 017662 | FADR 014754 | MODI1 017212 | PROCT1 016732 | SR1 = 177574 |
| BUS11B 020074 | FASTSW 015104 | MODI2 017236 | PROCT2 016746 | SR2 = 177576 |
| C = 004000 | FILL 014524 | MODI3 017256 | PROCT3 017000 | SSWR 014246 |
| CALLPC 014352 | FILL1 014556 | MODI4 017370 | PSW = 177776 | START 001000 |
| CAR 014226 | FILL2 014570 | MODI5 017426 | QEXIT 015462 | START1 001226 |
| CARX 014232 | FIRVMS 013110 | MODPRM 013325 | QEXIT1 015504 | START2 001232 |
| CARY 014234 | FRMSG 013070 | MODSAV 014304 | QEXIT2 015506 | START3 001266 |
| CA1170= 177746 | FSAPW 014412 | MODSPA 013321 | QEXIT3 015522 | STATUS 014342 |
| CHAR 014240 | FSTCNT 014256 | MODXIT 017446 | RANCLC 020500 | STBINC 011634 |
| CHDR 014236 | G = 100000 | MONIT 015344 | RAND 014276 | STRADD 014376 |

| | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|----------|
| STRLEN | 014400 | TSTB2 | 002322 | T1019 | 003764 | T4000 | 006276 | T5011 | 010336 |
| SWR | 014220 | TSTB3 | 002326 | T1019A | 004012 | T4001 | 006316 | T5012 | 010342 |
| SWRMSG | 013302 | TSTB4 | 002462 | T1020 | 004040 | T4002 | 006330 | T5013 | 010404 |
| SWRSET | 014506 | TYPCTC | 015454 | T1021 | 004056 | T4002A | 006320 | T5014 | 011614 |
| TABLE | 001272 | TYPD1 | 014300 | T1022 | 004120 | T4003 | 006400 | T5015 | 010636 |
| TABLE1 | 001316 | TYPOTA | 014274 | T1023 | 004126 | T4004 | 006446 | T5015A | 010552 |
| TESMSG | 013007 | TYPOTB | 015736 | T2000 | 004726 | T4005 | 006476 | T5016 | 010654 |
| TESTNO | 014216 | TYPOTC | 016016 | T2001 | 004746 | T4006 | 006536 | T5017 | 010716 |
| TESTR | 014416 | TYPOTD | 016052 | T2002 | 005026 | T4007 | 006610 | T5018 | 010754 |
| TESTO | 001332 | TYPOTE | 016070 | T2003 | 005032 | T4008 | 006662 | T5018A | 011000 |
| TEST1 | 002776 | TYPOUT | 016632 | T2003A | 005036 | T4009 | 006734 | T5018B | 011004 |
| TEST2 | 004710 | T0000 | 001350 | T2004 | 005042 | T4010 | 006776 | T5019 | 011044 |
| TEST3 | 005734 | T0001 | 001370 | T2004A | 005056 | T4015 | 007026 | T5020 | 011106 |
| TEST4 | 006260 | T0002 | 001454 | T2005 | 005146 | T4016 | 007066 | T5020A | 011142 |
| TEST5 | 007676 | T0003 | 001460 | T2006 | 005172 | T4017 | 007140 | T5020B | 011152 |
| TIME1 | 004230 | T0004 | 001564 | T2007 | 005200 | T4018 | 007212 | T5021 | 011250 |
| TIME1A | 004262 | T0005 | 001570 | T2008 | 005270 | T4019 | 007264 | T5022 | 011276 |
| TKB = | 177562 | T0007 | 002170 | T2009 | 005314 | T4020 | 007326 | T5023 | 011302 |
| TKS = | 177560 | T0008 | 002204 | T2010 | 005320 | T4021 | 007364 | T5024 | 011342 |
| TPB = | 177566 | T10WW | 004132 | T2011 | 005334 | T4022 | 007474 | T5025 | 011376 |
| TPREDY | 015074 | T1000 | 003014 | T2012 | 005420 | T4023 | 007524 | T5026 | 011402 |
| TPS = | 177564 | T1001 | 003034 | T2013 | 005444 | T4024 | 007634 | T5027 | 011446 |
| TRAPSV | 015124 | T1002 | 003146 | T2014 | 005450 | T50WY | 011630 | T5028 | 011506 |
| TRPARG | 014260 | T1003 | 003242 | T2015 | 005510 | T5000 | 007714 | T5029 | 011512 |
| TRPBAK | 015332 | T1004 | 003250 | T2016 | 005600 | T5001 | 007746 | T5030 | 011552 |
| TRPERR | 014312 | T1008 | 003254 | T2017 | 005624 | T5002 | 007770 | UPPCHR | 014404 |
| TRPLP | 015266 | T1009 | 003364 | T2018 | 005646 | T5003 | 007774 | VAMSG | 013366 |
| TRPMEM | 014264 | T1012 | 003370 | T3000 | 005752 | T5004 | 010102 | VECLEV | 014244 |
| TRPSCP | 015242 | T1013 | 003500 | T3001 | 005772 | T5005 | 010106 | VECTOR | 014516 |
| TRPSEL | 014262 | T1014 | 003504 | T3002 | 006014 | T5006 | 010164 | WAIT | 004156 |
| TRXEXM | 014360 | T1015 | 003544 | T3003 | 006044 | T5007 | 010210 | WMSG | 012736 |
| TRXPAR | 014356 | T1016 | 003562 | T3004 | 006060 | T5007A | 010214 | WTCO | 004204 |
| TRXVAD | 014354 | T1017 | 003636 | T3005 | 006104 | T5008 | 010216 | WTC1 | 004202 |
| TSTBIT | 002210 | T1017A | 003700 | T3006 | 006132 | T5009 | 010272 | SENDAD | 014740 |
| TSTB1 | 002232 | T1018 | 003716 | T40WW | 007662 | T5010 | 010332 | . | = 021106 |

. ABS. 021106 C00

ERRORS DETECTED: 0

CVVTAA.BIN, CVVTAA.SEQ=CVVTAA.SRC
 RUN-TIME: 13 26 1 SECONDS
 RUN-TIME RATIO: 172/41=4.1
 CORE USED: 11K (21 PAGES)