

135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OF THE LISTING.

6.2 ERROR DATA

PC LISTING ADDRESS WHERE THE ERROR WAS DETECTED.
TSTNUM TEST NUMBER WHERE THE ERROR OCCURRED.
BUSADRS DISPLAY BUS ADDRESS WHERE THE RESULTANT DATA WAS EXPECTED
EXPCT DATA THAT WAS EXPECTED.
RCVD DATA THAT WAS RECEIVED.

7.0 MISCELLANEOUS

7.1 VS60 BUS/VECTOR ADDRESS MODIFICATION

MODIFY LOCATION 'SVECT!' IF BASE VECTOR ADDRESS IS NOT 320.
MODIFY LOCATION 'SBSAE' IF BASE BUS ADDRESS IS NOT 172000.

NOTE: A RESTART IS REQUIRED AFTER THE ABOVE ADDRESS MODIFICATION.
THE ABOVE LOCATIONS ARE LOCATED IN THE 'APT MAILBOX-ETABLE'

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP.
THIS DIAGNOSTIC INCLUDES THE 'APT' SOFTWARE HOOKS HOWEVER, THEY HAVE NOT BEEN TESTED.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT WHICH TIME THE PROGRAM IS RESTARTED AT THE BEGINNING.

7.4 SINGLE VS60 TESTING

THIS DIAGNOSTIC DOES NOT TEST MULTIPLE VS60'S.

179
180
181
182
183
184
185
186
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
2348.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 3 SECONDS WITH NO ITERATIONS TO ABOUT 2 MINUTES WITH ITERATIONS ENABLED. AN 'END PASS' MESSAGE IS TYPED EACH PASS THRU THE DIAGNOSTIC.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING CONTAINS A BRIEF DESCRIPTION OF EACH TEST. IT WILL ALSO IDENTIFY WHAT MAINTENANCE MODE A PARTICULAR TEST USES. THE COMMENT FIELD IN EACH TEST ALSO CAN BE HELPFUL IN UNDERSTANDING A TEST.

9.2 TYPICAL TEST SEQUENCE

MOST TESTS ARE ORGANIZED AROUND THE FOLLOWING FORMAT: A MAINTENANCE MODE SWITCH IS SET (SW1, 2 OR 3). THEN A VS60 REGISTER BUS REGISTER ADDRESS IS LOADED INTO LOCATION '\$BDADR' WHICH IS WHERE THE EXPECTED DATA WILL BE READ. NEXT STEP WILL BE TO LOAD THE EXPECTED DATA INTO LOCATION '\$GDDAT'. THEN ONE OR MORE INSTRUCTIONS ARE LOADED STARTING AT LOCATION 'BUFFER'. THE DISPLAY PC IS THEN LOADED WITH ADDRESS 'BUFFER' WHICH CAUSES AN NPR TO 'BUFFER'. AT THE COMPLETION OF THE 'NPR' A NUMBER OF RESUMES COULD BE ISSUED IF ADDITIONAL 'NPRS' ARE REQUIRED FOR INSTRUCTIONS OR DATA. AT THIS TIME THE CONDITION UNDER TEST SHOULD HAVE BEEN SET UP AND AVAILABLE AT THE BUS ADDRESS SET UP IN '\$BDADR'. THIS CONDITION IS USUALLY LOADED INTO LOCATION '\$BDDAT' AND COMPARED TO THE EXPECTED RESULTS PREVIOUSLY SET UP IN '\$GDDAT'. IF AN ERROR IS DETECTED THEN AN ERROR TRAP (EMT) IS PERFORMED ACCORDING TO THE ITEM NUMBER IN THE '\$ERRTB' (ERROR TABLE). THIS TABLE DIRECTS OR POINTS TO THE PERTINENT 'TYPE OUT' INFORMATION.

10.0 LISTING

```
%
.*TITLE MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
.*COPYRIGHT (C) 1976
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY R. MOORE
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-B1), AUG 19, 1975.
.*
```

.SBTTL BASIC DEFINITIONS

```
.*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
PS= 177776 ;;PROCESSOR STATUS WORD
```

001100

177776

```

235 .EQUIV PS,PSW
236 177774 STKLM= 177774 ::STACK LIMIT REGISTER
237 177772 PIRQ= 177772 ::PROGRAM INTERRUPT REQUEST REGISTER
238 177570 DSWR= 177570 ::HARDWARE SWITCH REGISTER
239 177570 DDISP= 177570 ::HARDWARE DISPLAY REGISTER
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
290

```

000000
 000001
 000002
 000003
 000004
 000005
 000006
 000007

*GENERAL PURPOSE REGISTER DEFINITIONS
 R0= %0 ::GENERAL REGISTER
 R1= %1 ::GENERAL REGISTER
 R2= %2 ::GENERAL REGISTER
 R3= %3 ::GENERAL REGISTER
 R4= %4 ::GENERAL REGISTER
 R5= %5 ::GENERAL REGISTER
 R6= %6 ::GENERAL REGISTER
 R7= %7 ::GENERAL REGISTER
 .EQUIV R6,SP ::STACK POINTER
 .EQUIV R7,PC ::PROGRAM COUNTER

*PRIORITY LEVEL DEFINITIONS
 PR0= 0 ::PRIORITY LEVEL 0
 PR1= 40 ::PRIORITY LEVEL 1
 PR2= 100 ::PRIORITY LEVEL 2
 PR3= 140 ::PRIORITY LEVEL 3
 PR4= 200 ::PRIORITY LEVEL 4
 PR5= 240 ::PRIORITY LEVEL 5
 PR6= 300 ::PRIORITY LEVEL 6
 PR7= 340 ::PRIORITY LEVEL 7

*"SWITCH REGISTER" SWITCH DEFINITIONS
 SW15= 100000
 SW14= 40000
 SW13= 20000
 SW12= 10000
 SW11= 4000
 SW10= 2000
 SW09= 1000
 SW08= 400
 SW07= 200
 SW06= 100
 SW05= 40
 SW04= 20
 SW03= 10
 SW02= 4
 SW01= 2
 SW00= 1
 .EQUIV SW09,SW9
 .EQUIV SW08,SW8
 .EQUIV SW07,SW7
 .EQUIV SW06,SW6
 .EQUIV SW05,SW5
 .EQUIV SW04,SW4
 .EQUIV SW03,SW3
 .EQUIV SW02,SW2
 .EQUIV SW01,SW1
 .EQUIV SW00,SW0

20

20

335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390

.SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
12	HALT AT END OF DIAGNOSTIC
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>

.SBTTL TRAP CATCHER

```

.=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
.=174
DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER

```

.SBTTL STARTING ADDRESS(ES)

```

JMP      @#BEGIN          ;;JUMP TO STARTING ADDRESS OF PROGRAM
;*****

```

.SBTTL ACT11 HOOKS

```

;HOOKS REQUIRED BY ACT11
$SVPC=.          ;SAVE PC
.=46
$ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
.=52
.WORD 0          ;;2)SET LOC.52 TO ZERO
.$SVPC          ;; RESTORE PC
.=1000

```

.SBTTL APT PARAMETER BLOCK

```

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.$X=.          ;;SAVE CURRENT LOCATION
.=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200          ;;FOR APT START UP
.=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR       ;;POINT TO APT HEADER BLOCK
.=.$X        ;;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

```

```

$APTHD:
$HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.

```

391 001002 001200
392 001004 000012
393 001006 000036
394 001010 000000
395 001012 000052
396
397
398
399
400 000204 000204 000137 002506
401

\$MBADR: .WORD \$MAIL ;; ADDRESS OF APT MAILBOX (BITS 0-15)
\$STSM: .WORD 10. ;; RUN TIM OF LONGEST TEST
\$PASTM: .WORD 30. ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 0 ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
 .WORD SETEND-\$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)

;RESTART ADDRESS

 =204
JMP @#RESTRT ;RESTART GOOD AFTER START 200

[Handwritten scribble]

[Handwritten mark]

448
449
450
451
452
453
454
455
456 001200
457 001200 000000
458 001202 000000
459 001204 000000
460 001206 000000
461 001210 000000
462 001212 000000
463 001214 000000
464 001216 000000
465 001220
466 001220 000
467 001221 000
468 001222 000000
469 001224 000000
470 001226 000000
471
472
473
474
475
476
477 001230 000
478 001231 000
479
480
481
482
483 001232 000000
484
485 001234 000
486 001235 000
487 001236 000000
488 001240 000
489 001241 000
490 001242 000000
491 001244 000
492 001245 000
493 001246 000000
494 001250 320
495 001251 000
496 001252 004
497 001253 000
498
499 001254 172000
500 001256 000000
501 001260 000000
502 001262 000000
503 001264 000000

```

*****
*****

```

.SBTTL APT MAILBOX-ETABLE

```

.EVEN
SMAIL: .EVEN
SMMSGTY: .WORD AMMSGTY : APT MAILBOX
SFATAL: .WORD AFATAL : MESSAGE TYPE CODE
STESTN: .WORD ATESTN : FATAL ERROR NUMBER
SPASS: .WORD APASS : TEST NUMBER
SDEVCT: .WORD ADEVCT : PASS COUNT
SUNIT: .WORD AUNIT : DEVICE COUNT
SMSGAD: .WORD AMSGAD : I/O UNIT NUMBER
SMSGLG: .WORD AMSGLG : MESSAGE ADDRESS
SETABLE: : MESSAGE LENGTH
SENV: .BYTE AENV : APT ENVIRONMENT TABLE
SENV: .BYTE AENV : ENVIRONMENT BYTE
SSWREG: .WORD ASWREG : ENVIRONMENT MODE BITS
SUSWR: .WORD AUSWR : APT SWITCH REGISTER
SCPUOP: .WORD ACPUOP : USER SWITCHES
: * : CPU TYPE, OPTIONS
: * : BITS 15-11=CPU TYPE
: * : 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
: * : 11/70=06, PDQ=07, Q=10
: * : BIT 10=REAL TIME CLOCK
: * : BIT 9=FLOATING POINT PROCESSOR
: * : BIT 8=MEMORY MANAGEMENT
SMAMS1: .BYTE AMAMS1 : HIGH ADDRESS, M.S. BYTE
SMTYP1: .BYTE AMTYP1 : MEM. TYPE, BLK#1
: * : MEM. TYPE BYTE -- (HIGH BYTE)
: * : 900 NSEC CORE=001
: * : 300 NSEC BIPOLAR=002
: * : 500 NSEC MOS=003
SMADR1: .WORD AMADR1 : HIGH ADDRESS, BLK#1
: * : MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
SMAMS2: .BYTE AMAMS2 : HIGH ADDRESS, M.S. BYTE
SMTYP2: .BYTE AMTYP2 : MEM. TYPE, BLK#2
SMADR2: .WORD AMADR2 : MEM. LAST ADDRESS, BLK#2
SMAMS3: .BYTE AMAMS3 : HIGH ADDRESS, M.S. BYTE
SMTYP3: .BYTE AMTYP3 : MEM. TYPE, BLK#3
SMADR3: .WORD AMADR3 : MEM. LAST ADDRESS, BLK#3
SMAMS4: .BYTE AMAMS4 : HIGH ADDRESS, M.S. BYTE
SMTYP4: .BYTE AMTYP4 : MEM. TYPE, BLK#4
SMADR4: .WORD AMADR4 : MEM. LAST ADDRESS, BLK#4
SVECT1: .BYTE AVECT1 : INTERRUPT VECTOR#1
SVECT2: .BYTE AVECT2 : INTERRUPT VECTOR#2
SPRIOR: .BYTE APRIOR : BUS PRIORITY #1, #2
: .BYTE 0 : SPARE, NOT USED
: .EVEN
$BASE: .WORD ABASE : BASE ADDRESS OF EQUIPMENT UNDER TEST
SDEV: .WORD ADEV : DEVICE MAP
SCDW1: .WORD ACDW1 : CONTROLLER DESCRIPTION WORD#1
SCDW2: .WORD ACDW2 : CONTROLLER DESCRIPTION WORD#2
SDDWO: .WORD ADDWO : DEVICE DESCRIPTOR WORD#0

```


560
561
562
563
564
565
566
567
568
569
570
571
572
573

001354 025745
001356 027444
001360 027512
001362 000000

001364 025770
001366 027444
001370 027512
001372 000000

; ITEM 4

EM4
DH1
DT1
0

: SEQUENCING DPC REGISTER
: ERRPC TSTNUM BUSADR EXPCT RCVD
: SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT

; ITEM 5

EMS
DH1
DT1
0

: LOADING LINE TYPE REGISTER
: ERRPC TSTNUM BUSADR EXPCT RCVD
: SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT

Handwritten mark

6074	001374	026016	EM6	:LOADING INTENSITY LEVEL REGISTER
6075	001376	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6076	001400	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6077	001402	000000	0	
6078				
6079				
6080				
6081				
6082				
6083				
6084	001404	026052	EM7	:LOADING GRAPHLOT INCREMENT REGISTER
6085	001406	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6086	001410	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6087	001412	000000	0	
6088				
6089				
6090				
6091	001414	026104	EM10	:DISPLAY JUMP ABSOLUTE TO AN ADDRESS
6092	001416	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6093	001420	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6094	001422	000000	0	
6095				
6096				
6097				
6098	001424	026140	EM11	:DISPLAY JUMP RELATIVE TO AN ADDRESS
6099	001426	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6100	001430	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6101	001432	000000	0	
6102				
6103				
6104	001434	026174	EM12	:DISPLAY JSR ABSOLUTE TO AN ADDRESS
6105	001436	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6106	001440	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6107	001442	000000	0	
6108				
6109				
6110				
6111	001444	026227	EM13	:DISPLAY JSR RELATIVE TO AN ADDRESS
6112	001446	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6113	001450	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6114	001452	000000	0	
6115				
6116				
6117				
6118	001454	026262	EM14	:LOADING X POSITION REGISTER
6119	001456	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
6120	001460	027512	DT1	:SERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
6121	001462	000000	0	


```

823
824
825
826 002034 027526
827 002036 027530
828 002040 027532
829 002042 027534
830 002044 027536
831 002046 027540
832 002050 000000
833 002052 000000
834 002054 000000
835 002056 000000
836 002060 000000
837 002062 000020
838
839
840
841 002064 172000
842 002066 172002
843 002070 172004
844 002072 172006
845 002074 172010
846 002076 172012
847 002100 172014
848 002102 172016
849 002104 172020
850 002106 172022
851 002110 172024
852 002112 172026
853 002114 172030
854 002116 172032
855 002120 172034
856 002122 172036
857
858 002124 000320
859 002126 000322
860
861 002130 000324
862 002132 000326
863
864 002134 000330
865 002136 000332
866
867 002140 000334
868 002142 000336

```

;COMMON PROGRAM TAGS AND STORAGE LOCATIONS
DBUF: BUFFER ;FIRST WORD IN THE DISPLAY BUFFER
DBUF1: BUFFER+2 ;SECOND WORD
DBUF2: BUFFER+4 ;THIRD WORD
DBUF3: BUFFER+6 ;FOURTH WORD
DBUF4: BUFFER+10 ;FIFTH WORD
DBUF5: BUFFER+12 ;SIXTH WORD
DSAVE: 0 ;TEMP REG.
DSAVE1: 0
SIZE: 0 ;BUFFER SIZE
CNTR: 0
TSTNUM: 0 ;CONTAINS TEST NO. ON ER
DELAY: 20 ;COUNT USED FOR GENERAL PURPOSE DELAYS

;VS60 ADDRESSES AND VECTORS POINTERS
DPC: 172000 ;DISPLAY PC REGISTER
SREG0: 172002 ;DISPLAY STATUS REGISTER
XPOS: 172004 ;X AXIS REGISTER AND GRAPHLOT REG <READ ONLY>
YPOS: 172006 ;Y AXIS REGISTER AND CHARACTER REG <READ ONLY>
RLO: 172010 ;RELOCATION REGISTER
SREG1: 172012 ;MISC STATUS REGISTER #1
XDOFF: 172014 ;X DYNAMIC OFFSET REGISTER
YDOFF: 172016 ;Y DYNAMIC OFFSET REGISTER
ANAME: 172020 ;ASSOCIATIVE NAME REGISTER
CONS: 172022 ;CONSOLE STATUS REGISTER
DNAME: 172024 ;DISPLAY NAME REGISTER
STKVAL: 172026 ;VALUE OF CURRENT STACK WORD
TERMCH: 172030 ;TERMINATE CHARACTER REGISTER
STKPT: 172032 ;STACK POINTER REGISTER
ZPOS: 172034 ;Z POSITION REGISTER <READ ONLY>
ZDOFF: 172036 ;Z DYNAMIC OFFSET REGISTER
DDONE: 320 ;DISPLAY STOP <DONE> VECTOR
DDONE1: 322
LPVCT: 324 ;DISPLAY LIGHT PEN VECTOR
LPVCT1: 326
TIMEVT: 330 ;DISPLAY TIME-OUT <NXM.> ERROR VECTOR
TMEVT1: 332 ;OR "SHIFT-OUT" VECTOR
NAMEVT: 334 ;NAME MATCH VECTOR
NAMEV1: 336

```

869                                     ;SOFTWARE INITIALIZATION ROUTINE
870
871 002144                               BEGIN:
872 002144 012706 001100                 MOV    #SCMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
873 002150 005026                         CLR    (R6)+              ;;CLEAR MEMORY LOCATION
874 002152 022706 001126                 CMP    #SBDDAT,R6        ;;DONE?
875 002156 001374                         BNE    -6                 ;;LOOP BACK IF NO
876 002160 012706 001100                 MOV    #STACK,SP        ;;SETUP THE STACK POINTER
877 002164 012737 023202 000020         MOV    #SCOPE, @IOTVEC   ;;IOT VECTOR FOR SCOPE ROUTINE
878 002172 012737 000340 000022         MOV    #340, @IOTVEC+2  ;;LEVEL 7
879 002200 012737 023726 000030         MOV    #ERROR, @EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
880 002206 012737 000340 000032         MOV    #340, @EMTVEC+2  ;;LEVEL 7
881 002214 012737 025520 000034         MOV    #TRAP, @TRAPVEC  ;;TRAP VECTOR FOR TRAP CALLS
882 002222 012737 000340 000036         MOV    #340, @TRAPVEC+2;LEVEL 7
883 002230 012737 024736 000024         MOV    #SPWRDN, @PWRVEC ;;POWER FAILURE VECTOR
884 002236 012737 000340 000026         MOV    #340, @PWRVEC+2  ;;LEVEL 7
885 002244 013737 023046 023040         MOV    SENDCT, $EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
886 002252 005037 001164                 CLR    $TIMES            ;;INITIALIZE NUMBER OF ITERATIONS
887 002256 005037 001166                 CLR    $ESCAPE          ;;CLEAR THE ESCAPE ON ERROR ADDRESS
888 002262 112737 000001 001115         MOV    #1, $SERMAX      ;;ALLOW ONE ERROR PER TEST
889 002270 012737 002270 001106         MOV    #., $LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
890 002276 012737 002276 001110         MOV    #., $LPERR       ;;SETUP THE ERROR LOOP ADDRESS
891 002304 013746 000004                 MOV    @#4, -(SP)       ;;SAVE ERROR VECTOR
892 002310 013746 000006                 MOV    @#6, -(SP)
893 002314 012737 002330 000004         MOV    #15, 4          ;;SET UP TIMEOUT VECTOR
894 002322 005777 176610                 TST    @SWR             ;;TRY TO REFERENCE HARDWARE SWR
895 002326 000407                         BR     2$               ;;BRANCH IF NO TIMEOUT TRAP OCCURS
896 002330 012737 000176 001136 1$:     MOV    #SWREG, SWR      ;;POINT TO SOFTWARE SWITCH REG.
897 002336 012737 000174 001140         MOV    #DISPREG, DISPLAY;POINT TO SOFTWARE DISPLAY REG.
898 002344 022626                         CMP    (SP)+, (SP)+     ;;RESTORE STACK
899 002346 012637 000006 2$:           MOV    (SP)+, @#6      ;;RESTORE ERROR VECTOR
900 002352 012637 000004                 MOV    (SP)+, @#4
901 002356 005037 001206                 CLR    $PASS           ;;CLEAR PASS COUNT
902 002362 132737 000200 001221         BITB   #APTSIZE, $ENVM  ;;TEST USER SIZE UNDER APT
903 002370 001403                         BEQ    3$               ;;YES, USE NON-APT SWITCH
904 002372 012737 001222 001136         MOV    #$$SWREG, SWR   ;;NO, USE APT SWITCH REGISTER
905 002400
906 002400 012700 002064 3$:           SETUP1: MOV    #DPC, R0    ;;SET UP VS60 REG ADRS POINTERS
907 002404 013701 001254                 MOV    $BASE, R1       ;;GET VS60 BASE ADRS FROM APT MAILBOX-ETABLE
908 002410 010120                 SETUP2: MOV    R1, (0)+
909 002412 062701 000002                 ADD    #2, R1
910 002416 022700 002124                 CMP    #DDONE, R0      ;;TEST FOR LAST ADDRESS
911 002422 001372                         BNE    SETUP2
912 002424 012700 002124                 MOV    #DDONE, R0      ;;SET UP VS60 VECTOR ADRS POINTERS
913 002430 013701 001250                 MOV    $VECT1, R1     ;;GET VS60 BASE VECTOR ADRS FROM APT MAILBOX-ETABLE
914 002434 010120                 SETUP3: MOV    R1, (0)+
915 002436 062701 000002                 ADD    #2, R1
916 002442 022700 002144                 CMP    #BEGIN, R0     ;;TEST FOR LAST VECTOR
917 002446 001372                         BNE    SETUP3
918 002450 004737 023134                 JSR    PC, RSTVEC     ;;GO SET UP VS60 VECTOR ADR'S WITH HLTS
919 002454 004737 025140                 JSR    PC, $SIZE      ;;DETERMINE LAST MEM LOCATION
920 002460 013737 025232 002054         MOV    $LSTAD, SIZE    ;;LOAD LAST MEM ADDRESS
921 002466 162737 006000 002054         SUB    #6000, $SIZE   ;;SAVE LOADERS AND XXDP
922 002474 005737 000042                 TST    @#42           ;;TEST IF RUNNING CHAIN MODE UNDER XXDP
923 002500 001002                         BNE    RESTRT         ;;BR IF SO
924 002502 104400                         TYPE

```

925	002504	025554			HEADER			
926	002506	012737	000340	177776	RESTR:	MOV	#340,PSW	:SET PRIORITY TO HIGHEST LEVEL
927	002514	012706	001100			MOV	#STACK,SP	:ALWAYS SET UP STACK POINTER
928	002520	000005				RESET		:START TESTS WITH VS60 INITIALIZED
929	002522	105037	001102			CLRB	\$STNM	:RESET TEST NUMBER

```

930
931
932
933 002526 000004
934 002530 012737 002564 001110
935 002536 012737 002604 000004
936 002544 012737 000340 000006
937 002552 013737 001254 001122
938 002560 012700 000020
939 002564 005777 176332
940 002570 062737 000002 001122
941 002576 005300
942 002600 001371
943 002602 000402
944 002604 022626
945 002606 104016
946 002610 012737 000006 000004
947 002616 005037 000006
948
949
950
951
952 002622 000004
953 002624 012737 002646 001110
954 002632 013737 002114 001122
955 002640 012737 000100 001124
956 002646 013700 001124
957 002652 052700 000200
958 002656 010077 177232
959 002662 017737 177226 001126
960 002670 023737 001124 001126
961 002676 001401
962 002700 104017
963 002702 006237 001124
964 002706 001357
965
966
967
968
969 002710 000004
970 002712 012737 002740 001110
971 002720 013737 002114 001122
972 002726 012737 000077 001124
973 002734 012700 177677
974 002740 010077 177150
975 002744 017737 177144 001126
976 002752 023737 001124 001126
977 002760 001401
978 002762 104017
979 002764 006200
980 002766 006237 001124
981 002772 052737 000100 001124
982 003000 103757
983
984
985
;*****
;*TEST 1 SEE IF ALL VS60 REGS ARE THERE
;*****
TST1: SCOPE
MOV #1$,SLPERR ;SET UP SCOPE LOOP ADRS
MOV #2$,J#ERRVEC ;SET UP TIME OUT TRAP
MOV #340,J#ERRVEC+2 ;SET UP TIME OUT PRIORITY
MOV $BASE,$BDADR ;PUT 1ST VS60 BUS ADRS IN LOC $BDADR
MOV #20,R0 ;SET UP REG COUNT
1$: TST J$BDADR ;LOOK FOR VS60 REG
ADD #2,$BDADR ;OK-ADVANCE TO NEXT
DEC R0 ;HAVE WE TRIED ALL REGS?
BNE 1$ ;BR IF NOT
BR 3$ ;GO RESTORE LOCS 4 & 6
2$: CMP (R6)+,(R6)+ ;FIX STK SINCE NO RTI
ERROR 16 ;REG ADRS TIME OUT ER
3$: MOV #ERRVEC+2,J#ERRVEC ;POINT TO HALT IN ADRS 6
CLR J#ERRVEC+2 ;SET UP HALT
;*****
;*TEST 2 FLOAT A ONE THRU TERMINATE CHARACTER REG
;*****
TST2: SCOPE
MOV #1$,SLPERR ;SET UP SCOPE LOOP ADRS
MOV TERMCH,$BDADR ;SET UP TERMCH ADRS
MOV #100,$GDDAT ;EXPECT 100 INITIALLY
1$: MOV $GDDAT,R0 ;PUT PATTERN IN R0
BIS #200,R0 ;SET THE ENABLE BIT
MOV R0,JTERMCH ;LOAD UP TERMCH REG
MOV JTERMCH,$BDDAT ;READ IT BACK
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 2$ ;BR IF OK
ERROR 17 ;TERMINATE CHARACTER READ/WRITE ER
2$: ASR $GDDAT ;SHIFT ONE BIT RIGHT
BNE 1$ ;BR IF NOT DONE
;*****
;*TEST 3 FLOAT A ZERO THRU TERMINATE CHARACTER REG
;*****
TST3: SCOPE
MOV #1$,SLPERR ;SET UP SCOPE LOOP ADRS
MOV TERMCH,$BDADR ;SET UP TERMCH ADRS
MOV #77,$GDDAT ;EXPECT 77 INITIALLY
MOV #177677,R0 ;PUT PATTERN IN R0
1$: MOV R0,JTERMCH ;LOAD UP TERMCH REG - SEND HIGH BITS FOR NOISE
MOV JTERMCH,$BDDAT ;READ IT BACK
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 2$ ;BR IF OK
ERROR 17 ;TERMINATE CHARACTER READ/WRITE ER
2$: ASR R0 ;SHIFT ZERO BIT RIGHT
ASR $GDDAT ;SHIFT PATTERN IN COMPARE LOC
BIS #100,$GDDAT ;SET MSB
BCS 1$ ;BR IF NOT DONE
;*****
;*TEST 4 TEST TERMINATE CHARACTER REG IS NOT WRITEABLE WHEN CHANGE ENA=0
;*****

```

K02

MAINDEC-11-DZVSA-A
DZVSA.P11

VS60 INSTRUCTION TEST PART 1
T4 TEST TERMINATE CHARACTER REG IS NOT WRITEABLE WHEN CHANGE ENA=0

MACY11 27(732) 20-SEP-76 09:57 PAGE 23

```

986
987 003002 000004
988 003004 013737 002114 001122
989 003012 012737 000177 001124
990 003020 012777 177777 177066
991 003026 005077 177062
992 003032 017737 177056 001126
993 003040 023737 001124 001126
994 003046 001401
995 003050 104017

```

```

*****
TST4: SCOPE
MOV TERMCH,$BDADR ;SET UP TERMCH ADRS
MOV #177,$GDDAT ;SET UP DATA PATTERN
MOV #177777,@TERMCH ;LOAD UP ALL WRITEABLE BITS
CLR @TERMCH ;TRY TO CLEAR ALL BITS
MOV @TERMCH,$BDDAT ;READ IT BACK
CMP $GDDAT,$BDDAT ;SEE IF ANY BITS WERE CLEARED
BEQ TST5 ;ADVANCE TO NEXT TEST IF OK
ERROR 17 ;TERMINATE CHARACTER ENABLE CHANGE ER

```

```

996
997
998
999

```

```

*****
;*TEST 5 FLOAT A ONE THRU Y DYNAMIC OFFSET REG
*****
TST5: SCOPE

```

```

1000 003052 000004
1001 003054 012737 003076 001110
1002 003062 013737 002102 001122
1003 003070 012737 004000 001124
1004 003076 013777 001124 176776
1005 003104 017737 176772 001126
1006 003112 042737 170000 001126
1007 003120 023737 001124 001126
1008 003126 001401
1009 003130 104020
1010 003132 006237 001124
1011 003136 001357

```

```

MOV #1$, $LPERR ;SET UP SCOPE LOOP ADRS
MOV YDOFF,$BDADR ;SET UP YDOFF ADRS
MOV #4000,$GDDAT ;SET UP MSB TO A ONE
1$: MOV $GDDAT,@YDOFF ;LOAD UP YDOFF REG
MOV @YDOFF,$BDDAT ;READ IT BACK
BIC #170000,$BDDAT ;SAVE ONLY OFFSET VALUE
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 2$ ;BR IF OK
ERROR 20 ;Y OFFSET READ/WRITE ER
2$: ASR $GDDAT ;SHIFT ONE BIT RIGHT
BNE 1$ ;BR IF NOT DONE

```

```

1012
1013
1014
1015

```

```

*****
;*TEST 6 FLOAT A ZERO THRU Y DYNAMIC OFFSET REG
*****
TST6: SCOPE

```

```

1016 003140 000004
1017 003142 012737 003162 001110
1018 003150 013737 002102 001122
1019 003156 012700 173777
1020 003162 010037 001124
1021 003166 042737 170000 001124
1022 003174 010077 176702
1023 003200 017737 176676 001126
1024 003206 042737 170000 001126
1025 003214 023737 001124 001126
1026 003222 004101
1027 003224 104020
1028 003226 006200
1029 003230 103754
1030 003232 005077 176644

```

```

MOV #1$, $LPERR ;SET UP SCOPE LOOP ADRS
MOV YDOFF,$BDADR ;SET UP YDOFF ADRS
MOV #173777,R0 ;SET UP MSB TO A ZERO
1$: MOV R0,$GDDAT ;SAVE PATTERN FOR COMPARE
BIC #170000,$GDDAT ;GET RID OF HIGH BITS THAT ARE NOT WRITEABLE
MOV R0,@YDOFF ;LOAD UP YDOFF REG - SEND HIGH BITS FOR NOISE
MOV @YDOFF,$BDDAT ;READ IT BACK
BIC #170000,$BDDAT ;SAVE ONLY OFFSET VALUE
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 2$ ;BR IF OK
ERROR 20 ;Y OFFSET READ/WRITE ER
2$: ASR R0 ;SHIFT ZERO BIT RIGHT
BCS 1$ ;BR IF NOT DONE
CLR @YDOFF ;INSURE 0-Y OFFSET BEFORE ADVANCING

```

```

1031
1032
1033
1034

```

```

*****
;*TEST 7 FLOAT A ONE THRU X DYNAMIC OFFSET REG
*****
TST7: SCOPE

```

```

1035 003236 000004
1036 003240 012737 003262 001110
1037 003246 013737 002100 001122
1038 003254 012737 004000 001124
1039 003262 013777 001124 176610
1040 003270 017737 176604 001126
1041 003276 042737 170000 001126

```

```

MOV #1$, $LPERR ;SET UP SCOPE LOOP ADRS
MOV XDOFF,$BDADR ;SET UP XDOFF ADRS
MOV #4000,$GDDAT ;SET UP MSB TO A ONE
1$: MOV $GDDAT,@XDOFF ;LOAD UP XDOFF REG
MOV @XDOFF,$BDDAT ;READ IT BACK
BIC #170000,$BDDAT ;SAVE ONLY OFFSET VALUE

```


M02

```

1098 003576 104021          ERROR 21          ;X OFFSET POLARITY READ/WRITE ER
1099 003600 005037 001124 2$: CLR $GDDAT ;EXPECT ZERO
1100 003604 162737 020000 001124 SUB #20000,$GDDAT ;ADVANCE TO RESET STATE
1101 003612 100355          BPL 1$          ;BR IF NOT YET TESTED
1102
1103 ;*****
1104 ;*TEST 13  FLOAT A ONE THRU RELOCATE REG
1105 ;*****
1106 003614 000004          TST13: SCOPE
1107 003616 012737 003640 001110 MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS
1108 003624 013737 002074 001122 MOV RLO,$BDADR ;SET UP RLO ADRS
1109 003632 012737 004000 001124 MOV #4000,$GDDAT ;SET UP MSB TO A ONE
1110 003640 013777 001124 176226 1$: MOV $GDDAT,$RLO ;LOAD UP RLO REG
1111 003646 017737 176222 001126 MOV $RLO,$BDDAT ;READ IT BACK
1112 003654 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
1113 003662 001401          BEQ 2$          ;BR IF OK
1114 003664 104022          ERROR 22         ;RELOCATE REG READ/WRITE ER
1115 003666 006237 001124 2$: ASR $GDDAT ;SHIFT ONE BIT RIGHT
1116 003672 001362          BNE 1$          ;BR IF NOT DONE
1117
1118 ;*****
1119 ;*TEST 14  FLOAT A ZERO THRU RELOCATE REG
1120 ;*****
1121 003674 000004          TST14: SCOPE
1122 003676 012737 003716 001110 MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS
1123 003704 013737 002074 001122 MOV RLO,$BDADR ;SET UP RLO ADRS
1124 003712 012700 173777          MOV #173777,R0 ;SET MSB TO A ZERO
1125 003716 010037 001124 1$: MOV R0,$GDDAT ;SAVE PATTERN FOR COMPARE
1126 003722 042737 170000 001124 BIC #170000,$GDDAT ;GET RID OF HIGH BITS THAT DO NOT READ/WRITE
1127 003730 010077 176140          MOV R0,$RLO ;LOAD UP RLO REG - SEND HIGH BITS FOR NOISE
1128 003734 017737 176134 001126 MOV $RLO,$BDDAT ;READ IT BACK
1129 003742 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
1130 003750 001401          BEQ 2$          ;BR IF OK
1131 003752 104022          ERROR 22         ;RELOCATE REG READ/WRITE ER
1132 003754 006200 2$: ASR R0 ;SHIFT ZERO BIT RIGHT
1133 003756 103757          BCS 1$          ;BR IF NOT DONE
1134 003760 005077 176110          CLR $RLO ;CLR RELOCATE REG BEFORE ADVANCING
1135
1136 ;*****
1137 ;*TEST 15  TEST THAT ALL MAINT TYPE BITS ARE READ/WRITEABLE
1138 ;*****
1139 003764 000004          TST15: SCOPE
1140 003766 013737 002116 001122 MOV STKPT,$BDADR ;SET UP REG 32 ADRS
1141 003774 012737 004010 001110 MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS
1142 004002 012737 170000 001124 MOV #170000,$GDDAT ;EXPECT ALL BITS INITIALLY
1143 004010 013777 001124 176100 1$: MOV $GDDAT,$STKPT ;LOAD UP MAINT BITS
1144 004016 017737 176074 001126 MOV $STKPT,$BDDAT ;READ THEM BACK
1145 004024 042737 007777 001126 BIC #7777,$BDDAT ;SAVE ONLY MAINT SWITCHES
1146 004032 023737 001124 001126 CMP $GDDAT,$BDDAT ;ARE THEY CORRECT?
1147 004040 001401          BEQ 2$          ;BR IF SO
1148 004042 104001          ERROR 1 ;MAINT SWITCH READ/WRITE ERROR
1149 004044 162737 010000 001124 2$: SUB #10000,$GDDAT ;GO TO NEXT PATTERN
1150 004052 100356          BPL 1$          ;BR IF ALL STATES NOT TESTED
1151
1152 ;*****
1153 ;*TEST 16  TEST THAT RESET WILL CLEAR THE RELOCATE REG
    
```

```

1154
1155 004054 000004
1156 004056 012737 000010 001164
1157 004064 013737 002074 001122
1158 004072 005037 001124
1159 004076 012777 007777 175770
1160 004104 000005
1161 004106 017737 175762 001126
1162 004114 001401
1163 004116 104022
1164
1165
1166
1167
1168 004120 000004
1169 004122 012737 000010 001164
1170 004130 013737 002100 001122
1171 004136 005037 001124
1172 004142 012777 007777 175730
1173 004150 000005
1174 004152 017737 175722 001126
1175 004160 001401
1176 004162 104021
1177
1178
1179
1180
1181 004164 000004
1182 004166 012737 000010 001164
1183 004174 013737 002102 001122
1184 004202 005037 001124
1185 004206 012777 007777 175666
1186 004214 000005
1187 004216 017737 175660 001126
1188 004224 001401
1189 004226 104020
1190
1191
1192
1193
1194 004230 000004
1195 004232 012737 000040 001164
1196 004240 013737 002122 001122
1197 004246 005037 001124
1198 004252 012777 020000 175620
1199 004260 012777 020000 175614
1200 004266 000005
1201 004270 017737 175626 001126
1202 004276 042737 037777 001126
1203 004304 023737 001124 001126
1204 004312 001401
1205 004314 104032
1206
1207
1208
1209

```

```

*****
TST16: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV RLO,$BDADR ;:SET UP REG 10 ADRS
CLR $GDDAT ;:EXPECT ALL ZEROS
MOV #7777,@RLO ;:LOAD IT UP
RESET ;:CLR IT OUT
MOV @RLO,$BDDAT ;:READ IT
BEQ TST17 ;:GO TO NEXT TEST IF CLEARED
ERROR 22 ;:RESET FAILED TO CLR RELOCATE REG

```

```

*****
*TEST 17 TEST THAT RESET WILL CLEAR THE X DYNAMIC OFFSET REG
*****

```

```

TST17: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV XDOFF,$BDADR ;:SET UP REG 14 ADRS
CLR $GDDAT ;:EXPECT ALL ZEROS
MOV #7777,@XDOFF ;:LOAD IT UP
RESET ;:CLR IT OUT
MOV @XDOFF,$BDDAT ;:READ IT
BEQ TST20 ;:GO TO NEXT TEST IF CLEARED
ERROR 21 ;:RESET FAILED TO CLR X DYNAMIC OFFSET REG

```

```

*****
*TEST 20 TEST THAT RESET WILL CLEAR THE Y DYNAMIC OFFSET REG
*****

```

```

TST20: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV YDOFF,$BDADR ;:SET UP REG 16 ADRS
CLR $GDDAT ;:EXPECT ALL ZEROS
MOV #7777,@YDOFF ;:LOAD IT UP
RESET ;:CLR IT OUT
MOV @YDOFF,$BDDAT ;:READ IT
BEQ TST21 ;:GO TO NEXT TEST IF CLEARED
ERROR 20 ;:RESET FAILED TO CLR Y DYNAMIC OFF SET REG

```

```

*****
*TEST 21 TEST THAT RESET WILL CLEAR X & Y OFFSET POLARITY BITS
*****

```

```

TST21: SCOPE
MOV #40,$TIMES ;:DO 40 ITERATIONS
MOV ZDOFF,$BDADR ;:SET UP REG 36 ADRS
CLR $GDDAT ;:EXPECT ZERO
MOV #20000,@XDOFF ;:SET X POLARIYT BIT
MOV #20000,@YDOFF ;:SET Y POLARITY BIT
RESET ;:CLR THEM
MOV @ZDOFF,$BDDAT ;:READ REG 36
BIC #37777,$BDDAT ;:SAVE ONLY POLARITY BITS
CMP $GDDAT,$BDDAT ;:CORRECT?
BEQ TST22 ;:NEXT TEST IF SO
ERROR 32 ;:RESET FAILED TO CLR OFFSET POLARIY BITS

```

```

*****
*TEST 22 TEST THAT RESET WILL CLEAR TERMINATE CHARACTER REG
*****

```

```

1210 004316 000004 TST22: SCOPE
1211 004320 012737 000010 001164 MOV #10, $TIMES ;; DO 10 ITERATIONS
1212 004326 013737 002114 001122 MOV TERMCH, $BDADR ;; SET UP REG 30 ADRS
1213 004334 005037 001124 CLR $GDDAT ;; EXPECT ALL ZEROS
1214 004340 012777 000377 175546 MOV #377, $TERMCH ;; LOAD IT UP
1215 004346 000005 RESET ;; CLR IT OUT
1216 004350 017737 175540 001126 MOV $TERMCH, $BDDAT ;; READ IT
1217 004356 001401 BEQ TST23 ;; GO TO NEXT TEST IF CLEARED
1218 004360 104017 ERROR 17 ;; RESET FAILED TO CLR TERMINATE CHAR REG
1219
1220 *****
1221 ;*TEST 23 TEST THAT RESET CLEARS ALL MAINT BITS
1222 *****
1223 TST23: SCOPE
1224 004362 000004
1225 004364 012737 000100 001164 MOV #100, $TIMES ;; DO 100 ITERATIONS
1226 004372 013737 002116 001122 MOV STKPT, $BDADR ;; SET UP REG ADRS 32
1227 004400 005037 001124 CLR $GDDAT ;; LOAD EXPECTED
1228 004404 012777 170000 175504 MOV #170000, $STKPT ;; LOAD REGISTER
1229 004412 000005 RESET ;; INITILIZE
1230 004414 017737 175476 001126 MOV $STKPT, $BDDAT ;; READ REG.
1231 004422 042737 003777 001126 BIC #3777, $BDDAT ;; MASK OUT OTHER BITS
1232 004430 001401 BEQ TST24 ;; BR IF ALL CLEARED
1233 004432 104032 ERROR 32 ;; RESET FAILED TO CLEAR MAINT. BITS
1234
1235 *****
1236 ;*TEST 24 TEST THAT THE STACK SELECTION BITS ARE READ/WRITEABLE
1237 *****
1238 TST24: SCOPE
1239 004434 000004
1240 004436 012737 004460 001110 MOV #15, $LPERR ;; SET UP SCOPE LOOP ADRS
1241 004444 013737 002116 001122 MOV STKPT, $BDADR ;; SET UP REG ADRS 32
1242 004452 012737 000037 001124 MOV #37, $GDDAT ;; EXPECT ALL BITS INITIALLY
1243 004460 013777 001124 175430 15: MOV $GDDAT, $STKPT ;; SEND PATTERN OUT
1244 004466 017737 175424 001126 MOV $STKPT, $BDDAT ;; READ IT BACK
1245 004474 042737 177740 001126 BIC #177740, $BDDAT ;; SAVE ONLY STACK SELECTION BITS
1246 004502 023737 001124 001126 CMP $GDDAT, $BDDAT ;; CORRECT?
1247 004510 001401 BEQ 25 ;; BR IF SO
1248 004512 104051 ERROR 51 ;; STACK POINTER READ/WRITE ER
1249 004514 005337 001124 25: DEC $GDDAT ;; ADVANCE PATTERN
1250 004520 100357 BPL 15 ;; BR IF ALL PATTERNS NOT TESTED
1251
1252 *****
1253 ;*TEST 25 TEST THAT 'TOP OF STACK' AND RESET CLEARS STACK SELECTION BITS
1254 *****
1255 TST25: SCOPE
1256 004522 000004
1257 004524 012737 000040 001164 MOV #40, $TIMES ;; DO 40 ITERATIONS
1258 004532 013737 002116 001122 MOV STKPT, $BDADR ;; SET UP REG ADRS 32
1259 004540 012737 000041 001124 MOV #41, $GDDAT ;; LOAD EXPECTED
1260 004546 012777 000037 175342 MOV #37, $STKPT ;; SET ALL BITS
1261 004554 012777 000077 175334 MOV #77, $STKPT ;; CLR ALL BITS
1262 004562 017737 175330 001126 MOV $STKPT, $BDDAT ;; READ STACK SELECTION BITS
1263 004570 042737 177700 001126 BIC #177700, $BDDAT ;; SAVE SEL BITS ONLY
1264 004576 023737 001124 001126 CMP $GDDAT, $BDDAT ;; CORRECT?
1265 004604 001401 BEQ 15 ;; BR IF SO
1266 004606 104051 ERROR 51 ;; 'TOP OF STACK' FAILED TO CLR BITS
1267 004610 012737 000040 001124 15: MOV #40, $GDDAT ;; LD EXPECTED
1268 004616 012777 000037 175272 MOV #37, $STKPT ;; SET ALL BITS

```

```

1266 004624 000005          RESET          ;RESET CLRS THEM & SET 'TOP OF STACK'
1267 004626 017737 175264 001126      MOV          $STKPT,$BDDAT ;READ THEM
1268 004634 023737 001124 001126      CMP          $GDDAT,$BDDAT ;CORRECT?
1269 004642 001401          BEQ          TST26        ;NEXT TEST IF SO
1270 004644 104051          ERROR        51          ;RESET FAILED TO SET UP 'TOP OF STACK'

```

```

1271
1272          .SBTTL
1273          .SBTTL THESE TESTS USE MAINT SWITCH #1
1274          .SBTTL
1275

```

```

1276          ;*****
1277          ;*TEST 26      FLOAT A 1 ACROSS THE D.P.C.
1278          ;*****

```

```

1279 004646 000004          †TST26:  SCOPE
1280 004650 012737 000040 001164      MOV          #40,$TIMES    ;;DO 40 ITERATIONS
1281 004656 013737 002064 001122      MOV          DPC,$BDDADR  ;;SET UP REG ADRS 00
1282 004664 012737 000002 001124      MOV          #BIT1,$GDDAT ;LOAD EXPECTED
1283 004672 012777 010000 175216      MOV          #BIT12,$STKPT ;SET MAINT SWITCH #1
1284 004700 013777 001124 175156      1S:  MOV          $GDDAT,$DPC ;LOAD D.P.C.
1285 004706 017737 175152 001126      MOV          $DPC,$BDDAT  ;READ D.P.C. INTO $BDDAT
1286 004714 023737 001124 001126      CMP          $GDDAT,$BDDAT ;COMPARE VALUE EXPECTED TO READ
1287 004722 001402          BEQ          2S          ;BR IF SAME
1288 004724 104002          ERROR        2          ;FAILED TO LOAD DPC WITH A FLOATING 1
1289 004726 000403          BR          TST27
1290 004730 006337 001124      2S:  ASL          $GDDAT    ;CHANGE DATA EXPECTED
1291 004734 103361          BCC         1S          ;BR IF NOT COMPLETED
1292
1293

```

```

1294          ;*****
1295          ;*TEST 27      TEST THAT THE RELOCATE REG WILL CORRECTLY SET UP THE DPC
1296          ;*****

```

```

1296 004736 000004          †TST27:  SCOPE
1297 004740 012737 000040 001164      MOV          #40,$TIMES    ;;DO 40 ITERATIONS
1298 004746 012737 005006 001110      MOV          #1S,$LPERR   ;SET UP SCOPE LOOP ADRS
1299 004754 012777 010000 175134      MOV          #BIT12,$STKPT ;SET MAINT SWITCH #1
1300 004762 013737 002064 001122      MOV          DPC,$BDDADR  ;SET UP REG 00 ADRS
1301 004770 005077 175070          CLR          $DPC        ;CLEAR DPC
1302 004774 012737 000100 001124      MOV          #100,$GDDAT  ;EXPECT 100 INITIALLY
1303 005002 012700 000001          MOV          #1,$RO       ;START WITH 1
1304 005006 010077 175062      1S:  MOV          $RO,$ALO    ;LOAD RELOCATE REG
1305 005012 017737 175046 001126      MOV          $DPC,$BDDAT  ;READ RELOCATE REG - SHOULD BE TIMES 64
1306 005020 023737 001124 001126      CMP          $GDDAT,$BDDAT ;IS IT CORRECT?
1307 005026 001401          BEQ          2S          ;BR IF SO
1308 005030 104022          ERROR        22         ;RELOCATE FAILED TO SET UP DPC CORRECTLY
1309 005032 062737 000100 001124      2S:  ADD          #100,$GDDAT ;SET UP NEXT EXPECTED VALUE
1310 005040 001402          BEQ          TST30       ;GO TO NEXT TEST IF ALL VALUES TESTED
1311 005042 005200          INC         $RO         ;SET UP NEXT PATTERN
1312 005044 000760          BR          1S          ;GO TRY IT
1313
1314

```

```

1315          ;*****
1316          ;*TEST 30      TEST THAT RESET CLEARS THE D.P.C.
1317          ;*****

```

```

1317 005046 000004          †TST30:  SCOPE
1318 005050 012737 000100 001164      MOV          #100,$TIMES  ;;DO 100 ITERATIONS
1319 005056 013737 002064 001122      MOV          DPC,$BDDADR  ;SET UP REG ADRS 00
1320 005064 012777 010000 175024      MOV          #BIT12,$STKPT ;SET MAINT SWITCH #1
1321 005072 005037 001124          CLR          $GDDAT      ;LOAD EXPECTED

```

```

1322 005076 012777 177776 174760      MOV      #2,DPDC      ;LOAD REGISTER
1323 005104 000005                    RESET              ;INITILIZE
1324 005106 017737 174752 001126      MOV      DPDC,$BDDAT ;READ D.P.C. REG.
1325 005114 001401                    BEQ      TST31       ;:BR IF ALL BITS CLEARED
1326 005116 104032                    ERROR     32         ;RESET FAILED TO CLEAR THE D.P.C.

```

```

.SBTTL
.SBTTL THESE TESTS USE MAINT. SWITCH 2
.SBTTL

```

```

;*****
;*TEST 31      LOAD #17 INTO THE "MODE" REGISTERS
;*****

```

```

1334 TST31: SCOPE
1335 005120 000004                    MOV      SREG0,$BADR ;SET UP REG ADRS 02
1336 005122 013737 002066 001122      MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
1337 005130 012777 020000 174760      MOV      #174000,BUFFER ;LOAD MODE REGISTER=17
1338 005136 012737 174000 027526      MOV      DBUF,DPDC    ;LOAD DISPLAY PC.
1339 005144 013777 002034 174712      MOV      #74000,$GDDAT ;LOAD EXPECTED
1340 005152 012737 074000 001124      MOV      $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
1341 005160 017737 174702 001126      BIC      #103777,$BDDAT ;MASK TO BITS 14-11
1342 005166 042737 103777 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
1343 005174 023737 001124 001126      BEQ      TST32       ;:BR IF EQUAL
1344 005202 001401                    ERROR     3          ;MODE BITS (14-11) FAILED TO SET
1345 005204 104003

```

```

;*****
;*TEST 32      LOAD #10 INTO THE "MODE" REGISTERS
;*****

```

```

1349 TST32: SCOPE
1350 005206 000004                    MOV      SREG0,$BADR ;SET UP REG ADRS 02
1351 005210 013737 002066 001122      MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
1352 005216 012777 020000 174672      MOV      #140000,BUFFER ;LOAD MODE REGISTER=10
1353 005224 012737 140000 027526      MOV      DBUF,DPDC    ;LOAD DISPLAY P.C.
1354 005232 013777 002034 174624      MOV      #40000,$GDDAT ;LOAD EXPECTED
1355 005240 012737 040000 001124      MOV      $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
1356 005246 017737 174614 001126      BIC      #103777,$BDDAT ;MASK TO BITS 14-11
1357 005246 017737 174614 001126      CMP      #103777,$BDDAT ;COMPARE EXPCT TO RCVD
1358 005254 042737 103777 001126      BEQ      TST33       ;:BR IF EQUAL
1359 005262 023737 001124 001126      ERROR     3          ;MODE BIT 14 FAILED TO SET
1360 005270 001401
1361 005272 104003

```

```

;*****
;*TEST 33      LOAD #14 INTO THE "MODE" REGISTERS
;*****

```

```

1362 TST33: SCOPE
1363 005274 000004                    MOV      SREG0,$BADR ;SET UP REG ADRS 02
1364 005276 013737 002066 001122      MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
1365 005304 012777 020000 174604      MOV      #160000,BUFFER ;LOAD MODE REGISTER=14
1366 005312 012737 160000 027526      MOV      DBUF,DPDC    ;LOAD DISPLAY P.C.
1367 005320 013777 002034 174536      MOV      #60000,$GDDAT ;LOAD EXPECTED
1368 005326 012737 060000 001124      MOV      $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
1369 005334 017737 174526 001126      BIC      #103777,$BDDAT ;MASK TO BITS 14-11
1370 005342 042737 103777 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
1371 005350 023737 001124 001126      BEQ      TST34       ;:BR IF EQUAL
1372 005356 001401                    ERROR     3          ;MODE BIT 13 FAILED TO SET
1373 005360 104003
1374
1375
1376
1377

```

1378
1379
1380
1381 005362 000004
1382 005364 013737 002066 001122
1383 005372 012777 020000 174516
1384 005400 012737 170000 027526
1385 005406 013777 002034 174450
1386 005414 012737 070000 001124
1387 005422 017737 174440 001126
1388 005430 042737 103777 001126
1389 005436 023737 001124 001126
1390 005444 001401
1391 005446 104003
1392
1393
1394
1395 005450 000004
1396 005452 013737 002064 001122
1397 005460 012777 020000 174430
1398 005466 012700 027526
1399 005472 012720 164000
1400 005476 012720 164000
1401 005502 012720 164000
1402 005506 012720 164000
1403 005512 013777 002034 174344
1404 005520 012737 027530 001124
1405 005526 017737 174332 001126
1406 005534 023737 001124 001126
1407 005542 001402
1408 005544 104004
1409 005546 000451
1410 005550 005277 174310 1S:
1411 005554 062737 000002 001124
1412 005562 017737 174276 001126
1413 005570 023737 001124 001126
1414 005576 001402
1415 005600 104004
1416 005602 000433
1417 005604 2S:
1418 005604 005277 174254
1419 005610 062737 000002 001124
1420 005616 017737 174242 001126
1421 005624 023737 001124 001126
1422 005632 001402
1423 005634 104004
1424 005636 000415
1425 005640 3S:
1426 005640 005277 174220
1427 005644 062737 000002 001124
1428 005652 017737 174206 001126
1429 005660 023737 001124 001126
1430 005666 001401
1431 005670 104004
1432
1433

```
*****  
: *TEST 34 LOAD #16 INTO THE "MODE" REGISTERS  
*****  
TST34: SCOPE  
MOV SREGO, $B0ADR ;SET UP REG ADRS 02  
MOV #BIT13, $STKPT ;SET MAINT SWITCH #2  
MOV #170000, BUFFER ;LOAD MODE REGISTER=16  
MOV DBUF, $DPC ;LOAD DISPLAY P.C.  
MOV #70000, $GDDAT ;LOAD EXPECTED  
MOV $SREGO, $BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #103777, $BDDAT ;MASK TO BITS 14-11  
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST35 ;: BR IF EQUAL  
ERROR 3 ;MODE BIT 12 FAILED TO SET  
*****  
: *TEST 35 SIMPLE INCREMENT D.P.C. TEST  
*****  
TST35: SCOPE  
MOV DPC, $B0ADR ;SET UP REG ADRS 00  
MOV #BIT13, $STKPT ;SET MAINT SWITCH #2  
MOV #BUFFER, R0 ;LOAD BUFFER POINTER  
MOV #164000, (R0)+ ;LOAD DISPLAY NOP  
MOV #164000, (R0)+  
MOV #164000, (R0)+  
MOV #164000, (R0)+  
MOV DBUF, $DPC ;START THE DISPLAY  
MOV #BUFFER+2, $GDDAT ;LOAD EXPECTED  
MOV $DPC, $BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT, $BDDAT ;TEST FOR INCREMENT  
BEQ 1S ;DISPLAY P.C. FAILED TO INCREMENT  
ERROR 4  
BR TST36 ;  
1S: INC $DPC ;STEP THE DISPLAY P.C.  
ADD #2, $GDDAT ;UPDATE EXPECTED  
MOV $DPC, $BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT, $BDDAT ;TEST FOR INCREMENT  
BEQ 2S ;  
ERROR 4 ;DISPLAY P.C. FAILED TO INCREMENT  
BR TST36 ;  
2S: INC $DPC ;SINGLE STEP THE DISPLAY  
ADD #2, $GDDAT ;UPDATE EXPECTED  
MOV $DPC, $BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT, $BDDAT ;TEST FOR INCREMENT  
BEQ 3S ;  
ERROR 4 ;DISPLAY P.C. FAILED TO INCREMENT  
BR TST36 ;  
3S: INC $DPC ;SINGLE STEP THE DISPLAY  
ADD #2, $GDDAT ;UPDATE EXPECTED  
MOV $DPC, $BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT, $BDDAT ;TEST FOR INCREMENT  
BEQ TST36 ;: BR IF EQUAL  
ERROR 4 ;DISPLAY P.C. FAILED TO INCREMENT  
*****
```

```

1434 ;*TEST 36 EXPANDED D.P.C. INCREMENT TEST
1435 ;*****
1436 005672 000004 TST36: SCOPE
1437 005674 012737 000040 001164 MOV #40,$TIMES ;DO 40 ITERATIONS
1438 005702 013737 002064 001122 MOV DPC,$B0ADR ;SET UP REG ADRS 00
1439 005710 012777 020000 174200 MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
1440 005716 013702 002034 MOV DBUF,R2 ;SET UP POINTER
1441 005722 012722 164000 1S: MOV #164000,(R2)+ ;MOVE DNOP INTO THE BUFFER
1442 005726 023702 002054 CMP SIZE,R2 ;FINISHED FILLING THE BUFFER?
1443 005732 001373 BNE 1S ;NO
1444 005734 013777 002034 174122 MOV DBUF,$DPC ;YES, START THE DISPLAY
1445 005742 012737 027526 001124 MOV #BUFFER,$GDDAT ;LOAD EXPECTED
1446 005750 013702 002054 MOV SIZE,R2 ;SETUP A COUNT
1447 005754 024242 CMP -(R2),-(R2) ;DEC BY 2
1448 005756 062737 000002 001124 2S: ADD #2,$GDDAT ;UPDATE EXPECTED
1449 005764 017737 174074 001126 MOV $DPC,$BDDAT ;READ DISPLAY P.C.
1450 005772 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID IT INCREMENT BY 2?
1451 006000 001402 BEQ 3S ;YES
1452 006002 104004 ERROR 4 ;DISPLAY PC FAILED TO INCREMENT
1453 006004 000406 BR TST37 ;
1454
1455 006006 020237 001124 3S: CMP R2,$GDDAT ;FINISHED THE BUFFER
1456 006012 001403 BEQ TST37 ;BR TO NEXT TEST
1457 006014 005277 174044 INC $DPC ;SINGLE STEP THE DISPLAY
1458 006020 000756 BR 2S ;TRY AGAIN
1459
1460 ;*****
1461 ;*TEST 37 TEST THAT THE NAME INSTR CAN FLOAT A ONE THRU DPU NAME REG
1462 ;*****
1463 006022 000004 TST37: SCOPE
1464 006024 012777 020000 174064 MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
1465 006032 012737 006046 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
1466 006040 012737 002000 001124 MOV #2000,$GDDAT ;SET UP MSB
1467 006046 012737 150000 027526 1S: MOV #150000,BUFFER ;SET UP INSTR LOC
1468 006054 053737 001124 027526 BIS $GDDAT,BUFFER ;SET NAME BIT AT INSTR LOC
1469 006062 012777 027526 173774 MOV #BUFFER,$DPC ;START
1470 006070 013737 002110 001122 MOV DNAME,$B0ADR ;SET UP REG ADRS 24
1471 006076 017737 174006 001126 MOV $DNAME,$BDDAT ;READ NAME REG
1472 006104 042737 174000 001126 BIC #174000,$BDDAT ;ONLY INTERESTED IN THE NAME BITS
1473 006112 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID IT GET LOADED PROPERLY?
1474 006120 001401 BEQ 2S ;BR IF OK
1475 006122 104023 ERROR 23 ;DPU NAME INSTR ER
1476 006124 006237 001124 2S: ASR $GDDAT ;SHIFT 'ONE' BIT RIGHT
1477 006130 103346 BCC 1S ;BR IF MORE BIT POSITIONS
1478
1479 ;*****
1480 ;*TEST 40 TEST THAT THE NAME INSTR CAN FLOAT A ZERO THRU DPU NAME REG
1481 ;*****
1482 006132 000004 TST40: SCOPE
1483 006134 012777 020000 173754 MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
1484 006142 012737 006154 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
1485 006150 012700 175777 MOV #175777,R0 ;SET MSB TO A ZERO
1486 006154 010037 001124 1S: MOV R0,$GDDAT ;PUT PATTERN IN COMPARE LOC
1487 006160 042737 174000 001124 BIC #174000,$GDDAT ;GET RID OF EXTRA BITS
1488 006166 012737 150000 027526 MOV #150000,BUFFER ;SET UP INSTR
1489 006174 053737 001124 027526 BIS $GDDAT,BUFFER ;SET UP PATTERN AT INSTR LOC

```


1490	006202	012777	027526	173654	MOV	#BUFFER,JDPC	:START
1491	006210	013737	002110	001122	MOV	DNAME,\$BDADR	:SET UP REG ADRS 24
1492	006216	017737	173666	001126	MOV	JDNAME,\$BDDAT	:READ NAME REG
1493	006224	042737	170000	001126	BIC	#170000,\$BDDAT	:ONLY INTERESTED IN THE NAME BITS
1494	006232	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	:DID IT GET LOADED PROPERLY?
1495	006240	001401			BEQ	25	:BR IF OK
1496	006242	104023			ERROR	23	:DPU NAME INSTR ER
1497	006244	006200			ASR	RO	:SHIFT ZERO BIT RIGHT
1498	006246	103742			BCS	15	:BR IF MORE BIT POSITIONS

25:

 :*TEST 41 TEST THAT RESET WILL CLEAR DPU NAME

1503	006250	000004			TST41: SCOPE		
1504	006252	012737	000040	001164	MOV	#40,\$TIMES	:DO 40 ITERATIONS
1505	006260	012777	020000	173630	MOV	#BIT13,\$STKPT	:SET MAINT SWITCH 2
1506	006266	005037	001124		CLR	\$GDDAT	:EXPECT ALL ZEROS
1507	006272	012737	153777	027526	MOV	#153777,BUFFER	:SET UP NAME INSTR
1508	006300	012777	027526	173556	MOV	#BUFFER,JDPC	:START
1509	006306	013737	002110	001122	MOV	DNAME,\$BDADR	:SET UP REG ADRS 24
1510	006314	000005			RESET		:CLR DPU NAME REG
1511	006316	017737	173566	001126	MOV	JDNAME,\$BDDAT	:READ IT
1512	006324	001401			BEQ	TST42	:NEXT TEST IF ALL CLEARED
1513	006326	104023			ERROR	23	:RESET FAILED TO CLEAR DPU NAME

 :*TEST 42 TEST THAT THE SEARCH CODE BITS CAN BE READ AT REG 24

1518	006330	000004			TST42: SCOPE		
1519	006332	012737	006360	001110	MOV	#15,\$LPERR	:SET UP SCOPE LOOP ADRS
1520	006340	013737	002110	001122	MOV	DNAME,\$BDADR	:SET UP REG 24 ADRS
1521	006346	012737	030000	001124	MOV	#30000,\$GDDAT	:EXPECT BOTH BITS INITIALLY
1522	006354	012700	070000		MOV	#70000,RO	:SET UP RO WITH ENABLE
1523	006360	010077	173520		MOV	RO,\$ANAME	:SEND SEARCH CODE
1524	006364	017737	173520	001126	MOV	JDNAME,\$BDDAT	:READ IT BACK
1525	006372	042737	147777	001126	BIC	#147777,\$BDDAT	:SAVE ONLY SEARCH CODE
1526	006400	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	:IS IT CORRECT?
1527	006406	001401			BEQ	25	:BR IF OK
1528	006410	104024			ERROR	24	:SEARCH CODE READ/WRITE ER
1529	006412	162700	010000		SUB	#10000,RO	:SET UP NEXT SEARCH CODE
1530	006416	162737	010000	001124	SUB	#10000,\$GDDAT	:SET UP NEXT SEARCH CODE
1531	006424	100355			BPL	15	:BR IF MORE CODES TO TEST

15:

25:

 :*TEST 43 TEST THAT THE SEARCH CODE DOES NOT CHANGE WHEN ENA=0

1536	006426	000004			TST43: SCOPE		
1537	006430	013737	002110	001122	MOV	DNAME,\$BDADR	:SET UP REG 24 ADRS
1538	006436	012737	030000	001124	MOV	#30000,\$GDDAT	:EXPECT BOTH BITS
1539	006444	012777	070000	173432	MOV	#70000,\$ANAME	:LOAD UP SEARCH CODE
1540	006452	005077	173426		CLR	\$ANAME	:TRY TO CLR SEARCH CODE WITH ENA=0
1541	006456	017737	173426	001126	MOV	JDNAME,\$BDDAT	:READ IT BACK
1542	006464	042737	147777	001126	BIC	#147777,\$BDDAT	:SAVE ONLY SEARCH CODE
1543	006472	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	:IS IT CORRECT?
1544	006500	001401			BEQ	TST44	:ADVANCE TO NEXT TEST IF OK
1545	006502	104024			ERROR	24	:SEARCH CODE CHANGE ENABLE ER

H03

TEST THAT THE SEARCH CODE DOES NOT CHANGE WHEN ENA=0

```
1546
1547
1548
1549
1550 006504 000004
1551 006506 012737 000040 001164
1552 006514 013737 002110 001122
1553 006522 005037 001124
1554 006526 012777 070000 173350
1555 006534 000005
1556 006536 017737 173346 001126
1557 006544 001401
1558 006546 104024
1559
1560
1561
1562
1563 006550 000004
1564 006552 012737 000010 001164
1565 006560 012777 020000 173330
1566 006566 012737 000340 177776
1567 006574 012737 006630 001110
1568 006602 012737 033600 001124
1569 006610 052737 044000 001124
1570 006616 013777 001124 173260
1571 006624 012700 003600
1572 006630 042737 144000 001124
1573 006636 012737 150000 027526
1574 006644 050037 027526
1575 006650 012777 027526 173206
1576 006656 013737 002110 001122
1577 006664 017737 173220 001126
1578 006672 100410
1579 006674 023737 001124 001126
1580 006702 001017
1581 006704 052737 100000 001124
1582 006712 104024
1583 006714 042737 100000 001126
1584 006722 023737 001124 001126
1585 006730 001404
1586 006732 052737 100000 001126
1587 006740 104024
1588 006742 162700 000200
1589 006746 100330
1590 006750 162737 000200 001124
1591 006756 022737 027600 001124
1592 006764 001311
1593
1594
1595
1596
1597 006766 000004
1598 006770 012737 000004 001164
1599 006776 012777 020000 173112
1600 007004 012737 000340 177776
1601 007012 012737 007046 001110
```

```
*****
*TEST 44 TEST THAT RESET WILL CLEAR SEARCH CODE
*****
TST44: SCOPE
MOV #40,$TIMES ;;DO 40 ITERATIONS
MOV DNAME,$BDADR ;SET UP REG 24 ADRS
CLR $GDDAT ;EXPECT RESET STATE
MOV #70000,$ANAME ;SET BOTH SEARCH CODE BITS
RESET ;CLR THEM
MOV $DNAME,$BDDAT ;READ REG
BEQ TST45 ;NEXT TEST IF ALL CLEARED
ERROR 24 ;RESET FAILED TO CLR SEARCH CODE
```

```
*****
*TEST 45 TEST THAT THE NAME MATCH FLAG WILL SET ON 4 BIT COMPARES
*****
TST45: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV #340,$PSW ;SET UP PRIORITY TO HIGHEST LEVEL
MOV #25,$LPERR ;SET UP SCOPE LOOP ADRS
MOV #33600,$GDDAT ;START WITH MAX VALUE
1$: BIS #44000,$GDDAT ;SET THE SEARCH CODE ENA BIT
MOV $GDDAT,$ANAME ;SET UP ASSOCIATIVE NAME REG
MOV #3600,$R0 ;SET UP R0 WITH MAX VALUE
2$: BIC #144000,$GDDAT ;CLR MATCH FLAG & SEARCH ENA BIT
MOV #150000,$BUFFER ;SET UP NAME INSTR
BIS $R0,$BUFFER ;SET UP NAME VALUE
MOV #BUFFER,$DPC ;START
MOV DNAME,$BDADR ;SET UP REG ADRS 24
MOV $DNAME,$BDDAT ;READ NAME REG
BMI 3$ ;BR IF A MATCH OCCURRED
CMP $GDDAT,$BDDAT ;SHOULD A MATCH HAVE OCCURRED?
BNE 4$ ;BR IF NOT
BIS #100000,$GDDAT ;INDICATE IT SHOULD BE SET
ERROR 24 ;NAME MATCH FLAG FAILED TO SET
3$: BIC #100000,$BDDAT ;CLR OUT MATCH FLAG BIT
CMP $GDDAT,$BDDAT ;MAKE SURE THERE IS A MATCH
BEQ 4$ ;BR IF OK
BIS #100000,$BDDAT ;RESTORE THE MATCH FLAG BIT
ERROR 24 ;NAME MATCH FLAG SET IN ER
4$: SUB #200,$R0 ;ADVANCE NAME VALUE
BPL 2$ ;BR IF MORE VALUES TO TEST
SUB #200,$GDDAT ;ADVANCE VALUE FOR ASSO. NAME REG
CMP #27600,$GDDAT ;HAVE ALL VALUES BEEN TESTED?
BNE 1$ ;BR IF NOT
```

```
*****
*TEST 46 TEST THAT THE NAME MATCH FLAG WILL SET ON 8 BIT COMPARES
*****
TST46: SCOPE
MOV #4,$TIMES ;;DO 4 ITERATIONS
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV #340,$PSW ;SET PRIORITY TO HIGHEST LEVEL
MOV #25,$LPERR ;SET UP SCOPE LOOP ADRS
```

1602	007020	012737	023770	001124		MOV	#23770,\$GDDAT	:START WITH MAX VALUE
1603	007026	052737	044000	001124	1S:	BIS	#44000,\$GDDAT	:SET SEARCH ENA BIT
1604	007034	013777	001124	173042		MOV	\$GDDAT,\$ANAME	:SET UP ASSOCIATIVE NAME REG
1605	007042	012700	003770			MOV	#3770,RO	:SET UP RO WITH MAX VALUE
1606	007046	042737	144000	001124	2S:	BIC	#144000,\$GDDAT	:CLR MATCH FLAG & SEARCH ENA
1607	007054	012737	150000	027526		MOV	#150000,BUFFER	:SET UP NAME INSTR
1608	007062	050037	027526			BIS	RO,BUFFER	:SET UP NAME VALUE
1609	007066	012777	027526	172770		MOV	#BUFFER,\$DPC	:START
1610	007074	013737	002110	001122		MOV	DNAME,\$BADR	:SET UP REG ADRS 24
1611	007102	017737	173002	001126		MOV	\$DNAME,\$BDDAT	:READ NAME REG
1612	007110	100410				BMI	3S	:BR IF A MATCH OCCURED
1613	007112	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:SHOULD A MATCH HAVE OCCURED?
1614	007120	001017				BNE	4S	:BR IF NOT
1615	007122	052737	100000	001124		BIS	#100000,\$GDDAT	:INDICATE IT SHOULD BE SET
1616	007130	104024				ERROR	24	:NAME MATCH FLAG FAILED TO SET
1617	007132	042737	100000	001126	3S:	BIC	#100000,\$BDDAT	:CLR OUT MATCH FLAG BIT
1618	007140	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:MAKE SURE THERE IS A MATCH
1619	007146	001404				BEQ	4S	:BR IF OK
1620	007150	052737	100000	001126		BIS	#100000,\$BDDAT	:RESTORE THE MATCH FLAG BIT
1621	007156	104024				ERROR	24	:NAME MATCH FLAG SET IN ER
1622	007160	162700	000010		4S:	SUB	#10,RO	:ADVANCE NAME VALUE
1623	007164	100330				BPL	2S	:BR IF MORE VALUES TO TEST
1624	007166	162737	000010	001124		SUB	#10,\$GDDAT	:ADVANCE VALUE FOR ASSO. NAME REG
1625	007174	022737	017770	001124		CMP	#17770,\$GDDAT	:HAVE ALL VALUES BEEN TESTED?
1626	007202	001311				BNE	1S	:BR IF NOT

 ;*TEST 47 TEST THAT THE NAME MATCH FLAG WILL SET ON 11 BIT COMPARES

1630						†ST47:	SCOPE	
1631	007204	000004				MOV	#2,\$TIMES	::DO 2 ITERATIONS
1632	007206	012737	000002	001164		MOV	#BIT13,\$STKPT	:SET MAINT SWITCH #2
1633	007214	012777	020000	172674		MOV	#2S,\$LPERR	:SET UP SCOPE LOOP ADRS
1634	007222	012737	007256	001110		MOV	#10077,\$GDDAT	:START WITH 77 - WORK ONLY LOWER ORDER BITS
1635	007230	012737	010077	001124		MOV	#44000,\$GDDAT	:SET SEARCH ENA BIT
1636	007236	052737	044000	001124	1S:	BIS	#44000,\$GDDAT	:SET UP ASSOCIATIVE NAME REG
1637	007244	013777	001124	172632		MOV	\$GDDAT,\$ANAME	:SET UP RO WITH 77
1638	007252	012700	000077			MOV	#77,RO	:CLR MATCH FLAG & SEARCH ENA
1639	007256	042737	144000	001124	2S:	BIC	#144000,\$GDDAT	:SET UP NAME INSTR
1640	007264	012737	150000	027526		MOV	#150000,BUFFER	:SET UP NAME VALUE
1641	007272	050037	027526			BIS	RO,BUFFER	:START
1642	007276	012777	027526	172560		MOV	#BUFFER,\$DPC	:SET UP REG ADRS 24
1643	007304	013737	002110	001122		MOV	DNAME,\$BADR	:ALLOW TIME FOR NPR
1644	007312	000240				NOP		:READ NAME REG
1645	007314	017737	172570	001126		MOV	\$DNAME,\$BDDAT	:BR IF A MATCH OCCURED
1646	007322	100410				BMI	3S	:SHOULD A MATCH HAVE OCCURED?
1647	007324	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:BR IF NOT
1648	007332	001017				BNE	4S	:INDICATE IT SHOULD BE SET
1649	007334	052737	100000	001124		BIS	#100000,\$GDDAT	:NAME MATCH FLAG FAILED TO SET
1650	007342	104024				ERROR	24	:CLR OUT MATCH FLAG BIT
1651	007344	042737	100000	001126	3S:	BIC	#100000,\$BDDAT	:MAKE SURE THERE IS A MATCH
1652	007352	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:BR IF OK
1653	007360	001404				BEQ	4S	:RESTORE THE MATCH FLAG BIT
1654	007362	052737	100000	001126		BIS	#100000,\$BDDAT	:NAME MATCH FLAG SET IN ER
1655	007370	104024				ERROR	24	:ADVANCE NAME VALUE
1656	007372	005300			4S:	DEC	RO	:BR IF MORE VALUES TO TEST
1657	007374	100330				BPL	2S	

TEST THAT THE NAME MATCH FLAG WILL SET ON 11 BIT COMPARES

1658	007376	005337	001124	
1659	007402	022737	007777	001124
1660	007410	001312		
1661				
1662				
1663				
1664				
1665	007412	000004		
1666	007414	012737	000040	001164
1667	007422	012777	020000	172466
1668	007430	012777	057777	172446
1669	007436	012737	153777	027526
1670	007444	005037	001124	
1671	007450	012777	027526	172406
1672	007456	013737	002110	001122
1673	007464	000005		
1674	007466	017737	172416	001126
1675	007474	001401		
1676	007476	104023		
1677				
1678				
1679				
1680				
1681	007500	000004		
1682	007502	012777	020000	172406
1683	007510	012737	000340	177776
1684	007516	013737	002110	001122
1685	007524	012777	057777	172352
1686	007532	012777	050000	172344
1687	007540	012737	153777	027526
1688	007546	012777	027526	172310
1689	007554	012737	113777	001124
1690	007562	017737	172322	001126
1691	007570	023737	001124	001126
1692	007576	001401		
1693	007600	104024		
1694				
1695				
1696				
1697				
1698	007602	000004		
1699	007604	012737	000002	001164
1700	007612	012777	020000	172276
1701	007620	012737	000340	177776
1702	007626	012737	007702	001110
1703	007634	013701	001250	
1704	007640	012761	007756	000014
1705	007646	012761	000340	000016
1706	007654	012737	010077	001124
1707	007662	052737	044000	001124
1708	007670	013777	001124	172206
1709	007676	012700	000077	
1710	007702	042737	044000	001124
1711	007710	012737	150000	027526
1712	007716	050037	027526	
1713	007722	012777	027526	172134

```

DEC          $GDDAT          ;ADVANCE VALUE FOR ASSO. NAME REG
CMP          #7777,$GDDAT    ;HAVE ALL VALUES BEEN TESTED?
BNE          1$              ;BR IF NOT

;*****
;*TEST 50          TEST THAT RESET WILL CLEAR THE NAME MATCH FLAG
;*****
TST50:  SCOPE
        MOV          #40,$TIMES          ;;DO 40 ITERATIONS
        MOV          #BIT13,$STKPT      ;SET MAINT SWITCH 2
        MOV          #5777,$ANAME      ;SET ANAME FOR 11 BIT COMPARE ON 3777
        MOV          #153777,$BUFFER   ;SET UP NAME INSTR
        CLR          $GDDAT            ;EXPECT ALL ZEROS
        MOV          #BUFFER,$DPC      ;START
        MOV          DNAME,$BDADR      ;SET UP REG ADRS 24
        RESET       ;CLR DPU NAME REG (MATCH)
        MOV          $DNAME,$BDDAT     ;READ IT
        BEQ          TST51            ;NEXT TEST IF ALL CLEARED
        ERROR       23                ;RESET FAILED TO CLEAR NAME MATCH

;*****
;*TEST 51          TEST THAT THE ASSOCIATIVE NAME REG DOES NOT CHANGE WITH ENA=0
;*****
TST51:  SCOPE
        MOV          #BIT13,$STKPT      ;SET MAINT SWITCH #2
        MOV          #340,$PSW         ;DON'T ALLOW INTERRUPT
        MOV          DNAME,$BDADR      ;SET UP REG 24 ADRS
        MOV          #57777,$ANAME     ;SET UP FOR 11 BIT CMP ON VALUE #3777
        MOV          #50000,$ANAME     ;NOW TRY WRITING 0 WITH ENA DISABLED
        MOV          #153777,$BUFFER   ;SET UP NAME INSTR WITH VALUE #3777
        MOV          #BUFFER,$DPC      ;START
        MOV          #113777,$GDDAT    ;EXPECT NAME MATCH ON 11 BIT CMP VALUE #3777
        MOV          $DNAME,$BDDAT     ;GET DPU NAME REG CONTENTS
        CMP          $GDDAT,$BDDAT     ;IS THERE A MATCH?
        BEQ          TST52            ;ADVANCE TO NEXT TEST IF SO
        ERROR       24                ;ASSOCIATIVE NAME CHANGE ENABLE ER

;*****
;*TEST 52          TEST THAT AN INTERRUPT WILL OCCUR ON 11 BIT NAME MATCH COMPARES
;*****
TST52:  SCOPE
        MOV          #2,$TIMES          ;;DO 2 ITERATIONS
        MOV          #BIT13,$STKPT      ;SET MAINT SWITCH #2
        MOV          #340,$PSW         ;SET PSW TO 7
        MOV          #2$,$LPERR        ;SET UP SCOPE LOOP ADRS
        MOV          $VECT1,R1         ;GET VECTOR ADRS
        MOV          #3$,$14(R1)       ;SET UP NAME INT SER ADRS
        MOV          #340,$16(R1)      ;SET UP PSW TO 7 ON INT
        MOV          #10077,$GDDAT     ;START WITH MAX VALUE
        BIS          #44000,$GDDAT     ;SET SEACH CHANGE & NAME CHANGE ENABLES
        MOV          $GDDAT,$ANAME     ;SET UP ASSOCIATIVE NAME REG
        MOV          #77,$RO           ;SET UP RO WITH MAX VALUE
        BIC          #44000,$GDDAT     ;CLR ENA BITS
        MOV          #150000,$BUFFER   ;SET UP NAME INSTR
        BIS          RO,$BUFFER        ;SET UP NAME VALUE
        MOV          #BUFFER,$DPC      ;START

```

```

1714 007730 013737 002110 001122      MOV      DNAME,$BDADR      ;SET UP REG ADRS 24
1715 007736 005037 177776                CLR      PSW                ;ALLOW INT TO OCCUR
1716 007742 017737 172142 001126      MOV      @DNAME,$BDDAT    ;READ THE NAME REG
1717 007750 100010                BPL      4$                ;BR IF NO MATCH
1718 007752 104047                ERROR   47                ;NAME MATCH FAILED TO INTERRUPT
1719 007754 000416                BR       5$                ;GO LOOK FOR LOOP ON TEST SWITCH
1720 007756 022626                3$:    CMP      (R6)+,(R6)+    ;FIX STK SINCE NO RTI
1721 007760 017737 172124 001126      MOV      @DNAME,$BDDAT    ;READ NAME REG
1722 007766 100401                BMI      4$                ;BR IF MATCH FLAG SET
1723 007770 104047                ERROR   47                ;NAME MATCH INTERRUPTED BUT NO FLAG
1724 007772 005300                4$:    DEC      R0            ;ADVANCE NAME VALUE IN R0
1725 007774 100342                BPL      2$                ;BR IF MORE VALUES TO TEST
1726 007776 005337 001124                DEC      $GDDAT            ;ADVANCE VALUE FOR ASSO. NAME REG
1727 010002 022737 007777 001124      CMP      #7777,$GDDAT     ;SEE IF ALL VALUES HAVE BEEN TESTED
1728 010010 001324                BNE      1$                ;BR IF NOT
1729 010012 004737 023134                5$:    JSR      PC,RSTVEC     ;RESTORE NAME MATCH VECTOR WITH HALT
1730
1731
1732
1733

```

```

;*****
;*TEST 53      LOAD #1 INTO LINE TYPE REGISTERS
;*****

```

```

1734 010016 000004                TST53: SCOPE
1735 010020 012777 020000 172070      MOV      #BIT13,@STKPT    ;SET MAINT SWITCH #2
1736 010026 012777 100005 172000      MOV      #100005,@DBUF    ;LINE TYPE ENABLE =1 LINE TYPE =1
1737 010034 012737 000001 001124      MOV      #1,$GDDAT        ;LOAD EXPECTED
1738 010042 013777 002034 172014      MOV      DBUF,@DPC        ;LOAD DISPLAY P.C.
1739 010050 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1740 010056 017737 172004 001126      MOV      @SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1741 010064 042737 177774 001126      BIC      #177774,$BDDAT   ;MASK TO BITS 1-0
1742 010072 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1743 010100 001401                BEQ      TST54            ;:BR IF EQUAL
1744 010102 104005                ERROR   5                ;LINE BIT 0 FAILED TO SET
1745
1746
1747

```

```

;*****
;*TEST 54      LOAD #2 INTO LINE TYPE REGISTERS
;*****

```

```

1748
1749 010104 000004                TST54: SCOPE
1750 010106 012777 020000 172002      MOV      #BIT13,@STKPT    ;SET MAINT SWITCH #2
1751 010114 012777 100006 171712      MOV      #100006,@DBUF    ;LINE TYPE ENABLE =1 LINE TYPE =2
1752 010122 012737 000002 001124      MOV      #2,$GDDAT        ;LOAD EXPECTED
1753 010130 013777 002034 171726      MOV      DBUF,@DPC        ;LOAD DISPLAY P.C.
1754 010136 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1755 010144 017737 171716 001126      MOV      @SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1756 010152 042737 177774 001126      BIC      #177774,$BDDAT   ;MASK TO BITS 1-0
1757 010160 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1758 010166 001401                BEQ      TST55            ;:BR IF EQUAL
1759 010170 104005                ERROR   5                ;LINE BIT 1 FAILED TO SET
1760
1761
1762

```

```

;*****
;*TEST 55      CHECK LINE TYPE ENABLE GATE
;*****

```

```

1763
1764
1765 010172 000004                TST55: SCOPE
1766 010174 012777 020000 171714      MOV      #BIT13,@STKPT    ;SET MAINT SWITCH #2
1767 010202 012777 100007 171624      MOV      #100007,@DBUF    ;LINE TYPE ENABLE =1 LINE TYPE =3
1768 010210 013777 002034 171646      MOV      DBUF,@DPC        ;LOAD DISPLAY P.C.
1769 010216 012777 100000 171610      MOV      #100000,@DBUF

```

```

1770 010224 012737 000003 001124      MOV      #3,$GDDAT          ;LOAD EXPECTED
1771 010232 013777 002034 171624      MOV      DBUF,$DPC
1772 010240 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1773 010246 017737 171614 001126      MOV      $SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1774 010254 042737 177774 001126      BIC      #177774,$BDDAT   ;MASK TO BITS 1-0
1775 010262 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1776 010270 001401          BEQ      TST56             ;;BR IF EQUAL
1777 010272 104005          ERROR    5                ;LINE TYPE ENABLE FAILED TO INHIBIT
1778                                     ;CHANGING OF LINETYPE VALUE
1779

```

```

;*****
;*TEST 56      LOAD #0 INTO BLINK REGISTERS
;*****

```

```

1783 010274 000004          TST56: SCOPE
1784 010276 012777 020000 171612      MOV      #BIT13,$STKPT    ;SET MAINT SWITCH #2
1785 010304 012777 100020 171522      MOV      #100020,$DBUF   ;BLINK ENABLE =1  BLINK =0
1786 010312 005037 001124          CLR      $GDDAT          ;LOAD EXPECTED
1787 010316 013777 002034 171540      MOV      DBUF,$DPC       ;LOAD DISPLAY P.C.
1788 010324 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1789 010332 017737 171530 001126      MOV      $SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1790 010340 042737 177767 001126      BIC      #177767,$BDDAT   ;MASK TO BIT 3
1791 010346 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1792 010354 001401          BEQ      TST57             ;;BR IF EQUAL
1793 010356 104027          ERROR    27              ;BLINK BIT FAILED TO RESET
1794

```

```

;*****
;*TEST 57      LOAD #1 INTO BLINK REGISTERS
;*****

```

```

1797 010360 000004          TST57: SCOPE
1798 010362 012777 020000 171526      MOV      #BIT13,$STKPT    ;SET MAINT SWITCH #2
1799 010370 012737 100030 027526      MOV      #100030,$DBUF   ;BLINK ENABLE =1  BLINK =1
1800 010376 012737 000010 001124      MOV      #10,$GDDAT       ;LOAD EXPECTED
1801 010404 013777 002034 171452      MOV      DBUF,$DPC       ;LOAD DISPLAY P.C.
1802 010412 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1803 010420 017737 171442 001126      MOV      $SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1804 010426 042737 177767 001126      BIC      #177767,$BDDAT   ;MASK TO BIT 3
1805 010434 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1806 010442 001401          BEQ      TST60             ;;BR IF EQUAL
1807 010444 104027          ERROR    27              ;BLINK BIT FAILED TO SET
1808

```

```

;*****
;*TEST 60      CHECK BLINK ENABLE GATE
;*****

```

```

1813 010446 000004          TST60: SCOPE
1814 010450 012777 020000 171440      MOV      #BIT13,$STKPT    ;SET MAINT SWITCH #2
1815 010456 012777 100030 171350      MOV      #100030,$DBUF   ;LOAD BLINK ON
1816 010464 013777 002034 171372      MOV      DBUF,$DPC
1817 010472 012737 000010 001124      MOV      #BIT3,$GDDAT     ;LOAD EXPECTED
1818 010500 012777 100000 171326      MOV      #100000,$DBUF   ;BLINK ENABLE =0  BLINK =0
1819 010506 013777 002034 171350      MOV      DBUF,$DPC       ;LOAD DISPLAY P.C.
1820 010514 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1821 010522 017737 171340 001126      MOV      $SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1822 010530 042737 177767 001126      BIC      #177767,$BDDAT   ;MASK TO BIT 3
1823 010536 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1824 010544 001401          BEQ      TST61             ;;BR IF EQUAL
1825 010546 104027          ERROR    27              ;BLINK ENABLE FAILED TO INHIBIT

```

;CHANGING OF THE BLINK BIT

```

1826
1827
1828
1829
1830
1831 010550 000004
1832 010552 012777 020000 171336
1833 010560 012737 100100 001124
1834 010566 013777 001124 171240
1835 010574 013777 002034 171262
1836 010602 013737 002066 001122
1837 010610 017737 171252 001126
1838 010616 032737 000200 001126
1839 010624 001401
1840 010626 104034
1841
1842
1843
1844
1845 010630 000004
1846 010632 012777 020000 171256
1847 010640 012737 100140 001124
1848 010646 013777 001124 171160
1849 010654 013777 002034 171202
1850 010662 013737 002066 001122
1851 010670 017737 171172 001126
1852 010676 032737 000200 001126
1853 010704 001401
1854 010706 104034
1855
1856
1857
1858
1859 010710 000004
1860 010712 012777 020000 171176
1861 010720 012777 103000 171106
1862 010726 012737 002000 001124
1863 010734 013777 002034 171122
1864 010742 013737 002066 001122
1865 010750 017737 171112 001126
1866 010756 042737 174377 001126
1867 010764 023737 001124 001126
1868 010772 001401
1869 010774 104006
1870
1871
1872
1873
1874 010776 000004
1875 011000 012777 020000 171110
1876 011006 012777 102400 171020
1877 011014 012737 001000 001124
1878 011022 013777 002034 171034
1879 011030 013737 002066 001122
1880 011036 017737 171024 001126
1881 011044 042737 174377 001126

```

```

;*****
;*TEST 61 TEST FOR FALSE L.P. STATUS FLAG (L.P. - NOT ENABLED)
;*****
TST61: SCOPE
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #100100, $GDDAT ;LP ENABLE =1 LP=0
MOV $GDDAT, JDBUF ;LOAD BUFFER
MOV DBUF, JDPIC ;LOAD DISPLAY P.C.
MOV SREG0, $BDADR ;SET UP REG ADRS 02
MOV JSREG0, $BDDAT ;READ STATUS
BIT #200, $BDDAT
BEQ TST62 ;;BR IF EQUAL
ERROR 34 ;LIGHT PEN FLAG SET IN ERROR

;*****
;*TEST 62 TEST FOR FALSE L.P. STATUS FLAG (L.P. - ENABLED)
;*****
TST62: SCOPE
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #100140, $GDDAT ;LP ENABLE =1 LP=1
MOV $GDDAT, JDBUF ;LOAD BUFFER
MOV DBUF, JDPIC ;LOAD DISPLAY P.C.
MOV SREG0, $BDADR ;SET UP REG ADRS 02
MOV JSREG0, $BDDAT ;READ STATUS
BIT #200, $BDDAT
BEQ TST63 ;;BR IF EQUAL
ERROR 34 ;LIGHT PEN FLAG SET IN ERROR

;*****
;*TEST 63 LOAD #4 INTO INT LEVEL REGISTER
;*****
TST63: SCOPE
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #103000, JDBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =4
MOV #2000, $GDDAT ;LOAD EXPECTED
MOV DBUF, JDPIC ;LOAD DISPLAY P.C.
MOV SREG0, $BDADR ;SET UP REG ADRS 02
MOV JSREG0, $BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377, $BDDAT ;MASK TO BITS 8-10
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST64 ;;BR IF EQUAL
ERROR 6 ;INTENSITY LEVEL BIT 10 FAILED

;*****
;*TEST 64 LOAD #2 INTO INT LEVEL REGISTER
;*****
TST64: SCOPE
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #102400, JDBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =2
MOV #1000, $GDDAT ;LOAD EXPECTED
MOV DBUF, JDPIC ;LOAD DISPLAY P.C.
MOV SREG0, $BDADR ;SET UP REG ADRS 02
MOV JSREG0, $BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377, $BDDAT ;MASK TO BITS 8-10

```

```

1882 011052 023737 001124 001126
1883 011060 001401
1884 011062 104006
1885
1886
1887
1888
1889 011064 000004
1890 011066 012777 020000 171022
1891 011074 012777 102200 170732
1892 011102 012737 000400 001124
1893 011110 013777 002034 170746
1894 011116 013737 002066 001122
1895 011124 017737 170736 001126
1896 011132 042737 174377 001126
1897 011140 023737 001124 001126
1898 011146 001401
1899 011150 104006
1900
1901
1902
1903
1904
1905 011152 000004
1906 011154 012777 020000 170734
1907 011162 012777 103200 170644
1908 011170 012777 100000 170640
1909 011176 013777 002034 170660
1910 011204 005277 170654
1911 011210 012737 002400 001124
1912 011216 013737 002066 001122
1913 011224 017737 170636 001126
1914 011232 042737 174377 001126
1915 011240 023737 001124 001126
1916 011246 001401
1917 011250 104006
1918
1919
1920
1921
1922
1923 011252 000004
1924 011254 012777 020000 170634
1925 011262 012777 170040 170544
1926 011270 005037 001124
1927 011274 013777 002034 170562
1928 011302 013737 002066 001122
1929 011310 017737 170552 001126
1930 011316 042737 177757 001126
1931 011324 023737 001124 001126
1932 011332 001401
1933 011334 104030
1934
1935
1936
1937

```

```

CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST65 ;:BR IF EQUAL
ERROR 6 ;:INTENSITY LEVEL BIT 9 FAILED

;*****
;*TEST 65 LOAD #1 INTO INT LEVEL REGISTER
;*****
TST65: SCOPE
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV #102200,$DBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =1
MOV #400,$GDDAT ;LOAD EXPECTED
MOV $DBUF,$DPC ;LOAD DISPLAY P.C.
MOV $SREG0,$BDADR ;SET UP REG ADRS 02
MOV $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377,$BDDAT ;MASK TO BITS 8-10
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST66 ;:BR IF EQUAL
ERROR 6 ;:INTENSITY LEVEL BIT 8 FAILED

;*****
;*TEST 66 CHECK INTENSITY LEVEL ENABLE GATE
;*****
TST66: SCOPE
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV #103200,$DBUF ;LOAD LEVEL
MOV #100000,$DBUF1 ;INTENSITY LEVEL ENABLE =0 LEVEL =0
MOV $DBUF,$DPC ;LOAD DISPLAY P.C.
INC $DPC ;SINGLE STEP THE DISPLAY
MOV #2400,$GDDAT ;LOAD EXPECTED
MOV $SREG0,$BDADR ;SET UP REG ADRS 02
MOV $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377,$BDDAT ;MASK TO BITS 8-10
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST67 ;:BR IF EQUAL
ERROR 6 ;:INTENSITY LEVEL ENABLE FAILED TO INHIBIT
;:INTENSITY LEVEL CHANGE

;*****
;*TEST 67 LOAD #0 INTO ITALIC REGISTER
;*****
TST67: SCOPE
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV #170040,$DBUF ;ITALICS ENABLE=1 ITALICS=0
CLR $GDDAT ;LOAD EXPECTED
MOV $DBUF,$DPC ;LOAD DISPLAY P.C.
MOV $SREG0,$BDADR ;SET UP REG ADRS 02
MOV $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,$BDDAT ;MASK TO BIT 4
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST70 ;:BR IF EQUAL
ERROR 30 ;ITALICS BIT FAILED TO RESET

;*****
;*TEST 70 LOAD #1 INTO ITALIC REGISTER
;*****

```



```

1938
1939 011336 000004
1940 011340 012777 020000 170550
1941 011346 012777 170060 170460
1942 011354 012737 000020 001124
1943 011362 013777 002034 170474
1944 011370 013737 002066 001122
1945 011376 017737 170464 001126
1946 011404 042737 177757 001126
1947 011412 023737 001124 001126
1948 011420 001401
1949 011422 104030
1950
1951
1952
1953
1954
1955 011424 000004
1956 011426 012777 020000 170462
1957 011434 012777 170060 170372
1958 011442 012777 170000 170366
1959 011450 013777 002034 170406
1960 011456 012737 000020 001124
1961 011464 005277 170374
1962 011470 013737 002066 001122
1963 011476 017737 170364 001126
1964 011504 042737 177757 001126
1965 011512 023737 001124 001126
1966 011520 001401
1967 011522 104030
1968
1969
1970
1971
1972
1973 011524 000004
1974 011526 013737 002066 001122
1975 011534 012777 020000 170354
1976 011542 012737 172000 027526
1977 011550 013777 002034 170306
1978 011556 012737 100000 001124
1979 011564 017737 170276 001126
1980 011572 100401
1981 011574 104037
1982 011576 012737 160000 027526
1983 011604 012777 027526 170252
1984 011612 005037 001124
1985 011616 017737 170244 001126
1986 011624 100001
1987 011626 104040
1988
1989
1990
1991
1992 011630 000004
1993 011632 012737 000010 001164

```

```

*****
†TST70: SCOPE
MOV #BIT13,‡STKPT ;SET MAINT SWITCH #2
MOV #170060,‡DBUF ;ITALICS ENABLE=1 ITALICS=1
MOV #BIT4,‡SGDDAT ;LOAD EXPECTED
MOV ‡DBUF,‡DPC ;LOAD DISPLAY P.C.
MOV ‡SREG0,‡SBDADR ;SET UP REG ADRS 02
MOV ‡SREG0,‡SBDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,‡SBDAT ;MASK TO BIT 4
CMP ‡SGDDAT,‡SBDAT ;COMPARE EXPCT TO RCVD
BEQ TST71 ;;BR IF EQUAL
ERROR 30 ;ITALICS BIT FAILED TO SET

```

```

*****
*TEST 71 CHECK ITALIC ENABLE GATE
*****
†TST71: SCOPE
MOV #BIT13,‡STKPT ;SET MAINT SWITCH #2
MOV #170060,‡DBUF ;LOAD ENABLE
MOV #170000,‡DBUF1 ;ITALICS ENABLE=0 ITALICS=0
MOV ‡DBUF,‡DPC ;LOAD EXPECTED
INC ‡DPC ;SINGLE STEP THE DISPLAY
MOV ‡SREG0,‡SBDADR ;SET UP REG ADRS 02
MOV ‡SREG0,‡SBDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,‡SBDAT ;MASK TO BITS 4
CMP ‡SGDDAT,‡SBDAT ;COMPARE EXPCT TO RCVD
BEQ TST72 ;;BR IF EQUAL
ERROR 30 ;ITALICS ENABLE FAILED TO INHIBIT
;CLEARING OF ITALICS BIT

```

```

*****
*TEST 72 TEST INTERNAL STOP FLAG CAN SET AND RESET
*****
†TST72: SCOPE
MOV ‡SREG0,‡SBDADR ;SET UP REG ADRS 02
MOV #BIT13,‡STKPT ;SET MAINT SWITCH #2
MOV #172000,‡BUFFER ;"STOP" BIT =1
MOV ‡DBUF,‡DPC ;LOAD DISPLAY P.C.
MOV #BIT15,‡SGDDAT ;LOAD EXPECTED
MOV ‡SREG0,‡SBDAT ;READ DISPLAY STATUS REGISTER
BMI 15 ;;BR IF SET
ERROR 37 ;"INTERNAL-STOP" BIT FAILED TO SET
15: MOV #160000,‡BUFFER ;LOAD "NOP"
MOV #‡BUFFER,‡DPC ;START DISPLAY
CLR ‡SGDDAT ;CLEAR EXPECTED
MOV ‡SREG0,‡SBDAT ;READ DISPLAY STATUS REGISTER
BPL TST73 ;;BR IF CLEARED
ERROR 40 ;"INTERNAL-STOP" BIT FAILED TO RESET

```

```

*****
*TEST 73 TEST THAT RESET WILL CLEAR STOP, MODE, INTENSITY, ITALICS, BLINK & LINE
*****
†TST73: SCOPE
MOV #10,‡TIMES ;;DO 10 ITERATIONS

```

1994	011640	012777	020000	170250	MOV	#BIT13,STKPT	:SET UP MAINT SW 2
1995	011646	012737	102637	027526	MOV	#102637,BUFFER	:SET UP CHAR
1996	011654	012737	170060	027530	MOV	#170060,BUFFER+2	:SET UP ITALICS INSTR
1997	011662	012777	027526	170174	MOV	#BUFFER,JDPC	:START
1998	011670	012737	002000	001124	MOV	#2000,\$GDDAT	:EXPECT INTENSITY ONLY
1999	011676	005277	170162		INC	JDPC	:PICK UP ITALICS INSTR
2000	011702	012737	172000	027532	MOV	#172000,BUFFER+4	:SET UP STOP INSTR
2001	011710	005277	170150		INC	JDPC	:PICK UP STOP INSTR
2002	011714	013737	002066	001122	MOV	SREG0,\$BDADR	:SET UP REG ADRS 02
2003	011722	000005			RESET		:CLR REG 02
2004	011724	017737	170136	001126	MOV	\$SREG0,\$BDDAT	:READ REG
2005	011732	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	:ALL BUT THE MSB OF INTENSITY SHOULD HAVE BEEN CLEARED
2006	011740	001401			BEG	TST74	:NEXT TEST IF OK
2007	011742	104032			ERROR	32	:RESET FAILED TO SET UP REG 02

 :*TEST 74 TEST THAT INTERNAL STOP WILL CAUSE AN INTERRUPT

2012	011744	000004			TST74: SCOPE		
2013	011746	012737	000100	001164	MOV	#100,\$TIMES	:DO 100 ITERATIONS
2014	011754	012737	000074	001204	MOV	#STN-1,STESTN	:SET TEST NUMBER IN MAIL BOX
2015	011762	012737	000340	177776	MOV	#340,PSW	:HIGHEST PRIORITY
2016	011770	013737	002066	001122	MOV	SREG0,\$BDADR	:SET UP REG ADRS 02
2017	011776	012737	172000	001124	MOV	#172000,\$GDDAT	:EXPECT STOP MODE AND INTENSITY LEVEL ONLY
2018	012004	013700	001250		MOV	SVECT1,RO	:GET INTR ADRS
2019	012010	012710	012070		MOV	#2\$(RO)	:SET UP RETURN ADRS
2020	012014	012760	000340	000002	MOV	#340,2(RO)	:HIGHEST PRIORITY ON INTR
2021	012022	012777	020040	170066	MOV	#20040,STKPT	:SET MAINT SW 2
2022	012030	012737	173400	027526	MOV	#173400,BUFFER	:ENABLE STOP INTR + STOP
2023	012036	012777	027526	170020	1\$: MOV	#BUFFER,JDPC	:START
2024	012044	005037	177776		CLR	PSW	:ALLOW INTR
2025	012050	021616			CMP	(SP),(SP)	:FUMBLE
2026	012052	005700			TST	RO	:EXPECT INTR?
2027	012054	001430			BEG	5\$:BR IF NOT
2028	012056	017737	170004	001126	MOV	\$SREG0,\$BDDAT	:READ STATUS
2029	012064	104000			ERROR		:INTERNAL STOP FAILED TO INTR
2030	012066	000423			BR	5\$:LOOK FOR LOOP ON TEST SW
2031	012070	022626			2\$: CMP	(SP)+,(SP)+	:FIX STACK SINCE NO RTI
2032	012072	017737	167770	001126	MOV	\$SREG0,\$BDDAT	:READ REG 02
2033	012100	005700			TST	RO	:EXPECT INTR?
2034	012102	001002			BNE	3\$:BR IF SO
2035	012104	104033			ERROR	33	:INTERNAL STOP INTR'ED WHEN NOT ENABLED
2036	012106	000413			BR	5\$:GO LOOK FOR LOOP ON TEST SW
2037	012110	023737	001124	001126	3\$: CMP	\$GDDAT,\$BDDAT	:STATUS CORRECT?
2038	012116	001402			BEG	4\$:BR IF SO
2039	012120	104033			ERROR	33	:STOP FLAG NOT SET ON INTR
2040	012122	000405			BR	5\$:GO LOOK FOR LOOP ON TEST SW
2041	012124	012737	173000	027526	4\$: MOV	#173000,BUFFER	:SET UP NOW WITH INTR NOT ENABLED
2042	012132	005000			CLR	RO	:INDICATE NO INTR
2043	012134	000740			BR	1\$:REPEAT TEST
2044	012136	004737	023134		5\$: JSR	PC,RSTVEC	:GO RESTORE VECTORS

 :*TEST 75 LOAD #0 INTO MENU REGISTER

 TST75: SCOPE

2049 012142 000004

```

2050 012144 012777 020000 167744      MOV      #BIT13,JUSTKPT      ;SET MAINT SWITCH #2
2051 012152 012777 170002 167654      MOV      #170002,JUSTBUF    ;MENU ENABLE=1 MENU=0
2052 012160 005037 001124                CLR      $GDDAT             ;LOAD EXPECTED
2053 012164 013777 002034 167672      MOV      DBUF,JUSTPC        ;LOAD DISPLAY P.C.
2054 012172 013737 002076 001122      MOV      SREG1,$BDADR       ;SET UP REG ADRS12
2055 012200 017737 167672 001126      MOV      JSREG1,$BDDAT      ;READ DISP STATUS REGISTER
2056 012206 042737 177677 001126      BIC      #177677,$BDDAT     ;MASK TO BIT 2
2057 012214 023737 001124 001126      CMP      $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
2058 012222 001401                BEQ      TST76              ;;BR IF EQUAL
2059 012224 104031                ERROR   31                  ;MENU BIT FAILED TO RESET

```

```

*****
; *TEST 76      LOAD #1 INTO MENU REGISTER
*****

```

```

2065 012226 000004                TST76:  SCOPE
2066 012230 012777 020000 167660      MOV      #BIT13,JUSTKPT    ;SET MAINT SWITCH #2
2067 012236 012777 170003 167570      MOV      #170003,JUSTBUF   ;MENU ENABLE=1 MENU=1
2068 012244 012737 000100 001124      MOV      #BIT6,$GDDAT      ;LOAD EXPECTED
2069 012252 013777 002034 167604      MOV      DBUF,JUSTPC       ;LOAD DISPLAY P.C.
2070 012260 013737 002076 001122      MOV      SREG1,$BDADR       ;SET UP REG ADRS 12
2071 012266 017737 167604 001126      MOV      JSREG1,$BDDAT     ;READ DISPLAY STATUS REGISTER
2072 012274 042737 177677 001126      BIC      #177677,$BDDAT    ;MASK TO BIT 2
2073 012302 023737 001124 001126      CMP      $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
2074 012310 001401                BEQ      TST77              ;;BR IF EQUAL
2075 012312 104031                ERROR   31                  ;MENU BIT FAILED TO SET

```

```

*****
; *TEST 77      CHECK MENU ENABLE GATE
*****

```

```

2081 012314 000004                TST77:  SCOPE
2082 012316 013737 002076 001122      MOV      SREG1,$BDADR       ;SET UP REG ADRS 12
2083 012324 012777 020000 167564      MOV      #BIT13,JUSTKPT    ;SET MAINT SWITCH #2
2084 012332 012737 170002 027526      MOV      #170002,BUFFER    ;MENU ENABLE = 1 MENU = 0
2085 012340 012777 027526 167516      MOV      #BUFFER,JUSTPC    ;START DISPLAY
2086 012346 012737 170001 027530      MOV      #170001,BUFFER+2  ;MENU ENABLE=0 MENU=1
2087 012354 005277 167504                INC      JUSTPC             ;RESUME
2088 012360 005037 001124                CLR      $GDDAT            ;LOAD EXPECTED
2089 012364 017737 167506 001126      MOV      JSREG1,$BDDAT     ;READ DISPLAY STATUS REGISTER
2090 012372 042737 177677 001126      BIC      #177677,$BDDAT    ;MASK TO BIT 2
2091 012400 023737 001124 001126      CMP      $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
2092 012406 001401                BEQ      TST100            ;;BR IF EQUAL
2093 012410 104031                ERROR   31                  ;MENU ENABLE FAILED TO INHIBIT
2094                                ;RESETTING OF MENU BIT

```

```

*****
; *TEST 100     TEST THAT LOAD STATUS C INSTR CAN SET THE VECTOR SCALE
*****

```

```

2099 012412 000004                TST100: SCOPE
2100 012414 012777 020000 167474      MOV      #BIT13,JUSTKPT    ;SET MAINT SWITCH #2
2101 012422 012737 012436 001110      MOV      #1$,SLPERR        ;SET UP SCOPE LOOP ADRS
2102 012430 012737 000017 001124      MOV      #17,$GDDAT        ;START WITH ALL ONES
2103 012436 012737 154020 027526      MOV      #154020,BUFFER    ;SET UP INSTR AT BUFFER
2104 012444 053737 001124 027526      BIS      $GDDAT,BUFFER     ;SET UP VECTOR SCALE AT INSTR LOC
2105 012452 012777 027526 167404      MOV      #BUFFER,JUSTPC    ;START

```

E04

2106 012460 013737 002076 001122
2107 012466 017737 167404 001126
2108 012474 042737 177760 001126
2109 012502 023737 001124 001126
2110 012510 001401
2111 012512 104025
2112 012514 005337 001124
2113 012520 100346

MOV SREG1,\$B0ADR ;SET UP REG ADRS 12
MOV JSREG1,\$BDDAT ;READ VECTOR SCALE
BIC #177760,\$BDDAT ;ONLY INTERESTED IN VECTOR SCALE
CMP \$GDDAT,\$BDDAT ;ARE THEY CORRECT?
BEQ 25 ;BR IF OK
ERROR 25 ;VECTOR SCALE INSTR ER
25: DEC \$GDDAT ;ADVANCE PATTERN
BPL 15 ;BR IF NOT DONE

*TEST 101 TEST THAT THE VECTOR SCALE DOES NOT CHANGE WHEN CHANGE ENA=0

2118 012522 000004
2119 012524 012777 020000 167364
2120 012532 012737 154037 027526
2121 012540 012737 000017 001124
2122 012546 012777 027526 167310
2123 012554 012737 154000 027530
2124 012562 005277 167276
2125 012566 013737 002076 001122
2126 012574 017737 167276 001126
2127 012602 042737 177760 001126
2128 012610 023737 001124 001126
2129 012616 001401
2130 012620 104025

TST101: SCOPE
MOV #BIT13,\$STKPT ;SET MAINT SWITCH #2
MOV #154037,BUFFER ;SET UP INSTR AT BUFFER-VECTOR SCALE ENA SET
MOV #17,\$GDDAT ;SET UP VECTOR SCALE EXPECTED
MOV #BUFFER,\$DPC ;START
MOV #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO VECTOR SCALE ENA
INC \$DPC ;RESUME
MOV SREG1,\$B0ADR ;SET UP REG ADRS 12
MOV JSREG1,\$BDDAT ;READ VECTOR SCALE
BIC #177760,\$BDDAT ;SAVE VECTOR SCALE ONLY
CMP \$GDDAT,\$BDDAT ;DOES IT STILL EQ 17
BEQ TST102 ;ADVANCE TO NEXT TEST IF OK
ERROR 25 ;VECTOR SCALE CHANGE ENABLE INSTR ER

*TEST 102 TEST THAT LOAD STATUS C INSTR CAN SET THE CHARACTER SCALE

2135 012622 000004
2136 012624 012777 020000 167264
2137 012632 012737 012654 001110
2138 012640 012737 154340 027526
2139 012646 012737 001400 001124
2140 012654 012777 027526 167202
2141 012662 013737 002076 001122
2142 012670 017737 167202 001126
2143 012676 042737 176377 001126
2144 012704 023737 001124 001126
2145 012712 001402
2146 012714 001411
2147 012716 104026
2148 012720 162737 000400 001124
2149 012726 100404
2150 012730 162737 000040 027526
2151 012736 000746

TST102: SCOPE
MOV #BIT13,\$STKPT ;SET MAINT SWITCH #2
MOV #15,\$SLPERR ;SET UP SCOPE LOOP ADRS
MOV #154340,BUFFER ;SET UP INSTR AT BUFFER
MOV #1400,\$GDDAT ;START WITH ALL ONES
15: MOV #BUFFER,\$DPC ;START
MOV SREG1,\$B0ADR ;SET UP REG ADRS 12
MOV JSREG1,\$BDDAT ;READ CHAR SCALE
BIC #176377,\$BDDAT ;SAVE CHAR SCALE ONLY
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 25 ;BR IF OK
BEQ TST103 ;ADVANCE TO NEXT TEST IF OK
ERROR 26 ;CHARACTER SCALE INSTR ER
25: SUB #400,\$GDDAT ;ADVANCE PATTERN
BMI TST103 ;ADVANCE TO NEXT TEST IF DONE
SUB #40,BUFFER ;ADVANCE PATTERN AT INSTR LOC
BR 15 ;DO NEXT CHAR SCALE

*TEST 103 TEST THAT THE CHAR SCALE DOES NOT CHANGE WHEN CHANGE ENA=0

2156 012740 000004
2157 012742 012777 020000 167146
2158 012750 012737 154340 027526
2159 012756 012737 001400 001124
2160 012764 012777 027526 167072
2161 012772 012737 154000 027530

TST103: SCOPE
MOV #BIT13,\$STKPT ;SET MAINT SWITCH #2
MOV #154340,BUFFER ;SET UP INSTR AT BUFFER-CHAR SCALE ENA SET
MOV #1400,\$GDDAT ;SET UP CHAR SCALE EXPECTED
MOV #BUFFER,\$DPC ;START
MOV #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO CHAR SCALE ENA

F04

```

2162 013000 005277 167060          INC      JOPC          ;RESUME
2163 013004 013737 002076 001122    MOV      SREG1,$BDADR ;SET UP REG ADRS 12
2164 013012 017737 167060 001126    MOV      JSREG1,$BDDAT ;READ CHAR SCALE
2165 013020 042737 176377 001126    BIC      #176377,$BDDAT ;SAVE CHAR SCALE ONLY
2166 013026 023737 001124 001126    CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
2167 013034 001401          BEQ      TST104       ;GO TO NEXT TEST IF NO CHANGE
2168 013036 104026          ERROR    26          ;CHARACTER SCALE CHANGE ENABLE INSTR ER
2169
2170
2171 ;*****
2172 ;*TEST 104 TEST THAT LOAD STATUS C INSTR CAN SET THE CHARACTER ROTATE
2173 ;*****
2173 013040 000004          TST104: SCOPE
2174 013042 012777 020000 167046    MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
2175 013050 012737 013072 001110    MOV      #1$, $LPERR   ;SET UP SCOPE LOOP ADRS
2176 013056 012737 155400 027526    MOV      #155400,BUFFER ;SET UP INSTR AT BUFFER
2177 013064 012737 002000 001124    MOV      #2000,$GDDAT  ;EXPECT CHAR ROTATE BIT
2178 013072 012777 027526 166764 1$: MOV      #BUFFER,$OPC  ;START
2179 013100 013737 002076 001122    MOV      SREG1,$BDADR ;SET UP REG ADRS 12
2180 013106 017737 166764 001126    MOV      JSREG1,$BDDAT ;READ CHAR ROTATE BIT
2181 013114 042737 175777 001126    BIC      #175777,$BDDAT ;SAVE CHAR ROTATE BIT ONLY
2182 013122 023737 001124 001126    CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
2183 013130 001401          BEQ      2$          ;BR IF OK
2184 013132 104026          ERROR    26          ;CHARACTER ROTATE INSTR ER
2185 013134 162737 002000 001124 2$: SUB      #2000,$GDDAT ;ADVANCE PATTERN
2186 013142 100404          BMI      TST105     ;ADVANCE TO NEXT TEST IF OK
2187 013144 162737 000400 027526    SUB      #400,BUFFER  ;ADVANCE PATTERN AT INSTR LOC
2188 013152 000747          BR       1$          ;NOW WRITE A ZERO INTO CHAR ROTATE
2189
2190
2191 ;*****
2192 ;*TEST 105 TEST THAT THE CHARACTER ROTATE DOES NOT CHANGE WHEN CHANGE ENA=0
2193 ;*****
2193 013154 000004          TST105: SCOPE
2194 013156 012777 020000 166732    MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
2195 013164 012737 155400 027526    MOV      #155400,BUFFER ;SET UP INSTR AT BUFFER CHAR ROTATE ENA SET
2196 013172 012737 002000 001124    MOV      #2000,$GDDAT  ;EXPECT CHAR ROTATE BIT
2197 013200 012777 027526 166656    MOV      #BUFFER,$OPC  ;START
2198 013206 012737 154000 027530    MOV      #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-WITHOUT CHAR ROTATE ENA
2199 013214 005277 166644          INC      JOPC          ;RESUME
2200 013220 013737 002076 001122    MOV      SREG1,$BDADR ;SET UP REG ADRS 12
2201 013226 017737 166644 001126    MOV      JSREG1,$BDDAT ;READ CHAR ROTATE BIT
2202 013234 042737 175777 001126    BIC      #175777,$BDDAT ;SAVE CHAR ROTATE BIT
2203 013242 023737 001124 001126    CMP      $GDDAT,$BDDAT ;IS IT SET?
2204 013250 001401          BEQ      TST106     ;ADVANCE TO NEXT IF OK
2205 013252 104026          ERROR    26          ;CHARACTER ROTATE CHANGE ENABLE INSTR ER
2206
2207
2208 ;*****
2209 ;*TEST 106 TEST THAT RESET WILL CLEAR ROTATE, CHAR SCALE, MENU & VECTOR SCALE
2210 ;*****
2211 013254 000004          TST106: SCOPE
2212 013256 012737 000040 001164    MOV      #40,$TIMES   ;DO 40 ITERATIONS
2213 013264 012777 020000 166624    MOV      #BIT13,$STKPT ;SET MAINT SW 2
2214 013272 012737 155733 027526    MOV      #155733,BUFFER ;SET UP 'LOAD STATUS C' INSTR
2215 013300 012737 170003 027530    MOV      #170003,BUFFER+2 ;SET UP MENU INSTR
2216 013306 012777 027526 166550    MOV      #BUFFER,$OPC  ;START
2217 013314 012737 000404 001124    MOV      #404,$GDDAT  ;EXPECT MSB OF VECTOR SCALE ONLY
    
```

```

2218 013322 005277 166536          INC      QDPC          ;PICK UP MENU INSTR
2219 013326 013737 002076 001122      MOV      SREG1,$BADDR ;SET UP REG ADRS 12
2220 013334 000005          RESET          ;CLR REG 12
2221 013336 017737 166534 001126      MOV      QSREG1,$BDDAT ;READ STATUS REG 1
2222 013344 023737 001124 001126      CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
2223 013352 001401          BEQ      TST107       ;NEXT TEST IF OK
2224 013354 104032          ERROR      32         ;RESET FAILED TO SET UP REG 12

```

;GRAPHPLOT INCREMENT REGISTER TESTS

```

;*****
;*TEST 107      LOAD 0-77 INTO GRAPHPLOT INCREMENT REGISTER
;*****
TST107: SCOPE

```

```

2231 013356 000004          MOV      #15,$LPERA    ;SET UP SCOPE LOOP ADRS
2232 013360 012737 013406 001110      MOV      #BIT13,$STKPT ;SET MAINT SW 2
2233 013366 012777 020000 166522      MOV      #174100,$BUFFER ;LOAD GRAPHPLOT INSTR
2234 013374 012737 174100 027526      CLR      $GDDAT        ;LD EXPECTED
2235 013402 005037 001124          MOV      $BUFFER,$DPC  ;START
2236 013406 012777 027526 166450 1S:      MOV      XPOS,$BADDR   ;SET UP REG ADRS 04
2237 013414 013737 002070 001122      MOV      QXPOS,$BDDAT  ;READ GRAPHPLOT
2238 013422 017737 166442 001126      BIC      #1777,$BDDAT  ;SAVE ONLY GRAPHPLOT
2239 013430 042737 001777 001126      CMP      $GDDAT,$BDDAT ;CORRECT?
2240 013436 023737 001124 001126      BEQ      2S           ;BR IF SO
2241 013444 001401          ERROR      7          ;GRAPHPLOT FAILED TO LOAD CORRECTLY
2242 013446 104007          INC      BUFFER        ;ADVANCE GRAPHPLOT AT INSTR
2243 013450 005237 027526          ADD      #2000,$GDDAT  ;ADVANCE EXPECTED
2244 013454 062737 002000 001124      BCC     1S           ;BR IF NOT COMBS TESTED
2245 013462 103351

```

```

;*****
;*TEST 110     CHECK THE GRAPHPLOT INCREMENT ENABLE REGISTER
;*****
TST110: SCOPE

```

```

2246
2247
2248
2249
2250 013464 000004          MOV      XPOS,$BADDR   ;SET UP REG ADRS 04
2251 013466 013737 002070 001122      MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
2252 013474 012777 020000 166414      MOV      #174100,$DBUF ;LOAD GRAPHPLOT COUNTER WITH 0
2253 013502 012777 174100 166324      MOV      $DBUF,$DPC    ;START DISPLAY
2254 013510 013777 002034 166346      MOV      #174077,$BUFFER+2 ;LOAD GRAPHPLOT NO ENABLE
2255 013516 012737 174077 027530      INC      $DPC          ;RESUME
2256 013524 005277 166334          CLR      $GDDAT        ;LOAD EXPECTED
2257 013530 005037 001124          MOV      QXPOS,$BDDAT  ;READ INCREMENT REGISTER
2258 013534 017737 166330 001126      BIC      #1777,$BDDAT  ;MASK TO BITS 15-10
2259 013542 042737 001777 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2260 013550 023737 001124 001126      BEQ      TST111       ;BR IF EQUAL
2261 013556 001401          ERROR      7          ;GRAPHPLOT REGISTER CHANGED WITHOUT
2262 013560 104007          ; THE ENABLE BEING SET
2263
2264
2265

```

```

;*****
;*TEST 111     TEST THAT RESET WILL CLEAR GRAPHPLOT INCREMENT
;*****
TST111: SCOPE

```

```

2266
2267
2268 013562 000004          MOV      #40,$TIMES    ;DO 40 ITERATIONS
2269 013564 012737 000040 001164      MOV      XPOS,$BADDR   ;SET UP REG ADRS 04
2270 013572 013737 002070 001122      MOV      #BIT13,$STKPT ;SET MAINT SW 2
2271 013600 012777 020000 166310      MOV      #174177,$BUFFER ;SET UP GRAPHPLOT INCREMENT INSTR
2272 013606 012737 174177 027526      MOV      $BUFFER,$DPC  ;START
2273 013614 012777 027526 166242

```

```

2274 013622 005037 001124 CLR $GDDAT ;EXPECT ZERO
2275 013626 000005 RESET ;CLR REG 04
2276 013630 017737 166234 001126 MOV $XPOS,$BDDAT ;READ REG 04
2277 013636 001401 BEQ TST112 ;NEXT TEST IF ZERO
2278 013640 104032 ERROR 32 ;RESET FAILED TO CLR GRAPHPLOT INC
2279
2280 ;*****
2281 ;*TEST 112 TEST THAT LOAD STATUS B INSTR CAN SET THE COLOR LEVEL
2282 ;*****
2283 013642 000004 TST112: SCOPE
2284 013644 012777 020000 166244 MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
2285 013652 012737 013674 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
2286 013660 012737 175600 027526 MOV #175600,BUFFER ;SET UP INSTR AT BUFFER
2287 013666 012737 000014 001124 MOV #14,$GDDAT ;EXPECT ALL ONES AT START
2288 013674 012777 027526 166162 1S: MOV #BUFFER,$DPC ;START
2289 013702 013737 002106 001122 MOV CONS,$BDADR ;SET UP REG ADRS 22
2290 013710 017737 166172 001126 MOV $CONS,$BDDAT ;READ COLOR LEVEL
2291 013716 042737 177763 001126 BIC #177763,$BDDAT ;ONLY INTERESTED IN THE COLOR LEVEL
2292 013724 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
2293 013732 001401 BEQ 2S ;BR IF OK
2294 013734 104035 ERROR 35 ;COLOR LEVEL INSTR ER
2295 013736 162737 000004 001124 2S: SUB #4,$GDDAT ;ADVANCE PATTERN
2296 013744 100404 BMI TST113 ;ADVANCE TO NEXT TEST IF ALL PATTERNS TESTED
2297 013746 162737 000200 027526 SUB #200,BUFFER ;SET UP NEXT PATTERN AT INSTR LOC
2298 013754 000747 BR 1S ;TRY NEXT PATTERN
2299
2300 ;*****
2301 ;*TEST 113 TEST THAT THE COLOR LEVEL DOES NOT CHANGE WHEN CHANGE ENA=0
2302 ;*****
2303 013756 000004 TST113: SCOPE
2304 013760 012777 020000 166130 MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
2305 013766 012737 175600 027526 MOV #175600,BUFFER ;SET UP INSTR AT BUFFER-COLOR LEVEL ENA SET
2306 013774 012737 000014 001124 MOV #14,$GDDAT ;SET UP COLOR LEVEL EXPECTED
2307 014002 012777 027526 166054 MOV #BUFFER,$DPC ;START
2308 014010 012737 174000 027530 MOV #174000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO COLOR LEVEL ENA
2309 014016 005277 166042 INC $DPC ;RESUME
2310 014022 013737 002106 001122 MOV CONS,$BDADR ;SET UP REG ADRS 22
2311 014030 017737 166052 001126 MOV $CONS,$BDDAT ;READ COLOR LEVEL
2312 014036 042737 177763 001126 BIC #177763,$BDDAT ;ONLY INTERESTED IN THE COLOR LEVEL
2313 014044 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
2314 014052 001401 BEQ TST114 ;ADVANCE TO NEXT TEST IF OK
2315 014054 104035 ERROR 35 ;COLOR LEVEL CHANGE ENABLE INSTR ER
2316
2317 ;*****
2318 ;*TEST 114 TEST THAT NOP INSTR CAN SET & RESET INTENSITY ENABLED-CONSOLE 0
2319 ;*****
2320 014056 000004 TST114: SCOPE
2321 014060 012777 020000 166030 MOV #BIT13,$STKPT ;SET MAINT SWITCH #2
2322 014066 012737 014110 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
2323 014074 012737 100000 001124 MOV #100000,$GDDAT ;EXPECT IT TO BE SET INITIALLY
2324 014102 012737 164300 027526 MOV #164300,BUFFER ;SET UP INSTR LOC
2325 014110 012777 027526 165746 1S: MOV #BUFFER,$DPC ;START
2326 014116 013737 002106 001122 MOV CONS,$BDADR ;SET UP REG ADRS 22
2327 014124 017737 165756 001126 MOV $CONS,$BDDAT ;READ CONSOLE 0 INT ENA REG
2328 014132 042737 077777 001126 BIC #77777,$BDDAT ;SAVE ONLY INTENSITY ENABLE BIT
2329 014140 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?

```

```

2330 014146 001401          BEQ      25          ;BR IF SO
2331 014150 104036          ERROR    36          ;CONSOLE 0 INTENSITY ENABLE INSTR ER
2332 014152 162737 100000 001124 25:  SUB      #100000,$GDDAT ;GO TO RESET STATE
2333 014160 100404          BMI      TST115     ;GO TO NEXT TEST IF BOTH STATES TESTED
2334 014162 042737 000100 027526  BIC      #100,BUFFER ;CLR INT ENABLED BIT AT INSTR LOC
2335 014170 000747          BR       15          ;GO TRY RESET STATE
2336
2337 ;*****
2338 ;*TEST 115 TEST THAT CONSOLE 0 INTENSITY ENABLED DOES NOT RESET WHEN ENA=0
2339 ;*****
2340 014172 000004          TST115: SCOPE
2341 014174 012777 020000 165714  MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2342 014202 012737 100000 001124  MOV      #100000,$GDDAT ;EXPECT INTENSITY TO BE SET
2343 014210 012737 164300 027526  MOV      #164300,BUFFER ;SET UP INSTR
2344 014216 012777 027526 165640  MOV      #BUFFER,2DPC ;START
2345 014224 012737 164000 027530  MOV      #164000,BUFFER+2 ;SET UP INSTR WITH ENA=0
2346 014232 005277 165626          INC      2DPC ;ADVANCE TO NEXT INSTR
2347 014236 013737 002106 001122  MOV      CONS,$BDADR ;SET UP REG ADRS 22
2348 014244 005777 165636          TST      2CONS ;IS INTENSITY STILL SET?
2349 014250 100403          BMI      TST116     ;ADVANCE TO NEXT TEST IF STILL SET
2350 014252 005037 001126          CLR      $BDDAT ;INDICATE IT WAS CLEARED
2351 014256 104036          ERROR    36          ;CONSOLE 0 INTENSITY CHANGE ENABLE INSTR ER
2352
2353 ;*****
2354 ;*TEST 116 TEST THAT NOP INSTR CAN SET & RESET INTENSITY ENABLED-CONSOLE 1
2355 ;*****
2356 014260 000004          TST116: SCOPE
2357 014262 012777 020000 165626  MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2358 014270 013737 014312 001110  MOV      15,$LPERR ;SET UP SCOPE LOOP ADRS
2359 014276 012737 001000 001124  MOV      #1000,$GDDAT ;EXPECT IT TO BE SET INITIALLY
2360 014304 012737 164700 027526  MOV      #164700,BUFFER ;SET UP INSTR LOCATION
2361 014312 012777 027526 165544 15:  MOV      #BUFFER,2DPC ;START
2362 014320 013737 002106 001122  MOV      CONS,$BDADR ;SET UP REG ADRS 22
2363 014326 017737 165554 001126  MOV      2CONS,$BDDAT ;READ CONSOLE 1 INT ENA REG
2364 014334 042737 176777 001126  BIC      #176777,$BDDAT ;SAVE ONLY INT ENA
2365 014342 023737 001124 001126  CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
2366 014350 001401          BEQ      25          ;BR IF OK
2367 014352 104036          ERROR    36          ;CONSOLE 1 INTENSITY ENABLE INSTR ER
2368 014354 162737 001000 001124 25:  SUB      #1000,$GDDAT ;GO TO RESET STATE
2369 014362 100404          BMI      TST117     ;GO TO NEXT TEST IF BOTH STATES TESTED
2370 014364 042737 000100 027526  BIC      #100,BUFFER ;CLR INT ENABLED BIT AT INSTR LOC
2371 014372 000747          BR       15          ;GO TRY RESET STATE
2372
2373 ;*****
2374 ;*TEST 117 TEST THAT CONSOLE 1 INTENSITY ENABLED DOES NOT RESET WHEN ENA=0
2375 ;*****
2376 014374 000004          TST117: SCOPE
2377 014376 012777 020000 165512  MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2378 014404 012737 001000 001124  MOV      #1000,$GDDAT ;EXPECT INTENSITY TO BE SET
2379 014412 012737 164700 027526  MOV      #164700,BUFFER ;SET UP INSTR
2380 014420 012777 027526 165436  MOV      #BUFFER,2DPC ;START
2381 014426 012737 164400 027530  MOV      #164400,BUFFER+2 ;SET UP INSTR WITH ENA=0
2382 014434 005277 165424          INC      2DPC ;ADVANCE TO NEXT INSTR
2383 014440 013737 002106 001122  MOV      CONS,$BDADR ;SET UP REG ADRS 22
2384 014446 032777 001000 165432  BIT      #1000,2CONS ;IS INTENSITY STILL SET?
2385 014454 001003          BNE     TST120     ;ADVANCE TO NEXT TEST IF STILL SET

```


2386	014456	005037	001126	CLR	\$BDDAT		:INDICATE IT WAS CLEARED
2387	014462	104036		ERROR	36		:CONSOLE 1 INTENSITY CHANGE ENABLE INSTR ER

```

*****
:TEST 120 TEST THAT NOP INSTR CAN SET & RESET L.P. INTR ENABLED-CONSOLE 0
*****

```

2392	014464	000004		TST120:	SCOPE		
2393	014466	012777	020000	MOV	#BIT13,2STKPT		:SET MAINT SWITCH #2
2394	014474	012737	014516	MOV	#1\$,SLPERR		:SET UP SCOPE LOOP ADRS
2395	014502	012737	004000	MOV	#4000,\$GDDAT		:EXPECT IT TO BE SET INITIALLY
2396	014510	012737	164060	MOV	#164060,BUFFER		:SET UP INSTR LOC
2397	014516	012777	027526	1\$:	MOV	#BUFFER,2DPC	:START
2398	014524	013737	002106	MOV	CONS,\$BDADR		:SET UP REG ADRS 22
2399	014532	017737	165350	MOV	2CONS,\$BDDAT		:READ CONSOLE 0 LP INTR ENABLED REG
2400	014540	042737	173777	BIC	#173777,\$BDDAT		:SAVE ONLY LP INTR ENABLED BIT
2401	014546	023737	001124	CMP	\$GDDAT,\$BDDAT		:IS IT CORRECT?
2402	014554	001401		BEQ	2\$:BR IF OK
2403	014556	104042		ERROR	42		:CONSOLE 0 L.P. INTR ENABLE INSTR ER
2404	014560	162737	004000	2\$:	SUB	#4000,\$GDDAT	:GO TO RESET STATE
2405	014566	100404		BMI	TST121		:ADVANCE TO NEXT TEST IF BOTH STATES TESTED
2406	014570	042737	000020	BIC	#20,BUFFER		:CLR THE ENABLED BIT AT INSTR LOC
2407	014576	000747		BR	1\$:GO TRY RESET STATE

```

*****
:TEST 121 TEST THAT CONSOLE 0 L.P. INTR ENABLED DOES NOT RESET WHEN ENA=0
*****

```

2412	014600	000004		TST121:	SCOPE		
2413	014602	012777	020000	MOV	#BIT13,2STKPT		:SET MAINT SWITCH #2
2414	014610	012737	004000	MOV	#4000,\$GDDAT		:EXPECT L.P. INTR ENABLED TO BE SET
2415	014616	012737	164060	MOV	#164060,BUFFER		:SET UP INSTR LOC
2416	014624	012777	027526	MOV	#BUFFER,2DPC		:START
2417	014632	012737	164000	MOV	#164000,BUFFER+2		:SET UP INSTR WITH ENA=0
2418	014640	005277	165220	INC	2DPC		:ADVANCE TO NEXT INSTR
2419	014644	013737	002106	MOV	CONS,\$BDADR		:SET UP REG ADRS 22
2420	014652	032777	004000	BIT	#4000,2CONS		:IS L.P. INTR ENABLED STILL SET?
2421	014660	001003		BNE	TST122		:ADVANCE TO NEXT TEST IF STILL SET
2422	014662	005037	001126	CLR	\$BDDAT		:INDICATE IT WAS CLEARED
2423	014666	104042		ERROR	42		:CONSOLE 0 L.P. CHANGE INTR ENABLE INSTR ER

```

*****
:TEST 122 TEST THAT NOP INSTR CAN SET & RESET L.P. INTR ENABLED-CONSOLE 1
*****

```

2428	014670	000004		TST122:	SCOPE		
2429	014672	012777	020000	MOV	#BIT13,2STKPT		:SET MAINT SWITCH #2
2430	014700	012737	014722	MOV	#1\$,SLPERR		:SET UP SCOPE LOOP ADRS
2431	014706	012737	000040	MOV	#40,\$GDDAT		:EXPECT IT TO BE SET INITIALLY
2432	014714	012737	164460	MOV	#164460,BUFFER		:SET UP INSTR LOC
2433	014722	012777	027526	1\$:	MOV	#BUFFER,2DPC	:START
2434	014730	013737	002106	MOV	CONS,\$BDADR		:SET UP REG ADRS 22
2435	014736	017737	165144	MOV	2CONS,\$BDDAT		:READ CONSOLE 1 LP INTR ENABLED REG
2436	014744	042737	177737	BIC	#177737,\$BDDAT		:SAVE ONLY LP INTR ENABLED BIT
2437	014752	023737	001124	CMP	\$GDDAT,\$BDDAT		:IS IT CORRECT?
2438	014760	001401		BEQ	2\$:BR IF OK
2439	014762	104042		ERROR	42		:CONSOLE 1 L.P. INTR ENABLE INSTR ER
2440	014764	162737	000040	2\$:	SUB	#40,\$GDDAT	:GO TO RESET STATE
2441	014772	100404		BMI	TST123		:ADVANCE TO NEXT TEST IF BOTH STATES TESTED

```

2442 014774 042737 000020 027526
2443 015002 000747
2444
2445
2446
2447
2448 015004 000004
2449 015006 012777 020000 165102
2450 015014 012737 000040 001124
2451 015022 012737 164460 027526
2452 015030 012777 027526 165026
2453 015036 012737 164400 027530
2454 015044 005277 165014
2455 015050 013737 002106 001122
2456 015056 032777 000040 165022
2457 015064 001003
2458 015066 005037 001126
2459 015072 104042
2460
2461
2462
2463
2464 015074 000004
2465 015076 012777 020000 165012
2466 015104 012737 015126 001110
2467 015112 012737 002000 001124
2468 015120 012737 164014 027526
2469 015126 012777 027526 164730
2470 015134 013737 002106 001122
2471 015142 017737 164740 001126
2472 015150 042737 175777 001126
2473 015156 023737 001124 001126
2474 015164 001401
2475 015166 104043
2476 015170 162737 002000 001124
2477 015176 100404
2478 015200 042737 000004 027526
2479 015206 000747
2480
2481
2482
2483
2484 015210 000004
2485 015212 012777 020000 164676
2486 015220 012737 002000 001124
2487 015226 012737 164014 027526
2488 015234 012777 027526 164622
2489 015242 012737 164000 027530
2490 015250 005277 164610
2491 015254 013737 002106 001122
2492 015262 032777 002000 164616
2493 015270 001003
2494 015272 005037 001126
2495 015276 104043
2496
2497

```

```

BIC #20,BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
BR 1$ ;GO TRY RESET STATE

;*****
;*TEST 123 TEST THAT CONSOLE 1 L.P. INTR ENABLED DOES NOT RESET WHEN ENA=0
;*****
TST123: SCOPE
MOV #BIT13,JUSTKPT ;SET MAINT SWITCH #2
MOV #40,$GDDAT ;EXPECT L.P. INTR ENABLED TO BE SET
MOV #164460,BUFFER ;SET UP INSTR LOC
MOV #BUFFER,ADPC ;START
MOV #164400,BUFFER+2 ;SET UP INSTR WITH ENA=0
INC ADPC ;ADVANCE TO NEXT INSTR
MOV CONS,$BDADR ;SET UP REG ADRS 22
BIT #40,CONS ;IS L.P. INTR ENABLED STILL SET?
BNE TST124 ;ADVANCE TO NEXT TEST IF STILL SET
CLR $BDDAT ;INDICATE IT WAS CLEARED
ERROR 42 ;CONSOLE 1 L.P. CHANGE INTR ENABLE INSTR ER

;*****
;*TEST 124 TEST THAT NOP INSTR CAN SET & RESET L.P. SWITCH INTR ENABLED-CONSOLE 0
;*****
TST124: SCOPE
MOV #BIT13,JUSTKPT ;SET MAINT SWITCH #2
MOV #1$,SLPERR ;SET UP SCOPE LOOP ADRS
MOV #2000,$GDDAT ;EXPECT LP SW INTR ENABLED BIT
MOV #164014,BUFFER ;SET UP INSTR
1$: MOV #BUFFER,ADPC ;START
MOV CONS,$BDADR ;SET UP REG ADRS 22
MOV ADCONS,$BDDAT ;READ L.P. SWITCH INTR ENABLED REG
BIC #175777,$BDDAT ;SAVE ONLY L.P. SWITCH INTR ENABLED BIT
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 2$ ;BR IF OK
ERROR 43 ;CONSOLE 0 L.P. SW INTR ENABLE INSTR ER
2$: SUB #2000,$GDDAT ;GO TO RESET STATE
BMI TST125 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
BIC #4,BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
BR 1$ ;GO TRY RESET STATE

;*****
;*TEST 125 TEST THAT L.P. SWITCH INTR ENABLED DOES NOT RESET WHEN ENA=0
;*****
TST125: SCOPE
MOV #BIT13,JUSTKPT ;SET MAINT SWITCH #2
MOV #2000,$GDDAT ;EXPECT L.P. SWITCH INTR ENABLED TO BE SET
MOV #164014,BUFFER ;SET UP INSTR LOC
MOV #BUFFER,ADPC ;START
MOV #164000,BUFFER+2 ;SET UP INSTR WITH ENA=0
INC ADPC ;ADVANCE TO NEXT INSTR
MOV CONS,$BDADR ;SET UP REG ADRS 22
BIT #2000,ADCONS ;IS L.P. SWITCH INTR ENABLED STILL SET?
BNE TST126 ;ADVANCE TO NEXT TEST IF STILL SET
CLR $BDDAT ;INDICATE IT WAS CLEARED
ERROR 43 ;CONSOLE 0 L.P. SW CHANGE INTR ENABLE INSTR ER

;*****

```

```

2498 ;*TEST 126 TEST THAT NOP INSTR CAN SET & RESET L.P. SWITCH INTR ENABLED-CONSOLE 1
2499 ;*****
2500 TST126: SCOPE
2501 MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
2502 MOV #15, $LPERR ;SET UP SCOPE LOOP ADRS
2503 MOV #20, $GDDAT ;EXPECT L.P. SWITCH INTR ENABLED BIT
2504 MOV #164414, BUFFER ;SET UP INSTR
2505 1$: MOV #BUFFER, @DPC ;START
2506 MOV CONS, $BDADR ;SET UP REG ADRS 22
2507 MOV @CONS, $BDDAT ;READ L.P. SWITCH INTR ENABLED REG
2508 BIC #177757, $BDDAT ;SAVE ONLY L.P. SWITCH INTR ENABLED BIT
2509 CMP $GDDAT, $BDDAT ;IS IT CORRECT?
2510 BEQ 2$ ;BR IF OK
2511 ERROR 43 ;CONSOLE 1 L.P. SW INTR ENABLE INSTR ER
2512 2$: SUB #20, $GDDAT ;GO TO RESET STATE
2513 BMI TST127 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
2514 BIC #4, BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
2515 BR 1$ ;GO TRY RESET STATE
2516
2517 ;*****
2518 ;*TEST 127 TEST THAT L.P. SWITCH INTR ENABLED DOES NOT RESET WHEN ENA=0
2519 ;*****
2520 TST127: SCOPE
2521 MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
2522 MOV #20, $GDDAT ;EXPECT L.P. SW INTR ENABLED TO BE SET
2523 MOV #164414, BUFFER ;SET UP INSTR LOC
2524 MOV #BUFFER, @DPC ;START
2525 MOV #164400, BUFFER+2 ;SET UP INSTR LOC WITH ENA=0
2526 INC @DPC ;ADVANCE TO NEXT INSTR
2527 MOV CONS, $BDADR ;SET UP REG ADRS 22
2528 BIT #20, @CONS ;IS L.P. SW INTR ENABLED STILL SET?
2529 BNE TST130 ;ADVANCE TO NEXT TEST IF STILL SET
2530 CLR $BDDAT ;INDICATE IT WAS CLEARED
2531 ERROR 43 ;CONSOLE 1 L.P. SW CHANGE INTR ENABLE INSTR ER
2532
2533 ;*****
2534 ;*TEST 130 TEST THAT RESET WILL CLEAR L.P. INTR ENABLES & COLOR
2535 ;*****
2536 TST130: SCOPE
2537 MOV #40, $TIMES ;DO 40 ITERATIONS
2538 MOV CONS, $BDADR ;SET UP REG ADRS 22
2539 MOV #BIT13, @STKPT ;SET MAINT SW 2
2540 MOV #164274, BUFFER ;SET UP INSTR FOR CONSOLE 0 L.P. INTR ENAB
2541 MOV #BUFFER, @DPC ;START
2542 MOV #164774, BUFFER+2 ;CONS 1 L.P. INTR ENA
2543 INC @DPC ;RESUME
2544 MOV #175600, BUFFER+4 ;COLOR LEVEL
2545 INC @DPC ;RESUME
2546 MOV #BIT15, $GDDAT ;EXPECT INTENSITY ENABLE CONSOLE 0 ONLY
2547 RESET ;CLR FLAGS
2548 MOV @CONS, $BDDAT ;READ FLAGS
2549 CMP $GDDAT, $BDDAT ;IS REG 22 CORRECT?
2550 BEQ TST131 ;NEXT TEST IF SO
2551 ERROR 32 ;RESET FAILED TO SET UP REG 22
2552
2553

```

```

2554
2555
2556
2557 015620 000004
2558 015622 012777 020000 164266
2559 015630 012777 160000 164176
2560 015636 012737 025252 001124
2561 015644 013777 002034 164212
2562 015652 013737 001124 027530
2563 015660 005277 164200
2564 015664 013737 002064 001122
2565 015672 017737 164166 001126
2566 015700 023737 001124 001126
2567 015706 001401
2568 015710 104010
2569
2570
2571
2572
2573
2574 015712 000004
2575 015714 012777 020000 164174
2576 015722 012777 160000 164104
2577 015730 012737 012524 001124
2578 015736 013777 002034 164120
2579 015744 013737 001124 027530
2580 015752 005277 164106
2581 015756 013737 002064 001122
2582 015764 017737 164074 001126
2583 015772 023737 001124 001126
2584 016000 001401
2585 016002 104010
2586
2587
2588
2589
2590
2591 016004 000004
2592 016006 012737 000020 001164
2593 016014 013737 025232 001124
2594 016022 012777 020000 164066
2595 016030 012737 160000 027526
2596 016036 012777 027526 164020 1$:
2597 016044 013737 001124 027530
2598 016052 005277 164006
2599 016056 013737 002064 001122
2600 016064 017737 163774 001126
2601 016072 023737 001124 001126
2602 016100 001401
2603 016102 104010
2604
2605 016104 162737 000002 001124 2$:
2606 016112 103351
2607
2608
2609

```

```

*****
*TEST 131 JUMP TEST TO LOC. 25252
*****
TST131: SCOPE
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #160000, JDBUF ;MOVE DJUMP INTO THE BUFFER
MOV #25252, $GDDAT ;MOVE 25252 INTO THE NEXT LOC.
MOV DBUF, JDPCC ;START THE DISPLAY
MOV $GDDAT, BUFFER+2 ;LD ADRS
INC JDPCC ;SINGLE STEP THE DISPLAY
MOV DPC, $BDADR ;SET UP REG ADRS 00
MOV JDPCC, $BDDAT ;READ THE DISPLAY P.C.
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST132 ;;BR IF EQUAL
ERROR 10 ;DJUMP FAILED TO LOAD THE NEW
;DISPLAY P.C. PROPERLY

*****
*TEST 132 JUMP TEST TO LOC. 12524
*****
TST132: SCOPE
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #160000, JDBUF ;MOVE DJUMP INTO THE BUFFER
MOV #12524, $GDDAT ;MOVE 12524 INTO THE NEXT LOC.
MOV DBUF, JDPCC ;START THE DISPLAY
MOV $GDDAT, BUFFER+2 ;LD ADRS
INC JDPCC ;SINGLE STEP THE DISPLAY
MOV DPC, $BDADR ;SET UP REG ADRS 00
MOV JDPCC, $BDDAT ;READ THE DISPLAY P.C.
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST133 ;;BR IF EQUAL
ERROR 10 ;DJUMP FAILED TO LOAD THE NEW
;DISPLAY P.C. PROPERLY

*****
*TEST 133 JUMP TO EACH EXISTING LOCATION IN LOWER BANK
*****
TST133: SCOPE
MOV #20, $TIMES ;;DO 20 ITERATIONS
MOV $LSTAD, $GDDAT ;LOAD JUMP DESTINATION
MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
MOV #160000, BUFFER ;LOAD "DISPLAY JUMP" INST
1$: MOV #BUFFER, JDPCC ;START DISPLAY
MOV $GDDAT, BUFFER+2 ;SET UP ADRS
INC JDPCC ;SINGLE STEP THE DISPLAY
MOV DPC, $BDADR ;SET UP REG ADRS 00
MOV JDPCC, $BDDAT ;READ DESTINATION D.P.C.
CMP $GDDAT, $BDDAT ;TEST IF EXPECTED
BEQ 2$ ;;BR IF SAME
ERROR 10 ;JUMP TO LOCATION "N" FAILED
;N=ADDRESS IN $GDDAT
2$: SUB #2, $GDDAT ;ADJUST EXPECTED ADDRESS
BCC 1$ ;BR IF MORE LOCATIONS TO TEST

*****
*TEST 134 TEST 'JUMP RELATIVE' INSTR (BIT 8=0) AT LOC 'BUFFER' & VERIFY NEW DPC AD

```

```

2610
2611 016114 000004          *****
2612 016116 012737 016140 001110  †TST134: SCOPE
2613 016124 012777 020000 163764  MOV #1$, $LPERR ;SET UP SCOPE LOOP ADRS
2614 016132 012700 000400  MOV #BIT13, $STKPT ;SET MAINT SWITCH #2
2615 016136 000423  BR #400, R0 ;SET UP MSB FIRST
2616 016140 012737 161000 027526 1$: MOV #161000, BUFFER ;GO CALCULATE ADRS
2617 016146 050037 027526  BIS RO, BUFFER ;SET UP JUMP RELATIVE INSTR
2618 016152 012777 027526 163704  ;SET UP RELATIVE ADRS PATTERN
2619 016160 013737 002064 001122  MOV #BUFFER, $DPC ;START
2620 016166 017737 163672 001126  MOV $DPC, $BDDADR ;SET UP REG ADRS 00
2621 016174 023737 001124 001126  MOV $DPC, $BDDAT ;READ NEW DPC ADRS
2622 016202 001401  BEQ $GDDAT, $BDDAT ;DOES IT EQUAL THE CALCULATED ADRS?
2623 016204 104011  ERROR 11 ;BR IF OK
2624 016206 006200  2$: ASR RO ;JUMP RELATIVE POSITIVE FAILED
2625 016210 103410  BCS TST135 ;SHIFT RIGHT TO NEXT BIT
2626 016212 010037 001124  MOV RO, $GDDAT ;ADVANCE TO NEXT TEST IF DONE
2627 016216 006337 001124  ASL $GDDAT ;SET UP OFFSET VALUE
2628 016222 062737 027530 001124  ADD #BUFFER+2, $GDDAT ;MAKE INTO WORD OFFSET
2629 016230 000743  BR 1$ ;ADD IN ADRS OF INSTR PLUS 2
2630 ;TRY THIS RELATIVE ADRS

```

```

2631 *****
2632 *TEST 135 TEST 'JUMP RELATIVE' INSTR (BIT 8=1) AT LOC 'BUFFER' & VERIFY NEW DPC
2633 *****
2634 016232 000004          †TST135: SCOPE
2635 016234 012737 016256 001110  MOV #1$, $LPERR ;SET UP SCOPE LOOP ADRS
2636 016242 012777 020000 163646  MOV #BIT13, $STKPT ;SET MAINT SWITCH #2
2637 016250 012700 000400  MOV #400, R0 ;SET UP MSB FIRST
2638 016254 000423  BR 2$ ;GO CALCULATE ADRS
2639 016256 012737 161400 027526 1$: MOV #161400, BUFFER ;SET UP JUMP RELATIVE INSTR
2640 016264 050037 027526  BIS RO, BUFFER ;SET UP RELATIVE ADRS PATTERN
2641 016270 012777 027526 163566  MOV #BUFFER, $DPC ;START
2642 016276 013737 002064 001122  MOV $DPC, $BDDADR ;SET UP REG ADRS 00
2643 016304 017737 163554 001126  MOV $DPC, $BDDAT ;READ NEW DPC ADRS
2644 016312 023737 001124 001126  CMP $GDDAT, $BDDAT ;DOES IT EQUAL THE CALCULATED VALUE?
2645 016320 001401  BEQ 2$ ;BR IF OK
2646 016322 104011  ERROR 11 ;JUMP RELATIVE NEGATIVE FAILED
2647 016324 006200  2$: ASR RO ;SHIFT RIGHT TO NEXT BIT
2648 016326 103413  BCS TST136 ;ADVANCE TO NEXT TEST IF DONE
2649 016330 012737 027530 001124  MOV #BUFFER+2, $GDDAT ;SET UP CURRENT ADRS
2650 016336 010001  MOV RO, R1 ;GET RELATIVE ADRS VALUE
2651 016340 005401  NEG R1 ;MAKE NEG
2652 016342 006301  ASL R1 ;CORRECT FOR WORD OFFSET
2653 016344 042701 177000  BIC #177000, R1 ;GET RID OF MSBS
2654 016350 160137 001124  SUB R1, $GDDAT ;SUB OFFSET FROM DPC VALUE
2655 016354 000740  BR 1$ ;TRY NEXT RELATIVE ADRS

```

```

2656 *****
2657 *TEST 136 TEST THAT JUMP ABS INSTR WILL SETUP DPC WITH A FLOATING ONE
2658 *****
2659 †TST136: SCOPE
2660 016356 000004  MOV #BIT13, $STKPT ;SET MAINT SWITCH #2
2661 016360 012777 020000 163530  MOV #160000, BUFFER ;SET UP JUMP ABS INSTRUCTION AT 'BUFFER'
2662 016366 012737 160000 027526  CLR $GDDAT ;EXPECT ZEROS FIRST TIME
2663 016374 005037 001124  MOV #1, BUFFER+2 ;SET UP ABS ADRS TO #1
2664 016400 012737 000001 027530 1$: MOV #BUFFER, $DPC ;START
2665 016406 012777 027526 163450

```

B05

```

2666 016414 012737 016406 001110      MOV      #15,$LPERR      ;SET UP SCOPE LOOP ADRS
2667 016422 013737 002064 001122      MOV      DPC,$BDADR     ;SET UP REG ADRS 00
2668 016430 005277 163430      INC      JDPc           ;PICK UP ABS ADRS
2669 016434 000240      NOP                      ;ALLOW TIME FOR 2ND NPR
2670 016436 017737 163422 001126      MOV      JDPc,$BDDAT    ;GET ADRS JUMPED TO
2671 016444 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2672 016452 001401      BEQ      25            ;BR IF OK
2673 016454 104010      ERROR   10            ;JUMP ABSOLUTE FAILED (FLOAT A 1 ADDRESS)
2674 016456 006337 027530      25:  ASL      BUFFER+2      ;ADVANCE ONE BIT LEFT
2675 016462 103404      BCS      TST137        ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
2676 016464 013737 027530 001124      MOV      BUFFER+2,$GDDAT ;SET UP EXPECTED ADRS
2677 016472 000745      BR       15            ;TRY NEXT BIT

```

```

*****
;*TEST 137 TEST THAT JUMP ABS INSTR WILL SETUP DPC WITH A FLOATING ZERO
*****

```

```

2682 016474 000004      TST137: SCOPE
2683 016476 012737 016526 001110      MOV      #15,$LPERR     ;SET UP SCOPE LOOP ADRS
2684 016504 012777 020000 163404      MOV      #BIT13,$STKPT  ;SET MAINT SWITCH #2
2685 016512 012737 160000 027526      MOV      #160000,BUFFER ;SET UP JUMP ABS INSTR AT 'BUFFER'
2686 016520 012737 177774 001124      MOV      #177774,$GDDAT ;SET LSB TO ZERO
2687 016526 012777 027526 163330      15:  MOV      $BUFFER,$JDPc ;START
2688 016534 013737 001124 027530      MOV      $GDDAT,BUFFER+2 ;SET UP ADRS
2689 016542 005277 163316      INC      JDPc           ;PICK UP ABS ADRS
2690 016546 013737 002064 001122      MOV      DPC,$BDADR     ;SET UP REG ADRS 00
2691 016554 017737 163304 001126      MOV      JDPc,$BDDAT    ;GET ADRS JUMPED TO
2692 016562 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2693 016570 001401      BEQ      25            ;BR IF OK
2694 016572 104010      ERROR   10            ;JUMP ABSOLUTE FAILED (TO A FLOATING ZERO)
2695 016574 006337 001124      25:  ASL      $GDDAT        ;SHIFT ZERO BIT LEFT
2696 016600 103004      BCC      TST140        ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
2697 016602 062737 000002 001124      ADD      #2,$GDDAT      ;REPLACE LSB
2698 016610 000746      BR       15            ;TRY NEXT BIT

```

```

*****
;*TEST 140 TEST THAT THE 'JSR ABSOLUTE' INSTR WILL SET 'JSR' BIT IN REG 32
*****

```

```

2703 016612 000004      TST140: SCOPE
2704 016614 012777 020000 163274      MOV      #BIT13,$STKPT  ;SET MAINT SWITCH #2
2705 016622 012737 162000 027526      MOV      #162000,BUFFER ;SET UP JSR ABS INSTR
2706 016630 005037 027530      CLR      BUFFER+2       ;CLR ABS ADRS LOC
2707 016634 012737 001000 001124      MOV      #BIT9,$GDDAT   ;EXPECT JSR BIT
2708 016642 012777 027526 163214      MOV      $BUFFER,$JDPc  ;START
2709 016650 013737 002116 001122      MOV      $STKPT,$BDADR  ;SET UP REG ADRS 32
2710 016656 032777 001000 163232      BIT      #BIT9,$STKPT   ;IS JSR BIT SET?
2711 016664 001003      BNE      TST141        ;ADVANCE TO NEXT TEST IF OK
2712 016666 005037 001126      CLR      $BDDAT        ;SHOW THAT IT WAS ZERO
2713 016672 104012      ERROR   12            ;CALL ERROR ROUTINE USING MSG 23

```

```

*****
;*TEST 141 TEST 'JSR RELATIVE' INSTR (BIT 8=0) AT LOC 'BUFFER' & VERIFY NEW DPC ADR
*****

```

```

2718 016674 000004      TST141: SCOPE
2719 016676 012777 020000 163212      MOV      #BIT13,$STKPT  ;SET MAINT SWITCH #2
2720 016704 012737 016720 001110      MOV      #15,$LPERR     ;SET UP SCOPE LOOP ADRS
2721 016712 012700 000400      MOV      #400,$R0       ;SET UP MSB FIRST

```

C05

```

2722 016716 000426          BH      25          ;GO CALCULATE ADRS
2723 016720 052777 000040 163170 15:  BIS    #40,STKPT    ;RESET STK PTR
2724 016726 012737 163000 027526      MOV    #163000,BUFFER ;SET UP JSR RELATIVE INSTR
2725 016734 050037 027526          BIS    RO,BUFFER     ;SET UP RELATIVE ADRS PATTERN
2726 016740 012777 027526 163116      MOV    #BUFFER,DPC   ;START
2727 016746 013737 002064 001122      MOV    DPC,$BDADR    ;SET UP REG ADRS 00
2728 016754 017737 163104 001126      MOV    DPC,$BDDAT    ;READ NEW DPC ADRS
2729 016762 023737 001124 001126      CMP    $GDDAT,$BDDAT ;DOES IT EQUAL THE CALCULATED ADRS?
2730 016770 001401          BEQ    25           ;BR IF OK
2731 016772 104013          ERROR  13          ;JSR RELATIVE "POSITIVE" FAILED
2732 016774 006200          ASR    RO           ;SHIFT RIGHT TO NEXT BIT
2733 016776 103410          BCS    TST142      ;ADVANCE TO NEXT TEST IF DONE
2734 017000 010037 001124      MOV    RO,$GDDAT    ;SET UP OFFSET VALUE
2735 017004 006337 001124      ASL    $GDDAT       ;MAKE INTO WORD OFFSET
2736 017010 062737 027530 001124      ADD    #BUFFER+2,$GDDAT ;ADD IN ADRS OF INSTR PLUS 2
2737 017016 000740          BR     15          ;TRY THIS RELATIVE ADRS
2738
2739
2740          ;*****
2741          ;*TEST 142 TEST 'JSR RELATIVE' INSTR (BIT 8=1) AT LOC 'BUFFER' & VERIFY NEW DPC ADR
2742          ;*****
2742 017020 000004          TST142: SCOPE
2743 017022 012777 020000 163066      MOV    #BIT13,STKPT ;SET MAINT SWITCH #2
2744 017030 012737 017044 001110      MOV    #1,$SLPERR   ;SET UP SCOPE LOOP ADRS
2745 017036 012700 000400      MOV    #400,RO      ;SET UP MSB FIRST
2746 017042 000426          BR     25          ;GO CALCULATE ADRS
2747 017044 052777 000040 163044 15:  BIS    #40,STKPT    ;RESET STK PTR
2748 017052 012737 163400 027526      MOV    #163400,BUFFER ;SET UP JSR RELATIVE INSTR
2749 017060 050037 027526          BIS    RO,BUFFER     ;SET UP RELATIVE ADRS PATTERN
2750 017064 012777 027526 162772      MOV    #BUFFER,DPC   ;START
2751 017072 013737 002064 001122      MOV    DPC,$BDADR    ;SET UP REG ADRS 00
2752 017100 017737 162760 001126      MOV    DPC,$BDDAT    ;READ NEW DPC ADRS
2753 017106 023737 001124 001126      CMP    $GDDAT,$BDDAT ;DOES IT EQUAL THE CALCULATED VALUE?
2754 017114 001401          BEQ    25           ;BR IF OK
2755 017116 104013          ERROR  13          ;JSR RELATIVE NEGATIVE TO ADDRESS FAILED
2756 017120 006200          ASR    RO           ;SHIFT RIGHT TO NEXT BIT
2757 017122 103413          BCS    TST143      ;ADVANCE TO NEXT TEST IF DONE
2758 017124 012737 027530 001124      MOV    #BUFFER+2,$GDDAT ;SET UP CURRENT ADRS
2759 017132 010001          MOV    RO,R1        ;GET RELATIVE ADRS VALUE
2760 017134 005401          NEG    R1           ;MAKE NEG
2761 017136 006301          ASL    R1           ;CORRECT FOR WORD OFFSET
2762 017140 042701 177000      BIC    #177000,R1    ;GET RID OF MSBS
2763 017144 160137 001124      SUB    R1,$GDDAT    ;SUB OFFSET FROM DPC VALUE
2764 017150 000735          BR     15          ;TRY NEXT RELATIVE ADRS
2765
2766          ;*****
2767          ;*TEST 143 TEST THAT 'JSR ABS' INSTR WILL SET UP DPC WITH A FLOATING ONE
2768          ;*****
2769 017152 000004          TST143: SCOPE
2770 017154 012777 020000 162734      MOV    #BIT13,STKPT ;SET MAINT SWITCH #2
2771 017162 012737 162000 027526      MOV    #162000,BUFFER ;SET UP JSR ABSOLUTE INSTR AT 'BUFFER'
2772 017170 005037 001124          CLR    $GDDAT       ;EXPECT ZEROS FIRST TIME
2773 017174 012737 000001 027530      MOV    #1,BUFFER+2  ;SET UP ABS ADRS TO 1
2774 017202 052777 000040 162706 15:  BIS    #40,STKPT    ;RESET STK PTR
2775 017210 012777 027526 162646      MOV    #BUFFER,DPC   ;START
2776 017216 012737 017202 001110      MOV    #1,$SLPERR   ;SET UP SCOPE LOOP ADRS
2777 017224 005277 162634          INC    DPC          ;ADVANCE TO ABS ADRS
    
```

```

2778 017230 013737 002064 001122      MOV      DPC,$BDADR      ;SET UP REG ADRS 00
2779 017236 017737 162622 001126      MOV      JDPC,$BDDAT    ;GET ADRS JUMPED TO
2780 017244 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2781 017252 001401          BEQ      25              ;BR IF OK
2782 017254 104012          ERROR    12              ;JSR ABSOLUTE FAILED (TO A FLOATING ONE)
2783 017256 006337 027530          25:     ASL      BUFFER+2    ;ADVANCE ONE BIT LEFT
2784 017262 103404          BCS      TST144         ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
2785 017264 013737 027530 001124      MOV      BUFFER+2,$GDDAT ;SET UP EXPECTED ADRS
2786 017272 000743          BR       15              ;TRY NEXT BIT

```

```

*****
;*TEST 144 TEST THAT JSR ABS INSTR WILL SET UP DPC WITH A FLOATING ZERO
*****

```

```

2791 017274 000004          TST144: SCOPE
2792 017276 012777 020000 162612      MOV      #BIT13,$STKPT   ;SET MAINT SWITCH #2
2793 017304 012737 017326 001110      MOV      #15,$LPERA     ;SET UP SCOPE LOOP ADRS
2794 017812 012737 162000 027526      MOV      #162000,BUFFER  ;SET UP JSR ABSOLUTE AT 'BUFFER'
2795 017320 012737 177774 001124      MOV      #177774,$GDDAT ;SET LSB TO ZERO
2796 017326 052777 000040 162562          15:     BIS      #40,$STKPT    ;RESET STK PTR
2797 017334 012777 027526 162522      MOV      #BUFFER,$DPC   ;START
2798 017342 013737 001124 027530      MOV      $GDDAT,BUFFER+2 ;SET UP ADRS
2799 017350 005277 162510          INC      JDPC           ;ADVANCE TO ABS ADRS
2800 017354 013737 002064 001122      MOV      DPC,$BDADR     ;SET UP REG ADRS 03
2801 017362 017737 162476 001126      MOV      JDPC,$BDDAT    ;GET ADRS JUMPED TO
2802 017370 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2803 017376 001401          BEQ      25              ;BR IF OK
2804 017400 104012          ERROR    12              ;JSR ABSOLUTE FAILED (TO A FLOATING ZERO)
2805 017402 006337 001124          25:     ASL      $GDDAT     ;SHIFT ZERO BIT LEFT
2806 017406 103004          BCC      TST145         ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
2807 017410 062737 000002 001124      ADD      #2,$GDDAT      ;REPLACE LSB
2808 017416 000743          BR       15              ;TRY NEXT BIT

```

```

*****
;*TEST 145 TEST THAT GRAPHPLT X CAN LOAD X POSITION REGISTER WITH #1252
*****

```

```

2813 017420 000004          TST145: SCOPE
2814 017422 013737 002070 001122      MOV      XPOS,$BDADR    ;SET UP REG ADRS 04
2815 017430 012777 020000 162460      MOV      #BIT13,$STKPT  ;SET MAINT SWITCH #2
2816 017436 012700 027526          MOV      #BUFFER,R0     ;LOAD BUFFER POINTER
2817 017442 012720 122000          MOV      #122000,(R0)+  ;LOW INTERSITY - SET GRAPH PLOT X MODE
2818 017446 012720 001252          MOV      #1252,(R0)+   ;SET X POSITION
2819 017452 013777 002034 162404      MOV      DBUF,$DPC      ;LOAD DISPLAY P.C.
2820 017460 012737 001252 001124      MOV      #1252,$GDDAT   ;LOAD EXPECTED
2821 017466 005277 162372          INC      JDPC           ;SINGLE STEP THE DISPLAY
2822 017472 004737 023170          JSR      PC,DLAY        ;STALL
2823 017476 017737 162366 001126      MOV      #XPOS,$BDDAT   ;READ X POSITION
2824 017504 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPCT TO RCVD
2825 017512 001401          BEQ      TST146         ;BR IF EQUAL
2826 017514 104014          ERROR    14              ;X POSITION REGISTER FAILED TO LOAD
2827                                     ;PROPERLY USING GRAPH PLOT X MODE

```

```

*****
;*TEST 146 TEST THAT GRAPHPLT X CAN LOAD X POSITION REGISTER WITH #525
*****

```

```

2832 017516 000004          TST146: SCOPE
2833 017520 013737 002070 001122      MOV      XPOS,$BDADR    ;SET UP REG ADRS 04

```



```

2834 017526 012777 020000 162362      MOV      #BIT13,PT          ;SET MAINT SWITCH #2
2835 017534 012700 027526          MOV      #BUFFER,RO        ;LOAD BUFFER POINTER
2836 017540 012720 122000          MOV      #122000,(RO)+     ;LOW INTENSITY - SET GRAPH PLOT X MODE
2837 017544 012720 000525          MOV      #525,(RO)+       ;SET X POSITION
2838 017550 012777 027526 162306      MOV      #BUFFER,JDPC      ;STAR DISPLAY
2839 017556 012737 000525 001124      MOV      #525,$GDDAT       ;LOAD EXPECTED
2840 017564 005277 162274          INC      JDPC              ;SINGLE STEP THE DISPLAY
2841 017570 004737 023170          JSR      PC,DLAY           ;STALL
2842 017574 017737 162270 001126      MOV      JYPOS,$BDDAT      ;READ X POSITION
2843 017602 023737 001124 001126      CMP      $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
2844 017610 001401                    BEQ      TST147            ;:BR IF EQUAL
2845 017612 104014                    ERROR    14                ;X POSITION REGISTER FAILED TO LOAD
                                ;PROPERLY USING GRAPH PLOT X MODE

```

```

;*****
;*TEST 147      TEST THAT GRAPH PLOT Y CAN LOAD Y POSITION REGISTER WITH #1252
;*****
TST147: SCOPE

```

```

2851 017614 000004                    MOV      YPOS,$BDDADR      ;SET UP REG ADRS 06
2852 017616 013737 002072 001122      MOV      #BIT13,STKPT      ;SET MAINT SWITCH #2
2853 017624 012777 020000 162264      MOV      #BUFFER,RO        ;LOAD BUFFER POINTER
2854 017632 012700 027526          MOV      #126000,(RO)+     ;LOW INTENSITY - SET GRAPH PLOT Y
2855 017636 012720 126000          MOV      #1252,(RO)+       ;SET Y POSITION
2856 017642 012720 001252          MOV      DBUF,JDPC         ;LOAD THE DISPLAY P.C.
2857 017646 013777 002034 162210      MOV      #1252,$GDDAT      ;LOAD EXPECTED
2858 017654 012737 001252 001124      INC      JDPC              ;SINGLE STEP THE DISPLAY
2859 017662 005277 162176          JSR      PC,DLAY           ;STALL
2860 017666 004737 023170          MOV      JYPOS,$BDDAT      ;READ Y POSITION
2861 017672 017737 162174 001126      CMP      $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
2862 017700 023737 001124 001126      BEQ      TST150            ;:BR IF EQUAL
2863 017706 001401                    ERROR    15                ;Y POSITION REGISTER FAILED TO LOAD
2864 017710 104015                    ;PROPERLY USING GRAPH PLOT Y MODE

```

```

;*****
;*TEST 150      TEST THAT GRAPH PLOT Y CAN LOAD Y POSITION REGISTER WITH #525
;*****
TST150: SCOPE

```

```

2870 017712 000004                    MOV      YPOS,$BDDADR      ;SET UP REG ADRS 06
2871 017714 013737 002072 001122      MOV      #BIT13,STKPT      ;SET MAINT SWITCH #2
2872 017722 012777 020000 162166      MOV      #BUFFER,RO        ;LOAD BUFFER POINTER
2873 017730 012700 027526          MOV      #126000,(RO)+     ;LOW INTENSITY - SET GRAPH PLOT Y MODE
2874 017734 012720 126000          MOV      #525,(RO)+       ;SET Y POSITION
2875 017740 012720 000525          MOV      DBUF,JDPC         ;LOAD THE DISPLAY P.C.
2876 017744 013777 002034 162112      MOV      #525,$GDDAT      ;LD EXPECTED
2877 017752 012737 000525 001124      INC      JDPC              ;SINGLE STEP THE DISPLAY
2878 017760 005277 162100          JSR      PC,DLAY           ;STALL
2879 017764 004737 023170          MOV      JYPOS,$BDDAT      ;READ Y POSITION
2880 017770 017737 162076 001126      CMP      $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
2881 017776 023737 001124 001126      BEQ      TST151            ;:BR IF EQUAL
2882 020004 001401                    ERROR    15                ;Y POSITION REGISTER FAILED TO LOAD
2883 020006 104015                    ;PROPERLY USING GRAPH PLOT Y MODE

```

```

;*****
;*TEST 151      TEST FOR PROPER SELECTION OF X AND Y REGISTERS
;*****

```

```

2884
2885
2886
2887
2888
2889

```

F05

```

2890 020010 000004          TST151: SCOPE
2891 020012 013737 002070 001122  MOV      XPOS, $BDADR      ;SET UP REG ADRS 04
2892 020020 012777 020000 162070  MOV      #BIT13, $STKPT    ;SET MAINT SWITCH #2
2893 020026 012700 027526          MOV      #BUFFER, R0      ;LOAD BUFFER POINTER
2894 020032 012720 122000          MOV      #122000, (R0)+    ;LOW INTENSITY - SET GRAPHPLOT X MODE
2895 020036 012720 001234          MOV      #1234, (R0)+     ;SET X POSITION
2896 020042 012720 126000          MOV      #126000, (R0)+   ;SET GRAPHPLOT Y MODE
2897 020046 012720 001432          MOV      #1432, (R0)+     ;SET Y POSITION
2898 020052 013777 002034 162004  MOV      DBUF, $DPC       ;LOAD THE DISPLAY P.C.
2899 020060 012737 001234 001124  MOV      #1234, $GDDAT    ;LD EXPECTED
2900 020066 005277 161772          INC      $DPC             ;SINGLE STEP THE DISPLAY
2901 020072 004737 023170          JSR      PC, DLAY        ;STALL
2902 020076 017737 161766 001126  MOV      $XPOS, $BDDAT    ;READ X POSITION
2903 020104 023737 001124 001126  CMP      $GDDAT, $BDDAT   ;COMPARE EXPCT TO RCVD
2904 020112 001402          BEQ      15
2905 020114 104014          ERROR   14              ;GRAPHPLOT X MODE FAILED TO SELECT
2906 020116 000424          BR       TST152
2907
2908 020120          IS:
2909 020120 005277 161740          INC      $DPC             ;SINGLE STEP THE DISPLAY
2910 020124 013737 002072 001122  MOV      YPOS, $BDADR     ;SET UP REG ADRS 06
2911 020132 005277 161726          INC      $DPC             ;SINGLE STEP THE DISPLAY
2912 020136 012737 001432 001124  MOV      #1432, $GDDAT    ;LOAD EXPECTED
2913 020144 004737 023170          JSR      PC, DLAY        ;STALL
2914 020150 017737 161716 001126  MOV      $YPOS, $BDDAT    ;READ Y POSITION
2915 020156 023737 001124 001126  CMP      $GDDAT, $BDDAT   ;COMPARE EXPCT TO RCVD
2916 020164 001401          BEQ      TST152         ;;BR IF EQUAL
2917 020166 104015          ERROR   15              ;Y POSITION REGISTER FAILED TO LOAD
2918                                     ;PROPERLY USING GRAPHPLOT Y MODE
2919
2920
2921 *****
2922 *TEST 152 TEST THAT RESET CLEARS X AND Y POSITION REG.
2923 *****
2924 020170 000004          TST152: SCOPE
2925 020172 012737 000040 001164  MOV      #40, $TIMES      ;;DO 40 ITERATIONS
2926 020200 013737 002070 001122  MOV      XPOS, $BDADR     ;SET UP REG ADRS 04
2927 020206 012737 114000 027526  MOV      #114000, BUFFER  ;LOAD "POINT" INST.
2928 020214 012777 020000 161674  MOV      #BIT13, $STKPT    ;SET MAINT SWITCH #2
2929 020222 012777 027526 161634  MOV      #BUFFER, $DPC    ;START DISPLAY
2930 020230 012737 001777 027530  MOV      #1777, BUFFER+2  ;X=1777
2931 020236 005277 161622          INC      $DPC             ;SINGLE STEP THE DISPLAY
2932 020242 012737 001777 027532  MOV      #1777, BUFFER+4  ;Y=1777
2933 020250 005277 161610          INC      $DPC             ;SINGLE STEP THE DISPLAY
2934 020254 004737 023170          JSR      PC, DLAY        ;STALL
2935 020260 000005          RESET
2936 020262 005037 001124          CLR      $GDDAT          ;INITILIZE
2937 020266 017737 161576 001126  MOV      $XPOS, $BDDAT    ;CLEAR EXPECTED
2938 020274 001401          BEQ      15              ;READ X POSITION
2939 020276 104032          ERROR   32              ;;BR IF CLEARED
2940                                     ;RESET FAILED TO CLEAR X POS. REGISTER
2941 020300 013737 002072 001122  IS:  MOV      YPOS, $BDADR     ;SET UP REG ADRS 06
2942 020306 017737 161560 001126  MOV      $YPOS, $BDDAT    ;READ Y POSITION
2943 020314 001401          BEQ      TST153         ;;BR IF CLARED
2944 020316 104032          ERROR   32              ;RESET FAILED TO CLEAR Y POS. REGISTER
2945
*****

```

G05

```
2946 ;*TEST 153 TEST THAT DISPLAY BUSY WILL SET & THEN CLR ON INTERNAL STOP
2947 ;*****
2948 TST153: SCOPE
2949 MOV #BIT13, JSTKPT ;SET MAINT SW 2
2950 MOV #100000, $GDDAT ;EXPECT DISPLAY BUSY INITIALLY
2951 MOV #164000, BUFFER ;SET UP A NOP INSTR
2952 MOV #172000, BUFFER+2 ;SET UP LD STATUS A INSTR WITH STOP
2953 MOV #BUFFER, JOPC ;START
2954 MOV SREG1, $BDADR ;SET UP REG ADRS 12
2955 TST JSREG1 ;SEE THAT DISPLAY BUSY IS SET
2956 BMI IS ;BR IF 50
2957 CLR $BDDAT ;INDICATE DISPLAY BUSY NOT SET
2958 ERROR 44 ;DISPLAY BUSY NOT SET
2959 IS: INC JOPC ;ADVANCE TO STOP INSTR
2960 CLR $GDDAT ;EXPECTE ZERO
2961 TST JSREG1 ;SEE THAT DISPLAY BUSY HAS CLEARED
2962 BPL TST154 ;GO TO NEXT TEST IF OK
2963 MOV #100000, $BDDAT ;INDICATE DISPLAY BUSY WAS STILL SET
2964 ERROR 44 ;DISPLAY BUSY FAILED TO CLR ON STOP INSTR
2965
2966 ;*****
2967 ;*TEST 154 TEST THAT RESET WILL CLEAR DISPLAY BUSY
2968 ;*****
2969 TST154: SCOPE
2970 MOV #40, $TIMES ;DO 40 ITERATIONS
2971 MOV SREG1, $BDADR ;SET UP REG ADRS 12
2972 MOV #BIT13, JSTKPT ;SET MAINT SW 2
2973 MOV #164000, BUFFER ;SET UP NOP INSTR
2974 MOV #BUFFER, JOPC ;START
2975 MOV #404, $GDDAT ;EXPECT ONLY CHAR & VECTOR SCALE BITS
2976 RESET ;CLR BUSY
2977 MOV JSREG1, $BDDAT ;READ REG 12
2978 CMP $GDDAT, $BDDAT ;IS IT CORRECT?
2979 BEQ TST155 ;NEXT TEST IF OK
2980 ERROR 44 ;RESET FAILED TO CLR DISPLAY BUSY
2981
2982 ;*****
2983 ;*TEST 155 TEST THAT EXTERNAL STOP WILL SET & RESET WITH DISPLAY OFF
2984 ;*****
2985 TST155: SCOPE
2986 MOV #40, $TIMES ;DO 40 ITERATIONS
2987 MOV #BIT13, JSTKPT ;SET MAINT SWITCH #2
2988 MOV #340, PSW ;SET PRIORITY TO HIGEST LEVEL
2989 RESET ;MAKE SURE DISPLAY IS OFF
2990 MOV SREG1, $BDADR ;SET UP REG 12 ADRS
2991 MOV #200, $GDDAT ;EXPECT EXTERNAL STOP TO BE SET
2992 MOV $GDDAT, JSREG1 ;SET EXTERNAL STOP
2993 TSTB JSREG1 ;SEE IF BIT SET
2994 BMI IS ;BR IF IT DID SET
2995 CLR $BDDAT ;INDICATE IT WAS NOT SET
2996 ERROR 45 ;EXTERNAL STOP FAILED TO SET
2997 IS: MOV #BUFFER, JOPC ;CLR EXTERNAL STOP BIT
2998 TSTB JSREG1 ;CHECK THAT IT WAS CLEARED
2999 BPL TST156 ;BR TO NEXT TEST IF OK
3000 CLR $GDDAT ;EXPECTED ZERO
3001
```

H05

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 T155

MACY11 27(732) 20-SEP-76 09:57 PAGE 59
TEST THAT EXTERNAL STOP WILL SET & RESET WITH DISPLAY OFF

```
3002 020624 012737 000200 001126      MOV      #200,$BDDAT      ;INDICATE THAT IT DID NOT CLR
3003 020632 104045                      ERROR     45              ;EXTERNAL STOP FAILED TO CLR ON START
3004
3005
3006      ;*****
3007      ;*TEST 156      TEST THAT EXTERNAL STOP WILL SET & RESET WITH DISPLAY BUSY
3008      ;*****
3008 020634 000004      TST156: SCOPE
3009 020636 012777 020000 161252      MOV      #BIT13,2STKPT      ;SET MAINT SWITCH #2
3010 020644 012737 000340 177776      MOV      #340,PSW          ;SET PSW TO 7
3011 020652 012737 000200 001124      MOV      #200,$GDDAT      ;EXPECT EXTERNAL STOP TO BE SET INITIALLY
3012 020660 012737 164000 027526      MOV      #164000,BUFFER    ;SET UP A NOP INSTR
3013 020666 012777 027526 161170      MOV      #BUFFER,2DPC      ;START
3014 020674 012737 164000 027530      MOV      #164000,BUFFER+2  ;SET UP ANOTHER NOP
3015 020702 013777 001124 161166      MOV      $GDDAT,2SREG1     ;ISSUE A SOFTWARE STOP
3016 020710 005277 161150                      INC      2DPC              ;THIS RESUME SHOULD ALLOW EXTERNAL STOP TO SET
3017 020714 013737 002076 001122      MOV      SREG1,$BDADR      ;SET UP REG ADRS 12
3018 020722 032777 000200 161146      BIT      #200,2SREG1       ;SEE IF IT SET
3019 020730 001004                      BNE     1$                ;BR IF 0
3020 020732 005037 001126      CLR      $BDDAT           ;INDICATE IT DID NOT SET
3021 020736 104045                      ERROR     45              ;EXTERNAL STOP FAILED TO SET WHEN DISPLAY BUSY
3022 020740 000415                      BR      TST157            ;GO LOOK FOR LOOP ON TEST SW
3023 020742 012777 027526 161114      1$: MOV      #BUFFER,2DPC    ;START - THIS SHOULD CLR EXT STOP
3024 020750 032777 000200 161120      BIT      #200,2SREG1       ;CHECK THAT IT DID CLEAR
3025 020756 001406                      BEQ     TST157            ;GO TO NEXT TEST IF EXTERNAL STOP CLEARED
3026 020760 005037 001124                      CLR      $GDDAT           ;EXPECTED ZERO
3027 020764 012737 000200 001126      MOV      #200,$BDDAT      ;INDICATE IT DID NOT CLR
3028 020772 104045                      ERROR     45              ;EXTERNAL STOP FAILED TO CLR ON START
3029
3030      ;*****
3031      ;*TEST 157      TEST THAT RESET WILL CLEAR EXTERNAL STOP
3032      ;*****
3033 020774 000004      TST157: SCOPE
3034 020776 012737 000010 001164      MOV      #10,$TIMES        ;DO 10 ITERATIONS
3035 021004 000005                      RESET                    ;MAKE SURE DISPLAY IS OFF
3036 021006 013737 002076 001122      MOV      SREG1,$BDADR      ;SET UP REG 12 ADRS
3037 021014 005037 001124                      CLR      $GDDAT           ;EXPECT RESET STATE
3038 021020 012777 000200 161050      MOV      #200,2SREG1       ;SET EXTERNAL STOP
3039 021026 000005                      RESET                    ;CLR IT
3040 021030 032777 000200 161040      BIT      #200,2SREG1       ;DID IT CLR?
3041 021036 001404                      BEQ     TST160            ;NEXT TEST IF 0
3042 021040 012737 000200 001126      MOV      #200,$BDDAT      ;SHOW THAT IT WAS STILL SET
3043 021046 104045                      ERROR     45              ;RESET FAILED TO CLR EXTERNAL STOP
3044
3045      ;*****
3046      ;*TEST 160      TEST THAT EXTERNAL STOP WILL CAUSE AN INTERRUPT
3047      ;*****
3048 021050 000004      TST160: SCOPE
3049 021052 012737 000040 001164      MOV      #40,$TIMES        ;DO 40 ITERATIONS
3050 021060 012777 020000 161030      MOV      #BIT13,2STKPT     ;SET MAINT SWITCH #2
3051 021066 000005                      RESET                    ;MAKE SURE DISPLAY IS OFF
3052 021070 012737 000340 177776      MOV      #340,PSW          ;SET PSW TO 7
3053 021076 013737 002076 001122      MOV      SREG1,$BDADR      ;SET UP REG 12 ADRS
3054 021104 012737 000200 001124      MOV      #200,$GDDAT      ;EXPECT EXTERNAL STOP FLAG
3055 021112 013701 001250                      MOV      $VECT1,R1        ;GET BASIC VECTOR ADRS
3056 021116 012711 021162                      MOV      #1$(R1)          ;SET UP STOP INT ADRS
3057 021122 012761 000340 000002      MOV      #340,2(R1)        ;SET UP PSW TO 7 ON INT
```

3058	021130	012777	000200	160740	MOV	#200,JSREG1	;;CAUSE AN EXTERNAL STOP INT
3059	021136	005037	177776		CLR	PSW	;;ALLOW INT TO OCCUR
3060	021142	017737	160730	001126	MOV	JSREG1,\$BDDAT	;;READ REG 12
3061	021150	042737	177577	001126	BIC	#177577,\$BDDAT	;;SAVE ONLY STOP FLAG
3062	021156	104033			ERROR	33	;;EXTERNAL STOP FAILED TO INTERRUPT
3063	021160	000410			BR	2\$;;GO LOOK FOR LOOP ON TEST SWITCH
3064	021162	022626			1\$: CMP	(R6)+,(R6)+	;;FIX STK SINCE NO RETURN
3065	021164	032777	000200	160704	BIT	#200,JSREG1	;;CK THAT STOP FLAG IS SET
3066	021172	001003			BNE	2\$;;BR IF SO
3067	021174	005037	001126		CLR	\$BDDAT	;;INDICATE FLAG WAS NOT SET
3068	021200	104033			ERROR	33	;;EXTERNAL STOP INTERRUPTED BUT FLAG NOT SET
3069	021202	004737	023134		2\$: JSR	PC,RSTVEC	;;RESTORE EXTERNAL STOP VECTOR WITH HALT

3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102

```

*****
; *TEST 161 TEST THAT TIME OUT WILL CAUSE AN INTERRUPT
*****
TST161: SCOPE

```

3074	021206	000004			MOV	#40,\$TIMES	;;DO 40 ITERATIONS
3075	021210	012737	000040	001164	MOV	#BIT13,\$STKPT	;;SET MAINT SWITCH #2
3076	021216	012777	020000	160672	MOV	#340,PSW	;;SET PRIORITY TO HIGHEST LEVEL
3077	021224	012737	000340	177776	MOV	\$VECT1,R1	;;GET BASIC VECTOR ADRS
3078	021232	013701	001250		MOV	#1\$,10(R1)	;;SET UP TIMEOUT INT SERVICE ADRS
3079	021236	012761	021330	000010	MOV	#340,12(R1)	;;SET UP PRIORITY TO 7 ON INT
3080	021244	012761	000340	000012	MOV	SREG1,\$BDADR	;;SET UP REG 12 ADRS
3081	021252	013737	002076	001122	MOV	#4060,\$GDDAT	;;EXPECT TIMEOUT & DPC 16,17
3082	021260	012737	004060	001124	MOV	#7600,\$RLO	;;RELOCATE TO ADRS 76000C
3083	021266	012777	007600	160600	CLR	PSW	;;ALLOW INTERRUPT TO OCCUR
3084	021274	005037	177776		MOV	#0,\$DPC	;;DISPLAY WILL TIMEOUT AT ADRS 760000
3085	021300	012777	000000	160556	CMP	(SP),(SP)	;;ALLOW TIME
3086	021306	021616			MOV	JSREG1,\$BDDAT	;;READ REG 12
3087	021310	017737	160562	001126	BIC	#173717,\$BDDAT	;;SAVE ONLY TIMEOUT BIT & DPC 16,17
3088	021316	042737	173717	001126	ERROR	46	;;VS60 FAILED TO INTERRUPT ON TIME OUT
3089	021324	104046			BR	2\$;;GO LOOK FOR LOOP ON TEST SWITCH
3090	021326	000414			1\$: CMP	(R6)+,(R6)+	;;FIX STACK SINCE NO RETURN
3091	021330	022626			MOV	JSREG1,\$BDDAT	;;READ STATUS
3092	021332	017737	160540	001126	BIC	#173717,\$BDDAT	;;SAVE TIMEOUT & DPC 16,17
3093	021340	042737	173717	001126	CMP	\$GDDAT,\$BDDAT	;;CORRECT?
3094	021346	023737	001124	001126	BEQ	2\$;;BR IF SO
3095	021354	001401			ERROR	46	;;TIMEOUT BIT FAILED TO SET ON INTERRUPT
3096	021356	104046			2\$: JSR	PC,RSTVEC	;;RESET TIMEOUT VECTOR WITH HALT
3097	021360	004737	023134		RESET		;;CLR RELOCATE REG BEFORE ADVANCING
3098	021364	000005					

```

*****
; *TEST 162 TEST THAT RESET WILL CLEAR TIMEOUT FLAG
*****
TST162: SCOPE

```

3103	021366	000004			MOV	#40,\$TIMES	;;DO 40 ITERATIONS
3104	021370	012737	000040	001164	MOV	SREG1,\$BDADR	;;SET L.P. REG ADRS 12
3105	021376	013737	002076	001122	MOV	#BIT13,\$STKPT	;;SET MAINT SW 2
3106	021404	012777	020000	160504	MOV	#340,PSW	;;DON'T LET INTR OCCUR.
3107	021412	012737	000340	177776	MOV	#404,\$GDDAT	;;EXPECT ONLY CHAR & VECTOR SCALE BITS
3108	021420	012737	000404	001124	MOV	#7600,\$RLO	;;RELOCATE TO ADRS 760000
3109	021426	012777	007600	160440	MOV	#0,\$DPC	;;DISPLAY SHOULD TIMEOUT AT ADRS 760000
3110	021434	012777	000000	160422	CMP	(SP),(SP)	;;ALLOW TIME
3111	021442	021616			RESET		;;CLR REG REG 12
3112	021444	000005			MOV	JSREG1,\$BDDAT	;;READ REG 12
3113	021446	017737	160424	001126			

```

3114 021454 023737 001124 001126
3115 021462 001401
3116 021464 104032
3117
3118
3119
3120
3121 021466 000004
3122 021470 012737 000040 001164
3123 021476 012737 000340 177776
3124 021504 012777 020000 160404
3125 021512 012737 144000 001124
3126 021520 012737 164360 027526
3127 021526 012777 027526 160330
3128 021534 052777 040000 160354
3129 021542 012737 122000 027530
3130 021550 005277 160310
3131 021554 012737 001777 027532
3132 021562 005277 160276
3133 021566 013737 002106 001122
3134 021574 052777 040000 160304
3135 021602 017737 160300 001126
3136 021610 023737 001124 001126
3137 021616 001401
3138 021620 104034
3139 021622 000005
3140 021624 012737 100000 001124
3141 021632 017737 160250 001126
3142 021640 023737 001124 001126
3143 021646 001401
3144 021650 104032
3145
3146
3147
3148
3149 021652 000004
3150 021654 012737 000040 001164
3151 021662 012737 000340 177776
3152 021670 012777 020000 160220
3153 021676 012737 101440 001124
3154 021704 012737 164760 027526
3155 021712 012777 027526 160144
3156 021720 052777 040000 160170
3157 021726 012737 122000 027530
3158 021734 005277 160124
3159 021740 012737 001777 027532
3160 021746 005277 160112
3161 021752 013737 002106 001122
3162 021760 052777 000400 160120
3163 021766 017737 160114 001126
3164 021774 023737 001124 001126
3165 022002 001401
3166 022004 104034
3167 022006 000005
3168 022010 012737 100000 001124
3169 022016 017737 160064 001126

```

```

CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ TST163 ;:NEXT TEST IF OK
ERROR 32 ;:RESET FAILED TO CLR TIMEOUT

;*****
;:TEST 163 TEST THAT LIGHT PEN FLAG WILL SET THEN CLEAR ON RESET - CONS 0
;*****
TST163: SCOPE
MOV #40,$TIMES ;:DO 40 ITERATIONS
MOV #340,$PSW ;:HIGHEST PRIORITY
MOV #20000,$STKPT ;:SET MAINT SW 2
MOV #144000,$GDDAT ;:EXPECT L.P. FLAG INITIALLY
MOV #164360,$BUFFER ;:ENABLE L.P. FLG INTR
MOV $BUFFER,$DPC ;:START
BIS #40000,$STKPT ;:SET MAINT SW 3
MOV #122000,$BUFFER+2 ;:GRAPH X MODE
INC $DPC ;:RESUME
MOV #1777,$BUFFER+4 ;:X=1777
INC $DPC ;:RESUME
MOV $CONS,$BDADR ;:SET UP REG ADRS 22
BIS #40000,$CONS ;:SET L.P. FLAG
MOV $CONS,$BDDAT ;:READ FLAGS REG 22
CMP $GDDAT,$BDDAT ;:IS IT CORRECT?
BEQ 1$ ;:BR IF SO
ERROR 34 ;:LIGHT PEN FLAG FAILED TO SET - CONSOLE 0
RESET ;:CLR FLAGS
MOV #100000,$GDDAT ;:EXPECT ONLY INT ENA CONSOLE 0
MOV $CONS,$BDDAT ;:READ REG 22
CMP $GDDAT,$BDDAT ;:DID REG 22 RESET?
BEQ TST164 ;:NEXT TEST IF SO
ERROR 32 ;:RESET FAILED TO CLEAR FLAGS - CONSOLE 0

;*****
;:TEST 164 TEST THAT LIGHT PEN FLAG WILL SET THEN CLEAR ON RESET - CONS 1
;*****
TST164: SCOPE
MOV #40,$TIMES ;:DO 40 ITERATIONS
MOV #340,$PSW ;:HIGHEST PRIORITY
MOV #20000,$STKPT ;:SET MAINT SW 2
MOV #101440,$GDDAT ;:EXPECT L.P. FLAG INITIALLY
MOV #164760,$BUFFER ;:ENABLE L.P. FLG
MOV $BUFFER,$DPC ;:START
BIS #40000,$STKPT ;:SET MAINT SW 3
MOV #122000,$BUFFER+2 ;:GRAPH X MODE
INC $DPC ;:RESUME
MOV #1777,$BUFFER+4 ;:X=1777
INC $DPC ;:RESUME
MOV $CONS,$BDADR ;:SET UP REG ADRS 22
BIS #400,$CONS ;:SET L.P. FLAG
MOV $CONS,$BDDAT ;:READ IT BACK
CMP $GDDAT,$BDDAT ;:CORRECT?
BEQ 1$ ;:BR IF SO
ERROR 34 ;:LIGHT PEN FLAG FAILED TO SET - CONSOLE 1
RESET ;:CLR FLAGS
MOV #100000,$GDDAT ;:EXPECT ONLY INT EN FOR CONSOLE 0
MOV $CONS,$BDDAT ;:READ REG 22

```

K05

```

3170 022024 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;CORRECT?
3171 022032 001401                      BEQ      TST165             ;:NEXT TEST IF 50
3172 022034 104032                      ERROR    32                 ;:RESET FAILED TO CLR FLAGS - CONSOLE 1
3173
3174
3175
3176
3177 022036 000004                      TST165: SCOPE
3178 022040 012737 000100 001164      MOV      #100,$TIMES        ;:DO 100 ITERATIONS
3179 022046 012737 000340 177776      MOV      #340,PSW          ;:SET PRIORITY TO 7
3180 022054 013701 001250                      MOV      $VECT1,R1         ;:GET BASIC VS60 VECTOR ADRS
3181 022060 012761 022230 000004      MOV      #3$ 4(R1)         ;:SET UP INTR SERVICE ADRS
3182 022066 012761 000340 000006      MOV      #340,6(R1)        ;:SET UP PRIORITY TO 7 ON INT
3183 022074 012737 144000 001124      MOV      #144000,$GDDAT    ;:EXPECT INT EN, L.P. FLG, L.P. INT EN
3184 022102 012700 000001                      MOV      #1,R0             ;:RO WHEN NON-ZERO MEANS INT EXPECTED
3185 022106 012777 020040 160002      MOV      #20040,$STKPT     ;:SET MAINT SW 2
3186 022114 012737 164360 027526      MOV      #164360,BUFFER    ;:SET UP TO ENAB L.P. INT & INTENSITY EN
3187 022122 012777 027526 157734      MOV      #BUFFER,$DPC      ;:START
3188 022130 052777 040000 157760      BIS      #40000,$STKPT     ;:SET MAINT SW 3
3189 022136 012737 122000 027530      MOV      #122000,BUFFER+2  ;:GRAPH X MODE
3190 022144 005277 157714                      INC      $DPC              ;:PICK UP GRAPH X INSTR - INT WILL OCCUR IN DATA MODE
3191 022150 012737 001777 027532      MOV      #1777,BUFFER+4   ;:X=1777
3192 022156 005277 157702                      INC      $DPC              ;:RESUME FOR X
3193 022162 013737 002106 001122      MOV      CONS,$BDADR       ;:SET UP REG ADRS 22
3194 022170 052777 040000 157710      BIS      #40000,$CONS     ;:SET L.P. FLAG
3195 022176 005037 177776                      CLR      PSW               ;:ALLOW INTR
3196 022202 012701 000040                      MOV      #40,R1           ;:SET UP DELAY
3197 022206 005301 25: DEC      R1                 ;:COUNT AWAY
3198 022210 001376                      BNE      25                ;:STAY UNTIL DONE
3199 022212 005700                      TST     RO                 ;:DO WE EXPECT INT THIS PASS - NO IF ZERO
3200 022214 001451                      BEQ      75                ;:BR IF NOT
3201 022216 017737 157664 001126      MOV      $CONS,$BDDAT     ;:READ L.P. STATUS
3202 022224 104050                      ERROR    50                ;:LIGHT PEN CONSOLE 0 INTERRUPT FAILURE
3203 022226 000454                      BR       85                ;:GO LOOK FOR LOOP ON TEST SW
3204 022230 022626                      CMP      (R6)+,(R6)+       ;:FIX STACK SINCE NO RTI
3205 022232 017737 157650 001126      MOV      $CONS,$BDDAT     ;:GET CONTENTS OF REG 22
3206 022240 005700                      TST     RO                 ;:DID WE EXPECT AN L.P. INTERRUPT THIS PASS?
3207 022242 001002                      BNE      45                ;:BR IF 50
3208 022244 104050                      ERROR    50                ;:L.P. FLAG INTERRUPTED WHEN NOT ENABLED
3209 022246 000444                      BR       85                ;:GO LOOK FOR LOOP ON TEST SW
3210 022250 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;:SEE IF STATUS IS CORRECT AFTER INT
3211 022256 001402                      BEQ      55                ;:BR IF 50
3212 022260 104050                      ERROR    50                ;:INCORRECT INTERRUPT STATUS
3213 022262 000436                      BR       85                ;:GO LOOK FOR LOOP ON TEST SW
3214 022264 032777 000200 157574      BIT      #200,$SREG0       ;:L.P. FLG SET IN REG 02 ALSO?
3215 022272 001012                      BNE      65                ;:BR IF 50
3216 022274 013737 002066 001122      MOV      SREG0,$BDADR     ;:SET UP REG ADRS 02
3217 022302 012737 000200 001124      MOV      #200,$GDDAT      ;:EXPECTED L.P. FLAG
3218 022310 005037 001126                      CLR      $BDDAT           ;:SHOW IT WAS NOT SET
3219 022314 104050                      ERROR    50                ;:L.P. FLAG FAILED TO SET AT REG 02
3220 022316 000420                      BR       85                ;:GO LOOK FOR LOOP ON TEST SW
3221 022320 012737 100000 001124      MOV      #100000,$GDDAT   ;:SET UP EXPECTED WITH L.P. FLAG INT DISABLED
3222 022326 012737 164340 027526      MOV      #164340,BUFFER   ;:SET UP INSTR WITH L.P. FLAG INT DISABLED
3223 022334 005000                      CLR      RO               ;:SET TEST SWITCH TO NO INTERRUPT EXPECTED
3224 022336 000671                      BR       15                ;:GO REPEAT TEST WITH INT EN DISABLED
3225 022340 017737 157542 001126      MOV      $CONS,$BDDAT     ;:READ REG 22

```

L05

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 T165 TEST THAT LIGHT PEN FLAG - CONSOLE 0 WILL CAUSE AN INTERRUPT

MACY11 27(732) 20-SEP-76 09:57 PAGE 63

```

3226 022346 023737 001126 001124      CMP      $BDDAT,$GDDAT      ;FLAG CLR ON START?
3227 022354 001401                    BEQ      8$                  ;BR IF 50
3228 022356 104034                    ERROR   34                  ;START FAILED TO CLR L.P. FLAG
3229 022360 004737 023134      8$:    JSR      PC,RSTVEC      ;RESET L.P. INTERRUPT VECTOR WITH HALT
3230
3231
3232 ;*****
3233 ;*TEST 166 TEST THAT LIGHT PEN FLAG - CONSOLE 1 WILL CAUSE AN INTERRUPT
3234 ;*****
3234 022364 000004      †TST166: SCOPE
3235 022366 012737 000100 001164      MOV      #100,$TIMES      ;DO 100 ITERATIONS
3236 022374 012737 000340 177776      MOV      #340,PSW        ;SET PRIORITY TO 7
3237 022402 013701 001250                    MOV      $VECT1,R1        ;GET BASIC VS60 VECTOR ADRS
3238 022406 012761 022556 000004      MOV      #3$,4(R1)        ;SET UP INTR SERVICE ADRS
3239 022414 012761 000340 000006      MOV      #340,6(R1)       ;SET UP PRIORITY TO 7 ON INT
3240 022422 012737 101440 001124      MOV      #101440,$GDDAT   ;EXPECT INT EN, L.P. FLG, L.P. INT EN
3241 022430 012700 000001                    MOV      #1,RO            ;RO WHEN NON-ZERO MEANS INT EXPECTED
3242 022434 012777 020040 157454      MOV      #20040,‡STKPT    ;SET MAINT SW 2
3243 022442 012737 164760 027526      MOV      #164760,BUFFER   ;SET UP TO ENAB L.P. INT & INTENSITY EN
3244 022450 012777 027526 157406      1$:    MOV      #BUFFER,‡DPC    ;START
3245 022456 052777 040000 157432      BIS      #40000,‡STKPT    ;SET MAINT SW 3
3246 022464 012737 122000 027530      MOV      #122000,BUFFER+2 ;GRAPH X MODE
3247 022472 005277 157366                    INC      ‡DPC              ;PICK UP GRAPH X INSTR - INT WILL OCCUR IN DATA MODE
3248 022476 012737 001777 027532      MOV      #1777,BUFFER+4   ;X=1777
3249 022504 005277 157354                    INC      ‡DPC              ;RESUME FOR X
3250 022510 013737 002106 001122      MOV      CONS,$BDADR      ;SET UP REG ADRS 22
3251 022516 052777 000400 157362      BIS      #400,‡CONS      ;SET L.P. FLAG
3252 022524 005037 177776                    CLR      PSW              ;ALLOW INTR
3253 022530 012701 000040                    MOV      #40,R1          ;SET UP DELAY
3254 022534 005301                    2$:    DEC      R1              ;COUNT AWAY
3255 022536 001376                    BNE     2$                ;STAY UNTIL DONE
3256 022540 005700                    TST     RO                ;DO WE EXPECT INT THIS PASS - NO IF ZERO
3257 022542 001433                    BEQ     6$                ;BR IF NOT
3258 022544 017737 157336 001126      MOV      ‡CONS,$BDDAT     ;READ L.P. STATUS
3259 022552 104050                    ERROR   50                ;LIGHT PEN CONSOLE 1 INTERRUPT FAILURE
3260 022554 000436                    BR      7$                ;GO LOOK FOR LOOP ON TEST SW
3261 022556 022626                    3$:    CMP      (R6)+,(R6)+    ;FIX STACK SINCE NO RTI
3262 022560 017737 157322 001126      MOV      ‡CONS,$BDDAT     ;GET L.P. STATUS
3263 022566 005700                    TST     RO                ;DID WE EXPECT AN L.P. INTERRUPT THIS PASS?
3264 022570 001002                    BNE     4$                ;BR IF 50
3265 022572 104050                    ERROR   50                ;L.P. FLAG INTERRUPTED WHEN NOT ENABLED
3266 022574 000426                    BR      7$                ;GO LOOK FOR LOOP ON TEST SW
3267 022576 023737 001124 001126      4$:    CMP      $GDDAT,$BDDAT  ;SEE IF STATUS IS CORRECT AFTER INT
3268 022604 001402                    BEQ     5$                ;BR IF 50
3269 022606 104050                    ERROR   50                ;INCORRECT L.P. STATUS
3270 022610 000420                    BR      7$                ;GO LOOK FOR LOOP ON TEST SW
3271 022612 012737 101000 001124      5$:    MOV      #101000,$GDDAT  ;SET UP TEST WITH L.P. FLAG INT DISABLED
3272 022620 012737 164740 027526      MOV      #164740,BUFFER   ;SET UP INSTR WITH L.P. FLAG DISABLED
3273 022626 005000                    CLR     RO                ;SET TEST SWITCH TO NO INTERRUPT EXPECTED
3274 022630 000707                    BR      1$                ;GO REPEAT TEST WITH INT EN DISABLED
3275 022632 017737 157250 001126      6$:    MOV      ‡CONS,$BDDAT    ;READ REG 22
3276 022640 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;L.P. FLG CLR ON START?
3277 022646 001401                    BEQ     7$                ;BR IF 50
3278 022650 104034                    ERROR   34                ;START FAILED TO CLR L.P. FLAG
3279 022652 004737 023134      7$:    JSR      PC,RSTVEC      ;RESET L.P. INTERRUPT VECTOR WITH HALT
3280
3281 ;*****

```


M05

```

3282 ;*TEST 167 TEST THAT WORD 0,1 & 2 INDICATORS WILL SET
3283 ;*****
3284 022656 000004 TST167: SCOPE
3285 022660 012777 020000 157230 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
3286 022666 012737 000100 001124 MOV #100,$GDDAT ;EXPECT WORD 0 INDICATION INITIALLY
3287 022674 012737 11C300 027526 MOV #110000,BUFFER ;SET UP LONG VECTOR INSTR
3288 022702 005037 027530 CLR BUFFER+2 ;CLR X,Y
3289 022706 005037 027532 CLR BUFFER+4
3290 022712 012777 027526 157144 MOV #BUFFER,2DPC ;START
3291 022720 000402 BR 25 ;GO ALLOW TIME FOR INSTR NPR
3292 022722 005277 157136 15: INC 2DPC ;PICK UP DATA WORD
3293 022726 013737 002116 001122 25: MOV STKPT,$BDADR ;SET UP REG ADRS 32
3294 022734 017737 157156 001126 MOV 2STKPT,$BDDAT ;READ REG 32
3295 022742 042737 177077 001126 BIC #177077,$BDDAT ;SAVE ONLY WORD INDICATORS
3296 022750 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
3297 022756 001401 BEQ 35 ;BR IF SO
3298 022760 104041 ERROR 41 ;WORD INDICATOR FAILURE
3299 022762 006337 001124 35: ASL $GDDAT ;ADVANCE TO NEXT WORD
3300 022766 022737 001000 001124 CMP #1000,$GDDAT ;HAVE ALL BEEN TESTED?
3301 022774 001352 BNE 15 ;BR IF NOT
3302
3303 ;*****
3304 ;*TEST 170 TEST FOR HALT AT END OF DIAGNOSTIC (SW12 SET)
3305 ;*****
3306 022776 000004 TST170: SCOPE
3307 023000 032777 010000 156130 BIT #BIT12,2SWR ;IS SW12 SET?
3308 023006 001401 BEQ SEOP ;BR IF NOT
3309 023010 000000 HALT ;COMPLETED PASS - SW12 SAYS HALT

```

```

3310 ;*****
3311
3312 .SBTTL END OF PASS ROUTINE
3313
3314 ;*INCREMENT THE PASS NUMBER ($PASS)
3315 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3316 ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
3317 ;*IF THERES A MONITOR GO TO IT
3318 ;*IF THERE ISN'T JUMP TO RESTRT
3319
3320 023012 $EOP:
3321 023012 000004 SCOPE
3322 023014 005037 001102 CLR $STNM ;; ZERO THE TEST NUMBER
3323 023020 005037 001164 CLR $TIMES ;; ZERO THE NUMBER OF ITERATIONS...
3324 023024 005237 001206 INC $PASS ;; INCREMENT THE PASS NUMBER
3325 023030 042737 100000 001206 BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
3326 023036 005327 DEC (PC)+ ;; LOOP?
3327 023040 000001 $EOPCT: .WORD 1
3328 023042 003022 BGT $DOAGN ;; YES
3329 023044 012737 MOV (PC)+,2(PC)+ ;; RESTORE COUNTER
3330 023046 000001 $ENDCT: .WORD 1
3331 023050 023040 $EOPCT
3332 023052 104400 023114 TYPE ,SENDMG ;; TYPE "END PASS #"
3333 023056 013746 001206 MOV $PASS,-(SP) ;; SAVE $PASS FOR TYPEOUT
3334 023062 104404 TYPDS ;; GO TYPE--DECIMAL ASCII WITH SIGN
3335 023064 104400 023131 TYPE ,SENULL ;; TYPE A NULL CHARACTER
3336 023070 $GET42:
3337
3338 023070 013700 000042 MOV 2#42,R0 ;; GET MONITOR ADDRESS
3339 023074 001405 BEQ $DOAGN ;; BRANCH IF NO MONITOR
3340 023076 000005 RESET ;; CLEAR THE WORLD
3341 023100 004710 $ENDAD: JSR PC,(R0) ;; GO TO MONITOR
3342 023102 000240 NOP ;; SAVE ROOM
3343 023104 000240 NOP ;; FOR
3344 023106 000240 NOP ;; ACT11
3345 023110 $DOAGN:
3346 023110 000137 002506 JMP 2#RESTRT ;; RETURN
3347 023114 005015 047105 020104 $ENDMG: .ASCIZ <15><12>/END PASS #/
3348 023122 040520 051523 021440
3349 023130 000
3350 023131 377 377 000 $ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
3351

```

```

023134 012737 000340 177776 RSTVEC: MOV #340,PSW ;RESET PRIORITY TO HIGHEST LEVEL
023142 012701 000010 MOV #10,R1 ;SET UP VECTOR LOCATION COUNT
023146 013700 001250 MOV SVECT1,R0 ;GET 1ST VS60 VECTOR ADRS
023152 010010 1S: MOV R0,(R0) ;SET UP ADRS TO HALT
023154 062720 000002 ADD #2,(R0)+ ;POINT IT TO HALT
023160 005020 CLR (R0)+ ;SET UP HALT
023162 005301 DEC R1 ;COUNT THIS VECTOR RESTORE
023164 001372 BNE 1S ;BR IF MORE TO RESTORE
023166 000207 RTS PC ;RETURN IF ALL DONE

;*****
;THIS IS A GENERAL PURPOSE DELAY ROUTINE (REG CNT OF 20)
;*****
023170 013700 002062 DLAY: MOV DFLAY,R0 ;GET COUNT
023174 005300 1S: DEC R0 ;COUNT AWAY
023176 001376 BNE 1S ;BR TILL DONE
023200 000207 RTS PC ;EXIT

```

```

3373 ;*****
3374
3375 .SBTTL SCOPE HANDLER ROUTINE
3376
3377 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3378 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3379 ;*AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
3380 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3381 ;*SW14=1 LOOP ON TEST
3382 ;*SW11=1 INHIBIT ITERATIONS
3383 ;*SW09=1 LOOP ON ERROR
3384 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
3385 ;*CALL
3386 ;*
3387 SCOPE ::SCOPE=IOT
3388
3389 $SCOPE:
3390 023202 032777 040000 155726 1$: BIT #BIT14,$SWR ;;LOOP ON PRESENT TEST?
3391 023210 001114 BNE $OVER ;;YES IF SW14=1
3392 ;*****START OF CODE FOR THE XOR TESTER*****
3393 023212 000416 $XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3394 023214 013746 000004 MOV $ERRVEC,-(SP) ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3395 023220 012737 023240 000004 MOV $SS,$ERRVEC ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3396 023226 005737 177060 TST $177060 ;;SET FOR TIMEOUT
3397 023232 012637 000004 MOV (SP)+,$ERRVEC ;;TIME OUT ON XOR?
3398 023236 000463 BR $SVLAD ;;RESTORE THE ERROR VECTOR
3399 023240 022626 5$: *P (SP)+,(SP)+ ;;GO TO THE NEXT TEST
3400 023242 012637 000004 MOV (SP)+,$ERRVEC ;;CLEAR THE STACK AFTER A TIME OUT
3401 023246 000423 BR 7$ ;;RESTORE THE ERROR VECTOR
3402 023250 6$;*****END OF CODE FOR THE XOR TESTER***** ;;LOOP ON THE PRESENT TEST
3403 023250 032777 000400 155660 BIT #BIT08,$SWR ;;LOOP ON SPEC. TEST?
3404 023256 001404 BEQ 2$ ;;BR IF NO
3405 023260 127737 155652 001102 CMPB $SWR,$TSTNM ;;ON THE RIGHT TEST? SWR<7:0>
3406 023266 001465 BEQ $OVER ;;BR IF YES
3407 023270 105737 001103 2$: TSTB SERFLG ;;HAS AN ERROR OCCURRED?
3408 023274 001421 BEQ 3$ ;;BR IF NO
3409 023276 123737 001115 001103 CMPB $ERMAX,SERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
3410 023304 101015 BHI 3$ ;;BR IF NO
3411 023306 032777 001000 155622 BIT #BIT09,$SWR ;;LOOP ON ERROR?
3412 023314 001404 BEQ 4$ ;;BR IF NO
3413 023316 013737 001110 001106 7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
3414 023324 000446 BR $OVER
3415 023326 105037 001103 4$: CLRB SERFLG ;;ZERO THE ERROR FLAG
3416 023332 005037 001164 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3417 023336 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
3418 023340 032777 004000 155570 3$: BIT #BIT11,$SWR ;;INHIBIT ITERATIONS?
3419 023346 001011 BNE 1$ ;;BR IF YES
3420 023350 005737 001206 TST $PASS ;;IF FIRST PASS OF PROGRAM
3421 023354 001406 BEQ 1$ ;; INHIBIT ITERATIONS
3422 023356 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
3423 023362 023737 001164 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
3424 023370 002024 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
3425 023372 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
3426 023400 013737 023456 001164 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
3427 023406 105237 001102 $SVLAD: INCB $TSTNM ;;COUNT TEST NUMBERS
3428 023412 113737 001102 001204 MOVB $TSTNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
  
```

```

3429 023420 011637 001106      MOV      (SP), $LPADR      ;; SAVE SCOPE LOOP ADDRESS
3430 023424 011637 001110      MOV      (SP), $LPERR     ;; SAVE ERROR LOOP ADDRESS
3431 023430 005037 001166      CLR      $ESCAPE         ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
3432 023434 112737 000001 001115      MOV      #1, $ERMAX       ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3433 023442 013777 001102 155470  SOVER:  MOV      $STNM, $DISPLAY  ;; DISPLAY TEST NUMBER
3434 023450 013716 001106      MOV      $LPADR, (SP)    ;; FUDGE RETURN ADDRESS
3435 023454 000002      RTI                     ;; FIXES PS
3436 023456 003720      SMXCNT: 2000.          ;; MAX. NUMBER OF ITERATIONS
3437 *****
3438
3439
3440 023460 112737 000001 023724  .SBTTL  APT COMMUNICATIONS ROUTINE
3441 023466 112737 000001 023722  $ATY1:  MOV      #1, $FFLG      ;; TO REPORT FATAL ERROR
3442 023474 000403      $ATY3:  MOV      #1, $MFLG     ;; TO TYPE A MESSAGE
3443 023476 112737 000001 023724  BR      $ATYC
3444 023504      $ATY4:  MOV      #1, $FFLG     ;; TO ONLY REPORT FATAL ERROR
3445 023504 010046      $ATYC:
3446 023506 010146      MOV      RO, -(SP)        ;; PUSH RO ON STACK
3447 023510 105737 023722      MOV      R1, -(SP)        ;; PUSH R1 ON STACK
3448 023514 001450      TSTB    $MFLG            ;; SHOULD TYPE A MESSAGE?
3449 023516 122737 000001 001220      BEQ     $S
3450 023524 001031      CMPB    $APTENV, $ENV     ;; OPERATING UNDER APT?
3451 023526 132737 000100 001221      BNE     $S
3452 023534 001425      BITB    $APTPOOL, $ENVM  ;; SHOULD SPOOL MESSAGES?
3453 023536 017600 000004      BEQ     $S
3454 023542 062766 000002 000004      MOV     #4(SP), RO        ;; GET MESSAGE ADDR.
3455 023550 005737 001200      ADD     #2, 4(SP)         ;; BUMP RETURN ADDR.
3456 023554 001375      1$:     TST     $MSGTYPE      ;; SEE IF DONE W/ LAST XMISSION?
3457 023556 010037 001214      BNE     $S
3458 023562 105720      MOV     RO, $MSGAD        ;; PUT ADDR IN MAILBOX
3459 023564 001376      2$:     TSTB    (RO)+         ;; FIND END OF MESSAGE
3460 023566 163700 001214      BNE     2$
3461 023572 006200      SUB     $MSGAD, RO        ;; SUB START OF MESSAGE
3462 023574 010037 001216      ASR     RO                ;; GET MESSAGE LGTH IN WORDS
3463 023600 012737 000004 001200      MOV     RO, $MSGGLT       ;; PUT LENGTH IN MAILBOX
3464 023606 000413      MOV     #4, $MSGTYPE     ;; TELL APT TO TAKE MSG.
3465 023610 017637 000004 023634  3$:     BR      $S
3466 023616 062766 000002 000004      MOV     #4(SP), 4$        ;; PUT MSG ADDR IN JSR LINKAGE
3467 023624 013746 177776      ADD     #2, 4(SP)         ;; BUMP RETURN ADDRESS
3468 023630 004737 025234      MOV     177776, -(SP)    ;; PUSH 177776 ON STACK
3469 023634 000000      JSR    PC, $TYPE         ;; CALL TYPE MACRO
3470 023636      4$:     .WORD    0
3471 023636 105737 023724      5$:     TSTB    $FFLG          ;; SHOULD REPORT FATAL ERROR?
3472 023642 001416      BEQ     12$
3473 023644 005737 001220      TST     $ENV             ;; RUNNING UNDER APT?
3474 023650 001413      BEQ     12$
3475 023652 005737 001200      11$:   TST     $MSGTYPE        ;; FINISHED LAST MESSAGE?
3476 023656 001375      BNE     11$
3477 023660 017637 000004 001202      MOV     #4(SP), $FATAL   ;; GET ERROR #
3478 023666 062766 000002 000004      ADD     #2, 4(SP)         ;; BUMP RETURN ADDR.
3479 023674 005237 001200      INC     $MSGTYPE         ;; TELL APT TO TAKE ERROR
3480 023700 105037 023724      12$:   CLRB    $FFLG          ;; CLEAR FATAL FLAG
3481 023704 105037 023723      CLRB    $LFLG           ;; CLEAR LOG FLAG
3482 023710 105037 023722      CLRB    $MFLG           ;; CLEAR MESSAGE FLAG
3483 023714 012601      MOV     (SP)+, R1        ;; POP STACK INTO R1
3484 023716 012600      MOV     (SP)+, RO        ;; POP STACK INTO RO

```

```

3485 023720 000207          RTS      PC          ;RETURN
3486
3487 023722      000          $MFLG: .BYTE 0          ;MESSG. FLAG
3488 023723      000          $LFLG: .BYTE 0          ;LOG FLAG
3489 023724      000          $FFLG: .BYTE 0          ;FATAL FLAG
3490          023726          .EVEN
3491          000200          APTSIZE=200
3492          000001          APTENV=001
3493          000100          APTSPool=100
3494          000040          APTCSNP=040
3495          ;*****
3496          .SBTTL ERROR HANDLER ROUTINE
3497
3498          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3499          ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3500          ;*AND GO TO SERRTYP ON ERROR
3501          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3502          ;*SW15=1      HALT ON ERROR
3503          ;*SW13=1      INHIBIT ERROR TYPEOUTS
3504          ;*SW10=1     BELL ON ERROR
3505          ;*SW09=1     LOOP ON ERROR
3506          ;*CALL
3507          ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3508
3509
3510 023726          SERROR:
3511 023726 113737 001102 002060 7$:  MOVB  $TSTNM,$TSTNUM
3512 023734 105237 001103          INCB  $ERFLG          ;;SET THE ERROR FLAG
3513 023740 001775          BEQ   7$          ;;DON'T LET THE FLAG GO TO ZERO
3514 023742 013777 001102 155170  MOV  $TSTNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
3515 023750 032777 002000 155160  BIT  #BIT10,$SWR      ;;BELL ON ERROR?
3516 023756 001402          BEQ   1$          ;;NO - SKIP
3517 023760 104400 001170          TYPE $BELL          ;;RING BELL
3518 023764 005237 001112 1$:  INC  $ERTTL          ;;COUNT THE NUMBER OF ERRORS
3519 023770 011637 001116  MOV  (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
3520 023774 162737 000002 001116  SUB  #2,$ERRPC
3521 024002 117737 155110 001114  MOVB $ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
3522 024010 032777 020000 155120  BIT  #BIT13,$SWR      ;;SKIP TYPEOUT IF SET
3523 024016 001004          BNE  20$          ;;SKIP TYPEOUTS
3524 024020 004737 024130  JSR  PC,$SERRTYP      ;;GO TO USER ERROR ROUTINE
3525 024024 104400 001175          TYPE ,SCLF
3526 024030 20$:  CMPB #APTENV,$ENV      ;;RUNNING IN APT MODE
3527 024030 122737 000001 001220  BNE  2$          ;;NO SKIP APT ERROR REPORT
3528 024036 001007          MOVB $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
3529 024040 113737 001114 024052  JSR  PC,$SATY4      ;;REPORT FATAL ERROR TO APT
3530 024046 004737 023476
3531 024052      000          21$: .BYTE 0
3532 024053      000          .BYTE 0
3533 024054 000777          22$: BR   22$          ;;APT ERROR LOOP
3534 024056 005777 155054 2$:  TST  $SWR          ;;HALT ON ERROR
3535 024062 100006          BPL  3$          ;;SKIP IF CONTINUE
3536 024064 000000          HALT          ;;HALT ON ERROR!
3537 024066 022737 023100 000042  CMP  #SENDAD,$#42   ;;ACT-11 AUTO-ACCEPT?
3538 024074 001001          BNE  3$          ;;BRANCH IF NO
3539 024076 000000          HALT          ;;YES
3540 024100 032777 001000 155030 3$:  BIT  #BIT09,$SWR      ;;LOOP ON ERROR SWITCH SET?

```

3541	024106	001402		BEQ	4S	::BR IF NO
3542	024110	013716	001110	MOV	\$LPERR,(SP)	::FUDGE RETURN FOR LOOPING
3543	024114	005737	001166	4S: TST	\$ESCAPE	::CHECK FOR AN ESCAPE ADDRESS
3544	024120	001402		BEQ	5S	::BR IF NONE
3545	024122	013716	001166	MOV	\$ESCAPE,(SP)	::FUDGE RETURN ADDRESS FOR ESCAPE
3546	024126			5S:		
3547	024126	000002		RTI		::RETURN

```

3548
3549
3550
3551
3552
3553
3554
3555
3556
3557 024130
3558 024130 104400 001175
3559 024134 010046
3560 024136 005000
3561 024140 153700 001114
3562 024144 001004
3563
3564 024146 013746 001116
3565
3566 024152 104401
3567 024154 000426
3568 024156 005300
3569 024160 006300
3570 024162 006300
3571 024164 006300
3572 024166 062700 001324
3573 024172 012037 024202
3574 024176 001404
3575 024200 104400
3576 024202 000000
3577 024204 104400 001175
3578 024210 012037 024220
3579 024214 001404
3580 024216 104400
3581 024220 000000
3582 024222 104400 001175
3583 024226 011000
3584 024230 001004
3585 024232 012600
3586 024234 104400 001175
3587 024240 000207
3588 024242
3589 024242 013046
3590 024244 104401
3591 024246 005710
3592 024250 001770
3593 024252 104400 024260
3594 024256 000771
3595 024260 020040 000
3596 024264
    
```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

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

```

$ERRTYP:
      TYPE      $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
      MOV      RD,-(SP)    ;; SAVE RD
      CLR      RD          ;; PICKUP THE ITEM INDEX
      BISB     @#$ITEMB,RD
      BNE      1$        ;; IF ITEM NUMBER IS ZERO, JUST
                          ;; TYPE THE PC OF THE ERROR
      MOV      $ERRPC,-(SP) ;; SAVE $ERRPC FOR TYPEOUT
                          ;; ERROR ADDRESS
                          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
                          ;; GET OUT
      BR       6$        ;; ADJUST THE INDEX SO THAT IT WILL
                          ;; WORK FOR THE ERROR TABLE
1$:   DEC      RD
      ASL      RD
      ASL      RD
      ASL      RD
      ADD      @#$ERRTB,RD ;; FORM TABLE POINTER
      MOV      (RD)+,2$   ;; PICKUP "ERROR MESSAGE" POINTER
      BEQ      3$        ;; SKIP TYPEOUT IF NO POINTER
                          ;; TYPE THE "ERROR MESSAGE"
2$:   .WORD   0          ;; "ERROR MESSAGE" POINTER GOES HERE
      TYPE     $CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
3$:   MOV      (RD)+,4$   ;; PICKUP "DATA HEADER" POINTER
      BEQ      5$        ;; SKIP TYPEOUT IF 0
                          ;; TYPE THE "DATA HEADER"
4$:   .WORD   0          ;; "DATA HEADER" POINTER GOES HERE
      TYPE     $CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
5$:   MOV      (RD),RD    ;; PICKUP "DATA TABLE" POINTER
      BNE      7$        ;; GO TYPE THE DATA
6$:   MOV      (SP)+,RD   ;; RESTORE RD
      TYPE     $CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
7$:   RTS      PC        ;; RETURN
      MOV      @2(RD)+,-(SP) ;; SAVE @2(RD)+ FOR TYPEOUT
      TYPOC   (RD)      ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
      TST     (RD)      ;; IS THERE ANOTHER NUMBER?
      BEQ     6$        ;; BR IF NO
      TYPE    8$        ;; TYPE TWO(2) SPACES
8$:   BR      7$        ;; LOOP
      .ASCIZ  / /      ;; TWO(2) SPACES
      .EVEN
    
```



```

3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624 024264 017646 000000
3625 024270 116637 000001 024507
3626 024276 112637 024511
3627 024302 062716 000002
3628 024306 000406
3629 024310 112737 000001 024507
3630 024316 112737 000006 024511
3631 024324 112737 000005 024506
3632 024332 010346
3633 024334 010446
3634 024336 010546
3635 024340 113704 024511
3636 024344 005404
3637 024346 062704 000006
3638 024352 110437 024510
3639 024356 113704 024507
3640 024362 016605 000012
3641 024366 005003
3642 024370 006105
3643 024372 000404
3644 024374 006105
3645 024376 006105
3646 024400 006105
3647 024402 010503
3648 024404 006103
3649 024406 105337 024510
3650 024412 100016
3651 024414 042703 177770
3652 024420 001002

```

```

;*****
.SBTTL BINARY TO OCTAL (ASCII) AND TYPE
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
;OCTAL (ASCII) NUMBER AND TYPE IT.
;STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;CALL:
;      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
;      TYPOS   ;;CALL FOR TYPEOUT
;      .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;      .BYTE  M              ;;M=1 OR 0
;                               ;;1=TYPE LEADING ZEROS
;                               ;;0=SUPPRESS LEADING ZEROS
;STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
;STYPOS OR STYPOC
;CALL:
;      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
;      TYPON   ;;CALL FOR TYPEOUT
;STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
;CALL:
;      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
;      TYPOC   ;;CALL FOR TYPEOUT
;STYPOS: MOV     2(SP),-(SP)    ;;PICKUP THE MODE
;        MOVVB  1(SP),SOFILL   ;;LOAD ZERO FILL SWITCH
;        MOVVB  (SP)+,SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
;        ADD    #2,(SP)        ;;ADJUST RETURN ADDRESS
;        BR     STYPON
;STYPOC: MOVVB  #1,SOFILL      ;;SET THE ZERO FILL SWITCH
;        MOVVB  #6,SOMODE+1    ;;SET FOR SIX(6) DIGITS
;STYPON: MOVVB  #5,SOCNT       ;;SET THE ITERATION COUNT
;        MOV    R3,-(SP)       ;;SAVE R3
;        MOV    R4,-(SP)       ;;SAVE R4
;        MOV    R5,-(SP)       ;;SAVE R5
;        MOVVB  SOMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
;        NEG    R4
;        ADD    #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
;        MOVVB  R4,SOMODE      ;;SAVE IT FOR USE
;        MOVVB  SOFILL,R4     ;;GET THE ZERO FILL SWITCH
;        MOV    12(SP),R5     ;;PICKUP THE INPUT NUMBER
;        CLR    R3            ;;CLEAR THE OUTPUT WORD
;1$:     ROL    R5             ;;ROTATE MSB INTO "C"
;        BR     3$            ;;GO DO MSB
;2$:     ROL    R5             ;;FORM THIS DIGIT
;        ROL    R5
;        ROL    R5
;        MOV    R5,R3
;3$:     ROL    R3             ;;GET LSB OF THIS DIGIT
;        DECB  SOMODE         ;;TYPE THIS DIGIT?
;        BPL   7$             ;;BR IF NO
;        BIC   #177770,R3    ;;GET RID OF JUNK
;        BNE   4$             ;;TEST FOR 0

```

```

3653 024422 005704          TST      R4          ;;SUPPRESS THIS 0?
3654 024424 001403          BEQ     S$          ;;BR IF YES
3655 024426 005204          4$:    INC     R4          ;;DON'T SUPPRESS ANYMORE 0'S
3656 024430 052703 000060    BIS     #'0,R3      ;;MAKE THIS DIGIT ASCII
3657 024434 052703 000040    5$:    BIS     #' ,R3    ;;MAKE ASCII IF NOT ALREADY
3658 024440 110337 024504    MOVB   R3,S$        ;;SAVE FOR TYPING
3659 024444 104400 024504    TYPE   S$          ;;GO TYPE THIS DIGIT
3660 024450 105337 024506    7$:    DECB   $SOCNT   ;;COUNT BY 1
3661 024454 003347          BGT     2$          ;;BR IF MORE TO DO
3662 024456 002402          BLT     6$          ;;BR IF DONE
3663 024460 005204          INC     R4          ;;INSURE LAST DIGIT ISN'T A BLANK
3664 024462 000744          BR      2$          ;;GO DO THE LAST DIGIT
3665 024464 012605          6$:    MOV     (SP)+,R5  ;;RESTORE R5
3666 024466 012604          MOV     (SP)+,R4  ;;RESTORE R4
3667 024470 012603          MOV     (SP)+,R3  ;;RESTORE R3
3668 024472 016666 000002 000004  MOV     2(SP),4(SP) ;;SET THE STACK FOR RETURNING
3669 024500 012616          MOV     (SP)+,(SP)
3670 024502 000002          RTI                    ;;RETURN
3671 024504          8$:    .BYTE   0          ;;STORAGE FOR ASCII DIGIT
3672 024505          .BYTE   0          ;;TERMINATOR FOR TYPE ROUTINE
3673 024506          .BYTE   0          ;;OCTAL DIGIT COUNTER
3674 024507          .BYTE   0          ;;ZERO FILL SWITCH
3675 024510 000000          .WORD   0          ;;NUMBER OF DIGITS TO TYPE
*****
.SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
; *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
; *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
; *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
; *REPLACED WITH SPACES.
; *CALL:
; *      MOV     NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
; *      TYPDS                    ;;GO TO THE ROUTINE
STYPDS:
MOV     R0,-(SP)      ;;PUSH R0 ON STACK
MOV     R1,-(SP)      ;;PUSH R1 ON STACK
MOV     R2,-(SP)      ;;PUSH R2 ON STACK
MOV     R3,-(SP)      ;;PUSH R3 ON STACK
MOV     R5,-(SP)      ;;PUSH R5 ON STACK
MOV     #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
MOV     20(SP),R5     ;;GET THE INPUT NUMBER
BPL     1$           ;;BR IF INPUT IS POS.
NEG     R5           ;;MAKE THE BINARY NUMBER POS.
MOVB   #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
1$:    CLR     R0     ;;ZERO THE CONSTANTS INDEX
MOV     #SDBLK,R3    ;;SETUP THE OUTPUT POINTER
MOVB   #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
2$:    CLR     R2     ;;CLEAR THE BCD NUMBER
MOV     $DTBL(R0),R1 ;;GET THE CONSTANT
3$:    SUB     R1,R5   ;;FORM THIS BCD DIGIT
BLT     4$           ;;BR IF DONE
INC     R2           ;;INCREASE THE BCD DIGIT BY 1
BR      3$

```

3709	024576	060105		4\$:	ADD	R1,R5	::	ADD BACK THE CONSTANT
3710	024600	005702			TST	R2	::	CHECK IF BCD DIGIT=0
3711	024602	001002			BNE	5\$::	FALL THROUGH IF 0
3712	024604	105716			TSTB	(SP)	::	STILL DOING LEADING 0'S?
3713	024606	100407			BMI	7\$::	BR IF YES
3714	024610	106316		5\$:	ASLB	(SP)	::	MSD?
3715	024612	103003			BCC	6\$::	BR IF NO
3716	024614	116663	000001 177777		MOVB	1(SP),-1(R3)	::	YES--SET THE SIGN
3717	024622	052702	000060	6\$:	BIS	#'0,R2	::	MAKE THE BCD DIGIT ASCII
3718	024626	052702	000040	7\$:	BIS	#' ,R2	::	MAKE IT A SPACE IF NOT ALREADY A DIGIT
3719	024632	110223			MOVB	R2,(R3)+	::	PUT THIS CHARACTER IN THE OUTPUT BUFFER
3720	024634	005720			TST	(R0)+	::	JUST INCREMENTING
3721	024636	020027	000010		CMP	R0,#10	::	CHECK THE TABLE INDEX
3722	024642	002746			BLT	2\$::	GO DO THE NEXT DIGIT
3723	024644	003002			BGT	8\$::	GO TO EXIT
3724	024646	010502			MOV	R5,R2	::	GET THE LSD
3725	024650	000764			BR	6\$::	GO CHANGE TO ASCII
3726	024652	105726		8\$:	TSTB	(SP)+	::	WAS THE LSD THE FIRST NON-ZERO?
3727	024654	100003			BPL	9\$::	BR IF NO
3728	024656	116663	177777 177776		MOVB	-1(SP),-2(R3)	::	YES--SET THE SIGN FOR TYPING
3729	024664	105013		9\$:	CLRB	(R3)	::	SET THE TERMINATOR
3730	024666	012605			MOV	(SP)+,R5	::	POP STACK INTO R5
3731	024670	012603			MOV	(SP)+,R3	::	POP STACK INTO R3
3732	024672	012602			MOV	(SP)+,R2	::	POP STACK INTO R2
3733	024674	012601			MOV	(SP)+,R1	::	POP STACK INTO R1
3734	024676	012600			MOV	(SP)+,R0	::	POP STACK INTO R0
3735	024700	104400	024726		TYPE	\$DBLK	::	NOW TYPE THE NUMBER
3736	024704	016666	000002 000004		MOV	2(SP),4(SP)	::	ADJUST THE STACK
3737	024712	012616			MOV	(SP)+,(SP)		
3738	024714	000002			RTI		::	RETURN TO USER
3739	024716	023420		\$DTBL:	1000.			
3740	024720	001750			1000.			
3741	024722	000144			100.			
3742	024724	000012			10.			
3743	024726	000004		\$DBLK:	.BLKW 4			

```

3744
3745 ;*****
3746
3747 .SBTTL POWER DOWN AND UP ROUTINES
3748
3749 :POWER DOWN ROUTINE
3750 024736 012737 025064 000024 $PWRDN: MOV $SILLUP, @#PWRVEC ;; SET FOR FAST UP
3751 024744 012737 000340 000026 MOV #340, @#PWRVEC+2 ;; PRIO:7
3752 024752 010046 MOV R0, -(SP) ;; PUSH R0 ON STACK
3753 024754 010146 MOV R1, -(SP) ;; PUSH R1 ON STACK
3754 024756 010246 MOV R2, -(SP) ;; PUSH R2 ON STACK
3755 024760 010346 MOV R3, -(SP) ;; PUSH R3 ON STACK
3756 024762 010446 MOV R4, -(SP) ;; PUSH R4 ON STACK
3757 024764 010546 MOV R5, -(SP) ;; PUSH R5 ON STACK
3758 024766 010637 025070 MOV SP, $SAVR6 ;; SAVE SP
3759 024772 012737 025004 000024 MOV $PWRUP, @#PWRVEC ;; SET UP VECTOR
3760 025000 000000 HALT
3761 025002 000776 BR -2 ;; HANG UP
3762
3763 :POWER UP ROUTINE
3764 025004 013706 025070 $PWRUP: MOV $SAVR6, SP ;; GET SP
3765 025010 005037 025070 CLR $SAVR6 ;; WAIT LOOP FOR THE TTY
3766 025014 005237 025070 IS: INC $SAVR6 ;; WAIT FOR THE INC
3767 025020 001375 BNE IS OF WORD
3768 025022 012605 MOV (SP)+, R5 ;; POP STACK INTO R5
3769 025024 012604 MOV (SP)+, R4 ;; POP STACK INTO R4
3770 025026 012603 MOV (SP)+, R3 ;; POP STACK INTO R3
3771 025030 012602 MOV (SP)+, R2 ;; POP STACK INTO R2
3772 025032 012601 MOV (SP)+, R1 ;; POP STACK INTO R1
3773 025034 012600 MOV (SP)+, R0 ;; POP STACK INTO R0
3774 025036 012737 024736 000024 MOV $PWRDN, @#PWRVEC ;; SET UP THE POWER DOWN VECTOR
3775 025044 012737 000340 000026 MOV #340, @#PWRVEC+2 ;; PRIO:7
3776 025052 104400 TYPE PWRMSG ;; REPORT THE POWER FAILURE
3777 025054 025072 $PWRMG: .WORD PWRMSG ;; POWER FAIL MESSAGE POINTER
3778 025056 012716 MOV (PC)+, (SP) ;; RESTART AT BEGIN
3779 025060 002144 $PWRAD: .WORD BEGIN ;; RESTART ADDRESS
3780 025062 000002 RTI
3781 025064 000000 $SILLUP: HALT ;; THE POWER UP SEQUENCE WAS STARTED
3782 025066 000776 BR -2 ;; BEFORE THE POWER DOWN WAS COMPLETE
3783 025070 000000 $SAVR6: 0 ;; PUT THE SP HERE
3784 025072 005015 042522 052123 PWRMSG: .ASCIZ <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
3785 025100 051101 044524 043516
3786 025106 040440 052106 051105
3787 025114 040440 050040 053517
3788 025122 051105 043040 044501
3789 025130 052514 042522 005015
3790 025136 000
3791 025140 .EVEN

```

```

3792 ;*****
3793
3794 .SBTTL ROUTINE TO SIZE MEMORY
3795
3796 ;*CALL:
3797 ;* JSR PC,$SIZE
3798 ;* RETURN
3799 ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
3800
3801 $SIZE: MOV RO,-(SP) ;;SAVE RO ON THE STACK
3802 MOV R1,-(SP) ;;SAVE R1 ON THE STACK
3803 MOV @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
3804 MOV @#ERRVEC+2,-(SP)
3805 MOV SP,RO ;;SAVE THE STACK POINTER
3806 MOV @#PS,@#ERRVEC+2 ;;SET ERRVEC PS TO PRESENT PS
3807 MOV #2$,@#ERRVEC ;;SET FOR TIMEOUT
3808 MOV #20000,R1 ;;FIRST ADDRESS
3809 1$: TST (R1)+ ;;TEST THIS ADDRESS
3810 BR 1$ ;;TRY ANOTHER
3811 2$: SUB #4,R1 ;;DROP BACK
3812 MOV RO,SP ;;RESTORE THE STACK
3813 MOV (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
3814 MOV (SP)+,@#ERRVEC
3815 MOV R1,$LSTAD ;;LAST ADDRESS
3816 MOV (SP)+,R1 ;;RESTORE R1
3817 MOV (SP)+,RO ;;RESTORE RO
3818 RTS PC
3819 $LSTAD: .WORD 0 ;;CONTAINS THE LAST ADDRESS
    
```

3820
3821
3822
3823
3824
3825
3826
3827
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

```

;*****
.SBTTL TYPE ROUTINE
;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;*
;*CALL:
;*1) USING A TRAP INSTRUCTION
;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;*OR
;* TYPE
;* MESADR
;*
;*2) USING A JSR INSTRUCTION
;* MOV PS,-(SP) ;;PUSH PROCESSOR STATUS WORD ON THE STACK
;* JSR PC,$TYPE ;;CALL TYPE ROUTINE
;* MESADDR ;;FIRST ADDRESS OF MESSAGE
;*
;* IS THERE A TERMINAL?
;* BR IF YES
;* HALT HERE IF NO TERMINAL
;* LEAVE
;* SAVE RO
;* GET ADDRESS OF ASCIZ STRING
;* RUNNING IN APT MODE
;* NO GO CHECK FOR APT CONSOLE
;* SPOOL MESSAGE TO APT
;* NO GO CHECK FOR CONSOLE
;* SETUP MESSAGE ADDRESS FOR APT
;* SPOOL MESSAGE TO APT
;* MESSAGE ADDRESS
;* APT CONSOLE SUPPRESSED
;* YES, SKIP TYPE OUT
;* PUSH CHARACTER TO BE TYPED ONTO STACK
;* BR IF IT ISN'T THE TERMINATOR
;* IF TERMINATOR POP IT OFF THE STACK
;* RESTORE RO
;* ADJUST RETURN PC
;* RETURN
;* BRANCH IF <HT>
;* BRANCH IF NOT <CRLF>
;* POP <CR><LF> EQUIV
;* TYPE CR AND LF
;*
;* GET NEXT CHARACTER
;* GO TYPE THIS CHARACTER
;* IS IT TIME FOR FILLER CHARS.?
;* IF NO GO GET NEXT CHAR.

```

025234	105737	001155	\$TYPE:	TSTB	\$TPFLG		
025240	100002			BPL	1\$		
025242	000000			HALT			
025244	000430			BR	3\$		
025246	010046		1\$:	MOV	RO,-(SP)		
025250	017600	000002		MOV	22(SP),RO		
025254	122737	000001	001220	CMPB	#APTENV,\$ENV		
025262	001011			BNE	62\$		
025264	132737	000100	001221	BITB	#APTSPool,\$ENVm		
025272	001405			BEQ	62\$		
025274	010037	025304		MOV	RO,61\$		
025300	004737	023466		JSR	PC,\$ATY3		
025304	000000		61\$:	.WORD	0		
025306	132737	000040	001221	62\$:	BITB	#APTCSUP,\$ENVm	
025314	001003			BNE	60\$		
025316	112046		2\$:	MOVB	(RO)+,-(SP)		
025320	001005			BNE	4\$		
025322	005726			TST	(SP)+		
025324	012600		60\$:	MOV	(SP)+,RO		
025326	062716	000002	3\$:	ADD	#2,(SP)		
025332	000002			RTI			
025334	122716	000011	4\$:	CMPB	#THT,(SP)		
025340	001431			BEQ	8\$		
025342	122716	000200		CMPB	#TCRLF,(SP)		
025346	001007			BNE	5\$		
025350	005726			TST	(SP)+		
025352	013746	177776		MOV	PS,-(SP)		
025356	004737	025234		JSR	PC,\$TYPE		
025362	001175			\$CRLF			
025364	000754			BR	2\$		
025366	004737	025450	5\$:	JSR	PC,\$TYPEC		
025372	123726	001154	6\$:	CMPB	\$FILLC,(SP)+		
025376	001347			BNE	2\$		

```

3876 025400 013746 001152      MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
3877                                ;;AND THE NULL CHAR.
3878 025404 105366 000001      7$:     DECB      1(SP)      ;;DOES A NULL NEED TO BE TYPED?
3879 025410 002770                BLT      6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
3880 025412 004737 025450                JSR      PC,$TYPEC        ;;GO TYPE A NULL
3881 025416 105337 025514                DECB      $CHARCNT        ;;DO NOT COUNT AS A COUNT
3882 025422 000770                BR       7$                ;;LOOP
3883
3884                                ;HORIZONTAL TAB PROCESSOR
3885
3886 025424 112716 000040      8$:     MOVB      #40,(SP)      ;;REPLACE TAB WITH SPACE
3887 025430 004737 025450      9$:     JSR      PC,$TYPEC        ;;TYPE A SPACE
3888 025434 132737 000007 025514      BITB     #7,$CHARCNT        ;;BRANCH IF NOT AT
3889 025442 001372                BNE      9$                ;;TAB STOP
3890 025444 005726                TST      (SP)+            ;;POP SPACE OFF STACK
3891 025446 000723                BR       2$                ;;GET NEXT CHARACTER
3892 025450 105777 153472      $TYPEC: TSTB     @STPS        ;;WAIT UNTIL PRINTER IS READY
3893 025454 100375                BPL      $TYPEC
3894 025456 116677 000002 153464      MOVB     2(SP),@STPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3895 025464 122766 000015 000002      CMPB     #15,2(SP)        ;;BRANCH IF
3896 025472 001003                BNE      1$                ;;NOT <CR>
3897 025474 105037 025514      CLRB     $CHARCNT
3898 025500 000406                BR       $TYPEX
3899 025502 122766 000012 000002 1$:     CMPB     #12,2(SP)        ;;EXIT
3900 025510 002002                BGE      $TYPEX            ;;BRANCH IF
3901 025512 105227                INCB     (PC)+            ;;<LF>
3902 025514 000000                $CHARCNT: .WORD 0        ;;INC SPACE
3903 025516 000207                $TYPEX:  RTS      PC      ;;COUNT
3904                                ;;
3905                                EQUATES
3906                                THT=11
3907                                TCRLF=200
3908

```

009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100

.SBTTL TRAP DECODER

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

025520 010046
025522 016600 000002
025524 005740
025526 111000
025528 006300
025530 016000 025542
025532
025534
025536
025538
025540 000200

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

.SBTTL TRAP TABLE

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

025542 025234
025544 024310
025546 024264
025548 024324
025550 024512

ROUTINE

\$TRAPD:

STYPE ;;CALL=TYPE TRAP+0(104400) TTY TYPEOUT ROUTINE
STYPOC ;;CALL=TYPOC TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
STYPOS ;;CALL=TYPOS TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
STYPON ;;CALL=TYPON TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
STYPDS ;;CALL=TYPDS TRAP+4(104404) TYPE DECIMAL NUMBER (WITH SIGN)

Line	Hex 1	Hex 2	Hex 3	Hex 4	Label	Text
3941						
3942	025554	005015	051412	040524	HEADER:	.SBTTL ASCII MESSAGES
3943	025562	052122	047440	020106		.ASCIZ <15><12><12>/START OF MD-11-DZVSA-A VS60 INSTRUCTION TEST PART I/<15><1
3944	025570	042115	030455	026461		
3945	025576	055104	051526	026501		
3946	025604	020101	053040	033123		
3947	025612	020060	047111	052123		
3948	025620	052522	052103	047511		
3949	025626	020116	042524	052123		
3950	025634	050040	051101	020124		
3951	025642	006511	000012			
3952						
3953	025646	040515	047111	027124	EM1:	.ASCIZ /MAINT. SWITCH REG/
3954	025654	051440	044527	041524		
3955	025662	020110	042522	000107		
3956	025670	047514	042101	047111	EM2:	.ASCIZ /LOADING D.P.C. REG <STATIC>/
3957	025676	020107	027104	027120		
3958	025704	027103	051040	043505		
3959	025712	036040	052123	052101		
3960	025720	041511	000076			
3961	025724	047514	042101	047111	EM3:	.ASCIZ /LOADING MODE REG/
3962	025732	020107	047515	042504		
3963	025740	051040	043505	000		
3964	025745	123	050505	042525	EM4:	.ASCIZ /SEQUENCING DPC REG/
3965	025752	041516	047111	020107		
3966	025760	050104	020103	042522		
3967	025766	000107				
3968	025770	047514	042101	047111	EM5:	.ASCIZ /LOADING LINE TYPE REG/
3969	025776	020107	044514	042516		
3970	026004	052040	050131	020105		
3971	026012	042522	000107			
3972	026016	047514	042101	047111	EM6:	.ASCIZ /LOADING INTENSITY LEVEL REG/
3973	026024	020107	047111	042524		
3974	026032	051516	052111	020131		
3975	026040	042514	042526	020114		
3976	026046	042522	000107			
3977	026052	047514	042101	047111	EM7:	.ASCIZ /LOADING GRAPHPLOT INC REG/
3978	026060	020107	051107	050101		
3979	026066	050110	047514	020124		
3980	026074	047111	020103	042522		
3981	026102	000107				
3982	026104	044504	050123	040514	EM10:	.ASCIZ /DISPLAY JUMP ABS TO AN ADRS/
3983	026112	020131	052512	050115		
3984	026120	040440	051502	052040		
3985	026126	020117	047101	040440		
3986	026134	051104	000123			
3987	026140	044504	050123	040514	EM11:	.ASCIZ /DISPLAY JUMP REL TO AN ADRS/
3988	026146	020131	052512	050115		
3989	026154	051040	046105	052040		
3990	026162	020117	047101	040440		
3991	026170	051104	000123			
3992	026174	044504	050123	040514	EM12:	.ASCIZ /DISPLAY JSR ABS TO AN ADRS/
3993	026202	020131	051512	020122		
3994	026210	041101	020123	047524		
3995	026216	040440	020116	042101		
3996	026224	051522	000			

3997	026227	104	051511	046120	EM13:	.ASCIZ	/DISPLAY JSR REL TO AN ADRS/
3998	026234	054501	045040	051123			
3999	026242	051040	046105	052040			
4000	026250	020117	047101	040440			
4001	026256	051104	000123				
4002	026262	047514	042101	047111	EM14:	.ASCIZ	/LOADING X POS REG/
4003	026270	020107	020130	047520			
4004	026276	020123	042522	000107			
4005	026304	047514	042101	047111	EM15:	.ASCIZ	/LOADING Y POS REG/
4006	026312	020107	020131	047520			
4007	026320	020123	042522	000107			
4008	026326	052502	020123	044524	EM16:	.ASCIZ	/BUS TIME-OUT ER/
4009	026334	042515	047455	052125			
4010	026342	042440	000122				
4011	026346	044103	051101	052040	EM17:	.ASCIZ	/CHAR TERMINATE ER/
4012	026354	051105	044515	040516			
4013	026362	042524	042440	000122			
4014	026370	020131	054504	040516	EM20:	.ASCIZ	/Y DYNAMIC OFFSET ER/
4015	026376	044515	020103	043117			
4016	026404	051506	052105	042440			
4017	026412	000122					
4018	026414	020130	054504	040516	EM21:	.ASCIZ	/X DYNAMIC OFFSET ER/
4019	026422	044515	020103	043117			
4020	026430	051506	052105	042440			
4021	026436	000122					
4022	026440	042522	047514	040503	EM22:	.ASCIZ	/RELOCATE REG ER/
4023	026446	042524	051040	043505			
4024	026454	042440	000122				
4025	026460	027104	027120	027125	EM23:	.ASCIZ	/D.P.U. NAME REG ER/
4026	026466	047040	046501	020105			
4027	026474	042522	020107	051105			
4028	026502	000					
4029	026503	123	040505	041522	EM24:	.ASCIZ	/SEARCH CODE OF ASSOC. NAME REG ER/
4030	026510	020110	047503	042504			
4031	026516	047440	020106	051501			
4032	026524	047523	027103	047040			
4033	026532	046501	020105	042522			
4034	026540	020107	051105	000			
4035	026545	126	041505	047524	EM25:	.ASCIZ	/VECTOR SCALE ER/
4036	026552	020122	041523	046101			
4037	026560	020105	051105	000			
4038	026565	103	040510	020122	EM26:	.ASCIZ	/CHAR SCALE ER/
4039	026572	041523	046101	020105			
4040	026600	051105	000				
4041	026603	102	044514	045516	EM27:	.ASCIZ	/BLINK LOGIC ER/
4042	026610	046040	043517	041511			
4043	026616	042440	000122				
4044	026622	052111	046101	041511	EM30:	.ASCIZ	/ITALIC LOGIC ER/
4045	026630	046040	043517	041511			
4046	026636	042440	000122				
4047	026642	042515	052516	046040	EM31:	.ASCIZ	/MENU LOGIC ER/
4048	026650	043517	041511	042440			
4049	026656	000122					
4050	026660	042522	042523	020124	EM32:	.ASCIZ	/RESET FAILED TO CLEAR A REG/
4051	026666	040506	046111	042105			
4052	026674	052040	020117	046103			

4053	026702	040505	020122	020101	
4054	026710	042522	000107		
4055	026714	052123	050117	044440	EM33: .ASCIZ /STOP INTR ER/
4056	026722	052116	020122	051105	
4057	026730	000			
4058	026731	114	050056	020056	EM34: .ASCIZ /L.P. FLAG ER/
4059	026736	046106	043501	042440	
4060	026744	000122			
4061	026746	047503	047514	020122	EM35: .ASCIZ /COLOR LEVEL ER/
4062	026754	042514	042526	020114	
4063	026762	051105	000		
4064	026765	111	042524	051516	EM36: .ASCIZ /ITENSITY ENABLE - CONS 0 + 1 ER/
4065	026772	052111	020131	047105	
4066	027000	041101	042514	026440	
4067	027006	041440	047117	020123	
4068	027014	020060	020053	020061	
4069	027022	051105	000		
4070	027025	111	052116	051105	EM37: .ASCIZ /INTERNAL STOP FLAG FAILED TO SET/
4071	027032	040516	020114	052123	
4072	027040	050117	043040	040514	
4073	027046	020107	040506	046111	
4074	027054	042105	052040	020117	
4075	027062	042523	000124		
4076	027066	047111	042524	047122	EM40: .ASCIZ /INTERNAL STOP FLAG FAILED TO CLEAR/
4077	027074	046101	051440	047524	
4078	027102	020120	046106	043501	
4079	027110	043040	044501	042514	
4080	027116	020104	047524	041440	
4081	027124	042514	051101	000	
4082	027131	127	051117	020104	EM41: .ASCIZ /WORD 0-1-2 INDICATOR ER/
4083	027136	026460	026461	020062	
4084	027144	047111	044504	040503	
4085	027152	047524	020122	051105	
4086	027160	000			
4087	027161	114	050056	020056	EM42: .ASCIZ /L.P. INTR ENABLE - CONS 0 + 1 ER/
4088	027166	047111	051124	042440	
4089	027174	040516	046102	020105	
4090	027202	020055	047503	051516	
4091	027210	030040	025440	030440	
4092	027216	042440	000122		
4093	027222	027114	027120	051440	EM43: .ASCIZ /L.P. SWITCH INTR ENABLE - CONS 0 + 1 ER/
4094	027230	044527	041524	020110	
4095	027236	047111	051124	042440	
4096	027244	040516	046102	020105	
4097	027252	020055	047503	051516	
4098	027260	030040	025440	030440	
4099	027266	042440	000122		
4100	027272	044504	050123	040514	EM44: .ASCIZ /DISPLAY BUSY ER/
4101	027300	020131	052502	054523	
4102	027306	042440	000122		
4103	027312	054105	042524	047122	EM45: .ASCIZ /EXTERNAL STOP LOGIC ER/
4104	027320	046101	051440	047524	
4105	027326	020120	047514	044507	
4106	027334	020103	051105	000	
4107	027341	124	046511	026505	EM46: .ASCIZ /TIME-OUT INTR ER/
4108	027346	052517	020124	047111	

4109	027354	051124	042440	000122
4110	027362	040516	042515	046440
4111	027370	052101	044103	044440
4112	027376	052116	020122	051105
4113	027404	000		
4114	027405	114	050056	020056
4115	027412	046106	043501	044440
4116	027420	052116	020122	051105
4117	027426	000		
4118	027427	123	040524	045503
4119	027434	050040	051124	042440
4120	027442	000122		
4121	027444	051105	050122	020103
4122	027452	020040	051524	047124
4123	027460	046525	020040	052502
4124	027466	040523	051104	020040
4125	027474	054105	041520	020124
4126	027502	020040	041522	042126
4127	027510	000		
4128		027512		
4129	027512	001116	002060	001122
4130	027520	001124	001126	000000
4131				
4132				
4133				
4134				
4135				
4136	027526	000000		
4137				
4138		000001		

EM47: .ASCIZ /NAME MATCH INTR ER/

EM50: .ASCIZ /L.P. FLAG INTR ER/

EM51: .ASCIZ /STACK PTR ER/

DH1: .ASCIZ /ERRPC TSTNUM BUSADR EXPCT RCVD/

DT1: .EVEN
\$ERRPC, TSTNUM, \$BDADR, \$GDDAT, \$BDDAT, 0

; THIS IS THE WORKING AREA FOR ALL VS60 NPR'S (INSTRS & DATA)
; FROM HERE TO THE END OF 28K

BUFFER: 0

.END

BIT01 = 000002	306#	316												
BIT02 = 000004	305#	315												
BIT03 = 000010	304#	314												
BIT04 = 000020	303#	313												
BIT05 = 000040	302#	312												
BIT06 = 000100	301#	311												
BIT07 = 000200	300#	310												
BIT08 = 000400	299#	309	3403											
BIT09 = 001000	298#	308	3411	3540										
BIT1 = 000002	316#	1282												
BIT10 = 002000	297#	3515												
BIT11 = 004000	296#	3418												
BIT12 = 010000	295#	1283	1299	1320	3307									
BIT13 = 020000	294#	1337	1353	1368	1383	1397	1439	1464	1483	1505	1565	1599	1633	
	1667	1682	1700	1735	1750	1766	1784	1798	1814	1832	1846	1860	1875	
	1890	1906	1924	1940	1956	1975	1994	2050	2066	2083	2100	2119	2136	
	2157	2174	2194	2213	2233	2252	2271	2284	2304	2321	2341	2357	2377	
	2393	2413	2429	2449	2465	2485	2501	2521	2540	2558	2575	2594	2613	
	2636	2661	2684	2704	2719	2743	2770	2792	2815	2834	2853	2872	2892	
	2927	2949	2973	2988	3009	3050	3076	3106	3285	3522				
BIT14 = 040000	293#	3389												
BIT15 = 100000	292#	1978	2547											
BIT2 = 000004	315#													
BIT3 = 000010	314#	1817												
BIT4 = 000020	313#	1942	1960											
BIT5 = 000040	312#													
BIT6 = 000100	311#	2068												
BIT7 = 000200	310#													
BIT8 = 000400	309#													
BIT9 = 001000	308#	2707	2710											
BPTVEC = 000014	324#													
BUFFER 027526	826	827	828	829	830	831	1338*	1354*	1369*	1384*	1398	1404	1445	
	1467*	1468*	1469	1488*	1489*	1490	1507*	1508	1573*	1574*	1575	1607*	1608*	
	1609	1640*	1641*	1642	1669*	1671	1687*	1688	1711*	1712*	1713	1799*	1796*	
	1982*	1983	1995*	1996*	1997	2000*	2022*	2023	2041*	2084*	2085	2086*	2103*	
	2104*	2105	2120*	2122	2123*	2138*	2140	2150*	2158*	2160	2161*	2176*	2178	
	2187*	2195*	2197	2198*	2214*	2215*	2216	2234*	2236	2243*	2255*	2272*	2273	
	2286*	2288	2297*	2305*	2307	2308*	2324*	2325	2334*	2343*	2344	2345*	2360*	
	2361	2370*	2379*	2380	2381*	2396*	2397	2406*	2415*	2416	2417*	2432*	2433	
	2442*	2451*	2452	2453*	2468*	2469	2478*	2487*	2488	2489*	2504*	2505	2514*	
	2523*	2524	2525*	2541*	2542	2543*	2545*	2562*	2579*	2595*	2596	2597*	2616*	
	2617*	2618	2628	2639*	2640*	2641	2649	2662*	2664*	2665	2674*	2676	2685*	
	2687	2688*	2705*	2706*	2708	2724*	2725*	2726	2736	2748*	2749*	2750	2759	
	2771*	2773*	2775	2783*	2785	2794*	2797	2798*	2816	2835	2838	2854	2873	
	2893	2926*	2928	2929*	2931*	2951*	2952*	2953	2974*	2975	2998	3012*	3013	
	3014*	3023	3126*	3127	3129*	3131*	3154*	3155	3157*	3159*	3186*	3187	3189*	
	3191*	3222*	3243*	3244	3246*	3248*	3272*	3287*	3288*	3289*	3290	4136*		
	835#													
CNTR 002056	850#	2289	2290	2310	2311	2326	2327	2347	2348	2362	2363	2383	2384	
CONS 002106	2398	2399	2419	2420	2434	2435	2455	2456	2470	2471	2491	2492	2506	
	2507	2527	2528	2539	2549	3133	3134*	3135	3141	3161	3162*	3163	3169	
	3193	3194*	3201	3205	3225	3250	3251*	3258	3262	3275				
DEUF 002034	826#	1339	1355	1370	1385	1403	1440	1444	1736*	1738	1751*	1753	1767*	
	1768	1769*	1771	1785*	1787	1801	1815*	1816	1818*	1819	1834*	1835	1848*	
	1849	1861*	1863	1876*	1878	1891*	1893	1907*	1909	1925*	1927	1941*	1943	
	1957*	1959	1977	2051*	2053	2067*	2069	2253*	2254	2559*	2561	2576*	2578	

RESVEC= 000010	321#													
RLO 002074	845#	1108	1110*	1111	1123	1127*	1128	1134*	1157	1159*	1161	1304*	3083*	
RSTVEC 023134	3109#													
RO =%000000	918	1729	2044	3069	3097	3229	3279	3355#						
	242#	906*	910	912*	916	938*	941*	956*	957*	958	973*	974	979*	
	1019#	1020	1022	1028*	1054*	1055	1057	1063*	1073*	1075	1082*	1091*	1093	
	1124#	1125	1127	1132*	1303*	1304	1311*	1398*	1399*	1400*	1401*	1402*	1485*	
	1486	1497*	1522*	1523	1529*	1571*	1574	1588*	1605*	1608	1622*	1638*	1641	
	1656#	1709*	1712	1724*	2018*	2019*	2020*	2026	2033	2042*	2614*	2617	2624*	
	2626	2637*	2640	2647*	2650	2721*	2725	2732*	2734	2745*	2749	2756*	2759	
	2816#	2817*	2818*	2835*	2836*	2837*	2854*	2855*	2856*	2873*	2874*	2875*	2893*	
	2894#	2895*	2896*	2897*	3184*	3199	3206	3223*	3241*	3256	3263	3273*	3338*	
	3341	3357*	3358*	3359*	3360*	3368*	3369*	3445	3453*	3457	3458	3460*	3461*	
	3462	3484*	3559	3560*	3561*	3568*	3569*	3570*	3571*	3572*	3573	3578	3583*	
	3585#	3589	3591	3690	3700*	3704	3720	3721	3734*	3752	3773*	3801	3805*	
	3812	3817*	3847	3848*	3853	3858	3861*	3918	3919*	3920	3921*	3922*	3923*	
	3924#													
R1 =%000001	243#	907*	908	909*	913*	914	915*	1703*	1704*	1705*	2650*	2651*	2652*	
	2653#	2654	2759*	2760*	2761*	2762*	2763	3055*	3056*	3057*	3078*	3079*	3080*	
	3180#	3181*	3182*	3196*	3197*	3237*	3238*	3239*	3253*	3254*	3356*	3361*	3446	
	3483#	3691	3704*	3705	3709	3733*	3753	3772*	3802	3808*	3809	3811*	3815	
	3816#													
R2 =%000002	244#	1440*	1441*	1442	1446*	1447	1455	3692	3703*	3707*	3710	3717*	3718*	
	3719	3724*	3732*	3754	3771*									
R3 =%000003	245#	3632	3641*	3647*	3648*	3651*	3656*	3657*	3658	3667*	3693	3701*	3702*	
	3716#	3719*	3728*	3729*	3731*	3755	3770*							
R4 =%000004	246#	3633	3635*	3636*	3637*	3638	3639*	3653	3655*	3663*	3666*	3756	3769*	
R5 =%000005	247#	3634	3640*	3642*	3644*	3645*	3646*	3647	3665*	3694	3696*	3698*	3705*	
	3709#	3724	3730*	3757	3768*									
R6 =%000006	248#	250	872*	873*	874	944	1720	3064	3091	3204	3261			
R7 =%000007	249#	251												
SETUP1 002400	906#													
SETUP2 002410	908#	911												
SETUP3 002434	914#	917												
SIZE 002054	834#	920*	921*	1442	1446									
SP =%000006	250#	876*	891*	892*	898	899	900	927*	2025	2031	3086	3111	3333*	
	3394#	3397	3399	3400	3429	3430	3434*	3445*	3446*	3453	3454*	3465	3466*	
	3467#	3477	3478*	3483	3484	3519	3542*	3545*	3559*	3564*	3585	3589*	3624*	
	3625	3626	3627*	3632*	3633*	3634*	3640	3665	3666	3667	3668*	3669*	3690*	
	3691#	3692*	3693*	3694*	3695*	3696	3699*	3712	3714*	3716	3726	3728	3730	
	3731	3732	3733	3734	3736*	3737*	3752*	3753*	3754*	3755*	3756*	3757*	3758	
	3764#	3768	3769	3770	3771	3772	3773	3778*	3801*	3802*	3803*	3804*	3805	
	3812#	3813	3814	3816	3817	3847*	3848	3858*	3860	3861	3862*	3864	3866	
	3868	3869*	3874	3876*	3878*	3886*	3890	3894	3895	3899	3918*	3919		
SREG0 002066	842#	1336	1341	1352	1357	1367	1372	1382	1387	1739	1740	1754	1755	
	1772	1773	1788	1789	1802	1803	1820	1821	1836	1837	1850	1851	1864	
	1865	1879	1880	1894	1895	1912	1913	1928	1929	1944	1945	1962	1963	
	1974	1979	1985	2002	2004	2016	2028	2032	3214	3216				
SREG1 002076	846#	2054	2055	2070	2071	2082	2089	2106	2107	2125	2126	2141	2142	
	2163	2164	2179	2180	2200	2201	2219	2221	2954	2955	2961	2972	2978	
	2991	2993*	2994	2999	3015*	3017	3018	3024	3036	3038*	3040	3053	3058*	
	3060	3065	3081	3087	3092	3105	3113							
STACK = 001100	231#													
STKLMT= 177774	236#													
STKPT 002116	854#	1140	1143*	1144	1225	1227*	1229	1239	1241*	1242	1255	1257*	1258*	
	1259	1265*	1267	1283*	1299*	1320*	1337*	1353*	1368*	1383*	1397*	1439*	1464*	

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

TST104	013040	2167	2173#	
TST105	013154	2186	2193#	
TST106	013254	2204	2211#	
TST107	013356	2223	2231#	
TST11	003422	1070#		
TST110	013464	2250#		
TST111	013562	2261	2268#	
TST112	013642	2277	2283#	
TST113	013756	2296	2303#	
TST114	014056	2314	2320#	
TST115	014172	2333	2340#	
TST116	014260	2349	2356#	
TST117	014374	2369	2376#	
TST12	003516	1088#		
TST120	014464	2385	2392#	
TST121	014600	2405	2412#	
TST122	014670	2421	2428#	
TST123	015004	2441	2448#	
TST124	015074	2457	2464#	
TST125	015210	2477	2484#	
TST126	015300	2493	2500#	
TST127	015414	2513	2520#	
TST13	003614	1106#		
TST130	015504	2529	2537#	
TST131	015620	2551	2557#	
TST132	015712	2567	2574#	
TST133	016004	2584	2591#	
TST134	016114	2611#		
TST135	016232	2625	2634#	
TST136	016356	2648	2660#	
TST137	016474	2675	2682#	
TST14	003674	1121#		
TST140	016612	2696	2703#	
TST141	016674	2711	2718#	
TST142	017020	2733	2742#	
TST143	017152	2757	2769#	
TST144	017274	2784	2791#	
TST145	017420	2806	2813#	
TST146	017516	2825	2832#	
TST147	017614	2844	2851#	
TST15	003764	1139#		
TST150	017712	2863	2870#	
TST151	020010	2882	2890#	
TST152	020170	2906	2916	2923#
TST153	020320	2943	2948#	
TST154	020430	2962	2970#	
TST155	020520	2980	2986#	
TST156	020634	3000	3008#	
TST157	020774	3022	3025	3033#
TST16	004054	1155#		
TST160	021050	3041	3048#	
TST161	021206	3074#		
TST162	021366	3103#		
TST163	021466	3115	3121#	
TST164	021652	3143	3149#	
TST165	022036	3171	3177#	

TST166	022364	3234#			
TST167	022656	3284#			
TST17	004120	1162	1168#		
TST170	022776	3306#			
TST18	002622	952#			
TST20	004164	1175	1181#		
TST21	004230	1188	1194#		
TST22	004316	1204	1210#		
TST23	004362	1217	1223#		
TST24	004434	1231	1237#		
TST25	004522	1253#			
TST26	004646	1269	1279#		
TST27	004736	1289	1296#		
TST3	002710	969#			
TST30	005046	1310	1317#		
TST31	005120	1325	1335#		
TST32	005206	1344	1351#		
TST33	005274	1360	1366#		
TST34	005362	1375	1381#		
TST35	005450	1390	1395#		
TST36	005672	1409	1416	1424	1430
TST37	006022	1453	1456	1463#	1436#
TST4	003002	987#			
TST40	006132	1482#			
TST41	006250	1503#			
TST42	006330	1512	1518#		
TST43	006426	1536#			
TST44	006504	1544	1550#		
TST45	006550	1557	1563#		
TST46	006766	1597#			
TST47	007204	1631#			
TST5	003052	994	1000#		
TST50	007412	1665#			
TST51	007500	1675	1681#		
TST52	007602	1692	1698#		
TST53	010016	1734#			
TST54	010104	1743	1749#		
TST55	010172	1758	1765#		
TST56	010274	1776	1783#		
TST57	010360	1792	1797#		
TST6	003140	1016#			
TST60	010446	1806	1813#		
TST61	010550	1824	1831#		
TST62	010630	1839	1845#		
TST63	010710	1853	1859#		
TST64	010776	1868	1874#		
TST65	011064	1883	1889#		
TST66	011152	1898	1905#		
TST67	011252	1916	1923#		
TST7	003236	1035#			
TST70	011336	1932	1939#		
TST71	011424	1948	1955#		
TST72	011524	1966	1973#		
TST73	011630	1986	1992#		
TST74	011744	2006	2012#		
TST75	012142	2049#			

TST76	012226	2058	2065#											
TST77	012314	2074	2081#											
TYPOS =	104404	3334	3939#											
TYPE =	104400	924	3332	3335	3517	3525	3558	3575	3577	3580	3582	3586	3593	3659
		2735	3776	3935#										
TYPOC =	104401	3566	3590	3936#										
TYPON =	104403	3938#												
TYPOS =	104402	3937#												
XDOFF	002100	847#	1037	1039*	1040	1053	1057*	1058	1065*	1093*	1170	1172*	1174	1198*
XPOS	002070	843#	2237	2238	2251	2258	2270	2276	2814	2823	2833	2842	2891	2902
		2925	2937											
YDOFF	002102	848#	1002	1004*	1005	1018	1022*	1023	1030*	1075*	1183	1185*	1187	1199*
YPOS	002072	844#	2852	2861	2871	2880	2910	2914	2941	2942				
ZDOFF	002122	856#	1072	1076	1090	1094	1196	1201						
ZPOS	002120	855#												
\$APTHD	001000	383	389#											
\$ASTAT=	***** U	3471	3497											
\$ATYC	023504	3442	3444#											
\$ATY1	023460	3440#												
\$ATY3	023466	3441#	3854											
\$ATY4	023476	3443#	3530											
\$BASE	001254	499#	907	937										
\$BDADR	001122	422#	937*	939	940*	954*	971*	988*	1002*	1018*	1037*	1053*	1072*	1090*
		1108*	1123*	1140*	1157*	1170*	1183*	1196*	1212*	1225*	1233*	1255*	1291*	1300*
		1319*	1336*	1352*	1367*	1382*	1396*	1438*	1470*	1491*	1509*	1520*	1537*	1552*
		1576*	1610*	1643*	1672*	1684*	1714*	1739*	1754*	1772*	1788*	1802*	1820*	1836*
		1850*	1864*	1879*	1894*	1912*	1928*	1944*	1962*	1974*	2002*	2016*	2054*	2070*
		2082*	2106*	2125*	2141*	2163*	2179*	2200*	2219*	2237*	2251*	2270*	2289*	2310*
		2326*	2347*	2362*	2383*	2398*	2419*	2434*	2455*	2470*	2491*	2506*	2527*	2539*
		2564*	2581*	2599*	2619*	2642*	2667*	2690*	2709*	2727*	2751*	2778*	2800*	2814*
		2833*	2852*	2871*	2891*	2910*	2925*	2941*	2954*	2972*	2991*	3017*	3036*	3053*
		3081*	3105*	3133*	3161*	3193*	3216*	3250*	3293*	4129				
\$BDDAT	001126	424#	874	959*	960	975*	976	992*	993	1005*	1006*	1007	1023*	1024*
		1025	1040*	1041*	1042	1058*	1059*	1060	1076*	1077*	1078	1094*	1095*	1096
		1111*	1112	1128*	1129	1144*	1145*	1146	1161*	1174*	1187*	1201*	1202*	1203
		1216*	1229*	1230*	1242*	1243*	1244	1259*	1260*	1261	1267*	1268	1285*	1286
		1305*	1306	1324*	1341*	1342*	1343	1357*	1358*	1359	1372*	1373*	1374	1387*
		1388*	1389	1405*	1406	1412*	1413	1420*	1421	1428*	1429	1449*	1450	1471*
		1472*	1473	1492*	1493*	1494	1511*	1524*	1525*	1526	1541*	1542*	1543	1556*
		1577*	1579	1583*	1584	1586*	1611*	1613	1617*	1618	1620*	1645*	1647	1651*
		1652	1654*	1674*	1690*	1691	1716*	1721*	1740*	1741*	1742	1755*	1756*	1757
		1773*	1774*	1775	1789*	1790*	1791	1803*	1804*	1805	1821*	1822*	1823	1837*
		1838	1851*	1852	1865*	1866*	1867	1880*	1881*	1882	1895*	1896*	1897	1913*
		1914*	1915	1929*	1930*	1931	1945*	1946*	1947	1963*	1964*	1965	1979*	1985*
		2004*	2005	2028*	2032*	2037	2055*	2056*	2057	2071*	2072*	2073	2089*	2090*
		2091	2107*	2108*	2109	2126*	2127*	2128	2142*	2143*	2144	2164*	2165*	2166
		2180*	2181*	2182	2201*	2202*	2203	2221*	2222	2238*	2239*	2240	2258*	2259*
		2260	2276*	2290*	2291*	2292	2311*	2312*	2313	2327*	2328*	2329	2350*	2363*
		2364*	2365	2386*	2399*	2400*	2401	2422*	2435*	2436*	2437	2458*	2471*	2472*
		2473	2494*	2507*	2508*	2509	2530*	2549*	2550	2565*	2566	2582*	2583	2600*
		2601	2620*	2621	2643*	2644	2670*	2671	2691*	2692	2712*	2728*	2729	2752*
		2753	2779*	2780	2801*	2802	2823*	2824	2842*	2843	2861*	2862	2880*	2881
		2902*	2903	2914*	2915	2937*	2942*	2957*	2963*	2978*	2979	2996*	3002*	3020*
		3027*	3042*	3060*	3061*	3067*	3087*	3088*	3092*	3093*	3094	3113*	3114	3135*
		3136	3141*	3142	3163*	3164	3169*	3170	3201*	3205*	3210	3218*	3225*	3226
		3258*	3262*	3267	3275*	3276	3294*	3295*	3296	4129				

1078	1081*	1092*	1096	1099*	1100*	1109*	1110	1112	1115*	1125*	1126*	1129
1142*	1143	1146	1149*	1158*	1171*	1184*	1197*	1203	1213*	1226*	1240*	1241
1244	1247*	1256*	1261	1264*	1268	1282*	1284	1286	1290*	1302*	1306	1309*
1321*	1340*	1343	1356*	1359	1371*	1374	1386*	1389	1404*	1406	1411*	1413
1419*	1421	1427*	1429	1445*	1448*	1450	1455	1466*	1468	1473	1476*	1486*
1487*	1489	1494	1506*	1521*	1526	1530*	1538*	1543	1553*	1568*	1569*	1570
1572*	1579	1581*	1584	1590*	1591	1602*	1603*	1604	1606*	1613	1615*	1618
1624*	1625	1635*	1636*	1637	1639*	1647	1649*	1652	1658*	1659	1670*	1689*
1691	1706*	1707*	1708	1710*	1726*	1727	1737*	1742	1752*	1757	1770*	1775
1786*	1791	1800*	1805	1817*	1823	1833*	1834	1847*	1848	1862*	1867	1877*
1882	1892*	1897	1911*	1915	1926*	1931	1942*	1947	1960*	1965	1978*	1984*
1998*	2005	2017*	2037	2052*	2057	2068*	2073	2088*	2091	2102*	2104	2109
2112*	2121*	2128	2139*	2144	2148*	2159*	2166	2177*	2182	2185*	2196*	2203
2217*	2222	2235*	2240	2244*	2257*	2260	2274*	2287*	2292	2295*	2306*	2313
2323*	2329	2332*	2342*	2359*	2365	2368*	2378*	2395*	2401	2404*	2414*	2431*
2437	2440*	2450*	2467*	2473	2476*	2486*	2503*	2509	2512*	2522*	2547*	2550
2560*	2562	2566	2577*	2579	2583	2593*	2597	2601	2605*	2621	2626*	2627*
2628*	2644	2649*	2654*	2663*	2671	2676*	2686*	2688	2692	2695*	2697*	2707*
2729	2734*	2735*	2736*	2753	2758*	2763*	2772*	2780	2785*	2795*	2798	2802
2805*	2807*	2820*	2824	2839*	2843	2858*	2862	2877*	2881	2899*	2903	2912*
2915	2936*	2950*	2960*	2976*	2979	2992*	2993	3001*	3011*	3015	3026*	3037*
3054*	3082*	3094	3108*	3114	3125*	3136	3140*	3142	3153*	3164	3168*	3170
3183*	3210	3217*	3221*	3226	3240*	3267	3271*	3276	3286*	3296	3299*	3300
4129												
3336*												
227												
390*												
414*	3422*	3423	3425*	3436								
3750	3781*											
418*	3521*	3529	3548	3561								
447*	3548	3908										
3481*	3488*											
415*	889*	3413*	3429*	3434	3436							
416*	890*	934*	953*	970*	1001*	1017*	1036*	1052*	1071*	1099*	1107*	1122*
1141*	1238*	1298*	1465*	1484*	1519*	1567*	1601*	1634*	1702*	2101*	2137*	2175*
2232*	2285*	2322*	2358*	2394*	2430*	2466*	2502*	2612*	2635*	2666*	2683*	2720*
2744*	2776*	2793*	3413	3430*	3436	3542						
920	2593	3815*	3819*									
483*												
487*												
490*												
493*												
391	395	456*	901	2014	3428	3527	3849					
477*												
485*												
488*												
491*												
391*												
3441*	3447	3482*	3487*									
463*	3457*	3460										
464*	3462*											
457*	3455	3463*	3475	3479*								
478*												
486*												
489*												
492*												

\$GET42 023070
 \$HD = 000000
 \$HIBTS 001000
 \$ICNT 001104
 \$ILLUP 025064
 \$ITEMB 001114
 \$LF 001176
 \$LFLG 023723
 \$LPADR 001106
 \$LPERR 001110

 \$LSTAD 025232
 \$MADR1 001232
 \$MADR2 001236
 \$MADR3 001242
 \$MADR4 001246
 \$MAIL 001200
 \$MAMS1 001230
 \$MAMS2 001234
 \$MAMS3 001240
 \$MAMS4 001244
 \$MBADR 001002
 \$MFLG 023722
 \$MSGAD 001214
 \$MSGLG 001216
 \$MSGTY 001200
 \$MTYP1 001231
 \$MTYP2 001235
 \$MTYP3 001241
 \$MTYP4 001245

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 CROSS REFERENCE TABLE -- MACRO NAMES

.SCHTA	217#	402
.SEOP	217#	3310
.SERRO	217#	3495
.SERRT	217#	3549
.SPARM	217#	
.SPOWE	217#	3745
.SREAD	217#	
.SSAVE	217#	
.SSCOP	217#	3373
.SSIZE	217#	3792
.SSPAC	217#	
.SSWDO	217#	
.STRAP	217#	3909
.STYPD	217#	3676
.STYPE	217#	3821
.STYPO	217#	3598

ADD	909	915	940	1309	1411	1419	1427	1448	2244	2628	2697	2736	2807	3359	3454
	3466	3478	3572	3627	3637	3709	3862								
ASL	1290	2627	2652	2674	2695	2735	2761	2783	2805	3299	3569	3570	3571	3922	
ASLB	3714														
ASR	963	979	980	1010	1028	1045	1063	1115	1132	1476	1497	2624	2647	2732	2756
	3461														
BCC	1291	1477	2245	2606	2696	2806	3715								
BCS	982	1029	1064	1133	1498	2625	2648	2675	2733	2757	2784				
BEQ	903	961	977	994	1008	1026	1043	1061	1079	1097	1113	1130	1147	1162	1175
	1188	1204	1217	1231	1245	1262	1269	1287	1307	1310	1325	1344	1360	1375	1390
	1407	1414	1422	1430	1451	1456	1474	1495	1512	1527	1544	1557	1585	1619	1653
	1675	1692	1743	1758	1776	1792	1806	1824	1839	1853	1868	1883	1898	1916	1932
	1948	1966	2006	2027	2038	2058	2074	2092	2110	2129	2145	2146	2167	2183	2204
	2223	2241	2261	2277	2293	2314	2330	2366	2402	2438	2474	2510	2551	2567	2584
	2602	2622	2645	2672	2693	2730	2754	2781	2803	2825	2844	2863	2882	2904	2916
	2938	2943	2980	3025	3041	3095	3115	3137	3143	3165	3171	3200	3211	3227	3257
	3268	3277	3297	3308	3339	3404	3406	3408	3412	3421	3448	3452	3472	3474	3513
	3516	3541	3544	3574	3579	3592	3654	3852	3865						
BGE	3424	3900													
BGT	3328	3661	3723												
BHI	3410														
BIC	1006	1021	1024	1041	1056	1059	1077	1095	1126	1145	1202	1230	1243	1260	1342
	1358	1373	1388	1472	1487	1493	1525	1542	1572	1583	1606	1617	1639	1651	1710
	1741	1756	1774	1790	1804	1822	1866	1881	1896	1914	1930	1946	1964	2056	2072
	2090	2108	2127	2143	2165	2181	2202	2239	2259	2291	2312	2328	2334	2364	2370
	2400	2406	2436	2442	2472	2478	2508	2514	2653	2762	3061	3088	3093	3295	3325
	3651														
BIS	957	981	1468	1489	1569	1574	1581	1586	1603	1608	1615	1620	1636	1641	1649
	1654	1707	1712	2104	2617	2640	2723	2725	2747	2749	2774	2796	3128	3134	3156
	3162	3188	3194	3245	3251	3656	3657	3717	3718						
BISB	3561														
BIT	1838	1852	2384	2420	2456	2492	2528	2710	3018	3024	3040	3065	3214	3307	3389
	3403	3411	3418	3515	3522	3540									
BITB	902	3451	3851	3856	3888										
BLT	3662	3706	3722	3879											
BMI	1578	1612	1646	1722	1980	2149	2186	2296	2333	2349	2369	2405	2441	2477	2513
	2956	2995	3713												
BNE	875	911	917	923	942	964	1011	1046	1116	1443	1580	1592	1614	1626	1648
	1660	1728	2034	2385	2421	2457	2493	2529	2711	3019	3066	3198	3207	3215	3255
	3264	3301	3362	3370	3390	3419	3450	3456	3459	3476	3523	3528	3538	3562	3584
	3652	3711	3767	3850	3857	3859	3867	3875	3889	3896					
BPL	1083	1101	1150	1248	1531	1589	1623	1657	1717	1725	1986	2113	2962	3000	3535
	3650	3697	3727	3844	3893										
BR	895	943	1289	1312	1409	1416	1424	1453	1458	1719	2030	2036	2040	2043	2151
	2188	2298	2335	2371	2407	2443	2479	2515	2615	2629	2638	2655	2677	2698	2722
	2737	2746	2764	2786	2808	2906	3022	3063	3090	3203	3209	3213	3220	3224	3260
	3266	3270	3274	3291	3392	3398	3401	3414	3417	3442	3464	3533	3567	3594	3628
	3643	3664	3708	3725	3761	3782	3810	3846	3872	3882	3891	3898			
CLR	873	886	887	901	947	991	1030	1065	1081	1099	1134	1158	1171	1184	1197
	1213	1226	1301	1321	1506	1540	1553	1670	1715	1786	1926	1984	2024	2042	2052
	2088	2235	2257	2274	2350	2386	2422	2458	2494	2530	2663	2706	2712	2772	2936
	2957	2960	2996	3001	3020	3026	3037	3059	3067	3084	3195	3218	3223	3252	3273
	3288	3289	3322	3323	3360	3416	3431	3560	3641	3700	3703	3765			
CLRB	929	3415	3480	3481	3482	3729	3897								
CMP	874	898	910	916	944	960	976	993	1007	1025	1042	1060	1078	1096	1112
	1129	1146	1203	1244	1261	1268	1286	1306	1343	1359	1374	1389	1406	1413	1421

	1429	1442	1447	1450	1455	1473	1494	1526	1543	1579	1584	1591	1613	1618	1625
	1647	1652	1659	1691	1720	1727	1742	1757	1775	1791	1805	1823	1867	1882	1897
	1915	1931	1947	1965	2005	2025	2031	2037	2057	2073	2091	2109	2128	2144	2166
	2182	2203	2222	2240	2260	2292	2313	2329	2365	2401	2437	2473	2509	2550	2566
	2583	2601	2621	2644	2671	2692	2729	2753	2780	2802	2824	2843	2862	2881	2903
	2915	2979	3064	3086	3091	3094	3111	3114	3136	3142	3164	3170	3204	3210	3226
	3261	3267	3276	3296	3300	3399	3423	3537	3721						
CMPB	3405	3409	3449	3527	3849	3864	3866	3874	3895	3899					
DEC	941	1247	1656	1658	1724	1726	2112	3197	3254	3326	3361	3369	3568		
DEC8	3649	3660	3878	3881											
EMT	232														
HALT	356	3309	3536	3539	3760	3781	3845								
INC	1311	1410	1418	1426	1457	1910	1961	1999	2001	2087	2124	2162	2199	2218	2243
	2256	2309	2346	2382	2418	2454	2490	2526	2544	2546	2563	2580	2598	2668	2689
	2777	2799	2821	2840	2859	2878	2900	2909	2911	2930	2932	2959	3016	3130	3132
	3158	3160	3190	3192	3247	3249	3292	3324	3422	3479	3518	3655	3663	3707	3766
INCB	3427	3512	3901												
IOT	233														
JMP	361	400	3346												
JSR	918	919	1729	2044	2822	2841	2860	2879	2901	2913	2933	3069	3097	3229	3279
	3341	3468	3524	3530	3854	3870	3873	3880	3887						
MOV	872	876	877	878	879	880	881	882	883	884	885	889	890	891	892
	893	896	897	899	900	904	906	907	908	912	913	914	920	926	927
	934	935	936	937	938	946	953	954	955	956	958	959	970	971	972
	973	974	975	988	989	990	992	1001	1002	1003	1004	1005	1017	1018	1019
	1020	1022	1023	1036	1037	1038	1039	1040	1052	1053	1054	1055	1057	1058	1071
	1072	1073	1074	1075	1076	1089	1090	1091	1092	1093	1094	1107	1108	1109	1110
	1111	1122	1123	1124	1125	1127	1128	1140	1141	1142	1143	1144	1156	1157	1159
	1161	1169	1170	1172	1174	1182	1183	1185	1187	1195	1196	1198	1199	1201	1211
	1212	1214	1216	1224	1225	1227	1229	1238	1239	1240	1241	1242	1254	1255	1256
	1257	1258	1259	1264	1265	1267	1280	1281	1282	1283	1284	1285	1297	1298	1299
	1300	1302	1303	1304	1305	1318	1319	1320	1322	1324	1336	1337	1338	1339	1340
	1341	1352	1353	1354	1355	1356	1357	1367	1368	1369	1370	1371	1372	1382	1383
	1384	1385	1386	1387	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1412
	1420	1428	1437	1438	1439	1440	1441	1444	1445	1446	1449	1464	1465	1466	1467
	1469	1470	1471	1483	1484	1485	1486	1488	1490	1491	1492	1504	1505	1507	1508
	1509	1511	1519	1520	1521	1522	1523	1524	1537	1538	1539	1541	1551	1552	1554
	1556	1564	1565	1566	1567	1568	1570	1571	1573	1575	1576	1577	1598	1599	1600
	1601	1602	1604	1605	1607	1609	1610	1611	1632	1633	1634	1635	1637	1638	1640
	1642	1643	1645	1666	1667	1668	1669	1671	1672	1674	1682	1683	1684	1685	1686
	1687	1688	1689	1690	1699	1700	1701	1702	1703	1704	1705	1706	1708	1709	1711
	1713	1714	1716	1721	1735	1736	1737	1738	1739	1740	1750	1751	1752	1753	1754
	1755	1766	1767	1768	1769	1770	1771	1772	1773	1784	1785	1787	1788	1789	1798
	1799	1800	1801	1802	1803	1814	1815	1816	1817	1818	1819	1820	1821	1832	1833
	1834	1835	1836	1837	1846	1847	1848	1849	1850	1851	1860	1861	1862	1863	1864
	1865	1875	1876	1877	1878	1879	1880	1890	1891	1892	1893	1894	1895	1906	1907
	1908	1909	1911	1912	1913	1924	1925	1927	1928	1929	1940	1941	1942	1943	1944
	1945	1956	1957	1958	1959	1960	1962	1963	1974	1975	1976	1977	1978	1979	1982
	1983	1985	1993	1994	1995	1996	1997	1998	2000	2002	2004	2013	2014	2015	2016
	2017	2018	2019	2020	2021	2022	2023	2028	2032	2041	2050	2051	2053	2054	2055
	2066	2067	2068	2069	2070	2071	2082	2083	2084	2085	2086	2089	2100	2101	2102
	2103	2105	2106	2107	2119	2120	2121	2122	2123	2125	2126	2136	2137	2138	2139
	2140	2141	2142	2157	2158	2159	2160	2161	2163	2164	2174	2175	2176	2177	2178
	2179	2180	2194	2195	2196	2197	2198	2200	2201	2212	2213	2214	2215	2216	2217
	2219	2221	2232	2233	2234	2236	2237	2238	2251	2252	2253	2254	2255	2258	2269
	2270	2271	2272	2273	2276	2284	2285	2286	2287	2288	2289	2290	2304	2305	2306

	2307	2308	2310	2311	2321	2322	2323	2324	2325	2326	2327	2341	2342	2343	2344
	2345	2347	2357	2358	2359	2360	2361	2362	2363	2377	2378	2379	2380	2381	2383
	2393	2394	2395	2396	2397	2398	2399	2413	2414	2415	2416	2417	2419	2429	2430
	2431	2432	2433	2434	2435	2449	2450	2451	2452	2453	2455	2465	2466	2467	2468
	2469	2470	2471	2485	2486	2487	2488	2489	2491	2501	2502	2503	2504	2505	2506
	2507	2521	2522	2523	2524	2525	2527	2538	2539	2540	2541	2542	2543	2545	2547
	2549	2558	2559	2560	2561	2562	2564	2565	2575	2576	2577	2578	2579	2581	2582
	2592	2593	2594	2595	2596	2597	2599	2600	2612	2613	2614	2616	2618	2619	2620
	2626	2635	2636	2637	2639	2641	2642	2643	2649	2650	2661	2662	2664	2665	2666
	2667	2670	2676	2683	2684	2685	2686	2687	2688	2690	2691	2704	2705	2707	2708
	2709	2719	2720	2721	2724	2726	2727	2728	2734	2743	2744	2745	2748	2750	2751
	2752	2758	2759	2770	2771	2773	2775	2776	2778	2779	2785	2792	2793	2794	2795
	2797	2798	2800	2801	2814	2815	2816	2817	2818	2819	2820	2823	2833	2834	2835
	2836	2837	2838	2839	2842	2852	2853	2854	2855	2856	2857	2858	2861	2871	2872
	2873	2874	2875	2876	2877	2880	2891	2892	2893	2894	2895	2896	2897	2898	2899
	2902	2910	2912	2914	2924	2925	2926	2927	2928	2929	2931	2937	2941	2942	2949
	2950	2951	2952	2953	2954	2963	2971	2972	2973	2974	2975	2976	2978	2987	2988
	2989	2991	2992	2993	2998	3002	3009	3010	3011	3012	3013	3014	3015	3017	3023
	3027	3034	3036	3038	3042	3049	3050	3052	3053	3054	3055	3056	3057	3058	3060
	3075	3076	3077	3078	3079	3080	3081	3082	3083	3085	3087	3092	3104	3105	3106
	3107	3108	3109	3110	3113	3122	3123	3124	3125	3126	3127	3129	3131	3133	3135
	3140	3141	3150	3151	3152	3153	3154	3155	3157	3159	3161	3163	3168	3169	3178
	3179	3180	3181	3182	3183	3184	3185	3186	3187	3189	3191	3193	3196	3201	3205
	3216	3217	3221	3222	3225	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244
	3246	3248	3250	3253	3258	3262	3271	3272	3275	3285	3286	3287	3290	3293	3294
	3329	3333	3338	3355	3356	3357	3358	3368	3394	3395	3397	3400	3413	3425	3426
	3429	3430	3433	3434	3445	3446	3453	3457	3462	3463	3465	3467	3477	3483	3484
	3514	3519	3542	3545	3559	3564	3573	3578	3583	3585	3589	3624	3632	3633	3634
	3640	3647	3665	3666	3667	3668	3669	3690	3691	3692	3693	3694	3695	3696	3701
	3704	3724	3730	3731	3732	3733	3734	3736	3737	3750	3751	3752	3753	3754	3755
	3756	3757	3758	3759	3764	3768	3769	3770	3771	3772	3773	3774	3775	3778	3801
	3802	3803	3804	3805	3806	3807	3808	3812	3813	3814	3815	3816	3817	3847	3848
	3853	3861	3869	3876	3918	3919	3923	3511	3521	3529	3625	3626	3629	3630	3631
MOV B	888	3428	3432	3440	3441	3443	3511	3521	3529	3625	3626	3629	3630	3631	3635
	3638	3639	3658	3699	3702	3716	3719	3728	3858	3886	3894	3921			
NEG	2651	2760	3636	3698											
NOP	1644	2669	3342	3343	3344										
RESET	928	1160	1173	1186	1200	1215	1228	1266	1323	1510	1555	1673	2003	2220	2275
	2548	2935	2977	2990	3035	3039	3051	3098	3112	3139	3167	3340			
ROL	3642	3644	3645	3646	3648										
RTI	3435	3547	3670	3738	3780	3863									
RTS	3363	3371	3485	3587	3818	3903	3924								
SUB	921	1082	1100	1149	1529	1530	1588	1590	1622	1624	2148	2150	2185	2187	2295
	2297	2332	2368	2404	2440	2476	2512	2605	2654	2763	3460	3520	3705	3811	
TRAP	3926	3936	3937	3938	3939										
TST	894	922	939	2026	2033	2348	2955	2961	3199	3206	3256	3263	3396	3420	3455
	3473	3475	3534	3543	3591	3653	3710	3720	3809	3860	3868	3890	3920		
TSTB	2994	2999	3407	3447	3458	3471	3712	3726	3843	3892					
.ASCII	445	446													
.ASCIZ	444	447	3347	3595	3784	3942	3953	3956	3961	3964	3968	3972	3977	3982	3987
	3992	3997	4002	4005	4008	4011	4014	4018	4022	4025	4029	4035	4038	4041	4044
	4047	4050	4055	4058	4061	4064	4070	4076	4082	4087	4093	4100	4103	4107	4110
	4114	4118	4121												
.BLKW	3743														
.BYTE	412	413	418	419	434	435	436	437	466	467	477	478	485	486	488
	489	491	492	494	495	496	497	3350	3487	3488	3489	3531	3532	3671	3672

1464	1490	1481	1482	1483	1501	1502	1503	1504	1513	1516	1517	1518	1519	1534
1535	1536	1537	1545	1548	1549	1550	1551	1558	1561	1562	1563	1564	1595	1596
1597	1598	1629	1630	1631	1632	1663	1664	1665	1666	1676	1679	1680	1681	1682
1693	1696	1697	1698	1699	1732	1733	1734	1735	1744	1747	1748	1749	1750	1759
17E3	1764	1765	1766	1777	1781	1782	1783	1784	1793	1795	1796	1797	1799	1807
1811	1812	1813	1814	1825	1829	1830	1831	1832	1840	1843	1844	1845	1846	1854
1857	1858	1859	1860	1869	1872	1873	1874	1875	1894	1887	1888	1889	1890	1899
1903	1904	1905	1906	1917	1921	1922	1923	1924	1933	1937	1938	1939	1940	1949
1953	1954	1955	1956	1967	1971	1972	1973	1974	1980	1987	1990	1991	1992	1993
2007	2010	2011	2012	2013	2047	2048	2049	2050	2059	2063	2064	2065	2066	2075
2079	2080	2081	2082	2093	2097	2098	2099	2100	2116	2117	2118	2119	2130	2133
2134	2135	2136	2147	2150	2154	2155	2156	2157	2168	2171	2172	2173	2174	2187
2191	2192	2193	2194	2205	2209	2210	2211	2212	2224	2229	2230	2231	2232	2248
2249	2250	2251	2262	2266	2267	2268	2269	2278	2281	2282	2283	2284	2297	2301
2302	2303	2304	2315	2318	2319	2320	2321	2334	2338	2339	2340	2341	2350	2354
2355	2356	2357	2370	2374	2375	2376	2377	2386	2390	2391	2392	2393	2406	2410
411	412	413	422	426	427	428	429	442	446	447	448	449	458	462
463	464	465	478	482	483	484	485	494	498	499	500	501	514	518
519	520	521	530	535	536	537	538	552	555	556	557	558	568	572
573	574	575	585	589	590	591	592	602	609	610	611	612	626	632
633	634	635	649	656	659	660	661	676	680	681	682	683	697	701
702	703	704	712	716	717	718	719	734	740	741	742	743	758	767
768	769	770	785	789	790	791	792	807	811	812	813	814	826	830
831	832	833	845	849	850	851	852	864	868	869	870	871	883	888
889	890	891	907	917	921	922	923	924	938	944	946	947	948	949
963	968	969	970	971	981	984	985	986	987	3001	3006	3007	3008	3009
3023	3026	3031	3032	3033	3034	3042	3046	3047	3048	3049	3072	3073	3074	3075
3101	3102	3103	3104	3116	3119	3120	3121	3122	3144	3147	3148	3149	3150	3172
3175	3176	3177	3178	3232	3233	3234	3235	3282	3283	3284	3285	3304	3305	3306
3307	3311	3318	3322	3327	3330	3347	3353	3355	3366	3368	3374	3402	3405	3406
3409	3436	3437	3438	3496	3501	3515	3547	3548	3550	3568	3568	3597	3677	3746
3776	3793	3799	3807	3822	3910	3919	4133	4136						
3417	3525	3803	3807	3816	3820	3816								
3415	3524	3800	3803	3812	3816	342	344	347	348	356	448	452	877	879
217	222	227	340	341	342								885	895
886	887	889	890	3316	3322	3323	3334	3347	3351	3380	3381	3382	3383	3384
3385	3416	3417	3433	3436	3437	3502	3503	3504	3505	3506	3537	3548	3565	3590
3908	3935	3936	3937	3938	3939									
871	930	949	966	984	997	1013	1032	1048	1067	1085	1103	1118	1136	1152
1165	1178	1191	1207	1220	1234	1250	1276	1293	1314	1332	1348	1363	1378	1392
1433	1460	1479	1500	1515	1533	1547	1560	1594	1628	1662	1678	1695	1731	1746
1762	1780	1794	1810	1828	1842	1856	1871	1886	1902	1920	1936	1952	1970	1989
2009	2046	2062	2078	2096	2115	2132	2153	2170	2190	2208	2228	2247	2265	2280
2300	2317	2337	2353	2373	2389	2409	2425	2445	2461	2481	2497	2517	2534	2554
2571	2588	2608	2631	2657	2679	2700	2715	2739	2766	2788	2810	2829	2848	2867
2887	2920	2945	2967	2983	3005	3030	3045	3071	3100	3118	3146	3174	3231	3281
3303	3445	3446	3467	3483	3484	3511	3690	3730	3752	3768				
217	332	347	356	438	440	441	442	449	452	871	930	934	949	953
966	970	984	988	997	1001	1013	1017	1032	1036	1048	1052	1067	1071	1085
1089	1103	1107	1118	1122	1136	1140	1152	1156	1165	1169	1178	1182	1191	1195
1207	1211	1220	1224	1234	1238	1250	1254	1276	1280	1293	1297	1314	1318	1332
1336	1348	1352	1363	1367	1378	1382	1392	1396	1433	1437	1460	1464	1479	1483
1500	1504	1515	1519	1533	1537	1547	1551	1560	1564	1594	1598	1628	1632	1662
1666	1678	1682	1695	1699	1731	1735	1746	1750	1762	1766	1780	1784	1794	1798
1810	1814	1828	1832	1842	1846	1856	1860	1871	1875	1886	1890	1902	1906	1920
1924	1936	1940	1952	1956	1970	1974	1989	1993	2009	2013	2046	2050	2062	2066

.IFT
.IFTF
.IIF
.IRP
.LIST

	2078	2092	2096	2100	2115	2119	2132	2136	2153	2157	2170	2174	2190	2194	2208
	2208	2228	2232	2247	2251	2265	2269	2280	2284	2300	2304	2317	2321	2337	2341
	2353	2357	2373	2377	2389	2393	2409	2413	2425	2429	2445	2449	2461	2465	2481
	2531	2535	2551	2557	2571	2575	2588	2554	2558	2571	2575	2588	2592	2608	2612
	2631	2635	2657	2661	2679	2683	2700	2704	2715	2719	2739	2743	2766	2770	2788
	2810	2814	2829	2829	2833	2848	2852	2867	2871	2887	2891	2920	2924	2945	2949
	2971	2983	2987	2987	3005	3009	3030	3034	3045	3049	3071	3075	3100	3104	3118
	3146	3150	3174	3174	3178	3231	3235	3281	3285	3303	3307	3322	3384	3537	3926
	3936	3937	3938	3938	3939	3940									
.MACR		839													
.MACRO		402	838	3926											
.MCALL		332	449												
.NLIST		332	347	356	438	440	441	442	449	452	871	930	934	949	953
	966	970	984	988	997	1001	1013	1017	1032	1036	1048	1052	1067	1071	1085
	1089	1103	1107	1118	1122	1136	1140	1152	1156	1165	1169	1178	1182	1191	1195
	1207	1211	1220	1224	1234	1238	1250	1254	1276	1280	1293	1297	1314	1318	1332
	1336	1348	1352	1363	1367	1378	1382	1392	1396	1433	1437	1460	1464	1479	1483
	1500	1504	1515	1519	1533	1537	1547	1551	1560	1564	1594	1598	1628	1632	1662
	1666	1678	1682	1695	1699	1731	1735	1746	1750	1762	1766	1780	1784	1794	1798
	1810	1814	1828	1832	1842	1846	1856	1860	1871	1875	1886	1890	1902	1906	1920
	1924	1936	1940	1952	1956	1970	1974	1989	1993	2009	2013	2046	2050	2062	2066
	2078	2082	2096	2100	2115	2119	2132	2136	2153	2157	2170	2174	2190	2194	2208
	2212	2228	2232	2247	2251	2265	2269	2280	2284	2300	2304	2317	2321	2337	2341
	2353	2357	2373	2377	2389	2393	2409	2413	2425	2429	2445	2449	2461	2465	2481
	2485	2497	2501	2517	2521	2534	2538	2554	2558	2571	2575	2588	2592	2608	2612
	2631	2635	2657	2661	2679	2683	2700	2704	2715	2719	2739	2743	2766	2770	2788
	2792	2810	2814	2829	2833	2848	2852	2867	2871	2887	2891	2920	2924	2945	2949
	2967	2971	2983	2987	3005	3009	3030	3034	3045	3049	3071	3075	3100	3104	3118
	3122	3146	3150	3174	3178	3231	3235	3281	3285	3303	3307	3322	3384	3537	3926
	3936	3937	3938	3938	3939	3940									
.PAGE	335	402	448												
.REM															
.REPT															
.SBTTL	356	440	350	360	364	376	404	451	524	930	949	966	984	997	1013
	228	336	1067	1085	1103	1118	1136	1152	1165	1178	1191	1207	1220	1234	1250
	1032	1048	1274	1276	1293	1314	1328	1329	1330	1332	1348	1363	1378	1392	1433
	1272	1273	1479	1515	1533	1547	1560	1594	1628	1662	1678	1695	1731	1746	1762
	1460	1479	1500	1828	1842	1856	1871	1886	1902	1920	1936	1952	1970	1989	2009
	1780	1794	1810	2096	2115	2132	2153	2170	2190	2208	2228	2247	2265	2280	2300
	2046	2062	2078	2373	2389	2409	2425	2445	2461	2481	2497	2517	2534	2554	2571
	2317	2337	2353	2657	2679	2700	2715	2739	2766	2788	2810	2829	2848	2867	2887
	2588	2608	2631	2983	3005	3030	3045	3071	3100	3118	3146	3174	3231	3281	3303
	2920	2945	2967	3497	3551	3600	3678	3747	3794	3823	3911	3927	3941		
	3312	3375	3439												
.TITLE	217	357	358	370	390	391	392	393	394	395	411	414	415	416	417
.WORD	356	421	422	423	424	425	426	427	428	429	438	440	441	457	458
	420	460	461	462	463	464	468	469	470	483	487	490	493	499	500
	459	502	503	504	505	506	507	508	509	510	511	512	513	514	515
	501	517	518	3327	3330	3469	3576	3581	3675	3777	3779	3819	3855	3902	
	516														

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

E09

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1 MACY11 27(732) 20-SEP-76 09:57 PAGE 111
DZVSA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

*.DZVSA/SOL/CRF/PAGNUM=DZVSA
RUN-TIME: 75 57 11 SECONDS
RUN-TIME RATIO: 486/144=3.3
CORE USED: 24K (47 PAGES)

