

KD11-Z

11/44 CPU/EIS
CKKAABO

AH-F620B-MC
FICHE 1 OF 2

AUG 1981
COPYRIGHT © 79-81
MADE IN USA



The main body of the document is a large grid of approximately 15 columns and 20 rows. Each cell in the grid contains a small, faint version of the header information, including the document ID 'KD11-Z', the title '11/44 CPU/EIS CKKAABO', the model 'AH-F620B-MC', and the page number 'FICHE 1 OF 2'. The text is very light and difficult to read against the dark background.

KD11-Z

11/44 CPU/EIS
CKKAABO

AH-F620B-MC
FICHE 2 OF 2

AUG 1981
COPYRIGHT © 79-81
MADE IN USA



.REM %

I D E N T I F I C A T I O N

PRODUCT CODE: AC-F618B-MC
 PRODUCT NAME: CKKABO 11/44 CPU/EIS
 DATE CREATED: APRIL, 1981
 MAINTAINER: DIAGNOSTIC GROUP
 AUTHORS: CHUCK ROBINSON, DAN MILLEVILLE

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

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

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

COPYRIGHT (C) 1979, 1981
 DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42

43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59

* HISTORY SECTION *

REVISION	RELEASE DATE	COMMENTS
-----	-----	-----
CKKAAA	OCTOBER, 1979	INITIAL RELEASE
CKKAAB	APRIL, 1981	ADDITION OF TEST 340 TO CHECK CPU POWER MONITOR BIT AND USE OF NEW SYSMAC TO CHECK FOR RANDOM ^Q BEFORE ^S IN INPUT ROUTINE

60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88

*
* SUMMARY OF OPERATING INSTRUCTIONS *
*

THE FOLLOWING PROCEDURE CAN BE USED TO RUN THIS DIAGNOSTIC IN A STANDARD CONFIGURATION WITH AT LEAST 8K OF MEMORY AND A TELETYPE. IF THE PROGRAM DOES NOT RUN SUCCESSFULLY CONSULT THE FOLLOWING DOCUMENT FOR ASSISTANCE.

OPERATING PROCEDURES:

1. LOAD THE PROGRAM USING NORMAL PROCEDURES
2. START THE PROGRAM AT LOCATION 200
3. PROGRAM SHOULD PRINT 'END OF PASS' WITHIN THE 1ST SECOND AND REPEATABLY THEREAFTER AT APPROX. 5 SEC. INTERVALS WITH CACHE ON. (APPROX. 10 SEC INTERVALS WITH CACHE OFF)
4. IF THE PROGRAM DOES NOT RUN AS DESCRIBED ABOVE, CONSULT THE FULL OPERATING INSTRUCTIONS WHICH FOLLOW.

90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144

1.0 GENERAL PROGRAM INFORMATION

1.1 PROGRAM PURPOSE

THIS DIAGNOSTIC PROGRAM IS DESIGNED TO BE A COMPREHENSIVE CHECK OF THE PDP-11/44 BASIC INSTRUCTION SET. THE PROGRAM EXERCISES ALL OF THE PROCESSOR LOGIC AND MICROCODE FOR ALL INSTRUCTIONS EXCEPT THE TRAP AND MEMORY MANAGEMENT INSTRUCTIONS. THE PROGRAM DOES NOT TEST INSTRUCTIONS OR HARDWARE RELATED TO THE TRAP OR INTERRUPT MECHANISMS OF THE 11/44 (E.G. RTT, RT1, WAIT, RESET, TRAP, EMT).

NOTE: HOWEVER, IF THE OPTIONAL CPU/CIS TESTS ARE RUN THEN THE BREAKPOINT TRAP INSTRUCTION AND THE ABILITY TO TRAP UNDER ILLEGAL INSTRUCTION (TRAP TO 10) WILL BE ASSUMED TO BE FUNCTIONAL. (SEE SECT. 2.3)

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE

PDP-11/44 PROCESSOR
16K MEMORY -- THE PROGRAM USES LOCATIONS 0 - 36000

1.2.2 SOFTWARE

THIS PROGRAM IS WRITTEN TO BE RUN AS A STAND-ALONE PROGRAM. HOWEVER, THE PROGRAM IS DESIGNED TO RUN UNDER AUTOMATED PRODUCT TEST SYSTEM (APT) IN ALL THREE MODES.

THE PROGRAM CAN ALSO BE RUN UNDER THE ACT 11 MONITOR

1.3 RELATED DOCUMENTS AND STANDARDS

PDP-11/44 MICROCODE LISTING

PDP-11/44 ELECTRICAL SCHEMATICS

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

NONE

146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196

1.5 FAILURE ASSUMPTIONS

NONE

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

2.1.1 LOADING

USE NORMAL PROCEDURES FOR LOADING ABSOLUTE BINARY TAPES.

2.1.2 NORMAL START

THIS IS THE PROCEDURE FOR NORMAL PROGRAM RUNNING (I.E.,
STARTING WITH TEST 1 AND EXECUTING ENTIRE DIAGNOSTIC).

START AT ADDRESS = 200

2.1.3 SUBTEST START

THIS IS THE PROCEDURE FOR STARTING AT A SUBTEST OTHER THAN 1.

1. LOAD \$TESTN (IN MAILBOX SECTION) WITH THE NUMBER OF SUBTEST
MINUS ONE (IN OCTAL). FOR EXAMPLE, TO START AT SUBTEST 100,
\$TESTN=77.

2. LOAD STARTING ADDRESS OF SUBTEST IN LOC. 216

3. START AT ADDRESS = 204

2.2 SPECIAL ENVIRONMENTS

THIS PROGRAM IS WRITTEN TO COMPLY WITH ALL THE REQUIREMENTS
OF THE APT INTERFACE SPECIFICATION. IT WILL RUN UNDER APT
IN EITHER QUICK VERIFY, PROGRAM OR RUN-TIME MODES.

THIS PROGRAM IS WRITTEN TO COMPLY WITH ALL OF THE REQUIREMENTS
OF PROGRAMS TO RUN UNDER THE ACT11 MONITOR.

198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239

2.2.1 RUNNING UNDER APT

THE EXECUTION TIMES PROVIDED IN THE APT SCRIPT THAT FOLLOWS ARE FOR EXECUTION WITH A 11/44 PROCESSOR, CACHE, 16K CORE MEMORY, AND 300 BAUD.

THE FOLLOWING IS A PROGRAM LOAD FILE USED BY APT:

1. E TABLE 'A' IS USED FOR APT DUMP MODE.
 - A. IN ADDITION TO NORMAL CPU DIAGNOSTIC TESTS THIS TABLE WILL SELECT THE OPTIONAL CACHE AND CIS TESTS. (\$SWREG=400)
2. E TABLE 'B' IS USED FOR APT QV MODE WHILE RUNNING ON A MANUFACTURING QV STATION. IT ACCOMPLISHES WHAT ETABLE 'A' DOES BUT ADDITIONALLY SUPPRESSES TYPEOUTS. (\$ENVM=240)
3. ETABLE 'C' IS USED FOR APT QV OR RUNTIME MODES WHILE RUNNING ON SYSTEMS OTHER THAN MFG. QV STATIONS. THIS TABLE DESELECTS THE OPTIONAL CACHE AND CIS TESTS.

	1ST PASS RUN TIME 10	LONGEST TEST TIME 10	ADDITIONAL RUN TIME 0	
.....		E TABLES	
		A	B	C
E-MODE/S-MODE (\$ENVM/\$ENV)		200/000	240/001	240/001
SWITCH REGISTER 1 (\$SWREG)		000400	000400	000000
SWITCH REGISTER 2 CPU TYPE/OPTIONS		000000 00/0000	000000 00/0000	000000 00/0000

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
291
292
293
294
295
296
297

2.3 PROGRAM OPTIONS

THIS PROGRAM IS INTENDED TO BE A BASIC PROCESSOR TEST. IT IS INTENDED TO BE THE LOWEST LEVEL DIAGNOSTIC RUN. HOWEVER, IT DOES PROVIDE FOR OPTIONAL CACHE AND CIS TESTS. THESE TESTS CAN BE SELECTED AND RUN IN MANUFACTURING ON THE MFG. QV STATION. THE TESTS ARE SPECIALLY DESIGNED TO EXERCISE THOSE SIGNALS ON THE CPU DATA PATH/CONTROL MODULES WHICH RELATE TO THE CACHE AND CIS. IT IS ASSUMED THAT CACHE AND CIS MODULES ARE KNOWN GOOD. RUNNING THESE TESTS ELIMINATE HAVING TO RUN THE CACHE AND CIS DIAGNOSTICS IN THE CPU APT QUICK VERIFY SCRIPT.

THE OPTIONAL TESTS ARE SELECTED THROUGH APT SCRIPTING OR BY LOADING BIT08 OF HARDWARE SWITCH REGISTER(177570). SEE SECTION 2.2.1.

2.4 EXECUTION TIMES

THE DIAGNOSTIC COMPLETES THE FIRST PASS IN LESS THAN 1 SEC. SUBSEQUENT PRINTING OF END OF PASS MESSAGE REQUIRE APPROXIMATELY. 5 TO 10 SECS. INTERVALS
THE PROGRAM WILL RUN CONTINUOUSLY UNTIL EXTERNALLY HALTED.

3.0 ERROR INFORMATION

3.1 ERROR TYPES

THERE ARE TWO BASIC TYPES OF ERRORS IN THE DIAGNOSTIC.

3.1.1 FUNCTIONAL ERRORS

THESE ARE ERRORS WHICH REPRESENT A MALFUNCTION OF AN INSTRUCTION OR SEQUENCE OF INSTRUCTION. (E.G., THE PROPER CONDITION CODE NOT SET OR IMPROPER RESULT OF AN ARITHMETIC OR LOGICAL OPERATION).

3.1.2 SEQUENCE ERRORS

THE RESULT OF A TEST BEING EXECUTED OUT OF SEQUENCE. (E.G. WILD MACHINE OR IMPROPER BRANCH OR JUMP).

3.2 ERROR REPORTING PROCEDURES

THE DIAGNOSTIC RESPONDS TO THE DETECTION OF ALL ERRORS BY STORING CERTAIN INFORMATION IN MEMORY AND HALTING THE PROCESSOR. THE INFORMATION STORED IN MEMORY CAN BE USED BY THE OPERATOR

298
299
300
301
302
303

TO IDENTIFY THE ERROR DETECTED.

CERTAIN FAILURES WILL CAUSE THE PROCESSOR TO HANG.
THIS TYPE OF FAILURE IS INDICATED IF THE PROGRAM
DOES NOT PRINT ITS END OF PASS INDICATION WITHIN A REASONABLE
AMOUNT OF TIME. (FIRST MESSAGE SHOULD APPEAR WITHIN 1 SEC.)

305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347

3.3 ERROR DESCRIPTOR INFORMATION

THE DIAGNOSTIC MAILBOX HOLDS THE ERROR INFORMATION NECESSARY TO IDENTIFY THE DETECTED ERROR. THIS INFORMATION HAS BEEN DESIGNED FOR COMPLIANCE WITH THE APT DIAGNOSTIC INTERFACE SPECIFICATION. IT IS THE PRIMARY MEDIUM FOR IDENTIFYING ERRORS.

3.3.1 \$MSGTYP

THIS LOCATION IS INCREMENTED FROM ZERO TO ONE BEFORE THE PROGRAM COMES TO A PROGRAMMED HALT. IF THIS LOCATION IS NOT ONE, THEN THE DIAGNOSTIC HAS COME TO AN UNPROGRAMMED HALT. CHECK THE STACK AND PC FOR A CLUE TO THE CAUSE. SUSPECT A TRAP.

3.3.2 \$FATAL

THIS LOCATION IS LOADED WITH A NUMBER BEFORE A HALT IS EXECUTED. EACH PROGRAMMED HALT HAS A UNIQUE NUMBER ASSOCIATED WITH IT WHICH CAN BE USED TO IDENTIFY THE ERROR WHICH HAS BEEN DETECTED.

3.3.3 \$PASS

THIS LOCATION IS INCREMENTED FOR EVERY COMPLETE PASS OF THE DIAGNOSTIC. MONITORING THE LOCATION WILL INDICATE WHETHER OR NOT THE PROGRAM IS HUNG. IT WILL ALSO INDICATE THE NUMBER OF SUCCESSFUL PASSES COMPLETED BEFORE THE ERROR HALT. A HIGH PASS COUNT MIGHT INDICATE THAT THE ERROR HALT IS ASSOCIATED WITH AN INTERMITTANT FAULT.

3.3.4 \$TESTN

THIS LOCATION IS INCREMENTED IN EACH NEW SUBTEST. THIS SHOULD INDICATE THE TEST BEING EXECUTED WHEN THE ERROR WAS DETECTED. THIS LOCATION IS ALSO USED TO DETECT A SEQUENCE ERROR.

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

3.4 ERROR IDENTIFICATION

BECAUSE OF THE OVERHEAD ASSOCIATED WITH EACH HALT IN AN APT COMPATIBLE PROGRAM THE SEQUENCE CHECK CODE WILL SHARE THE ERROR HALT OF FUNCTIONAL ERROR WITHIN EACH SUBTEST. TO DETERMINE WHICH ERROR IS BEING REPORTED, LOCATIONS \$FATAL AND \$TESTN ARE USED TOGETHER. WHEN AN ERROR HALT OCCURS, CHECK \$FATAL TO DETERMINE THE NUMBER OF THE ERROR DETECTED. NOW, CHECK THAT THE TEST NUMBER WHERE THIS ERROR IS DETECTED CORRESPONDS TO THE VALUE IN \$TESTN. IF THESE AGREE THE ERROR WAS A FUNCTIONAL ERROR AS DESCRIBED IN THE LISTINGS. IF THESE NUMBERS DO NOT AGREE, THEN A SEQUENCE ERROR WAS DETECTED. IN THIS CASE \$TESTN WILL CONTAIN ONE MORE THAN THE NUMBER OF THE LAST TEST SUCCESSFULLY COMPLETED. SEQUENCE ERRORS WHICH SHARE THE ERROR HALTS OF FUNCTIONAL ERRORS WILL ALWAYS BE REPORTED BY THE LAST HALT IN SUBTEST IN WHICH THEY WERE DISCOVERED.

4.0 PROGRESS REPORT

THE MESSAGE CKKAABO 11/44 CPU/EIS IS PRINTED ON THE CONSOLE TELETYPE AFTER THE FIRST PASS, AND FOLLOWING EVERY SUBSEQUENT 400 PASSES.

379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423

5.0 TROUBLE SHOOTING

WHEN THE PROGRAM DISCOVERS A FAULT IT WILL HALT. TO DETERMINE THE CAUSE OF THE HALT, THE DIAGNOSTIC PROVIDES ERROR INFORMATION. THIS INFORMATION IS STORED IN THE APT MAILBOX AND IS THE PRIMARY SOURCE OF ERROR IDENTIFICATION.

UPON FINDING AN ERROR, THE FOLLOWING PROCEDURE SHOULD AID IN ISOLATING THE FAULT.

5.1 CHECK THE MAILBOX

1. \$MSGTY THIS LOCATION SHOULD CONTAIN A 1. IF THE PROCESSOR HALTS AND THIS LOCATION IS ZERO, THEN THE PROCESSOR HAS COME TO AN UNEXPECTED HALT. FIRST SUSPECT A TRAP. CHECK THE PC AND IF A TRAP CHECK R6 AND THE STACK FOR THE LOCATION OF THE FAILING INSTRUCTION.
2. \$FATAL THIS LOCATION IS USED TO HOLD THE NUMBER OF THE ERROR WHICH DETECTED. EACH ERROR BEING CHECKED BY THE DIAGNOSTIC IS ASSIGNED A UNIQUE NUMBER WHICH IS STORED IN \$FATAL WHEN THAT ERROR IS DETECTED.

WHEN AN ERROR IS DETECTED, CHECK THE LISTING TO SEE THAT THE ERROR NUMBER STORED IN \$FATAL IS ONE WHICH IS DETECTED IN THE TEST WHOSE NUMBER IS IN \$TESTN. IF THERE IS A DISAGREEMENT THEN THE ERROR BEING REPORTED IS A SEQUENCE ERROR. \$TESTN CONTAINS ONE MORE THAN THE LAST TEST WHICH WAS SUCCESSFULLY COMPLETED.
3. \$TESTN THIS LOCATION IS USED TO INDICATE THE NUMBER OF THE TEST WHICH WAS BEING EXECUTED WHEN THE FAULT WAS DETECTED. \$TESTN IS USED IN CONJUNCTION WITH \$FATAL TO DISTINGUISH BETWEEN SEQUENCE AND FUNCTIONAL ERRORS. (SEE ? THIS SECTION)
4. \$PASS THIS LOCATION IS USED TO INDICATE THE NUMBER OF SUCCESSFUL PASSES WHICH THE DIAGNOSTIC HAS COMPLETED. THIS WILL GIVE AN INDICATION THAT THE DIAGNOSTIC HAS NOT JUST BEEN HUNG IN A LOOP

IF AN ERROR HAS BEEN DETECTED \$PASS WILL SHOW WHETHER IT WAS A HARD ERROR DISCOVERED DURING THE FIRST TRY OR WHETHER IT WAS INTERMITTANT OR DEVELOPED DURING THE RUNNING OF THE DIAGNOSTIC.

425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
579
580
581

5.2 SCOPING

WHILE THIS DIAGNOSTIC IS PRIMARILY INTENDED TO BE A FAULT DETECTION PROGRAM, PROVISIONS ARE MADE TO ASSIST A TECHNICIAN WHO MIGHT WANT TO USE THE PROGRAM AS A TROUBLE SHOOTING TEST.

THE PROCEDURE FOR SCOPING A SUBTEST INVOLVES MODIFYING SEVERAL MEMORY LOCATIONS IN THE TEST ITSELF. THE PHILOSOPHY IS TO PROVIDE A SCOPING LOOP WHICH WILL INCLUDE THE CODE WHERE THE ERROR WAS DETECTED. THE LOOP IS SET UP SO THAT THE LOOP WILL NOT BE TERMINATED SHOULD THE ERROR INTERMITTANTLY DISAPPEAR.

THE PROCEDURE IS AS FOLLOWS:

1. DETERMINE WHICH ERROR IS TO BE SCOPED. USE \$FATAL AND \$TESTN FOR THIS (SEE ABOVE)
2. LOCATE THE ERROR ROUTINE IN THE LISTING.
3. CLEAR THE RIGHT BYTE OF THE CONDITIONAL BRANCH INSTRUCTION ASSOCIATED WITH THE ERROR. (THIS IS MARKED WITH <--- 'S IN THE LISTING.)
4. REPLACE THE INSTRUCTION FOLLOWING <MOV #XXX, -(R2)> WITH THE SCOPING BRANCH PROVIDED IN THE LISTING COMMENTS.
5. RESTART THE PROGRAM. THE PROGRAM MAY BE RESTARTED FROM THE BEGINNING OR FROM THE SUBTEST (SEE 2.0).

6.0 LISTING

-----%
.TITLE CKKAABO 11/44 CPU/EIS
.ENABLE ABS
STBOT=1000
.NLIST CND,MC,MD
.LIST ME
SCOPE=NOP
R7=%7
R6=%6
PS=177776
TPS=177564
TPB=177566
USRM=140000
PUSRM=30000
PIRQ=177772
BIT4=20
MFPT=7
.MCALL \$APTHDR,\$APT S,\$ACT11
.MCALL \$CATCH,1170
.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100 ;:FIRST ADDRESS OF THE STACK

000000
001000
000240
000007
000006
177776
177564
177566
140000
030000
177772
000020
000007
001100


```

001100 KERSTK= STACK          ;;KERNEL STACK
000700 SUPSTK= STACK-200   ;;SUPERVISOR STACK
000600 USESTK= STACK-300     ;;USER STACK
104000 ERROR=EMT
000004 SCOPE=IOT
177776 PS= 177776       ;;PROCESSOR STATUS WORD
177776 PSW=PS
177774 STKLMT= 177774    ;;STACK LIMIT REGISTER
177772 PIRQ= 177772      ;;PROGRAM INTERRUPT REQUEST REGISTER
177570 DSWR= 177570     ;;HARDWARE SWITCH REGISTER
177570 DDISP= 177570    ;;HARDWARE DISPLAY REGISTER
177546 LKS= 177546      ;;LINE CLOCK (KW11-L) STATUS REGISTER
;*MISCELLANEOUS DEFINITIONS
000011 HT= 11           ;;CODE FOR HORIZONTAL TAB
000012 LF= 12           ;;CODE LINE FEED
000015 CR= 15           ;;CODE CARRIAGE RETURN
000200 CRLF= 200          ;;CODE FOR CARRIAGE RETURN-LINE FEED
;*GENERAL PURPOSE REGISTER DEFINITIONS
000000 R0= %0             ;;GENERAL REGISTER
000001 R1= %1             ;;GENERAL REGISTER
000002 R2= %2             ;;GENERAL REGISTER
000003 R3= %3             ;;GENERAL REGISTER
000004 R4= %4             ;;GENERAL REGISTER
000005 R5= %5             ;;GENERAL REGISTER
000006 R6= %6             ;;GENERAL REGISTER
000007 R7= %7             ;;GENERAL REGISTER
000000 R10=R0
000001 R11=R1
000002 R12=R2
000003 R13=R3
000004 R14=R4
000005 R15=R5
000006 SP= %6           ;;STACK POINTER
000006 KSP=SP
000006 SSP=SP
000006 USP=SP
000007 PC= %7           ;;PROGRAM COUNTER
;*PRIORITY LEVEL DEFINITIONS
000000 PR0= 0            ;;PRIORITY LEVEL 0
000040 PR1= 40          ;;PRIORITY LEVEL 1
000100 PR2= 100         ;;PRIORITY LEVEL 2
000140 PR3= 140         ;;PRIORITY LEVEL 3
000200 PR4= 200         ;;PRIORITY LEVEL 4
000240 PR5= 240         ;;PRIORITY LEVEL 5
000300 PR6= 300         ;;PRIORITY LEVEL 6
000340 PR7= 340         ;;PRIORITY LEVEL 7
;*SWITCH REGISTER SWITCH DEFINITIONS
100000 SW15= 100000
040000 SW14= 40000
020000 SW13= 20000
010000 SW12= 10000
004000 SW11= 4000
002000 SW10= 2000
001000 SW09= 1000
000400 SW08= 400
000200 SW07= 200
000100 SW06= 100
    
```

```

000040 SW05= 40
000020 SW04= 20
000010 SW03= 10
000004 SW02= 4
000002 SW01= 2
000001 SW00= 1
001000 SW9=SW09
000400 SW8=SW08
000200 SW7=SW07
000100 SW6=SW06
000040 SW5=SW05
000020 SW4=SW04
000010 SW3=SW03
000004 SW2=SW02
000002 SW1=SW01
000001 SW0=SW00

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000 BIT15= 100000
040000 BIT14= 40000
020000 BIT13= 20000
010000 BIT12= 10000
004000 BIT11= 4000
002000 BIT10= 2000
001000 BIT09= 1000
000400 BIT08= 400
000200 BIT07= 200
000100 BIT06= 100
000040 BIT05= 40
000020 BIT04= 20
000010 BIT03= 10
000004 BIT02= 4
000002 BIT01= 2
000001 BIT00= 1
001000 BIT9=BIT09
000400 BIT8=BIT08
000200 BIT7=BIT07
000100 BIT6=BIT06
000040 BIT5=BIT05
000020 BIT4=BIT04
000010 BIT3=BIT03
000004 BIT2=BIT02
000002 BIT1=BIT01
000001 BIT0=BIT00

;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004 ERRVEC= 4 ;; TIME OUT AND OTHER ERRORS
000010 RESVEC= 10 ;; RESERVED AND ILLEGAL INSTRUCTIONS
000014 TBITVEC= 14 ;; "T" BIT
000014 TRTVEC= 14 ;; TRACE TRAP
000014 BPTVEC= 14 ;; BREAKPOINT TRAP (BPT)
000020 IOTVEC= 20 ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024 PWRVEC= 24 ;; POWER FAIL
000030 EMTVEC= 30 ;; EMULATOR TRAP (EMT) **ERROR**
000034 TRAPVEC= 34 ;; "TRAP" TRAP
000060 TKVEC= 60 ;; TTY KEYBOARD VECTOR
000064 TPVEC= 64 ;; TTY PRINTER VECTOR
000100 LKVEC= 100 ;; LINE CLOCK (KW11-L) VECTOR
000114 CACHVEC= 114 ;; CACHE ERROR INTERRUPT VECTOR

```



```

000240 PIRQVEC=240          ;;PROGRAM INTERRUPT REQUEST VECTOR
000250 MMVEC= 250         ;;MEMORY MANAGEMENT VECTOR
      .SBTTL CACHE REGISTER DEFINITIONS
177740 LOADRS = 177740  ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
177742 HIADRS = 177742  ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
177744 MEMERR = 177744  ;;CACHE ERROR REGISTER
177746 CONTRL = 177746  ;;MEMORY CONTROL REGISTER
177750 MAINT = 177750   ;;MEMORY MAINTENENCE REGISTER
177752 HITMIS = 177752  ;;HIT MISS REGISTER '1' IMPLIES HIT IN CACHE
      .SBTTL CPU REGISTER DEFINITIONS
177760 SIZELO = 177760  ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
      ;;TO GET TO THE LAST 32 WORDS OF MEMORY
17776  SIZEHI = 177762  ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
      ;;CURRENTLY ALL ZERO
177764 SYSTID = 177764  ;;SYSTEM ID REGISTER
177766 CPUERR = 177766  ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
      ;;THE TRAP TO ERRVEC (000004)
      .SBTTL MEMORY MANAGEMENT DEFINITIONS
      ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
177572 MMRO= 177572
177574 MMR1= 177574
177576 MMR2= 177576
172516 MMR3= 172516
      SR0=MMRO
177572 SR1=MMR1
177574 SR2=MMR2
177576 SR3=MMR3
172516
      ;*USER 'I' PAGE DESCRIPTOR REGISTERS
177600 UIPDR0= 177600
177602 UIPDR1= 177602
177604 UIPDR2= 177604
177606 UIPDR3= 177606
177610 UIPDR4= 177610
177612 UIPDR5= 177612
177614 UIPDR6= 177614
177616 UIPDR7= 177616
      ;*USER 'D' PAGE DESCRIPTOR REGISTORS
177620 UDPDR0= 177620
177622 UDPDR1= 177622
177624 UDPDR2= 177624
177626 UDPDR3= 177626
177630 UDPDR4= 177630
177632 UDPDR5= 177632
177634 UDPDR6= 177634
177636 UDPDR7= 177636
      ;*USER 'I' PAGE ADDRESS REGISTERS
177640 UIPAR0= 177640
177642 UIPAR1= 177642
177644 UIPAR2= 177644
177646 UIPAR3= 177646
177650 UIPAR4= 177650
177652 UIPAR5= 177652
177654 UIPAR6= 177654
177656 UIPAR7= 177656
      ;*USER 'D' PAGE ADDRESS REGISTERS
177660 UDPAR0= 177660
177662 UDPAR1= 177662
  
```

```
177664 UDPAR2= 177664
177666 UDPAR3= 177666
177670 UDPAR4= 177670
177672 UDPAR5= 177672
177674 UDPAR6= 177674
177676 UDPAR7= 177676
;*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS
172200 SIPDR0= 172200
172202 SIPDR1= 172202
172204 SIPDR2= 172204
172206 SIPDR3= 172206
172210 SIPDR4= 172210
172212 SIPDR5= 172212
172214 SIPDR6= 172214
172216 SIPDR7= 172216
;*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS
172220 SDPDR0= 172220
172222 SDPDR1= 172222
172224 SDPDR2= 172224
172226 SDPDR3= 172226
172230 SDPDR4= 172230
172232 SDPDR5= 172232
172234 SDPDR6= 172234
172236 SDPDR7= 172236
;*SUPERVISOR 'I' PAGE ADDRESS REGISTERS
172240 SIPAR0= 172240
172242 SIPAR1= 172242
172244 SIPAR2= 172244
172246 SIPAR3= 172246
172250 SIPAR4= 172250
172252 SIPAR5= 172252
172254 SIPAR6= 172254
172256 SIPAR7= 172256
;*SUPERVISOR 'D' PAGE ADDRESS REGISTERS
172260 SDPAR0= 172260
172262 SDPAR1= 172262
172264 SDPAR2= 172264
172266 SDPAR3= 172266
172270 SDPAR4= 172270
172272 SDPAR5= 172272
172274 SDPAR6= 172274
172276 SDPAR7= 172276
;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
172300 KIPDR0= 172300
172302 KIPDR1= 172302
172304 KIPDR2= 172304
172306 KIPDR3= 172306
172310 KIPDR4= 172310
172312 KIPDR5= 172312
172314 KIPDR6= 172314
172316 KIPDR7= 172316
;*KERNEL 'D' PAGE DESCRIPTOR REGISTERS
172320 KDPDR0= 172320
172322 KDPDR1= 172322
172324 KDPDR2= 172324
172326 KDPDR3= 172326
172330 KDPDR4= 172330
```



```
172332 KDPDR5= 172332
172334 KDPDR6= 172334
172336 KDPDR7= 172336
;*KERNEL 'I' PAGE ADDRESS REGISTERS
172340 KIPAR0= 172340
172342 KIPAR1= 172342
172344 KIPAR2= 172344
172346 KIPAR3= 172346
172350 KIPAR4= 172350
172352 KIPAR5= 172352
172354 KIPAR6= 172354
172356 KIPAR7= 172356
;*KERNEL 'D' PAGE ADDRESS REGISTERS
172360 KDPAR0= 172360
172362 KDPAR1= 172362
172364 KDPAR2= 172364
172366 KDPAR3= 172366
172370 KDPAR4= 172370
172372 KDPAR5= 172372
172374 KDPAR6= 172374
172376 KDPAR7= 172376
.SBTTL UNIBUS MAP REGISTER DEFINITIONS
;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHYX'
170200 MAPL00 = 170200
170202 MAPH00 = 170202
170204 MAPL01 = 170204
170206 MAPH01 = 170206
170210 MAPL02 = 170210
170212 MAPH02 = 170212
170214 MAPL03 = 170214
170216 MAPH03 = 170216
170220 MAPL04 = 170220
170222 MAPH04 = 170222
170224 MAPL05 = 170224
170226 MAPH05 = 170226
170230 MAPL06 = 170230
170232 MAPH06 = 170232
170234 MAPL07 = 170234
170236 MAPH07 = 170236
170240 MAPL10 = 170240
170242 MAPH10 = 170242
170244 MAPL11 = 170244
170246 MAPH11 = 170246
170250 MAPL12 = 170250
170252 MAPH12 = 170252
170254 MAPL13 = 170254
170256 MAPH13 = 170256
170260 MAPL14 = 170260
170262 MAPH14 = 170262
170264 MAPL15 = 170264
170266 MAPH15 = 170266
170270 MAPL16 = 170270
170272 MAPH16 = 170272
170274 MAPL17 = 170274
170276 MAPH17 = 170276
170300 MAPL20 = 170300
```

170302	MAPH20 = 170302
170304	MAPL21 = 170304
170306	MAPH21 = 170306
170310	MAPL22 = 170310
170312	MAPH22 = 170312
170314	MAPL23 = 170314
170316	MAPH23 = 170316
170320	MAPL24 = 170320
170320	MAPH24 = 170320
170324	MAPL25 = 170324
170326	MAPH25 = 170326
170330	MAPL26 = 170330
170332	MAPH26 = 170332
170334	MAPL27 = 170334
170336	MAPH27 = 170336
170340	MAPL30 = 170340
170342	MAPH30 = 170342
170344	MAPL31 = 170344
170346	MAPH31 = 170346
170350	MAPL32 = 170350
170352	MAPH32 = 170352
170354	MAPL33 = 170354
170356	MAPH33 = 170356
170360	MAPL34 = 170360
170362	MAPH34 = 170362
170364	MAPL35 = 170364
170366	MAPH35 = 170366
170370	MAPL36 = 170370
170372	MAPH36 = 170372
170374	MAPL37 = 170374
170376	MAPH37 = 170376
170200	MAPL0=MAPL00
170202	MAPH0=MAPH00
170204	MAPL1=MAPL01
170206	MAPH1=MAPH01
170210	MAPL2=MAPL02
170212	MAPH2=MAPH02
170214	MAPL3=MAPL03
170216	MAPH3=MAPH03
170220	MAPL4=MAPL04
170222	MAPH4=MAPH04
170224	MAPL5=MAPL05
170226	MAPH5=MAPH05
170230	MAPL6=MAPL06
170232	MAPH6=MAPH06
170234	MAPL7=MAPL07
170236	MAPH7=MAPH07

582

000000

.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

000174 000174
 000174 000000
 000176 000000

DISPREG: .WORD 0 ;:SOFTWARE DISPLAY REGISTER
 SWREG: .WORD 0 ;:SOFTWARE SWITCH REGISTER

583

.SBTTL ACT11 HOOKS

000046 000200
000052 000046
000052 042046
000052 000052
000052 000000
584 000200
585 000300

```

;HOOKS REQUIRED BY ACT11
$SVPC=. ;SAVE PC
.=46 ;:1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
$ENDAD
.=52 ;:2)SET LOC.52 TO ZERO
.WORD 0 ;: RESTORE PC
.=$SVPC
.=300

```

.SBTTL APT MAILBOX-ETABLE

000300
000300 000000
000302 000000
000304 000000
000306 000000
000310 000000
000312 000000
000314 000000
000316 000000
000320
000320 000
000321 000
000322 000000
000324 000000
000326 000000

```

.EVEN
$MAIL: ;:APT MAILBOX
$MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE
$FATAL: .WORD AFATAL ;:FATAL ERROR NUMBER
$TESTN: .WORD ATESTN ;:TEST NUMBER
$PASS: .WORD APASS ;:PASS COUNT
$DEVCT: .WORD ADEVCT ;:DEVICE COUNT
$UNIT: .WORD AUNIT ;:I/O UNIT NUMBER
$MSGAD: .WORD AMSGAD ;:MESSAGE ADDRESS
$MSGLG: .WORD AMSGLG ;:MESSAGE LENGTH
$ETABLE: ;:APT ENVIRONMENT TABLE
$ENV: .BYTE AENV ;:ENVIRONMENT BYTE
$ENVM: .BYTE AENVM ;:ENVIRONMENT MODE BITS
$SWREG: .WORD ASWREG ;:APT SWITCH REGISTER
$USWR: .WORD AUSWR ;:USER SWITCHES
$CPUOP: .WORD ACPUOP ;: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

```

000330
586

```

$ETEND:
.MEXIT
.SBTTL APT PARAMETER BLOCK
*****

```

000024 000330
000044 000024
000044 000200
000044 000044
000044 000330
000044 000330

```

;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

```

000330
000330 000000
000332 000300
000334 000010
000336 000010
000340 000000
000342 000014

```

;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PPD-1 DIAGNOSTIC
;INTERFACE SPEC.
$APTHD:
$HIBTS: .WORD 0 ;:TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL ;:ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT: .WORD 10 ;:RUN TIM OF LONGEST TEST
$PASTM: .WORD 10 ;: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 $ETEND-$MAIL/2 ;:LENGTH MAILBOX-ETABLE(WORDS)

```

587
588
589

```

*****
;SOME POINTERS TO CPU TRAP HANDLERS
*****

```


590		000004			. = 4
591	000004	042402			T04
592	000006	000000			0
593	000010	042416			T010
594	000012	000000			0
595	000014	042432			T014
596	000016	000000			0
597		000030			. = 30
598	000030	042446			T030
599	000032	000000			0
600	000034	042462			T034
601	000036	000000			0
602		000114			. = 114
603	000114	042476			T0114
604	000116	000000			0
605		000244			. = 244
606	000244	042512			T0244
607	000246	000000			0
608	000250	042526			T0250
609	000252	000000			0

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

	000370			
000370	000000	000000	000000	
000376	000000	000000	000000	
000404	000001	000001	177777	
	001100			
	001100			
	000200		000674	
000204	012706	001000		
000210	012702	000304		
000214	000137			
000216	000000			
	001100			
	000302			
	000304			
001100	012737	042266	000024	
001106	012737	000000	000306	
001114	012737	177777	042100	
001122	012706	001000		
001126	012702	000304		
001132	012737	000000	000304	
001140	012737	000000	000302	
001146	012737	000000	000300	

```
*****
:DATA TABLE FOR USE IN ADDRESSING MODE TESTS
*****
      -370
      0,0,0,0,0,0
      1,1,-1
      -1100
*****
:SET UP STARTING ADDRESS
      .SX-
      . = 200
      JMP      START
      MOV      #STBOT,R6          ;SET STACK POINTER
      MOV      #STESTN,R2        ;SET MAILBOX POINTER
      JMP      @ (PC)+           ;JUMP TO SUBTEST
      0                          ;ADDR. OF SUBTEST GOES HERE

      . = .SX
$ERROR-$FATAL
$STSTNM-$STESTN
START:  MOV      #PWRDN,@#24          ;SET UP FOR POWER FAIL
        MOV      #0,@#SPASS          ;CLEAR PASS COUNT
        MOV      #-1,@#PASSPT        ;SET PRINT COUNTER
RESTR:  MOV      #STBOT,R6          ;INITIALIZE STACK POINTER
        MOV      #STESTN,R2         ;SET UP POINTER TO MESSAGE TYPE
        MOV      #0,@#STSTNM        ;CLEAR TEST NUMBER
        MOV      #0,@#$ERROR        ;CLEAR ERROR NUMBER
        MOV      #0,@#MSGTY        ;CLEAR MESSAGE TYPE (FOR APT)
```

641

.SBITL TEST # 1 - CHECK BRANCHES ON Z BIT

:TEST 1 - CHECK BRANCHES ON Z BIT

001154 005212
001156 022712 000001
001162 001034
642 001164 000257
643 001166 001401
644 001170 000406

TST1: INC (R2) ;UPDATE TEST NUMBER
CMP #1,(R2) ;SEQUENCE ERROR?
BNE TST2-10 ;BR TO ERROR HALT ON SEQ ERROR
CCC ;CLEAR ALL CONDITION CODES
BEQ BR1 ;SHOULD BRANCH
BR BR2 ;BAD BRANCH OF Z-BIT
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; BRANCH INSTRUCTION AND <====
; REPLACE THE MOVE INSTRUCTION <====
; FOLLOWING W/ 774 <====

645
646
647
648
649 001172
001172 012762 000001 177776
001200 005262 177774
001204 000000
650 001206
001206 001006

BR1: MOV #1,-2(R2) ;MOVE TO MAILBOX # ***** 1 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;SHOULD HAVE BRANCHED: Z=0

651 001210 012762 000002 177776
001216 005262 177774
001222 000000
652 001224 000264
653 001226 001001
654 001230 000406

BR2: BNE BR3
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 765 <====

655
656
657
658 001232
001232 012762 000003 177776
001240 005262 177774
001244 000000
659 001246
001246 001406

BR3: MOV #2,-2(R2) ;MOVE TO MAILBOX # ***** 2 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT
SEZ
BNE BR4
BR BR5

660
661
662
663
664
665
666
667
668

BR4: MOV #3,-2(R2) ;MOVE TO MAILBOX # ***** 3 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;SHOULD NOT HAVE BRANCHED HERE ON Z 1
BR5: BEQ TST2
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 745 <====

001250 012762 000004 177776
001256 005262 177774
001262 000000

MOV #4,-2(R2) ;MOVE TO MAILBOX # ***** 4 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;SHOULD HAVE BRANCHED ON Z-1
; OR SEQUENCE ERROR

:SBITL DATA PATH TESTS

: THE DATA PATH TESTS ARE USED TO VERIFY THAT VARIOUS
: DATA PATTERNS CAN BE SUCCESSFULLY MOVED THROUGH THE DATA PATHS
: MOVE AND COMPARE MODE 2,3 INSTRUCTIONS ARE USED TO PASS AND
: TEST VARIOUS DATA PATTERNS IN THE DATA PATHS.
: THE TEST EXERCISES THE INTERNAL DATA PATHS, THE UNIBUS

669
670
671
672
673
674

:DATA TRANSCEIVERS, AND AMUX CONTROL FOR ALU AND USUS INPUTS.
: IF THESE TESTS FAIL, EXAMINE THE TARGET LOCATION (LOC. 0)
: TO SEE WHICH BITS OF THE DATA PATH ARE FAILING. IF THIS PROVIDES
: INCONCLUSIVE DATA, TRY TO CHECK MODE 3 IR DECODE BY RUNNING
: JUST THE MICROCODE AND IR DECODE TESTS FOR THE MOVE AND COMPARE
: INSTRUCTIONS.

675

.SBTTL TEST # 2 - TEST OF ZEROES IN THE DATA PATH
:*****
:TEST 2 - TEST OF ZEROES IN THE DATA PATH
:*****

001264 005212
001266 022712 000002
001272 001010
676 001274 012737 000000 000000
677
678 001302 005737 000000
679 001306 001406

TST2: INC (R2) ;UPDATE TEST NUMBER
CMP #2,(R2) ;SEQUENCE ERROR?
BNE TST3-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #0,#0 ;MOVE ZEROES THRU ADDRESS LINES, DATA
;LINES AND INTERNAL PATHS
TST @#0 ;SUCCESSFUL?
BEQ TST3

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
; CONDITIONAL BRANCH INST. AND <=====
; REPLACE THE MOVE INSTRUCTION <=====
; WHICH FOLLOWS W/ 771 <=====
;MOVE TO MAILBOX # ***** 5 *****
;SET MSGTYP TO FATAL ERROR
;DATA INCORRECT
; OR SEQUENCE ERROR

001310 012762 000005 177776
001316 005262 177774
001322 000000

MOV #5,-2(R2)
INC -4(R2)
HALT

680

681

.SBTTL TEST # 3 - TEST OF PATTERN 125252 IN DATA PATH
:*****
:TEST 3 - TEST OF PATTERN 125252 IN DATA PATH
:*****

001324 005212
001326 022712 000003
001332 001011
682 001334 012737 125252 000000
683
684 001342 022737 125252 000000
685 001350 001406

TST3: INC (R2)
CMP #3,(R2)
BNE TST4-10
MOV #125252,@#0
CMP #125252,@#0
BEQ TST4

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:MOVE ALTERNATING ONES AND ZEROES
:THRU DATA PATHS
:SUCCESSFUL

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <---
: WHICH FOLLOWS W/ 770 <---

001352 012762 000006 177776
001360 005262 177774
001364 000000

MOV #6,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 6 *****
:SET MSGTYP TO FATAL ERROR
:DATA INCORRECT
: OR SEQUENCE ERROR

686

687

.SBTTL TEST # 4 - TEST OF PATTERN 052525 IN DATA PATH
:*****
:TEST 4 - TEST OF PATTERN 052525 IN DATA PATH
:*****

001366 005212
001370 022712 000004
001374 001011
688 001376 012737 052525 000000
689
690 001404 022737 052525 000000
691 001412 001406

TST4: INC (R2) ;UPDATE TEST NUMBER
CMP #4,(R2) ;SEQUENCE ERROR?
BNE TST5-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #052525,@#0 ;MOVE ALTERNATING ZEROES AND ONES
;THRU DATA PATH
CMP #052525,@#0 ;SUCCESSFUL?
BEQ TST5

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION < ---
; WHICH FOLLOWS W/ 770 <=
; MOVE TO MAILBOX # ***** 7 *****
; SET MSGTYP TO FATAL ERROR
; DATA INCORRECT
; OR SEQUENCE ERROR

001414 012762 000007 177776
001422 005262 177774
001426 000000

MOV #7,-2(R2)
INC -4(R2)
HALT

692

693

.SBTTL TEST # 5 - TEST OF ALL ONES IN DATA PATH
:*****
:TEST 5 - TEST OF ALL ONES IN DATA PATH
:*****

001430 005212
001432 022712 000005
001436 001011
694 001440 012737 177777 000000
695 001446 022737 177777 000000
696 001454 001406

TST5: INC (R2) ;UPDATE TEST NUMBER
CMP #5,(R2) ;SEQUENCE ERROR?
BNE TST6-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #177777,@#0 ;MOVE ONES THRU DATA PATH
CMP #177777,@#0 ;SUCCESSFUL
BEQ TST6
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 770 <====
MOV #10,-2(R2) ;MOVE TO MAILBOX # ***** 10 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DATA INCORRECT
; OR SEQUENCE ERROR

697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712

:*****
:SBTTL B-REGISTER TEST
:
: THE B-REGISTER SHIFTING LOGIC TESTS ARE USED TO TEST THAT THE
:B-REGISTER CAN HOLD VARIOUS DATA PATTERNS AND THAT THE ASSOCIATED
:LOGIC SUPPORTS THE SHIFTING FUNCTIONS WITHIN THE B-REGISTER AND C-BIT.
:A ONE IS SHIFTED THROUGH EVERY BIT IN THE B-REGISTER AND C-BIT IN
:BOTH DIRECTIONS.
: THE B-REGISTER ITSELF IS TESTED IN ITS ABILITY AS A BUFFER AND AS
:A SHIFT REGISTER. DATA IS ALSO PASSED THROUGH THE DATA PATH AND ALU.
: IF THESE TESTS FAIL, EXAMINE THE TARGET LOCATION (LOC. 0) TO SEE
:WHICH BITS OF THE B-REGISTER MAY BE FAILING. IF THIS PROVIDES
:INCONCLUSIVE DATA TRY TO CHECK THE MODE 3 IR DECODE BY RUNNING JUST
:THE MICROCODE AND IR DECODE TESTS FOR THE PARTICULAR INSTRUCTIONS.
:

713

.SBTTL TEST # 6 - SHIFT BIT 0 TO BIT 1

.....
:TEST 6 - SHIFT BIT 0 TO BIT 1
:.....

001472 005212
001474 022712 000006
001500 001014
714 001502 000241
715 001504 012737 000001 000000
716 001512 006137 000000
717 001516 022737 000002 000000
718 001524 001406

TST6: INC (R2)
CMP #6,(R2)
BNE TST7-10
CLC
MOV #1,@#0
ROL @#0
CMP #2,@#0
BEQ TST7

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:CLEAR CARRY BIT
:LOAD A 1
:SHIFT LEFT
:SUCCESSFUL

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 765 <
:MOVE TO MAILBOX # ***** 11 *****
:SET MSGTYP TO FATAL ERROR
:BIT 1 NOT SET
: OR SEQUENCE ERROR

001526 01276 000011 177776
001534 00526 177774
001540 000000

MOV #11,-2(R2)
INC -4(R2)
HALT

719

720

.SBTTL TEST # 7 - SHIFT CARRY INTO BIT 0

 :TEST 7 - SHIFT CARRY INTO BIT 0

001542 005212
 001544 022712 000007
 001550 001023
 721 001552 012737 000000 000000
 722 001560 000261
 723 001562 006137 000000
 724 001566 103020

TST7: INC (R2) ;UPDATE TEST NUMBER
 CMP #7,(R2) ;SEQUENCE ERROR?
 BNE TST10-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,@#0 ;CLEAR LOCATION
 SEC ;SET CARRY
 ROL @#0 ;ROTATE CARRY BIT TO BIT 0
 BCC TST10

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 770 <

001570 012762 000012 177776
 001576 005262 177774
 001602 000000
 725 001604 022737 000001 000000
 726 001612 001406

MOV #12,-2(R2) ;MOVE TO MAILBOX # ***** 12 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CARRY CLEAR
 ; OR SEQUENCE ERROR
 CMP #1,@#0 ;BIT 0 SET
 BEQ TST10

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
 ; CONDITIONAL BRANCH INST. AND <===
 ; REPLACE THE MOVE INSTRUCTION <===
 ; WHICH FOLLOWS W/ 756 <===
 ; *****

001614 012762 000013 177776
 001622 005262 177774
 001626 000000

MOV #13,-2(R2) ;MOVE TO MAILBOX # ***** 13 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BIT 0 NOT SET
 ; OR SEQUENCE ERROR

727

728

.SBTTL TEST # 10 - LEFT SHIFT FROM BIT 0 TO C-BIT

 :TEST 10 - LEFT SHIFT FROM BIT 0 TO C-BIT

001630 005212
 001632 022712 000010
 001636 001016
 729 001640 012737 000001 000000
 730 001646 012700 177757
 731 001652 000241
 732 001654 005200
 733 001656 001404
 734 001660 006137 000000
 735 001664 103373
 736 001666 001406

TST10: INC (R2) ;UPDATE TEST NUMBER
 CMP #10,(R2) ;SEQUENCE ERROR?
 BNE TST11-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #1,@#0 ;SET BIT 0
 MOV #-21,R0 ;SET BIT COUNTER
 CLC ;CLEAR C-BIT
 SHL: INC R0 ;INCREMENT BIT COUNTER
 BEQ SHLE ;BR TO ERROR HALT IF BIT IS LOST
 ROL @#0 ;SHIFT LEFT ONE POSITION
 BCC SHL ;BRANCH IF C-BIT NOT SET
 BEQ TST11

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 ; CONDITIONAL BRANCH INST. AND <=====
 ; REPLACE THE MOVE INSTRUCTION <=====
 ; WHICH FOLLOWS W/ 763 <=====
 ;

001670
 001670 012762 000014 177776
 001676 005262 177774
 001702 000000

SHLE: MOV #14,-2(R2) ;MOVE TO MAILBOX # ***** 14 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;LEFT SHIFTING LOGIC FAILED
 ; OR SEQUENCE ERROR

737

738

.SBTTL TEST # 11 - SHIFT BIT 15 TO BIT 14
 :*****
 :TEST 11 - SHIFT BIT 15 TO BIT 14
 :*****

001704 005212
 001706 022712 000011
 001712 001014
 739 001714 012737 100000 000000
 740 001722 000241
 741 001724 006037 000000
 742 001730 022737 040000 000000
 743 001736 001406

TST11: INC (R2) ;UPDATE TEST NUMBER
 CMP #11,(R2) ;SEQUENCE ERROR?
 BNE TST12-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #100000,@#0 ;SET BIT 15
 CLC ;CLEAR CARRY
 ROR @#0 ;SHIFT BIT 15 TO BIT 14
 CMP #40000,@#0 ;SUCCESSFUL
 BEQ TST12
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 765 <====
 MOV #15,-2(R2) ;MOVE TO MAILBOX # ***** 15 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BIT 14 NOT SET
 ; OR SEQUENCE ERROR

001740 012762 000015 177776
 001746 005262 177774
 001752 000000

744

TEST # 12 - RIGHT SHIFT FROM BIT 15 TO C-BIT

745

.SBTTL TEST # 12 - RIGHT SHIFT FROM BIT 15 TO C-BIT

:TEST 12 - RIGHT SHIFT FROM BIT 15 TO C-BIT

001754 005212
001756 022712 000012
001762 001016
746 001764 012737 100000 000000
747 001772 012700 177757
748 001776 000241
749 002000 005200
750 002002 001404
751 002004 006037 000000
752 002010 103373
753 002012 001406

TST12: INC (R2) ;UPDATE TEST NUMBER
CMP #12,(R2) ;SEQUENCE ERROR?
BNE TST13-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #100000,@#0 ;SET BIT 15
MOV #-21,R0 ;SET BIT COUNTER
CLC ;CLEAR C-BIT
SHR: INC R0 ;INCREMENT BIT COUNTER
BEQ SHRE ;BR TO ERROR HALT IF BIT IS LOST
ROR @#0 ;ROTATE RIGHT ONE POSITION
BCC SHR ;BRANCH IF C-BIT CLEAR
BEQ TST13

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 763 <

002014
002014 012762 000016 177776
002022 005262 177774
002026 000000

SHRE: MOV #16,-2(R2) ;MOVE TO MAILBOX # ***** 16 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RIGHT SHIFT LOGIC FAILED
; OR SEQUENCE ERROR

754
755
756
757
758
759
760
76
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781

.SBTTL SCRATCH PAD TESTS

: THE SCRATCH PAD TESTS ARE USED TO VERIFY THAT VARIOUS
: DATA PATTERNS CAN BE SUCCESSFULLY HELD IN THE SCRATCH PAD
: CIRCUITRY. MOVE AND COMPARE INSTRUCTIONS ARE USED TO TEST THAT
: R0 CAN HOLD VARIOUS DATA PATTERNS. EACH DATA PATTERN IS
: MOVED AND TESTED IN A SMALL LOOP CONVENIENT FOR SCOPING. THE
: SUCCESSFUL COMPLETION OF THESE TESTS SHOULD VERIFY THE CIRCUITRY EXTERNAL
: TO THE SCRATCH PAD ITSELF.
: THE REMAINDER OF THE GENERAL REGISTERS ARE TESTED BY MOVING
: A BIT INTO BIT 0 OF THE REGISTER AND SHIFTING IT LEFT ONE
: BIT AT A TIME INTO THE CARRY BIT. THE RESULT IS THEN CHECKED TO INSURE THAT
: NO BITS WERE PICKED. THE PROCEDURE IS THEN REPEATED UNDER OPPOSITE
: CONDITIONS. THE GENERAL REGISTER AND THE CARRY BIT ARE SET TO
: ALL ONES, AND A ZERO IS SHIFTED LEFT FROM BIT 0 INTO THE CARRY BIT.
: THE RESULT IS THEN CHECKED TO INSURE THAT NO ZEROES WERE PICKED.
: AT THIS POINT ALL OF THE GENERAL REGISTERS HAVE BEEN EXERCISED
: AS WELL AS REGISTER 11. REGISTERS 10 AND 12 HAVE BEEN ACCESSED BY
: THE INSTRUCTIONS. REGISTERS 13,14,AND 17 WILL BE TESTED LATER IN THE
: MICROCODE TESTS.
: IF THE PATTERN TESTS WITH REGISTER 0 FAIL CHECK THE RESULTANT
: DATA FOR A CLUE TO A FAULT IN THE EXTERNAL CIRCUITRY. IF THE
: PATTERN TESTS WITH R0 ARE SUCCESSFUL BUT THE TESTS WITH THE OTHER
: REGISTERS FAIL, SUSPECT THE REGISTER SELECT LINES AND THEN THE SCRATCH
: PAD ITSELF.
: *****

782

.SBTTL TEST # 13 - TEST IF R0 CAN HOLD ALL ZEROES

:TEST 13 - TEST IF R0 CAN HOLD ALL ZEROES

002030 005212
002032 022712 000013
002036 001006

TST13: INC (R2) ;UPDATE TEST NUMBER
CMP #13,(R2) ;SEQUENCE ERROR?
BNE TST14-10 ;BR TO ERROR HALT ON SEQ ERROR

783

784 002040 012700 000000
785 002044 005700
786 002046 001406

MOV #0,R0 ;MOVE ZEROES TO R0
TST R0 ;SUCCESSFUL?
BEQ TST14

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 773 .

002050 012762 000017 177776
002056 005262 177774
002062 000000

MOV #17,-2(R2) ;MOVE TO MAILBOX # ***** 17 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 NOT 0
; OR SEQUENCE ERROR

787

788

.SBTTL TEST # 14 - TEST IF R0 CAN HOLD ONES AND ZEROES
 :*****
 :TEST 14 - TEST IF R0 CAN HOLD ONES AND ZEROES
 :*****

002064 005212
 002066 022712 000014
 002072 001007
 789 002074 012700 125252
 790 002100 020027 125252
 791 002104 001406

TST14: INC (R2) ;UPDATE TEST NUMBER
 CMP #14,(R2) ;SEQUENCE ERROR?
 BNE TST15-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125252,R0 ;MOVE ALTERNATING ONES AND ZEROES TO R0
 CMP R0,#125252 ;SUCCESSFUL?
 BEQ TST15

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 ; CONDITIONAL BRANCH INST. AND <---
 ; REPLACE THE MOVE INSTRUCTION <---
 ; WHICH FOLLOWS W/ 772 <---

002106 012762 000020 177776
 002114 005262 177774
 002120 000000

MOV #20,-2(R2) ;MOVE TO MAILBOX # ***** 20 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 NOT 125252
 ; OR SEQUENCE ERROR

792

793

.SBTTL TEST # 15 - TEST IF RO CAN HOLD ZEROES AND ONES
:*****
:TEST 15 - TEST IF RO CAN HOLD ZEROES AND ONES
:*****

002122 005212
002124 022712 000015
002130 001007
794 002132 012700 052525
795 002136 020027 052525
796 002142 001406

TST15: INC (R2) ;UPDATE TEST NUMBER
CMP #15,(R2) ;SEQUENCE ERROR?
BNE TST16-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #052525,RO ;MOVE ALTERNATING ZEROES AND ONES TO RO
CMP RO,#052525 ;SUCCESSFUL?
BEQ TS-16

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <= --
: CONDITIONAL BRANCH INST. AND <-- --
: REPLACE THE MOVE INSTRUCTION <-- --
: WHICH FOLLOWS W/ 772 <= --

002144 012762 000021 177776
002152 005262 177774
002156 000000

MOV #21,-2(R2) ;MOVE TO MAILBOX # ***** 21 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RO NOT 52525
; OR SEQUENCE ERROR

797

798

.SBTTL TEST # 16 - TEST IF R0 CAN HOLD ALL ONES
:*****
:TEST 16 - TEST IF R0 CAN HOLD ALL ONES
:*****

002160 005212
002162 022712 000016
002166 001007
799 002170 012700 177777
800 002174 020027 177777
801 002200 001406

TST16: INC (R2) ;UPDATE TEST NUMBER
CMP #16,(R2) ;SEQUENCE ERROR?
BNE TST17-10 ;BR TO ERROR HALT ON SEQ FRROR
MOV #177777,R0 ;MOVE ALL ONES TO R0
CMP R0,#177777 ;SUCCESSFUL?
BEG TST17

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--=
; CONDITIONAL BRANCH INST. AND <--
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 772 <

002202 012762 000022 177776
002210 005262 177774
002214 000000

MOV #22,-2(R2) ;MOVE TO MAILBOX # ***** 22 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 NOT 177777
; OR SEQUENCE ERROR

802

803

.SBTTL TEST # 17 - TEST IF R1 CAN HOLD A ONE IN ALL BITS
 :*****
 :TEST 17 - TEST IF R1 CAN HOLD A ONE IN ALL BITS
 :*****

002216 005212
 002220 022712 000017
 002224 001014
 804 002226 012701 000001
 805 002232 012700 177757
 806 002236 000241
 807 002240 005200
 808 002242 001403
 809 002244 006101
 810 002246 103374
 811 002250 001406

TST17: INC (R2) ;UPDATE TEST NUMBER
 CMP #17,(R2) ;SEQUENCE ERROR?
 BNE TST20-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #1,R1 ;SET BIT 0
 MOV #-21,R0 ;SET BIT COUNTER
 CLC ;CLEAR C-BIT
 REG1: INC R0 ;INCREMENT BIT COUNTER
 BEQ REG1E ;BR TO ERROR HALT IF BIT IS LOST
 ROL R1 ;ROTATE 1 POSITION
 BCC REG1 ;ALL DONE
 BEQ TST20

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND < -
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 765 < -

002252
 002252 012762 000023 177776
 002260 005262 177774
 002264 000000

REG1E: MOV #23,-2(R2) ;MOVE TO MAILBOX # ***** 23 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R1
 ; OR SEQUENCE ERROR

812

813

.SBTTL TEST # 20 - TEST IF R1 CAN HOLD A ZERO IN ALL BITS
 :*****
 :TEST 20 - TEST IF R1 CAN HOLD A ZERO IN ALL BITS
 :*****

002266 005212
 002270 022712 000020
 002274 001016
 814 002276 012701 177776
 815 002302 012700 177757
 816 002306 000261
 817 002310 005200
 818 002312 001405
 819 002314 006101
 820 002316 103774
 821 002320 022701 177777
 822 002324 001406

TST20: INC (R2) ;UPDATE TEST NUMBER
 CMP #20,(R2) ;SEQUENCE ERROR?
 BNE TST21-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #-2,R1 ;SET ALL ONES IN R1 EXCEPT FOR BIT 0
 MOV #-21,R0 ;SET BIT COUNTER
 SEC ;SET C-BIT
 REG1A: INC R0 ;INCREMENT COUNTER
 BEQ R1ERR ;BR TO ERROR HALT IF COUNTER=0
 ROL R1 ;ROTATE 1 POSITION
 BCS REG1A ;CONTINUE UNTIL C-BIT IS CLEAR
 CMP #-1,R1 ;CHECK DATA IN R1
 BEQ TST21

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 763 <====

002326
 002326 012762 000024 177776
 002334 005262 177774
 002340 000000

R1ERR: MOV #24,-2(R2) ;MOVE TO MAILBOX # ***** 24 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R1
 ; OR SEQUENCE ERROR

823

```
.SBITL TEST # 21 - TEST IF R2 CAN HOLD A ONE IN ALL BITS
:*****
:TEST 21 - TEST IF R2 CAN HOLD A ONE IN ALL BITS
:*****
```

002342	005212			TST21:	INC	(R2)		;UPDATE TEST NUMBER
002344	022712	000021			CMP	#21,(R2)		;SEQUENCE ERROR?
002350	001014				BNE	REG2A-14		;BR TO ERROR HALT ON SEQ ERROR
824 002352	012702	000001			MOV	#1,R2		;SET BIT 0
825 002356	012700	177757			MOV	#-21,R0		;SET BIT COUNTER
826 002362	000241				CLC			;CLEAR C-BIT
827 002364	005200			REG2:	INC	R0		;INCREMENT BIT COUNTER
828 002366	001405				BEQ	REG2A-14		;BR TO ERROR HALT IF BIT IS LOST
829 002370	006102				ROL	R2		;ROTATE 1 POSITION
830 002372	103374				BCC	REG2		;ALL DONE
831 002374	001410				BEQ	REG2A		
832								; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
833								; BRANCH INSTRUCTION AND <====
834								; REPLACE THE MOVE INSTRUCTION <====
835								; FOLLOWING W/ 771 <====
836 002376	012702	0C0304			MOV	#\$TESTN,R2		;RESTORE POINTER
837 002402	012762	000025	177776		MOV	#25,-2(R2)		;MOVE TO MAILBOX # ***** 25 *****
	002410	005262	177774		INC	-4(R2)		;SET MSGTYP TO FATAL ERROR
	002414	000600			HALT			;FAILURE WITH R2
838 002416	012702	000304		REG2A:	MOV	#\$TESTN,R2		;RESTORE POINTER
839								

```

840          .SBTTL TEST # 22 - TEST IF R2 CAN HOLD A ZERO IN ALL BITS
          ;*****
          ;TEST 22 - TEST IF R2 CAN HOLD A ZERO IN ALL BITS
          ;*****
          TST22: INC      (R2)          ;UPDATE TEST NUMBER
          CMP      #22,(R2)          ;SEQUENCE ERROR?
          BNE     TST23-10          ;BR TO ERROR HALT ON SEQ ERROR
841 002432 012702 177776          MOV      #-2,R2          ;SET ALL ONES IN R2 EXCEPT FOR BIT 0
842 002436 012700 177757          MOV      #-21,R0         ;SET BIT COUNTER
843 002442 000261          SEC          ;SET C-BIT
844 002444 005200          REG2B: INC      R0          ;INCREMENT BIT COUNTER
845 002446 001407          BEQ      R2ERR         ;BR TO ERROR HALT IF COUNTER=0
846 002450 006102          ROL      R2          ;ROTATE 1 POSITION
847 002452 103774          BCS     REG2B         ;CONTINUE UNTIL C-BIT IS CLEAR
848 002454 022702 177777          CMP      #-1,R2         ;CHECK DATA IN R2
849 002460 001410          BEQ     REG2C         ;
850 002462 012702 000304          MOV     #$TESTN,R2      ;RESTORE POINTER
851 002466          R2ERR: MOV     #26,-2(R2)      ;MOVE TO MAILBOX # ***** 26 *****
          002466 012762 000026 177776          INC     -4(R2)          ;SET MSGTYP TO FATAL ERROR
          002474 005262 177774          HALT          ;FAILURE WITH R2
852 002500 000000          REG2C: MOV     #$TESTN,R2      ;RESTORE POINTER
853 002502 012702 000304
    
```

854

.SBTTL TEST # 23 - TEST IF R3 CAN HOLD A ONE IN ALL BITS

.....
 .TEST 23 - TEST IF R3 CAN HOLD A ONE IN ALL BITS

002506 005212
 002510 022712 000023
 002514 001014
 855 002516 012703 000001
 856 002522 012700 177757
 857 002526 000241
 858 002530 005200
 859 002532 001403
 860 002534 006103
 861 002536 103374
 862 002540 001406

TST23: INC (R2) ;UPDATE TEST NUMBER
 CMP #23,(R2) ;SEQUENCE ERROR?
 BNE TST24-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #1,R3 ;SET BIT 0
 MOV #-21,R0 ;SET BIT COUNTER
 CLC ;CLEAR C-BIT
 REG3: INC R0 ;INCREMENT BIT COUNTER
 BEQ REG3E ;BR TO ERROR HALT IF BIT IS LOST
 ROL R3 ;ROTATE 1 POSITION
 BCC REG3 ;ALL DONE
 BEQ TST24

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <-
 ; REPLACE THE MOVE INSTRUCTION <- -
 ; WHICH FOLLOWS W/ 765 <

002542
 002542 012762 000027 177776
 002550 005262 177774
 002554 000000

REG3E: MOV #27,-2(R2) ;MOVE TO MAILBOX # ***** 27 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R3
 ; OR SEQUENCE ERROR

863

864

.SBTTL TEST # 24 - TEST IF R3 CAN HOLD A ZERO IN ALL BITS
 :*****
 :TEST 24 - TEST IF R3 CAN HOLD A ZERO IN ALL BITS
 :*****

002556 005212
 002560 022712 000024
 002564 001016
 865 002566 012703 177776
 866 002572 012700 177757
 867 002576 000261
 868 002600 005200
 869 002602 001405
 870 002604 006103
 871 002606 103774
 872 002610 022703 177777
 873 002614 001406

TST24: INC (R2) ;UPDATE TEST NUMBER
 CMP #24,(R2) ;SEQUENCE ERROR?
 BNE TST25-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #-2,R3 ;SET ALL 'ONES IN R3 EXCEPT FOR BIT 0
 MOV #-21,P0 ;SET BIT COUNTER
 SEC ;SET C-BIT
 REG3A: INC R0 ;INCREMENT BIT COUNTER
 BEQ R3ERR ;BR TO ERROR HALT IF COUNTER=0
 ROL R3 ;ROTATE 1 POSITION
 BCS REG3A ;CONTINUE UNTIL C-BIT IS CLEAR
 CMP #-1,R3 ;CHECK DATA
 BEQ TST25

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 763 <

002616
 002616 012762 000030 177776
 002624 005262 177774
 002630 000000

R3ERR: MOV #30,-2(R2) ;MOVE TO MAILBOX # ***** 30 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R3
 ; OR SEQUENCE ERROR

874

875

.SBTTL TEST # 25 - TEST IF R4 CAN HOLD A ONE IN ALL BITS
:*****
:TEST 25 - TEST IF R4 CAN HOLD A ONE IN ALL BITS
:*****

002632 005212
002634 022712 000025
002640 001014
876 002642 012704 000001
877 002646 012700 177757
878 002652 000241
879 002654 005200
880 002656 001403
881 002660 006104
882 002662 103374
883 002664 001406

TST25: INC (R2) ;UPDATE TEST NUMBER
CMP #25,(R2) ;SEQUENCE ERROR?
BNE TST26-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #1,R4 ;SET BIT 0
MOV #-21,R0 ;SET BIT COUNTER
CLC ;CLEAR C-BIT
REG4: INC R0 ;INCREMENT BIT COUNTER
BEQ REG4E ;BR TO ERROR HALT IF BIT IS LOST
ROL R4 ;ROTATE 1 POSITION
BCC REG4 ;ALL DONE
BEQ TST26

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS w/ 765 <

002666
002666 012762 000031 177776
002674 005262 177774
002700 000000

REG4E: MOV #31,-2(R2) ;MOVE TO MAILBOX # ***** 31 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;FAILURE WITH R4
; OR SEQUENCE ERROR

884

885

.SBTTL TEST # 26 - TEST IF R4 CAN HOLD A ZERO IN ALL BITS
 :*****
 :TEST 26 - TEST IF R4 CAN HOLD A ZERO IN ALL BITS
 :*****

002702 005212
 002704 022712 000026
 002710 001016
 886 002712 012704 177776
 887 002716 012700 177757
 888 002722 000261
 889 002724 005200
 890 002726 001405
 891 002730 006104
 892 002732 103774
 893 002734 022704 177777
 894 002740 001406

TST26: INC (R2) ;UPDATE TEST NUMBER
 CMP #26,(R2) ;SEQUENCE ERROR?
 BNE TST27-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #-2,R4 ;SET ALL ONES IN R4 EXCEPT FOR BIT 0
 MOV #-21,R0 ;SET BIT COUNTER
 SEC ;SET C-BIT
 REG4A: INC R0 ;INCREMENT BIT COUNTER
 BEQ R4ERR ;BR TO ERROR HALT IF COUNTER=0
 ROL R4 ;ROTATE 1 POSITION
 BCS REG4A ;CONTINUE UNTIL C-BIT IS CLEAR
 CMP #-1,R4 ;CHECK DATA
 BEQ TST27

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 763 <====

002742
 002742 012762 000032 177776
 002750 005262 177774
 002754 000000

R4ERR: MOV #32,-2(R2) ;MOVE TO MAILBOX # ***** 32 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R4
 ; OR SEQUENCE ERROR

895
 896

897

.SBTTL TEST # 27 - TEST IF R5 CAN HOLD A ONE IN ALL BITS

 :TEST 27 - TEST IF R5 CAN HOLD A ONE IN ALL BITS

002756 005212
 002760 022712 000027
 002764 001014
 898 002766 012705 000001
 899 002772 012700 177757
 900 002776 000241
 901 003000 005200
 902 003002 001403
 903 003004 006105
 904 003006 103374
 905 003010 001406

TST27: INC (R2) ;UPDATE TEST NUMBER
 CMP #27,(R2) ;SEQUENCE ERROR?
 BNE TST30-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #1,R5 ;SET BIT 0
 MOV #-2,R0 ;SET BIT COUNTER
 CLC ;CLEAR C-BIT
 REG5: INC R0 ;INCREMENT BIT COUNTER
 BEQ REG5E ;BR TO ERROR HALT IF BIT IS LOST
 ROL R5 ;ROTATE 1 POSITION
 BCC REG5 ;ALL DONE
 BEQ TST30

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <...>
 ; CONDITIONAL BRANCH INST. AND <...>
 ; REPLACE THE MOVE INSTRUCTION <...>
 ; WHICH FOLLOWS W/ 765 <...>

003012
 003012 012762 000033 177776
 003020 005262 177774
 003024 000000

REG5E: MOV #33,-2(R2) ;MOVE TO MAILBOX # ***** 33 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R5
 ; OR SEQUENCE ERROR

906

907

.SBTTL TEST # 30 - TEST IF R5 CAN HOLD A ZERO IN ALL BITS

 :TEST 30 - TEST IF R5 CAN HOLD A ZERO IN ALL BITS

003026 005212
 003030 022712 000030
 003034 001016
 908 003036 012705 177776
 909 003042 012700 177757
 910 003046 000261
 911 003050 005200
 912 003052 001405
 913 003054 006105
 914 003056 103774
 915 003060 022705 177777
 916 003064 001406

TST30: INC (R2) ;UPDATE TEST NUMBER
 CMP #30,(R2) ;SEQUENCE ERROR?
 BNE TST31-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #-2,R5 ;SET ALL ONES IN R5 EXCEPT FOR BIT 0
 MOV #-21,R0 ;SET BIT COUNTER
 SEC ;SET C-BIT
 REG5A: INC R0 ;INCREMENT BIT COUNTER
 BEQ R5ERR ;BR TO ERROR HALT IF COUNTER=0
 ROL R5 ;ROTATE 1 POSITION
 BCS REG5A ;CONTINUE UNTIL C-BIT IS C;EAR
 CMP #-1,R5 ;CHECK DATA
 BEQ TST31

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 ; CONDITIONAL BRANCH INST. AND <=====
 ; REPLACE THE MOVE INSTRUCTION <=====
 ; WHICH FOLLOWS W/ 763 <=====
 ;

003066
 003066 012762 000034 177776
 003074 005262 177774
 003100 000000

R5ERR: MOV #34,-2(R2) ;MOVE TO MAILBOX # ***** 34 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;FAILURE WITH R5
 ; OR SEQUENCE ERROR

917

918

.SBTTL TEST # 31 - TEST IF R6 CAN HOLD A ONE IN ALL BITS

:TEST 31 - TEST IF R6 CAN HOLD A ONE IN ALL BITS

003102 005212
003104 022712 000031
003110 001014
919 003112 012706 000001
920 003116 012700 177757
921 003122 000241
922 003124 005200
923 003126 001403
924 003130 006106
925 003132 103374
926 003134 001406

TST31: INC (R2) ;UPDATE TEST NUMBER
CMP #31,(R2) ;SEQUENCE ERROR?
BNE TST32-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #1,R6 ;SET BIT 0
MOV #-21,R0 ;SET BIT COUNTER
CLC ;CLEAR C-BIT
REG6: INC R0 ;INCREMENT BIT COUNTER
BEQ REG6E ;BR TO ERROR HALT IF BIT IS LOST
ROL R6 ;ROTATE 1 POSITION
BCC REG6 ;ALL DONE
BEQ TST32

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
; CONDITIONAL BRANCH INST. AND <=====
; REPLACE THE MOVE INSTRUCTION <=====
; WHICH FOLLOWS W/ 765 <=====
;

003136
003136 012762 000035 177776
003144 005262 177774
003150 000000

REG6E: MOV #35,-2(R2) ;MOVE TO MAILBOX # ***** 35 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;FAILURE WITH R6
; OR SEQUENCE ERROR

927

928

.SBTTL TEST # 32 - TEST IF R6 CAN HOLD A ZERO IN ALL BITS

003152 005212
 003154 022712 000032
 003160 001016
 929 003162 012706 177776
 930 003166 012700 177757
 931 003172 000261
 932 003174 005200
 933 003176 001405
 934 003200 006106
 935 003202 103774
 936 003204 022706 177777
 937 003210 001406

```

:*****
:TEST 32 - TEST IF R6 CAN HOLD A ZERO IN ALL BITS
:*****
TST32:  INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #32,(R2)      ;SEQUENCE ERROR?
        BNE     TST33-10      ;BR TO ERROR HALT ON SEQ ERROR
        MOV     #-2,R6         ;SET ALL ONES IN R6 EXCEPT FOR BIT 0
        MOV     #-21,R0        ;SET BIT COUNTER
        SEC     ;SET C-BIT
REG6A:  INC      R0            ;INCREMENT BIT COUNT
        BEQ     R6ERR         ;BR TO ERROR HALT IF COUNTER=0
        ROL     R6            ;ROTATE 1 POSITION
        BCS     REG6A         ;CONTINUE UNTIL C-BIT IS CLEAR
        CMP     #-1,R6        ;CHECK DATA
        BEQ     TST33
  
```

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
;           CONDITIONAL BRANCH INST. AND <===
;           REPLACE THE MOVE INSTRUCTION <===
;           WHICH FOLLOWS W/ 763 <--
  
```

003212
 003212 012762 000036 177776
 003220 005262 177774
 003224 000000

```

R6ERR:  MOV     #36,-2(R2)     ;MOVE TO MAILBOX # ***** 36 *****
        INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT   ;FAILURE WITH R6
        ; OR SEQUENCE ERROR
  
```

938
 939
 940
 941
 942
 943
 944
 945
 946
 947
 948
 949
 950
 951
 952
 953

```

:*****
:SBTTL PSW TESTS
:
: THE PSW TESTS ARE USED TO VERIFY THAT VARIOUS DATA
: PATTERNS CAN BE SUCCESSFULLY HELD IN THE PSW AND THAT THE
: PSW ADDRESSING LOGIC IS FUNCTIONING. MOVE AND COMPARE INSTRUCTIONS
: ARE USED TO TEST THAT THE PSW CAN HOLD VARIOUS DATA PATTERNS.
: EACH DATA PATTERN IS MOVED AND TESTED IN A SMALL LOOP CONVENIENT FOR
: SCOPING.
:
: THE PSW REGISTER ITSELF IS TESTED AS WELL AS THE ADDRESS
: SELECT CIRCUITRY. THE AMUX INPUTS TO THE PSW MUX ARE TESTED. THE
: CC INPUTS ARE TESTED LATER IN THE MICROCODE TESTS. SETTING OF
: THE T-BIT BY THE TEST PATTERNS IS PURPOSELY AVOIDED; TESTING OF THE
: T-BIT TRAP CIRCUITRY IS LEFT FOR THE TRAP TEST.
:
  
```

954

.SBTTL TEST # 33 - TEST IF PSW WILL HOLD ZEROES
:*****
:TEST 33 - TEST IF PSW WILL HOLD ZEROES
:*****

003226 005212
003230 022712 000033
003234 001012
955 003236 012706 001000
956 003242 012737 000000 177776
957 003250 005737 177776
958 003254 001406

TST33: INC (R2) ;UPDATE TEST NUMBER
CMP #33,(R2) ;SEQUENCE ERROR?
BNE TST34-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #STBOT,R6
MOV #0,@#PS ;SET PSW TO ZERO
TST @#S ;SUCCESSFUL
BEQ TST34

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 767 <====
; MOVE TO MAILBOX # ***** 37 *****
; SET MSGTYP TO FATAL ERROR
; PSW NOT 0
; OR SEQUENCE ERROR

003256 012762 000037 177776
003264 005262 177774
003270 000000

MOV #37,-2(R2)
INC -4(R2)
HALT

959

960

.SBTTL TEST # 34 - TEST IF PSW WILL HOLD ONES AND ZEROES
:*****
:TEST 34 - TEST IF PSW WILL HOLD ONES AND ZEROES
:*****

003272 005212
003274 022712 000034
003300 001011
961 003302 012737 000252 177776
962 003310 023727 177776 000252
963 003316 001406

TST34: INC (R2) ;UPDATE TEST NUMBER
CMP #34,(R2) ;SEQUENCE ERROR?
BNE TST35-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #252,@#PS ;MOVE ALT. ONES AND ZEROES TO PSW
CMP @#PS,#252 ;SUCCESSFUL?
BEQ TS-35

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <==
; CONDITIONAL BRANCH INST. AND <===
; REPLACE THE MOVE INSTRUCTION <==
; WHICH FOLLOWS W/ 770 <===

003320 012762 000040 177776
003326 005262 177774
003332 000000

MOV #40,-2(R2) ;MOVE TO MAILBOX # ***** 40 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PSW NOT 252
; OR SEQUENCE ERROR

964

965

.SBTTL TEST # 35 - TEST IF PSW (EXCEPT T-BIT) WILL HOLD 0'S & 1'S
:*****
:TEST 35 - TEST IF PSW (EXCEPT T-BIT) WILL HOLD 0'S & 1'S
:*****

003334 005212
003336 022712 000035
003342 001011
966 003344 012737 000105 177776
967 003352 023727 177776 000105
968 003360 001406

TST35: INC (R2) ;UPDATE TEST NUMBER
CMP #35,(R2) ;SEQUENCE ERROR?
BNE TST36-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #105,@#PS ;MOVE ALT. ONES AND ZEROES TO PSW
CMP @#PS,#105 ;SUCCESSFUL?
BEQ TS-36

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < =
; CONDITIONAL BRANCH INST. AND <- -
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 770 <- -

003362 012762 000041 177776
003370 005262 177774
003374 000000

MOV #41,-2(R2) ;MOVE TO MAILBOX # ***** 41 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PSW NOT 105
; OR SEQUENCE ERROR

969

```

970 .SBTTL TEST # 36 - TEST IF PSW (EXCEPT T-BIT) WILL HOLD ALL ONES
:*****
:TEST 36 - TEST IF PSW (EXCEPT T-BIT) WILL HOLD ALL ONES
:*****
003376 005212          TST36: INC      (R2)          ;UPDATE TEST NUMBER
003400 022712 000036  CMP      #36,(R2)      ;SEQUENCE ERROR?
003404 001011          BNE      TST37-10     ;BR TO ERROR HALT ON SEQ ERROR
971 003406 012737 000357 177776  MOV      #357,@#PS    ;MOVE ONES TO PSW
972 003414 023727 177776 000357  CMP      @#PS,#357    ;SUCCESSFUL
973 003422 001406          BEQ      TST37

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <----
; WHICH FOLLOWS W/ 770 <----
003424 012762 000042 177776  MOV      #42,-2(R2)   ;MOVE TO MAILBOX # ***** 42 *****
003432 005262 177774          INC      -4(R2)      ;SET MSGTYP TO FATAL ERROR
003436 000000          HALT                ;PSW NOT 357
; OR SEQUENCE ERROR
  
```

```

974 .SBTTL CONDITION CODE TEST
975 :*****
976 :
977 :
978 : THIS TEST CHECKS THE CONDITIONAL BRANCHES INVOLVING THE Z-BIT.
979 : THE Z-BIT IS SET WITH ALL OTHER CC BITS ZERO AND BOTH CONDITIONS
980 : BEQ AND BNE ARE TESTED FOR PROPER EXECUTION. THEN THE Z-BIT IS
981 : SET WITH ALL OTHER CC BITS CLEAR AND BOTH CONDITIONS ARE TESTED
982 : AGAIN FOR PROPER OPERATION.
983 : THIS TEST CHECKS THE OPERATION OF THE SET AND CLEAR CONDITION
984 : CODE INSTRUCTIONS AND CHECKS THE CIRCUITRY EXTERNAL TO THE CONDITIONAL
985 : BRANCH ROM. THE BRANCH MICROCODE FOR ALTERING THE PC AND FOR
986 : LEAVING THE PC UNALTERED IS TESTED. ONLY THOSE ROM ADDRESSES SPECIFICALLY
987 : USED IN THE TEST ARE VERIFIED HERE.
988 :
  
```


1014

.SBITL TEST # 40 - TEST BRANCHES AROUND N-BIT

.....
 :TEST 40 - TEST BRANCHES AROUND N-BIT
 :.....

003520 005212
 003522 022712 000040
 003526 001020
 1015
 1016 003530 000257
 1017 003532 000270
 1018 003534 100001
 1019 003536 100406

TST40: INC (R2) ;UPDATE TEST NUMBER
 CMP #40,(R2) ;SEQUENCE ERROR?
 BNE TST41-10 ;BR TO ERROR HALT ON SEQ ERROR
 ;FIRST WITH N-BIT ON
 CCC ;CC=1000: JUST N-BIT
 SEN
 BPL BRN1 ;CHECK OPPOSITE CONDITION
 BMI BRN2
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 ; CONDITIONAL BRANCH INST. AND
 ; REPLACE THE MOVE INSTRUCTION
 ; WHICH FOLLOWS W/ 773

003540
 003540 012762 000045 177776
 003546 005262 177774
 003552 000000
 1020
 1021 003554 000277
 1022 003556 000250
 1023 003560 100401
 1024 003562 100006

BRN1: MOV #45,-2(R2) ;MOVE TO MAILBOX # ***** 45 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;IMPROPER BR W/ N=1
 ;CHECK WITH N-BIT OFF
 BRN2: SCC ;CC=0111
 CLN
 BMI BRN3 ;CHECK OPPOSITE CONDITION
 BPL TST41
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 ; CONDITIONAL BRANCH INST. AND
 ; REPLACE THE MOVE INSTRUCTION
 ; WHICH FOLLOWS W/ 761

003564
 003564 012762 000046 177776
 003572 005262 177774
 003576 000000

BRN3: MOV #46,-2(R2) ;MOVE TO MAILBOX # ***** 46 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;IMPROPER BR W/ N=0
 ; OR SEQUENCE ERROR

1025
 1026
 1027
 1028
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1038

.....
 : THIS TEST CHECKS THE CONDITIONAL BRANCHES INVOLV'G THE V-BIT.
 : THE V-BIT IS SET WITH ALL OTHER CC BITS ZERO AND BOTH CONDITIONS
 : BVS AND BVC ARE TESTED FOR PROPER EXECUTION. THEN THE V-BIT IS
 : SET WITH ALL OTHER CC BITS CLEAR AND BOTH CONDITIONS ARE TESTED
 : AGAIN FOR PROPER OPERATION.
 : THIS TEST CHECKS THE OPERATION OF THE SET AND CLEAR CONDITION
 : CODE INSTRUCTIONS AND CHECKS THE CIRCUITRY EXTERNAL TO THE CONDITIONAL
 : BRANCH ROM. THE BRANCH MICROCODE FOR ALTERING THE PC AND FOR
 : LEAVING THE PC UNALTERED IS TESTED. ONLY THOSE ROM ADDRESSES SPECIFICALLY
 : USED IN THE TEST ARE VERIFIED HERE.
 :
 :

1064

.SBTTL TEST # 42 - TEST BRANCHES AROUND C-BIT

 :TEST 42 - TEST BRANCHES AROUND C-BIT

003660 005212
 003662 022712 000042
 003666 001020
 1065
 1066 003670 000257
 1067 003672 000261
 1068 003674 103001
 1069 003676 103406

TST42: INC (R2) ;UPDATE TEST NUMBER
 CMP #42,(R2) ;SEQUENCE ERROR?
 BNE TST43-10 ;BR TO ERROR HALT ON SEQ ERROR
 ;FIRST WITH C-BIT ON
 CCC ;CC=0001: JUST C-BIT
 SEC
 BCC BRC1 ;CHECK OPPOSITE CONDITION
 BCS BRC2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 : CONDITIONAL BRANCH INST. AND <-
 : REPLACE THE MOVE INSTRUCTION <-
 : WHICH FOLLOWS W/ 773 <-

003700
 003700 012762 000051 177776
 003706 005262 177774
 003712 000000
 1070
 1071 003714 000277
 1072 003716 000241
 1073 003720 103401
 1074 003722 100406

BRC1: MOV #51,-2(R2) ;MOVE TO MAILBOX # ***** 51 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;IMPROPER BR W/ C=1
 ;CHECK WITH C-BIT OFF

BRC2: SCC ;CC=1110
 CLC
 BCS BRC3 ;CHECK OPPOSITE CONDITION
 BMI TST43

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 : CONDITIONAL BRANCH INST. AND <-
 : REPLACE THE MOVE INSTRUCTION <-
 : WHICH FOLLOWS W/ 761 <-

003724
 003724 012762 000052 177776
 003732 005262 177774
 003736 000000

BRC3: MOV #52,-2(R2) ;MOVE TO MAILBOX # ***** 52 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;IMPROPER BR W/ C=0
 ; OR SEQUENCE ERROR

1075
 1076
 1077
 1078
 1079
 1080
 1081
 1082
 1083
 1084
 1085
 1086
 1087
 1088
 1089
 1090
 1091
 1092
 1093
 1094
 1095
 1096
 1097

 .SBTTL MICROCODE TESTS

: THE MICROCODE TESTS ARE USED TO VERIFY THE MICROPROGRAMM
 : FLOW. THE GOAL OF THESE TESTS IS TO EXERCISE EVERY POSSIBLE
 : BRANCH IN THE MICROPROGRAM FLOW.
 : THE TEST EXERCISES EVERY BRANCH IN THE MICROCODE BY
 : TESTING AT LEAST ONE INSTRUCTION FROM EVERY CLASS OF INSTRUCTION IN
 : ALL POSSIBLE MODES. FOR EXAMPLE, TO TEST THE SINGLE OPERAND INSTRUCTIONS,
 : AT LEAST ONE SINGLE OPERAND INSTRUCTION IS VERIFIED IN ALL UNIQUE
 : ADDRESSING MODES. BYTE MODES ARE ALSO TESTED. AS EACH NEW
 : MODE IS INTRODUCED THE SAME INSTRUCTION IS TRIED AND TESTED IN
 : A SMALL LOOP CONVENIENT FOR SCOPING. THE TEST IS SET UP USING
 : ONLY INSTRUCTIONS AND ADDRESSING MODES WHICH HAVE BEEN PREVIOUSLY
 : VERIFIED.
 : IF THESE TESTS FAIL, CHECK THE RESULTS FOR A CLUE TO THE
 : FAULT.

1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110

.....
: THE CLR INSTRUCTION IS USED TO INTRODUCE EACH ADDRESSING
: MODE WITH THE SINGLE OPERAND INSTRUCTION. FOLLOWING THE SEQUENCE CHECK,
: THE CLR INSTRUCTION IS EXECUTED AND A BRANCH TEST IS EXECUTED WHICH
: CHECKS THAT THE Z-BIT WAS PROPERLY SET. THIS SMALL TEST IS SELF-SUFFICIENT
: AND CAN BE SCOPED TO TROUBLE SHOOT ALL OF THE IR DECODE LOGIC AND
: MICROCODE FOR SOP INSTRUCTIONS WITH MODE 0. FOLLOWING THIS TEST
: SEVERAL OTHER SOP INSTRUCTIONS ARE INTRODUCED WITH MODE 0. THESE
: INSTRUCTIONS MAINPULATE DATA AND SERVE TO CHECK THE DATA RESULTS
: OF THE SOP INSTRUCTIONS IN THIS TEST. THE DATA IN THIS TEST IS
: OPERATED ON BY EACH INSTRUCTION WITHOUT REINITIALIZING.
:

1111

.SBTTL TEST # 43 - TEST MODE 0 USING SOP INST.

 :TEST 43 - TEST MODE 0 USING SOP INST.

003740 005212
 003742 022712 000043
 003746 001026
 1112 003750 005000
 1113 003752 001406

TST43: INC (R2) ;UPDATE TEST NUMBER
 CMP 4'3,(R2) ;SEQUENCE ERROR?
 BNE TST44-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;TRY THE CLEAR INST.
 BEQ SOPOA

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 : CONDITIONAL BRANCH INST. AND <-
 : REPLACE THE MOVE INSTRUCTION <-
 : WHICH FOLLOWS w/ 775 <-

003754 012762 000053 177776
 003762 005262 177774
 003766 000000
 1114 003770 005200
 1115 003772 005100
 1116 003774 005200
 1117 003776 100406

MOV #53,-2(R2) ;MOVE TO MAILBOX # ***** 53 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLR DID NOT SET Z-BIT
 SOPOA: INC R0 ;TRY THE INCREMENT INST.
 COM R0 ;TRY COMPLEMENT
 INC R0
 BMI SOPOB

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS w/ 763 <

004000 012762 000054 177776
 004006 005262 177774
 004012 000000
 1118 004014 005100
 1119 004016 001406

MOV #54,-2(R2) ;MOVE TO MAILBOX # ***** 54 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGATE DID NOT SET N-BIT
 SOPOB: COM R0 ;TRY COMPLEMENT INST.
 BEQ TST44

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS w/ 753 <

004020 012762 000055 177776
 004026 005262 177774
 004032 000000

MOV #55,-2(R2) ;MOVE TO MAILBOX # ***** 55 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CUMULATIVE RESULT OF CLR,INC,NEG AND COM INSTS. FAILED
 ; OR SEQUENCE ERROR

1120
 1121
 1122
 1123
 1124
 1125
 1126
 1127
 1128
 1129
 1130

 : THIS TEST INTRODUCES THE REMAINING SOP INSTRUCTIONS AND TESTS
 : THEM IN MODE 0. THE PURPOSE IS TO PROVIDE A BASELINE OF
 : INSTRUCTIONS FOR USE IN THE SUBSEQUENT TESTS. SINCE THE MICROCODE FOR
 : THESE INSTRUCTIONS IS IDENTICAL TO THAT ALREADY TESTED, ANY TROUBLE
 : SHOOTING EFFORTS SHOULD BE AIMED AT THE ACTUAL IR DECODE AND ALU
 : FUNCTIONING.
 :

1131

.SBTTL TEST # 44 - TEST REMAINDER OF SOP INSTS IN MODE 0

:TEST 44 - TEST REMAINDER OF SOP INSTS IN MODE 0

004034 005212
004036 022712 000044
004042 001025
1132 004044 005000
1133 004046 005300
1134 004050 100406

TST44: INC (R2) ;UPDATE TEST NUMBER
CMP #44,(R2) ;SEQUENCE ERROR?
BNE TST45-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;INITIALIZE
DEC R0 ;TRY DECREMENT INST.
BMI SOP0C

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
: CONDITIONAL BRANCH INST. AND <===
: REPLACE THE MOVE INSTRUCTION <===
: WHICH FOLLOWS W/ 774 <===

004052 012762 000056 177776
004060 005262 177774
004064 000000
1135 004066 000261
1136 004070 005500
1137 004072 001007
1138 004074 000261
1139 004076 005600
1140 004100 100004
1141 004102 005100
1142 004104 005200
1143 004106 005300
1144 004110 001406

SOPOC: MOV #56,-2(R2) ;MOVE TO MAILBOX # ***** 56 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;N-BIT NOT SET ON DEC
SEC ;INITIALIZE CARRY
ADC R0 ;TRY ADD CARRY INST
BNE SOPOD
SEC ;INITIALIZE CARRY
SBC R0 ;TRY SUBTRACT-CARRY INST
BPL SOPOD
COM R0
INC R0
DEC R0
BEQ TST45

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--=
: CONDITIONAL BRANCH INST. AND <--=
: REPLACE THE MOVE INSTRUCTION <--=
: WHICH FOLLOWS W/ 754 <--=

004112
004112 012762 000057 177776
004120 005262 177774
004124 000000

SOPOD: MOV #57,-2(R2) ;MOVE TO MAILBOX # ***** 57 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CUMULATIVE RESULT OF ADC,SBC,COM,INC AND DEC INSTS. FAIL
; OR SEQUENCE ERROR

1145
1146
1147
1148
1149
1150
1151

: THIS TEST INTRODUCES THE BYTE CONTROL LOGIC OF THE PROCESSOR.
: THE MODE 0 BYTE MICROCODE IS TESTED. THE METHOD AND SEQUENCE
: OF TESTING IS THE SAME AS THAT USED IN THE SOP MODE 0 TESTS.

```
1152 .SBTTL TEST # 45 - TEST MODE 0 EVEN BYTE USING SOP INST
:*****
:TEST 45 - TEST MODE 0 EVEN BYTE USING SOP INST
:*****
004126 005212
004130 022712 000045
004134 001016
1153 004136 105000
1154 004140 001406
TST45: INC (R2) ;UPDATE TEST NUMBER
CMP #45,(R2) ;SEQUENCE ERROR?
BNE TST46-10 ;BR TO ERROR HALT ON SEQ ERROR
CLRB R0 ;TRY CLEARING EVEN BYTE OF REGISTER
BEQ SOPBOA
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 775 <====
004142 012762 000060 177776 MOV #60,-2(R2) ;MOVE TO MAILBOX # ***** 60 *****
004150 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
004154 000000 HALT ;CLRB DID NOT SET Z-BIT
1155 004156 105100 SOPBOA: COMB R0 ;TRY SETTING EVEN BYTE OF REGISTER
1156 004160 100002 BPL SOPBOB
1157 004162 105200 INCB R0 ;TRY INCREMENTING EVEN BYTE OF REGISTER>>
1158 004164 001406 BEQ TST46
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 763 <====
004166 SOPBOB: MOV #61,-2(R2) ;MOVE TO MAILBOX # ***** 61 *****
004166 012762 000061 177776 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
004174 005262 177774 HALT ;TEST CUMULATIVE RESULT OF ABOVE BYTE INST.
004200 000000 ; OK SEQUENCE ERROR
```

```
1159
1160 :*****
1161 :
1162 : THIS TEST USES THE CLR INSTRUCTION TO INTRODUCE AND TEST
1163 : SINGLE OPERAND MODE 1 INSTRUCTIONS. AGAIN, THE CLR INSTRUCTION
1164 : IS USED TO INTRODUCE THE MICROCODE AND TO TEST THAT THE PROPER
1165 : CONDITION CODES ARE SET. OTHER SOP INSTRUCTIONS ARE USED TO MANIPULATE
1166 : COMMON DATA TO VERIFY THAT THE CORRECT DATA IS PRODUCED.
1167 :
```


1168

.SBTTL TEST # 46 - TEST MODE 1 USING SOP INST.

.....
 :TEST 46 - TEST MODE 1 USING SOP INST.
 :.....

004202 005212
 004204 022712 000046
 004210 001020
 1169 004212 005000
 1170 004214 005010
 1171 004216 001406

TST46: INC (R2) ;UPDATE TEST NUMBER
 CMP #46,(R2) ;SEQUENCE ERROR?
 BNE TST47-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;INITIALIZE R0
 CLR (R0) ;TRY CLEAR INST W/MODE 1
 BEQ SOP1A

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
 : CONDITIONAL BRANCH INST. AND <===
 : REPLACE THE MOVE INSTRUCTION <===
 : WHICH FOLLOWS W/ 774 <===

004220 012762 000062 177776
 004226 005262 177774
 004232 000000
 1172 004234 005310
 1173 004236 100003
 1174 004240 000261
 1175 004242 005510
 1176 004244 001406

MOV #62,-2(R2) ;MOVE TO MAILBOX # ***** 62 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLR DID NOT SET Z-BIT
 SOP1A: DEC (R0) ;TRY DECREMENT INST W/MODE 1
 BPL SOP1B
 SEC
 ADC (R0) ;INITIALIZE CARRY
 BEQ TST47 ;TRY ADD-CARRY W/MODE 1

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
 : CONDITIONAL BRANCH INST. AND <===
 : REPLACE THE MOVE INSTRUCTION <===
 : WHICH FOLLOWS W/ 761 <===

004246
 004246 012762 000063 177776
 004254 005262 177774
 004260 000000

SOP1B: MOV #63,-2(R2) ;MOVE TO MAILBOX # ***** 63 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;TEST CUMULATIVE RESULT OF ABOVE INST
 ; OR SEQUENCE ERROR

1177
 1178
 1179
 1180
 1181
 1182
 1183
 1184

.....
 : THIS TEST VERIFIES THE BYTE INSTRUCTION MICROCODE FOR MODE 1
 : SINGLE OPERAND INSTRUCTIONS.
 : THIS IS THE FIRST PLACE THE SIGN EXTEND LOGIC IS EXERCISED
 : AND VERIFIED.
 :

1185

.SBTTL TEST # 47 - TEST MODE 1 EVEN BYTE USING SOP INST
:*****
:TEST 47 - TEST MODE 1 EVEN BYTE USING SOP INST
:*****

004262 005212
004264 022712 000047
004270 001024
1186 004272 005000
1187 004274 005010
1188 004276 005110
1189 004300 105010
1190 004302 001406

TST47: INC (R2) ;UPDATE TEST NUMBER
CMP #47,(R2) ;SEQUENCE ERROR?
BNE TST50-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;INITIALIZE R0
CLR (R0) ;INITIALIZE LOC. 0
COM (R0)
CLRB (R0) ;TRY TO CLEAR BYTE 0
BEQ SOPB1A
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 772 <====

004304 012762 000064 177776
004312 005262 177774
004316 000000
1191 004320 005210
1192 004322 100005
1193 004324 105110
1194 004326 105210
1195 004330 100002
1196 004332 105210
1197 004334 001406

MOV #64,-2(R2) ;MOVE TO MAILBOX # ***** 64 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CLRB DID NOT SET Z-BIT
SOPB1A: INC (R0) ;INCREMENT TO TEST WORD
BPL SOPB1B
COMB (R0) ;COMPLEMENT: ODD BYTE = 376
INCB (R0) ;INC: ODD BYTE = 377
BPL SOPB1B
INCB (R0)
BEQ TST50 ;INCREMENT ODD BYTE=0

004336
004336 012762 000065 177776
004344 005262 177774
004350 000000

SOPB1B: MOV #65,-2(R2) ;MOVE TO MAILBOX # ***** 65 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CHECK CUMMULATIVE RESULT OF ABOVE INST
; OR SEQUENCE ERROR

1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208

:*****
: THIS TEST VERIFIES THAT SINGLE OPERAND BYTE INSTRUCTIONS WILL
: FUNCTION CORRECTLY FOR ODD BYTES.
: THIS IS THE FIRST TIME THAT ADDRESS LINE 0 HAS BEEN
: EXERCISED. CHECKS ARE MADE THAT THE PROPER BYTE IS MODIFIED AND
: THE CONDITION CODES ARE CHECKED. IT IS ALSO VERIFIED THAT THE UNADDRESSED
: BYTE IS NOT ALTERED BY THE INSTRUCTION.
:*****

1209

.SBTTL TEST # 50 - TEST MODE 1 ODD BYTE USING SOP INST
:*****
:TEST 50 - TEST MODE 1 ODD BYTE USING SOP INST
:*****

004352 005212
004354 022712 000050
004360 001026
1210 004362 005000
1211 004364 005010
1212 004366 005110
1213 004370 005200
1214 004372 105010
1215 004374 001406

TST50: INC (R2) ;UPDATE TEST NUMBER
CMP #50,(R2) ;SEQUENCE ERROR?
BNE TST51-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;INITIALIZE R0
CLR (R0) ;INITIALIZE LOC. 0
COM (R0)
INC R0 ;R0=ODD BYTE
CLRB (R0) ;TRY TO CLEAR BYTE 1
BEQ SOPB1C

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 771 <====

004376 012762 000066 177776
004404 005262 177774
004410 000000
1216 004412 005300
1217 004414 005210
1218 004416 005200
1219 004420 105110
1220 004422 105210
1221 004424 100002
1222 004426 105210
1223 004430 001406

MOV #66,-2(R2) ;MOVE TO MAILBOX # ***** 66 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CLRB DID NOT SET Z-BIT
SOPB1C: DEC R0 ;R0=WORD ADDR.
INC (R0) ;INCREMENT TO TEST WORD
INC R0 ;R0=ODD BYTE
COMB (R0) ;TRY TO COMPLEMENT BYTE 1
INCB (R0)
BPL SOPB1D
INCB (R0)
BEQ TST51

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 753 <====

004432
004432 012762 000067 177776
004440 005262 177774
004444 000000

SOPB1D: MOV #67,-2(R2) ;MOVE TO MAILBOX # ***** 67 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TEST CUMMULATIVE RESULT OF ABOVE INST.
; OR SEQUENCE ERROR

1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234

:*****
: THIS TEST VERIFIES MODE 2 SINGLE-OPERAND INSTRUCTIONS. PREVIOUSLY
: TESTED INSTRUCTIONS ARE USED TO SET A POINTER IN R0 TO LOC. 400.
: LOC. 400 IS INITIALIZED TO -1 BEFORE A CLR MODE 2 IS EXECUTED.
: THEN R0 IS DECREMENTED BY TWO TO AGAIN POINT TO 400 BEFORE EACH
: OF SEVERAL MODE 2 INSTRUCTIONS ARE USED TO VERIFY THE DATA RESULTS OF
: THE TEST. THIS PROCEDURE ALSO VERIFIES THE PROPER INCREMENTING OF THE
: REGISTER.
:*****

1235

.SBTTL TEST # 51 - TEST MODE 2 USING SOP INST.

 :TEST 51 - TEST MODE 2 USING SOP INST.

004446 005212
 004450 022712 000051
 004454 001027
 1236 004456 005000
 1237 004460 105100
 1238 004462 005200
 1239 004464 005010
 1240 004466 005110
 1241 004470 005020
 1242 004472 001406

TST51: INC (R2) ;UPDATE TEST NUMBER
 CMP #51,(R2) ;SEQUENCE ERROR?
 BNE TST52-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;SET R0=400
 COMB R0
 INC R0
 CLR (R0) ;CLEAR 400
 COM (R0) ;INITIALIZE: 400=-1
 CLR (R0)+ ;TRY CLEARING WITH MODE 2
 BEQ SOPZA

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 770 <====

004474 012762 000070 177776
 004502 005262 177774
 004506 000000
 1243 004510 005300
 1244 004512 005300
 1245 004514 005120
 1246 004516 100004
 1247 004520 005300
 1248 004522 005300
 1249 004524 005220
 1250 004526 001406

MOV #70,-2(R2) ;MOVE TO MAILBOX # ***** 70 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLR INST DID NOT SET Z-BIT
 SOPZA: DEC R0 ;RESET R0
 DEC R0
 COM (R0)+ ;TRY COMPLEMENTING WITH MODE 2
 BPL SOP2B
 DEC R0 ;RESET R0
 DEC R0
 INC (R0)+ ;TRY INCREMENTING WITH MODE 2
 BEQ TST52

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 752 <====

004530
 004530 012762 000071 177776
 004536 005262 177774
 004542 000000

SOP2B: MOV #71,-2(R2) ;MOVE TO MAILBOX # ***** 71 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CHECK CUMMULATIVE RESULT OF ABOVE INST
 ; OR SEQUENCE ERROR

1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258
 1259
 1260
 1261

 : THIS TEST VERIFIES MODE 2 SINGLE OPERAND INSTRUCTIONS WHICH
 : ADDRESS EVEN BYTES. R0 IS SET TO 400 AND USED TO INITIALIZE LOCATION
 : 400 TO -1. CLRB INSTRUCTION IS THEN EXECUTED ON BYTE 400 WITH
 : MODE 2.
 : R0 IS THEN DECREMENTED BEFORE EACH OF SEVERAL MODE 2 INSTRUCTIONS
 : WHICH ARE USED TO VERIFY THE DATA RESULTS OF THE TEST. THIS PROCEDURE ALSO
 : VERIFIES THE PROPER INCREMENTING OF THE REGISTER.
 :

```

1262          .SBTTL TEST # 52 - TEST MODE 2 EVEN BYTE USING SOP INST.
:*****
:TEST 52 - TEST MODE 2 EVEN BYTE USING SOP INST.
:*****
          004544 005212          TST52: INC      (R2)          ;UPDATE TEST NUMBER
          004546 022712 000052  CMP      #52,(R2)       ;SEQUENCE ERROR?
          004552 001027          BNE     TST53-10       ;BR TO ERROR HALT ON SEQ ERROR
1263 004554 005000          CLR     R0            ;SET R0=400
1264 004556 105100          COMB   R0
1265 004560 005200          INC     R0
1266 004562 005010          CLR     (R0)          ;CLEAR 400
1267 004564 005110          COM     (R0)          ;INITIALIZE: 400=-1
1268 004566 105020          CLRB  (R0)+          ;TRY TO CLEAT 400 W/MGDE 2
1269 004570 001406          BEQ    SOPB2A

          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
          ;          CONDITIONAL BRANCH INST. AND
          ;          REPLACE THE MOVE INSTRUCTION
          ;          WHICH FOLLOWS W/ 770
          ;          <==
          ;          <==
          ;          <==
          ;          <==

          004572 012762 000072 177776  MOV     #72,-2(R2)    ;MOVE TO MAILBOX # ***** 72 *****
          004600 005262 177774          INC     -4(R2)
          004604 000000          HALT
1270 004606 005300  SOPB2A: DEC     R0            ;SET MSGTYP TO FATAL ERROR
1271 004610 005210          INC     (R0)          ;CLR DID NOT SET Z-BIT
1272 004612 105110          COMB   (R0)          ;RESULT R0=400
1273 004614 105220          INCB  (R0)+          ;INC 400 TO TEST WORD
1274 004616 100003          BPL    SOPB2B        ;TRY TO INC EVEN BYTE
1275 004620 005300          DEC     R0            ;RESET R0=400
1276 004622 105220          INCB  (R0)+          ;TRY INCREMENT OF EVEN BYTE
1277 004624 001406          BEQ    TST53

          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
          ;          CONDITIONAL BRANCH INST. AND
          ;          REPLACE THE MOVE INSTRUCTION
          ;          WHICH FOLLOWS W/ 752
          ;          <
          ;          <-
          ;          -
          ;          <

          004626          SOPB2B: MOV     #73,-2(R2)    ;MOVE TO MAILBOX # ***** 73 *****
          004626 012762 000073 177776  INC     -4(R2)
          004634 005262 177774          HALT
          004640 000000

1278
1279
1280
1281
1282
1283
          ;*****
          ; THIS TEST FOLLOWS THE SAME PROCEDURE DESCRIBED IN THE PREVIOUS
          ; TEST. HERE, THE BYTE INSTRUCTION IS USED TO ADDRESS AN ODD BYTE.
          ;
    
```

1284

.SBTTL TEST # 53 - TEST MODE 2 ODD BYTE USING SOP INST.
:*****
:TEST 53 - TEST MODE 2 ODD BYTE USING SOP INST.
:*****

004642 005212
004644 022712 000053
004650 001032
1285 004652 005000
1286 004654 105100
1287 004656 005200
1288 004660 005010
1289 004662 005110
1290 004664 005200
1291 004666 105020
1292 004670 001406

TST53: INC (R2) ;UPDATE TEST NUMBER
CMP #53,(R2) ;SEQUENCE ERROR?
BNE TST54-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;SET R0=400
COMB R0
INC R0
CLR (R0) ;CLEAR LOC 400
COM (R0) ;INITIALIZE: 400=-1
INC R0 ;R0=ODD BYTE
CLRB (R0)+ ;TRY TO CLEAR ODD BYTE
BEQ SOPB2C
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <==--
; CONDITIONAL BRANCH INST. AND <==--
; REPLACE THE MOVE INSTRUCTION <==--
; WHICH FOLLOWS W/ 767 <==--

004672 012762 000074 177776
004700 005262 177774
004704 000000
1293 004706 005300
1294 004710 005300
1295 004712 005220
1296 004714 005300
1297 004716 105110
1298 004720 105220
1299 004722 100003
1300 004724 005300
1301 004726 105220
1302 004730 001406

MOV #74,-2(R2) ;MOVE TO MAILBOX # ***** 74 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CLRB DID NOT SET Z-BIT
SOPB2C: DEC R0 ;R0=WORD ADDR.
DEC R0
INC (R0)+ ;INCREMENT WORD
DEC R0 ;POINT TO ODD BYTE
COMB (R0) ;COMPLEMENT ODD BYTE
INCB (R0)+ ;TRY TO INCREMENT ODD BYTE
BPL SOPB2D
DEC R0 ;RESET R0 TO ODD BYTE
INCB (R0) ;TRY TO INCREMENT ODD BYTE
BEQ TST54
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 747 <====

004732
004732 012762 000075 177776
004740 005262 177774
004744 000000

SOPB2D: MOV #75,-2(R2) ;MOVE TO MAILBOX # ***** 75 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TEST CUMMULATIVE RESULT OF ABOVE INST.
; OR SEQUENCE ERROR

1303
1304
1305
1306
1307
1308

:*****
: THESE TESTS CHECK THE NEGATE INSTRUCTION IN ALL MODES. PREVIOUSLY
: TESTED SINGLE-OPERAND INSTRUCTIONS ARE USED TO TEST THE NEGATE INSTRUCTION.
:*****

*309

.SBTTL TEST # 54 - TEST MODE 0 USING NEGATE INSTRUCTION

.....
 :TEST 54 - TEST MODE 0 USING NEGATE INSTRUCTION

004746 005212
 004750 022712 000054
 004754 001045
 1310 004756 005000
 1311 004760 005200
 1312 004762 005400
 1313 004764 100003
 1314 004766 001402
 1315 004770 102401
 1316 004772 103406

TST54: INC (R2) ;UPDATE TEST NUMBER
 CMP #54,(R2) ;SEQUENCE ERROR?
 BNE TST55-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;SET R0=0
 INC R0 ; R0=1
 NEG R0 ;TRY NEGATE MODE 0: R0--'
 BPL NEG00 ;CC=1001?
 BEQ NEG00
 BVS NEG00
 BCS NEG01

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 770 <====

004774
 004774 012762 000076 177776
 005002 005262 177774
 005006 000000
 1317
 1318 005010 005200
 1319 005012 001406

NEG00: MOV #76,-2(R2) ;MOVE TO MAILBOX # ***** 76 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGATE DID NOT SET CC'S CORRECTLY
 NEG01: INC R0 ;TEST DATA RESULT
 BEQ NEG02

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 760 <====

005014 012762 000077 177776
 005022 005262 177774
 005026 000000
 1320
 1321 005030 105100
 1322 005032 105400
 1323 005034 100403
 1324 005036 001402
 1325 005040 102401
 1326 005042 103406

MOV #77,-2(R2) ;MOVE TO MAILBOX # ***** 77 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DATA RESULT OF NEGATE INCORRECT
 NEG02: COMB R0 ;R0=377
 NEGB R0 ;R0=1
 BMI NEG03 ;CC=0001?
 BEQ NEG03
 BVS NEG03
 BCS NEG04

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 744 <====

005044
 005044 012762 000100 177776
 005052 005262 177774
 005056 000000
 1327 005060 005300
 1328 005062 001406

NEG03: MOV #100,-2(R2) ;MOVE TO MAILBOX # ***** 100 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGB DID NOT SET CC'S CORRECTLY
 NEG04: DEC R0 ;TEST DATA RESULT
 BEQ TST55

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 734 <====

005064 012762 000101 177776
 005072 005262 177774
 005076 000000

MOV #101,-2(R2) ;MOVE TO MAILBOX # ***** 101 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DATA RESULT OF NEGB INCORRECT
 ; OR SEQUENCE ERROR

1330

.SBTTL TEST # 55 - TEST MODE 1 USING NEGATE INST.

:TEST 55 - TEST MODE 1 USING NEGATE INST.

005100 005212
 005102 022712 000055
 005106 001050
 1330 005110 005000
 1331 005112 005010
 1332 005114 005210
 1333 005116 005410
 1334 005120 100003
 1335 005122 001402
 1336 005124 102401
 1337 005126 103406

TST55: INC (R2) ;UPDATE TEST NUMBER
 CMP #55,(R2) ;SEQUENCE ERROR?
 BNE TST56-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;POINT TO LOC. 0
 CLR (R0) ;CLEAR LOC. 0
 INC (R0) ;LOC. 0=1
 NEG (R0) ;TRY NEG. LOC. 0=-1
 BPL NEG10 ;CC=1001
 BEQ NEG10
 BVS NEG10
 BCS NEG11

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 767 <====

005130
 005130 012762 000102 177776
 005136 005262 177774
 005142 000000
 1338
 1339 005144 005237 000000
 1340 005150 001406

NEG10: MOV #102,-2(R2) ;MOVE TO MAILBOX # ***** 102 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGATE DID NOT SET CC'S CORRECTLY

1338
 1339 005144 005237 000000
 1340 005150 001406

NEG11: INC @R0 ;TEST DATA RESULT
 BEQ NEG12

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 756 <

005152 012762 000103 177776
 005160 005262 177774
 005164 000000
 1341 005166 105110
 1342 005170 105410
 1343 005172 100403
 1344 005174 001402
 1345 005176 102401
 1346 005200 103406

NEG12: MOV #103,-2(R2) ;MOVE TO MAILBOX # ***** 103 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DATA RESULT OF NEGATE INCORRECT
 COMB (R0) ;LOC. 0=377
 NEGB (R0) ;TRY NEGB LOC. 0=1
 BMI NEG13 ;CC=0001?
 BEQ NEG13
 BVS NEG13
 BCS NEG14

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 742 <

005202
 005202 012762 000104 177776
 005210 005262 177774
 005214 000000
 1347 005216 005337 000000
 1348 005222 001406

NEG13: MOV #104,-2(R2) ;MOVE TO MAILBOX # ***** 104 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGB DID NOT SET CC'S CORRECTLY
 NEG14: DEC @R0 ;TEST DATA RESULT
 BEQ TST56

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 731 <

005224 012762 000105 177776
 005232 005262 177774
 005236 000000

MOV #105,-2(R2) ;MOVE TO MAILBOX # ***** 105 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DATA RESULT OF NEGB INCORRECT
 ; OR SEQUENCE ERROR

1349

..SBTTL TEST # 56 - TEST MODE 2 USING NEGATE INSTRUCTION

.....
 :TEST 56 - TEST MODE 2 USING NEGATE INSTRUCTION

005240 005212
 005242 022712 000056
 005246 001040
 1350 005250 005000
 1351 005252 005010
 1352 005254 005210
 1353 005256 005420
 1354 005260 100003
 1355 005262 001402
 1356 005264 102401
 1357 005266 103406

TST56: INC (R2) ;UPDATE TEST NUMBER
 CMP #56,(R2) ;SEQUENCE ERROR?
 BNE TST57-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;POINT TO LOC. 0
 CLR (R0) ;CLEAR LOC. 0
 INC (R0) ;LOC. 0=1
 NEG (R0)+ ;TRY NEG.: LOC. 0=-1
 BPL NEG20 ;CC=1001?
 BEQ NEG20
 BVS NEG20
 BCS NEG21

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 767 <====

005270
 005270 012762 000106 177776
 005276 005262 177774
 005302 000000
 1358 005304 105300
 1359 005306 105300
 1360 005310 105420
 1361 005312 105420
 1362 005314 105340
 1363 005316 005300
 1364 005320 001406

NEG20: MOV #106,-2(R2) ;MOVE TO MAILBOX # ***** 106 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGATE DID NOT SET CC'S CORRECTLY
 NEG21: DECB R0 ;R0=LOC. 0
 DECB R0
 NEGB (R0)+ ;BYTE 0=1 R0=1
 NEGB (R0)+ ;BYTE 1=1 R0=2
 DECB -(R0) ;R0=1 LOC. 0=01
 DEC R0 ;R0=0
 BEQ NEG22

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 752 <====

005322 012762 000107 177776
 005330 005262 177774
 005334 000000
 1365 005336 005337 000000
 1366 005342 001406

NEG22: MOV #107,-2(R2) ;MOVE TO MAILBOX # ***** 107 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REGISTER NOT INCREMENTED CORRECTLY
 DEC @R0 ;LOC. 0=0
 BEQ TST57

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
 ; CONDITIONAL BRANCH INST. AND <--
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 741 <

005344 012762 000110 177776
 005352 005262 177774
 005356 000000

MOV #110,-2(R2) ;MOVE TO MAILBOX # ***** 110 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEG BYTE INSTRUCTIONS FAILED
 ; OR SEQUENCE ERROR

1367
 1368
 1369
 1370
 1371
 1372
 1373
 1374
 1375
 1376

.....
 : THIS TEST VERIFIES MODE 3 SINGLE OPERAND INSTRUCTIONS. IT
 : USES LOCATION 0 AS ITS TARGET DATA. A TABLE LOCATED AT LOC. 400
 : THRU 402 IS USED TO SUPPLY THE ADDRESS OF LOCATION 0 TO THE
 : INSTRUCTIONS UNDER TEST.
 : R0 IS SET TO 400, THE START OF THE ADDRESS TABLE, AND A CLR
 : INSTRUCTION IS EXECUTED WITH MODE 3 TO CLEAR LOC. 0. THEN R0
 : IS DECREMENTED BY TWO AND TWO OTHER MODE 3 INSTRUCTIONS OPERATE ON

1377
1378
1379
1380
1381

:LOC. 0 TO VERIFY THE DATA RESULTS OF THE TEST. THE PROPER INCREMENTING
:OF THE REGISTER IS ALSO VERIFIED IN THIS MANNER.
: IF A FAILURE IS DETECTED BE SURE TO VERIFY THAT THE TABLE
:(LOC. 400-402) HAS THE PROPER VALUES (0).
:

1382

.SBTTL TEST # 57 - TEST MODE 3 USING SOP INST.

.....
:TEST 57 - TEST MODE 3 USING SOP INST.
.....

005360 005212
005362 022712 000057
005366 001024
1383 005370 005000
1384 005372 105100
1385 005374 005200
1386 005376 005010
1387 005400 005030
1388 005402 001406

TST57: INC (R2) ;UPDATE TEST NUMBER
CMP #57,(R2) ;SEQUENCE ERROR?
BNE TST60-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;SET R0=400
COMB R0
INC R0
CLR (R0) ;CLEAR LOC 400
CLR @ (R0)+ ;TRY TO CLEAR LOC 0 USING MODE 3 ;R0=402
BEQ SOP3A

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 771 <--->

005404 012762 000111 177776
005412 005262 177774
005416 000000
1389 005420 005300
1390 005422 005300
1391 005424 005130
1392 005426 100002
1393 005430 005230
1394 005432 001406

MOV #111,-2(R2) ;MOVE TO MAILBOX # ***** 111 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CLR DID NOT SET Z-BIT
SOP3A: DEC R0 ;RESET R0=400
DEC R0
COM @ (R0)+ ;TRY TO COMPLEMENT LOC 0 OF MODE 3 ;R0=402
BPL SOP3B
INC @ (R0)+ ;TRY TO INCREMENT LOC 0 W/MODE 3 ;R0=404
BEQ TST60

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 755 <--->

005434
005434 012762 000112 177776
005442 005262 177774
005446 000000

SOP3B: MOV #112,-2(R2) ;MOVE TO MAILBOX # ***** 112 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CUMULATIVE RESULT OF ABOVE INST FAILED
: OR SEQUENCE ERROR

1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407

.....
: THIS TEST VERIFIES MODE 3 SINGLE OPERAND BYTE INSTRUCTIONS
: WHICH ADDRESS EVEN BYTES. AGAIN, THE TARGET LOCATION 0 IS USED
: AND THE SAME TABLE AT 400 IS EMPLOYED.
: AFTER POINTING R4 TO THE TABLE (400) AND SETTING LOCATION
: 0 TO -1, A CLRB INSTRUCTION IS USED TO CLEAR BYTE 0.
: SEVERAL OTHER MODE 3 INSTRUCTIONS ARE THEN USED WITH THE TABLE
: TO VERIFY THE DATA RESULTS AND THE PROPER INCREMENTING OF THE REGISTER.
: IF A FAILURE IS DETECTED, BE SURE THAT THE TABLE (LOCATION 400-402) HAS
: THE PROPER VALUES (0).
:

1428

.SBTTL TEST # 60 - TEST MODE 3 EVEN BYTE USING SOP INST.

.....
 :TEST 60 - TEST MODE 3 EVEN BYTE USING SOP INST.
 :.....

005450 005212
 005452 022712 000060
 005456 001032
 1409 005460 005004
 1410 005462 105104
 1411 005464 005204
 1412 005466 005000
 1413 005470 005010
 1414 005472 005110
 1415 005474 105034
 1416 005476 001406

TST60: INC (R2) ;UPDATE TEST NUMBER
 CMP #60,(R2) ;SEQUENCE ERROR?
 BNE TST61-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R4 ;SET R4=400
 COMB R4
 INC R4
 CLR R0 ;INITIALIZE LOC. 0=-1
 CLR (R0)
 COM (R0) ;LOC. 0=-1
 CLRB @(R4)+ ;TRY TO CLEAR EVEN BYTE ;LOC. 0=177400 R4 400
 BEQ SOPB3A

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS w/ 767 <

005500 012762 000113 177776
 005506 005262 177774
 005512 000000
 1417 005514 005304
 1418 005516 005304
 1419 005520 005234
 1420 005522 100006
 1421 005524 105434
 1422 005526 100004
 1423 005530 005304
 1424 005532 005304
 1425 005534 105234
 1426 005536 001406

MOV #113,-2(R2) ;MOVE TO MAILBOX # ***** 113 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLRB DID NOT SET Z-BIT
 SOPB3A: DEC R4 ;RESET POINTER R4=400
 DEC R4
 INC @(R4)+ ;TRY INCREMENTING WORD LOC.0=177401 R4 400
 BPL SOPB3B
 NEGB @(R4)+ ;TRY TO NEGATE EVEN BYTE ;LOC.0--1 R4=404
 BPL SOPB3B
 DEC R4 ;R4=402
 DEC R4
 INCB @(R4)+ ;TRY TO INCREMENT EVEN BYTE ;LOC. 0 17400
 BEQ TST61

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
 : CONDITIONAL BRANCH INST. AND <=
 : REPLACE THE MOVE INSTRUCTION <=
 : WHICH FOLLOWS w/ 747 <=

005540
 005540 012762 000114 177776
 005546 005262 177774
 005552 000000

SOPB3B: MOV #114,-2(R2) ;MOVE TO MAILBOX # ***** 114 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CUMULATIVE RESULT OF ABOVE INST FAILED
 ; OR SEQUENCE ERROR

1427
 1428
 1429
 1430
 1431
 1432
 1433
 1434
 1435
 1436
 1437
 1438
 1439
 1440

.....
 : THIS TEST VERIFIES MODE 3 SINGLE OPERAND BYTE INSTRUCTIONS
 : WHICH ADDRESS ODD BYTES. THE TARGET IS BYTE 1. A TABLE AT
 : LOC. 400-406 IS USED. R0 SERVES AS THE TABLE POINTER.
 : R0 IS INITIALIZED TO 400. LOC. 0 IS SET TO -1 USING THE
 : FIRST TWO TABLE ENTRIES. A CLRB MODE 3 IS EXECUTED ON BYTE 1 USING
 : TABLE ADDRESS AT 404. R0 IS DECREMENTED TO 402 AND SEVERAL SOP
 : MODE 3 INSTRUCTIONS ARE USED TO VERIFY DATA RESULTS AND PROPER
 : REGISTER INCREMENTING.
 : THE TABLE (400-406) SHOULD CONTAIN 0,0,1,1 BEFORE AND
 : AFTER THE TEST IS RUN.
 :

144

.SBTTL TEST # 61 - TEST MODE 3 ODD BYTE USING SOP INST.
 :*****
 :TEST 61 - TEST MODE 3 ODD BYTE USING SOP INST.
 :*****

005554 005212
 005556 022712 000061
 005562 001030
 1442 005564 005000
 1443 005566 105100
 1444 005570 005200
 1445 005572 005030
 1446 005574 005130
 1447 005576 105030
 1448 005600 001406

TST61: INC (R2) ;UPDATE TEST NUMBER
 CMP #61,(R2) ;SEQUENCE ERROR?
 BNE TST62-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;SET R0=400
 COMB R0
 INC R0
 CLR @(R0)+ ;INITIALIZE
 COM @(R0)+ ;LOC 0=-1 R0=404
 CLRB @(R0)+ ;TRY TO CLEAR ODD BYTE LOC. 0=377 R0=406
 BEQ SOPB3C

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 770 <====

005602 012762 000115 177776
 005610 005262 177774
 005614 000000
 1449 005616 005300
 1450 005620 005300
 1451 005622 005300
 1452 005624 005300
 1453 005626 005230
 1454 005630 105430
 1455 005632 100002
 1456 005634 105230
 1457 005636 001406

MOV #115,-2(R2) ;MOVE TO MAILBOX # ***** 115 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLRB DID NOT SET Z-BIT
 SOPB3C: DEC R0 ;RESET R0=402
 DEC R0
 DEC R0 ;POINT TO EVEN BYTE ADDR.
 DEC R0
 INC @(R0)+ ;INCREMENT WORD LOC. 0=400 R0=404
 NEGB @(R0)+ ;TRY TO NEGATE ODD BYTE LOC. 0=177400 R0=406
 BPL SOPB3D
 INCB @(R0)+ ;TRY TO INCREMENT ODD BYTE LOC.0=0 R0=410
 BEQ TST62

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 751 <====

005640
 005640 012762 000116 177776
 005646 005262 177774
 005652 000000

SOPB3D: MOV #116,-2(R2) ;MOVE TO MAILBOX # ***** 116 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CUMULATIVE RESULT OF ABOVE INSTS FAILED
 ; OR SEQUENCE ERROR

1458

.SBTTL TEST # 62 - TEST MODE 3 USING NEGATE INSTRUCTION

:*****
 :TEST 62 - TEST MODE 3 USING NEGATE INSTRUCTION
 :*****

005654 005212
 005656 022712 000062
 005662 001066
 1459 005664 075000
 1460 005666 105100
 1461 005670 005200
 1462 005672 005010
 1463 005674 005004
 1464 005676 005014
 1465 005700 005214
 1466 005702 005430
 1467 005704 100003
 1468 005706 001402
 1469 005710 102401
 1470 005712 103406

TST62: INC (R2) ;UPDATE TEST NUMBER
 CMP #62,(R2) ;SEQUENCE ERROR?
 BNE TST63-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=400
 COMB R0
 INC R0
 CLR (R0) ;LOC. 400=0
 CLR R4 ;R4=0
 CLR (R4) ;LOC. 0=0
 INC (R4) ;LOC. 0=1
 NEG @ (R0)+ ;TRY NEGATE LOC. 0 -1 R0=402
 BPL NEG30 ;CC-1001?
 BEQ NEG30
 BVS NEG30
 BCS NEG31

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 763 <--->

005714
 005714 012762 000117 177776
 005722 005262 177774
 005726 000000
 1471 005730 005214
 1472 005732 001406

NEG30: MOV #117,-2(R2) ;MOVE TO MAILBOX # ***** 117 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEG DID NOT SET CC'S CORRECTLY
 NEG31: INC (R4) ;LOC. 0=0
 BEQ NEG32

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 753 <--->

005734 012762 000120 177776
 005742 005262 177774
 005746 000000
 1473 005750 105137 000001
 1474 005754 005237 000000
 1475 005760 105430
 1476 005762 100406

MOV #120,-2(R2) ;MOVE TO MAILBOX # ***** 120 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DATA RESULT OF NEG INCORRECT
 NEG32: COMB @#1 ;LOC 0=177400
 INC @#0 ;LOC. 0=177401
 NEGB @ (R0)+ ;TRY NEGB LOC. 0=177777 R0=404
 BMI NEG33

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 737 <--->

005764 012762 000121 177776
 005772 005262 177774
 005776 000000
 1477 006000 105430
 1478 006002 100006

MOV #121,-2(R2) ;MOVE TO MAILBOX # ***** 121 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGB FAILED WITH EVEN BYTE
 NEG33: NEGB @ (R0)+ ;TRY NEGB LOC.0=777 R0=406
 BPL NEG34

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 727 <--->

006004 012762 000122 177776
 006012 005262 177774
 006016 000000
 1479 006020 105137 000001

MOV #122,-2(R2) ;MOVE TO MAILBOX # ***** 122 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEGB FAILED WITH ODD BYTE
 NEG34: COMB @#1 ;LOC. 0=177377

```

1480 006024 105237 000001      INCB  @#1      ;LOC. 0=177777
1481 006030 005214              INC   (R4)     ;LOC. 0=0
1482 006032 001406      BEQ   TST63

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
;           CONDITIONAL BRANCH INST. AND
;           REPLACE THE MOVE INSTRUCTION
;           WHICH FOLLOWS W/ 713
;           <==--
;           <==--
;           <==--
;           <==--

006034 012762 000123 177776      MOV   #123,-2(R2) ;MOVE TO MAILBOX # ***** 123 *****
006042 005262 177774              INC   -4(R2)    ;SET MSGTYP TO FATAL ERROR
006046 000000      HALT          ;DATA RESULT OF NEGB'S INCORRECT
; OR SEQUENCE ERROR
  
```

1483
 1484
 1485
 1486
 1487
 1488
 1489
 1490
 1491

```

:*****
:
: THIS TEST VERIFIES MODE 4 SINGLE OPERAND INSTRUCTIONS.
: RO IS SET TO 400. A CLR INSTRUCTION IS EXECUTED IN MODE 4 TO CLEAR
: LOC. 376. RO IS RESET TO 400 AND A COM INSTRUCTION USING MODE 4
: COMPLMENTS LOC.376.
: TWO INC INSTRUCTIONS AND A MODE 4 INSTRUCTION ARE EXECUTED
: TO COMPLETE THE TEST.
  
```


1492

.SBTTL TEST # 63 - TEST MODE 4 USING SOP INSTS
 :*****
 :TEST 63 - TEST MODE 4 USING SOP INSTS
 :*****

006050 005212
 006052 022712 000063
 006056 001025
 1493 006060 005000
 1494 006062 105100
 1495 006064 005200
 1496 006066 005040
 1497 006070 001406

TST63: INC (R2) ;UPDATE TEST NUMBER
 CMP #63,(R2) ;SEQUENCE ERROR?
 BNE TST64-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;SET R0=400
 COMB R0
 INC R0
 CLR -(R0) ;TRY TO CLEAR USING MODE 4
 BEQ SOP4A

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 ; CONDITIONAL BRANCH INST. AND <=====
 ; REPLACE THE MOVE INSTRUCTION <=====
 ; WHICH FOLLOWS W/ 772 <=====
 ;

006072 012762 000124 177776
 006100 005262 177774
 006104 000000
 1498 006106 005200
 1499 006110 005200
 1500 006112 005140
 1501 006114 100004
 1502 006116 005200
 1503 006120 005200
 1504 006122 005240
 1505 006124 001406

MOV #124,-2(R2) ;MOVE TO MAILBOX # ***** 124 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLR DID NOT SET Z-BIT
 SOP4A: INC R0 ;RESET R0
 INC R0
 COM -(R0) ;TRY TO COMPLEMENT USING MODE 4
 BPL SOP4B
 INC R0 ;MOVE POINTER
 INC R0
 INC -(R0)
 BEQ TST64

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 ; CONDITIONAL BRANCH INST. AND <=====
 ; REPLACE THE MOVE INSTRUCTION <=====
 ; WHICH FOLLOWS W/ 754 <=====
 ;

006126
 006126 012762 000125 177776
 006134 005262 177774
 006140 000000

SOP4B: MOV #125,-2(R2) ;MOVE TO MAILBOX # ***** 125 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CHECK CUMULATIVE RESULT OF ABOVE INST.
 ; OR SEQUENCE ERROR

1506
 1507
 1508
 1509
 1510
 1511
 1512
 1513
 1514
 1515
 1516
 1517
 1518
 1519
 1520
 1521

:*****
 : THIS TEST VERIFIES MODE 5 SINGLE OPERAND INSTRUCTIONS. IT
 : USES LOCATION 0 AS ITS TARGET DATA. A TABLE LOCATED AT LOC. 372
 : THRU 374 IS USED TO SUPPLY THE ADDRESS OF LOCATION 0 TO THE
 : INSTRUCTIONS UNDER TEST.
 : R0 IS SET TO 376, (THE START OF THE ADDRESS TABLE) +2,
 : AND A CLR INSTRUCTION IS EXECUTED WITH MODE 3 TO CLEAR
 : LOC. 0. THEN R0 IS INCREMENTED BY TWO AND TWO OTHER MODE 3
 : INSTRUCTIONS OPERATE ON LOC. 0 TO VERIFY THE DATA RESULTS OF
 : THE TEST. THE PROPER DECREMENTING OF THE REGISTER IS ALSO
 : VERIFIED IN THIS MANNER.
 : IF A FAILURE IS DETECTED BE SURE TO VERIFY THAT THE TABLE
 : (LOC. 372 THRU 374) HAS THE PROPER VALUES (0).
 :*****

1522

.SBTTL TEST # 64 - TEST MODE 5 USING SOP INSTS
:*****
:TEST 64 - TEST MODE 5 USING SOP INSTS
:*****

006142	005212			TST64:	INC	(R2)		:UPDATE TEST NUMBER
006144	022712	000064			CMP	#64,(R2)		:SEQUENCE ERROR?
006150	001023				BNE	TST65-10		:BR TO ERROR HALT ON SEQ ERROR
1523	006152	005000			CLR	R0		:SET R0=376
1524	006154	005020			CLR	(R0)+		
1525	006156	105400			NEGB	R0		
1526	006160	005050			CLR	@-(R0)		:TRY TO CLEAR LOC 0 W/MODE 5
1527	006162	001406			BEQ	SOP5A		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
: CONDITIONAL BRANCH INST. AND <=
: REPLACE THE MOVE INSTRUCTION <=
: WHICH FOLLOWS W/ 772 <=
:*****

006164	012762	000126	177776		MOV	#126,-2(R2)		:MOVE TO MAILBOX # ***** 126 *****
006172	005262	177774			INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
006176	000000				HALT			:CLR DID NOT SET Z-BIT
1528	006200	005200		SOP5A:	INC	R0		:RESET R0
1529	006202	005200			INC	R0		
1530	006204	005150			COM	@-(R0)		:TRY TO COMPLEMENT LOC. 0 W/MODE 5
1531	006206	100002			BPL	SOP5B		
1532	006210	005250			INC	@-(R0)		:TRY TO INCREMENT LOC. 0 W/MODE 5
1533	006212	001406			BEQ	TST65		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 756 <

006214				SOP5B:	MOV	#127,-2(R2)		:MOVE TO MAILBOX # ***** 127 *****
006214	012762	000127	177776		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
006222	005262	177774			HALT			:TEST CUMMULATIVE RESULT OF ABOVE INSTS
006226	000000							: OR SEQUENCE ERROR

1534
1535
1536
1537
1538
1539
1540
1541
1542

:*****
: THIS TEST VERIFIES MODE 6 SINGLE OPERAND INSTRUCTIONS. IT
: USES LOCATION 0 AS ITS TARGET DATA. R0 IS SET TO 400 USING
: PREVIOUSLY TESTED INSTRUCTIONS AND A MODE 6 CLR INSTRUCTION IS
: EXECUTED ON LOC. 0 USING R0 AND A -400 OFFSET. COM AND INC
: INSTRUCTIONS ARE THEN USED TO VERIFY THE DATA.
:*****

```

1543          .SBTTL TEST # 65 - TEST MODE 6 USING SOP INSTS
              :*****
              :TEST 65 - TEST MODE 6 USING SOP INSTS
              :*****
006230 005212          TST65: INC      (R2)          ;UPDATE TEST NUMBER
006232 022712 000065  CMP      #65,(R2)       ;SEQUENCE ERROR?
006236 001024          BNE      TST66-10      ;BR TO ERROR HALT ON SEQ ERROR
1544 006240 005000          CLR      R0          ;SET R0=400
1545 006242 105100          COMB     R0
1546 006244 005200          INC      R0
1547 006246 005060 177400  CLR      -400(R0)      ;TRY TO CLEAR LOCATION 0 W/MODE 6
1548 006252 001406          BEQ      SOP6A
              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ;          CONDITIONAL BRANCH INST. AND
              ;          REPLACE THE MOVE INSTRUCTION
              ;          WHICH FOLLOWS W/ 771
              ;-----
006254 012762 000130 177776  MOV      #130,-2(R2)    ;MOVE TO MAILBOX # ***** 130 *****
006262 005262 177774          INC      -4(R2)
006266 000000          HALT
1549 006270 005160 177400  SOP6A: COM      -400(R0) ;SET MSGTYP TO FATAL ERROR
1550 006274 100003          BPL      SOP6B        ;CLR DID NOT SET Z-BIT
1551 006276 005260 177400  INC      -400(R0)      ;TRY TO COMPLEMENT LOCATION 0 W/MODE 6
1552 006302 001406          BEQ      TST66
              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ;          CONDITIONAL BRANCH INST. AND
              ;          REPLACE THE MOVE INSTRUCTION
              ;          WHICH FOLLOWS W/ 755
              ;=====
006304 012762 000131 177776  SOP6B: MOV      #131,-2(R2) ;MOVE TO MAILBOX # ***** 131 *****
006304 005262 177774          INC      -4(R2)
006312 000000          HALT
              ;SET MSGTYP TO FATAL ERROR
              ;TEST CUMMULATIVE RESULT OF ABOVE INSTS
              ; OR SEQUENCE ERROR

1553          :*****
1554          :
1555          :
1556          : THIS TEST VERIFIES MODE 7 SINGLE OPERAND INSTRUCTIONS. IT USFS
1557          : THE POINTER TO LOC. 0 WHICH IS STORED AT LOC. 402.
1558          : R0 IS SET TO 400 AND A MODE 7 CLR INSTRUCTION IS
1559          : EXECUTED WITH A +2 OFFSET TO CLEAR LOC. 0.
1560          : SEVERAL OTHER MODE 7 INSTRUCTIONS ARE THEN USED ON THE COMMON
1561          : LOCATION TO VERIFY THE DATA RESULTS.
1562          :
    
```

1563

.SBTTL TEST # 66 - TEST MODE 7 USING SOP INST.

 :TEST 66 - TEST MODE 7 USING SOP INST.

006320 005212
 006322 022712 000066
 006326 001025
 1564 006330 005000
 1565 006332 105100
 1566 006334 005200
 1567 006336 005210
 1568 006340 005070 000002
 1569 006344 001406

TST66: INC (R2) ;UPDATE TEST NUMBER
 CMP #66,(R2) ;SEQUENCE ERROR?
 BNE TST67-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;SET R0=400
 COMB R0
 INC R0
 INC (R0) ;R0=1
 CLR @2(R0) ;TRY TO CLEAR LOC. 0 W/MODE 7
 BEQ SOP7A

006346 012762 000132 177776
 006354 005262 177774
 006360 000000
 1570 006362 005170 000002
 1571 006366 100003
 1572 006370 005270 000002
 1573 006374 001406

MOV #132,-2(R2) ;MOVE TO MAILBOX # ***** 132 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLR DID NOT SET Z-BIT
 SOP7A: COM @2(R0) ;TRY TO COMPLEMENT LOC. 0 W/MODE 7
 BPL SOP7B
 INC @2(R0) ;TRY TO INCREMENT LOC. 0 W/MODE 7
 BEQ TST67

006376
 006376 012762 000133 177776
 006404 005262 177774
 006410 000000

SOP7B: MOV #133,-2(R2) ;MOVE TO MAILBOX # ***** 133 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;TEST CUMULATIVE RESULT OF ABOVE INSTS.
 ; OR SEQUENCE ERROR

1574

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--=
 : CONDITIONAL BRANCH INST. AND <==
 : REPLACE THE MOVE INSTRUCTION <--=
 : WHICH FOLLOWS W/ 770 <=
 :
 : TO SCOPE: CLEAR THE RIGHT BYIL OF THIS <=
 : CONDITIONAL BRANCH INST. AND <--=
 : REPLACE THE MOVE INSTRUCTION <--=
 : WHICH FOLLOWS W/ 754 <=

SBTTL TEST # 67 - TEST MODE 4 WITH NEGATE INSTRUCTION

TEST 67 - TEST MODE 4 WITH NEGATE INSTRUCTION

006412 005212
006414 022712 000067
006420 001032
1576 006422 005000
1577 006424 005010
1578 006426 005120
1579 006430 005440
1580 006432 100403
1581 006434 001402
1582 006436 102401
1583 006440 103406

ST67: INC (R2) ;UPDATE TEST NUMBER
CMP #67(R2) ;SEQUENCE ERROR?
BNE TST70-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0
CLR (R0)
COM (R0)+ ;LOC. 0=177777, R0=2
NEG -(R0) ;TRY NEGATE, LOC. 0=1
BMI NEG40 ;CC=0001?
BEQ NEG40
BVS NEG40
BCS NEG41

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 767 <====

006442
006442 012762 000134 177776
006450 005262 177774
006454 000000
1584 006456 005400
1585 006460 001406

NEG40: MOV #134,-2(R2)
INC -4(R2)
HALT
NEG41: NEG R0
BEQ NEG42

: MOVE TO MAILBOX # ***** 134 *****
: SET MSGTYP TO FATAL ERROR
: NEG DID NOT SET CC'S CORRECTLY
: TST R0 WITH A NEG.

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 757 <====

006462 012762 000135 177776
006470 005262 177774
006474 000000
1586 006476 005310
1587 006500 001406

NEG42: MOV #135,-2(R2)
INC -4(R2)
HALT
DEC (R0)
BEQ TST70

: MOVE TO MAILBOX # ***** 135 *****
: SET MSGTYP TO FATAL ERROR
: R0 NOT DECREMENTED PROPERLY
: TEST DTA RESULT OF NEG

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 747 <====

006502 012762 000136 177776
006510 005262 177774
006514 000000

MOV #136,-2(R2)
INC -4(R2)
HALT

: MOVE TO MAILBOX # ***** 136 *****
: SET MSGTYP TO FATAL ERROR
: DATA RESULT OF NEG INCORRECT
: OR SEQUENCE ERROR

HP

.SBTTL TEST # 70 - TEST MODE 5 WITH NEGATE INSTRUCTION

TEST 70 - TEST MODE 5 WITH NEGATE INSTRUCTION

006516 005212
 006520 022712 000070
 006524 001037
 1589 006526 005000
 1590 006530 005010
 1591 006532 105100
 1592 006534 005200
 1593 006536 005010
 1594 006540 005004
 1595 006542 005314
 1596 006544 005450
 1597 006546 100403
 1598 006550 001402
 1599 006552 102401
 1600 006554 103406

TST70: INC (R2) :UPDATE TEST NUMBER
 CMP #70 (R2) :SEQUENCE ERROR?
 BNE TST71-10 :BR TO ERROR HALT ON SEQ ERROR
 CLR R0 :R0=0
 CLR (R0) :LOC. 0=0
 COMB R0 :R0=377
 INC R0 :R0=400
 CLR (R0) :SET 400 = 0
 CLR R4 :R4=0
 DEC (R4) :LOC. 0=177777
 NEG @-(R0) :TRY NEGATE: LOC. 0 1
 BMI NEG50 :CC-0001?
 BEQ NEG50
 BVS NEG50
 BCS NEG51

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 763

006556
 006556 012762 000137 177776
 006564 005262 177774
 006570 000000
 1601 006572 005314
 1602 006574 001406

NEG50: MOV #137,-2(R2) :MOVE TO MAILBOX # ***** 137 *****
 INC -4(R2) :SET MSGTYP TO FATAL ERROR
 HALT :NEG DID NOT SET CC'S CORRECTLY
 NEG51: DEC (R4)
 BEQ NEG52

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 763

006576 012762 000140 177776
 006604 005262 177774
 006610 000000
 1603 006612 105100
 1604 006614 005300
 1605 006616 001406

NEG52: MOV #140,-2(R2) :MOVE TO MAILBOX # ***** 140 *****
 INC -4(R2) :SET MSGTYP TO FATAL ERROR
 HALT :DATA RESULT OF NEG INCORRECT
 COMB R0
 DEC R0
 BEQ TST71

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 742

006620 012762 000141 177776
 006626 005262 177774
 006632 000000

MOV #141,-2(R2) :MOVE TO MAILBOX # ***** 141 *****
 INC -4(R2) :SET MSGTYP TO FATAL ERROR
 HALT :REGISTER NOT DECREMENTED PROPERLY
 : OR SEQUENCE ERROR

1606

.SBTTL TEST # 71 - TEST MODE 6 WITH NEGATE

:TEST 71 - TEST MODE 6 WITH NEGATE

006634 005212
 006636 022712 000071
 006642 001026
 1607 006644 005000
 1608 006646 005004
 1609 006650 105100
 1610 006652 005014
 1611 006654 105024
 1612 006656 105114
 1613 006660 005460 177401
 1614 006664 100403
 1615 006666 001402
 1616 006670 102401
 1617 006672 103406

TST71: INC (R2) :UPDATE TEST NUMBER
 CMP #71,(R2) :SEQUENCE ERROR?
 BNE TST72-10 :BR TO ERROR HALT ON SEQ ERROR
 CLR R0 :R0=0
 CLR R4 :R4=0
 COMB R0 :R0=377
 CLR (R4) :LOC. 0=0
 CLRB (R4)+ :LOC. 0=177777, R4 1
 COMB (R4) :LOC. 0=177400
 NEG -377(R0) :LOC. 0=400
 BMI NEG60 :CC=0001
 BEQ NEG60
 BVS NEG60
 BCS NEG61

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 763 <

006674
 006674 012762 000142 177776
 006702 005262 177774
 006706 000000
 1618 006710 105314
 1619 006712 001406

NEG60: MOV #142,-2(R2)
 INC -4(R2)
 HALT
 NEG61: DECB (R4)
 BEQ TST72

:MOVE TO MAILBOX # ***** 142 *****
 :SET MSGTYP TO FATAL ERROR
 :NEG DID NOT SET CC'S CORRECTLY

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 753 <

006714 012762 000143 177776
 006722 005262 177774
 006726 000000

MOV #143,-2(R2)
 INC -4(R2)
 HALT

:MOVE TO MAILBOX # ***** 143 *****
 :SET MSGTYP TO FATAL ERROR
 :DATA RESULT OF NEG INCORRECT
 : OR SEQUENCE ERROR

.SBTTL TEST # 72 - TEST MODE 7 W/ NEGATE

.....
 :TEST 72 - TEST MODE 7 W/ NEGATE
 :.....

006730 005212
 006732 022712 000072
 006736 001030
 1621 006740 005000
 1622 006742 005010
 1623 006744 005110
 1624 006746 105100
 1625 006750 105470 000005
 1626 006754 100403
 1627 006756 001402
 1628 006760 102401
 1629 006762 103406

TST72: INC (R2) ;UPDATE TEST NUMBER
 CMP #72,(R2) ;SEQUENCE ERROR?
 BNE TST73-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 CLR (R0) ;LOC. 0=0
 COM (R0) ;LOC. 0=177777
 COMB R0 ;R0=377
 NEGB @5(R0) ;R0+5=404, 404=1, LOC. 0-777
 BMI NEG70 ;CC=0001?
 BEQ NEG70
 BVS NEG70
 BCS NEG71

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 765 <

006764
 006764 012762 000144 177776
 006772 005262 177774
 006776 000000
 1630 007000 105100
 1631 007002 105120
 1632 007004 105310
 1633 007006 005467 170766
 1634 007012 001406

NEG70: MOV #144,-2(R2) ;MOVE TO MAILBOX # ***** 144 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;NEG DID NOT SET CC'S CORRECTLY
 NEG71: COMB R0 ;R0=0
 COMB (R0)+ ;LOC. 0=400, R0=1
 DECB (R0) ;LOC. 0=0
 NEG 0 ;USE NEG MODE 67 TO TST FOR ZERO
 BEQ TST73

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 751 <

007014 012762 000145 177776
 007022 005262 177774
 007026 000000

MOV #145,-2(R2) ;MOVE TO MAILBOX # ***** 145 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DATA RESULT OF NEG WAS INCORRECT
 ; OR SEQUENCE ERROR

1635
 1636
 1637
 1638
 1639
 1640
 1641
 1642
 1643

.....
 : THIS TEST VERIFIES PROGRAM COUNTER ADDRESSING WITH SOP
 : INSTRUCTIONS. CLR MODE 77 IS USED TO CLEAR THE LOCATION FOLLOWING THE
 : INSTRUCTION (SOPX). THEN SINGLE OPERAND INSTRUCTIONS WITH MODES 37, 67, AND
 : 77, USING INDIRECT POINTER SOPXAD ARE USED TO VERIFY THE DATA RESULTS
 : OF THESE INSTRUCTIONS.
 :

1644

SBTTL TEST # 73 - TEST SOP OPCODE MODES 2,3,6,7 WITH P.C.

TEST 73 - TEST SOP OPCODE MODES 2,3,6,7 WITH P.C.

007030 005212
 007032 022712 000073
 007036 001021
 1645 007040 005027
 1646 007042 177777
 1647 007044 001406

TST73: INC (R2) ;UPDATE TEST NUMBER
 CMP #73,(R2) ;SEQUENCE ERROR?
 BNE SOPB ;BR TO ERROR HALT ON SEQ ERROR
 CLR (R7)+ ;CLEAR NEXT LOCATION: (SOPX)
 SOPX: -1 ;USE MODE 27
 BEQ SOPA

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 ; CONDITIONAL BRANCH INST. AND
 ; REPLACE THE MOVE INSTRUCTION
 ; WHICH FOLLOWS W/ 774

007046 012762 000146 177776
 007054 005262 177774
 007060 000000
 1648 007062 005237 007042
 1649 007066 005467 177750
 1650 007072 100003
 1651 007074 005277 000016
 1652 007100 001407

MOV #146,-2(R2) ;MOVE TO MAILBOX # ***** 146 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CLR DID NOT SET Z-BIT
 SOPA: INC @SOPX ;INC SOPX W/MODE 37
 NEG SOPX ;NEGATE SOPX W/MODE 67
 BPL SOPB
 INC @SOPXAD ;INC SOPX W/MODE 77
 BEQ TST74

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 ; CONDITIONAL BRANCH INST. AND
 ; REPLACE THE MOVE INSTRUCTION
 ; WHICH FOLLOWS W/ 756

007102
 007102 012762 000147 177776
 007110 005262 177774
 007114 000000

SOPB: MOV #147,-2(R2) ;MOVE TO MAILBOX # ***** 147 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;INC DID NOT SET Z-BIT
 ; OR SEQUENCE ERROR
 SOPXAD: SOPX ;INDIRECT ADDRESS OF SOPX

1653 007116 007042
 1654
 1655
 1656
 1657
 1658
 1659
 1660
 1661
 1662

 ; THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING INSTRUCTIONS
 ; USING MODE 0. R0 IS SET TO ZERO AND THE CONDITION CODES ARE SET
 ; TO THE COMPLEMENT OF THAT EXPECTED BY THE INSTRUCTION. A TST INSTRUCTION
 ; IS EXECUTED AND CONDITIONAL BRANCHES ARE USED TO TEST THE CONDITION
 ; CODES.
 ;

1663

.SBTTL TEST # 74 - TEST MODE 0 SOP NON-MODIFYING

.....
 :TEST 74 - TEST MODE 0 SOP NON-MODIFYING
 :.....

007120 005212
 007122 022712 000074
 007126 001012
 1664 007130 005000
 1665 007132 000277
 1666 007134 000244
 1667 007136 005700
 1668 007140 102403
 1669 007142 100402
 1670 007144 103401
 1671 007146 001406

TST74: INC (R2) ;UPDATE TEST NUMBER
 CMP #74,(R2) ;SEQUENCE ERROR?
 BNE TST75-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;INITIALIZE R0=0
 SCC ;SET CC=1011
 CLZ
 TST R0 ;TRY TST W/ MODE 0
 BVS SNMOA ;CHECK THAT CC=0100
 BMI SNMOA
 BCS SNMOA
 BEQ TST75

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 767 <---

007150
 007150 012762 000150 177776
 007156 005262 177774
 007162 000600

SNMOA: MOV #150,-2(R2) ;MOVE TO MAILBOX # ***** 150 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CONDITION CODES NOT SET PROPERLY
 ; OR SEQUENCE ERROR

1672
 1673
 1674
 1675
 1676
 1677
 1678
 1679
 1680

.....
 : THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING BYTE INSTRUCTIONS WITH MODE 0.
 : R0 IS SET TO 377 AND COMPLEMENT OF THE EXPECTED CONDITION CODES
 : IS LOADED IN PSW. A TSTB INSTRUCTION IS EXECUTED AND THE RESULTS
 : ARE CHECKED WITH SEVERAL CONDITIONAL BRANCH INSTRUCTIONS.
 : THIS VERIFIES THAT THE PROPER BYTE WAS TESTED.
 :

1681

.SBTTL TEST # 75 - TEST MODE 0 EVEN BYTE W/ SOP NON-MODIFYING
 :.....
 :TEST 75 - TEST MODE 0 EVEN BYTE W/ SOP NON-MODIFYING
 :.....

007164 005212
 007166 022712 000075
 007172 001012
 1682 007174 005000
 1683 007176 105100
 1684 007200 000277
 1685 007202 000250
 1686 007204 105700
 1687 007206 102402
 1688 007210 101401
 1689 007212 100406

TST75: INC (R2) ;UPDATE TEST NUMBER
 CMP #75,(R2) ;SEQUENCE ERROR?
 BNE TST76-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;INITIALIZE
 COMB R0 ;R0=377
 SCC ;SET CC=0111
 CLN
 TSTB R0 ;TRY TST EVEN BYTE
 BVS SNMBOA ;CHECK CC=1000
 BLOS SNMBOA
 BMI TST76
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 ; CONDITIONAL BRANCH INST. AND <=====
 ; REPLACE THE MOVE INSTRUCTION <=====
 ; WHICH FOLLOWS W/ 767 <=====
 ;

007214
 007214 012762 000151 177776
 007222 005262 177774
 007226 000000

SNMBOA: MOV #151,-2(R2) ;MOVE TO MAILBOX # ***** 151 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CONDITION CODES NOT SET PROPERLY
 ; OR SEQUENCE ERROR

1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698

:.....
 : THIS TEST VERIFIES SINGLE OPERAND INSTRUCTIONS WITH MODE 1.
 : RO IS USED TO POINT TO AND CLEAR LOC. 0. THE COMPLEMENT OF THE
 : EXPECTED CONDITION CODES ARE LOADED IN THE PSW. A TST INSTRUCTION
 : IS THEN EXECUTED ON LOC. 0 USING RO AND CONDITIONAL BRANCHES TEST
 : THE RESULTS.
 :.....

1709

.SBTTL TEST # 76 - TEST MODE 1 SOP NON-MODIFYING
:.....
:TEST 76 - TEST MODE 1 SOP NON-MODIFYING
:.....

007230 005212
007232 022712 000076
007236 001013
1700 007240 005000
1701 007242 005010
1702 007244 000277
1703 007246 000244
1704 007250 005710
1705 007252 102403
1706 007254 103402
1707 007256 100401
1708 007260 001406

TST76: INC (R2) ;UPDATE TEST NUMBER
CMP #76,(R2) ;SEQUENCE ERROR?
BNE TST77-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;POINT TO LOC 0
CLR (R0) ;CLEAR LOC 0
SCC ;INITIALIZE
CLZ ;CC=1011
TST (R0) ;TRY TST W/ MODE 1
BVS SNM1A ;CHECK CC=0100
BCS SNM1A
BMI SNM1A
BEQ TST77

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <---
: WHICH FOLLOWS W/ 766 <---

007262
007262 012762 000152 177776
007270 005262 177774
007274 000000

SNM1A: MOV #152,-2(R2) ;MOVE TO MAILBOX # ***** 152 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT SET PROPERLY
; OR SEQUENCE ERROR

1709
1710
1711
1712
1713
1714
1715
1716

:.....
: THIS TEST SETS LOCATION 0 TO 377 AND THEN USES R0 TO TEST
: THE EVEN BYTE AND THE ODD BYTE USING SOP BYTE INSTRUCTIONS WITH MODE 1.
: AGAIN, CONDITIONAL BRANCHES ARE USED TO VERIFY THE SETTING OF THE
: PROPER CONDITION CODE BITS.
:.....


```

1744 .SBTTL TEST # 100 - TEST MODE 2 WITH SOP NON-MODIFYING
:*****
:TEST 100 - TEST MODE 2 WITH SOP NON-MODIFYING
:*****
007402 005212          000100 TST100: INC      (R2)          ;UPDATE TEST NUMBER
007404 022712          000100          CMP      #100,(R2)        ;SEQUENCE ERROR?
007410 001024          000100          BNE     TST101-10       ;BR TO ERROR HALT ON SEQ ERROR
1745 007412 005000          000100          CLR     R0              ;INITIALIZE R0=0
1746 007414 005010          000100          CLR     (R0)           ;CLEAR LOC 0
1747 007416 000277          000100          SCC     ;SET CC=1011
1748 007420 000244          000100          CLZ     ;
1749 007422 005720          000100          TST     (R0)+          ;TRY TST W/ MODE 2
1750 007424 102403          000100          BVS     SNM2A          ;CHECK CC=0100
1751 007426 103402          000100          BCS     SNM2A
1752 007430 100401          000100          BMI     SNM2A
1753 007432 001406          000100          BEQ     SNM2B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
; CONDITIONAL BRANCH INST. AND <===
; REPLACE THE MOVE INSTRUCTION <===
; WHICH FOLLOWS W/ 766 <===

007434 012762 000155 177776 SNM2A: MOV      #155,-2(R2)    ;MOVE TO MAILBOX # ***** 155 *****
007434 012762 000155 177776          INC      -4(R2)        ;SET MSGTYP TO FATAL ERROR
007442 005262 177774          000155 177776          HALT     ;CC'S NOT CORRECT
1754 007450 005300          000155 177776 SNM2B: DEC      R0              ;RESET R0
1755 007452 005300          000155 177776          DEC      R0
1756 007454 001406          000155 177776          BEQ     TST101

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
; CONDITIONAL BRANCH INST. AND <===
; REPLACE THE MOVE INSTRUCTION <===
; WHICH FOLLOWS W/ 755 <===

007456 012762 000156 177776          000156 177776          MOV      #156,-2(R2)    ;MOVE TO MAILBOX # ***** 156 *****
007464 005262 177774          000156 177776          INC      -4(R2)        ;SET MSGTYP TO FATAL ERROR
007470 000000          000156 177776          HALT     ;MODE 2 DID NOT INC REQ CORRECTLY
; OR SEQUENCE ERROR

1757
1758
1759
1760
1761
1762
1763
1764
1765
:*****
: THIS TEST VERIFIES MODE 2 SINGLE OPERAND NON-MODIFYING BYTE
: INSTRUCTIONS IT USES R0 TO POINT TO LOC. 0. WITH LOCATION 0
: SET TO 377, THE EVEN AND ODD BYTE IS TESTED WITH TSTB INSTRUCTIONS
: TO VERIFY THE CORRECT CC ARE SET. THE REGISTER IS CHECKED FOR
: PROPER INCREMENTING.
:

```

```
1766 .SBTTL TEST # 101 - TEST MODE 2 - BYTE W/ SOP NON-MODIFYING
:*****
:TEST 101 - TEST MODE 2 - BYTE W/ SOP NON-MODIFYING
:*****
007472 005212          TST101: INC      (R2)          ;UPDATE TEST NUMBER
007474 022712 000101  CMP      #101,(R2)        ;SEQUENCE ERROR?
007500 001052          BNE      TST102-10       ;BR TO ERROR HALT ON SEQ ERROR
1767 007502 005000     CLR      R0              ;CLEAR R0
1768 007504 005010     CLR      (R0)           ;CLEAR LOC 0
1769 007506 105110     COMB    (R0)           ;SET LOC 0=377
1770 007510 000277     SCC              ;SET CC=0111
1771 007512 000250     CLN              ;
1772 007514 105720     TSTB    (R0)+          ;TRY TST OF EVEN BYTE
1773 007516 102402     BVS     SNMB2A         ;
1774 007520 101401     BLOS    SNMB2A         ;
1775 007522 100406     BMI     SNMB2B         ;
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 766 <====
:
007524          SNMB2A:
007524 012762 000157 177776  MOV     #157,-2(R2)    ;MOVE TO MAILBOX # ***** 157 *****
007532 005262 177774  INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
007536 000000          HALT              ;CC'S NOT SET CORRECTLY
1776 007540 005300     SNMB2B: DEC     R0      ;DECREMENT R0
1777 007542 001406     BEQ     SNMB2C        ;
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
: CONDITIONAL BRANCH INST. AND <= --
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 756 <====
:
007544 012762 000160 177776  MOV     #160,-2(R2)    ;MOVE TO MAILBOX # ***** 160 *****
007552 005262 177774  INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
007556 000000          HALT              ;MODE 2 DID NOT INC REG CORRECTLY
1778 007560 005200     SNMB2C: INC     R0      ;POINT TO ODD BYTE
1779 007562 000277     SCC              ;SET CC=1011
1780 007564 000244     CLZ              ;
1781 007566 105720     TSTB    (R0)+          ;TRY TST OF ODD BYTE
1782 007570 102403     BVS     SNMB2D         ;CHECK CC'S=0100
1783 007572 103402     BCS     SNMB2D         ;
1784 007574 100401     BMI     SNMB2D         ;
1785 007576 001406     BEQ     SNMB2E         ;
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <==
: CONDITIONAL BRANCH INST. AND <=-
: REPLACE THE MOVE INSTRUCTION <- -
: WHICH FOLLOWS W/ 740 < =-
:
007600          SNMB2D:
007600 012762 000161 177776  MOV     #161,-2(R2)    ;MOVE TO MAILBOX # ***** 161 *****
007606 005262 177774  INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
007612 000000          HALT              ;CC'S NOT CORRECT
1786 007614 005300     SNMB2E: DEC     R0      ;
1787 007616 005300     DEC     R0            ;
1788 007620 001406     BEQ     TST102        ;
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
: CONDITIONAL BRANCH INST. AND <-
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 727 <
:
C07622 012762 000162 177776  MOV     #162,-2(R2)    ;MOVE TO MAILBOX # ***** 162 *****
```

007630 005262 177774
007634 000000

INC -4(R2)
HALT

:SET MSGTYP TO FATAL ERROR
:RO DID NOT INCREMENT PROPERLY
: OR SEQUENCE ERROR

1789
1790
1791
1792
1793
1794
1795
1796

.....
: THIS TEST VERIFIES MODE 3 SINGLE OPERAND NON-MODIFYING INSTRUCTIONS.
: A POINTER IN A TABLE AT LOC. 376 IS USED TO TEST LOCATION 0.
: THE CC'S AND THE REGISTER ARE CHECKED FOLLOWING THE
: TST MODE 3 INSTRUCTION.
:

1797

.SBTTL TEST # 102 - TEST MODE 3 W/ SOP NON-MODIFYING INSTS

 :TEST 102 - TEST MODE 3 W/ SOP NON-MODIFYING INSTS

007636 005212
 007640 022712 000102
 007644 001026
 1798 007646 005000
 1799 007650 005010
 1800 007652 105100
 1801 007654 005300
 1802 007656 000277
 1803 007660 000244
 1804 007662 005730
 1805 007664 102403
 1806 007666 103402
 1807 007670 100401
 1808 007672 001406

TST102: INC (R2) ;UPDATE TEST NUMBER
 CMP #102,(R2) ;SEQUENCE ERROR?
 BNE TST103-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 CLR (R0) ;CLEAR LOC 0
 COMB R0 ;R0=376
 DEC R0
 SCC ;SET CC=1011
 CLZ
 TST @(R0)+ ;TRY TST W/ MODE 3
 BVS SNM3A ;CHECK CC=0100
 BCS SNM3A
 BMI SNM3A
 BEQ SNM3B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < ---
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 764 <====

007674
 007674 012762 000163 177776
 007702 005262 177774
 007706 000000
 1809 007710 005300
 1810 007712 105100
 1811 007714 001406

SNM3A: MOV #163,-2(R2) ;MOVE TO MAILBOX # ***** 163 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CC'S NOT CORRECT
 SNM3B: DEC R0 ;R0=377
 COMB R0 ;R0=0
 BEQ TST103

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 753 <====

007716 012762 000164 177776
 007724 005262 177774
 007730 000000

MOV #164,-2(R2) ;MOVE TO MAILBOX # ***** 164 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;MODE 3 DID NOT INC REG CORRECTLY
 ; OR SEQUENCE ERROR

1812
 1813
 1814
 1815
 1816
 1817
 1818
 1819
 1820
 1821

 : THIS TEST VERIFIES SOP NON-MODIFYING BYTE INSTRUCTIONS MODE 3
 : LOC. 0 IS SET TO 377. TABLE AT LOC. 402-404 IS USED TO TEST
 : BYTE 0 AND BYTE 1. THE REGISTER IS CHECKED FOR PROPER INCREMENTING AND
 : THE CC'S ARE VERIFIED.
 : THE TABLE AT LOC. 402-404 SHOULD CONTAIN 0 AND 1 BEFORE AND
 : AFTER THE TEST IS RUN.
 :

1822

.SBTTL TEST # 103 - TEST MODE 3 - BYTES W/ SOP NON-MODIFYING INSTS.

:TEST 103 - TEST MODE 3 - BYTES W/ SOP NON-MODIFYING INSTS.

007732 005212
007734 022712 000103
007740 001044
1823 007742 005000
1824 007744 005010
1825 007746 105110
1826 007750 105100
1827 007752 005200
1828 007754 005720
1829 007756 000277
1830 007760 000250
1831 007762 105730
1832 007764 102402
1833 007766 101401
1834 007770 100406

TST103: INC (R2) ;UPDATE TEST NUMBER
CMP #103,(R2) ;SEQUENCE ERROR?
BNE TST104-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;CLEAR LOC 0
COMB (R0) ;LOC. 0 =377
COMB R0
INC R0
TST (R0)+ ;R0=402
SCC ;CC=0111
CLN
TSTB @ (R0)+ ;TRY TST OF EVEN BYTE
BVS SNMB3A ;CHECK CC=1000
BLOS SNMB3A
BMI SNMB3B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 763

007772
007772 012762 000165 177776
010000 005262 177774
010004 000000
1835 010006 000277
1836 010010 000244
1837 010012 105730
1838 010014 102403
1839 010016 103402
1840 010020 100401
1841 010022 001406

SNMB3A: MOV #165,-2(R2) ;MOVE TO MAILBOX # ***** 165 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT
SNMB3B: SCC ;SET CC=1011
CLZ
TSTB @ (R0)+ ;TRY TST OF ODD BYTE
BVS SNMB3C ;CHECK CC=0100
BCS SNMB3C
BMI SNMB3C
BEQ SNMB3D

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 746

010024
010024 012762 000166 177776
010032 005262 177774
010036 000000
1842 010040 005720
1843 010042 005710
1844 010044 100406

SNMB3C: MOV #166,-2(R2) ;MOVE TO MAILBOX # ***** 166 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT
SNMB3D: TST (R0)+ ;R0=410
TST (R0)
BMI TST104

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 735

010046 012762 000167 177776
010054 005262 177774
010060 000000

MOV #167,-2(R2) ;MOVE TO MAILBOX # ***** 167 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TSTB DID NOT INCREMENT R0 CORRECTLY
; OR SEQUENCE ERROR

1845
1846
1847
1848

:*****
: THIS TEST VERIFIES MODE 4 SOP NON-MODIFYING INSTRUCTIONS.
: LOC. 0 IS SET TO -1 AND THE CC'S ARE SET TO THE COMPLEMENT OF THE

1849
1850
1851
1852

: EXPECTED RESULTS. RO AND SET TO 2 AND A TST MODE 4 IS EXECUTED.
: THE CC'S ARE CHECKED WITH CONDITIONAL BRANCH INSTRUCTIONS AND THE REGISTER
: IS CHECKED FOR PROPER DECREMENTING.
:

1853

.SBTTL TEST # 104 - TEST MODE 4 W/ SOP NON-MODIFYING INSTS

:TEST 104 - TEST MODE 4 W/ SOP NON-MODIFYING INSTS

010062 005212
010064 022712 000104
010070 001023
1854 010072 005000
1855 010074 005010
1856 010076 005120
1857 010100 000277
1858 010102 000244
1859 010104 005740
1860 010106 102402
1861 010110 101401
1862 010112 100406

TST104: INC (R2) ;UPDATE TEST NUMBER
CMP #104,(R2) ;SEQUENCE ERROR?
BNE TST105-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC 0=0
COM (R0)+ ;LOC 0=-1
SCC ;SET CC=1011
CLZ
TST -(R0) ;TRY TST W/ MODE 4
BVS SNM4A ;CHECK CC=0100
BLOS SNM4A
BMI SNM4B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 766

010114
010114 012762 000170 177776
010122 005262 177774
010126 000000
1863 010130 005700
1864 010132 001406

SNM4A: MOV #170,-2(R2) ;MOVE TO MAILBOX # ***** 170 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT
SNM4B: TST R0
BEQ TST105

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 756

010134 012762 000171 177776
010142 005262 177774
010146 000000

MOV #171,-2(R2) ;MOVE TO MAILBOX # ***** 171 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TST MODE 4 DID NOT DEC R0 CORRECTLY
; OR SEQUENCE ERROR

1865
1866
1867
1868
1869
1870
1871
1872

.....
: THIS TEST VERIFIES MODE 5 SOP NON-MODIFYING INSTRUCTIONS.
: IT USES A POINTER AT LOC. 376 TO TEST LOC. 0. R0 IS SET
: TO 400. A TST MODE 5 INSTRUCTION IS EXECUTED AND THE CC'S CHECKED.
: R0 IS CHECKED TO INSURE PROPER DECREMENTING.
:

1873

.SBTTL TEST # 105 - TEST MODE 5 W/ SOP NON-MODIFYING INSTS

 :TEST 105 - TEST MODE 5 W/ SOP NON-MODIFYING INSTS

010150 005212
 010152 022712 000105
 010156 001026
 1874 010160 005000
 1875 010162 005010
 1876 010164 005110
 1877 010166 105100
 1878 010170 005200
 1879 010172 000277
 1880 010174 000250
 1881 010176 005750
 1882 010200 102402
 1883 010202 101401
 1884 010204 100406

TST105: INC (R2) ;UPDATE TEST NUMBER
 CMP #105,(R2) ;SEQUENCE ERROR?
 BNE TST106-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 CLR (R0) ;LOC 0=0
 COM (R0) ;LOC 0=-1
 COMB R0 ;R0=377
 INC R0 ;R0=400
 SCC ;SET CC=0111
 CLN
 TST @-(R0) ;TRY TST W/ MODE 5
 BVS SNM5A ;CHECK CC=1000
 BLOS SNM5A
 BMI SNM5B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <----
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 764 <----

010206
 010206 012762 000172 177776
 010214 005262 177774
 010220 000000
 1885 010222 005200
 1886 010224 105100
 1887 010226 001406

SNM5A: MOV #172,-2(R2) ;MOVE TO MAILBOX # ***** 172 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CC'S NOT SET PROPERLY
 SNM5B: INC R0 ;R0=377
 COMB R0 ;R0=0
 BEQ TST106

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <----
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 753 <----

010230 012762 000173 177776
 010236 005262 177774
 010242 000000

MOV #173,-2(R2) ;MOVE TO MAILBOX # ***** 173 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;MODE 5 DID NOT DEC R0 CORRECTLY
 ; OR SEQUENCE ERROR

1888
 1889
 1890
 1891
 1892
 1893
 1894
 1895

 : THIS TEST VERIFIFS MODE 6 SOP NON-MODIFYING INSTRUCTIONS.
 : RO IS SET TO 377 AND A MODE 6 TST INSTRUCTION IS EXECUTED
 : USING RO AND AN OFFSET OF -377. THE CC'S ARE CHECKED AS WELL
 : AS RO TO INSURE IT WAS NOT ALTERED.
 :

1896

SBTTL TEST # 106 - TEST MODE 6 W/ SOP NON-MODIFYING INSTS

TEST 106 - TEST MODE 6 W/ SOP NON-MODIFYING INSTS

010244 005212
010246 022712 000106
010252 001025
1897 010254 005000
1898 010256 005010
1899 010260 005110
1900 010262 105100
1901 010264 000277
1902 010266 000250
1903 010270 005760 177401
1904 010274 102402
1905 010276 101401
1906 010300 100406

ST106: INC (R2) ;UPDATE TEST NUMBER
CMP #106,(R2) ;SEQUENCE ERROR?
BNE TST107-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR RO ;RO=0
CLR (R0) ;LOC 0=0
COM (R0) ;LOC 0=-1
COMB RO ;RO=377
SCC ;SET CC=0111
CLN
TST -377(R0) ;TRY TST W/ MODE 6
BVS SNM6A ;CHECK CC=1000
BLOS SNM6A
BMI SNM6B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 764 <---

010302
010302 012762 000174 177776
010310 005262 177774
010314 000000
1907 010316 105100
1908 010320 001406

SNM6A: MOV #174,-2(R2) ;MOVE TO MAILBOX # ***** 174 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S INCORRECT
SNM6B: COMB RO ;RO=0
BEQ TST107

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 754 <

010322 012762 000175 177776
010330 005262 177774
010334 000000

MOV #175,-2(R2) ;MOVE TO MAILBOX # ***** 175 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TST MODE 6 INCORRECTLY CHANGED RO
; OR SEQUENCE ERROR

1909
1910
1911
1912
1913
1914
1915
1916

.....
: THIS TEST VERIFIES MODE 7 SOP NON-MODIFYING INSTRUCTIONS.
: IT USES A POINTER TO LOC. 0 STORED AT LOC. 400 TO TST LOC. 0.
: RO IS SET TO 377 AND LOC. 0 IS TESTED THRU THE POINTER AT 400 USING
: RO AND AN OFFSET OF 1.
:

.....
:SBTTL TEST # 107 - TEST MODE 7 W/ SOP NON-MODIFYING INSTS.
:.....

:TEST 107 - TEST MODE 7 W/ SOP NON-MODIFYING INSTS.
:.....

010336 005212
010340 022712 000107
010344 001025
1918 010346 005000
1919 010350 005010
1920 010352 005110
1921 010354 105100
1922 010356 000277
1923 010360 000250
1924 010362 005770 000001
1925 010366 102402
1926 010370 101401
1927 010372 100406

TST107: INC (R2) ;UPDATE TEST NUMBER
CMP #107,(R2) ;SEQUENCE ERROR?
BNE TST110-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC 0=0
COM (R0) ;LOC 0=-1
COMB R0 ;R0=377
SCC ;CC=0111
CLN
TST @1(R0) ;TRY TST W/ MODE 7
BVS SNM7A ;CHECK CC=1000
BLOS SNM7A
BMI SNM7B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 764 <====

010374
010374 012762 000176 177776
010402 005262 177774
010406 000000
1928 010410 105100
1929 010412 001406

SNM7A: MOV #176,-2(R2) ;MOVE TO MAILBOX # ***** 176 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT
SNM7B: COMB R0 ;R0=0
BEQ TST110

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 754 <====

010414 012762 000177 177776
010422 005262 177774
010426 000000

MOV #177,-2(R2) ;MOVE TO MAILBOX # ***** 177 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TST MODE 7 INCORRECTLY CHANGED R0
; OR SEQUENCE ERROR

1930
1931
1932
1933
1934
1935
1936

.....
: THIS TEST VERIFIES MODE 0 DOUBLE OPERAND INSTRUCTIONS. IT SETS
: DATA IN R0 AND R4 AND USES THE ADD INSTRUCTION TO TEST THE DOP
: MICROCODE.
:.....

.SBTTL TEST # 110 - TEST MODE 0 DOUBLE-OPERAND (DOP) INSTS.

:TEST 110 - TEST MODE 0 DOUBLE-OPERAND (DOP) INSTS.

010430 005212
010432 022712 000110
010436 001010
1938 010440 005000
1939 010442 005100
1940 010444 005004
1941 010446 060004
1942 010450 005204
1943 010452 001406

TST110: INC (R2) ;UPDATE TEST NUMBER
CMP #110,(R2) ;SEQUENCE ERROR?
BNE TST111-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
COM R0 ;R0=-1
CLR R4 ;R4=0
ADD R0,R4 ;TRY ADD: R4=-1
INC R4 ;R4=0
BEQ TST111

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 771
: MOVE TO MAILBOX # ***** 200 *****
: SET MSGTYP TO FATAL ERROR
: ADD INST. FAILED W/ MODE 0
: OR SEQUENCE ERROR

010454 012762 000200 177776
010462 005262 177774
010466 000000

MOV #200,-2(R2)
INC -4(R2)
HALT

1944
1945
1946
1947
1948
1949
1950

.....
: THIS TEST VERIFIES THE MOVE INSTRUCTION WITH MODE 0 TO MODE 0.
: THIS TEST IS NECESSARY BECAUSE THIS PARTICULAR INSTRUCTION UTILIZES UNIQUE
: MICROCODE.
:

1951

SBTTL TEST # 111 - MOV MODE 0 TO MODE 0

TEST 111 - MOV MODE 0 TO MODE 0

010470	005212		
010472	022712	000111	
010476	001010		
1952 010500	005000		
1953 010502	005004		
1954 010504	005100		
1955 010506	010004		
1956 010510	005204		
1957 010512	001406		

```

TST111: INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #111,(R2)      ;SEQUENCE ERROR?
        BNE     TST112-10      ;BR TO ERROR HALT ON SEQ ERROR
        CLR     R0              ;R0=0
        CLR     R4              ;R4=0
        COM     R0              ;R0=-1
        MOV     R0,R4           ;TRY MOVE -1 TO R4
        INC     R4              ;INC R4
        BEQ     TST112

```

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
; CONDITIONAL BRANCH INST. AND <===
; REPLACE THE MOVE INSTRUCTION <===
; WHICH FOLLOWS W/ 771 <===
; MOVE TO MAILBOX # ***** 201 *****
; SET MSGTYP TO FATAL ERROR
; MOVE FAILED MODE 0 TO MODE 0
; OR SEQUENCE ERROR

```

010514	012762	000201	177776
010522	005262	177774	
010526	000000		

```

MOV     #201,-2(R2)
INC     -4(R2)
HALT

```

1958
1959
1960
1961
1962
1963
1964

```

.....
THIS TEST VERIFIES THE SUBTRACT INSTRUCTION WITH MODE 0,0.
THIS TEST IS NECESSARY BECAUSE THIS PARTICULAR INSTRUCTION UTILIZES SOME
UNIQUE MICROCODE.
.....

```

```

:*****
:SBYTL TEST # 112 - TEST SUB MODE 0,0
:*****
:TEST 112 - TEST SUB MODE 0,0
:*****
1965
010530 005212
010532 022712 000112
010536 001022
1966 010540 005000
1967 010542 005004
1968 010544 005204
1969 010546 160400
1970 010550 100003
1971 010552 001402
1972 010554 102401
1973 010556 103406

TST112: INC (R2) ;UPDATE TEST NUMBER
        CMP #112,(R2) ;SEQUENCE ERROR?
        BNE TST113-10 ;BR TO ERROR HALT ON SEQ ERROR
        CLR R0 ;R0=0
        CLR R4 ;R4=0
        INC R4 ;R4=1
        SUB R4,R0 ;TRY SUB 0,0 R0--1
        BPL SUB0 ;CC=1001
        BEQ SUB0
        BVS SUB0
        BCS SUB0A

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 767 <====

010560
010560 012762 000202 177776 SUB0: MOV #202,-2(R2) ;MOVE TO MAILBOX # ***** 202 *****
010566 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
010572 000000 HALT ;CONDITION CODE FAILED ON SUB
1974 010574 005200 SUB0A: INC R0
1975 010576 001406 BEQ TST113

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 757 <====

010600 012762 000203 177776 MOV #203,-2(R2) ;MOVE TO MAILBOX # ***** 203 *****
010606 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
010612 000000 HALT ;DATA RESULT OF SUB FAILED
; OR SEQUENCE ERROR

:*****
: THIS TEST QUICKLY VERIFIES THE REMAINING DOP MODIFYING INSTRUCTIONS
: WITH MODE 0,0 TO PROVIDE A BASELINE FOR SUBSEQUENT TESTS.
: SINGLE OPERAND INSTRUCTIONS ARE USED TO SET UP DATA IN R0 AND R4
: BEFORE EACH OF THE SEVERAL DOP MODIFYING INSTRUCTIONS ARE USED AND
: VERIFIED.
:
1976
1977
1978
1979
1980
1981
1982
1983
1984
```

```
1985          .SBTTL TEST # 113 - TEST ALL THE DOP INSTRUCTIONS W/ SOURCE MODE 0,0
              :*****
              :TEST 113 - TEST ALL THE DOP INSTRUCTIONS W/ SOURCE MODE 0,0
              :*****
010614 005212          TST113: INC      (R2)          ;UPDATE TEST NUMBER
010616 022712 000113  CMP      #113,(R2)        ;SEQUENCE ERROR?
010622 001063          BNE      TST114-10        ;BR TO ERROR HALT ON SEQ ERROR
1986 010624 005000          CLR      R0          ;R0=0
1987 010626 010004          MOV      R0,R4        ;TRY MOVE MODE 0,0
1988 010630 001406          BEQ      DOP0A        ;
              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 774 <====
010632 012762 000204 177776  MOV      #204,-2(R2)    ;MOVE TO MAILBOX # ***** 204 *****
010640 005262 177774          INC      -4(R2)        ;SET MSGTYP TO FATAL ERROR
010644 000000          HALT                    ;Z-BIT NOT SET
1989 010646 005200          DOPOA: INC      R0          ;R0=1
1990 010650 005100          COM      R0          ;R0=177776
1991 010652 005104          COM      R4          ;R4=177777
1992 010654 040004          BIC      R0,R4        ;TRY BIC: R4=1
1993 010656 005304          DEC      R4          ;R4=0
1994 010660 001406          BEQ      DOP0B        ;
              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 760 <====
010662 012762 000205 177776  MOV      #205,-2(R2)    ;MOVE TO MAILBOX # ***** 205 *****
010670 005262 177774          INC      -4(R2)        ;SET MSGTYP TO FATAL ERROR
010674 000000          HALT                    ;BIC CLEAR RESULT INCORRECT
1995 010676 050004          DOPOB: BIS      R0,R4    ;TRY BIS: R4=177777
1996 010700 005204          INC      R4
1997 010702 005204          INC      R4          ;R4=0
1998 010704 001406          BEQ      DOP0C        ;
              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 746 <====
010706 012762 000206 177776  MOV      #206,-2(R2)    ;MOVE TO MAILBOX # ***** 206 *****
010714 005262 177774          INC      -4(R2)        ;SET MSGTYP TO FATAL ERROR
010720 000000          HALT                    ;RESULT OF BIS INCORRECT
1999 010722 005000          DOP0C: CLR      R0          ;R0=0
2000 010724 105100          COMB     R0          ;R0=377
2001 010726 005004          CLR      R4          ;R4=0
2002 010730 005104          COM      R4          ;R4=177777
2003 010732 040004          BIC      R0,R4        ;R4=177400
2004 010734 060004          ADD      R0,R4        ;TRY ADD: R4=177777
2005 010736 005204          INC      R4          ;R4=0
2006 010740 001406          BEQ      DOP0D        ;
              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 730 <====
010742 012762 000207 177776  MOV      #207,-2(R2)    ;MOVE TO MAILBOX # ***** 207 *****
010750 005262 177774          INC      -4(R2)        ;SET MSGTYP TO FATAL ERROR
010754 000000          HALT                    ;RESULT OF ADD INCORRECT
2007 010756 160004          DOP0D: SUB      R0,R4    ;177401=R4
```

```
2008 010760 105404          NEGB   R4          :R4=177777
2009 010762 005204          INC    R4          :RD=0
2010 010764 001406          BEQ    TST114      :
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
:          CONDITIONAL BRANCH INST. AND <===
:          REPLACE THE MOVE INSTRUCTION <===
:          WHICH FOLLOWS W/ 716 <===
010766 012762 000210 177776  MOV    #210,-2(R2) :MOVE TO MAILBOX # ***** 210 *****
010774 005262 177774          INC    -4(R2)      :SET MSGTYP TO FATAL ERROR
011000 000C00          HALT                :RESULT OF SUB INCORRECT
: OR SEQUENCE ERROR
```

2011
2012
2013
2014
2015
2016

.....
: THIS TEST VERIFIES MODE 0,X DOUBLE OPERAND INSTRUCTIONS. IT SETS
: DATA IN R0 AND LOCATION 0 AND OPERATES UPON IT USING DOP INSTRUCTIONS.
:

2017

.SBTTL TEST # 114 - TEST MODE 0,X DOUBLE-OPERAND INSTRUCTIONS

:TEST 114 - TEST MODE 0,X DOUBLE-OPERAND INSTRUCTIONS

011002 005212
011004 022712 000114
011010 001030
2018 011012 005000
2019 011014 005010
2020 011016 105110
2021 011020 005220
2022 011022 005400
2023 011024 060037 000000
2024 011030 100403
2025 011032 001402
2026 011034 102401
2027 011036 103406

TST114: INC (R2) ;UPDATE TEST NUMBER
CMP #114,(R2) ;SEQUENCE ERROR?
BNE TST115-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
COMB (R0) ;LOC. 0=377
INC (R0)+ ;LOC. 0=400 R0=2
NEG R0 ;R0=-2
ADD R0,@#0 ;TRY ADD 0,3; LOC. 0=376
BMI DOP03A ;CC=0001?
BEQ DOP03A
BVS DOP03A
BCS DOP03B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
CONDITIONAL BRANCH INST. AND <====
REPLACE THE MOVE INSTRUCTION <====
WHICH FOLLOWS W/ 764 <====

011040
011040 012762 000211 177776
011046 005262 177774
011052 000000
2028 011054 105137 000000
2029 011060 005337 000000
2030 011064 001406

DOP03A: MOV #211,-2(R2) ;MOVE TO MAILBOX # ***** 211 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT SET CORRECTLY
DOP03B: COMB @#0 ;LOC. 0=1
DEC @#0 ;LOC. 0=0
BEQ TST115

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
CONDITIONAL BRANCH INST. AND <====
REPLACE THE MOVE INSTRUCTION <====
WHICH FOLLOWS W/ 751 <====

011066 012762 000212 177776
011074 005262 177774
011100 000000

MOV #212,-2(R2) ;MOVE TO MAILBOX # ***** 212 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DATA RESULT INCORRECT
OR SEQUENCE ERROR

2031
2032
2033
2034
2035
2036

: THIS TEST VERIFIES MODE 0,0 DOP NON-MODIFYING INSTRUCTIONS.
: R0 AND R4 ARE PRESET TO 0 AND 1 RESPECTIVELY. COMPARE INSTRUCTIONS ARE
: THEN EXECUTED AND CHECKED. FIRST R4 IS COMPARED TO R0 THEN R0 TO R4.
:

2037

.SBTTL TEST # 115 - TEST DOP NON-MODIFYING INST. W/ SOURCE MODE 0,0
:*****
:TEST 115 - TEST DOP NON-MODIFYING INST. W/ SOURCE MODE 0,0
:*****

011102 005212
011104 022712 000115
011110 001054
2038 011112 005000
2039 011114 005004
2040 011116 005204
2041 011120 020400
2042 011122 003006

TST115: INC (R2) ;UPDATE TEST NUMBER
CMP #115,(R2) ;SEQUENCE ERROR?
BNE TST116-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR R4 ;R4=0
INC R4 ;R4=1
CMP R4,R0 ;TRY COMPARE R4 TO R0
BGT DNM1

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 772 <====

011124 012762 000213 177776
011132 005262 177774
011136 000000
2043 011140 020004
2044 011142 002406

DNM1: MOV #213,-2(R2) ;MOVE TO MAILBOX # ***** 213 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT FOR CMP
CMP R0,R4 ;TRY COMPARE R0 TO R4
BLT DNM2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 762 <====

011144 012762 000214 177776
011152 005262 177774
011156 000000
2045 011160 005200
2046 011162 020400
2047 011164 001406

DNM2: MOV #214,-2(R2) ;MOVE TO MAILBOX # ***** 214 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT FOR CMP
INC R0 ;R0=1
CMP R4,R0 ;TRY COMPARE R4=1 TO R0=1
BEQ DNM3

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
; CONDITIONAL BRANCH INST. AND <=
; REPLACE THE MOVE INSTRUCTION <=
; WHICH FOLLOWS W/ 751 <=

011166 012762 000215 177776
011174 005262 177774
011200 000000
2048 011202 005000
2049 011204 005100
2050 011206 005004
2051 011210 030004
2052 011212 001406

DNM3: MOV #215,-2(R2) ;MOVE TO MAILBOX # ***** 215 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT (Z=1) FOR CMP
CLR R0 ;R0=0
COM R0 ;R0=177777
CLR R4 ;R4=0
BIT R0,R4 ;TRY BIT R0 TO R4
BEQ DNM4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 736 <

011214 012762 000216 177776
011222 005262 177774
011226 000000
2053 011230 005304
2054 011232 030004
2055 011234 100406

DNM4: MOV #216,-2(R2) ;MOVE TO MAILBOX # ***** 216 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT FOR BIT
DEC R4 ;R4=177777
BIT R0,R4 ;TRY BIT AGAIN
BMI TST116

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
; CONDITIONAL BRANCH INST. AND <=
; REPLACE THE MOVE INSTRUCTION <=
; WHICH FOLLOWS W/ 725 <=

01236 012762 GC0217 177776
011244 005262 177776
011250 000000

MOV #217,-2(R2) :MOVE TO MAILBOX # 217
INC -4(R2) :SET MSGTYP TO FATAL ERROR
HALT :CC'S NOT CORRECT FGR BIT
: OR SEQUENCE ERROR

2056
2057
2058
2059
2060

.....
: THIS TEST VERIFIES MODE 0, X DOUBLE OPERAND NON-MODIFYING INSTRUCTIONS.
: IT SETS DATA IN R0 AND LOCATION 0 AND COMPARES THEM USING DOPNM INSTRUCTIONS.
:

2061

.SBTTL TEST # 116 - TEST MODE 0,X DOUBLE-OPERAND NON-MODIFYING INSTS.
:.....
:TEST 116 - TEST MODE 0,X DOUBLE-OPERAND NON-MODIFYING INSTS.
:.....

011252 005212
011254 022712 000116
011260 001026
2062 011262 005000
2063 011264 005010
2064 011266 005110
2065 011270 005200
2066 011272 020037 000000
2067 011276 100403
2068 011300 001402
2069 011302 102401
2070 011304 103406

TST116: INC (R2) ;UPDATE TEST NUMBER
CMP #116,(R2) ;SEQUENCE ERROR?
BNE TST117-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
COM (R0) ;LOC. 0=177777
INC R0 ;R0=1
CMP R0,#0 ;TRY CMP MODE 0,3
BMI DNM03A ;CC=0001
BEC DNM03A
BVS DNM03A
BCS DNM03B

: TC SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 765 <

011306
011306 012762 000220 177776
011314 005262 177774
011320 000000
2071 011322 005300
2072 011324 001002
2073 011326 005210
2074 011330 001406

DNM03A: MOV #220,-2(R2) ;MOVE TO MAILBOX # ***** 220 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT SET CORRECTLY
DNM03B: DEC R0
BNE DNM03C
INC (R0)
BEQ TST117

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 753 <====

011332
011332 012762 000221 177776
011340 005262 177774
011344 000000

DNM03C: MOV #221,-2(R2) ;MOVE TO MAILBOX # ***** 221 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DATA INCORRECTLY MODIFIED BY CMP
; OR SEQUENCE ERROR

2075
2076
2077
2078
2079
2080
2081

:.....
: THIS TEST VERIFIES MODE 1 DOP INSTRUCTIONS. R0 IS SET TO -1
: AND LOC 0 TO 1. R4 IS THEN CLEARED AND USED TO POINT TO LOC 0.
: IN THE ADD MODE 1 INSTRUCTION, LOC 0 IS ADDED TO R0 AND THE
: RESULTS VERIFIED.
:.....

2082

.SBTTL TEST # 117 - TEST MODE 1 W/ DOP INST.

 :TEST 117 - TEST MODE 1 W/ DOP INST.

011346 005212
 011350 022712 000117
 011354 001011
 2083 011356 005000
 2084 011360 005100
 2085 011362 005004
 2086 011364 005014
 2087 011366 005214
 2088 011370 061400
 2089 011372 001406

TST117: INC (R2) ;UPDATE TEST NUMBER
 CMP #117,(R2) ;SEQUENCE ERROR?
 BNE TST120-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 COM R0 ;R0=177777
 CLR R4 ;R4=0
 CLR (R4) ;LOC 0=0
 INC (R4) ;LOC 0=1
 ADD (R4),R0 ;TRY ADD SOURCE MODE 1
 BEQ TST120

TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 CONDITIONAL BRANCH INST. AND <=====
 REPLACE THE MOVE INSTRUCTION <=====
 WHICH FOLLOWS W/ 770 <=====
 ; MOVE TO MAILBOX # ***** 222 *****
 ; SET MSGTYP TO FATAL ERROR
 ; RESULT OF ADD INCORRECT
 ; OR SEQUENCE ERROR

011374 012762 000222 177776
 011402 005262 177774
 011406 000000

MOV #222,-2(R2)
 INC -4(R2)
 HALT

2090
 2091
 2092
 2093
 2094
 2095
 2096

 : THIS TEST VERIFIES MODE 1 DOP BYTE INSTRUCTIONS WHICH ADDRESS
 : EVEN BYTES. LOC. 0 IS SET TO -1 AND R4 IS CLEARED. THEN R4 IS
 : SET TO -1 USING A BISB THRU R0 WITH MODE 1.
 : .

2097

.SBTTL TEST # 120 - TEST MODE 1 - EVEN BYTE W/ DOP INSTS.

.....
:TEST 120 - TEST MODE 1 - EVEN BYTE W/ DOP INSTS.
:.....

011410 005212
011412 022712 000120
011416 001011
2098 011420 005000
2099 011422 005010
2100 011424 005110
2101 011426 005004
2102 011430 151004
2103 011432 105104
2104 011434 001406

*ST120: INC (R2) ;UPDATE TEST NUMBER
CMP #120,(R2) ;SEQUENCE ERROR?
BNE TST121-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
COM (R0) ;LOC. 0=177777
CLR R4 ;R4=0
BISB (R0),R4 ;TRY MODE 1- EVEN BYTE W/ DOP
COMB R4 ;R4=0
BEQ TST121

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 770
: MOVE TO MAILBOX # ***** 223 *****
: SET MSGTYP TO FATAL ERROR
: RESULT OF BISB IS INCORRECT
: OR SEQUENCE ERROR

011436 012762 000223 177776
011444 005262 177774
011450 000000

MOV #223,-2(R2)
INC -4(R2)
HALT

2105
2106
2107
2108
2109
2110
2111
2112

.....
: THIS TEST VERIFIES MODE 1 DOP NON-MODIFYING INSTRUCTIONS
: WHICH ADDRESS EVEN BYTES. LOC. 0 IS SET TO -1 AND R0 IS CLEARED
: AND USED AS THE ADDRESSING REGISTER. R4 IS SET TO 377 AND A
: MODE 1,0 COMB INSTRUCTION IS USED THE RESULTS VERIFIED.
:.....

2113

..SBTTL TEST # 121 - TEST MODE 1 - EVEN BYTE W/ DOP NON-MOD INST

.....
:TEST 121 - TEST MODE 1 - EVEN BYTE W/ DOP NON-MOD INST
.....

011452 005212
011454 022712 000121
011460 001011
2114 011462 005000
2115 011464 005010
2116 011466 005110
2117 011470 005004
2118 011472 105104
2119 011474 121004
2120 011476 001406

TST121: INC (R2) ;UPDATE TEST NUMBER
CMP #121,(R2) ;SEQUENCE ERROR?
BNE TST122-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC 0=0
COM (R0) ;LOC 0=177777
CLR R4 ;R4=0
COMB R4 ;R4=377
CMPB (R0),R4 ;TRY MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING
BEQ TST122

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 770 <=====
:

011500 012762 000224 177776
011506 005262 177774
011512 000000

MOV #224,-2(R2) ;MOVE TO MAILBOX # ***** 224 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF CMPB INCORRECT
; OR SEQUENCE ERROR

2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132

.....
: THIS TEST VERIFIES MODE 1,0 MOVB INSTRUCTIONS
: WHICH ADDRESS EVEN BYTES. LOC. 0 IS SET TO 177400, R0 IS CLEARED AND
: R4 IS SET TO -1. MOVB ARE USED TO MOVE BYTE 0 TO R4. THIS
: VERIFIES THAT THE PROPER BYTE WAS SELECTED AND THAT THE SIGN-X-TEND
: FUNCTION WITH MODE 0.
: THEN LOC. 0 IS COMPLEMENTED AND THE SAME PROCEDURE EXERCISES
: THE LOGIC FOR COMPLEMENTARY DATA.
: THIS TEST EXERCISES UNIQUE MICROCODE.
.....

2133

.SBTTL TEST # 122 - TEST MOV INSTRUCTION MODE 1,0 EVEN BYTE

.....
 :TEST 122 - TEST MOV INSTRUCTION MODE 1,0 EVEN BYTE
 :.....

011514 005212
 011516 022712 000122
 011522 001024
 2134 011524 005000
 2135 011526 005010
 2136 011530 105110
 2137 011532 005110
 2138 011534 005004
 2139 011536 005104
 2140 011540 111004
 2141 011542 005704
 2142 011544 001406

TST122: INC (R2) ;UPDATE TEST NUMBER
 CMP #122,(R2) ;SEQUENCE ERROR?
 BNE TST123-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 CLR (R0) ;LOC 0=0
 COMB (R0) ;LOC 0=177400
 COM (R0)
 CLR R4 ;R4=0
 COM R4 ;R4=177777
 MOVB (R0),R4 ;R4=0
 TST R4 ;CHECK SIGN OF WORD
 BEQ DOP1

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 766

011546 012762 000225 177776
 011554 005262 177774
 011560 000000
 2143 011562 005110
 2144 011564 111004
 2145 011566 100406

MOV #225,-2(R2) ;MOVE TO MAILBOX # ***** 225 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;MOVB SHOULD SIGN X-TEND
 DOP1: COM (R0) ;LOC 0=177777
 MOVB (R0),R4 ;DO MOVB W/ EVEN BYTE
 BMI TST123

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 755

011570 012762 000226 177776
 011576 005262 177774
 011602 000000

MOV #226,-2(R2) ;MOVE TO MAILBOX # ***** 226 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;MOVB SHOULD SIGN X-TEND
 ; OR SEQUENCE ERROR

2146
 2147
 2148
 2149
 2150
 2151
 2152
 2153

.....
 : THIS TEST VERIFIES MODE 1 DOP INSTRUCTIONS WHICH REFERENCE
 : ODD BYTES. LOC. 0 IS SET TO 177400. R0 IS SET TO 0 AND R4 IS
 : SET TO 1. THE BISS INSTRUCTION USES THE DATA IN BYTE 1 TO SET BYTE 0.
 : THE RESULT IS CHECKED BY INCREMENTING THE WORD (LOC. 0) TO ZERO.
 :

SBTTL TEST # 123 - TEST MODE 1-ODD BYTE W/ DOP INSTS.

TEST 123 - TEST MODE 1-ODD BYTE W/ DOP INSTS.

011604 005212
011606 022712 000123
011612 001012
2155 011614 005000
2156 011616 005010
2157 011620 005004
2158 011622 005204
2159 011624 105114
2160 011626 151410
2161 011630 005210
2162 011632 001406

ST123: INC (R2) ;UPDATE TEST NUMBER
CMP #123,(R2) ;SEQUENCE ERROR?
BNE TST124-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
CLR R4 ;R4=0
INC R4 ;R4=1
COMB (R4) ;LOC. 0=177400
BISB (R4),(R0) ;TRY TO BIS LOW ORDER BITS W/ MODE 1
INC (R0) ;CHECK RESULT
BEQ TST124

TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
CONDITIONAL BRANCH INST. AND
REPLACE THE MOVE INSTRUCTION
WHICH FOLLOWS W/ 767
MOVE TO MAILBOX # ***** 227 *****
SET MSGTYP TO FATAL ERROR
RESULT OF BISB INCORRECT
OR SEQUENCE ERROR

011634 012762 000227 177776
011642 005262 177774
011646 000000

MOV #227,-2(R2)
INC -4(R2)
HALT

2163
2164
2165
2166
2167
2168
2169
2170

THIS TEST VERIFIES MODE 2 DOP INSTRUCTIONS. LOC. 0 IS SET TO -1.
R0 IS CLEARED AND USED AS THE MODE 2 ADDRESSING REGISTER TO MOVE LOC. 0
TO R7. THE DATA RESULTS ARE VERIFIED AND THE INCREMENTING OF THE REGISTER
IS CHECKED.

2171

```
.SBTTL TEST # 124 - TEST MODE 2 W/ DOP INSTS.
:*****
:TEST 124 - TEST MODE 2 W/ DOP INSTS.
:*****
```

```
011650 00521 000124
011652 022712
011656 001021
2172 011660 005000
2173 011662 005010
2174 011664 005110
2175 011666 012004
2176 011670 005204
2177 011672 001406
```

```
TST124: INC (R2) ;UPDATE TEST NUMBER
CMP #124,(R2) ;SEQUENCE ERROR?
BNE TST125-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
COM (R0) ;LOC. 0=177777
MOV (R0)+,R4 ;TRY MOVE MODE 2,0
INC R4 ;CHECK R4
BEQ DOP2
```

```
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <== =
; CONDITIONAL BRANCH INST. AND <== =
; REPLACE THE MOVE INSTRUCTION <=== =
; WHICH FOLLOWS W/ 771 <--- =
```

```
011674 012762 000230 177776
011702 005262 177774
011706 000000
2178 011710 005300
2179 011712 005300
2180 011714 001406
```

```
MOV #230,-2(R2) ;MOVE TO MAILBOX # ***** 230 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF MOV INST INCORRECT
DOP2: DEC R0 ;TEST R0 AFTER MODE 2
DEC R0
BEQ TST125
```

```
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <= =
; CONDITIONAL BRANCH INST. AND <= =
; REPLACE THE MOVE INSTRUCTION <= =
; WHICH FOLLOWS W/ 760 <= =
```

```
011716 012762 000231 177776
011724 005262 177774
011730 000000
```

```
MOV #231,-2(R2) ;MOVE TO MAILBOX # ***** 231 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REGISTER NOT INCREMENTED IN MODE 2
; OR SEQUENCE ERROR
```

2181
2182
2183
2184
2185
2186
2187
2188
2189
2190

```
*****
: THIS TEST VERIFIES MODE 2 DOP BYTE INSTRUCTIONS WHICH ADDRESS
: EVEN BYTES. LOC. 0 IS SET TO -1. R0 IS CLEARED AND USED AS THE
: ADDRESSING REGISTER IN A TEST WHICH TRIES TO CLEAR BYTE 1 USING
: BYTE 0 DATA AND A BICB. UNIQUE IN THIS TEST IS USE OF THE
: SAME ADDRESSING REGISTER FOR BOTH SOURCE AND DESTINATION. THE SOURCE AND
: DESTINATION IS CHECKED TO INSURE PROPER FUNCTIONING.
:*****
```

```

2191          .SBTTL TEST # 125 - TEST MODE 2 - EVEN BYTE W/ DOP INST.
:*****
:TEST 125 - TEST MODE 2 - EVEN BYTE W/ DOP INST.
:*****
011732 005212          TST125: INC      (R2)          ;UPDATE TEST NUMBER
011734 022712 000125  CMP      #125,(R2)      ;SEQUENCE ERROR?
011740 001022          BNE     TST126-10      ;BR TO ERROR HALT ON SEQ ERROR
2192 011742 005000          CLR     R0             ;R0=0
2193 011744 010010          MOV     R0,(R0)        ;LOC. 0=0
2194 011746 005110          COM     (R0)          ;LOC. 0=177777
2195 011750 142010          BICB   (R0)+,(R0)     ;TRY TO CLEAR BYTE 1 FROM BYTE 0 W/ BICB
2196 011752 105737 000001  TSTB   @#1            ;CHECK RESULT
2197 011756 001406          BEQ    DOPB2A         ;
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
:          CONDITIONAL BRANCH INST. AND <---
:          REPLACE THE MOVE INSTRUCTION <---
:          WHICH FOLLOWS W/ 770 <---
011760 012762 000232 177776          MOV     #232,-2(R2)    ;MOVE TO MAILBOX # ***** 232 *****
011766 005262 177774          INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
011772 000000          HALT                    ;BICB DESTINATION INCORRECT
2198 011774 105137 000000  DOPB2A: COMB   @#0      ;CHECK BICB SOURCE
2199 012000 001406          BEQ    TST126         ;
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
:          CONDITIONAL BRANCH INST. AND <---
:          REPLACE THE MOVE INSTRUCTION <---
:          WHICH FOLLOWS W/ 757 <---
012002 012762 000233 177776          MOV     #233,-2(R2)    ;MOVE TO MAILBOX # ***** 233 *****
012010 005262 177774          INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
012014 000000          HALT                    ;BICB SOURCE INCORRECTLY CHANGED
:          OR SEQUENCE ERROR
:*****
:
: THIS TEST VERIFIES MODE 2 DOP BYTE INSTRUCTIONS WHICH REFERENCE
: ODD BYTES. R0 IS SET TO 1, LOC. 0 IS SET TO 177400, AND R4 IS CLEARED.
: A MODE 2 MOVB USES R0 TO MOVE BYTE 1 TO R4. AN INCREMENT
: IS USED TO CHECK THAT THE PROPER BYTE WAS MOVED AND SIGN X-TENDED.
:
2200
2201
2202
2203
2204
2205
2206
    
```

2207

.SBTTL TEST # 126 - TEST MODE 2 - ODD BYTE W/ DOP INST.
:*****
:TEST 126 - TEST MODE 2 - ODD BYTE W/ DOP INST.
:*****

012016 005212
012020 022712 000126
012024 001023
2208 012026 005000
2209 012030 005004
2210 012032 005010
2211 012034 005110
2212 012036 105120
2213 012040 112004
2214 012042 005204
2215 012044 001406

TST126: INC (R2) ;UPDATE TEST NUMBER
CMP #126,(R2) ;SEQUENCE ERROR?
BNE TST127-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR R4 ;R4=0
CLR (R0) ;LOC. 0=0
COM (R0) ;LOC. 0=177777
COMB (R0)+ ;LOC 0=177400; R0=1
MOVB (R0)+,R4 ;TRY DOP MODE 2 W/ ODD BYTE
INC R4 ;CHECK RESULT OF MOVB
BEQ DOPB2B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 767 <

012046 012762 000234 177776
012054 005262 177774
012060 000000
2216 012062 005740
2217 012064 005700
2218 012066 001406

MOV #234,-2(R2) ;MOVE TO MAILBOX # ***** 234 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF MOVB INCORRECT
DOPB2B: TST -(R0) ;BUMP RO DOWN BY 2
TST R0 ;CHECK RO
BEQ TST127

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 756 <

012070 012762 000235 177776
012076 005262 177774
012102 000000

MOV #235,-2(R2) ;MOVE TO MAILBOX # ***** 235 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;MODE 2 BYTE DID NOT INCREMENT REG. CORRECTLY
; OR SEQUENCE ERROR

2219
2220
2221
2222
2223
2224
2225
2226

:*****
: THIS TEST VERIFIES MODE 3 DOUBLE-OPERAND INSTRUCTIONS.
: LOC. 0 IS LOADED WITH ALTERNATING ZEROES AND ONES; AND R0 IS LOADED
: WITH ALTERNATING ONES AND ZEROES. A MODE 3 BIS IS USED TO SET R0
: TO -1 BY USING LOC. 0 AS THE SOURCE TO BIS THE ZEROES IN R0. THE
: RESULT IS TESTED BY INCREMENTING R0 AND CHECKING FOR ZERO.
:*****

2227

.SBTTL TEST # 127 - TEST MODE 3 W/ DOP INSTS.
 :*****
 :TEST 127 - TEST MODE 3 W/ DOP INSTS.
 :*****

012104 005212
 012106 022712 000127
 012112 001013
 2228 012114 012737 052525 000000
 2229 012122 012700 125252
 2230 012126 053700 000000
 2231 012132 005200
 2232 012134 001406

TST127: INC (R2) ; JDATE TEST NUMBER
 CMP #127,(R2) ; SEQUENCE ERROR?
 BNE TST130-10 ; BR TO ERROR HALT ON SEQ ERROR
 MOV #052525,@#0 ; MOVE 52525 TO LOC. 0
 MOV #125252,R0 ; SET ALT. ONE AND ZERO IN R0
 BIS @#0,R0 ; TRY TO SET ALL OTHER BITS W/ MODE 3
 INC R0 ; TEST RESULT
 BEQ TST130
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 766 <====
 MOV #236,-2(R2) ; MOVE TO MAILBOX # ***** 236 *****
 INC -4(R2) ; SET MSGTYP TO FATAL ERROR
 HALT ; BIS W/ MODE 3 INCORRECT RESULT
 ; OR SEQUENCE ERROR

2233
 2234
 2235
 2236
 2237
 2238
 2239

:*****
 : THIS TEST VERIFIES MODE 3 DOUBLE OPERAND BYTE INSTRUCTIONS WHICH
 : ADDRESS EVEN BYTES. BYTE 0 IS SET TO ALTERNATING 1'S AND 0'S; BYTE 1,
 : ALTERNATING 0'S AND 1'S. R0 IS CLEARED AND A BISB IS USED TO
 : SET THE LOW BYTE OF R0 TO 252.
 :*****

2240

.SBTTL TEST # 130 - TEST MODE 3 - EVEN BYTE W/ DOP INSTS.
:*****
:TEST 130 - TEST MODE 3 - EVEN BYTE W/ DOP INSTS.
:*****

012152 005212
012154 022712 000130
012160 001013
2241 012162 012737 052652 000000
2242 012170 005000
2243 012172 153700 000000
2244 012176 022700 000252
2245 012202 001406

TST130: INC (R2) ;UPDATE TEST NUMBER
CMP #130,(R2) ;SEQUENCE ERROR?
BNE TST131-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #52652,@#0 ;MOVE 1'S AND 0' PATTERN TO LOC. 0
CLR R0 ;R0=0
BISB @#0,R0 ;TRY R0=252 W/ MODE 3 - EVEN BYTE
CMP #252,R0 ;BISB W/ EVEN BYTE SUCCESSFUL?
BEQ TST131
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 766 <====
MOV #237,-2(R2) ;MOVE TO MAILBOX # ***** 237 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;BISB W/ MODE 3 - EVEN BYTE FAILED
; OR SEQUENCE ERROR

2246
2247
2248
2249
2250
2251
2252

:*****
: THIS TEST VERIFIES MODE 3 DOUBLE OPERAND BYTE INSTRUCTIONS
: WHICH ADDRESS ODD BYTES. THE SAME PROCEDURE USED IN PREVIOUS
: TEST IS USED HERE. THIS TIME BYTE 1 IS USED AS THE SOURCE BYTE.
: THE EXPECTED RESULT IS: R0 = 125.
:*****

2253

.SBTTL TEST # 131 - TEST MODE 3 - ODD BYTE W/ DOP INSTS.
:*****
:TEST 131 - TEST MODE 3 - ODD BYTE W/ DOP INSTS.
:*****

012220 005212
012222 022712 000131
012226 001013
2254 012230 012737 052652 000000
2255 012236 005000
2256 012240 153700 000001
2257 012244 022700 000125
2258 012250 001406

TST131: INC (R2)
CMP #131,(R2)
BNE TST132-10
MOV #52652,@#0
CLR R0
BISB @# ,R0
CMP #125,R0
BEQ TST132

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:MOVE 1'S AND 0'S PATTERN TO LOC 0
:R0=0
:TRY R0=152 W/ MODE 3 - ODD BYTE
:R0=125?

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 766 <====

012252 012762 000240 177776
012260 005262 177774
012264 000000

MOV #240,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 240 *****
:SET MSGTYP TO FATAL ERROR
:BISB W/ MODE 3 - ODD BYTE FAILED
: OR SEQUENCE ERROR

2259

2260

.SBYTL TEST # 132 - TEST DEST. MODE 0-BYTE W/ DOP NON-MODIFYING MST

.....
 :TEST 132 - TEST DEST. MODE 0-BYTE W/ DOP NON-MODIFYING MST
 :.....

012266 005212
 012270 022712 000132
 012274 001023
 2261 012276 005000
 2262 012300 105100
 2263 012302 000263
 2264 012304 132700 000200
 2265 012310 001403
 2266 012312 102402
 2267 012314 103001
 2268 012316 100406

```

*ST132: INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #132,(R2)     ;SEQUENCE ERROR?
        BNE     TST133-10     ;BR TO ERROR HALT ON SEQ ERROR
        CLR     R0             ;R0=0
        COMB    R0             ;R0=377
        +SEC    SEV           ;SET C AND V BITS
        BITB    #20C,R0       ;TRY DOPNM DEST. MODE 0-BYTE
        BEQ     DNMB0A        ;BR TO ERROR IF Z BIT SET
        BVS     DNMB0A        ;BR TO ERROR IF V BIT SET
        BCC     DNMB0A        ;BR TO ERROR IF C BIT CLEAR.
        BMI     DNMB0B
  
```

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
;           CONDITIONAL BRANCH INST. AND <
;           REPLACE THE MOVE INSTRUCTION <-
;           WHICH FOLLOWS W/ 766 <-
  
```

012320
 012320 012762 000241 177776
 012326 005262 177776
 012332 000000
 2269 012334 105100
 2270 012336 001406

```

DNMB0A: MOV      #241,-2(R2)    ;MOVE TO MAILBOX # ***** 241 *****
        INC      -4(R2)       ;SET MSGTYP TO FATAL ERROR
        HALT
DNMB0B: COMB    R0             ;CC'S INCORRECT
        BEQ     *ST133        ;CHECK DESTINATION DATA
  
```

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
;           CONDITIONAL BRANCH INST. AND <
;           REPLACE THE MOVE INSTRUCTION <
;           WHICH FOLLOWS W/ 756 <-
  
```

012340 012762 000242 177776
 012346 005262 177776
 012352 000000

```

        MOV      #242,-2(R2)  ;MOVE TO MAILBOX # ***** 242 *****
        INC      -4(R2)       ;SET MSGTYP TO FATAL ERROR
        HALT
  
```

```

; DEST. DATA MODIFIED
; OR SEQUENCE ERROR
  
```

2271

2272

.SBTTL TEST # 133 - TEST DEST. MODE 1 W/ DOP NON-MODIFYING INST

.....
 :TEST 133 - TEST DEST. MODE 1 W/ DOP NON-MODIFYING INST
 :.....

012354 005212
 012356 022712 000133
 012362 001023
 2273 012364 005000
 2274 012366 005010
 2275 012370 000241
 2276 012372 032710 17777
 2277 012376 100403
 2278 012400 102402
 2279 012402 103401
 2280 012404 001406

TST133: INC (R2) ;UPDATE TEST NUMBER
 CMP #133,(R2) ;SEQUENCE ERROR?
 BNE TST134-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 CLR (R0) ;LOC. 0=0
 CLC ;CLEAR C BIT
 BIT #177777,(R0) ;TRY DOPNM DEST. MODE 1
 BMI DNM1A ;BR TO ERROR IF N BIT SET
 BVS DNM1A ;BR TO ERROR IF V BIT SET
 BCS DNM1A ;BR TO ERROR IF C BIT SET
 BEQ DNM1B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 766 <

012406
 012406 012762 000243 177776
 012414 005262 177774
 012420 000000
 2281 012422 005710
 2282 012424 001406

DNM1A: MOV #243,-2(R2) ;MOVE TO MAILBOX # ***** 243 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;COND. CODES INCORRECT
 DNM1B: TST (R0) ;CHECK TEST DATA
 BEQ TST134

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 756 <

012426 012762 000244 177776
 012434 005262 177774
 012440 000000

MOV #244,-2(R2) ;MOVE TO MAILBOX # ***** 244 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DESTINATION DATA MODIFIED
 ; OR SEQUENCE ERROR

2283

2284

.SBTTL TEST # 134 - TEST DEST, MODE 2 W/ DOP NON-MODIFYING INST.

.....
:TEST 134 - TEST DEST, MODE 2 W/ DOP NON-MODIFYING INST.
.....

012442 005212
012444 022712 000134
012450 001035
2285 012452 005000
2286 012454 005010
2287 012456 052710 125252
2288 012462 032720 077777
2289 012466 102402
2290 012470 001401
2291 012472 100006

*ST134: INC (R2)
CMP #134,(R2)
BNE TST135-10
CLR R0
CLR (R0)
BIS #125252,(R0)
BIT #77777,(R0)+
BVS DNM2A
BEQ DNM2A
BPL DNM2B

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:R0=0
:LOC. 0=0
:LOC. 0=125252
:TRY DOPNM INST W/ MODE 2
:BR TO ERROR IF V BIT SET
:BR TO ERROR IF Z-BIT SET

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 766

012474
012474 012762 000245 177776
012502 005262 177774
012506 000000
2292 012510 005300
2293 012512 005300
2294 012514 001406

DNM2A: MOV #245,-2(R2)
INC -4(R2)
HALT
DNM2B: DEC R0
DEC R0
BEQ DNM2D

:MOVE TO MAILBOX # ***** 245 *****
:SET MSGTYP TO FATAL ERROR
:COND. CODES INCORRECT
:DECREMENT R0 TO CHECK IT.

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 755

012516
012516 012762 000246 177776
012524 005262 177774
012530 000000
2295 012532 022710 125252
2296 012536 001406

DNM2C: MOV #246,-2(R2)
INC -4(R2)
HALT
DNM2D: CMP #125252,(R0)
BEQ TST135

:MOVE TO MAILBOX # ***** 246 *****
:SET MSGTYP TO FATAL ERROR
:MODE 2 REGISTER NOT INCREMENTED BY 2
:CHECK DEST. DATA

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 744

012540 012762 000247 177776
012546 005262 177774
012552 000000

MOV #247,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 247 *****
:SET MSGTYP TO FATAL ERROR
:DEST. DATA MODIFIED
: OR SEQUENCE ERROR

2297

2298

..SBTTL TEST # 135 - TEST DEST. MODE 2-BYTE, W/DOP NON-MODIFYING INST

.....
 :TEST 135 - TEST DEST. MODE 2-BYTE, W/DOP NON-MODIFYING INST

012554 005212
 012556 022712 000135
 012562 001063
 2299 012564 005000
 2300 012566 005010
 2301 012570 052710 052652
 2302 012574 000263
 2303 012576 132720 000201
 2304 012602 001403
 2305 012604 103002
 2306 012606 102401
 2307 012610 100406

TST135: INC (R2)
 CMP #135,(R2)
 BNE TST136-10
 CLR R0
 CLR (R0)
 BIS #52652,(R0)
 +SEC.SEV
 BITB #201,(R0)+
 BEQ DNMB2A
 BCC DNMB2A
 BVS DNMB2A
 BMI DNMB2B

:UPDATE TEST NUMBER
 :SEQUENCE ERROR?
 :BR TO ERROR HALT ON SEQ ERROR
 :R0=0
 :LOC. 0=0
 :LOC. 0=52652
 :SET C AND V BITS
 :TRY DOPNM INST. W/ MODE 2 EVEN BYTE
 :BR TO ERROR IF Z-BIT SET
 :BR TO ERROR IF C-BIT CLEAR
 :BR TO ERROR IF V-BIT SET

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 764 <====

012612
 012612 012762 000250 177776
 012620 005262 177776
 012624 000000
 2308 012626 005300
 2309 012630 001406

DNMB2A: MOV #250,-2(R2)
 INC -4(R2)
 HALT
 DNMB2B: DEC R0
 BEQ DNMB2C

:MOVE TO MAILBOX # ***** 250 *****
 :SET MSGTYP TO FATAL ERROR
 :COND. CODES INCORRECT
 :CHECK DEST. REGISTER.

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 754 <

012632 012762 000251 177776
 012640 005262 177776
 012644 000000
 2310 012646 005200
 2311 012650 132720 000201
 2312 012654 001402
 2313 012656 102401
 2314 012660 100006

MOV #251,-2(R2)
 INC -4(R2)
 HALT
 DNMB2C: INC R0
 BITB #201,(R0)+
 BEQ DNMB2D ;BR TO ERROR IF Z-BIT SET
 BVS DNMB2D
 BPL DNMB2E

:MOVE TO MAILBOX # ***** 251 *****
 :SET MSGTYP TO FATAL ERROR
 :DEST. REGISTER NOT INCREMENTED BY 1
 :R0=1
 :TRY DOPNM INST. W/MODE 2-ODD BYTE
 :BR TO ERROR IF Z-BIT SET
 :BR TO ERROR IF V-BIT SET

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 740 <

012662
 012662 012762 000252 177776
 012670 005262 177776
 012674 000000
 2315 012676 005300
 2316 012700 005300
 2317 012702 001406

DNMB2D: MOV #252,-2(R2)
 INC -4(R2)
 HALT
 DNMB2E: DEC R0
 DEC R0
 BEQ DNMB2F

:MOVE TO MAILBOX # ***** 252 *****
 :SET MSGTYP TO FATAL ERROR
 :COND. CODES INCORRECT
 :DEC R0 TO CHECK IT.

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 727 <====

012704 012762 000253 177776
 012712 005262 177776
 012716 000000
 2318 012720 022710 052652

MOV #253,-2(R2)
 INC -4(R2)
 HALT
 DNMB2F: CMP #52652,(R0)

:MOVE TO MAILBOX # ***** 253 *****
 :SET MSGTYP TO FATAL ERROR
 :DEST. REGISTER NOT INCREMENTED BY 1
 :CHECK DEST. DATA IS UNMODIFIED

2519 012724 001406

BEQ TST136

012726 012762 000254 177776
012734 005262 177776
012740 000000

MOV #254,-2(R2)
INC -4(R2)
HALT

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 716 <====
: MOVE TO MAILBOX # ***** 254 *****
: SET MSGTYP TO FATAL ERROR
: DEST. DATA WAS MODIFIED.
: OR SEQUENCE ERROR

23c
23c


```

.....
:SBITL TEST # 136 - TEST DEST. MODE 3-BYTES W/DOP NON-MODIFYING INST.
:TEST 136 - TEST DEST. MODE 3-BYTES W/DOP NON-MODIFYING INST.
:.....
2322 012742 005212          ST136: INC      (R2)          ;UPDATE TEST NUMBER
      012744 022712 000136  CMP      #136,(R2)      ;SEQUENCE ERROR?
      012750 001060          BNE      TST137-10     ;BR TO ERROR HALT ON SEQ ERROR
2323 012752 005000          CLR      R0           ;R0=0
2324 012754 005010          CLR      (R0)         ;LOC. 0=0
2325 012756 052710 125125  BIS      #125125,(R0) ;LOC. 0=125125
2326 012762 105100          COMB    R0           ;R0=377
2327 012764 005200          INC      R0           ;R0=400
2328 012766 005010          CLR      (R0)         ;LOC. 400=0
2329 012770 000263          +SEC   SEV          ;C-BIT=V-BIT=1
2330 012772 132730 000201  BITB    #201,@(R0)+  ;TRY DOPNM W/MODE 3-EVEN BYTE
2331 012776 001403          BEQ     DNMB3A        ;BR TO ERROR IF Z BIT SET
2332 013000 102402          BVS     DNMB3A        ;BR TO ERROR IF V BIT SET
2333 013002 103001          BCC     DNMB3A        ;BR TO ERROR IF C BIT CLEAR
2334 013004 100006          BPL     DNMB3B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
; CONDITIONAL BRANCH INST. AND
; REPLACE THE MOVE INSTRUCTION
; WHICH FOLLOWS W/ 761
.....
      013006          DNMB3A:
2335 013006 012762 000255 177776  MOV     #255,-2(R2)   ;MOVE TO MAILBOX # ***** 255 *****
      013014 005262 177774          INC     -4(R2)       ;SET MSGTYP TO FATAL ERROR
      013020 000000          HALT                    ;COND. CODES INCORRECT
2336 013022 022700 000402  DNMB3B: CMP     #402,R0 ;CHECK DEST. REGISTER INC. BY 2 AND INC BY 2 AGAIN
      013026 001406          BEQ     DNMB3C

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
; CONDITIONAL BRANCH INST. AND
; REPLACE THE MOVE INSTRUCTION
; WHICH FOLLOWS W/ 750
.....
      013030 012762 000256 177776  MOV     #256,-2(R2)   ;MOVE TO MAILBOX # ***** 256 *****
      013036 005262 177774          INC     -4(R2)       ;SET MSGTYP TO FATAL ERROR
      013042 000000          HALT                    ;DEST. REGISTER NOT INCREMENTED BY 2
2337 013044 005200          DNMB3C: INCL    R0     ;R0=404
2338 013046 005200          INC     R0
2339 013050 132730 000201  BITB    #201,@(R0)+  ;TRY DOPNM DEST MODE 3-BYTE(ODD)
2340 013054 001402          BEQ     DNMB3D        ;BR TO ERROR IF Z BIT SET
2341 013056 102401          BVS     DNMB3D        ;BR TO ERROR IF V BIT SFT
2342 013060 100406          BMI     DNMB3E

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
; CONDITIONAL BRANCH INST. AND
; REPLACE THE MOVE INSTRUCTION
; WHICH FOLLOWS W/ 733
.....
      013062          DNMB3D:
2343 013062 012762 000257 177776  MOV     #257,-2(R2)   ;MOVE TO MAILBOX # ***** 257 *****
      013070 005262 177774          INC     -4(R2)       ;SET MSGTYP TO FATAL ERROR
      013074 000000          HALT                    ;COND. CODES INCORRECT
2344 013076 005004          DNMB3E: CLR     R4     ;R4=0
2345 013100 022714 125125  CMP     #125125,(R4) ;CHECK DEST. DATA
      013104 001406          BEQ     TST137

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
; CONDITIONAL BRANCH INST. AND
; REPLACE THE MOVE INSTRUCTION
; WHICH FOLLOWS W/ 721
.....

```

013106 012762 000260 177776
013114 005262 177776
013120 000000

MOV #260,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # 260
:SET MSGTYP TO FATAL ERROR
:DEST. DATA MODIFIED
: OR SEQUENCE ERROR

248

SBTTL TEST # 137 - TEST DEST. MODE 4 W/DOP NON-MODIFYING INST.

TEST 137 - TEST DEST. MODE 4 W/DOP NON-MODIFYING INST.

013122	005212				ST137: INC	(R2)	:UPDATE TEST NUMBER	
013124	022712	000137			CMP	#137,(R2)	:SEQUENCE ERROR?	
013130	001041				BNE	TST140-10	:BR TO ERROR HALT ON SEQ ERROR	
2348 013132	005000				CLR	R0	:R0=0	
2349 013134	005010				CLR	(R0)	:LOC. 0=0	
2350 013136	052710	125252			BIS	#125252,(R0)	:LOC. 0=125125	
2351 013142	052700	000002			BIS	#2,R0	:R0=2	
2352 013146	000277				SCC		:SET ALL COND. CODE BITS	
2353 013150	032740	020000			BIT	#20000,-(R0)	:TRY DOPNM W/ MODE 4	
2354 013154	100403				BMI	DNM4A	:BR TO ERROR IF N-BIT SET	
2355 013156	102402				BVS	DNM4A	:BR TO ERROR IF V-BIT SET	
2356 013160	103001				BCC	DNM4A	:BR TO ERROR IF C-BIT CHAR	
2357 013162	001006				BNE	DNM4B		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<-- =
							CONDITIONAL BRANCH INST. AND	<-- =
							REPLACE THE MOVE INSTRUCTION	<-- =
							WHICH FOLLOWS W/ 762	<-- =
013164				DNM4A:				
013164	012762	000261	177776		MOV	#261,-2(R2)	:MOVE TO MAILBOX # ***** 261 *****	
013172	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
013176	000000				HALT		:COND. CODES INCORRECT	
2358 013200	005700			DNM4B:	TST	R0	:CHECK DEST. REGISTER	
2359 013202	001406				BEG	DNM4C		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<-- =
							CONDITIONAL BRANCH INST. AND	<-- =
							REPLACE THE MOVE INSTRUCTION	<-- =
							WHICH FOLLOWS W/ 752	<-- =
013204	012762	000262	177776		MOV	#262,-2(R2)	:MOVE TO MAILBOX # ***** 262 *****	
013212	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
013216	000000				HALT		:DEST. REGISTER NOT DECREMENTED BY 2	
2360 013220	022737	125252	000000	DNM4C:	CMP	#125252,R0	:CHECK DEST. DATA	
2361 013226	001406				BEG	TST140		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<-- =
							CONDITIONAL BRANCH INST. AND	<-- =
							REPLACE THE MOVE INSTRUCTION	<-- =
							WHICH FOLLOWS W/ 740	<-- =
013230	012762	000263	177776		MOV	#263,-2(R2)	:MOVE TO MAILBOX # ***** 263 *****	
013236	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
013242	000000				HALT		:DEST. DATA MODIFIED	
							: OR SEQUENCE ERROR	

2362

363

.....
:SBITL TEST # 140 - TEST DEST. MODE 4-BYTE W/ DOP NON-MODIFYING INST.
:TEST 140 - TEST DEST. MODE 4-BYTE W/ DOP NON-MODIFYING INST.
:TEST 140: INC (R2) :UPDATE TEST NUMBER
:.....

013244 005212
013246 022712 000140
013252 001063
2364 013254 005000
2365 013256 005010
2366 013260 052710 052652
2367 013264 052700 000002
2368 013270 000257
2369 013272 132740 000201
2370 013276 102403
2371 013300 001402
2372 013302 103401
2373 013304 001006

ST140: INC (R2) :UPDATE TEST NUMBER
CMP #140,(R2) :SEQUENCE ERROR?
BNE TST141-10 :BR TO ERROR HALT ON SEQ ERROR
CLR R0 :R0=0
CLR (R0) :LOC. 0=0
BIS #52652,(R0) :LOC. 0=52652
BIS #2,R0 :R0=2
CCC :COND. CODES=0
BITB #201,-(R0) :TRY DOPNM INST W/MODE 4 ODD BYTE
BVS DNMB4A :BR TO ERROR IF V BIT SET
BEQ DNMB4A :BR TO ERROR IF Z BIT SET
BCS DNMB4A :BR TO ERROR IF C BIT SET
BNE DNMB4B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
: CONDITIONAL BRANCH INST. AND <----=
: REPLACE THE MOVE INSTRUCTION <---=
: WHICH FOLLOWS W/ 762 <---=

013306
013306 012762 000264 177776
013314 005262 177776
013320 000000
2374 013322 022700 000001
2375 013326 001406

DNMB4A: MOV #264,-2(R2) :MOVE TO MAILBOX # ***** 264 *****
INC -4(R2) :SET MSGTYP TO FATAL ERROR
HALT :COND. CODES INCORRECT
DNMB4B: CMP #1,R0 :CHECK DEST. REGISTER
BEQ DNMB4C

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
: CONDITIONAL BRANCH INST. AND <---=
: REPLACE THE MOVE INSTRUCTION <---=
: WHICH FOLLOWS W/ 751 <---=

013330 012762 000265 177776
013336 005262 177776
013342 000000
2376 013344 132740 000201
2377 013350 001401
2378 013352 100406

MOV #265,-2(R2) :MOVE TO MAILBOX # ***** 265 *****
INC -4(R2) :SET MSGTYP TO FATAL ERROR
HALT :DEST REG. NOT DECREMENTED BY 1
DNMB4C: BITB #201,-(R0) :TRY DOPNM INST. W/MODE 4 EVEN BYTE
BEQ DNMB4D :BR TO ERROR IF Z-BIT SET
BMI DNMB4E

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
: CONDITIONAL BRANCH INST. AND <---=
: REPLACE THE MOVE INSTRUCTION <---=
: WHICH FOLLOWS W/ 737 <---=

013354
013354 012762 000266 177776
013362 005262 177776
013366 000000
2379 013370 005700
2380 013372 001406

DNMB4D: MOV #266,-2(R2) :MOVE TO MAILBOX # ***** 266 *****
INC -4(R2) :SET MSGTYP TO FATAL ERROR
HALT :COND. CODES INCORRECT
DNMB4E: TST R0 :CHECK DEST. REGISTER
BEQ DNMB4F

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
: CONDITIONAL BRANCH INST. AND <---=
: REPLACE THE MOVE INSTRUCTION <---=
: WHICH FOLLOWS W/ 727 <---=

013374 012762 000267 177776
013402 005262 177776
013406 000000
2381 013410 022710 052652
2382 013414 001406

MOV #267,-2(R2) :MOVE TO MAILBOX # ***** 267 *****
INC -4(R2) :SET MSGTYP TO FATAL ERROR
HALT :DEST. REG. NOT DECREMENTED BY 1
DNMB4F: CMP #52652,(R0) :CHECK DESTINATION DATA
BEQ TST141

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---

013416 012762 000270 77776
013424 005262 177774
013430 000000

MOV #270,-2(R2)
INC -4(R2)
HALT

:
: CONDITIONAL BRANCH INST. AND <===
: REPLACE THE MOVE INSTRUCTION <===
: WHICH FOLLOWS W/ 716 <===
: MOVE TO MAILBOX # ***** 270 *****
: SET MSGTYP TO FATAL ERROR
: DEST. DATA MODIFIED
: OR SEQUENCE ERROR

2383

2384

.....
 .SBTTL TEST # 141 - TEST DEST MODE 5 W/DOP NON-MODIFYING INST.

 :TEST 141 - TEST DEST MODE 5 W/DOP NON-MODIFYING INST.

013432	005212			TST141:	INC	(R2)		:UPDATE TEST NUMBER
013434	022712	000141			CMP	#141,(R2)		:SEQUENCE ERROR?
013440	001042				BNE	TST142-10		:BR TO ERROR HALT ON SEQ ERROR
2385	013442	005000			CLR	R0		:R0=0
2386	013444	005010			CLR	(R0)		:LOC 0=0
2387	013446	052710	100000		BIS	#100000,(R0)		:LOC, 0=100000
2388	013452	052700	000402		BIS	#402,R0		:R0=2
2389	013456	000277			SCC			:SET ALL COND. CODE BITS
2390	013460	032750	100000		BIT	#100000,@-(R0)		:TRY DOPNM W/MODE 5
2391	013464	102403			BVS	DNM5A		:BR TO ERROR IF V-BIT SET
2392	013466	103002			BCC	DNM5A		:BR TO ERROR IF C-BIT CLEAR
2393	013470	001401			BEQ	DNM5A		:BR TO ERROR IF Z-BIT SET
2394	013472	100406			BMI	DNM5B		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <==--
 : CONDITIONAL BRANCH INST. AND <---=
 : REPLACE THE MOVE INSTRUCTION <==--=
 : WHICH FOLLOWS W/ 762 <==--

013474				DNM5A:	MOV	#271,-2(R2)		:MOVE TO MAILBOX # ***** 271 *****
013474	012762	000271	177776		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
013502	005262	177774			HALT			:COND. CODES INCORRECT
013506	000000				DNM5B:	CMP #400,R0		:CHECK DEST. REGISTER
2395	013510	022700	000400		BEQ	DNM5C		
2396	013514	001406						

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 751 <====

013516	012762	000272	177776		MOV	#272,-2(R2)		:MOVE TO MAILBOX # ***** 272 *****
013524	005262	177774			INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
013530	000000				HALT			:DEST. REGISTER NOT DECREMENTED BY 2
2397	013532	022737	100000	000000	DNM5C:	CMP #100000,@#0		:CHECK DESTINATION DATA
2398	013540	001406			BEQ	TST142		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < ==-
 : CONDITIONAL BRANCH INST. AND < -
 : REPLACE THE MOVE INSTRUCTION < -
 : WHICH FOLLOWS W/ 737 < -

013542	012762	000273	177776		MOV	#273,-2(R2)		:MOVE TO MAILBOX # ***** 273 *****
013550	005262	177774			INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
013554	000000				HALT			:DEST. DATA INCORRECTLY MODIFIED

: OR SEQUENCE ERROR

2399

2400

.....
:SBTTL TEST # 142 - TEST DEST. MODE 6 W/DOP NON-MODIFYING INST.
:TEST 142 - TEST DEST. MODE 6 W/DOP NON-MODIFYING INST.
:.....

013556 005212
013560 022712 000142
013564 001041
2401 013566 005000
2402 013570 005010
2403 013572 052710 000001
2404 013576 005100
2405 013600 032760 000001 000001
2406 013606 001403
2407 013610 102402
2408 013612 103001
2409 013614 100006

TST142: INC (R2) ;UPDATE TEST NUMBER
CMP #142,(R2) ;SEQUENCE ERROR?
BNE TST143-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC> 0=0
BIS #1,(R0) ;LOC. 0=1
COM R0 ;R0=-1 C-BIT=1
BIT #1,1(R0) ;TRY DOPNM W/MODE 6
BEQ DNM6A ;BR TO ERROR IF Z-BIT SET
BVS DNM6A ;BR TO ERROR IF V-BIT SET
BCC DNM6A ;BR TO ERROR IF C-BIT CLEAR
BPL DNM6B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
; CONDITIONAL BRANCH INST. AND <---
; REPLACE THE MOVE INSTRUCTION <---
; WHICH FOLLOWS W/ 763 <---

013616
013616 012762 000274 177776
013624 005262 177774
013630 000000
2410 013632 022700 177777
2411 013636 001406

DNM6A: MOV #274,-2(R2) ;MOVE TO MAILBOX # ***** 274 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;COND CODES INCORRECT
DNM6B: CMP #-1,R0 ;CHECK DEST. REGISTER
BEQ DNM6C

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
; CONDITIONAL BRANCH INST. AND <=====
; REPLACE THE MOVE INSTRUCTION <=====
; WHICH FOLLOWS W/ 752 <=====
; MOVE TO MAILBOX # ***** 275 *****

013640 012762 000275 177776
013646 005262 177774
013652 000000
2412 013654 022737 000001 000000
2413 013662 001406

DNM6C: MOV #275,-2(R2) ;MOVE TO MAILBOX # ***** 275 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DEST. REGISTER MODIFIED
DNM6C: CMP #1,R0 ;CHECK DEST. DATA
BEQ TST143

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
; CONDITIONAL BRANCH INST. AND <=====
; REPLACE THE MOVE INSTRUCTION <=====
; WHICH FOLLOWS W/ 740 <=====
; MOVE TO MAILBOX # ***** 276 *****

013664 012762 000276 177776
013672 005262 177774
013676 000000

MOV #276,-2(R2) ;MOVE TO MAILBOX # ***** 276 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DEST. DATA MODIFIED
; OR SEQUENCE ERROR

2414

2415

.SBTTL TEST # 143 - TEST DEST MODE 7 W/DOP NON-MODIFYING INST.
:*****
:TEST 143 - TEST DEST MODE 7 W/DOP NON-MODIFYING INST.
:*****

013700	005212			TST143:	INC	(R2)	:UPDATE TEST NUMBER
013702	022712	000143			CMP	#143,(R2)	:SEQUENCE ERROR?
013706	001042				BNE	TST144-10	:BR TO ERROR HALT ON SEQ ERROR
2416 013710	005000				CLR	R0	:R0=0
2417 013712	005010				CLR	(R0)	:LOC. 0=0 (C-BIT=0
2418 013714	052710	125125			BIS	#125125,(R0)	:LOC. 0=125125
2419 013720	052700	000001			BIS	#1,R0	:R0=1
2420 013724	132770	000125	000403		BITB	#125,@403(R0)	:TRY DOPNM W/MODE 7
2421 013732	102403				BVS	DNM7A	:BR TO ERROR IF V-BIT SET
2422 013734	100402				BMI	DNM7A	:BR TO ERROR IF N-BIT SET
2423 013736	103401				BCS	DNM7A	:BR TO ERROR IF C-BIT SET
2424 013740	001406				BEQ	DNM7B	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < --
: CONDITIONAL BRANCH INST. AND < =
: REPLACE THE MOVE INSTRUCTION < =
: WHICH FOLLOWS W/ 762 < -

013742				DNM7A:			
013742	012762	000271	177776		MOV	#277,-2(R2)	:MOVE TO MAILBOX # ***** 277 *****
013750	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
013754	000000				HALT		:COND. CODES INCORRECT
2425 013756	022700	000001		DNM7B:	CMP	#1,R0	:CHECK DEST. REGISTER
2426 013762	001406				BEQ	DNM7C	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < --
: CONDITIONAL BRANCH INST. AND < - -
: REPLACE THE MOVE INSTRUCTION < -
: WHICH FOLLOWS W/ 751 < -

013764	012762	000300	177776		MOV	#300,-2(R2)	:MOVE TO MAILBOX # ***** 300 *****
013772	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
013776	000000				HALT		:DESTINATION REGISTER MODIFIED
2427 014000	022737	125125	000000	DNM7C:	CMP	#125125,@#0	:CHECK DEST. DATA
2428 014006	001406				BEQ	TST144	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <-
: WHICH FOLLOWS W/ 737 <

014010	012762	000301	177776		MOV	#301,-2(R2)	:MOVE TO MAILBOX # ***** 301 *****
014016	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
014022	000000				HALT		:DEST. DATA INCORRECT
							: OR SEQUENCE ERROR

2429
2430
2431
2432
2433
2434
2435

:*****
: THIS TEST VERIFIES THE MOV DESTINATION MODE 1 INSTRUCTION.
: DATA IS SET IN R0 USING SOP INSTRUCTIONS AND THEN MOVED TO LOC. 0
: USING MOV SRC MODE 0, DEST. MODE 1.
:*****

2436

.SBTTL TEST # 144 - TEST MOV DESTINATION MODE 1

.....
:TEST 144 - TEST MOV DESTINATION MODE 1
:.....

014024 005212
014026 022712 000144
014032 001022
2437 014034 005000
2438 014036 005010
2439 014040 005100
2440 014042 005004
2441 014044 010014
2442 014046 102402
2443 014050 001401
2444 014052 100406

TST144: INC (R2) ;UPDATE TEST NUMBER
CMP #144,(R2) ;SEQUENCE ERROR?
BNE TST145-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
COM R0 ;R0=-1
CLR R4 ;R4 POINTS TO LOC. 0
MOV R0,(R4) ;TRY MOVE MODE 0.1
BVS MDM1A ;BR TO ERROR IF V SET
BEQ MDM1A ;BR TO ERROR IF Z SET
BMI MDM1B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 767 <==

014054
014054 012762 000302 177776
014062 005262 177774
014066 000000
2445 014070 005704
2446 014072 001406

MDM1A: MOV #302,-2(R2) ;MOVE TO MAILBOX # ***** 302 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CONDITION CODE NOT CORRECT
MDM1B: TST R4
BEQ TST145

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 757 <

014074 012762 000303 177776
014102 005262 177774
014106 000000

MOV #303,-2(R2) ;MOVE TO MAILBOX # ***** 303 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DESTINATION REGISTER INCORRECTLY ALTERED
; OR SEQUENCE ERROR

2447
2448
2449
2450
2451
2452
2453

.....
: THIS TEST VERIFIES THE MOV DESTINATION MODE 2 INSTRUCTION.
: DATA IS SET IN R0 USING SOP INSTRUCTIONS AND THEN MOVED
: TO LOCATION 0 USING MOV SRC MODE 0, DFST. MODE 1.
:

2454

.SBTTL TEST # 145 - TEST MOV DESTINATION MODE 2

:TEST 145 - TEST MOV DESTINATION MODE 2

014110 005212
014112 022712 000145
014116 001033
2455 014120 005000
2456 014122 005010
2457 014124 005110
2458 014126 010020
2459 014130 100402
2460 014132 102401
2461 014134 001406

TST145: INC (R2) ;UPDATE TEST NUMBER
CMP #145,(R2) ;SEQUENCE ERROR?
BNE TST146-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC.0=0
COM (R0) ;LOC. 0-
MOV R0,(R0)+ ;TRY MOVF MODE 0,2
BMI MDM2A ;BR TO ERROR IF N SET
BVS MDM2A ;BR TO ERROR IF V SET
BEQ MDM2B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 770 <====

014136
014136 012762 000304 177776
014144 005262 177774
014150 000000
2462 014152 005300
2463 014154 005300
2464 014156 001406

MDM2A: MOV #304,-2(R2) ;MOVE TO MAILBOX # ***** 304 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S INCORRECT
MDM2B: DEC R0
DEC R0
BEQ MDM2D

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
; CONDITIONAL BRANCH INST. AND <---
; REPLACE THE MOVE INSTRUCTION <---
; WHICH FOLLOWS W/ 757 <---

014160
014160 012762 000305 177776
014166 005262 177774
014172 000000
2465 014174 005737 000000
2466 014200 001406

MDM2C: MOV #305,-2(R2) ;MOVE TO MAILBOX # ***** 305 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DESTINATION REGISTER NOT INCREMENTED PROPERLY
MDM2D: TST @R0
BEQ TST146

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 746 <====

014202 012762 000306 177776
014210 005262 177774
014214 000000

MOV #306,-2(R2) ;MOVE TO MAILBOX # ***** 306 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DESTINATION DATA INCORRECT
; OR SEQUENCE ERROR

2467
2468
2469
2470
2471
2472

: THIS TEST VERIFIES DESTINATION MODE 2 W/MOVB INSTS. TWO DIFFERENT MOVB
: INSTRUCTIONS ARE USED TO MOVE A TEST PATTERN FIRST TO BYTE 0 THEN TO BYTE 1.
:

2473

.SBTTL TEST # 146 - TEST MOV-BYTE DESTINATION MODE 2

:TEST 146 - TEST MOV-BYTE DESTINATION MODE 2

014216 005212
014220 022712 000146
014224 001060
2474 014226 005000
2475 014230 005010
2476 014232 112720 000125
2477 014236 102402
2478 014240 001401
2479 014242 100006

TST146: INC (R2) ;UPDATE TEST NUMBER
CMP #146,(R2) ;SEQUENCE ERROR?
BNE TST147-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
MOVB #125,(R0)+ ;TRY DESTINATION MODE 2 W/EVEN BYTE
BVS MBDM2A ;BR TO ERROR IF V SET
BEQ MBDM2A ;BR TO ERROR IF Z SET
BPL MBDM2B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-767
: CONDITIONAL BRANCH INST. AND <-767
: REPLACE THE MOVE INSTRUCTION <-
: WHICH FOLLOWS W/ 770 <

014244
014244 012762 000307 177776
014252 005262 177774
014256 000000
2480 014260 022700 000001
2481 014264 001406

MBDM2A: MOV #307,-2(R2) ;MOVE TO MAILBOX # ***** 307 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S INCORRECT
MBDM2B: CMP #1,R0
BEQ MBDM2C

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 757 <

014266 012762 000310 177776
014274 005262 177774
014300 000000
2482 014302 112720 000252
2483 014306 102402
2484 014310 001401
2485 014312 100406

MOV #310,-2(R2) ;MOVE TO MAILBOX # ***** 310 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REGISTER NOT INCREMENTED BY ONE
MBDM2C: MOVB #252,(R0)+ ;TRY DESTINATION MODE 2 W/ODD BYTE
BVS MBDM2D
BEQ MBDM2D
BMI MBDM2E

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 744 <

014314
014314 012762 000311 177776
014322 005262 177774
014326 000000
2486 014330 022700 000002
2487 014334 001406

MBDM2D: MOV #311,-2(R2) ;MOVE TO MAILBOX # ***** 311 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT SET CORRECT
MBDM2E: CMP #2,R0
BEQ MBDM2F

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 733 <

014336 012762 000312 177776
014344 005262 177774
014350 000000
2488 014352 022737 125125 000000
2489 014360 001406

MOV #312,-2(R2) ;MOVE TO MAILBOX # ***** 312 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REGISTER NOT INCREMENTED BY ONE
MBDM2F: CMP #125125,@#0 ;CHECK DATA
BEQ TST147

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 721 <

C14362	012762	000313	177776	MOV	#313,-2(R2)	;MOVE TO MAILBOX # ***** 313 *****
014370	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
014374	000000			HALT		;DESTINATION DATA INCORRECT
						; OR SEQUENCE ERROR

2490
2491
2492
2493
2494
2495

.....
: THIS TEST VERIFIES MOV DESTINATION MODE 3. R0 IS USED TO PICK UP
: AN ADDRESS AT LOC. 400. LOC 400 POINTS TO LOC. 0 THE EFFECTIVE DEST. ADDR.. ALSO, MOV8
: INST. ARE USED W/ EVEN AND ODD BYTES TO CHECK MOV BYTES INST AND MODE 37 DESTINATIONS.

2496

.SBTTL TEST # 147 - TEST MOV(B) DESTINATION MODE 3

 :TEST 147 - TEST MOV(B) DESTINATION MODE 3

014376 005212
 014400 022712 000147
 014404 001071
 2497 014406 012700 000400
 2498 014412 005010
 2499 014414 005037 000000
 2500 014420 012730 125252
 2501 014424 102402
 2502 014426 001401
 2503 014430 100406

TST147: INC (R2) ;UPDATE TEST NUMBER
 CMP #147,(R2) ;SEQUENCE ERROR?
 BNE TST150-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #400,R0 ;R0=400
 CLR (R0) ;LOC. 400 POINTS TO LOC. 0
 CLR @#0 ;LOC. 0=0
 MOV #125252,@(R0)+ ;TRY MOV DESTINATION MODE 2
 BVS MDM3A ;BR TO ERROR IF V SET
 BEQ MDM3A ;BR TO ERROR IF Z SET
 BMI MDM3B

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <- -
 : CONDITIONAL BRANCH INST. AND <- -
 : REPLACE THE MOVE INSTRUCTION <- -
 : WHICH FOLLOWS W/ 765 <- -

014432
 014432 012762 000314 177776
 014440 005262 177774
 014444 000000
 2504 014446 022700 000402
 2505 014452 001406

MDM5A: MOV #314,-2(R2) ;MOVE TO MAILBOX # ***** 314 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;CC'S INCORRECT
 MDM3B: CMP #402,R0 ;CHECK DEST. MODE REGISTER
 BEQ MDM3C

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 754 <

014454 012762 000315 177776
 014462 005262 177774
 014466 000000
 2506 014470 022737 125252 000000
 2507 014476 001406

MOV #315,-2(R2) ;MOVE TO MAILBOX # ***** 315 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REGISTER NOT INCREMENTED BY 2
 MDM3C: CMP #125252,@#0 ;CHECK DESTINATION DATA
 BEQ MDM3D

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 742 <

014500 012762 000316 177776
 014506 005262 177774
 014512 000000
 2508 014514 112737 000125 000000
 2509 014522 022737 125125 000000
 2510 014530 001406

MOV #316,-2(R2) ;MOVE TO MAILBOX # ***** 316 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DESTINATION DATA INCORRECT
 MDM3D: MOVB #125,@#0 ;TRY MOV(B) DESTINATION MODE 2 EVEN BYTE
 CMP #125125,@#0 ;CHECK DATA
 BEQ MDM3E

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <- -
 : CONDITIONAL BRANCH INST. AND <- -
 : REPLACE THE MOVE INSTRUCTION <- -
 : WHICH FOLLOWS W/ 725 <- -

014532 012762 000317 177776
 014540 005262 177774
 014544 000000
 2511 014546 112737 000525 000001
 2512 014554 022737 052525 000000
 2513 014562 001406

MOV #317,-2(R2) ;MOVE TO MAILBOX # ***** 317 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DESTINATION DATA INCORRECT
 MDM3E: MOVB #525,@#1 ;TRY MOV(B) DESTINATION MODE 2 ODD BYTE
 CMP #52525,@#0 ;CHECK DATA
 BEQ TST150

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 710 <====

014564	012762	000320	177776	MOV	#320,-2(R2)	;MOVE TO MAILBOX #	320
014572	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR		
014576	000000			HALT				

2514
2515
2516
2517
2518
2519
2520
2521

.....
: THIS TEST VERIFIES THE MOV DESTINATION MODE 4 INSTRUCTION.
: SOP INSTRUCTIONS ON R0 ARE USED TO CLEAR TARGET LOCATION 0.
: R4 IS USED AS THE MODE 4 ADDRESSING REGISTER, AND
: CONDITIONAL BRANCHES ARE USED TO VERIFY THE DATA.
:

2522

.SBTTL TEST # 150 - TEST MOV DESTINATION MODE 4

TEST 150 - TEST MOV DESTINATION MODE 4

014600 005212
014602 022712 000150
014606 001034
2523 014610 005000
2524 014612 005010
2525 014614 012704 000002
2526 014620 012744 012345
2527 014624 102402
2528 014626 001401
2529 014630 100006

TST150: INC (R2) ;UPDATE TEST NUMBER
CMP #150,(R2) ;SEQUENCE ERROR?
BNE TST151-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC 0=0
MOV #2,R4 ;R4=2
MOV #12345,-(R4) ;TRY MOV DEST. MODE 4
BVS MDM4A ;BR TO ERROR IF V-BIT SET
BEQ MDM4A ;BR TO ERROR IF Z-BIT SET
BPL MDM4B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 766 <====

014632
014632 012762 000321 177776
014640 005262 177774
014644 000000
2530 014646 005704
2531 014650 001406

MDM4A: MOV #321,-2(R2) ;MOVE TO MAILBOX # ***** 321 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CC'S NOT CORRECT
MDM4B: TST R4 ;CHECK DECREMENTING OF MODE 4 REG.
BEQ MDM4C

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 756 <====

014652 012762 000322 177776
014660 005262 177774
014664 000000
2532 014666 022710 012345
2533 014672 001406

MDM4C: MOV #322,-2(R2) ;MOVE TO MAILBOX # ***** 322 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DESTINATION MODE REGISTER NOT DECREMENTED BY 2
CMP #12345,(R0) ;CHECK DESTINATION DATA
BEQ TST151

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 745 <====

014674 012762 000323 177776
014702 005262 177774
014706 000000

MDM4D: MOV #323,-2(R2) ;MOVE TO MAILBOX # ***** 323 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DESTINATION DATA INCORRECT
; OR SEQUENCE ERROR

2534
2535
2536
2537
2538
2539
2540
2541
2542

THIS TEST VERIFIES THE MOV B DESTINATION MODE 4 INSTRUCTION
ON BOTH ODD AND EVEN BYTES. SOP INSTRUCTIONS ON R4 ARE
USED TO CLEAR TARGET LOCATION 0. R0 IS USED AS THE MODE 4
ADDRESSING REGISTER, AND CMP AND CONDITIONAL BRANCH
INSTRUCTIONS ARE USED TO VERIFY THE DATA.

```

2543          .SBTTL TEST # 151 - TEST MOV B DESTINATION MODE 4
              :*****
              :TEST 151 - TEST MOV B DESTINATION MODE 4
              :*****
014710 005212          TST151: INC      (R2)          ;UPDATE TEST NUMBER
014712 022712 000151  CMP      #151,(R2)      ;SEQUENCE ERROR?
014716 001060          BNE      TST152-10      ;BR TO ERROR HALT ON SEQ ERROR
2544 014720 005004          CLR      R4          ;R4=0
2545 014722 005014          CLR      (R4)         ;LOC. 0=0
2546 014724 012700 000002  MOV      #2,R0        ;R0 = 2
2547 014730 112740 125125  MOV B   #125125,-(R0) ;TRY MOV B DEST. MODE 4-ODD BYTE
2548 014734 020027 000001  CMP      R0,#1        ;CHECK THAT DEST. REG. WAS DECREMENTED
2549 014740 001406          BEQ      MBDM4A

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ; CONDITIONAL BRANCH INST. AND
              ; REPLACE THE MOVE INSTRUCTION
              ; WHICH FOLLOWS W/ 766
              ;
014742 012762 000324 177776  MOV      #324,-2(R2) ;MOVE TO MAILBOX # ***** 324 *****
014750 005262 177774          INC      -4(R2)      ;SET MSGTYP TO FATAL ERROR
014754 000000          HALT
2550 014756 021427 052400  MBDM4A: CMP      (R4),#52400 ;DESTINATION REG. NOT DECREMENTED BY 1
2551 014762 001406          BEQ      MBDM4B

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ; CONDITIONAL BRANCH INST. AND
              ; REPLACE THE MOVE INSTRUCTION
              ; WHICH FOLLOWS W/ 755
              ;
014764 012762 000325 177776  MOV      #325,-2(R2) ;MOVE TO MAILBOX # ***** 325 *****
014772 005262 177774          INC      -4(R2)      ;SET MSGTYP TO FATAL ERROR
014776 000000          HALT
2552 015000 112740 125125  MBDM4B: MOV B   #125125,-(R0) ;DEST. DATA NOT CORRECT
2553 015004 102402          BVS      MBDM4C      ;TRY MOV B DEST. MODE 4--EVEN BYTE
2554 015006 001401          BEQ      MBDM4C      ;BR. TO ERROR IF V-BIT SET
2555 015010 100006          BPL      MBDM4D      ;BR TO ERROR IF Z-BIT SET

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ; CONDITIONAL BRANCH INST. AND
              ; REPLACE THE MOVE INSTRUCTION
              ; WHICH FOLLOWS W/ 742
              ;
015012          MBDM4C:
015012 012762 000326 177776  MOV      #326,-2(R2) ;MOVE TO MAILBOX # ***** 326 *****
015020 005262 177774          INC      -4(R2)      ;SET MSGTYP TO FATAL ERROR
015024 000000          HALT
2556 015026 005700          MBDM4D: TST      R0        ;COND. CODES INCORRECT
2557 015030 001406          BEQ      MBDM4E      ;CHECK MODE 4 DEST. REGISTER

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ; CONDITIONAL BRANCH INST. AND
              ; REPLACE THE MOVE INSTRUCTION
              ; WHICH FOLLOWS W/ 732
              ;
015032 012762 000327 177776  MOV      #327,-2(R2) ;MOVE TO MAILBOX # ***** 327 *****
015040 005262 177774          INC      -4(R2)      ;SET MSGTYP TO FATAL ERROR
015044 000000          HALT
2558 015046 021427 052525  MBDM4E: CMP      (R4),#52525 ;DESTINATION REG NOT DECREMENTED BY 1
2559 015052 001406          BEQ      TST152      ;CHECK DEST. DATA

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
              ; CONDITIONAL BRANCH INST. AND
              ; REPLACE THE MOVE INSTRUCTION
              ; WHICH FOLLOWS W/ 721
              ;
015054 012762 000330 177776  MOV      #330,-2(R2) ;MOVE TO MAILBOX # ***** 330 *****
  
```


015062 005262 177774
015066 000000

INC -4(R2)
HALT

;SET MSGTYP TO FATAL ERROR
;DESTINATION DATA INCORRECT
; OR SEQUENCE ERROR

2560
2561
2562
2563
2564
2565
2566
2567
2568
2569

.....
: THIS TEST VERIFIES THE MOV DESTINATION MODE 5 AND THE MOV8
: DESTINATION MODE 5 - EVEN BYTE INSTRUCTIONS. R4 IS A
: POINTER TO TARGET LOCATION 0 AND R0 IS SETUP TO
: POINT TO LOCATION 376 FOR THE MOV, AND LOCATION 404 FOR
: THE MOV8 INSTRUCTIONS. CMP INSTRUCTIONS ARE USED TO VERIFY
: PROPER ADDRESSING AND DATA.
:

2570

.SBTTL TEST # 152 - TEST MOV DESTINATION MODE 5
:*****
:TEST 152 - TEST MOV DESTINATION MODE 5
:*****

015070 005212
015072 022712 000152
015076 001063
2571 015100 005004
2572 015102 005014
2573 015104 012700 000400
2574 015110 012750 004321
2575 015114 102402
2576 015116 001401
2577 015120 100006

TST152: INC (R2) ;UPDATE TEST NUMBER
CMP #152,(R2) ;SEQUENCE ERROR?
BNE TST153-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R4 ;R4=0
CLR (R4) ;LOC. 0 = 0
MOV #400,R0 ;R0=400
MOV #4321,@-(R0) ;TRY MOV DEST. MODE 5
BVS MDM5A ;BR TO ERROR IF V-BIT SET
BEQ MDM5A ;BR TO ERROR IF Z-BIT SET
BPL MDM5B
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-- ==
; CONDITIONAL BRANCH INST. AND <==--
; REPLACE THE MOVE INSTRUCTION <==--
; WHICH FOLLOWS W/ 766 <==--

015122
015122 012762 000331 177776
015130 005262 177774
015134 000000
2578 015136 022700 000376
2579 015142 001406

MDM5A: MOV #331,-2(R2) ;MOVE TO MAILBOX # ***** 331 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;COND. CODES INCORRECT
MDM5B: CMP #376,R0 ;CHECK MODE 5 REG. WAS DECREMENTED
BEQ MDM5C
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
; CONDITIONAL BRANCH INST. AND <=====
; REPLACE THE MOVE INSTRUCTION <=====
; WHICH FOLLOWS W/ 755 <=====
; ***** 332 *****

015144 012762 000332 177776
015152 005262 177774
015156 000000
2580 015160 022714 004321
2581 015164 001406

MDM5C: MOV #332,-2(R2) ;MOVE TO MAILBOX # ***** 332 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;MODE 5 REGISTER NOT DECREMENTED BY 2
CMP #4321,(R4) ;CHECK DEST. DATA
SEQ MDM5D
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <==--=
; CONDITIONAL BRANCH INST. AND <==--=
; REPLACE THE MOVE INSTRUCTION <==--=
; WHICH FOLLOWS W/ 744 <==--=
; ***** 333 *****

015166 012762 000333 177776
015174 005262 177774
015200 000000
2582 015202 012700 000406
2583 015206 112750 000377
2584 015212 022700 000404
2585 015216 001406

MDM5D: MOV #333,-2(R2) ;MOVE TO MAILBOX # ***** 333 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DEST. DATA INCORRECT
MOV #406,R0 ;R0=406
MOV #377,@-(R0) ;TRY MOV DEST. MODE 5 --EVEN BYTE
CMP #404,R0 ;CHECK MODE 5 REG.
BEQ MDM5E
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
; CONDITIONAL BRANCH INST. AND <===
; REPLACE THE MOVE INSTRUCTION <===
; WHICH FOLLOWS W/ 727 <===
; ***** 334 *****

015220 012762 000334 177776
015226 005262 177774
015232 000000
2586 015234 022714 177721
2587 015240 001406

MDM5E: MOV #334,-2(R2) ;MOVE TO MAILBOX # ***** 334 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;MODE 5 REGISTER NOT DECREMENTED BY 2
CMP #177721,(R4) ;CHECK DEST. DATA
BEQ TST153
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-- ==
; CONDITIONAL BRANCH INST. AND <-- ==
; REPLACE THE MOVE INSTRUCTION <-- ==
; WHICH FOLLOWS W/ 716 <-- ==

```
015242 012762 000335 177776      MOV    #335,-2(R2)      ;MOVE TO MAILBOX # ***** 335 *****  
015250 005262 177774              INC    -4(R2)          ;SET MSGTYP TO FATAL ERROR  
015254 000000                    HALT                    ;DEST. DATA INCORRECT  
                                   ; OR SEQUENCE ERROR
```

2588
2589
2590
2591
2592
2593
2594
2595
2596

.....
: THIS TEST VERIFIES THE MOV DESTINATION MODE 6 AND MOVB - EVEN BYTE
: DESTINATION MODE 6 INSTRUCTIONS. R0 IS USED TO SETUP TARGET LOC.0
: FOR BOTH TESTS. PATTERNS OF ONES AND ZEROES ARE MOVED INTO LOC.0
: BY MODE 6 INSTRUCTIONS, AND CMP INSTRUCTIONS ARE USED TO VERIFY
: PROPER ADDRESSING AND DATA.
:

2597

.SBTTL TEST # 153 - TEST MOV DESTINATION MODE 6

:TEST 153 - TEST MOV DESTINATION MODE 6

015256	005212			TST153:	INC	(R2)		;UPDATE TEST NUMBER	
015260	022712	000153			CMP	#153,(R2)		;SEQUENCE ERROR?	
015264	001066				BNE	TST154-10		;BR TO ERROR HALT ON SEQ ERROR	
2598 015266	005000				CLR	R0		;R0=0	
2599 015270	005010				CLR	(R0)		;LOC. 0=0	
2600 015272	005200				INC	R0		;R0=1	
2601 015274	012760	052525	177777		MOV	#052525,-1(R0)		;TRY MOV DEST. MODE 6	
2602 015302	102402				BVS	MDM6A		;BR TO ERROR IF V-BIT SET	
2603 015304	001401				BEQ	MDM6A		;BR TO ERROR IF Z-BIT SET	
2604 015306	100006				RPL	MDM6B			
								; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<====
								CONDITIONAL BRANCH INST. AND	<====
								REPLACE THE MOVE INSTRUCTION	<====
								WHICH FOLLOWS W/ 766	<====
015310				MDM6A:					
015310	012762	000336	177776		MOV	#336,-2(R2)		;MOVE TO MAILBOX # ***** 336 *****	
015316	005262	177774			INC	-4(R2)		;SET MSGTYP TO FATAL ERROR	
015322	000000				HALT			;COND. CODES INCORRECT	
2605 015324	022700	000001		MDM6B:	CMP	#1,R0		;CHECK DEST. REGISTER UNALTERED	
2606 015330	001406				BEQ	MDM6C			
								; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<== =
								CONDITIONAL BRANCH INST. AND	<= --
								REPLACE THE MOVE INSTRUCTION	<= --
								WHICH FOLLOWS W/ 755	<= --
015332	012762	000337	177776		MOV	#337,-2(R2)		;MOVE TO MAILBOX # ***** 337 *****	
015340	005262	177774			INC	-4(R2)		;SET MSGTYP TO FATAL ERROR	
015344	000000				HALT			;DEST. REGISTER INCORRECTLY ALTERED	
2607 015346	022737	052525	000000	MDM6C:	CMP	#52525,@#0		;CHECK DEST. DATA	
2608 015354	001406				BEQ	MDM6D			
								; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<= -- =
								CONDITIONAL BRANCH INST. AND	<= --
								REPLACE THE MOVE INSTRUCTION	<= --
								WHICH FOLLOWS W/ 743	<= --
015356	012762	000340	177776		MOV	#340,-2(R2)		;MOVE TO MAILBOX # ***** 340 *****	
015364	005262	177774			INC	-4(R2)		;SET MSGTYP TO FATAL ERROR	
015370	000000				HALT			;DEST. DATA INCORRECT	
2609 015372	012700	000002		MDM6D:	MOV	#2,R0		;R0=2	
2610 015376	112760	000377	177777		MOVB	#377,-1(R0)		;TRY MOVB DEST. MODE 6	
2611 015404	022700	000002			CMP	#2,R0		;CHECK DEST. REGISTER UNALTERED	
2612 015410	001406				BEQ	MDM6E			
								; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<= -- =
								CONDITIONAL BRANCH INST. AND	<= --
								REPLACE THE MOVE INSTRUCTION	<= --
								WHICH FOLLOWS W/ 725	<= --
015412	012762	000341	177776		MOV	#341,-2(R2)		;MOVE TO MAILBOX # ***** 341 *****	
015420	005262	177774			INC	-4(R2)		;SET MSGTYP TO FATAL ERROR	
015424	000000				HALT			;DEST. REGISTER INCORRECTLY ALTERED	
2613 015426	022737	177525	000000	MDM6E:	CMP	#177525,@#0		;CHECK DEST. DATA	
2614 015434	001406				BEQ	TST154			
								; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<= -- =
								CONDITIONAL BRANCH INST. AND	<= -- =
								REPLACE THE MOVE INSTRUCTION	<= -- =
								WHICH FOLLOWS W/ 713	<= -- =

015436	012762	000342	177776	MOV	#342,-2(R2)	;MOVE TO MAILBOX # ***** 342 *****
015444	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
015450	000000			HALT		;DEST. DATA INCORRECT ; OR SEQUENCE ERROR

2615
2616
2617
2618
2619
2620
2621
2622

: THIS TEST VERIFIES THE MOV DESTINATION MODE 7 AND MOVB - ODD BYTE
: DESTINATION MODE 7 INSTRUCTIONS. R4 POINTS TO TARGET LOC.0 AND R0
: IS USED AS THE MODE 7 ADDRESSING REGISTER. CMP INSTRUCTIONS ARE
: USED TO VERIFY PROPER ADDRESSING AND DATA.
:

2623

.SBTTL TEST # 154 - TEST MOV DESTINATION MODE 7

 :TEST 154 - TEST MOV DESTINATION MODE 7

015452 005212
 015454 022712 000154
 015460 001065
 2624 015462 005004
 2625 015464 005014
 2626 015466 012700 000403
 2627 015472 012770 070707 177777
 2628 015500 102402
 2629 015502 001401
 2630 015504 100006

TST154: INC (R2)
 CMP #154,(R2)
 BNE TST155-10
 CLR R4
 CLR (R4)
 MOV #403,R0
 MOV #70707,@-1(R0)
 BVS MDM7A
 BEQ MDM7A
 BPL MDM7B

:UPDATE TEST NUMBER
 :SEQUENCE ERROR?
 :BR TO ERROR HALT ON SEQ ERROR
 :R4=0
 :LOC.0=0
 :R0=403
 :TRY MOV W/DEST MODE 7
 :BR. TO ERROR IF V-BIT SET
 :BR TO ERROR IF Z-BIT SFT

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 765 <====

015506
 015506 012762 000343 177776
 015514 005262 177774
 015520 000000
 2631 015522 022700 000403
 2632 015526 001406

MDM7A: MOV #343,-2(R2)
 INC -4(R2)
 HALT
 MDM7B: CMP #403,R0
 BEQ MDM7C

:MOVE TO MAILBOX # ***** 343 *****
 :SET MSGTYP TO FATAL ERROR
 :COND. CODES INCORRECT
 :CHECK DEST. REGISTER

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 754 <====

015530 012762 000344 177776
 015536 005262 177774
 015542 000000
 2633 015544 022737 070707 000000
 2634 015552 001406

MDM7C: MOV #344,-2(R2)
 INC -4(R2)
 HALT
 MDM7C: CMP #70707,@#0
 BEQ MDM7D

:MOVE TO MAILBOX # ***** 344 *****
 :SET MSGTYP TO FATAL ERROR
 :DEST. REGISTER INCORRECTLY ALTERED
 :CHECK DEST. DATA

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <--
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 742 <--

015554 012762 000345 177776
 015562 005262 177774
 015566 000000
 2635 015570 112770 107070 000001
 2636 015576 022700 000403
 2637 015602 001406

MDM7D: MOV #345,-2(R2)
 INC -4(R2)
 HALT
 MDM7D: MOV #107070,@1(R0)
 CMP #403,R0
 BEQ MDM7E

:MOVE TO MAILBOX # ***** 345 *****
 :SET MSGTYP TO FATAL ERROR
 :DEST. DATA INCORRECT
 :TRY MOVW W/DEST MODE 7--ODD BYTE
 :CHECK MODE 7 DEST. REG.

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
 : CONDITIONAL BRANCH INST. AND <--
 : REPLACE THE MOVE INSTRUCTION <--
 : WHICH FOLLOWS W/ 726 <--

015604 012762 000346 177776
 015612 005262 177774
 015616 000000
 2638 015620 022737 034307 000000
 2639 015626 001406

MDM7E: MOV #346,-2(R2)
 INC -4(R2)
 HALT
 MDM7E: CMP #34307,@#0
 BEQ TST155

:MOVE TO MAILBOX # ***** 346 *****
 :SET MSGTYP TO FATAL ERROR
 :DEST. DATA INCORRECT
 :CHECK DEST. DATA

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 714 <

015630 012762 000347 177776

MOV #347,-2(R2)

:MOVE TO MAILBOX # ***** 347 *****

015636 005262 177774
015642 000000

INC -4(R2)
HALT

:SET MSGTYP TO FATAL ERROR
:DESTINATION DATA INCORRECT
: OR SEQUENCE ERROR

2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652

:*****
:
: THIS TEST VERIFIES MODE 4 DOUBLE OPERAND INSTRUCTIONS.
: THE TEST USES MODE 4 ADDRESSING WITH REGISTER 0 TO MOVE THRU A
: TABLE OF OPERANDS. THE TABLE OF OPERANDS AND THE WORK LOCATION IS
: STORED FOLLOWING THE TEST CODE. A SERIES OF 5 DOP INSTRUCTIONS UTILIZES
: THE DATA IN THE TABLE TO CYCLE THE WORK LOCATION THRU A SET OF
: VALUE. THE DATA HAS BEEN CHOSEN TO INSURE THAT NO SINGLE ERROR WILL
: GO UNDETECTED. WORD AND BYTE INSTRUCTION ACCESSING BOTH EVEN AND
: ODD ADDRESSES ARE USED IN THE TEST. THE LISTING SHOWS THE
: EXPECTED INTERMEDIATE RESULT AS EACH INSTRUCTION IS EXECUTED.
:

2654

.SBTTL TEST # 155 - TEST MODE 4 W/ DOP INSTS.

:TEST 155 - TEST MODE 4 W/ DOP INSTS.

015644 005212
015646 022712 000155
015652 001015
2654 015654 012700 015732
2655 015660 014037 015732
2656 015664 064037 015732
2657 015670 144037 015732
2658 015674 154037 015732
2659 015700 024037 015732
2660 015704 001413

TST155: INC (R2) ;UPDATE TEST NUMBER
CMP #155,(R2) ;SEQUENCE ERROR?
BNE DOP4 ;BR TO ERROR HALT ON SEQ ERROR
MOV #TBL1,R0 ;INITIALIZE R0
MOV -(R0),@#TBL1 ;TBL1=125252
ADD -(R0),@#TBL1 ;TBL1=000377
BICB -(R0),@#TBL1 ;TBL1=000252
BISB -(R0),@#TBL1+1 ;TBL1=125252
CMP -(R0),@#TBL1 ;CHECK RESULT
BEQ TST156

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 762 <====

015706
015706 012762 000350 177776
015714 005262 177774
015720 000000

DOP4: MOV #350,-2(R2) ;MOVE TO MAILBOX # ***** 350 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF MODE 4 INSTS. INCORRECT
; OR SEQUENCE ERROR

2661
2662 015722 125252
2663 015724 052652
2664 015726 053125
2665 015730 125252
2666 015732 000000
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676

125252
52652
53125
125252
TBL1: 0

: THIS TEST VERIFIES MODE 5 DOUBLE OPERAND INSTRUCTIONS.
: THE TEST USES AN ADDRESS TABLE STORED FOLLOWING THE TEST CODE.
: THIS TABLE IS SIMPLY A TABLE OF ADDRESS POINTERS WHICH ADDRESS
: THE DATA TABLE USED IN THE PREVIOUS TEST. THE TEST IS IDENTICAL TO
: THE PREVIOUS TEST EXCEPT THE DATA IS REFERENCED USING THIS ADDRESS
: TABLE AND MODE 5 ADDRESSING. (SEE PREVIOUS TEST).
:

2677

.SBTTL TEST # 156 - TEST MODE 5 W/ DOP INSTS.

 :TEST 156 -- TEST MODE 5 W/ DOP INSTS.

015734 005212
 015736 022712 000156
 015742 001015
 2678 015744 012700 016024
 2679 015750 015037 015732
 2680 015754 065037 015732
 2681 015760 145037 015732
 2682 015764 155037 015733
 2683 015770 025037 015732
 2684 015774 001413

TST156: INC (R2) ;UPDATE TEST NUMBER
 CMP #156,(R2) ;SEQUENCE ERROR?
 BNE DOP5 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #TBL2+2,R0 ;INITIALIZE R0
 MOV @-(R0),@#TBL1 ;TBL1=125252
 ADD @-(R0),@#TBL1 ;TBL1=000377
 BICB @-(R0),@#TBL1 ;TBL1=000252
 BISB @-(R0),@#TBL1+1 ;TBL1=125252
 CMP @-(R0),@#TBL1 ;CHECK RESULT
 BEQ TST157

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 762 <====

015776
 015776 012762 000351 177776
 016004 005262 177774
 016010 000000

DOP5: MOV #351,-2(R2) ;MOVE TO MAILBOX # ***** 351 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;RESULT OF MODE 5 INSTS. INCORRECT
 ; OR SEQUENCE ERROR

2685 016012 015722
 2686 016014 015724
 2687 016016 015725
 2688 016020 015726
 2689 016022 015730
 2690
 2691
 2692
 2693
 2694
 2695
 2696
 2697
 2698
 2699

TBL1-10
 TBL1-6
 TBL1-5
 TBL1-4
 TBL2: TBL1-2

 : THIS TEST VERIFIES MODE 6 DOUBLE OPERAND INSTRUCTIONS.
 : IT USES THE SAME DATA AS THAT USED IN THE MODE 4 TESTS.
 : THIS TIME THE DATA IS ACCESSED USING MODE 6. R0 IS SET
 : TO POINT TO THE MIDDLE OF THE TABLE. THE TABLE IS ACCESSED FROM
 : BOTTOM TO TOP BY VARYING THE OFFSET IN THE MODE 6 INSTRUCTIONS.
 : THE DATA RESULTS ARE IDENTICAL TO THOSE EXPECTED IN THE MODE 4
 : TESTS.
 :

2700

.SBTTL TEST # 157 - TEST MODE 6 W/ DOP INSTS.

.....
:TEST 157 - TEST MODE 6 W/ DOP INSTS.
:.....

016024	005212		
016026	022712	000157	
016032	001024		
2701 016034	012700	015726	
2702 016040	016037	000002	015732
2703 016046	066037	000000	015732
2704 016054	146037	177777	015732
2705 016062	156037	177776	015733
2706 016070	026037	177774	015732
2707 016076	001406		

```

TST157: INC      (R2)          ;UPDATE TEST NUMBER
        CMP      #157,(R2)    ;SEQUENCE ERROR?
        BNE      TST160-10    ;BR TO ERROR HALT ON SEQ ERROR
        MOV      #TBL1-4,R0    ;INITIALIZE R0
        MOV      2(R0),@TBL1   ;TBL1=125252
        ADD      0(R0),@TBL1   ;TBL1=000377
        BICB    -1(R0),@TBL1   ;TBL1=000252
        BISB    -2(R0),@TBL1+1 ;TBL1=125252
        CMP      -4(R0),@TBL1  ;CHECK RESULT
        BEQ      TST160

```

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
;           CONDITIONAL BRANCH INST. AND <====
;           REPLACE THE MOVE INSTRUCTION <====
;           WHICH FOLLOWS W/ 755 <====
; MOVE TO MAILBOX # ***** 352 *****
; SET MSGTYP TO FATAL ERROR
; RESULT OF MODE 6 INSTS. INCORRECT
; OR SEQUENCE ERROR

```

016100	012762	000352	177776
016106	005262	177774	
016112	000000		

```

MOV      #352,-2(R2)
INC      -4(R2)
HALT

```

2708
2709
2710
2711
2712
2713
2714
2715
2716
2717

```

.....
: THIS TEST VERIFIES MODE 7 DOUBLE OPERAND INSTRUCTIONS.
: THIS TEST USES THE SAME ADDRESS TABLE AND DATA TABLE USED BY
: THE MODE 5 TESTS. THIS TIME THE DATA IS ACCESSED USING MODE 7.
: R0 IS SET TO POINT TO THE MIDDLE OF THE ADDRESS TABLE IN THE MODE 5
: TEST. THE TABLE IS ACCESSED FROM BOTTOM TO TOP BY VARYING THE OFFSET
: IN THE MODE 7 INSTRUCTIONS. THE DATA RESULTS ARE IDENTICAL TO
: THOSE EXPECTED IN THE MODE 5 TESTS.
:

```

2718

.SBTTL TEST # 160 - TEST MODE 7 W/ DOP INSTS.

:TEST 160 - TEST MODE 7 W/ DOP INSTS.

016114 005212
016116 022712 000160
016122 001024
2719 016124 012700 016016
2720 016130 017037 000004 015732
2721 016136 067037 000002 015732
2722 016144 147037 000000 015732
2723 016152 157037 177776 015733
2724 016160 027037 177774 015732
2725 016166 001406

TST160: INC (R2) ;UPDATE TEST NUMBER
CMP #160,(R2) ;SEQUENCE ERROR?
BNE TST161-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #TBL2-4,R0 ;INITIALIZE R0
MOV @4(R0),@#TBL1 ;TBL1=125252
ADD @2(R0),@#TBL1 ;TBL1=000377
BICB @0(R0),@#TBL1 ;TBL1=000252
BISB @-2(R0),@#TBL1+1 ;TBL1-125252
CMP @-4(R0),@#TBL1 ;CHECK RESULT
BEQ TST161

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 755

016170 012762 000353 177776
016176 005262 177774
016202 000000

MOV #353,-2(R2) ;MOVE TO MAILBOX # ***** 353 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF MODE 7 INSTS INCORRECT
; OR SEQUENCE ERROR

2726
2727
2728
2729
2730
2731
2732
2733

: THIS TEST VERIFIES THE ROTATE MODE 0 INSTRUCTIONS.
: RO IS LOADED WITH A DATA PATTERN, THE C-BIT IS LOADED, AND
: AN ROL INSTRUCTION IS EXECUTED WITH MODE 0. THE OPERATION IS CHECKED
: BY TESTING THE RESULTING DATA AND THE STATE OF THE C AND V BITS.
: NEXT, THE SAME PROCEDURE IS EXECUTED TO TEST MODE 0 BYTE INSTRUCTIONS.
: *****

2734

..SBTTL TEST # 161 - TEST ROTATE INSTRUCTIONS OF MODE 0
 :*****
 :TEST 161 - TEST ROTATE INSTRUCTIONS OF MODE 0
 :*****

016204 005212
 016206 022712 000161
 016212 001032
 2735 016214 012700 125252
 2736 016220 000261
 2737 016222 006100
 2738 016224 102004
 2739 016226 103003
 2740 016230 022700 052525
 2741 016234 001406

TST161: INC (R2) ;UPDATE TEST NUMBER
 CMP #161,(R2) ;SEQUENCE ERROR?
 BNE TST162-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125252,R0 ;INITIALIZE DATA
 SEC ;SET C-BIT
 ROL R0 ;TRY ROL W/ MODE 0
 BVC R0A ;CC=0011
 BCC R0A
 CMP #05252,R0 ;CHECK DATA
 BEQ R0B
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 ; CONDITIONAL BRANCH INST. AND
 ; REPLACE THE MOVE INSTRUCTION
 ; WHICH FOLLOWS W/ 766

016236
 016236 012762 000354 177776
 016244 005262 177774
 016250 000000
 2742 016252 012700 125252
 2743 016256 000261
 2744 016260 106100
 2745 016262 102004
 2746 016264 103003
 2747 016266 022700 125125
 2748 016272 001406

ROTA: MOV #354,-2(R2) ;MOVE TO MAILBOX # ***** 354 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROL MODE 0 FAILED
 ROTOB: MOV #125252,R0 ;INITIALIZE DATA
 SEC ;SET C-BIT
 ROLB R0 ;TRY ROL W/ MODE 0 EVEN BYTE
 BVC ROTOC ;CC=0011
 BCC ROTOC
 CMP #125125,R0 ;CHECK DATA
 BEQ TST162
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 ; CONDITIONAL BRANCH INST. AND
 ; REPLACE THE MOVE INSTRUCTION
 ; WHICH FOLLOWS W/ 747

016274
 016274 012762 000355 177776
 016302 005262 177774
 016306 000000

ROTC: MOV #355,-2(R2) ;MOVE TO MAILBOX # ***** 355 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROLB MODE 0 FAILED
 ; OR SEQUENCE ERROR

2749
 2750
 2751
 2752
 2753
 2754
 2755
 2756
 2757
 2758

 : THIS TEST VERIFIES THE ROTATE MODE 1 INSTRUCTIONS.
 : THE DATA TO BE ROTATED IS IN LOC 0. R0 IS USED AS THE
 : ADDRESSING REGISTER. THE C-BIT IS LOADED AND AN ROL IS EXECUTED.
 : THE RESULTS ARE CHECKED BY COMPARING THE DATA RESULTS AND TESTING
 : THE C AND V BITS. THIS PROCEDURE IS THEN REPEATED TWICE MORE
 : TO TEST THE BYTE ROTATES. FIRST ON BYTE 0, THEN ON BYTE 1.
 :

.SBTTL TEST # 162 - TEST ROTATE INSTRUCTIONS W/ MODE 1

 :TEST 162 - TEST ROTATE INSTRUCTIONS W/ MODE 1

016310	005212			TST162:	INC	(R2)	:UPDATE TEST NUMBER
016312	022712	000162			CMP	#162,(R2)	:SEQUENCE ERROR?
016316	001057				BNE	TST163-10	:BR TO ERROR HALT ON SEQ ERROR
2760 016320	005000				CLR	R0	:POINT TO LOC. 0
2761 016322	012710	052525			MOV	#52525,(R0)	:INITIALIZE DATA
2762 016326	000241				CLC		:CLEAR C-BIT
2763 016330	006110				ROL	(R0)	:TRY ROL W/ MODE 1
2764 016332	102005				BVC	ROT1A	:CC=1010
2765 016334	103404				BCS	ROT1A	
2766 016336	023727	000000	125252		CMP	@#0,#125252	:CHECK RESULT
2767 016344	001406				BEQ	ROT1B	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 764

016346				ROT1A:			
016346	012762	000356	177776		MOV	#356,-2(R2)	:MOVE TO MAILBOX # ***** 356 *****
016354	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
016360	000000				HALT		:ROL MODE 1 FAILED
2768 016362	000261			ROT1B:	SEC		
2769 016364	012710	125252			MOV	#125252,(R0)	:INITIALIZE DATA
2770 016370	106110				ROLB	(R0)	:TRY ROLB W/ MODE 1 EVEN BYTE
2771 016372	102005				BVC	ROT1C	:CC=1011
2772 016374	103004				BCC	ROT1C	
2773 016376	022737	125125	000000		CMP	#125125,@#0	:TEST RESULT
2774 016404	001406				BEQ	ROT1D	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 744

016406				ROT1C:			
016406	012762	000357	177776		MOV	#357,-2(R2)	:MOVE TO MAILBOX # ***** 357 *****
016414	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
016420	000000				HALT		:ROLB W/ MODE 1 EVEN BYTE FAILED
2775 016422	012710	125252		ROT1D:	MOV	#125252,(R0)	:POINT TO ODD BYTE
2776 016426	005000				CLR	R0	
2777 016430	005200				INC	R0	
2778 016432	000261				SEC		:SET C-BIT
2779 016434	106110				ROLB	(R0)	:TRY ROLB W/ MODE 1 ODD BYTE
2780 016436	102005				BVC	ROT1E	:CC=0011
2781 016440	103004				BCC	ROT1E	
2782 016442	022737	052652	000000		CMP	#052652,@#0	:CHECK DATA
2783 016450	001406				BEQ	TST163	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 722

016452				ROT1E:			
016452	012762	000360	177776		MOV	#360,-2(R2)	:MOVE TO MAILBOX # ***** 360 *****
016460	005262	177774			INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
016464	000000				HALT		:ROLB W/ MODE 1 ODD BYTE FAILED
							: OR SEQUENCE ERROR

2786
2787
2788
2789
2790
2791

.....
: THIS TEST VERIFIES MODE 2 ROTATE INSTRUCTIONS.
: THE SAME PROCEDURE AS IN THE OTHER ROTATE TESTS ARE USED. RC
: IS USED AS THE ADDRESSING REGISTER AND IS CHECKED FOR PROPER
: INCREMENTING. BYTE INSTRUCTIONS ARE ALSO CHECKED.
:

```
2792 .SBTTL TEST # 163 - TEST ROTATE INSTRUCTIONS W/ MODE 2
:*****
:TEST 163 - TEST ROTATE INSTRUCTIONS W/ MODE 2
:*****
016466 005212 TST163: INC (R2) ;UPDATE TEST NUMBER
016470 022712 000163 CMP #163,(R2) ;SEQUENCE ERROR?
016474 001065 BNE TST164-10 ;BR TO ERROR HALT ON SEQ ERROR
2793 016476 005000 CLR R0 ;POINT TO LOC 0
2794 016500 012710 173737 MOV #173737,(R0) ;INITIALIZE DATA
2795 016504 000241 CLC ;CLEAR C-BIT
2796 016506 006120 ROL (R0)+ ;TRY ROL W/ MODE 2
2797 016510 103007 BCC ROT2A ;CHECK C-BIT
2798 016512 022737 167676 000000 CMP #167676,@#0 ;CHECK DATA
2799 016520 001003 BNE ROT2A ;BRANCH IF RESULT INCORRECT
2800 016522 005300 DEC R0 ;TEST R0
2801 016524 005300 DEC R0
2802 016526 001406 BEQ ROT2B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
; CONDITIONAL BRANCH INST. AND <---
; REPLACE THE MOVE INSTRUCTION <-
; WHICH FOLLOWS W/ 762 <-

016530 ROT2A:
016530 012762 000361 177776 MOV #361,-2(R2) ;MOVE TO MAILBOX # ***** 361 *****
016536 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
016542 000000 HALT ;ROL W/ MODE 2 FAILED
2803 016544 005000 ROT2B: CLR R0 ;POINT TO LOC 0
2804 016546 012710 004040 MOV #4040,(R0) ;INITIALIZE DATA
2805 016552 000241 CLC ;CLEAR C-BIT
2806 016554 106120 ROLB (R0)+ ;TRY ROLB W/ MODE 2 EVEN BYTE
2807 016556 103406 BCS ROT2C ;CHECK C-BIT
2808 016560 022737 004100 000000 CMP #4100,@#0 ;CHECK DATA
2809 016566 001002 BNE ROT2C ;BRANCH IF DATA INCORRECT
2810 016570 005300 DEC R0 ;CHECK R0
2811 016572 001406 BEQ ROT2D

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 740 <

016574 ROT2C:
016574 012762 000362 177776 MOV #362,-2(R2) ;MOVE TO MAILBOX # ***** 362 *****
016602 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
016606 000000 HALT ;ROLB W/ MODE 2 EVEN BYTE FAILED
2812 016610 005000 ROT2D: CLR R0 ;POINT TO LOC 0
2813 016612 012710 004040 MOV #4040,(R0) ;INITIALIZE DATA
2814 016616 005200 INC R0 ;POINT TO ODD BYTE OF DATA
2815 016620 000261 SEC ;SET C-BIT
2816 016622 106120 ROLB (R0)+ ;TRY ROL W/ MODE 2 ODD BYTE
2817 016624 103407 BCS ROT2E ;CHECK C-BIT
2818 016626 022737 010440 000000 CMP #10440,@#0 ;CHECK DATA
2819 016634 001003 BNE ROT2E ;BRANCH IF DATA INCORRECT
2820 016636 005300 DEC R0 ;CHECK R0
2821 016640 005300 DEC R0
2822 016642 001406 BEQ TST164

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
; CONDITIONAL BRANCH INST. AND <---
; REPLACE THE MOVE INSTRUCTION <---
; WHICH FOLLOWS W/ 714 <---
```

016644				ROT2E:			
016644	012762	000363	177776	MOV	#363,-2(R2)	:MOVE TO MAILBOX # ***** 363 *****	
016652	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
016656	000000			HALT		:ROLB W/ MODE 2 ODD BYTE FAILED	
						: OR SEQUENCE ERROR	

2823
2824
2825
2826
2827
2828
2829
2830

.....
: THIS TEST VERIFIES MODE 3 ROTATE INSTRUCTIONS.
: THIS TEST USES THE SAME PROCEDURES AS IN THE OTHER ROTATE
: TESTS. THE DATA IS STORED IN LOC. 0 AND IS ADDRESSSED USING
: MODE 37. BYTE ADDRESSING IS ALSO CHECKED FOR E.FM AND ODD BYTES.
:


```

2831 .SBTTL TEST # 164 - TEST ROTATE INSTRUCTIONS /W MODE 3
:*****
:TEST 164 - TEST ROTATE INSTRUCTIONS /W MODE 3
:*****
      016660 005212          TST164: INC      (R2)          ;UPDATE TEST NUMBER
      016662 022712 000164    CMP      #164,(R2)        ;SEQUENCE ERROR?
      016666 001057          BNE      TST165-10       ;BR TO ERROR HALT ON SEQ ERROR
2832 016670 012737 052525 000000 MOV      #52525,@#0      ;INITIALIZE DATA IN LOC 0
2833 016676 000261          SEC                      ;SET C-BIT
2834 016700 006137 000000    ROL      @#0             ;TRO ROL W/ MODE 3
2835 016704 103404          BCS      ROT3A           ;CHECK C-BIT
2836 016706 022737 125253 000000 CMP      #125253,@#0     ;CHECK DATA
2837 016714 001406          BEQ      ROT3B

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 764 <====

      016716          ROT3A: MOV      #364,-2(R2)      ;MOVE TO MAILBOX # ***** 364 *****
      016716 012762 000364 177776 INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
      016724 005262 177774    HALT                     ;ROL W/ MODE 3 FAILED
2838 016732 012737 125252 000000 ROT3B: MOV      #125252,@#0    ;INITIALIZE DATA
2839 016740 000241          CLC                      ;CLEAR C-BIT
2840 016742 106137 000000    ROLB     @#0             ;TRY ROL W/ MODE 3 EVEN BYTE
2841 016746 103004          BCC      ROT3C           ;CHECK C-BIT
2842 016750 023727 000000 4$: CMP      @#0,#125124      ;CHECK DATA
2843 016756 001406          BEQ      ROT3D

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 743 <====

      016760          ROT3C: MOV      #365,-2(R2)      ;MOVE TO MAILBOX # ***** 365 *****
      016760 012762 000365 177776 INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
      016766 005262 177774    HALT                     ;ROL W/ MODE 3 EVEN BYTE FAILED
2844 016774 012737 125252 000000 ROT3D: MOV      #125252,@#0    ;INITIALIZE DATA IN LOC. 0
2845 017002 000261          SEC                      ;SET C-BIT
2846 017004 106137 000001    ROLB     @#1             ;TRY ROL W/ MODE 3 ODD BYTE
2847 017010 103004          BCC      ROT3E           ;CHECK C-BIT
2848 017012 022737 052652 000000 CMP      #052652,@#0     ;CHECK DATA
2849 017020 001406          BEQ      TST165

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 722 <====

      017022          ROT3E: MOV      #366,-2(R2)      ;MOVE TO MAILBOX # ***** 366 *****
      017022 012762 000366 177776 INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
      017030 005262 177774    HALT                     ;ROL W/ MODE 3 ODD BYTE FAILED
      017034 000000          ; OR SEQUENCE ERROR

2850
2851
2852
2853
2854
2855
2856
:*****
: THIS TEST VERIFIES MODE 4 ROTATE INSTRUCTIONS. THE DATA IS
: STORED IN LOC. 0. R0 IS SET TO 2 AND THE CARRY IS SET. AN ROL MODE 4
: IS USED TO ROTATE LOCATION 0 USING R0. THE DATA IS CHECKED
: AND THE C AND V BITS ARE TESTED. THE PROPER DECREMENTING OF
  
```

2857
2858

:RO IS VERIFIED.
:

2859

.SBTTL TEST # 165 - TEST MODE 4 W/ ROTATE INSTRUCTIONS

 :TEST 165 - TEST MODE 4 W/ ROTATE INSTRUCTIONS

017036 005212
 017040 022712 000165
 017044 001020
 2860 017046 012737 070707 000000
 2861 017054 012700 000002
 2862 017060 000261
 2863 017062 006140
 2864 017064 103406
 2865 017066 022737 161617 000000
 2866 017074 001002
 2867 017076 005700
 2868 017100 001406

TST165: INC (R2) ;UPDATE TEST NUMBER
 CMP #165,(R2) ;SEQUENCE ERROR?
 BNE TST166-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #070707,@#0 ;INITIALIZE DATA IN LOC. 0
 MOV #2,R0 ;INITIALIZE R0 AS POINTER
 SEC ;SET C-BIT
 ROL -(R0) ;TRY ROL W/ MODE 4
 BCS ROT4 ;CHECK C-BIT
 CMP #161617,@#0 ;CHECK DATA
 BNE ROT4 ;BRANCH IF DATA INCORRECT
 TST R0 ;CHECK MODE 4 REGISTER
 BEQ TST166

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 ; CONDITIONAL BRANCH INST. AND <-
 ; REPLACE THE MOVE INSTRUCTION <-
 ; WHICH FOLLOWS W/ 761 <-

017102
 017102 012762 000367 177776
 017110 005262 177774
 017114 000000

ROT4: MOV #367,-2(R2) ;MOVE TO MAILBOX # ***** 367 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROL MODE 4 FAILED
 ; OR SEQUENCE ERROR

2869
 2870
 2871
 2872
 2873
 2874
 2875
 2876
 2877
 2878
 2879

 : THIS TEST VERIFIES MODE 5 ROTATE INSTRUCTIONS.
 : THE DATA IS STORED IN A WORK LOCATION (ROTX) AT THE END OF THE
 : TEST CODE. LOC. 0 IS LOADED WITH THE ADDRESS OF THE DATA (ROTX).
 : R0 IS SET TO 2. THE CARRY IS CLEARED AND A MODE 5 ROL
 : IS EXECUTED USING R0 AS AN ADDRESSING REGISTER. THE DATA IS
 : CHECKED, THE C AND V BITS TESTED, AND R0 CHECKED FOR PROPER
 : DECREMENTING.
 :

2880

```
.SBTTL TEST # 166 - TEST MODE 5 W/ ROTATE INSTRUCTIONS
:*****
:TEST 166 - TEST MODE 5 W/ ROTATE INSTRUCTIONS
:*****
```

```
017116 005212
017120 022712 000166
017124 001021
2881 017126 012737 017204 000000
2882 017134 012700 000002
2883 017140 012767 107070 000036
2884 017146 000241
2885 017150 006150
2886 017152 103006
2887 017154 022737 016160 017204
2888 017162 001002
2889 017164 005700
2890 017166 001407
```

```
TST166: INC (R2) ;UPDATE TEST NUMBER
CMP #166,(R2) ;SEQUENCE ERROR?
BNE ROT5 ;BR TO ERROR HALT ON SEQ ERROR
MOV #ROTX,@#0 ;MOVE POINTER TO LOC. 0
MOV #2,R0 ;SET MODE 5 REG. TO LOC. 0
MOV #107070,ROTX ;INITIALIZE DATA
CLC ;CLEAR C-BIT
ROL @-(R0) ;TRY ROL W/ MODE 5
BCC ROT5 ;CHECK C-BIT
CMP #016160,@#ROTX ;CHECK DATA
BNE ROT5 ;BRANCH IF DATA INCORRECT
TST R0 ;CHECK MODE 5 REGISTER
BEQ TST167
```

```
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 756 <====
```

```
017170
017170 012762 000370 177776
017176 005262 177774
017202 000000
```

```
ROT5: MOV #370,-2(R2) ;MOVE TO MAILBOX # ***** 370 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ROL MODE 5 FAILED
; OR SEQUENCE ERROR
```

```
2891 017204 000000
2892
2893
2894
2895
2896
2897
2898
2899
```

```
ROTX: 0
:*****
: THIS TEST VERIFIES MODE 6 ROTATE INSTRUCTIONS.
: IT USES THE SAME PROCEDURE AS THE ABOVE TEST EXCEPT THE
: ROTATE INSTRUCTION USES MODE 6 ADDRESSING WITH REGISTER 7.
: THE DATA IS STILL OPERATED ON IN LOC. ROTX (SEE PREVIOUS TEST).
:
```

2900

```
.SBTTL TEST # 167 - TEST MODE 6 W/ ROTATE INSTRUCTIONS
:*****
:TEST 167 - TEST MODE 6 W/ ROTATE INSTRUCTIONS
:*****
```

```
017206 005212
017210 022712 000167
017214 001015
2901 017216 012737 125252 017204
2902 017224 000261
2903 017226 006167 177752
2904 017232 103004
2905 017234 022737 052525 017204
2906 017242 001406
```

```
TST167: INC (R2) ;UPDATE TEST NUMBER
CMP #167,(R2) ;SEQUENCE ERROR?
BNE TST170-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125252,@#ROTX ;INITIALIZE DATA
SEC ;SET C-BIT
ROL ROTX ;TRY ROL W/ MODE 6
BCC ROT6 ;CHECK C-BIT
CMP #52525,@#ROTX ;CHECK DATA
BEQ TST170
```

```
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 764 <====
```

```
017244
017244 012762 000371 177776
017252 005262 177774
017256 000000
```

```
ROT6: MOV #371,-2(R2) ;MOVE TO MAILBOX # ***** 371 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ROL W/ MODE 6 FAILED
; OR SEQUENCE ERROR
```

2907
2908
2909
2910
2911
2912
2913
2914

```
:*****
:THIS TEST VERIFIES MODE 7 ROTATE INSTRUCTIONS.
:THE DATA IS SET IN LOC. ROTX, (SEE PREVIOUS TEST). THE ROL INSTRUCTION
:ADDRESSES IT INDIRECTLY USING MODE 7 AND INDIRECT ADDRESS LOCATION
: (ROTXAD) FOLLOWING THE TEST CODE.
:
```

2915

.SBTTL TEST # 170 - TEST MODE 7 W/ ROTATE INSTRUCTIONS

:TEST 170 - TEST MODE 7 W/ ROTATE INSTRUCTIONS

017260 005212
017262 022712 000170
017266 001016
2916 017270 012737 052525 017204
2917 017276 012737 017204 017340
2918 017304 000241
2919 017306 006177 000026
2920 017312 103404
2921 017314 023727 017204 125252
2922 017322 001407

TST170: INC (R2) ;UPDATE TEST NUMBER
CMP #170,(R2) ;SEQUENCE ERROR?
BNE ROT7 ;BR TO ERROR HALT ON SEQ ERROR
MOV #52525,@#ROTX ;INITIALIZE DATA
MOV #ROTX,@#ROTXAD ;INITIALIZE ADDRESS POINTER
CLC ;CLEAR C-BIT
ROL @ROTXAD ;TRY ROL W/ MODE 7
BCS ROT7 ;CHECK C-BIT
CMP @#ROTX,#125252 ;CHECK DATA
BEQ TST171

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < ---
: CONDITIONAL BRANCH INST. AND < ---
: REPLACE THE MOVE INSTRUCTION <= ---
: WHICH FOLLOWS W/ 761 < ---

017324
017324 012762 000372 177776
017332 005262 177774
017336 000000

ROT7: MOV #372,-2(R2) ;MOVE TO MAILBOX # ***** 372 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ROL W/ MODE 7 FAILED
; OR SEQUENCE ERROR

2923 017340 000000

ROTXAD: 0

2924
2925
2926
2927
2928
2929
2930
293
2932

: THIS TEST VERIFIES MODE 0 SWAB INSTRUCTION. R0 IS SET TO
: 177400. A SWAB MODE 0 IS EXECUTED AND THE CONDITIONAL BRANCH
: IS USED TO CHECK THE SIGN OF THE RESULT. ALSO, A COMPARISON
: IS MADE TO CHECK THE DATA RESULTS.
:

2933

.SBITL TEST # 171 - TEST MODE 0 W/ SWAB INST.

:TEST 171 - TEST MODE 0 W/ SWAB INST.

017342 005212
017344 022712 000171
017350 001017
2934 017352 012700 177400
2935 017356 000300
2936 017360 100406

TST171: INC (R2) ;UPDATE TEST NUMBER
CMP #171,(R2) ;SEQUENCE ERROR?
BNE TST172-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #177400,R0 ;MOVE TEST PATTERN TO R0
SWAB R0 ;TRY SWAB MODE 0
BMI SBO

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 773 <====

017362 012762 000373 177776
017370 005262 177774
017374 000000
2937 017376 022700 000377
2938 017402 001406

MOV #373,-2(R2) ;MOVE TO MAILBOX # ***** 373 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;SWAB DID NOT SET CC'S CORRECT
SBO: CMP #377,R0 ;CHECK RESULT
BEQ TST172

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 762 <====

017404 012762 000374 177776
017412 005262 177774
017416 000000

MOV #374,-2(R2) ;MOVE TO MAILBOX # ***** 374 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF SWAB MODE 0 FAILED
; OR SEQUENCE ERROR

2939
2940
2941
2942
2943
2944
2945
2946

: THIS TEST VERIFIES MODE 1 SWAB INSTRUCTION. THE TEST
: PATTERN IS MOVED TO LOC 0. R0 IS CLEARED AND USED AS THE ADDRESSING
: REGISTER IN THE MODE 1 SWAB. THE DATA RESULTS ARE CHECKED WITH
: A COMPARE.
:

TEST # 172 - TEST MODE 1 W/ SWAB INST

.SBTTL TEST # 172 - TEST MODE 1 W/ SWAB INST

:*****
:TEST 172 - TEST MODE 1 W/ SWAB INST
:*****

017420 005212
017422 022712 000172
017426 001013
2948 017430 012737 125652 000000
2949 017436 005000
2950 017440 000310
2951 017442 022737 125253 000000
2952 017450 001406

TST172: INC (R2) ;UPDATE TEST NUMBER
CMP #172,(R2) ;SEQUENCE ERROR?
BNE TST173-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125652,@#0 ;MOVE TEST PATTERN TO LOC. 0
CLR R0 ;R0=0
SWAB (R0) ;TRY SWAB MODE 1
CMP #125253,@#0 ;CHECK RESULT
BEQ TST173

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
; CONDITIONAL BRANCH INST. AND
; REPLACE THE MOVE INSTRUCTION
; WHICH FOLLOWS W/ 766
MOV #375,-2(R2) ;MOVE TO MAILBOX # ***** 375 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF SWAB MODE 1 FAILED
; OR SEQUENCE ERROR

2953
2954
2955
2956
2957
2958
2959
2960
2961

:*****
: THIS TEST VERIFIES MODE 2 SWAB INSTRUCTION. THE TEST
: PATTERN IS MOVED TO LOC 0. R0 IS CLEARED AND USED AS THE MODE
: 2 ADDRESSING REGISTER. THE RESULTS ARE CHECKED WITH A COMPARE.
: R0 IS CHECKED FOR PROPER DECREMENTING.
:*****

2962

..SBTTL TEST # 173 - TEST MODE 2 W/ SWAB INST
:.....
:TEST 173 - TEST MODE 2 W/ SWAB INST
:.....

017466 005212
017470 022712 000173
017474 001024
2963 017476 012737 125152 000000
2964 017504 005000
2965 017506 000320
2966 017510 022737 065252 000000
2967 017516 001406

TST173: INC (R2) ;UPDATE TEST NUMBER
CMP #173,(R2) ;SEQUENCE ERROR?
BNE TST174-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125152,@#0 ;MOVE TEST PATTERN TO LOC. 0
CLR R0 ;R0=0
SWAB (R0)+ ;TRY SWAB MODE 2
CMP #65252,@#0 ;CHECK RESULT
BEQ SB2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-----
: CONDITIONAL BRANCH INST. AND <-----
: REPLACE THE MOVE INSTRUCTION <-----
: WHICH FOLLOWS W/ 766 <-----

017520 012762 000376 177776
017526 005262 177774
017532 000000
2968 017534 162700 000002
2969 017540 001406

MOV #376,-2(R2) ;MOVE TO MAILBOX # ***** 376 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF SWAB MODE 0 FAILED
SB2: SUB #2,R0 ;CHECK EFFECT OF REG.
BEQ TST174

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-----
: CONDITIONAL BRANCH INST. AND <-----
: REPLACE THE MOVE INSTRUCTION <-----
: WHICH FOLLOWS W/ 755 <-----

017542 012762 000377 177776
017550 005262 177774
017554 000000

MOV #377,-2(R2) ;MOVE TO MAILBOX # ***** 377 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REGISTER VALUE INCORRECT
; OR SEQUENCE ERROR

2970
2971
2972
2973
2974
2975
2976
2977
2978

.....
: THIS TEST VERIFIES MODE 3 SWAB INSTRUCTION. THE TEST
: PATTERN IS MOVED TO LOC 0. A MODE 3 SWAB INSTRUCTION IS EXECUTED
: USING R7 AS THE ADDRESSING REGISTER. A COMPARE VERIFIES THE
: DATA RESULTS.
:

2979

.SBTTL TEST # 174 - TEST MODE 3 W/SWAB INST.

:TEST 174 - TEST MODE 3 W/SWAB INST.

017556 005212
017560 022712 000174
017564 001013
2980 017566 012737 000377 000000
2981 017574 000337 000000
2982 017600 022737 177400 000000
2983 017606 001406

TST174: INC (R2) ;UPDATE TEST NUMBER
CMP #174,(R2) ;SEQUENCE ERROR?
BNE TST175-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #377,@#0 ;MOVE TEST PATTERN TO LOC. 0
SWAB @#0 ;TRY SWAB W/ MODE 3
CMP #177400,@#0 ;CHECK RESULT
BEQ TST175
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <== -
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 766 <====
MOV #400,-2(R2) ;MOVE TO MAILBOX # ***** 400 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF SWAB INCORRECT
; OR SEQUENCE ERROR

2984
2985
2986
2987
2988
2989
2990
2991
2992

: THIS TEST VERIFIES MODE 4 SWAB INSTRUCTIONS. THE DATA
: IS MOVED TO LOC 0. R0 IS SET TO 2 AND USED AS THE MODE 4 ADDRESSING
: REGISTER. THE DATA IS CHECKED WITH A COMPARE AND R0 IS CHECKED
: FOR PROPER DECREMENTING.
:

2993

.SBTTL TEST # 175 - TEST MODE 4 W/ SWAB INST
:*****
:TEST 175 - TEST MODE 4 W/ SWAB INST
:*****

017624 005212
017626 022712 000175
017632 001024
2994 017634 012737 125652 000000
2995 017642 012700 000002
2996 017646 000340
2997 017650 022737 125253 000000
2998 017656 001406

TST175: INC (R2)
CMP #175,(R2)
BNE TST176-10
MOV #125652,@#0
MOV #2,R0
SWAB -(R0)
CMP #125253,@#0
BEQ SB4

;UPDATE TEST NUMBER
;SEQUENCE ERROR?
;BR TO ERROR HALT ON SEQ ERROR
;MOVE TEST PATTERN TO LOC. 0
;SET UP REGISTER POINTFR
;TRY SWAB MODE 4
;CHECK RESULT

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 765 <====

017660 012762 000401 177776
017666 005262 177774
017672 000000
2999 017674 005700
3000 017676 001406

MOV #401,-2(R2)
INC -4(R2)
HALT
SB4: TST R0
BEQ TST176

;MOVE TO MAILBOX # ***** 401 *****
;SET MSGTYP TO FATAL ERROR
;RESULT OF SWAB INCORRECT
;CHECK EFFECT ON REG.

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 755 <====

017700 012762 000402 177776
017706 005262 177774
017712 000000

MOV #402,-2(R2)
INC -4(R2)
HALT

;MOVE TO MAILBOX # ***** 402 *****
;SET MSGTYP TO FATAL ERROR
;REGISTER VALUE INCORRECT
; OR SEQUENCE ERROR

3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011

:*****
: THIS TEST VERIFIES MODE 5 SWAB INSTRUCTION. THE TEST USES
: TWO LOCATIONS FOLLOWING THE TEST CODE. SB5X HOLDS THE DATA;
: SB5XAD IS A POINTER TO THE DATA LOCATION. THE DATA IS MOVED TO
: SB5X AND R0 IS SET TO TWO PLUS THE ADDRESS OF SB5XAD. FOLLOWING
: THE MODE 5 SWAB SB5X IS CHECKED FOR THE PROPER DATA. R0 IS
: CHECKED TO SEE THAT IT WAS DECREMENTED PROPERLY.
:*****

3012

.SBTTL TEST # 176 - TEST MODE 5 W/ SWAB INST.
 :*****
 :TEST 176 - TEST MODE 5 W/ SWAB INST.
 :*****

017714 005212
 017716 022712 000176
 017722 001023
 3013 017724 012700 020012
 3014 017730 012767 125125 000050
 3015 017736 000350
 3016 017740 022767 052652 000040
 3017 017746 001406

TST176: INC (R2) ;UPDATE TEST NUMBER
 CMP #176,(R2) ;SEQUENCE ERROR?
 BNE SB5 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #SB5XAD+2,R0 ;SET UP POINTER TO WORK LOCATION
 MOV #125125,SB5X ;MOVE PATTERN TO WORK LOCATION
 SWAB @-(R0) ;TRY SWAB MODE 5
 CMP #52652,SB5X ;CHECK RESULT
 BEQ SB5A

017750 012762 000403 177776
 017756 005262 177774
 017762 000000
 3018 017764 020027 020010
 3019 017770 001410

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 765 <====
 MOV #403,-2(R2) ;MOVE TO MAILBOX # ***** 403 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;RESULT OF SWAB INCORRECT
 SB5A: CMP R0,#SB5XAD ;CHECK RESULT OF RFG.
 BEQ TST177

017772
 017772 012762 000404 177776
 020000 005262 177774
 020004 000000

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 754 <====
 SB5: MOV #404,-2(R2) ;MOVE TO MAILBOX # ***** 404 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REGISTER VALUE INCORRECT
 ; OR SEQUENCE ERROR
 ;WORK LOCATION

3020 020006 000000
 3021 020010 020006

SB5X: 0
 SB5XAD: SB5X

3022
 3023
 3024
 3025
 3026
 3027
 3028
 3029
 3030
 3031
 3032

:*****
 : THIS TEST VERIFIES MODE 6 SWAB INSTRUCTION. THIS TEST
 : USES A WORK LOCATION (SB6X) FOLLOWING THE TEST CODE. TEST DATA
 : IS LOADED INTO THE WORK LOCATION. R0, THE ADDRESSING REGISTER
 : IS LOADED WITH 6 LESS THEN THE ADDRESS OF THE WORK LOCATION.
 : THE MODE 6 SWAB IS EXECUTED WITH A +6 OFFSET. THE DATA IS
 : VERIFIED WITH A COMPARE.
 :

3033

.SBTTL TEST # 177 - TEST MODE 6 W/ SWAB INST.
 :*****
 :TEST 177 - TEST MODE 6 W/ SWAB INST.
 :*****

020012 005212
 020014 022712 000177
 020020 001013
 3034 020022 012767 125125 000034
 3035 020030 012700 020056
 3036 020034 000360 000006
 3037 020040 022760 052652 000006
 3038 020046 001407

TST177: INC (R2) ;UPDATE TEST NUMBER
 CMP #177,(R2) ;SEQUENCE ERROR?
 BNE SB6 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125125,SB6X ;MOVE PATTERN TO WORK LOCATION
 MOV #SB6X-6,R0 ;MOVE OFFSET POINTER TO R0
 SWAB 6(R0) ;TRY SWAB W/ MODE 6
 CMP #52652,6(R0) ;CHECK RESULT
 BEQ TST200
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 ; CONDITIONAL BRANCH INST. AND <---
 ; REPLACE THE MOVE INSTRUCTION <---
 ; WHICH FOLLOWS W/ 764 <---

020050
 020050 012762 000405 177776
 020056 005262 177774
 020062 000000

SB6: MOV #405,-2(R2) ;MOVE TO MAILBOX # ***** 405 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;RESULT OF SWAB INCORRECT
 ; OR SEQUENCE ERROR
 SB6X: 0 ;WORK LOCATION

3039 020064 000000
 3040
 3041
 3042
 3043
 3044
 3045
 3046
 3047
 3048
 3049
 3050
 3051

:*****
 : THIS TEST VERIFIES MODE 7 SWAB INSTRUCTION. THIS TEST
 : USES TWO LOCATIONS FOLLOWING THE TEST CODE: A WORK LOCATION
 : (SB7X) AND A POINTER TO THE WORK LOCATION (SB7XAD). DATA IS MOVED
 : TO THE WORK LOCATION. R0 IS LOADED WITH 72 LESS THAN THE ADDRESS
 : OF THE ADDRESS POINTER. THE DATA IS SWAB'ED USING A MODE 7
 : INSTRUCTION WITH AN OFFSET OF +72. THE DATA IS VERIFIED WITH A
 : COMPARE.
 :*****

```

3052          .SBTTL TEST # 200 - TEST MODE 7 W/ SWAB INST.
:*****
:TEST 200 - TEST MODE 7 W/ SWAB INST.
:*****
020066 005212          TST200: INC      (R2)          ;UPDATE TEST NUMBER
020070 022712 000200    CMP      #200,(R2)      ;SEQUENCE ERROR?
020074 001013          BNE      SB7           ;BR TO ERROR HALT ON SEQ ERROR
3053 020076 012767 177400 000034    MOV      #177400,SB7X   ;MOVE PATTERN TO WORK LOCATION
3054 020104 012700 020050    MOV      #SB7XAD-72,R0 ;MOVE OFFSET POINTER TO R0
3055 020110 000370 000072          SWAB     @72(R0)       ;TRY SWAB MODE 7
3056 020114 027027 000072 000377    CMP      @72(R0),#377  ;CHECK RESULTS
3057 020122 001410          BEQ     TST201

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:          CONDITIONAL BRANCH INST. AND <====
:          REPLACE THE MOVE INSTRUCTION <====
:          WHICH FOLLOWS W/ 764 <====

020124          SB7:
020124 012762 000406 177776    MOV      #406,-2(R2)   ;MOVE TO MAILBOX # ***** 406 *****
020132 005262 177774          INC      -4(R2)       ;SET MSGTYP TO FATAL ERROR
020136 000000          HALT

3058 020140 000000          SB7X: 0
3059 020142 020140          SB7XAD: SB7X         ;WORK LOCATION
:          POINTER TO WORK LOCATION

:*****
:
:          THIS TEST VERIFIES ALL LEGAL MODES OF THE JMP INSTRUCTION.
:BECAUSE OF THE NATURE OF THE INSTRUCTION UNDER TEST, THIS TEST
:UTILIZES SEVERAL DIFFERENT TECHNIQUES. THE CODE IS NOT EXECUTED
:IN A LINEAR FASHION. THE DIFFERENT MODES ARE EXECUTED IN ORDER
:FROM 1-7; HOWEVER, THE CODE IS ARRANGED SO THAT CONTROL LEAP
:FROGS THRU THE TEST CODE. THE ORDER OF APPEARANCE OF THE CODE
:IS:
:          JMP MODE 1
:          JMP MODE 3
:          JMP MODE 2
:          JMP MODE 4
:          JMP MODE 6
:          JMP MODE 5
:          JMP MODE 7
:AN INTERNAL SEQUENCE TEST (JMPSEQ) IS USED TO INSURE THAT THE
:JUMPS ARE OCCURRING IN THE PROGRAMMED SEQUENCE.
:          THE TEST IS MADE UP OF SEVERAL BLOCKS OF CODE. EACH CODE
:BLOCK BEGINS WITH A LABEL WHICH INDICATES THE MODE BEING EXECUTED IN
:THAT BLOCK. A SIMPLE PROCEDURE IS FOLLOWED IN EACH BLOCK. FOR
:EXAMPLE THE CODE BEGINNING AT JMP3 WILL FIRST COMPARE THE RESULTS
:OF THE PREVIOUS MODE 2 JUMP. (ANY REGISTER CHANGES ARE VERIFIED
:AND THE SEQUENCE CHECK IS MADE). THEN THE REGISTERS ARE SETUP
:FOR A MODE 3 JUMP TO THE NEXT TEST BLOCK (HERE, JMP4), THE SEQUENCE
:CHECKER IS UPDATED AND THE JUMP IS EXECUTED.
:          IF A FAILURE OCCURS, THE SEQUENCE CHECKER WILL ASSIST IN
:DETERMINING JUST WHICH MODE FAILED. IF THE SEQUENCE IS CORRECT
:THEN THE ERROR DETECTED WAS A MODE FAILURE (E.G. FAILURE OF THE
:REGISTER TO BE INCREMENTED IN MODE 2 JUMP.)
:

```

```

3093          .SBTTL TEST # 201 - TEST THE JMP INSTRUCTION IN ALL MODES
              :*****
              :TEST 201 - TEST THE JMP INSTRUCTION IN ALL MODES
              :*****
020144 005212          TST201: INC      (R2)          ;UPDATE TEST NUMBER
020146 022712 000201      CMP      #201,(R2)        ;SEQUENCE ERROR?
020152 001170          BNE      JMPCK+6        ;BR TO ERROR HALT ON SEQ ERROR
3094 020154 005067 000372      CLR      JMPSEQ          ;ESTABLISH A SEQUENCE CHECKER
3095 020160 012700 020250      MOV      #JMP2,R0        ;SET R0=JUMP TARGET
3096 020164 000110          JMP      (R0)          ;TRY JMP MODE 1
3097 020166 022700 020170      JMP3:  CMP      #.+2,R0        ;CHECK RESULT OF MODE 2 JUMP
3098 020172 001406          BEQ      JMP3A

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 767 <====
020174 012762 000407 177776      MOV      #407,-2(R2)      ;MOVE TO MAILBOX # ***** 407 *****
020202 005262 177774          INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
020206 000000          HALT
3099 020210 026727 000336 000001      JMP3A: CMP      JMPSEQ,#1    ;REGISTER VALUE AFTER JMP MODE 2 INCORRECT
3100 020216 001406          BEQ      JMP3B          ;MAKE SURE JMPS ARE IN SEQUENCE: JMPSEQ=1?

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 755 <====
020220 012762 000410 177776      MOV      #410,-2(R2)      ;MOVE TO MAILBOX # ***** 410 *****
020226 005262 177774          INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
020232 000000          HALT
3101 020234 012700 020246      JMP3B: MOV      #IJMP4,R0    ;SHOULD BE HERE FROM JMP MODE 2 ONLY
3102 020240 005267 000306          INC      JMPSEQ          ;POINT R0 TO INDIRECT JMP ADDR.
3103 020244 000130          JMP      @ (R0)+          ;UPDATE SEQUENCE CHECKER
3104 020246 020304          IJMP4: JMP4          ;TRY JMP MODE 3
3105
3106 020250 005767 000276      JMP2:  TST      JMPSEQ      ;ADDRESS INDIRECT JUMP
3107 020254 001406          BEQ      JMP2A          ;CHECK THAT JMPS ARE IN SEQUENCE: JMPSEQ=0?

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 736 <====
020256 012762 000411 177776      MOV      #411,-2(R2)      ;MOVE TO MAILBOX # ***** 411 *****
020264 005262 177774          INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
020270 000000          HALT
3108 020272 005267 000254      JMP2A: INC      JMPSEQ      ;SHOULD BE HERE FROM JMP MODE 1 ONLY
3109 020276 012700 020166          MOV      #JMP3,R0        ;UPDATE SEQUENCE CHECKER
3110 020302 000120          JMP      (R0)+          ;SET R0=JUMP TARGET
3111 020304 022700 020250      JMP4:  CMP      #IJMP4+2,R0 ;TRY A JUMP MODE 2 TO 'JMP3'
3112 020310 001406          BEQ      JMP4A          ;CHECK RESULT OF REGISTER IN MODE 3 JUMP

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
              ;          CONDITIONAL BRANCH INST. AND <====
              ;          REPLACE THE MOVE INSTRUCTION <====
              ;          WHICH FOLLOWS W/ 720 <====
020312 012762 000412 177776      MOV      #412,-2(R2)      ;MOVE TO MAILBOX # ***** 412 *****
020320 005262 177774          INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
020324 000000          HALT
3113 020326 022767 000002 000216      JMP4A: CMP      #2,JMPSEQ    ;REGISTER VALUE AFTER MODE 3 JUMP INCORRECT
3114 020334 001406          BEQ      JMP4B          ;CHECK JUMP SEQUENCE: JMPSEQ=2?

              ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
    
```



```
020536 012762 000417 177776      MOV    #417,-2(R2)      ;MOVE TO MAILBOX # ***** 417 *****
020544 005262 177774              INC    -4(R2)          ;SET MSGTYP TO FATAL ERROR
020550 000000                    HALT                   ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
; OR SEQUENCE ERROR
```

```
3141 020552 000000
```

```
JMPSEQ: 0
```

3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158

```
:*****  
: THIS TEST VERIFIES ALL LEGAL MODES OF THE JSR INSTRUCTION.  
: THE CONCEPT OF LEAP FROGGING AND SEQUENCE CHECKING (JSRSEQ) IS  
: IDENTICAL TO THAT USED IN JMP TEST (SEE PREVIOUS TEST). EACH  
: BLOCK OF CODE VERIFIES THE PREVIOUS JSR BY CHECKING THE SEQUENCE,  
: CHECKING THAT THE PC WAS SAVED IN THE SPECIFIED REGISTER, CHECKING  
: THAT THE SP WAS DECREMENTED, CHECKING THAT THE REGISTER WAS  
: SAVED ON THE STACK, AND FINALLY CHECKING THAT ANY MODE ADDRESS  
: REGISTER ALTERATIONS (E.G. INCREMENT REGISTER IN MODE 2) WERE  
: SUCCESSFUL. R1 IS USED AS THE REGISTER IN ALL JSR INSTRUCTIONS.  
: IF A FAILURE OCCURS, THE SEQUENCE CHECKER WILL ASSIST IN  
: DETERMINING JUST WHICH MODE FAILED. IF THE SEQUENCE IS CORRECT  
: THEN THE ERROR DETECTED WAS A FUNCTIONAL FAILURE (E.G., INCORRECT  
: REGISTER SAVED).  
:
```

```

3159          .SBTTL TEST # 202 - TEST JSR INSTRUCTION W/ ALL MODES
              :*****
              :TEST 202 - TEST JSR INSTRUCTION W/ ALL MODES
              :*****
              TST202: INC      (R2)          ;UPDATE TEST NUMBER
                  020554 005212          ;
                  020556 022712 000207    ;CMP      * 02,(R2)      ;SEQUENCE ERROR?
                  020562 001001          ;BNE     JSR0          ;BR TO ERROR HALT IN SEQ ERROR
3160 020564 000402          ;BR      JSR1
3161 020566 000137 021256    JSR0:  JMP      @#JSRCK1
3162
3163 020572 012706 001000    JSR1:  MOV      #STBOT,R6      ;SET STACK POINTER
3164 020576 012700 020714    ;MOV     #JSR2,R0          ;SET TARGET ADDRESS
3165 020602 005037 021236    ;CLR     @#JSRSEQ         ;INITIALIZE SEQUENCE CHECKER
3166 020606 005001          ;CLR     R1              ;INITIALIZE R1
3167 020610 005101          ;COM     R1
3168 020612 004110          ;JSR     R1,(R0)         ;TRY JSR MODE 1
3169
3170
3171 020614          JSR1A:
3172 020614 012762 000420 177776    ;MOV     #420,-2(R2)     ;MOVE TO MAILBOX # ***** 420 *****
3173 020622 005262 177774          ;INC     -4(R2)         ;SET MSGTYP TO FATAL ERROR
3174 020626 000000          ;HALT
3175
3176 020630 022737 000001 021236    JSR3:  CMP      #1,@#JSRSEQ    ;CHECK SEQUENCE: JSRSEQ=1?
3177 020636 001014          ;BNE     JSR3A          ;BRANCH IF OUT OF SEQUENCE
3178 020640 020127 021002          ;CMP     R1,#JSR4       ;PROPER PC SAVED?
3179 020644 001011          ;BNE     JSR3A          ;BRANCH IF PC WRONG
3180 020646 022706 000776          ;CMP     #STBOT-2,R6    ;STACK POINTER DECREMENTED?
3181 020652 001006          ;BNE     JSR3A          ;BRANCH IF SP WRONG
3182 020654 022716 125252          ;CMP     #125252,(R6)   ;REG SAVED ON STACK?
3183 020660 001003          ;BNE     JSR3A          ;BRANCH IF REG. NOT SAVED
3184 020662 022700 020632          ;CMP     #JSR3+2,R0     ;MODE 2 INCREMENT CORRECT?
3185 020666 001406          ;BEQ     JSR3B
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
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
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595
4596
4597
4598
4599
4600
4601
4602
4603
4604
4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054
5055
5056
5057
5058
5059
5060
5061
5062
5063
5064
5065
5066
5067
5068
5069
5070
5071
5072
5073
5074
5075
5076
5077
5078
5079
5080
5081
5082
5083
5084
5085
5086
5087
5088
5089
5090
5091
5092
5093
5094
5095
5096
5097
5098
5099
5100
5101
5102
5103
5104
5105
5106
5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397
5398
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408
5409
5410
5411
541
```

```

020744 012762 000422 177776      MOV      #422,-2(R2)      ;MOVE TO MAILBOX # ***** 422 *****
020752 005262 177774      INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
020756 000000      HALT                    ;JSR MODE 1 MALFUNCTIONED
3194 020760 012706 001000      JSR2B:  MOV      #STBOT,R6      ;INITIALIZE R6
3195 020764 012701 125252      MOV      #125252,R1      ;INITIALIZE R1
3196 020770 005237 021236      INC      @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
3197 020774 012700 020630      MOV      #JSR3,R0        ;SET TARGET ADDRESS
3198 021000 004120      JSR      R1,(R0)+        ;TRY JSR MODE 2
3199
3200 021002 022737 000002 021236  JSR4:   CMP      #2,@#JSRSEQ      ;CHECK SEQUENCE: JSRSEQ=2?
3201 021010 001003      BNE      JSR4A           ;BRANCH IF OUT OF SEQUENCE
3202 021012 022701 020714      CMP      #JSR2,R1        ;PROPER PC SAVED?
3203 021016 001406      BEQ      JSR4B           ;
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 661 <
021020
021020 012762 000423 177776      JSR4A:  MOV      #423,-2(R2)      ;MOVE TO MAILBOX # ***** 423 *****
021026 005262 177774      INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
021032 000000      HALT                    ;JSR MODE 3 MALFUNCTIONED
3204 021034 005237 021236      JSR4B:  INC      @#JSRSEQ      ;UPDATE SEQUENCE CHECKER
3205 021040 012700 021120      MOV      #JSR5+2,R0      ;SET TARGET ADDRESS
3206 021044 004140      JSR      R1,-(R0)        ;TRY JSR MODE 4
3207
3208 021046 022767 000004 000162  JSR6:   CMP      #4,JSRSEQ      ;CHECK SEQUENCE: JSRSEQ=4?
3209 021054 001006      BNE      JSR6A           ;BRANCH IF OUT OF SEQUENCE
3210 021056 022701 021170      CMP      #JSR7,R1        ;PROPER PC SAVED?
3211 021062 001003      BNE      JSR6A           ;BRANCH IF PC WRONG
3212 021064 022700 021232      CMP      #JSR6AD,R0      ;MODE 5 REGISTER CORRECT?
3213 021070 001406      BEQ      JSR6B           ;
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 634 <
021072
021072 012762 000424 177776      JSR6A:  MOV      #424,-2(R2)      ;MOVE TO MAILBOX # ***** 424 *****
021100 005262 177774      INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
021104 000000      HALT                    ;JSR MODE 5 FAILED
3214 021106 005237 021236      JSR6B:  INC      @#JSRSEQ      ;UPDATE SEQUENCE CHECKER
3215 021112 004167 000052      JSR      R1,JSR7         ;TRY JSR MODE 6
3216 021116 022767 000003 000112  JSR5:   CMP      #3,JSRSEQ      ;CHECK SEQUENCE: JSRSEQ=3?
3217 021124 001006      BNE      JSR5A           ;BRANCH IF OUT OF SEQUENCE
3218 021126 022701 021046      CMP      #JSR6,R1        ;PROPER PC SAVED?
3219 021132 001003      BNE      JSR5A           ;BRANCH IF PC WRONG
3220 021134 022700 021116      CMP      #JSR5,R0        ;CHECK MODE 4 REGISTER
3221 021140 001406      BEQ      JSR5B           ;
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 610 <
021142
021142 012762 000425 177776      JSR5A:  MOV      #425,-2(R2)      ;MOVE TO MAILBOX # ***** 425 *****
021150 005262 177774      INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
021154 000000      HALT                    ;JSR MODE 4 MALFUNCTIONED
3222 021156 005237 021236      JSR5B:  INC      @#JSRSEQ      ;UPDATE SEQUENCE CHECKER
3223 021162 012700 021234      MOV      #JSR6AD+2,R0    ;POINT R0 TO TARGET ADDRESS

```

```

3224 021166 004150      JSR      R1,@-(R0)      ;TRY JSR MODE 5
3225
3226 021170 022737 000005 021236 JSR7:  CMP      #5,@#JSRSEQ      ;CHECK SEQUENCE: JSRSEQ=5?
3227 021176 001003      BNE     JSR7A          ;BRANCH IF OUT OF SEQUENCE
3228 021200 022701 021116      CMP     #JSR5,R1      ;PROPER PC SAVED?
3229 021204 001406      BEQ     JSR7B          ;
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS      <====
;          CONDITIONAL BRANCH INST. AND      <====
;          REPLACE THE MOVE INSTRUCTION      <====
;          WHICH FOLLOWS W/ 566              <====
021206 JSR7A:
021206 012762 000426 177776      MOV     #426,-2(R2)    ;MOVE TO MAILBOX # ***** 426 *****
021214 005262 177774      INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
021220 000000      HALT                    ;JSR MODE 6 FAILED
3230 021222 005237 021236 JSR7B:  INC     @#JSRSEQ    ;UPDATE SEQUENCE CHECKER
3231 021226 004177 000002      JSR     R1,@JSRCKAD   ;TRY JSR MODE 7
3232
3233 021232 021046      JSR6AD: JSR6           ;MODE 5 TARGET ADDRESS
3234 021234 021240      JSRCKAD:JSRCK         ;MODE 7 TARGET ADDRESS
3235 021236 000000      JSRSEQ: 0             ;SEQUENCE CHECKER
3236
3237 021240 022767 000006 177770 JSRCK:  CMP     #6,JSRSEQ  ;CHECK SEQUENCE: JSRSEQ=6?
3238 021246 001003      BNE     JSRCK1        ;BRANCH IF OUT OF SEQUENCE
3239 021250 022701 021232      CMP     #JSR6AD,R1    ;PROPER PC SAVED?
3240 021254 001406      BEQ     TST203        ;
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS      <====
;          CONDITIONAL BRANCH INST. AND      <====
;          REPLACE THE MOVE INSTRUCTION      <====
;          WHICH FOLLOWS W/ 542              <====
021256 JSRCK1:
021256 012762 000427 177776      MOV     #427,-2(R2)    ;MOVE TO MAILBOX # ***** 427 *****
021264 005262 177774      INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
021270 000000      HALT                    ;JSR MODE 7 MALFUNCTIONED
; OR SEQUENCE ERROR

```

3241
 3242
 3243
 3244
 3245
 3246
 3247
 3248
 3249

```

:*****
:
: THIS TEST VERIFIES THE RTS INSTRUCTION. THE STACK POINTER
: IS INITIALIZED AND A TEST PATTERN STORED ON STACK. R0 IS LOADED
: WITH RETURN ADDRESS. AN RTS IS EXECUTED, AND, AT THE TARGET
: ADDRESS, A CHECK IS MADE THAT R0 WAS PROPERLY RESTORED FROM THE
: STACK.

```

3250

.SBTTL TEST # 203 - TEST RTS INSTRUCTION

:TEST 203 - TEST RTS INSTRUCTION

021272 005212
021274 022712 000203
021300 001022
3251 021302 012706 001000
3252 021306 012746 052525
3253 021312 012700 021334
3254 021316 000200
3255
3256
3257 021320 012762 000430 177776
021326 005262 177774
021332 000000
3258 021334 022700 052525
3259 021340 001406

TST203: INC (R2) ;UPDATE TEST NUMBER
CMP #203,(R2) ;SEQUENCE ERROR?
BNE TST204-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #STBOT,R6 ;INITIALIZE STACK POINTER
MOV #52525,-(R6) ;INITIALIZE TOP OF STACK
MOV #R^S1,R0 ;INITIALIZE RETURN REGISTER
RTS R0 ;TRY RTS THROUGH R0
; TO SCOPE: REPLACE THE MOVE INSTRUCTION <====
; FOLLOWING W/ 770 <====
MOV #430,-2(R2) ;MOVE TO MAILBOX # ***** 430 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RTS FAILED
RTS1: CMP #52525,R0 ;CHECK THAT R0 RESTORED FROM STACK
BEQ TST204
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 757 <====
MOV #431,-2(R2) ;MOVE TO MAILBOX # ***** 431 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RTS MALFUNCTIONED
; OR SEQUENCE ERROR

3260

3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274

:
: THESE NEXT FOUR TESTS VERIFY THE FUNCTIONING OF A GROUP
: OF FOUR INSTRUCTIONS. THE GROUP CONSISTS OF THE INSTRUCTIONS:
: MOV, BIC, BIT, AND BIS. THESE INSTRUCTIONS ARE SIMILAR IN THE
: WAY THEY EFFECT THE C AND V BITS. THEY ALL LEAVE THE V-BIT
: CLEAR AND THE C-BIT UNAFFECTED.
: THE TEST PROCEDURE IS AS FOLLOWS: THE N, Z, AND V BITS
: ARE LOADED WITH THE COMPLEMENT OF THE EXPECTED RESULTS, THE C-BIT
: IS LOADED WITH THE DESIRED RESULT. THE INSTRUCTION IS EXECUTED
: WITH DIFFERENT DATA PATTERNS AND THE RESULTS ARE VERIFIED WITH
: A SERIES OF CONDITIONAL BRANCH INSTRUCTIONS. THE DATA IS CHOSEN
: TO PRODUCT ALL POSSIBLE COMBINATIONS OF THE C AND V BITS.
:

3275

.SBTTL TEST # 204 - TEST MOV INSTRUCTION
 :*****
 :TEST 204 - TEST MOV INSTRUCTION
 :*****

021356 005212
 021360 022712 000204
 021364 001026
 3276 021366 000277
 3277 021370 000251
 3278 021372 012700 100000
 3279 021376 101402
 3280 021400 102401
 3281 021402 100406

TST204: INC (R2) ;UPDATE TEST NUMBER
 CMP #204,(R2) ;SEQUENCE ERROR?
 BNE TST205-10 ;BR TO ERROR HALT ON SEQ ERROR
 SCC ;CC=0110
 +CLN.CLC
 MOV #100000,R0 ;CC=1000
 BLOS MOV1
 BVS MOV1
 BMI MOV2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 770 <====

021404
 021404 012762 000432 177776
 021412 005262 177774
 021416 000000
 3282
 3283 021420 000277
 3284 021422 000244
 3285 021424 012700 000000
 3286 021430 101002
 3287 021432 102401
 3288 021434 100006

MOV1: MOV #432,-2(R2) ;MOVE TO MAILBOX # ***** 432 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;MOV DID NOT SET CC'S CORRECTLY
 MOV2: SCC ;CC=1011
 CLZ
 MOV #0,R0 ;CC=0101
 BHI MOV3 ;C OR Z = 0?
 BVS MOV3 ;V=1?
 BPL TST205

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 753 <

021436
 021436 012762 000433 177776
 021444 005262 177774
 021450 000000

MOV3: MOV #433,-2(R2) ;MOVE TO MAILBOX # ***** 433 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;MOV DID NOT SET CC'S CORRECTLY
 ; OR SEQUENCE ERROR

3289

.SBTTL TEST # 205 - TEST BIT INSTRUCTION

 :TEST 205 - TEST BIT INSTRUCTION

021452 005212
 021454 022712 000205
 021460 001030
 3290 021462 012700 100001
 3291 021466 000277
 3292 021470 000251
 3293 021472 032700 100000
 3294 021476 101402
 3295 021500 102401
 3296 021502 100406

TST205: INC (R2) ;UPDATE TEST NUMBER
 CMP #205,(R2) ;SEQUENCE ERROR?
 BNE TST206-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #100001,R0
 SCC ;CC=0110
 +CLN.CLC
 BIT #100000,R0 ;CC=1000
 BLOS XBIT1
 BVS XBIT1
 BMI XBIT2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 766

021504
 021504 012762 000434 177776
 021512 005262 177774
 021516 000000
 3297
 3298 021520 000277
 3299 021522 000244
 3300 021524 032700 077776
 3301 021530 101002
 3302 021532 102401
 3303 021534 100006

XBIT1: MOV #434,-2(R2) ;MOVE TO MAILBOX # ***** 434 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BIT DID NOT SET CC'S CORRECTLY
 XBIT2: SCC ;CC=1011
 CLZ
 BIT #77776,R0 ;CC=0101
 BHI XBIT3
 BVS XBIT3
 BPL TST206

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 751

021536
 021536 012762 000435 177776
 021544 005262 177774
 021550 000000

XBIT3: MOV #435,-2(R2) ;MOVE TO MAILBOX # ***** 435 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BIT DID NOT SET CC'S CORRECTLY
 ; OR SEQUENCE ERROR

3304

.SBTTL TEST # 206 - TEST BIC INSTRUCTION

:TEST 206 - TEST BIC INSTRUCTION

021552 005212
021554 022712 000206
021560 001030
3305 021562 012700 177777
3306 021566 000277
3307 021570 000251
3308 021572 042700 077777
3309 021576 101402
3310 021600 102401
3311 021602 100406

TST206: INC (R2) ;UPDATE TEST NUMBER
CMP #206,(R2) ;SEQUENCE ERROR?
BNE TST207-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #177777,R0
SCC ;CC=0110
+CLN!CLC
BIC #77777,R0 ;CC=1000
BLOS BIC1
BVS BIC1
BMI BIC2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 766 <

021604
021604 012762 000436 177776
021612 005262 177774
021616 000000
3312 021620 000277
3313 021622 000244
3314 021624 042700 100000
3315 021630 101002
3316 021632 102401
3317 021634 100006

BIC1: MOV #436,-2(R2) ;MOVE TO MAILBOX # ***** 436 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;BIC DID NOT SET CC'S CORRECTLY
BIC2: SCC ;CC=1011
CLZ
BIC #100000,R0 ;CC=0101
BHI BIC3
BVS BIC3
BPL TST207

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <-
: WHICH FOLLOWS W/ 751 <

021636
021636 012762 000437 177776
021644 005262 177774
021650 000000

BIC3: MOV #437,-2(R2) ;MOVE TO MAILBOX # ***** 437 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;BIC DID NOT SET CC'S CORRECTLY
; OR SEQUENCE ERROR

3318

.SBTTL TEST # 207 - TEST BIS INSTRUCTION
 :*****
 :TEST 207 - TEST BIS INSTRUCTION
 :*****

021652 005212
 021654 022712 000207
 021660 001031
 3319 021662 005000
 3320 021664 000277
 3321 021666 000251
 3322 021670 052700 000000
 3323 021674 103403
 3324 021676 102402
 3325 021700 100401
 3326 021702 001406

TST207: INC (R2) ;UPDATE TEST NUMBER
 CMP #207,(R2) ;SEQUENCE ERROR?
 BNE TST210-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0 ;R0=0
 SCC ;CC=1010
 +CLN!CLC
 BIS #0,R0 ;CC=0100 R0=0
 BCS BIS1
 BVS BIS1
 BMI BIS1
 BEQ BIS2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 ; CONDITIONAL BRANCH INST. AND <--
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 766 <

021704
 021704 012762 000440 177776
 021712 005262 177774
 021716 000000
 3327 021720 000277
 3328 021722 000250
 3329 021724 052700 177777
 3330 021730 103003
 3331 021732 102402
 3332 021734 001401
 3333 021736 100406

BIS1: MOV #440,-2(R2) ;MOVE TO MAILBOX # ***** 440 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BIS DID NOT SET CC'S CORRECTLY
 BIS2: SCC ;CC=0111
 CLN
 BIS #177777,R0 ;CC=1001
 BCC BIS3
 BVS BIS3
 BEQ BIS3
 BMI TST210

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 750 <

021740
 021740 012762 000441 177776
 021746 005262 177774
 021752 000000

BIS3: MOV #441,-2(R2) ;MOVE TO MAILBOX # ***** 441 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BIS DID NOT SET CC'S CORRECTLY
 ; OR SEQUENCE ERROR

3334
 3335
 3336
 3337
 3338
 3339
 3340
 3341
 3342
 3343
 3344
 3345

:*****
 : THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE INC AND
 : DEC INSTRUCTIONS. THESE INSTRUCTIONS BOTH EFFECT THE C AND V
 : BITS THE SAME; THE C-BIT IS LEFT UNCHANGED AND THE V-BIT IS DEPENDENT
 : UPON THE DATA RESULTS. THE SAME PROCEDURE IS USED. THE CONDITION
 : CODE BITS ARE INITIALIZED, THE INSTRUCTION IS EXECUTED AND THE
 : RESULTS ARE VERIFIED WITH A SERIES OF CONDITIONAL BRANCH INSTRUCTIONS.
 : THIS PROCEDURE IS REPEATED WITH SEVERAL DATA PATTERNS TO PRODUCE
 : DIFFERENT COMBINATIONS OF THE C AND V BITS.
 :

3346

```
.SBTTL TEST # 210 - TEST INC INSTRUCTION
:*****
:TEST 210 - TEST INC INSTRUCTION
:*****
```

```
021754 005212
021756 022712 000210
021762 001045
3347 021764 012700 077777
3348 021770 000257
3349 021772 000264
3350 021774 005200
3351 021776 101402
3352 022000 100001
3353 022002 102406

TST210: INC (R2) ;UPDATE TEST NUMBER
        CMP #210,(R2) ;SEQUENCE ERROR?
        BNE TST211-10 ;BR TO ERROR HALT ON SEQ ERROR
        MOV #077777,R0 ;R0=077777
        CCC ;CC=0100
        SEZ ;CC=1010 R0=10000
        INC R0
        BLOS INC1
        BPL INC1
        BVS INC2
```

```
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:          CONDITIONAL BRANCH INST. AND <====
:          REPLACE THE MOVE INSTRUCTION <====
:          WHICH FOLLOWS W/ 767 <====
```

```
022004
022004 012762 000442 177776 INC1: MOV #442,-2(R2) ;MOVE TO MAILBOX # ***** 442 *****
022012 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
022016 000000 HALT ;INC DID NOT SET CC'S CORRECTLY
3354 022020 052700 077777 INC2: BIS #77777,R0 ;R0=177777
3355 022024 000261 SEC ;CC=1011
3356 022026 000244 CLZ
3357 022030 005200 INC R0 ;CC=0101 R0=0
3358 022032 100403 BMI INC3
3359 022034 102402 BVS INC3
3360 022036 103001 BCC INC3
3361 022040 001406 BEQ INC4
```

```
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
:          CONDITIONAL BRANCH INST. AND <
:          REPLACE THE MOVE INSTRUCTION <
:          WHICH FOLLOWS W/ 750 <
```

```
022042
022042 012762 000443 177776 INC3: MOV #443,-2(R2) ;MOVE TO MAILBOX # ***** 443 *****
022050 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
022054 000000 HALT ;INC DID NOT SET CC'S CORRECTLY
3362
3363 022056 000277 INC4: SCC ;CC=1110
3364 022060 000241 CLC
3365 022062 005200 INC R0 ;CC=0000 R0=1
3366 022064 101402 BLOS INC5
3367 022066 100401 BMI INC5
3368 022070 100006 BPL TST211
```

```
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
:          CONDITIONAL BRANCH INST. AND <
:          REPLACE THE MOVE INSTRUCTION <
:          WHICH FOLLOWS W/ 734 <
```

```
022072
022072 012762 000444 177776 INC5: MOV #444,-2(R2) ;MOVE TO MAILBOX # ***** 444 *****
022100 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
022104 000000 HALT ;INC DID NOT SET CC'S CORRECTLY
```

```
: OR SEQUENCE ERROR
```

3369

3370

.SBTTL TEST # 211 - TEST DEC INSTRUCTION

 ;TEST 211 - TEST DEC INSTRUCTION

022106 005212
 022110 022712 000211
 022114 001061
 3371 022116 012700 000002
 3372 022122 000277
 3373 022124 005300
 3374 022126 100403
 3375 022130 001402
 3376 022132 102401
 3377 022134 103406

TST211: INC (R2) ;UPDATE TEST NUMBER
 CMP #211,(R2) ;SEQUENCE ERROR?
 BNE TST212-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #2,R0 ;R0=2
 SCC ;CC=1111
 DEC R0 ;CC=0001 R0=1
 BMI DEC1
 BEQ DEC1
 BVS DEC1
 BCS DEC2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
 ; CONDITIONAL BRANCH INST. AND <---=
 ; REPLACE THE MOVE INSTRUCTION <---=
 ; WHICH FOLLOWS W/ 767 <---=

022136
 022136 012762 000446 177776
 022144 005262 177774
 022150 000000
 3378 022152 000261
 3379 022154 000244
 3380 022156 005300
 3381 022160 101002
 3382 022162 100401
 3383 022164 102006

DEC1: MOV #445,-2(R2) ;MOVE TO MAILBOX # ***** 445 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DEC DID NOT SET CC'S CORRECTLY
 DEC2: SEC ;CC=1011
 CLZ
 DEC R0 ;CC=0101 R0=0
 BHI DEC3
 BMI DEC3
 BVC DEC4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
 ; CONDITIONAL BRANCH INST. AND <---=
 ; REPLACE THE MOVE INSTRUCTION <---=
 ; WHICH FOLLOWS W/ 753 <---

022166
 022166 012762 000446 177776
 022174 005262 177774
 022200 000000
 3384 022202 000277
 3385 022204 000251
 3386 022206 005300
 3387 022210 101402
 3388 022212 102401
 3389 022214 100406

DEC3: MOV #446,-2(R2) ;MOVE TO MAILBOX # ***** 446 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DEC DID NOT SET CC'S CORRECTLY
 DEC4: SCC ;CC=0110
 +CLN!CLC
 DEC R0 ;CC=1000 R0=177777
 BLOS DEC5
 BVS DEC5
 BMI DEC6

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---=
 ; CONDITIONAL BRANCH INST. AND <---=
 ; REPLACE THE MOVE INSTRUCTION <---=
 ; WHICH FOLLOWS W/ 737 <---=

022216
 022216 012762 000447 177776
 022224 005262 177774
 022230 000000
 3390 022232 042700 077777
 3391 022236 000277
 3392 022240 000252
 3393 022242 005300
 3394 022244 100403
 3395 022246 001402
 3396 022250 102001

DEC5: MOV #447,-2(R2) ;MOVE TO MAILBOX # ***** 447 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DEC DID NOT SET CC'S CORRECTLY
 DEC6: BIC #77777,R0 ;R0=100000
 SCC ;CC=0101
 +CLN.CLV
 DEC R0 ;CC=1011 R0=77777
 BMI DEC7
 BEQ DEC7
 BVC DEC7

;CC=1011 R0=77777
 ;CC 0011

3397 022252 103406

BCS TST212

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <- - -
: CONDITIONAL BRANCH INST. AND <--- -
: REPLACE THE MOVE INSTRUCTION <--- -
: WHICH FOLLOWS W/ 720 <- -

022254
022254 012762 000450 177776
022262 005262 177774
022266 000000

DEC7:

MOV #450,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 450 *****
:SET MSGTYP TO FATAL ERROR
:DEC DID NOT SET CC'S CORRECTLY
: OR SEQUENCE ERROR

3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408

.....
: THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE CLR,
: TST, AND SWAB INSTRUCTIONS. THESE THREE INSTRUCTIONS ALL LEAVE
: THE C AND V BITS CLEARED. AGAIN, THE CONDITION CODES ARE PRESET,
: THE INSTRUCTION EXECUTED AND THE RESULTS CHECKED WITH CONDITIONAL
: BRANCH INSTRUCTIONS. THE PROCEDURE IS REPEATED TO PRODUCE OTHER
: COMBINATIONS OF CONDITION CODES.
:

3409

.SBTTL TEST # 212 - TEST CLR INSTRUCTION

:TEST 212 - TEST CLR INSTRUCTION

	022270	005212	
	022272	022712	000212
	022276	001011	
3410	022300	000277	
3411	022302	000244	
3412	022304	005000	
3413	022306	100403	
3414	022310	102402	
3415	022312	103401	
3416	022314	001406	

TST212:	INC	(R2)		;UPDATE TEST NUMBER
	CMP	#212,(R2)		;SEQUENCE ERROR?
	BNE	TST213-10		;BR TO ERROR HALT ON SEQ ERROR
	SCC			;CC=1011
	CLZ			
	CLR	R0		;CC=0100 R0=0
	BMI	CLR1		
	BVS	CLR1		
	BCS	CLR1		
	BEQ	TST213		

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
;          CONDITIONAL BRANCH INST. AND <====
;          REPLACE THE MOVE INSTRUCTION <====
;          WHICH FOLLOWS W/ 770 <====

```

	022316			
	022316	012762	000451	177776
	022324	005262	177774	
	022330	000000		

CLR1:	MOV	#451,-2(R2)		;MOVE TO MAILBOX # ***** 451 *****
	INC	-4(R2)		;SET MSGTYP TO FATAL ERROR
	HALT			;CLR DID NOT SET CC'S CORRECTLY
				; OR SEQUENCE ERROR

3417

3418

.SBTTL TEST # 213 - TEST TST INSTRUCTION

:TEST 213 - TEST TST INSTRUCTION

022332 005212
022334 022712 000213
022340 001026
3419 022342 000277
3420 022344 000244
3421 022346 005700
3422 022350 100403
3423 022352 102402
3424 022354 103401
3425 022356 001406

TST213: INC (R2) ;UPDATE TEST NUMBER
CMP #213,(R2) ;SEQUENCE ERROR?
BNE TST214-10 ;BR TO ERROR HALT ON SEQ ERROR
SCC ;CC=1011
CLZ
TST R0 ;CC=0100
BMI TEST1
BVS TEST1
BCS TEST1
BEQ TEST2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <---
: WHICH FOLLOWS W/ 770 <---

022360
022360 012762 000452 177776
022366 005262 177774
022372 000000
3426 022374 005300
3427 022376 000277
3428 022400 000250
3429 022402 005700
3430 022404 101402
3431 022406 102401
3432 022410 100406

TEST1: MOV #452,-2(R2) ;MOVE TO MAILBOX # ***** 452 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TEST DID NOT SET CC'S CORRECTLY
TEST2: DEC R0 ;MAKE R0 NEGATIVE
SCC ;CC=0111
CLN
TST R0 ;CC=1000
BLOS TEST3
BVS TEST3
BMI TST214

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <---
: WHICH FOLLOWS W/ 753 <---

022412
022412 012762 000453 177776
022420 005262 177774
022424 000000

TEST3: MOV #453,-2(R2) ;MOVE TO MAILBOX # ***** 453 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;TEST DID NOT SET CC'S CORRECTLY
: OR SEQUENCE ERROR

3433

```
.SBTTL TEST # 214 - TEST SWAB INSTRUCTION
*****
:TEST 214 - TEST SWAB INSTRUCTION
*****
```

```
022426 005212
022430 022712 000214
022434 001027
3434 022436 012700 170000
3435 022442 000277
3436 022444 000250
3437 022446 000300
3438 022450 101402
3439 022452 102401
3440 022454 100406
```

```
TST214: INC (R2) ;UPDATE TEST NUMBER
CMP #214,(R2) ;SEQUENCE ERROR?
BNE TST215-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #170000,R0 ;R0=170000
SCC ;CC=0111
CLN
SWAB R0 ;CC=1000 R0-360
BLOS SWB1
BVS SWB1
BMI SWB2
```

```
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 767 <====
```

```
022456
022456 012762 000454 177776
022464 005262 177774
022470 000000
3441 022472 000277
3442 022474 000244
3443 022476 000300
3444 022500 102403
3445 022502 103402
3446 022504 100401
3447 022506 001406
```

```
SWB1: MOV #454,-2(R2) ;MOVE TO MAILBOX # ***** 454 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;SWAB DID NOT SET CC'S CORRECTLY
SWB2: SCC ;CC=1011
CLZ
SWAB R0 ;CC=0100 R0-170000
BVS SWB3
BCS SWB3
BMI SWB3
BEQ TST215
```

```
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION < -
: WHICH FOLLOWS W/ 752 <
```

```
022510
022510 012762 000455 177776
022516 005262 177774
022522 000000
```

```
SWB3: MOV #455,-2(R2) ;MOVE TO MAILBOX # ***** 455 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT
```

3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458

```
*****
:
: THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE ADD AND
: ADC INSTRUCTIONS. BOTH OF THESE INSTRUCTIONS HANDLE THE C AND
: V BITS IDENTICALLY. THE PROCEDURE IS TO PRESET THE CONDITION
: CODES, EXECUTE THE INSTRUCTION WITH A PARTICULAR SET OF DATA, AND
: THEN CHECK THE RESULTS BY EXECUTING A SERIES OF CONDITIONAL
: BRANCHES. THIS PROCEDURE IS REPEATED SEVERAL TIMES WITH DIFFERENT
: DATA TO PRODUCE EVERY COMBINATION OF C AND V BITS.
:
```


3459

.SBTTL TEST # 215 - TEST ADD INSTRUCTION

:TEST 215 - TEST ADD INSTRUCTION

	022524	005212			TST215: INC	(R2)	:UPDATE TEST NUMBER	
	022526	022712	000215		CMP	#215,(R2)	:SEQUENCE ERROR?	
	022532	001074			BNE	TST216-10	:BR TO ERROR HALT ON SEQ ERROR	
3460	022534	012700	040000		MOV	#40000,R0	:R0=40000	
3461	022540	000277			SCC		:CC=1111	
3462	022542	062700	030000		ADD	#30000,R0	:CC=0000 R0=70000	
3463	022546	101402			BLOS	ADD1		
3464	022550	102401			BVS	ADD1		
3465	022552	100006			BPL	ADD2		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	< ---
							: CONDITIONAL BRANCH INST. AND	< ---
							: REPLACE THE MOVE INSTRUCTION	< ---
							: WHICH FOLLOWS W/ 767	<====
	022554			ADD1:				
	022554	012762	000456	177776	MOV	#456,-2(R2)	:MOVE TO MAILBOX # ***** 456 *****	
	022562	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
	022566	000000			HALT		:ADD DID NOT SET CC'S CORRECTLY	
3466	022570	000264		ADD2:	SEZ		:CC=0100	
3467								
3468	022572	062700	010000		ADD	#10000,R0	:CC-1010 40-100000	
3469	022576	101402			BLOS	ADD3		
3470	022600	102001			BVC	ADD3		
3471	022602	100406			BMI	ADD4		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<====
							: CONDITIONAL BRANCH INST. AND	<====
							: REPLACE THE MOVE INSTRUCTION	<====
							: WHICH FOLLOWS W/ 753	<====
	022604			ADD3:				
	022604	012762	000457	177776	MOV	#457,-2(R2)	:MOVE TO MAILBOX # ***** 457 *****	
	022612	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
	022616	000000			HALT		:ADD DID NOT SET CC'S CORRECTLY	
3472	022620	000257		ADD4:	CCC		:CC=1000	
3473	022622	000270			SEN			
3474	022624	062700	100000		ADD	#100000,R0	:CC=0111 R0=0	
3475	022630	101002			BHI	ADD5		
3476	022632	102001			BVC	ADD5		
3477	022634	100006			BPL	ADD6		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<
							: CONDITIONAL BRANCH INST. AND	<
							: REPLACE THE MOVE INSTRUCTION	<
							: WHICH FOLLOWS W/ 736	<
	022636			ADD5:				
	022636	012762	000460	177776	MOV	#460,-2(R2)	:MOVE TO MAILBOX # ***** 460 *****	
	022644	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR	
	022650	000000			HALT		:ADD DID NOT SET CC'S CORRECTLY	
3478	022652	062700	177777	ADD6:	ADD	#177777,R0	:CC=1000 R0=177777	
3479	022656	101402			BLUS	ADD7		
3480	022660	102401			BVS	ADD7		
3481	022662	100406			BMI	ADD8		
							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<====
							: CONDITIONAL BRANCH INST. AND	<====
							: REPLACE THE MOVE INSTRUCTION	<====
							: WHICH FOLLOWS W/ 723	<====

	022664			ADD7:			
	022664	012762	C00461		MOV	#461,-2(R2)	;MOVE TO MAILBOX # ***** 461 *****
	022672	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
	022676	000000			HALT		;ADD DID NOT SET CC'S CORRECTLY
3482	022700	000277		ADD8:	SCC		;CC=1010
3483	022702	000245			+CLC.CLZ		
3484	022704	062700	000001		ADD	#1,R0	;CC=0101 R=0
3485	022710	102403			BVS	ADD9	
3486	022712	103002			BCC	ADD9	
3487	022714	100401			BMI	ADD9	
3488	022716	001406			BEQ	TST216	

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <----=
; CONDITIONAL BRANCH INST. AND <----=
; REPLACE THE MOVE INSTRUCTION <----=
; WHICH FOLLOWS W/ 705 <----=

```

	022720			ADD9:			
	022720	012762	000462		MOV	#462,-2(R2)	;MOVE TO MAILBOX # ***** 462 *****
	022726	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
	022732	000000			HALT		;ADD DID NOT SET CC'S CORRECTLY

3489

; OR SEQUENCE ERROR

3490

.SBTTL TEST # 216 - TEST ADC INSTRUCTION

 :TEST 216 - TEST ADC INSTRUCTION

022734 005212
 022736 022712 000216
 022742 001045
 3491 022744 012700 077777
 3492 022750 000277
 3493 022752 000252
 3494 022754 005500
 3495 022756 101402
 3496 022760 102001
 3497 022762 100406

TST216: INC (R2) ;UPDATE TEST NUMBER
 CMP #216,(R2) ;SEQUENCE ERROR?
 BNE TST217-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #077777,R0
 SCC ;CC=0101
 +CLN!CLV
 ADC R0 ;CC=1010
 BLOS ADC1
 BVC ADC1
 BMI ADC2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 767 <====

022764
 022764 012762 0C0463 177776
 022772 005262 177774
 022776 000000
 3498 023000 052700 077777
 3499 023004 000277
 3500 023006 000244
 3501 023010 005500
 3502 023012 101002
 3503 023014 102401
 3504 023016 100006

ADC1: MOV #463,-2(R2) ;MOVE TO MAILBOX # ***** 463 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ADC DID NOT SET CC'S CORRECTLY
 ADC2: BIS #77777,R0
 SCC ;CC=1011
 CLZ
 ADC R0 ;CC=0101 R0=0
 BHI ADC3
 BVS ADC3
 BPL ADC4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 751 <====

023020
 023020 012762 000464 177776
 023026 005262 177774
 023032 000000
 3505 023034 000277
 3506 023036 000245
 3507 023040 005500
 3508 023042 102403
 3509 023044 103402
 3510 023046 100401
 3511 023050 001406

ADC3: MOV #464,-2(R2) ;MOVE TO MAILBOX # ***** 464 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ADC DID NOT SET CC'S CORRECTLY
 ADC4: SCC ;CC=1010
 +CLZ!CLC
 ADC R0 ;CC=0100
 BVS ADC5
 BCS ADC5
 BMI ADC5
 BEQ TST217

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 734 <====

023052
 023052 012762 000465 177776
 023060 005262 177774
 023064 000000

ADC5: MOV #465,-2(R2) ;MOVE TO MAILBOX # ***** 465 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ADC DID NOT SET CC'S CORRECTLY
 ; OR SEQUENCE ERROR

3512
 3513
 3514
 3515

 : THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE NEG.

TEST # 216 - TEST ADC INSTRUCTION

3516
3517
3518
3519
3520
3521
3522

:CMP, AND COM INSTRUCTIONS. EACH OF THESE INSTRUCTIONS GENERATE
:THE C AND V BITS IDENTICALLY. THE CONDITION CODES ARE PRESET,
:THE INSTRUCTIONS EXECUTED, AND THE RESULTS CHECKED WITH A SERIES
:OF CONDITIONAL BRANCH INSTRUCTIONS. THIS PROCEDURE IS REPEATED
:SEVERAL TIMES WITH DIFFERENT DATA IN ORDER TO GENERATE DIFFERENT
:COMBINATIONS OF THE C AND V BITS.
:

3523

.SBTTL TEST # 217 - TEST NEG INSTRUCTION
:*****
:TEST 217 - TEST NEG INSTRUCTION
:*****

023066 005212
023070 022712 000217
023074 001050
3524 023076 012700 000001
3525 023102 000277
3526 023104 000251
3527 023106 005400
3528 023110 103003
3529 023112 102402
3530 023114 001401
3531 023116 100406

TST217: INC (R2) ;UPDATE TEST NUMBER
CMP #217,(R2) ;SEQUENCE ERROR?
BNE TST220-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #1,R0
SCC ;CC=0110
+CLN,CLC
NEG R0 ;CC=1001 R0=177777
BCC NEG1
BVS NEG1
BEQ NEG1
BMI NEG2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
; CONDITIONAL BRANCH INST. AND <===
; REPLACE THE MOVE INSTRUCTION <===
; WHICH FOLLOWS W/ 766 <===

023120
023120 012762 000466 177776
023126 005262 177774
023132 000000
3532 023134 042700 077777
3533 023140 000257
3534 023142 000264
3535 023144 005400
3536 023146 102003
3537 023150 103002
3538 023152 001401
3539 023154 100406

NEG1: MOV #466,-2(R2) ;MOVE TO MAILBOX # ***** 466 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;NEG DID NOT SET CC'S CORRECTLY
NEG2: BIC #77777,R0
CCC ;CC=0100
SEZ
NEG R0 ;CC=1011 R0=100000
BVC NFG3
BCC NEG3
BEQ NEG3
BMI NEG4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < - -
; CONDITIONAL BRANCH INST. AND < - -
; REPLACE THE MOVE INSTRUCTION < - -
; WHICH FOLLOWS W/ 747 < - -

023156
023156 012762 000467 177776
023164 005262 177774
023170 000000
3540 023172 005000
3541 023174 000277
3542 023176 000244
3543 023200 005400
3544 023202 102403
3545 023204 103402
3546 023206 001001
3547 023210 100006

NEG3: MOV #467,-2(R2) ;MOVE TO MAILBOX # ***** 467 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;NEG DID NOT SET CC'S CORRECTLY
NEG4: CLR R0
SCC ;CC=1011
CLZ
NEG R0 ;CC=0100 R0=0
BVS NEG5
BCS NEG5
BNE NEG5
BPL TST220

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <== =
; CONDITIONAL BRANCH INST. AND <== =
; REPLACE THE MOVE INSTRUCTION <== =
; WHICH FOLLOWS W/ 731 <== =

023212
023212 012762 000470 177776
023220 005262 177774
023224 000000

NEG5: MOV #470,-2(R2) ;MOVE TO MAILBOX # ***** 470 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;NEG DID NOT SET CC'S CORRECTLY

; OR SEQUENCE ERROR

3548

3549

.SBTTL TEST # 220 - TEST CMP INSTRUCTION
 :*****
 :TEST 220 - TEST CMP INSTRUCTION
 :*****

023226	005212			TST220: INC	(R2)	;UPDATE TEST NUMBER
023230	022712	000220		CMP	#220,(R2)	;SEQUENCE ERROR?
023234	001070			BNE	TST221-10	;BR TO ERROR HALT ON SEQ ERROR
3550 023236	012700	000005		MOV	#5,R0	
3551 023242	000257			CCC		;CC=1010
3552 023244	000271			+SEN!SEC		
3553 023246	022700	000005		CMP	#5,R0	;CC=0101
3554 023252	101002			BHI	CMP1	
3555 023254	102401			BVS	CMP1	
3556 023256	100006			BPL	CMP2	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 766 <---

023260				CMP1:		
023260	012762	000471	177776	MOV	#471,-2(R2)	;MOVE TO MAILBOX # ***** 471 *****
023266	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
023272	000000			HALT		;CMP DID NOT SET CC'S CORRECTLY
3557 023274	012700	100000		CMP2:		
3558 023300	000277			MOV	#100000,R0	
3559 023302	000242			SCC		;CC=1101
3560 023304	020027	077777		CLV		
3561 023310	101402			CMP	R0,#77777	;CC=0010
3562 023312	102001			BLOS	CMP3	
3563 023314	100006			BVC	CMP3	
				BPL	CMP4	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 747 <---

023316				CMP3:		
023316	012762	000472	177776	MOV	#472,-2(R2)	;MOVE TO MAILBOX # ***** 472 *****
023324	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
023330	000000			HALT		;CMP DID NOT SET CC'S CORRECTLY
3564 023332	052700	040000		CMP4:		
3565 023336	000257			BIS	#40000,R0	;R0=140000
3566 023340	000264			CCC		;CC=0100
3567 023342	022700	040000		SEZ		
3568 023346	102003			CMP	#40000,R0	;CC=1011
3569 023350	103002			BVC	CMP5	
3570 023352	001401			BCC	CMP5	
3571 023354	100406			BEQ	CMP5	
				BMI	CMP6	

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 727 <---

023356				CMP5:		
023356	012762	000473	177776	MOV	#473,-2(R2)	;MOVE TO MAILBOX # ***** 473 *****
023364	005262	177774		INC	-4(R2)	;SET MSGTYP TO FATAL ERROR
023370	000000			HALT		;CMP DID NOT SET CC'S CORRECTLY
3572 023372	042700	040000		CMP6:		
3573 023376	000277			BIC	#40000,R0	
3574 023400	022700	177777		SCC		;CC=1111
3575 023404	101402			CMP	#-1,R0	;CC=0000
				BLOS	CMP7	

3576 023406 102401
3577 023410 100006

BVS CMP7
BPL TST221

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 711 <====

023412
023412 012762 000474 177776 CMP7:
023420 005262 177774
023424 000000

MOV #474,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 474 *****
:SET MSGTYP TO FATAL ERROR
:CMP DID NOT SET CC'S CORRECTLY
: OR SEQUENCE ERROR

3578

3579

.SBTTL TEST # 221 - TEST COM INSTRUCTION
:*****
:TEST 221 - TEST COM INSTRUCTION
:*****

023426 005212
023430 022712 000221
023434 001012
3580 023436 012700 177777
3581 023442 000257
3582 023444 000265
3583 023446 005100
3584 023450 101002
3585 023452 102401
3586 023454 100006

TST221: INC (R2) ;UPDATE TEST NUMBER
CMP #221,(R2) ;SEQUENCE ERROR?
BNE TST222-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #-1,R0
CCC ;CC=1010
+SEC.SEZ
COM R0 ;CC=0101
BHI COM1
BVS COM1
BPL TST222

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 767 <====

023456
023456 012762 000475 177776
023464 005262 177774
023470 000000

COM1: MOV #475,-2(R2) ;MOVE TO MAILBOX # ***** 475 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;COM DID NOT SET CC'S CORRECTLY
; OR SEQUENCE ERROR

3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598

:*****
: THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE SUB
: AND SBC INSTRUCTIONS. BOTH OF THESE INSTRUCTIONS HANDLE THE
: C AND V BITS IDENTICALLY. THE PROCEDURE IS TO PRESET THE CONDITION
: CODES, EXECUTE THE INSTRUCTION WITH A PARTICULAR SET OF DATA, AND
: THEN CHECK THE RESULTS BY EXECUTING A SERIES OF CONDITIONAL
: BRANCHES. THIS PROCEDURE IS REPEATED SEVERAL TIMES WITH DIFFERENT
: DATA PATTERNS TO PROVIDE EVERY COMBINATION OF THE C AND V BITS.
:

3599

.SBTTL TEST # 222 - TEST SUB INSTRUCTION

:TEST 222 - TEST SUB INSTRUCTION

023472	005212				TST222: INC	(R2)		:UPDATE TEST NUMBER
023474	022712	000222			CMP	#222,(R2)		:SEQUENCE ERROR?
023500	001065				BNE	TST223-10		:BR TO ERROR HALT ON SEQ ERROR
3600 023502	012700	125252			MOV	#125252,R0		
3601 023506	000257				CCC			:CC=1010
3602 023510	000271				+SEN. SEC			
3603 023512	162700	125252			SUB	#125252,R0		:CC=0101 R0=0
3604 023516	101002				BHI	SUB1		
3605 023520	102401				BVS	SUB1		
3606 023522	100006				BPL	SUB2		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <---
: WHICH FOLLOWS W/ 766 <---

023524					SUB1:			
023524	012762	000476	177776		MOV	#476,-2(R2)		:MOVE TO MAILBOX # ***** 476 *****
023532	005262	177774			INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
023536	000000				HALT			:SUB DID NOT SET CC'S CORRECTLY
3607 023540	052700	100000			SUB2:			
3608 023544	000277				BIS	#100000,R0		:CC=1101
3609 023546	000242				CLV			
3610 023550	162700	077777			SUB	#77777,R0		:CC=0010 R0=1
3611 023554	101402				BLOS	SUB3		
3612 023556	102001				BVC	SUB3		
3613 023560	100006				BPL	SUB4		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 747 <====

023562					SUB3:			
023562	012762	000477	177776		MOV	#477,-2(R2)		:MOVE TO MAILBOX # ***** 477 *****
023570	005262	177774			INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
023574	000000				HALT			
3614 023576	005100				SUB4:			
3615 023600	000277				COM	R0		:R0=177777
3616					SCC			:CC=11111
3617 023602	162700	100000			SUB	#100000,R0		:CC=0000 R0 77777
3618 023606	101402				BLOS	SUB5		
3619 023610	102401				BVS	SUB5		
3620 023612	100006				BPL	SUB6		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <---
: WHICH FOLLOWS W/ 732 <---

023614					SUB5:			
023614	012762	000500	177776		MOV	#500,-2(R2)		:MOVE TO MAILBOX # ***** 500 *****
023622	005262	177774			INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
023626	000000				HALT			:SUB DID NOT SET CC'S CORRECTLY
3621 023630	000257				SUB6:			
3622 023632	000264				CCC			:CC=0100
3623 023634	162700	140000			SEZ			
3624 023640	102003				SJB	#140000,R0		:CC=1011
3625 023642	103002				BVC	SUB7		
					BCC	SUB7		

3626 023644 001401
3627 023646 100406

BEQ SUB7
BMI TST223

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 714

023650
023650 012762 000501 177776 SUB7:
023656 005262 177774
023662 000000
3628

MOV #501,-2(R2)
INC -4(R2)
HALT

: MOVE TO MAILBOX # ***** 50. *****
: SET MSGTYP TO FATAL ERROR
:

3629

.SBTTL TEST # 223 - TEST SBC INSTRUCTION

 :TEST 223 - TEST SBC INSTRUCTION

023664 005212
 023666 022712 000223
 023672 001063
 3630 023674 012700 000001
 3631 023700 000277
 3632 023702 000244
 3633 023704 005600
 3634 023706 103403
 3635 023710 102402
 3636 023712 100401
 3637 023714 001406

TST223: INC (R2) ;UPDATE TEST NUMBER
 CMP #223,(R2) ;SEQUENCE ERROR?
 BNE TST224-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #1,R0
 SCC ;CC=1011
 CLZ
 SBC R0 ;CC=0100 R=0
 BCS SBC1
 BVS SBC1
 BMI SBC1
 BEQ SBC2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 765 <---

023716
 023716 012762 000502 177776
 023724 005262 177774
 023730 000000
 3638 023732 000277
 3639 023734 000245
 3640 023736 005600
 3641 023740 103403
 3642 023742 102402
 3643 023744 100401
 3644 023746 001406

SBC1: MOV #502,-2(R2) ;MOVE TO MAILBOX # ***** 502 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;SBC DID NOT SET CC'S CORRECTLY
 SBC2: SCC ;CC=1010
 +CLZ!CLC
 SBC R0 ;CC=0100 R=0
 BCS SBC3
 BVS SBC3
 BMI SBC3
 BEQ SBC4

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 751 <---

023750
 023750 012762 000503 177776
 023756 005262 177774
 023762 000000
 3645 023764 000277
 3646 023766 000250
 3647 023770 005600
 3648 023772 103003
 3649 023774 102402
 3650 023776 001401
 3651 024000 100406

SBC3: MOV #503,-2(R2) ;MOVE TO MAILBOX # ***** 503 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;SBC DID NOT SFT CC'S CORRECTLY
 SBC4: SCC ;CC=0111
 CLN
 SBC R0 ;CC=1001 R0=177777
 BCC SBC5
 BVS SBC5
 BEQ SBC5
 BMI SBC6

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 : CONDITIONAL BRANCH INST. AND <---
 : REPLACE THE MOVE INSTRUCTION <---
 : WHICH FOLLOWS W/ 734 <---

024002
 024002 012762 000504 177776
 024010 005262 177774
 024014 000000
 3652 024016 042700 077777
 3653 024022 000277
 3654 024024 000242
 3655 024026 005600

SBC5: MOV #504,-2(R2) ;MOVE TO MAILBOX # ***** 504 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;SBC DID NOT SET CC'S CORRECTLY
 SBC6: BIC #77777,R0 ;R0=100000
 SCC ;CC=1101
 CLV
 SBC R0 ;CC=0010

3656 024030 101402
 3657 024032 102001
 3658 024034 100006

BLOS SBC7
 BVC SBC7
 BPL TST224

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 716 <-

024036
 024036 012762 000505 177776
 024044 005262 177774
 024050 000000

SBC7:
 MOV #505,-2(R2)
 INC -4(R2)
 HALT

:MOVE TO MAILBOX # ***** 505 *****
 :SET MSGTYP TO FATAL ERROR
 :SBC DID NOT SET CC'S CORRECTLY
 : OR SEQUENCE ERROR

3659
 3660
 3661
 3662
 3663
 3664
 3665
 3666
 3667
 3668
 3669

.....
 :
 : THESE NEXT FOUR TESTS VERIFY THE FUNCTIONING OF THE ROL,
 :ROR, ASL AND ASR INSTRUCTIONS. SPECIAL DATA PATTERNS ARE LOADED
 :AND ROTATED SEVERAL TIMES FOR EACH TEST. THE CONDITION CODES
 :ARE PRESET BEFORE EACH ROTATION AND THE CONDITION CODES ARE
 :CHECKED AFTER EACH ROTATION. THE FINAL CHECK IN EACH TEST IS
 :TO VERIFY THE CUMULATIVE DATA RESULT. THE DATA PATTEPNS HAVE
 :BEEN SELECTED TO PRODUCE ALL COMBINATIONS OF THE C AND V BITS.
 :

3670

.SBTTL TEST # 224 - TEST ROL INSTRUCTION

 :TEST 224 - TEST ROL INSTRUCTION

024052 005212
 024054 022712 000224
 024060 001063
 3671 024062 012700 144000
 3672 024066 000257
 3673 024070 000266
 3674 024072 006100
 3675 024074 103003
 3676 024076 102402
 3677 024100 001401
 3678 024102 100406

TST224: INC (R2) ;UPDATE TEST NUMBER
 CMP #224,(R2) ;SEQUENCE ERROR?
 BNE TST225-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #144000,R0 ;R0=144000
 CCC ;CC=0110
 +SEZ,SEV
 ROL R0 ;CC=1001 R0=110000
 BCC ROL1
 BVS ROL1
 BEQ ROL1
 BMI ROL2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < -
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 766 <

024104
 024104 012762 000506 177776
 024112 005262 177774
 024116 000000
 3679 024120 000277
 3680 024122 000243
 3681 024124 006100
 3682 024126 103003
 3683 024130 102002
 3684 024132 001401
 3685 024134 100006

ROL1: MOV #506,-2(R2) ;MOVE TO MAILBOX # ***** 506 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT
 ROL2: SCC ;CC=1100
 +CLV!CLC
 ROL R0 ;CC=0011 R0=020000
 BCC ROL3
 BVC ROL3
 BEQ ROL3
 BPL ROL4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 751 <

024136
 024136 012762 000507 177776
 024144 005262 177774
 024150 000000
 3686 024152 000277
 3687 024154 000250
 3688 024156 006100
 3689 024160 101402
 3690 024162 102401
 3691 024164 100006

ROL3: MOV #507,-2(R2) ;MOVE TO MAILBOX # ***** 507 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROL DID NOT SET CC'S CORRECTLY
 ROL4: SCC ;CC=0111
 CLN
 ROL R0 ;CC=0000 R0=040001
 BLOS ROL5
 BVS ROL5
 BPL ROL6

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <-
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 735 <

024166
 024166 012762 000510 177776
 024174 005262 177774
 024200 000000
 3692 024202 000257
 3693 024204 000265
 3694 024206 006100
 3695 024210 101405
 3696 024212 102004

ROL5: MOV #510,-2(R2) ;MOVE TO MAILBOX # ***** 510 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROL DID NOT SET CC'S CORRECTLY
 ROL6: CCC ;CC=0101
 +SEZ!SEC
 ROL R0 ;CC=1010 R0=100003
 BLOS ROL7
 BVC ROL7

3697	024214	100003		BPL	ROL7
3698	024216	022700	100003	CMP	#100003,RC
3699	024222	001406		BEQ	TST25

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < ---
; CONDITIONAL BRANCH INST. AND <
; REPLACE THE MOVE INSTRUCTION <
; WHICH FOLLOWS W/ 716 <

024224				ROL7:	
024224	012762	000511	177776	MOV	#511,-2(R2)
024232	005262	177774		INC	-4,R2)
024236	000000			HALT	

;MOVE TO MAILBOX # ***** 511 *****
;SET MSGTYP TO FATAL ERROR
;ROL MALFUNCTIONED
; OR SEQUENCE ERROR

3700

.SBTTL TEST # 225 - TEST ROR INSTRUCTION

 :TEST 225 - TEST ROR INSTRUCTION

024240 005212
 024242 022712 000225
 024246 001061
 3701 024250 012700 000023
 3702 024254 000277
 3703 024256 000250
 3704 024260 006000
 3705 024262 102403
 3706 024264 103002
 3707 024266 001401
 3708 024270 100406

TST225: INC (R2) ;UPDATE TEST NUMBER
 CMP #225,(R2) ;SEQUENCE ERROR?
 BNE TST226-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #23,R0 ;R0=23
 SCC ;CC=0111
 CLN
 ROR R0 ;CC=1001 R0=100011
 BVS ROR1
 BCC ROR1
 BEQ ROR1
 BMI ROR2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 ; CONDITIONAL BRANCH INST. AND <=====
 ; REPLACE THE MOVE INSTRUCTION <=====
 ; WHICH FOLLOWS W/ 766 <=====
 ;

024272
 024272 012762 000512 177776
 024300 005262 177774
 024304 000000
 3709 024306 000257
 3710 024310 000274
 3711 024312 006000
 3712 024314 102003
 3713 024316 103002
 3714 024320 001401
 3715 024322 100006

ROR1: MOV #512,-2(R2) ;MOVE TO MAILBOX # ***** 512 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROR DID NOT SET CC'S CORRECTLY
 ROR2: CCC ;CC=1100
 +SEN!SEZ
 ROR R0 ;CC=0011 R0=040004
 BVC ROR3
 BCC ROR3
 BEQ ROR3
 BPL ROR4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <--
 ; REPLACE THE MOVE INSTRUCTION <-
 ; WHICH FOLLOWS W/ 751 <

024324
 024324 012762 000513 177776
 024332 005262 177774
 024336 000000
 3716 024340 000277
 3717 024342 000241
 3718 024344 006000
 3719 024346 101403
 3720 024350 102402
 3721 024352 001401
 3722 024354 100006

ROR3: MOV #513,-2(R2) ;MOVE TO MAILBOX # ***** 513 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROR DID NOT SET CC'S CORRECTLY
 ROR4: SCC ;CC=1110
 CLC
 ROR R0 ;CC=0000 R0=020002
 BLOS ROR5
 BVS ROR5
 BEQ ROR5
 BPL ROR6

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <--
 ; REPLACE THE MOVE INSTRUCTION <-
 ; WHICH FOLLOWS W/ 734 <

024356
 024356 012762 000514 177776
 024364 005262 177774
 024370 000000
 3723 024372 000257
 3724 024374 000265
 3725 024376 006000
 3726 024400 101402

ROR5: MOV #514,-2(R2) ;MOVE TO MAILBOX # ***** 514 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ROR DID NOT SET CC'S CORRECTLY
 ROR6: CCC ;CC=0101
 +SEC.SEZ
 ROR R0 ;CC=1010 R0=110001
 BLOS ROR7

3727 024402 102001
3728 024404 100406

BVC ROR7
BMI TST226

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 720 <====

024406
024406 012762 000515 177776 ROR7:
024414 005262 177774
024420 000000

MOV #515,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 515 *****
:SET MSGTYP TO FATAL ERROR
:ROR DID NOT PRODUCE CORRECT RESULTS
: OR SEQUENCE ERROR

3729

.SBTTL TEST # 226 - TEST ASL INSTRUCTION

 :TEST 226 - TEST ASL INSTRUCTION

024422 005212
 024424 022712 000226
 024430 001064
 3730 024432 012700 144000
 3731 024436 000257
 3732 024440 000271
 3733 024442 006300
 3734 024444 103003
 3735 024446 102402
 3736 024450 001401
 3737 024452 100406

TST226: INC (R2) ;UPDATE TEST NUMBER
 CMP #226,(R2) ;SEQUENCE ERROR?
 BNE TST227-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #144000,R0 ;R0=14000
 CCC ;CC=0110
 +SEN.SEC
 ASL R0 ;CC=1001 R0=110000
 BCC ASL1
 BVS ASL1
 BEQ ASL1
 BMI ASL2

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <---
 ; REPLACE THE MOVE INSTRUCTION <----
 ; WHICH FOLLOWS W/ 766 <----

024454
 024454 012762 000516 177776
 024462 005262 177774
 024466 000000
 3738 024470 000277
 3739 024472 000243
 3740 024474 006300
 3741 024476 103003
 3742 024500 102002
 3743 024502 001401
 3744 024504 100006

ASL1: MOV #516,-2(R2) ;MOVE TO MAILBOX # ***** 516 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT
 ASL2: SCC ;CC=1100
 +CLV!CLC
 ASL R0 ;CC=0011 R0=020000
 BCC ASL3
 BVC ASL3
 BEQ ASL3
 BPL ASL4

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 751 <

024506
 024506 012762 000517 177776
 024514 005262 177774
 024520 000000
 3745 024522 000277
 3746 024524 000250
 3747 024526 006300
 3748 024530 101402
 3749 024532 102401
 3750 024534 100006

ASL3: MOV #517,-2(R2) ;MOVE TO MAILBOX # ***** 517 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ASL DID NOT SET CC'S CORRECTLY
 ASL4: SCC ;CC=0111
 CLN
 ASL R0 ;CC=0000 R0=040000
 BLOS ASL5
 BVS ASL5
 BPL ASL6

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <-
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 735 <

024536
 024536 012762 000520 177776
 024544 005262 177774
 024550 000000
 3751 024552 000257
 3752 024554 000265
 3753 024556 006300
 3754 024560 103406
 3755 024562 001405

ASL5: MOV #520,-2(R2) ;MOVE TO MAILBOX # ***** 520 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;ASL DID NOT SET CC'S CORRECTLY
 ASL6: CCC ;CC=0101
 +SEZ!SEC
 ASL R0 ;CC=1010 R0=100000
 BCS ASL7
 BEQ ASL7

3756 024564 102004
3757 024566 100003
3758 024570 022700 100000
3759 024574 001406

BVC ASL7
BPL ASL7
CMP #100000,R0
BEQ TST227

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <==.-
; WHICH FOLLOWS W/ 715 <--.-

024576
024576 012762 000521 177776 ASL7:
024604 005262 177774
024610 000000

MOV #521,-2(R2)
INC -4(R2)
HALT

;MOVE TO MAILBOX # ***** 521 *****
;SET MSGTYP TO FATAL ERROR
;ASL MALFUNCTIONED
; OR SEQUENCE ERROR

3760

.SBTTL TEST # 227 - TEST ASR INSTRUCTION

:TEST 227 - TEST ASR INSTRUCTION

024612 005212
024614 022712 000227
024620 001070
3761 024622 012700 100023
3762 024626 000277
3763 024630 000250
3764 024632 006200
3765 024634 102403
3766 024636 103002
3767 024640 001401
3768 024642 100406

TST227: INC (R2) ;UPDATE TEST NUMBER
CMP #227,(R2) ;SEQUENCE ERROR?
BNE TST230-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #100023,R0 ;R0=100023
SCC ;CC=0110
CLN
ASR R0 ;CC=1001 RP=140011
BVS ASR1
BCC ASR1
BEQ ASR1
BMI ASR2

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 766 <

024644
024644 012762 000522 177776
024652 005262 177774
024656 000000
3769 024660 042700 100000
3770 024664 000277
3771 024666 000243
3772 024670 006200
3773 024672 102003
3774 024674 103002
3775 024676 001401
3776 024700 100006

ASR1: MOV #522,-2(R2) ;MOVE TO MAILBOX # ***** 522 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ASR DID NOT SET CC'S CORRECTLY
ASR2: BIC #100000,R0 ;R0=40011
SCC ;CC=1100
+CLV!CLC
ASR R0 ;CC=0011 R0=020004
BVC ASR3
BCC ASR3
BEQ ASR3
BPL ASR4

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
: CONDITIONAL BRANCH INST. AND <=
: REPLACE THE MOVE INSTRUCTION <--
: WHICH FOLLOWS W/ 747 <--

024702
024702 012762 000523 177776
024710 005262 177774
024714 000000
3777 024716 000277
3778
3779 024720 006200
3780 024722 101403
3781 024724 102402
3782 024726 001401
3783 024730 100006

ASR3: MOV #523,-2(R2) ;MOVE TO MAILBOX # ***** 523 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ASR DID NOT SET CC'S CORRECTLY
ASR4: SCC ;CC=1111
ASR R0 ;CC=0000 R0=010002
BLOS ASR5
BVS ASR5
BEQ ASR5
BPL ASR6

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 733 <====

024732
024732 012762 000524 177776
024740 005262 177774
024744 000000
3784 024746 052700 100000
3785 024752 000257
3786 024754 000265

ASR5: MOV #524,-2(R2) ;MOVE TO MAILBOX # ***** 524 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ASR DID NOT SET CC'S CORRECTLY
ASR6: BIS #100000,R0 ;R0=110002
CCC ;CC=0101
+SEZ!SEC

```

3787 024756 006200          ASR      R0          ;C=1010  R0=144001
3788 024760 101406          BLOS   ASR7
3789 024762 102005          BVC    ASR7
3790 024764 100004          BPL    ASR7
3791 024766 001403          BEQ    ASR7
3792 024770 022700 144001   CMP     #144001,R0   ;CHECK RESULT OF ASR'S
3793 024774 001406          BEQ    TST230

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
; CONDITIONAL BRANCH INST. AND
; REPLACE THE MOVE INSTRUCTION
; WHICH FOLLOWS W/ 711
;-----

024776          ASR7:
024776 012762 000525 177776   MOV     #525,-2(R2) ;MOVE TO MAILBOX # ***** 525 *****
025004 005262 177774          INC     -4(R2)      ;SET MSGTYP TO FATAL ERROR
025010 000000          HALT           ;ASR DID NOT FUNCTION CORRECTLY
; OR SEQUENCE ERROR

```

3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810

```

:*****
:
: THIS TEST VERIFIES THE SXT INSTRUCTION. CONDITION CODES
: ARE PRESET IN EACH OF THE TWO POSSIBLE CASES. WITH THE N-BIT SET,
: THE TEST CHECKS FOR ALL ONES IN THE DESTINATION. WITH THE N-BIT
: CLEAR, THE DESTINATION SHOULD CONTAIN ALL ZEROS. THE DATA
: IS VERIFIED BY CONDITIONAL BRANCHES.
:
:

```

3811

.SBTTL TEST # 230 - TEST THE SXT INSTRUCTION
 :*****
 :TEST 230 - TEST THE SXT INSTRUCTION
 :*****

025012 005212
 025014 022712 000230
 025020 001037
 3812 025022 005000
 3813 025024 000277
 3814 025026 000244
 3815 025030 006700
 3816 025032 100006
 3817 025034 001405
 3818 025036 102404
 3819 025040 103003
 3820 025042 022700 177777
 3821 025046 001406

TST230: INC (R2) ;UPDATE TEST NUMBER
 CMP #230,(R2) ;SEQUENCE ERROR?
 BNE TST231-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR R0
 SCC ;SET CC=1011
 CLZ
 SXT R0 ;TRY SXT
 BPL SXT0 ;TEST CC=1001
 BEQ SXT0
 BVS SXT0
 BCC SXT0
 CMP #-1,R0 ;CHECK DATA RESULT
 BEQ SXT1
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <---
 ; CONDITIONAL BRANCH INST. AND <---
 ; REPLACE THE MOVE INSTRUCTION <---
 ; WHICH FOLLOWS W/ 764 <---

025050
 025050 012762 000526 177776
 025056 005262 177774
 025062 000000
 3822 025064 005000
 3823 025066 005010
 3824 025070 005110
 3825 025072 000257
 3826 025074 000266
 3827 025076 006710
 3828 025100 001005
 3829 025102 103404
 3830 025104 102403
 3831 025106 100402
 3832 025110 005710
 3833 025112 001405

SXT0: MOV #526,-2(R2) ;MOVE TO MAILBOX # ***** 526 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;RESULTS OF SXT INCORRECT
 SXT1: CLR R0 ;R0=0
 CLR (R0) ;LOC. 0=0
 COM (R0) ;LOC. 0=177777
 CCC ;SET CC=0110
 +SEZ!SEV
 SXT (R0)
 BNE SXT2 ;TEST CC=0100
 BCS SXT2
 BVS SXT2
 BMI SXT2
 TST (R0)
 BEQ TST231
 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
 ; CONDITIONAL BRANCH INST. AND <=
 ; REPLACE THE MOVE INSTRUCTION <=
 ; WHICH FOLLOWS W/ 742 <==

025114
 025114 012762 000527 177776
 025122 005262 177774
 025126 000000

SXT2: MOV #527,-2(R2) ;MOVE TO MAILBOX # ***** 527 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;RESULTS OF SXT INCORRECT
 ; OR SEQUENCE ERROR

3834
 3835
 3836
 3837
 3838
 3839
 3840
 3841

:*****
 : THIS TEST VERIFIES THE XOR INSTRUCTION. UNIQUE PATTERNS
 : OF ONES AND ZEROES ARE MOVED TO DATA REGISTERS R0 AND R1.
 : AFTER THE FIRST XOR INSTRUCTION R0=36146. AN XOR IS THEN
 : EXECUTED WITH THIS NEW VALUE AND THE CONTENTS OF R1 TO
 : REPRODUCE THE ORIGINAL VALUE IF R0=31525.
 :*****

3842

.SBTTL TEST # 231 - TEST THE XOR INSTRUCTION

:TEST 231 - TEST THE XOR INSTRUCTION

025130	005212			TST231: INC	(R2)		:UPDATE TEST NUMBER
025132	022712	000231		CMP	#231,(R2)		:SEQUENCE ERROR?
025136	001041			BNE	TST232-10		:BR TO ERROR HALT ON SEQ ERROR
3843 025140	012700	007463		MOV	#7463,R0		:SET UP R0
3844 025144	012701	031525		MOV	#31525,R1		:SET UP R1
3845 025150	000277			SCC			:SET CC=1110
3846 025152	000241			CLC			
3847 025154	074100			XOR	R1,R0		:TRY XOR
3848 025156	101406			BLOS	XOR1		:CC=0000?
3849 025160	102405			BVS	XOR1		
3850 025162	001404			BEQ	XOR1		
3851 025164	100403			BMI	XOR1		
3852 025166	022700	036146		CMP	#36146,R0		:DATA RESULT CORRECT?
3853 025172	001406			BEQ	XOR2		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <----
: CONDITIONAL BRANCH INST. AND <---
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 761 <----

025174				XOR1:			
025174	012762	000530	177776	MOV	#530,-2(R2)		:MOVE TO MAILBOX # ***** 530 *****
025202	005262	177774		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
025206	000000			HALT			:
3854 025210	010104			XOR2:			
3855 025212	000261			MOV	R1,R4		
3856 025214	000241			SEC			:CC=1110
3857 025216	074400			CLC			
3858 025220	101406			XOR	R4,R0		:TRY XOR MODE 0,0
3859 025222	102405			BLOS	XOR3		:CC=0000?
3860 025224	001404			BVS	XOR3		
3861 025226	100403			BEQ	XOR3		
3862 025230	022700	007463		BMI	XOR3		
3863 025234	001406			CMP	#7463,R0		
				BEQ	TST232		

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 740 <====

025236				XOR3:			
025236	012762	000531	177776	MOV	#531,-2(R2)		:MOVE TO MAILBOX # ***** 531 *****
025244	005262	177774		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
025250	000000			HALT			:RESULT OF XOR INCORRECT
							: OR SEQUENCE ERROR

3864
3865
3866
3867
3868
3869
3870

: THIS TEST VERIFIES THE SOB INSTRUCTION. R4 IS USED AS A
: COUNTER WHILE R0 IS THE ADDRESS REGISTER. CONDITIONAL
: BRANCHES ARE USED TO VERIFY PROPER TRANSFER OF CONTROL
: WHILE R4 IS CHECKED TO INSURE PROPER DECREMENTING OF R0.
:

3871

.SBTTL TEST # 232 - TEST SOB INSTRUCTION
 :*****
 :TEST 232 - TEST SOB INSTRUCTION
 :*****

025252 005212
 025254 022712 000232
 025260 061027
 3872 025262 012700 000525
 3873 025266 010004
 3874 025270 000277
 3875 025272 101002
 3876 025274 100001
 3877 025276 102406

TST232: INC (R2) ;UPDATE TEST NUMBER
 CMP #232,(R2) ;SEQUENCE ERROR?
 BNE TST233-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #525,R0
 MOV R0,R4
 SOB1: SCC ;SET CC=1111
 BHI SOB2 ;CC=1111?
 BPL SOB2
 BVS SOB3

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 770 <====

025300
 025300 012762 000532 177776
 025306 005262 177774
 025312 000000
 3878 025314 005304
 3879 025316 000277
 3880 025320 077014
 3881 025322 101004
 3882 025324 100003
 3883 025326 102002
 3884 025330 005704
 3885 025332 001406

SOB2: MOV #532,-2(R2) ;MOVE TO MAILBOX # ***** 532 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT
 SOB3: DEC R4 ;COUNT ITERATIONS
 SCC ;CC=1111
 SOB RO,SOB1 ;DO SOB W/ R0
 BHI SOB4 ;CHECK CC=1111
 BPL SOB4
 BVC SOB4
 TST R4 ;ITERATION COUNT OK?
 BEQ TST233

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < -
 : CONDITIONAL BRANCH INST. AND < -
 : REPLACE THE MOVE INSTRUCTION < =
 : WHICH FOLLOWS W/ 752 <

025334
 025334 012762 000533 177776
 025342 005262 177774
 025346 000000

SOB4: MOV #533,-2(R2) ;MOVE TO MAILBOX # ***** 533 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;INCORRECT # OF BRANCHES OR CC'S CHANGED
 ; OR SEQUENCE ERROR

3886
 3887
 3888
 3889
 3890
 3891
 3892

:*****
 : THIS TEST VERIFIES THE MARK INSTRUCTION. THE EFFECTS
 : OF THE MARK INSTRUCTION ARE SIMULATED BY THE PROGRAM INSTRUCTIONS.
 : THE CONTENTS OF R5 AND THE STACK POINTER ARE CHECKED AFTER EACH
 : OF THE TWO ROUTINES IN THE TEST.
 :*****

3893

.SHTTL TEST # 233 - TEST MARK INSTRUCTION
:*****
:TEST 233 - TEST MARK INSTRUCTION
:*****

025350 005212
025352 022712 000233
025356 001072
3894 025360 012706 001000
3895 025364 012746 125252
3896 025370 162706 000074
3897 025374 012705 025426
3898 025400 012746 006436
3899 025404 000277
3900 025406 000137 000700
3901 025412 012762 000534 177776
025420 005262 177774
025424 000000
3902 025426 101010
3903 025430 100007
3904 025432 102006
3905 025434 020527 125252
3906 025440 001003
3907 025442 022706 001000
3908 025446 001406

TST233: INC (R2) ;UPDATE TEST NUMBER
CMP #233,(R2) ;SEQUENCE ERROR?
BNE TST234-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #STBOT,SP
MOV #125252,-(SP) ;PUT R5 VALUE ON STACK
SUB #74,SP ;EFFECTIVELY PUT 36 ARGUMENTS ON STACK
MOV #MRK1,R5 ;SET NEW PC IN R5
MOV #6436,-(SP) ;PUT MARK 36 INST. ON STACK
SCC ;SET CC=1111
JMP @#700 ;XFER CONTL TO MARK 36 INST. ON STACK
MOV #534,-2(R2) ;MOVE TO MAILBOX # ***** 534 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;MARK INST. SHOULD HAVE JUMPED TO MRK1
MRK1: BHI MRK2 ;TEST CC UNAFFECTED
BPL MRK2 ;IE. CC=1111
BVC MRK2
CMP R5,#125252 ;CHECK R5 RESTORED FROM STACK
BNE MRK2
CMP #STBOT,R6 ;CHECK STACK POINTER READJUSTED CORRECTLY.
BEQ MRK3

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 743 <

025450
025450 012762 000535 177776
025456 005262 177774
025462 000000
3909 025464 012746 052525
3910 025470 012746 006400
3911 025474 010605
3912 025476 004737 025506
3913 025502 000137 025524
3914 025506 000205
3915 025510 012762 000536 177776
025516 005262 177774
025522 000000
3916 025524 022706 001000
3917 025530 001003
3918 025532 022705 052525
3919 025536 001406

MRK2: MOV #535,-2(R2) ;MOVE TO MAILBOX # ***** 535 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULTS OF MARK INCORRECT
MRK3: MOV #52525,-(SP) ;PUT MARK 0 INST. ON STACK
MOV #6400,-(SP) ;SET ADDR. OF MARK INST. IN R5
MOV SP,R5 ;DO JSR
JSR PC,@MRK4
JMP @MRK5
MRK4: RTS R5 ;DO RTS WITH R5 TO MARK INST ON STACK
MOV #536,-2(R2) ;MOVE TO MAILBOX # ***** 536 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RTS,MARK SEQUENCE FAILED
MRK5: CMP #STBOT,R6 ;STACK ADJUSTED CORRECTLY
BNE MRK6 ;IF NOT: BR
CMP #52525,R5 ;CHECK IF R5 RESTORED FROM STACK
BEQ TST234

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
: CONDITIONAL BRANCH INST. AND <- -
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 707 <-

025540
025540 012762 000537 177776
025546 005262 177774
025552 000000

MPK6: MOV #537,-2(R2) ;MOVE TO MAILBOX # ***** 537 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULTS OF MARK INCORRECT
; OR SEQUENCE ERROR

3920
3921
3922

:*****
: THIS TEST VERIFIES THAT RESET DOES NOT CLEAR THE PSW.
:*****

3923
3924
3925
3926
3927

:THE PSW IS LOADED WITH ONES, A RESET IS ISSUED, AND THE
:CONTENTS OF THE PSW ARE CHECKED TO VERIFY THAT THEY HAVE NOT
:CHANGED. THIS TEST IS EXECUTED ONLY ONCE EVERY 256 (DECIMAL)
:PASSES.
:

3928

.SBTTL TEST # 234 - TEST THAT RESET DOES NOT CLEAR PSW

 :TEST 234 - TEST THAT RESET DOES NOT CLEAR PSW

025554 005212
 025556 022712 000234
 025562 001016
 3929 025564 123727 042100 000377
 3930 025572 001016
 3931 025574 012737 000357 177776
 3932 025602 000005
 3933 025604 022737 000357 177776
 3934 025612 001406

TST234: INC (R2) ;UPDATE TEST NUMBER
 CMP #234,(R2) ;SEQUENCE ERROR?
 BNE TST235-10 ;BR TO ERROR HALT ON SEQ ERROR
 CMPB @PASSPT,#377 ;ONLY DUE RESET EVERY 256. PASSES
 BNE REST ;BR IF TO SKIP TEST
 MOV #357,@#PS ;MOV ONES TO PSW
 RESET
 CMP #357,@#PS ;PSW CORRECT?
 BEQ TST235

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
 : CONDITIONAL BRANCH INST. AND <--
 : REPLACE THE MOVE INSTRUCTION <--
 : WHICH FOLLOWS W/ 763 <--
 : MOVE TO MAILBOX # ***** 540 *****
 : SET MSGTYP TO FATAL ERROR
 : RESET ALTERED PSW
 : OR SEQUENCE ERROR

025614 012762 000540 177776
 025622 005262 177774
 025626 000000

MOV #540,-2(R2)
 INC -4(R2)
 HALT

3935 025630
 3936
 3937
 3938
 3939
 3940
 3941

REST:

 : THE FOLLOWING TEST CHECKS THE INDEPENDENT FUNCTIONING OF BASIC
 : DATA PATH COMPONENTS WITH USER/SUPERVISOR MODE SET.
 :

3942

.SBTTL TEST # 235 - TEST USER/SUPER S.P. CAN HOLD A 1 IN EVERY BIT
 :*****
 :TEST 235 - TEST USER/SUPER S.P. CAN HOLD A 1 IN EVERY BIT
 :*****

025630 005212
 025632 022712 000235
 025636 001040
 3943 025640 052767 140000 152130
 3944 025646 012706 000001
 3945 025652 000241
 3946 025654 006106
 3947 025656 103376
 3948 025660 042767 140000 152110
 3949 025666 001406

TST235: INC (R2) ;UPDATE TEST NUMBER
 CMP #235,(R2) ;SEQUENCE ERROR?
 BNE TST236-10 ;BR TO ERROR HALT ON SEQ ERROR
 BIS #USRM,PS ;SET USER MODE
 MOV #1,R6 ;SET BIT0
 CLC ;CLEAR C-BIT
 USP1: ROL R6 ;ROTATE 1 POSITION
 BCC USP1 ;BR IF NOT ALL DONE
 BIC #USRM,PS ;CLEAR USER MODE
 BEQ SSP1A

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 763 <=====
 :

025670 012762 000541 177776
 025676 005262 177774
 025702 000000
 3950 025704 042767 100000 152064
 3951 025712 012706 000001
 3952 025716 000241
 3953 025720 006106
 3954 025722 103376
 3955 025724 042767 140000 152044
 3956 025732 001406

MOV #541,-2(R2) ;MOVE TO MAILBOX # ***** 541 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;USER MODE R6 PICKED A BIT
 SSP1A: BIC #100000,PS ;SET SUPERVISION MODE
 MOV #1,R6 ;SET BIT0
 CLC ;CLEAR C-BIT
 SSP2: ROL R6 ;ROTATE 1 POSITION
 BCC SSP2 ;BR IF NOT ALL DONE
 BIC #140000,PS ;CLEAR SUPERVISION MODE
 BEQ TST236

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 741 <=====
 :

025734 012762 000542 177776
 025742 005262 177774
 025746 000000

MOV #542,-2(R2) ;MOVE TO MAILBOX # ***** 542 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;SUPER MODE R6 PICKED A BIT
 ; OR SEQUENCE ERROR

3957 025750
 3958
 3959
 3960
 3961
 3962
 3963
 3964
 3965

USP2:
 :*****
 : THIS TEST CHECKS THE INDEPENDENT FUNCTIONING OF THE USER
 : SUPERVISOR AND KERNEL MODE R6'S. R6 IS SETUP AND ADDRESSED IN EACH
 : OF THE TWO MODES TO VERIFY THAT THE TWO R6'S ARE INDEPENDENT
 : OF EACH OTHER.
 :

3966

.SBTTL TEST # 236 - TEST INDEPENDENCE OF USER/SUPER/KERNEL MODE R6,R6
 :*****
 :TEST 236 - TEST INDEPENDENCE OF USER/SUPER/KERNEL MODE R6,R6
 :*****

025750 005212
 025752 022712 000236
 025756 001134
 3967 025760 052767 140000 152010
 3968 025766 012706 011111
 3969 025772 042767 100000 151776
 3970 026000 012706 022222
 3971 026004 042767 140000 151764
 3972 026012 012706 033333
 3973 026016 022706 033333
 3974 026022 001406

TST236: INC (R2) ;UPDATE TEST NUMBER
 CMP #236,(R2) ;SEQUENCE ERROR?
 BNE TST237-10 ;BR TO ERROR HALT ON SEQ ERROR
 BIS #USRM,PS ;SET USER MODE
 MOV #011111,R6 ;SET USER R6 TO #011111
 BIC #100000,PS ;SET SUPERVISOR MODE
 MOV #022222,R6 ;SET SUPER R6 TO #022222
 BIC #140000,PS ;SET KERNEL MODE
 MOV #033333,R6 ;SET KERNEL R6 TO #033333
 CMP #033333,R6 ;VERIFY R6 WITH KERNEL
 BEQ USP2A

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 755 <--

026024 012762 000543 177776
 026032 005262 177774
 026036 000000
 3975 026040 052767 040000 151730 USP2A:
 3976 026046 022706 022222
 3977 026052 001406

MOV #543,-2(R2) ;MOVE TO MAILBOX # ***** 543 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DUAL ADDRESSING SEQUENCE ERROR
 BIS #040000,PS ;SET SUPER MODE
 CMP #022222,R6 ;VERIFY R6 WITH SUPER
 BEQ USP3

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
 : CONDITIONAL BRANCH INST. AND <--
 : REPLACE THE MOVE INSTRUCTION <--
 : WHICH FOLLOWS W/ 741 <--

026054 012762 000544 177776
 026062 005262 177774
 026066 000000
 3978 026070 052767 140000 151700 USP3:
 3979 026076 022706 011111
 3980 026102 001406

MOV #544,-2(R2) ;MOVE TO MAILBOX # ***** 544 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DUAL ADDRESSING SEQUENCE ERROR
 BIS #USRM,PS ;SET USER MODE
 CMP #011111,R6 ;VERIFY R6 WITH USER
 BEQ USP4

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
 : CONDITIONAL BRANCH INST. AND <--
 : REPLACE THE MOVE INSTRUCTION <--
 : WHICH FOLLOWS W/ 725 <--

026104 012762 000545 177776
 026112 005262 177774
 026116 000000
 3981 026120 042767 140000 151650 USP4:
 3982 026126 012706 044444
 3983 026132 052767 040000 151636
 3984 026140 012706 055555
 3985 026144 052767 140000 151624
 3986 026152 012706 066666
 3987 026156 022706 066666
 3988 026162 001406

MOV #545,-2(R2) ;MOVE TO MAILBOX # ***** 545 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DUAL ADDRESSING SEQUENCE ERROR
 BIS #140000,PS ;SET KERNEL MODE
 MOV #044444,R6 ;SET R6 TO #044444
 BIS #040000,PS ;SET SUPER MODE
 MOV #055555,R6 ;SET R6 TO #055555
 BIS #140000,PS ;SET USER MODE
 MOV #066666,R6 ;SET R6 TO #066666
 CMP #066666,R6 ;CHECK R6 TO #066666
 BEQ USP5

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 675 <

026164 012762 000546 177776
 026172 005262 177774
 026176 000000

MOV #546,-2(R2) ;MOVE TO MAILBOX # ***** 546 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;DUAL ADDRESSING SEQUENCE ERROR

3989 026200 042767 100000 151570 USP5: BIC #100000,PS
3990 026206 022706 055555 CMP #055555,R6
3991 026212 001406 BEQ USP6

;SET SUPER MODE
;CHECK R6 #055555

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 661 <====

026214 012762 000547 177776 MOV #547,-2(R2)
026222 005262 177774 INC -4(R2)
026226 000000 HALT
3992 026230 042767 140000 151540 USP6: BIC #140000,PS
3993 026236 022706 044444 CMP #044444,R6
3994 026242 001406 BEQ TST237

;MOVE TO MAILBOX # ***** 547 *****
;SET MSGTYP TO FATAL ERROR
;DUAL ADDRESSING SEQUENCE ERROR
;SET KERNEL MODE
;CHECK R6 FOR #055555

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 645 <====

026244 012762 000550 177776 MOV #550,-2(R2)
026252 005262 177774 INC -4(R2)
026256 000000 HALT

;MOVE TO MAILBOX # ***** 550 *****
;SET MSGTYP TO FATAL ERROR
;DUAL ADDRESSING SEQUENCE ERROR
; OR SEQUENCE ERROR

3995
3996
3997
3998
3999
4000

.....
: THESE NEXT TWO TESTS VERIFY MFPI AND MTPI INSTRUCTIONS
: WITH R6 IN MODE 0.
:

4001

.SBTTL TEST # 237 - TEST MFPI WITH R6 IN MODE 0

 :TEST 237 - TEST MFPI WITH R6 IN MODE 0

026260 005212
 026262 022712 000237
 026266 001037
 4002 026270 012706 001000
 4003 026274 012767 140000 151474
 4004 026302 012706 042402
 4005 026306 006506
 4006 026310 022767 140000 151460
 4007 026316 001412
 4008 026320 042767 140000 151450
 4009 026326 001423

TST237: INC (R2) ;UPDATE TEST NUMBER
 CMP #237,(R2) ;SEQUENCE ERROR?
 BNE TST240-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #STBOT,R6 ;INITIALIZE KERNEL STACK POINTER
 MOV #USRM,PS ;SET USER MODE.PREVIOUS KERNEL
 MOV #USTBOT,R6 ;INITIALIZE USER STACK POINTER
 MFPI R6 ;TRY MFPI WITH MODE 0
 CMP #140000,PS ;CHECK PSW
 BEQ MFPI0 ;BR IF NO ERROR
 BIC #USRM,PS ;CLEAR USER MODE
 BEQ TST240

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 757 <=====
 :

026330 012762 000551 177776
 026336 005262 177774
 026342 000000
 4010 026344 022767 001000 014026 MFPI0:
 4011 026352 001411
 4012 026354 042767 140000 151414
 4013 026362 012762 000552 177776
 026370 005262 177774
 026374 000000
 4014 026376
 4015

MOV #551,-2(R2) ;MOVE TO MAILBOX # ***** 551 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;INCORRECT PSW FROM MFPI
 ; OR SEQUENCE ERROR
 MFPI0: CMP #STBOT,USTBOT-2 ;CHECK DATA ON STACK
 BEQ MFPI0A ;BR IF NO ERROR
 BIC #USRM,PS ;CLEAR USER MODE
 MOV #552,-2(R2) ;MOVE TO MAILBOX # ***** 552 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;INCORRECT DATA FROM MFPI

MFPI0A:

4016

```

026376 005212
026400 022712 000240
026404 001037
4017 026406 005067 151364
4018 026412 005006
4019 026414 012767 140000 151354
4020 026422 012706 042402
4021 026426 012746 001000
4022 026432 006606
4023 026434 022767 140000 151334
4024 026442 001411
4025 026444 042767 140000 151324
4026 026452 012762 000553 177776
026460 005262 177774
026464 000000
4027 026466 005067 151304
4028 026472 020627 001000
4029 026476 001406
  
```

```

.SBTTL TEST # 240 - TEST MTPJ WITH R6 IN MODE 0
*****
:TEST 240 - TEST MTPJ WITH R6 IN MODE 0
*****
TST240: INC      (R2)          ;UPDATE TEST NUMBER
        CMP      #240,(R2)    ;SEQUENCE ERROR?
        BNE     TST241-10    ;BR TO ERROR HALT ON SEQ ERROR
        CLR     PS           ;SET KERNEL MODE
        CLR     R6          ;INITIALIZE KERNEL R6
        MOV     #USRM,PS     ;SET USER MODE/PREVIOUS KERNEL
        MOV     #USTBOT,R6   ;INITIALIZE USER STACK POINTER
        MOV     #STBOT,-(R6) ;SET UP TARGET DATA
        MTPJ    R6          ;TRY MODE 0 MTPJ
        CMP     #USRM,PS     ;CHECK PSW
        BEQ     MTPJ0       ;BR IF NO ERROR
        BIC     #USRM,PS     ;CLEAR USER MODE
        MOV     #553,-2(R2)   ;MOVE TO MAILBOX # ***** 553 *****
        INC     -4(R2)       ;SET MSGTYP TO FATAL ERROR
        HALT                    ;PS INCORRECT FOLLOWING MTPJ
MTPJ0:  CLR     PS           ;SET KERNEL MODE
        CMP     R6,#STBOT    ;CHECK TARGET DATA
        BEQ     TST241
        ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < -
        ; CONDITIONAL BRANCH INST. AND <
        ; REPLACE THE MOVE INSTRUCTION <
        ; WHICH FOLLOWS W/ 742 <
        MOV     #554,-2(R2)   ;MOVE TO MAILBOX # ***** 554 *****
        INC     -4(R2)       ;SET MSGTYP TO FATAL ERROR
        HALT                    ;DATA INCORRECT FOLLOWING MTPJ
        ; OR SEQUENCE ERROR
  
```

4030
 4031
 4032
 4033
 4034
 4035
 4036
 4037
 4038
 4039
 4040
 4041
 4042
 4043
 4044
 4045
 4046
 4047
 4048
 4049
 4050
 4051
 4052
 4053
 4054

```

*****
: THIS TEST VERIFIES THE CONTENTS OF THE BRANCH ROM. THE TEST
: EXECUTES EVERY POSSIBLE BRANCH WITH EVERY POSSIBLE CONDITION
: CODE COMBINATION.
: THE ROUTINE USES TWO TABLES. THE BRANCH TABLE HOLDS ALL THE
: POSSIBLE BRANCH INSTRUCTIONS, THE OTHER TABLE (YNTAB) HOLDS BIT MAPS FOR
: EACH BRANCH. A ONE IN THE BIT MAP INDICATES THAT THE CORRESPONDING
: BRANCH INSTRUCTION SHOULD BRANCH FOR THE CONDITION CODE SETTING WHICH
: CORRESPONDS TO THE BIT POSITION WITHIN THE MAP. FOR EXAMPLE IF THE LEFT
: MOST BIT IS A ONE THEN THE CORRESPONDING BRANCH INSTRUCTION SHOULD BRANCH
: WHEN THE CONDITION CODES ARE 0.
: THE ROUTINE CONSISTS OF NESTED LOOPS; THE OUTER LOOP SETS UP
: ALL THE POSSIBLE BRANCH INSTRUCTIONS. THE INNER LOOP SETS UP EVERY POSSIBLE
: CONDITION CODE FOR EACH BRANCH.
: THE BIT MAP IS USED TO SET THE ADDRESS LOCATION IN TWO
: JUMP MODE 3 INSTRUCTIONS. THE ADDRESSES ARE CHANGED TO ALLOW THE
: PROGRAM TO CONTINUE OR JUMP TO AN ERROR ROUTINE DEPENDING UPON
: WHETHER IT HANDLED THE BRANCH INSTRUCTION CORRECTLY.
: AT ANY ERROR HALT, LOCATION, BRH, HOLDS THE BRANCH INSTRUCTION
: UNDER TEST AND LOCATION, CC, HOLDS THE VALUE OF THE CONDITION CODES
: AT THE TIME THE BRANCH WAS EXECUTED.
:
  
```

4055

.SBTTL TEST # 241 - TEST THE BRANCH ROM

 :TEST 241 - TEST THE BRANCH ROM

026514 005212
 026516 022712 000241
 026522 001062
 4056 026524 012700 042154
 4057 026530 012704 042230
 4058 026534 012767 000017 000146
 4059 026542 012067 000110
 4060 026546 012401
 4061 026550 012767 177777 000074
 4062 026556 012703 000020
 4063 026562 005267 000064
 4064 026566 032701 100000
 4065 026572 013705 177776
 026576 042705 177773
 026602 000165 026606
 026606 000167 000020
 4066 026612 012767 026712 000042
 4067 026620 012767 026670 000040
 4068 026626 000167 000014
 4069 026632 012767 026670 000022
 4070 026640 012767 026712 000020
 4071 026646 006101
 4072
 4073 026650 012737
 4074 026652 000000
 4075 026654 177776
 4076 026656 000000
 4077 026660 000137
 4078 026662 000000
 4079 026664 000137
 4080 026666 000000
 4081 026670 012702 000304
 4082 026674 012762 000555 177776
 026702 005262 177774
 026706 000000
 4083 026710 000000
 4084 026712 005303
 4085 026714 013705 177776
 026720 042705 177773
 026724 000165 026730
 026730 000167 177626
 4086 026734 005367 177750
 4087 026740 013705 177776
 026744 042705 177773
 026750 000165 026754
 026754 000167 177562

```

TST241: INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #241,(R2)     ;SEQUENCE ERROR?
        BNE     ER            ;BR TO ERROR HALT ON SEQ ERROR
SETUP:  MOV     #BRTAB,R0      ;INITIALIZE BRANCH TABLE POINTER
        MOV     #YNTAB,R4     ;INITIALIZE YES/NO BRANCH MAP POINTER
        MOV     #15.,BRCT     ;INITIALIZE BRANCH TABLE COUNT
SETBR:  MOV     (R0)+,BRH     ;GET NEXT BRANCH INST.
        MOV     (R4)+,R1      ;GET NEXT BRANCH MAP
        MOV     #-1,CC        ;INITIALIZE CONDITION CODE VALUE
        MOV     #16.,R3       ;INITIALIZE CONDITION CODE COUNT
SETCC:  INC     CC            ;SET FOR NEXT CC VALUE
        BIT     #100000,R1     ;SEE IF SHOULD BR W/ THESE CC'S
        MOV     @#177776,R5   ;SIMULATE A JNE
        BIC     #177773,R5    ;
        JMP     .+4(R5)       ; (JUMP NOT EQUAL)
        JMP     SET2BR       ; TO SET2BR
        MOV     #CONT,NBR     ;SET TO CONTINUE IF NO BRANCH
        MOV     #ER,YBR       ;SET TO REPORT ERROR IF BRANCH
        JMP     AROUND        ;GO AROUND OPPOSITE CONDITION
SET2BR: MOV     #ER,NBR       ;SET TO REPORT ERROR IF NO BRANCH
        MOV     #CONT,YBR     ;SET TO CONTINUE IF BRANCH
AROUND: ROL     R1           ;UPDATE BIT MAP
        MOV     (PC)+,@(PC)+  ;SET CONDITION CODE
        CC:     0             ;NEW CC VALUE GOES HERE
        BRH:    0             ;BRANCH INST. GOES HERE
        JMP     @ (PC)+       ;THIS JUMP IF NO BRANCH
        NBR:    0             ;WHERE TO GO IF NO BRANCH OCCURS
        JMP     @ (PC)+       ;THIS JUMP IF BRANCH OCCURS
        YBR:    0             ;WHERE TO GO IF BRANCH OCCURS
ER:     MOV     #TESTN,R2     ;RESTORE POINTER
        MOV     #555,-2(R2)   ;MOVE TO MAILBOX # ***** 555 *****
        INC     -4(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT
BRCT:   0
CONT:   DEC     R3            ;CC'S DONE?
        MOV     @#177776,R5   ;SIMULATE A JNE
        BIC     #177773,R5    ;
        JMP     .+4(R5)       ; (JUMP NOT EQUAL)
        JMP     SETCC        ; TO SETCC
        DEC     BRCT         ;BR'S DONE?
        MOV     @#177776,R5   ;SIMULATE A JNE
        BIC     #177773,R5    ;
        JMP     .+4(R5)       ; (JUMP NOT EQUAL)
        JMP     SETBR        ; TO SETBR
  
```

4088
 4089
 4090
 4091
 4092
 4093
 4094

 :THE FOLLOWING TEST VERIFIES THAT NO DUAL ADDRESSING OF THE GENERAL
 :REGISTERS OCCURS. ALL REGISTERS ARE CLEARED, AND A UNIQUE BIT IS SET
 :IN EACH. CMP INSTRUCTIONS CHECK THAT ONLY ONE BIT IS SET IN EACH
 :REGISTER.

4095

:

4096

```
.SBTTL TEST # 242 - DUAL REGISTER ADDRESSING TEST
:*****
:TEST 242 - DUAL REGISTER ADDRESSING TEST
:*****
```

```
026760 005212
026762 022712 000242
026766 001052
4097 026770 005000
4098 026772 005001
4099 026774 005002
4100 026776 005003
4101 027000 005004
4102 027002 005005
4103 027004 005006
4104 027006 052700 000001
4105 027012 052701 000002
4106 027016 052702 000004
4107 027022 052703 000010
4108 027026 052704 000020
4109 027032 052705 000040
4110 027036 052706 000100
4111 027042 022706 000100
4112 027046 001022
4113 027050 022705 000040
4114 027054 001017
4115 027056 022704 000020
4116 027062 001014
4117 027064 022703 000010
4118 027070 001011
4119 027072 022702 000004
4120 027076 001006
4121 027100 022701 000002
4122 027104 001003
4123 027106 022700 000001
4124 027112 001406
```

```
TST242: INC (R2) ;UPDATE TEST NUMBER
          CMP #242,(R2) ;SEQUENCE ERROR?
          BNE DAERR ;BR TO ERROR HALT ON SEQ ERROR
BITCLR: CLR R0 ;INITIALIZE ALL REGISTERS
         CLR R1
         CLR R2
         CLR R3
         CLR R4
         CLR R5
         CLR R6
BITSET: BIS #1,R0 ;SET R0=1
        BIS #2,R1 ;R1=2
        BIS #4,R2 ;R2=4
        BIS #10,R3 ;R3=10
        BIS #20,R4 ;R4=20
        BIS #40,R5 ;R5=40
        BIS #100,R6 ;R6=100
BITCHK: CMP #100,R6 ;TEST THAT NO DUAL ADDRESSING OCCURRED
        BNE DAERR ;BR TO ERROR HALT IF ANY OTHER BITS ARE SET
        CMP #40,R5
        BNE DAERR
        CMP #20,R4
        BNE DAERR
        CMP #10,R3
        BNE DAERR
        CMP #4,R2
        BNE DAERR
        CMP #2,R1
        BNE DAERR
        CMP #1,R0
        BEQ BITCON
```

```
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 725 <
```

```
027114
027114 012762 000556 177776
027122 005262 177774
027126 000000
4125 027130 012702 000304
```

```
DAERR: MOV #556,-2(R2) ;MOVE TO MAILBOX # ***** 556 *****
        INC -4(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;DUAL ADDRESSING ERROR
BITCON: MOV #STESTN,R2 ;RESTORE POINTER
```

```
4126
4127
4128
4129
4130
4131
4132
```

```
:*****
: THIS TEST VERIFIES THAT THE UPPER BYTE OF THE PSW IS NOT AFFECTED
: WHEN THE PRIORITY LEVEL OR CC'S ARE CHANGED. ALL BITS ARE
: INITIALLY SET IN THE PSW, AND THE LOW BYTE IS CLEARED. A BIT
: INSTRUCTION VERIFIES THE DATA.
:*****
```

4133

.SBTTL TEST # 243 - TEST BYTE INSTRUCTION ON PSW

 :TEST 243 - TEST BYTE INSTRUCTION ON PSW

027134 005212
 027136 022712 000243
 027142 001012
 4134 027144 052737 170357 177776
 4135 027152 105037 177776
 4136 027156 013700 177776
 4137 027162 032700 170000
 4138 027166 001010
 4139 027170 005037 177776
 4140 027174 012762 000557 177776
 027202 005262 177774
 027206 000000
 4141 027210 005037 177776

TST243: INC (R2) ;UPDATE TEST NUMBER
 (MP #243,(R2) ;SEQUENCE ERROR?
 BNE BTERR ;BR TO ERROR HALT ON SEQ ERROR
 BIS #170357,@#PS ;SET ALL POSSIBLE BITS IN PSW
 CLRB @#PS ;CLR PR LEVEL AND CC'S
 MOV @#PS,R0 ;COPY CONTENTS OF PSW
 BIT #170000,R0 ;TEST THAT UPPER BYTE IS UNAFFECTED
 BNE BTCON ;CONTINUE IF OK
 BTERR: CLR @#PS ;RETURN TO KERNEL MODE
 MOV #557,-2(R2) ;MOVE TO MAILBOX # ***** 557 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;BYTE INSTRUCTION ALTERED PSW
 BTCON: CLR @#PS ;RETURN TO KERNEL MODE

4142
 4143
 4144
 4145
 4146
 4147
 4148

 : THIS TEST VERIFIES THAT A JMP INSTRUCTION DOES NOT ALTER THE
 : CONDITION CODES IN THE PSW. THE CC'S ARE PRESET, THE JMP IS
 : EXECUTED, AND CONDITIONAL BRANCHES VERIFY THE STATE OF THE CC'S.
 :

4149

.SBTTL TEST # 244 - TEST THAT JMP OPCODE DOES NOT AFFECT C.C.'S
 :*****
 :TEST 244 - TEST THAT JMP OPCODE DOES NOT AFFECT C.C.'S
 :*****

027214 005212
 027216 022712 000244
 027222 001012
 4150 027224 000277
 4151 027226 000252
 4152 027230 000167 000000
 4153 027234 100403
 4154 027236 001002
 4155 027240 102401
 4156 027242 103406

TST244: INC (R2) ;UPDATE TEST NUMBER
 CMP #244,(R2) ;SEQUENCE ERROR?
 BNE TST245-10 ;BR TO ERROR HALT ON SEQ ERROR
 SCC
 +CLN!CLV ;CC=0101
 JMP JMPT ;JUMP TO TEST PSW
 JMPT: BMI JMPERR ;BR TO ERROR HALT IF N-BIT IS SET
 BNE JMPERR ;BR TO ERROR HALT IF Z-BIT IS CLEAR
 BVS JMPERR ;BR TO ERROR HALT IF V-BIT IS SET
 BCS TST245

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
 ; CONDITIONAL BRANCH INST. AND <=
 ; REPLACE THE MOVE INSTRUCTION <-
 ; WHICH FOLLOWS W/ 767 <--

027244
 027244 012762 000560 177776
 027252 005262 177774
 027256 000000

JMPERR: MOV #560,-2(R2) ;MOVE TO MAILBOX # ***** 560 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;JMP INSTRUCTION AFFECTED CC'S
 ; OR SEQUENCE ERROR

4157
 4158
 4159
 4160
 4161
 4162
 4163
 4164
 4165
 4166
 4167
 4168
 4169
 4170

:*****
 : THIS TEST VERIFIES THE SET AND CLEAR CONDITION CODE INSTRUCTIONS.
 : THE TEST CONSISTS OF TWO ROUTINES, ONE TO TEST ALL CLEAR CC
 : INSTRUCTIONS, AND THE SECOND TO TEST ALL SET CC INSTRUCTIONS. ALL
 : POSSIBLE COMBINATIONS OF CONDITION CODES ARE TESTED, INCLUDING NCP'S.
 : TO TEST THE CLEAR CC INSTRUCTIONS, ALL CONDITION CODES ARE
 : INITIALLY SET. THE INSTRUCTION IS EXECUTED, AND THE PSW IS CHECKED
 : TO VERIFY THE PROPER COMBINATION OF CONDITION CODES.
 : TO TEST THE SET CC INSTRUCTIONS, THE CONDITION CODES ARE
 : INITIALLY CLEARED, AND ONLY THE REQUIRED BITS ARE SET BY THE SET CC
 : INSTRUCTION. THE CONTENTS OF THE PSW ARE CHECKED TO VERIFY THAT
 : ONLY THE REQUIRED BITS WERE SET.
 :*****

4171

.SBTTL TEST # 245 - TEST SET CC AND CLEAR CC INSTRUCTIONS

 :TEST 245 - TEST SET CC AND CLEAR CC INSTRUCTIONS

027260 005212
 027262 022712 000245
 027266 001064
 4172 027270 012767 000240 000024
 4173 027276 012767 000017 000032
 4174 027304 012767 000261 000106
 4175 027312 012767 000001 000114
 4176 027320 000277
 4177 027322 000000
 4178 027324 013704 177776
 4179 027330 042704 177760
 4180 027334 022704
 4181 027336 000000
 4182 027340 001406

TST245: INC (R2)
 CMP #245,(R2)
 BNE CCERR
 MOV #240,CC1
 MOV #17,CC2
 MOV #261,SC3
 MOV #1,SC4
 CLRCD: SCC
 CC1: 0
 MOV @#PS,R4
 BIC #177760,R4
 CMP (PC)+,R4
 CC2: 0
 BEQ CON1

:UPDATE TEST NUMBER
 :SEQUENCE ERROR?
 :BR TO ERROR HALT ON SEQ ERROR
 :INITIALIZE CLR CC INSTRUCTION CODES
 :INITIALIZE OCTAL MAP
 :INITIALIZE SET CC INSTRUCTION CODES
 :INITIALIZE OCTAL MAP
 :SET ALL CONDITION CODES
 :CONDITION CODE INSTRUCTION
 :COPY THE PSW
 :ISOLATE CONDITION CODES
 :CHECK THAT PROPER CC'S WERE CLEARED
 :OCTAL REPRESENTATION OF CC'S

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 752 <

027342 012762 000561 177776
 027350 005262 177774
 027354 000000
 4183 027356 005367 177754
 4184 027362 005267 177734
 4185 027366 026727 177730 000257
 4186 027374 003751
 4187 027376 026727 177720 000260
 4188 027404 001004
 4189 027406 012767 000017 177722
 4190 027414 000741
 4191 027416 000257
 4192 027420 000000
 4193 027422 013704 177776
 4194 027426 042704 177760
 4195 027432 022704
 4196 027434 000000
 4197 027436 001406

MOV #561,-2(R2)
 INC -4(R2)
 HALT
 CON1: DEC CC2
 INC CC1
 CMP CC1,#257
 BLE CLRCD
 CMP CC1,#260
 BNE SETCD
 MOV #17,CC2
 BR CLRCD
 SETCD: CCC
 SC3: 0
 MOV @#PS,R4
 BIC #177760,R4
 CMP (PC)+,R4
 SC4: 0
 BEQ CON2

:MOVE TO MAILBOX # ***** 561 *****
 :SET MSGTYP TO FATAL ERROR
 :CLEAR CC INSTRUCTION FAILED
 :SET NEXT OCTAL MAP OF CC'S
 :GET NEXT CLEAR CC INSTRUCTION
 :TEST FOR CCC INSTRUCTION
 :GO TEST NEXT INSTRUCTION IF NOT FOUND
 :CHECK FOR NOP=260
 :GO TEST SET CC INSTRUCTIONS
 :SET OCTAL MAP TO TEST NOP
 :GO TEST NOP
 :CLEAR ALL CONDITION CODES
 :CONDITION CODE INSTRUCTION
 :COY PSW
 :CLEAR AWAY UNWANTED BITS
 :CHECK THAT PROPER CC'S WERE SET
 :OCTAL REPRESENTATION OF CC'S

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 713 <

027440
 027440 012762 000562 177776
 027446 005262 177774
 027452 000000
 4198 027454 005267 177754
 4199 027460 005267 177734
 4200 027464 026727 177730 000277
 4201 027472 003751

CCERR: MOV #562,-2(R2)
 INC -4(R2)
 HALT
 CON2: INC SC4
 INC SC3
 CMP SC3,#277
 BLE SETCD

:MOVE TO MAILBOX # ***** 562 *****
 :SET MSGTYP TO FATAL ERROR
 :SET CC FAILED OR SEQUENCE ERROR
 :SET NEXT OCTAL MAP
 :PREPARE NEXT SET CC INSTRUCTION
 :FINISHED?
 :BR IF NO

4202
 4203
 4204
 4205
 4206

 : THESE NEXT TWO TEST VERIFY MFPD AND MTPD INSTRUCTIONS
 : WITH R6 IN MODE 0.

4207

:

4208

```

.SBTTL TEST # 246 - TEST MFPD WITH R6 IN MODE 0
:*****
:TEST 246 - TEST MFPD WITH R6 IN MODE 0
:*****
TST246: INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #246,(R2)     ;SEQUENCE ERROR?
        BNE      TST247-10     ;BR TO ERROR HALT ON SEQ ERROR
        MOV      #STBOT,R6     ;INITIALIZE KERNEL STACK POINTER
        MOV      #USRM,PS      ;SETUP USER MODE .PREVIOUS KERNEL
        MOV      #USTBOT,R6    ;INITIALIZE USER STACK POINTER
        MFPD     R6           ;TRY MFPD WITH MODE 0
        CMP      #140000,PS    ;CHECK PSW
        BEQ      MFPD0        ;BR IF NO ERROR
        BIC      #USRM,PS      ;CLEAR USER MODE
        MOV      #563,-2(R2)   ;MOVE TO MAILBOX # ***** 563 *****
        INC      -4(R2)       ;SET MSGTYP TO FATAL ERROR
        HALT                    ;INCORRECT PSW FROM MFPD
MFPD0:  CMP      #STBOT,USTBOT-2 ;CHECK DATA ON STACK
        BEQ      MFPDOA        ;BR IF NO ERROR
        BIC      #USRM,PS      ;CLEAR USER MODE
        MOV      #564,-2(R2)   ;MOVE TO MAILBOX # ***** 564 *****
        INC      -4(R2)       ;SET MSGTYP TO FATAL ERROR
        HALT                    ;INCORRECT DATA FROM MFPD
MFPDOA:
  
```

```

027474 005212
027476 022712 000246
027502 001036
4209 027504 012706 001000
4210 027510 012767 140000 150260
4211 027516 012706 042402
4212 027522 106506
4213 027524 022767 140000 150244
4214 027532 001411
4215 027534 042767 140000 150234
4216 027542 022762 000563 177776
    027550 005262 177774
    027554 000000
4217 027556 022767 001000 012614 MFPD0:
4218 027564 001411
4219 027566 042767 140000 150202
4220 027574 012762 000564 177776
    027602 005262 177774
    027606 000000
4221 027610
4222
  
```

4223

.SBTTL TEST # 247 - TEST MTPD WITH R6 IN MODE 0

:TEST 247 - TEST MTPD WITH R6 IN MODE 0

027610 005212
027612 022712 000247
027616 001035
4224 027620 005067 150152
4225 027624 005006
4226 027626 012767 140000 150142
4227 027634 012746 001000
4228 027640 106606
4229 027642 022767 140000 150126
4230 027650 001411
4231 027652 042767 140000 150116
4232 027660 012762 000565 177776
027666 005262 177774
027672 000000
4233 027674 005067 150076
4234 027700 020627 001000
4235 027704 001406

TST247: INC (R2) ;UPDATE TEST NUMBER
CMP #247,(R2) ;SEQUENCE ERROR?
BNE TST250-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR PS ;SET KERNEL MODE
CLR R6 ;INITIALIZE KERNEL R6
MOV #USRM,PS ;SET USER MODE/PREVIOUS KERNEL
MOV #STBOT,-(R6) ;SET UP TARGET DATA
MTPD R6 ;TRY MODE 0 MTPD
CMP #USRM,PS ;CHECK PSW
BEQ MTPD0 ;BR IF NO ERROR
BIC #USRM,PS ;CLEAR USER MODE
MOV #565,-2(R2) ;MOVE TO MAILBOX # ***** 565 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS INCORRECT FOLLOWING MTPD
MTPD0: CLR PS ;SET KERNEL MODE
CMP R6,#STBOT ;CHECK TARGET DATA
BEQ TST250

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <----
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 744 <====

027706 012762 000566 177776
027714 005262 177774
027720 000000

MOV #566,-2(R2) ;MOVE TO MAILBOX # ***** 566 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DATA INCORRECT FOLLOWING MTPD
; OR SEQUENCE ERROR

4236
4237

4238

.SBTTL TEST # 250 - TEST MFPT INSTRUCTION

:TEST 250 - TEST MFPT INSTRUCTION

027722 005212
027724 022712 000250
027730 001007
4239 027732 005000
4240 027734 000007
4241 027736 122700 000001
4242 027742 001406

TST250: INC (R2) ;UPDATE TEST NUMBER
CMP #250,(R2) ;SEQUENCE ERROR?
BNE TST251-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;SET UP R0
MFPT ;MOVE FROM PROCSSOR TYPE
CMPB #1,R0 ;CHECK FOR A ONE IN R0
BEQ TST251

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 772 <
: MOVE TO MAILBOX # ***** 567 *****
: SET MSGTYP TO FATAL ERROR
: MFPT FAILED TO RETURN A 1, IN R0
: OR SEQUENCE ERROR

027744 012762 000567 177776
027752 005262 177774
027756 000000

MOV #567,-2(R2)
INC -4(R2)
HALT

4243
4244
4245
4246
4247
4248
4249
4250

:
: .SBTTL BIT TEST OF PIRQ REGISTER
: A COUNT PATTERN IS THEN RUN THRU THE REGISTER TO ENSURE THAT
: THE ENCODER FUNCTIONS PROPERLY.
:

4251

.SBTTL TEST # 251 - BIT TEST OF PIRQ REGISTER

 :TEST 251 - BIT TEST OF PIRQ REGISTER

027760 005212
 027762 022712 000251
 027766 001066
 4252 027770 012706 001000
 4253 027774 012767 030010 012222
 4254 030002 012767 030010 012216
 4255 030010 052737 000340 177776 4\$:
 4256 030016 005067 012170
 4257 030022 005037 177772
 4258 030026 026737 012160 177772
 4259 030034 001035
 4260 030036 012700 000177
 4261 030042 012767 001042 012142
 4262 030050 012701 000002
 4263 030054 062737 001000 177772 2\$:
 4264 030062 026737 012124 177772
 4265 030070 001017
 4266 030072 120137 177773
 4267 030076 001005
 4268 030100 062767 000042 012104
 4269 030106 005201
 4270 030110 006101
 4271 030112 062767 001000 012072 3\$:
 4272 030120 077023
 4273 030122 005037 177772
 4274 030126 000412
 4275 030130 013767 177772 012064 1\$:
 4276 030136 001406

TST251: INC (R2) ;UPDATE TEST NUMBER
 CMP #251,(R2) ;SEQUENCE ERROR?
 BNE TST252-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #STB01,SP ;INITIALIZE THE SP
 MOV #4\$, \$LPADR ;SETUP LOOP ADR
 MOV #4\$, \$LPERR ;SETUP ERROR LOOP
 4\$: BIS #340,@#177776 ;SET THE CPU PRIORITY AT 7.
 CLR \$TMP0 ;SETUP COMPARISON LOCATION
 CLR @#PIRQ ;CLEAR PIRQ REGISTER
 CMP \$TMP0,@#PIRQ ;DID PIRQ CLEAR
 BNE 1\$;BRANCH IF NO
 MOV #177,R0 ;SETUP ITERATION COUNT
 MOV #1042,\$TMP0 ;SETUP COMPARISON LOCATION
 MOV #2,R1 ;SETUP R1
 2\$: ADD #1000,@#PIRQ ;START COUNT PATTERN
 CMP \$TMP0,@#PIRQ ;DID REGISTER SET CORRECT?
 BNE 1\$;BRANCH IF NO
 CMPB R1,@#PIRQ+1 ;IS PIRQ READY TO GO TO NEXT LEVEL?
 BNE 3\$;BRANCH IF NO
 ADD #42,\$TMP0 ;INCREMENT ENCODED VALUE IN TEST LOC.
 INC R1 ;SETUP R1 FOR ROTATE
 ROL R1 ;SET R1 TO NEXT CHECK LEVEL
 3\$: ADD #1000,\$TMP0 ;INC. PIRQ LEVEL IN TEST LOCATION
 SOB R0,2\$;CONTINUE COUNT
 CLR @#PIRQ ;ENSURE PIRQ CLEAR
 BR EIS ;GO TO NEXT TEST
 1\$: MOV @#PIRQ,\$EPIRQ ;SAVE PIRQ FOR ERROR CHECKING
 BEQ TST252

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 713 <
 : MOVE TO MAILBOX # ***** 570 *****
 : SET MSGTYP TO FATAL ERROR
 : PIRQ REG. FAILED
 : OR SEQUENCE ERROR

030140 012762 000570 177776
 030146 005262 177774
 030152 000000

MOV #570,-2(R2)
 INC -4(R2)
 HALT

4277 030154
 4278
 4279
 4280
 4281

EIS:

.SBTTL EIS ASH/ASCH/MUL/DIV TESTS
 ;ASH SHIFTING RIGHT USING DM2 REG 7
 ;SHIFT RIGHT CLEAR N-BIT AND C-BIT
 ;MODE 2-REG 7

4282

.SBTTL TEST # 252 - ASH 40000 SHIFTED BY 177765=10 PS=0
:*****
:TEST 252 - ASH 40000 SHIFTED BY 177765=10 PS=0
:*****

030154 005212
030156 022712 000252
030162 001032
4283 030164 012706 001000
4284 030170 005037 177776
4285 030174 012700 040000
4286 030200 072027 177765
4287 030204 013767 177776 012330
4288 030212 122767 000000 012322
4289 030220 001406

TST252: INC (R2) ;UPDATE TEST NUMBER
CMP #252,(R2) ;SEQUENCE ERROR?
BNE TST253-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #STBOT,R6
CLR @#PS
MOV #40000,R0 ;LOAD R0 WITH 40000
ASH #177765,R0 ;SHIFT R0 BY 177765
MOV @#PS,SPSW ;SAVE PS
CMPB #0,SPSW ;IS THE PS 0?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 760 <

030222 012762 000571 177776
030230 005262 177774
030234 000000
4290 030236 022700 000010
4291 030242 001406

1\$: MOV #571,-2(R2) ;MOVE TO MAILBOX # ***** 571 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;THE PS IS NOT EQUAL TO 0
CMP #10,R0 ;IS THE RESULT 10?
BEQ TST253

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 747 <

030244 012762 000572 177776
030252 005262 177774
030256 000000

MOV #572,-2(R2) ;MOVE TO MAILBOX # ***** 572 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 IS NOT EQUAL TO 10
; OR SEQUENCE ERROR

4292
4293
4294

;SHIFT RIGHT SET N-BIT CLEAR C-BIT
;MODE 2 /REG 7

4295

.SBTTL TEST # 253 - ASH 125252 SHIFTED BY -2=165252 PS 11
:*****
:TEST 253 - ASH 125252 SHIFTED BY -2=165252 PS-11
:*****

030260 005212
030262 022712 000253
030266 001026
4296 030270 012700 125252
4297 030274 072027 177776
4298 030300 013767 177776 012234
4299 030306 122767 000011 012226
4300 030314 001406

TST253: INC (R2) ;UPDATE TEST NUMBER
CMP #253,(R2) ;SEQUENCE ERROR?
BNE TST254-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125252,R0 ;LOAD R0 WITH 125252
ASH #-2,R0 ;SHIFT R0 BY -2
MOV @#PS,SPSW ;SAVE PS
CMPB #11,SPSW ;IS THE PS 11?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 764 <

030316 012762 000573 177776
030324 005262 177774
030330 000000
4301 030332 022700 165252
4302 030336 001406

1\$: MOV #573,-2(R2) ;MOVE TO MAILBOX # ***** 573 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;THE PS IS NOT EQUAL TO 11
CMP #165252,R0 ;IS THE RESULT 165252?
BEQ TST254

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 753 <

030340 012762 000574 177776
030346 005262 177774
030352 000000

MOV #574,-2(R2) ;MOVE TO MAILBOX # ***** 574 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 IS NOT EQUAL TO 165252
; OR SEQUENCE ERROR

4303
4304
4305

:SHIFT RIGHT CLEAR N-BIT SET Z-BIT
:MODE 2 /REG 7

4306

.SBTTL TEST # 254 - ASH 0 SHIFTED BY -16. =0 PS-4

 :TEST 254 - ASH 0 SHIFTED BY -16. =0 PS=4

030354 005212
 030356 022712 000254
 030362 001026
 4307 030364 012700 000000
 4308 030370 072027 177760
 4309 030374 013767 177776 012140
 4310 030402 122767 000004 012132
 4311 030410 001406

TST254: INC (R2) ;UPDATE TEST NUMBER
 CMP #254,(R2) ;SEQUENCE ERROR?
 BNE TST255-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD R0 WITH 0
 ASH #-16.,R0 ;SHIFT R0 BY -16.
 MOV @#PS,SPSW ;SAVE PS
 CMPB #4,SPSW ;IS THE PS 4?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 764 <====

030412 012762 000575 177776
 030420 005262 177774
 030424 000000
 4312 030426 022700 000000
 4313 030432 001406

MOV #575,-2(R2) ;MOVE TO MAILBOX # ***** 575 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 4
 1\$: CMP #0,R0 ;IS THE RESULT 0?
 BEQ TST255

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 753 <====

030434 012762 000576 177776
 030442 005262 177774
 030446 000000

MOV #576,-2(R2) ;MOVE TO MAILBOX # ***** 576 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 0
 ; OR SEQUENCE ERROR

4314
 4315
 4316
 4317

.SBTTL ASH SHIFTING LEFT STORE ASH DM2 REG 7
 ;SHIFT LEFT SET C-BIT=0 STORE ASH
 ;MODE 2 /REG 7

4318

.SBTTL TEST # 255 - ASH 0 SHIFTED BY 0=0 PS 4
 :*****
 :TEST 255 - ASH 0 SHIFTED BY 0=0 PS=4
 :*****

030450 005212
 030452 022712 000255
 030456 001026
 4319 030460 012700 000000
 4320 030464 072027 000000
 4321 030470 013767 177776 012044
 4322 030476 122767 000004 012036
 4323 030504 001406

TST255: INC (R2) ;UPDATE TEST NUMBER
 CMP #255,(R2) ;SEQUENCE ERROR?
 BNE TST256-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD R0 WITH 0
 ASH #0,R0 ;SHIFT R0 BY 0
 MOV @#PS,SPSW ;SAVE PS
 CMPB #4,SPSW ;IS THE PS 4?
 BEQ 1\$

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 764 <

030506 012762 000577 177776
 030514 005262 177774
 030520 000000
 4324 030522 022700 000000
 4325 030526 001406

1\$: MOV #577,-2(R2) ;MOVE TO MAILBOX # ***** 577 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 4
 CMP #0,R0 ;IS THE RESULT 0?
 BEQ TST256

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 753 <

030530 012762 000600 177776
 030536 005262 177774
 030542 000000

MOV #600,-2(R2) ;MOVE TO MAILBOX # ***** 600 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 0
 ; OR SEQUENCE ERROR

4326
 4327
 4328
 4329
 4330

.SBTTL ASH SHIFTING LEFT DM6 REG 7
 ;SHIFT LEFT SET C-BIT=1
 ;MODE 6 /REG 7

4331

.SBTTL TEST # 256 - ASH 125252 SHIFTED BY S1=125250 PS=12

 :TEST 256 - ASH 125252 SHIFTED BY S1=125250 PS=12

030544 005212
 030546 022712 000256
 030552 001030
 4332 030554 012700 125252
 4333 030560 012704 042550
 4334 030564 072067 011760
 4335 030570 013767 177776 011744
 4336 030576 122767 000012 011736
 4337 030604 001406

TST256: INC (R2) ;UPDATE TEST NUMBER
 CMP #256 (R2) ;SEQUENCE ERROR?
 BNE TST257-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125252,R0 ;LOAD R0 WITH 125252
 MOV #S1,R4 ;SET UP R4
 ASH S1,R0 ;SHIFT R0 BY S1
 MOV @#PS,SPSW ;SAVE PS
 CMPB #12,SPSW ;IS THE PS 12?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <-
 : WHICH FOLLOWS W/ 762 <

030606 012762 000601 177776
 030614 005262 177774
 030620 000000
 4338 030622 022700 125250 1\$:
 4339 030626 001406

MOV #601,-2(R2) ;MOVE TO MAILBOX # ***** 601 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 12
 CMP #125250,R0 ;IS THE RESULT 125250?
 BEQ TST257

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <-
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 751 <

030630 012762 000602 177776
 030636 005262 177774
 030642 000000

MOV #602,-2(R2) ;MOVE TO MAILBOX # ***** 602 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 177525
 ; OR SEQUENCE ERROR

4340
 4341
 4342
 4343

.SBTTL ASH SHIFTING LEFT TEST SIGN Z-BIT DM7 REG 7
 ;SHIFT LEFT COUNT DOWN TO ZERO TEST SIGN Z-BIT
 ;MODE 7 /REG 7

4344

.SBTTL TEST # 257 - ASH 125252 SHIFTED BY @S2= 177525 PS=10

:TEST 257 - ASH 125252 SHIFTED BY @S2= 177525 PS=10

030644 005212
030646 022712 000257
030652 001030
4345 030654 012700 125252
4346 030660 012703 042556
4347 030664 072077 011666
4348 030670 013767 177776 011644
4349 030676 122767 000010 011636
4350 030704 001406

TST257: INC (R2) ;UPDATE TEST NUMBER
CMP #257,(R2) ;SEQUENCE ERROR?
BNE TST260-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125252,R0 ;LOAD R0 WITH 125252
MOV #S4,R3 ;SET UP R3
ASH @S4,R0 ;SHIFT R0 BY @S2
MOV @WPS,SPSW ;SAVE PS
CMPB #10,SPSW ;IS THE PS 10?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 762 <

030706 012762 000603 177776
030714 005262 177774
030720 000000
4351 030722 022700 177525 1\$:
4352 030726 001406

MOV #603,-2(R2) ;MOVE TO MAILBOX # ***** 603 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;THE PS IS NOT EQUAL TO 10
CMP #177525,R0 ;IS RESULT 177525?
BEQ TST260

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
: CONDITIONAL BRANCH INST. AND <-
: REPLACE THE MOVE INSTRUCTION <-
: WHICH FOLLOWS W/ 751 <--

030730 012762 000604 177776
030736 005262 177774
030742 000000

MOV #604,-2(R2) ;MOVE TO MAILBOX # ***** 604 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 IS NOT EQUAL TO 177525
; OR SEQUENCE ERROR

4353
4354
4355
4356

.SBTTL ASH SHIFTING LEFT TESTING IR9 FOR ASH/ASHC DM3 REG 7
:SHIFT LEFT TEST IR9 TO DETERMINE ASH/ASHC
:MODE 3 /REG 7

4357

.SBTTL TEST # 260 - ASH 125252 SHIFTED BY @#S1=177525 PS=10

 :TEST 260 - ASH 125252 SHIFTED BY @#S1=177525 PS=10

030744 005212
 030746 022712 000260
 030752 001030
 4358 030754 012700 125252
 4359 030760 012704 042554
 4360 030764 072037 042554
 4361 030770 013767 177776 011544
 4362 030776 122767 000010 011536
 4363 031004 001406

TST260: INC (R2) ;UPDATE TEST NUMBER
 CMP #260,(R2) ;SEQUENCE ERROR?
 BNE TST261-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125252,R0 ;LOAD R0 WITH 125252
 MOV #S3,R4 ;SET UP R4
 ASH @#S3,R0 ;SHIFT R0 BY @#S1
 MOV @#PS,SPSW ;SAVE PS
 CMPB #10,SPSW ;IS THE PS 10?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 762 <

031006 012762 000605 177776
 031014 005262 177774
 031020 000000
 4364 031022 022700 177525
 4365 031026 001406

1\$: MOV #605,-2(R2) ;MOVE TO MAILBOX # ***** 605 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 10
 CMP #177525,R0 ;IS THE RESULT 177525?
 BEQ TST261

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 751 <

031030 012762 000606 177776
 031036 005262 177774
 031042 000000

MOV #606,-2(R2) ;MOVE TO MAILBOX # ***** 606 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 177525
 ; OR SEQUENCE ERROR

4366
 4367
 4368
 4369
 4370
 4371

.SBTTL ASH SHIFTING LEFT USING DM6 REG 7
 ;SHIFT LEFT TEST R17 TO DETERMINE C-BIT
 ;CLEAR BX TO INDICATE C-BIT=0
 ;MODE 6 /REG 7

4372

.SBTTL TEST # 261 - ASH 025252 SHIFTED S1=125250 PS 12
:*****
:TEST 261 - ASH 025252 SHIFTED S1=125250 PS=12
:*****

031044 005212
031046 022712 000261
031052 001030
4373 031054 012700 025252
4374 031060 012704 042550
4375 031064 072067 011460
4376 031070 013767 177776 011444
4377 031076 122767 000012 011436
4378 031104 001406

TST261: INC (R2) ;UPDATE TEST NUMBER
CMP #261,(R2) ;SEQUENCE ERROR?
BNE TST262-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #025252,R0 ;LOAD R0 WITH 025252
MOV #S1,R4 ;SET UP R4
ASH S1,R0 ;SHIFT R0 BY S1
MOV @RPS,SPSW ;SAVE PS
CMPB #12,SPSW ;IS THE PS 10?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 762 <

031106 012762 000607 177776
031114 005262 177774
031120 000000
4379 031122 022700 125250
4380 031126 001406

1\$: MOV #607,-2(R2) ;MOVE TO MAILBOX # ***** 607 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;THE PS IS NOT EQUAL TO 10
CMP #125250,R0 ;IS THE RESULT 125250
BEQ TST262

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 751 <====

031130 012762 000610 177776
031136 005262 177774
031142 000000

MOV #610,-2(R2) ;MOVE TO MAILBOX # ***** 610 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 IS NOT EQUAL TO 077525
; OR SEQUENCE ERROR

4381
4382
4383
4384
4385

.SBTTL ASHC SHIFTING POS.USING DMO REG 2
;MODE 0 REG 4

4386

.SBTTL TEST # 262 - ASHC 125252,125252,SHIF BY R4=177525 52525 PS=10
:*****
:TEST 262 - ASHC 125252,125252,SHIF BY R4=177525 52525 PS=10
:*****

031144 005212
031146 022712 000262
031152 001043
4387 031154 012700 125252
4388 031160 012701 125252
4389 031164 000241
4390 031166 012704 177771
4391 031172 073004
4392 031174 013767 177776 011340
4393 031202 122767 000010 011332
4394 031210 001406

TST262: INC (R2) ;UPDATE TEST NUMBER
CMP #262,(R2) ;SEQUENCE ERROR?
BNE TST263-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125252,R0 ;LOAD R0 WITH 125252
MOV #125252,R1 ;LOAD R0!1 WITH 125252
CLC
MOV #-7,R4 ;SET UP R4
ASHC R4,R0 ;SHIFT R0,R0!1 BY R4
MOV @#PS,SPSW ;SAVE PS
CMPB #10,SPSW ;IS THE PS 10?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 760 <====

031212 012762 000611 177776
031220 005262 177774
031224 000000
4395 031226 022700 177525 1\$:
4396 031232 001406

MOV #611,-2(R2) ;MOVE TO MAILBOX # ***** 611 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;THE PS IS NOT EQUAL TO 10
CMP #177525,R0 ;IS RESULT 177525?
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 747 <====

031234 012762 000612 177776
031242 005262 177774
031246 000000
4397 031250 022701 052525 2\$:
4398 031254 001406

MOV #612,-2(R2) ;MOVE TO MAILBOX # ***** 612 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 IS NOT EQUAL TO 177525
CMP #52525,R1 ;IS THE RESULT 52525?
BEQ TST263

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 736 <====

031256 012762 000613 177776
031264 005262 177774
031270 000000

MOV #613,-2(R2) ;MOVE TO MAILBOX # ***** 613 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R1 IS NOT EQUAL TO 52525
; OR SEQUENCE ERROR

4399
4400
4401

.SBTTL ASHC SHIFTING NEG. USING DM4 REG 3

:MODE 4 REG 3

4402

.SBTTL TEST # 263 - ASHC 125252,125252,SHIF,-(3)-177525,52525 PS=10
:*****
:TEST 263 - ASHC 125252,125252,SHIF,-(3)=177525,52525 PS=10
:*****

031272 005212
031274 022712 000263
031300 001043
4403 031302 012700 125252
4404 031306 012701 125252
4405 031312 000241
4406 031314 012703 042556
4407 031320 073043
4408 031322 013767 177776 011212
4409 031330 122767 000010 011204
4410 031336 001406

TST263: INC (R2) ;UPDATE TEST NUMBER
CMP #263,(R2) ;SEQUENCE ERROR?
BNE TST264-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125252,R0 ;LOAD R0 WITH 125252
MOV #125252,R1 ;LOAD R0!1 WITH 125252
CLC
MOV #S3+2,R3 ;SET UP R3
ASHC -(R3),R0 ;SHIFT R0,R0.1 BY -(3)
MOV @#PS,SPSW ;SAVE PS
CMPB #10,SPSW ;IS THE PS 10?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 760 <

031340 012762 000614 177776
031346 005262 177774
031352 000000
4411 031354 022700 177525 1\$:
4412 031360 001406

MOV #614,-2(R2)
INC -4(R2)
HALT
CMP #177525,R0
BEQ 2\$

: MOVE TO MAILBOX # ***** 614 *****
: SET MSGTYP TO FATAL ERROR
: THE PS IS NOT EQUAL TO 10
: IS THE RESULT 177525

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 747 <

031362 012762 000615 177776
031370 005262 177774
031374 000000
4413 031376 022701 052525 2\$:
4414 031402 001406

MOV #615,-2(R2)
INC -4(R2)
HALT
CMP #52525,R1
BEQ TST264

: MOVE TO MAILBOX # ***** 615 *****
: SET MSGTYP TO FATAL ERROR
: R0 IS NOT EQUAL TO 177252
: IS THE RESULT 52525?

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 736 <

031404 012762 000616 177776
031412 005262 177774
031416 000000

MOV #616,-2(R2)
INC -4(R2)
HALT

: MOVE TO MAILBOX # ***** 616 *****
: SET MSGTYP TO FATAL ERROR
: R0!1 IS NOT EQUAL TO 52525
: OR SEQUENCE ERROR

4415
4416
4417

.SBTTL ASHC SHIFTED BY @-(4) DMS REG 4
:MODE 5 REG 4

4418

.SBTTL TEST # 264 - ASHC 125252 SHIFTED BY @-(4)=177525 52525 PS-10
:*****
:TEST 264 - ASHC 125252 SHIFTED BY @-(4)=177525 52525 PS=10
:*****

031420 005212
031422 022712 000264
031426 001043
4419 031430 012700 125252
4420 031434 012701 125252
4421 031440 000241
4422 031442 012704 042560
4423 031446 073054
4424 031450 013767 177776 011064
4425 031456 122767 000010 011056
4426 031464 001406

TST264: INC (R2) ;UPDATE TEST NUMBER
CMP #264,(R2) ;SEQUENCE ERROR?
BNE TST265-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125252,R0 ;LOAD R0 WITH 125252
MOV #125252,R1 ;LOAD R0:1 WITH 125252
CLC
MOV #54+2,R4 ;SET UP R4
ASHC @-(R4),R0 ;SHIFT R0:1 BY @-(4)
MOV @#PS,SPSW ;SAVE PS
CMPB #10,SPSW ;IS THE PS 10?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 760 <====

031466 012762 000617 177776
031474 005262 177774
031500 000000
4427 031502 022700 177525 1\$:
4428 031506 001406

MOV #617,-2(R2)
INC -4(R2)
HALT
CMP #177525,R0
BEQ 2\$

: MOVE TO MAILBOX # ***** 617 *****
: SET MSGTYP TO FATAL ERROR
: THE PS IS NOT EQUAL TO 10
: IS THE RESULT 177525?

031510 012762 000620 177776
031516 005262 177774
031522 000000
4429 031524 022701 052525 2\$:
4430 031530 001406

MOV #620,-2(R2)
INC -4(R2)
HALT
CMP #52525,R1
BEQ TST265

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 747 <

: MOVE TO MAILBOX # ***** 620 *****
: SET MSGTYP TO FATAL ERROR
: R0 IS NOT EQUAL TO 177525
: IS THE RESULT 52525?

031532 012762 000621 177776
031540 005262 177774
031544 000000

MOV #621,-2(R2)
INC -4(R2)
HALT

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 736 <

: MOVE TO MAILBOX # ***** 621 *****
: SET MSGTYP TO FATAL ERROR
: R0:1 IS NOT EQUAL TO 52525
: OR SEQUENCE ERROR

4431
4432
4433

.SBTTL ASHC SHIFTED BY @-(4) DM7 REG 4
:MODE 7 REG 4

4434

.SBTTL TEST # 265 - ASHC 1252552,125252,SHIF,@(4)=177525,52525 PS=10

 :TEST 265 - ASHC 1252552,125252,SHIF,@(4)=177525,52525 PS=10

031546 005212
 031550 022712 000265
 031554 001044
 4435 031556 012700 125252
 4436 031562 012701 125252
 4437 031566 000241
 4438 031570 012704 042556
 4439 031574 073074 000000
 4440 031600 013767 177776 010734
 4441 031606 122767 000010 010726
 4442 031614 001406

TST265: INC (R2) ;UPDATE TEST NUMBER
 CMP #265,(R2) ;SEQUENCE ERROR?
 BNE TST266-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125252,R0 ;LOAD R0 WITH 125252
 MOV #125252,R1 ;LOAD R0!1 WITH 125252
 CLC
 MOV #54,R4 ;SET UP R4
 ASHC @(R4),R0 ;SHIFT R0,R0!1 BY @(4)
 MOV @#PS,SPSW ;SAVE PS
 CMPB #10,SPSW ;IS THE PS 10?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 757 <=====
 :

031616 012762 000622 177776
 031624 005262 177774
 031630 000000
 4443 031632 022700 177525 1\$:
 4444 031636 001406

MOV #622,-2(R2)
 INC -4(R2)
 HALT
 CMP #177525,R0
 BEQ 2\$

: MOVE TO MAILBOX # ***** 622 *****
 : SET MSGTYP TO FATAL ERROR
 : THE PS IS NOT EQUAL TO 10
 : IS THE RESULT 177525?

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 746 <=====
 :

031640 012762 000623 177776
 031646 005262 177774
 031652 000000
 4445 031654 022701 052525 2\$:
 4446 031660 001406

MOV #623,-2(R2)
 INC -4(R2)
 HALT
 CMP #52525,R1
 BEQ TST266

: MOVE TO MAILBOX # ***** 623 *****
 : SET MSGTYP TO FATAL ERROR
 : R0 IS NOT EQUAL TO 177525
 : IS THE RESULT 52525?

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
 : CONDITIONAL BRANCH INST. AND <=====
 : REPLACE THE MOVE INSTRUCTION <=====
 : WHICH FOLLOWS W/ 735 <=====
 :

031662 012762 000624 177776
 031670 005262 177774
 031674 000000

MOV #624,-2(R2)
 INC -4(R2)
 HALT

: MOVE TO MAILBOX # ***** 624 *****
 : SET MSGTYP TO FATAL ERROR
 : R0!1 IS NOT EQUAL TO 52525
 : OR SEQUENCE ERROR

4447
 4448
 4449

.SBTTL ASHC SHIFTED BY -32. DM2 REG 7
 ;MODE 2 REG 7

4450

.SBTTL TEST # 266 - ASHC 100000 0 SHIFTED BY -32.--1 -1 PS=11

:TEST 266 - ASHC 100000 0 SHIFIED BY -32.--1 -1 PS 11

031676 005212
031700 022712 000266
031704 001042
4451 031706 012700 100000
4452 031712 012701 000000
4453 031716 000241
4454 031720 073027 177740
4455 031724 013767 177776 010610
4456 031732 122767 000011 010602
4457 031740 001406

TST266: INC (R2) ;UPDATE TEST NUMBER
CMP #266,(R2) ;SEQUENCE ERROR?
BNE TST267-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #100000,R0 ;LOAD R0 WITH 100000
MOV #0,R1 ;LOAD R0!1 WITH 0
CLC
ASHC #-32.,R0 ;SHIFT R0,R0!1 BY -32.
MOV @APS,SPSW ;SAVE PS
CMPB #11,SPSW ;IS THE PS 11?
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 761 <====

031742 012762 000625 177776
031750 005262 177774
031754 000000
4458 031756 022700 177777 1\$:
4459 031762 001406

MOV #625,-2(R2) ;MOVE TO MAILBOX # ***** 625 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;THE PS IS NOT EQUAL TO 11
CMP #-1,R0 ;IS THE RESULT -1?
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 750 <====

031764 012762 000626 177776
031772 005262 177774
031776 000000
4460 032000 022701 177777 2\$:
4461 032004 001406

MOV #626,-2(R2) ;MOVE TO MAILBOX # ***** 626 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0 IS NOT EQUAL TO -1
CMP #-1,R1 ;IS THE RESULT -1?
BEQ TST267

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 737 <

032006 012762 000627 177776
032014 005262 177774
032020 000000

MOV #627,-2(R2) ;MOVE TO MAILBOX # ***** 627 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;R0!1 IS NOT EQUAL TO -1
; OR SEQUENCE ERROR

4462
4463
4464

.SBTTL ASHC SHIFTED BY 15 DM2 REG 7
:MODE 2 REG 7

4465

.SBTTL TEST # 267 - ASHC 0 -1 SHIFTED BY 15.=77777 100000 PS=0

 :TEST 267 - ASHC 0 -1 SHIFTED BY 15.=77777 100000 PS=0

032022 005212
 032024 022712 000267
 032030 001042
 4466 032032 012700 000000
 4467 032036 012701 177777
 4468 032042 000241
 4469 032044 073027 000017
 4470 032050 013767 177776 010464
 4471 032056 122767 000000 010456
 4472 032064 001406

TST267: INC (R2) ;UPDATE TEST NUMBER
 CMP #267,(R2) ;SEQUENCE ERROR?
 BNE TST270-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD R0 WITH 0
 MOV #-1,R1 ;LOAD R0!1 WITH -1
 CLC
 ASHC #15.,R0 ;SHIFT R0,R0!1 BY 15.
 MOV @PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS THE PS 0?
 BEQ 1\$

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 761 <====

032066 012762 000630 177776
 032074 005262 177774
 032100 000000
 4473 032102 022700 077777 1\$:
 4474 032106 001406

MOV #630,-2(R2) ;MOVE TO MAILBOX # ***** 630 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 0
 CMP #77777,R0 ;IS THE RESULT 77777?
 BEQ 2\$

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 750 <====

032110 012762 000631 177776
 032116 005262 177774
 032122 000000
 4475 032124 022701 100000 2\$:
 4476 032130 001406

MOV #631,-2(R2) ;MOVE TO MAILBOX # ***** 631 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 77777
 CMP #100000,R1 ;IS THE RESULT 100000?
 BEQ TST270

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 737 <====

032132 012762 000632 177776
 032140 005262 177774
 032144 000000

MOV #632,-2(R2) ;MOVE TO MAILBOX # ***** 632 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0!1 IS NOT EQUAL TO 100000
 ; OR SEQUENCE ERROR

4477
 4478
 4479

.SBTTL ASHC SHIFTED BY 16. DM2 REG 7
 ;MODE 2 REG 7

4480

.SBTTL TEST # 270 - ASHC 0 52525 SHIFTED BY 16.=52525 0 PS=0

 :TEST 270 - ASHC 0 52525 SHIFTED BY 16.=52525 0 PS=0

032146 005212
 032150 022712 000270
 032154 001042
 4481 032156 012700 000000
 4482 032162 012701 052525
 4483 032166 000241
 4484 032170 073027 000020
 4485 032174 013767 177776 010340
 4486 032202 122767 000000 010332
 4487 032210 001406

TST270: INC (R2) ;UPDATE TEST NUMBER
 CMP #270,(R2) ;SEQUENCE ERROR?
 BNE TST271-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD R0 WITH 0
 MOV #52525,R1 ;LOAD R0!1 WITH 52525
 CLC
 ASHC #16.,R0 ;SHIFT R0,R0!1 BY 16.
 MOV @#PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS THE PS 0?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 761 <====

032212 012762 000633 177776
 032220 005262 177774
 032224 000000
 4488 032226 022700 052525 1\$:
 4489 032232 001406

MOV #633,-2(R2)
 INC -4(R2)
 HALT
 CMP #52525,R0
 BEQ 2\$

:MOVE TO MAILBOX # ***** 633 *****
 :SET MSGTYP TO FATAL ERROR
 :THE PS IS NOT EQUAL TO 0
 :IS THE RESULT 52525?

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 750 <====

032234 012762 000634 177776
 032242 005262 177774
 032246 000000
 4490 032250 022701 000000 2\$:
 4491 032254 001406

MOV #634,-2(R2)
 INC -4(R2)
 HALT
 CMP #0,R1
 BEQ TST271

:MOVE TO MAILBOX # ***** 634 *****
 :SET MSGTYP TO FATAL ERROR
 :R0 IS NOT EQUAL TO 52525
 :IS THE RESULT 0?

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 737 <====

032256 012762 000635 177776
 032264 005262 177774
 032270 000000

MOV #635,-2(R2)
 INC -4(R2)
 HALT

:MOVE TO MAILBOX # ***** 635 *****
 :SET MSGTYP TO FATAL ERROR
 :R0!1 IS NOT EQUAL TO 0
 : OR SEQUENCE ERROR

4492
 4493

:MODE 2 REG 7

4494

.SBTTL TEST # 271 - ASHC -1 0 SHIFTED BY 16. =0 0 PS 7

 :TEST 271 - ASHC -1 0 SHIFTED BY 16. =0 0 PS=7

032272 005212
 032274 022712 000271
 032300 001042
 4495 032302 012700 177777
 4496 032306 012701 000000
 4497 032312 000241
 4498 032314 073027 000020
 4499 032320 013767 177776 010214
 4500 032326 122767 000007 010206
 4501 032334 001406

TST271: INC (R2) ;UPDATE TEST NUMBER
 CMP #271,(R2) ;SEQUENCE ERROR?
 BNE TST272-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #-1,R0 ;LOAD R0 WITH -1
 MOV #0,R1 ;LOAD R0!1 WITH 0
 CLC
 ASHC #16.,R0 ;SHIFT R0,R0!1 BY 16.
 MOV @#PS,SPSW ;SAVE PS
 CMPB #7,SPSW ;IS THE PS 7?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 761 <====

032336 012762 000636 177776
 032344 005262 177774
 032350 000000
 4502 032352 022700 000000 1\$:
 4503 032356 001406

MOV #636,-2(R2) ;MOVE TO MAILBOX # ***** 636 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 7
 CMP #0,R0 ;IS THE RESULT 0?
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 750 <====

032360 012762 000637 177776
 032366 005262 177774
 032372 000000
 4504 032374 022701 000000 2\$:
 4505 032400 001406

MOV #637,-2(R2) ;MOVE TO MAILBOX # ***** 637 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 0
 CMP #0,R1 ;IS THE RESULT 0?
 BEQ TST272

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 737 <====

032402 012762 000640 177776
 032410 005262 177774
 032414 000000

MOV #640,-2(R2) ;MOVE TO MAILBOX # ***** 640 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0!1 IS NOT EQUAL TO 0
 ; OR SEQUENCE ERROR

4506
 4507
 4508

.SBTTL ASHC SHIFTED BY @ (4) DM7 REG 4
 ;MODE 7 REG 4

4509

.SBTTL TEST # 272 - ASHC 125252,125252,SHIF,@(4) 177525,52525 PS 10
 :*****
 :TEST 272 - ASHC 125252,125252,SHIF,@(4)=177525,52525 PS 10
 :*****

032416 005212
 032420 022712 000272
 032424 001044
 4510 032426 012701 125252
 4511 032432 012700 125252
 4512 032436 000241
 4513 032440 012704 042556
 4514 032444 073074 000000
 4515 032450 013767 177776 010064
 4516 032456 122767 000010 010056
 4517 032464 001406

TST272: INC (R2) ;UPDATE TEST NUMBER
 CMP #272,(R2) ;SEQUENCE ERROR?
 BNE TST273-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #125252,R1 ;LOAD R0!1 WITH 125252
 MOV #125252,R0 ;LOAD R0 WITH 125252
 CLC
 MOV #S4,R4 ;SET UP R4
 ASHC @(R4),R0 ;SHIFT R0,R0!1 BY @(4)
 MOV @#PS,SPSW ;SAVE PS
 CMPB #10,SPSW ;IS THE PS 10?
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 757 <

032466 012762 000641 177776
 032474 005262 177774
 032500 000000
 4518 032502 022701 052525 1\$:
 4519 032506 001406

MOV #641,-2(R2) ;MOVE TO MAILBOX # ***** 641 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;THE PS IS NOT EQUAL TO 10
 CMP #52525,R1 ;IS THE RESULT 52525?
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 746 <

032510 012762 000642 177776
 032516 005262 177774
 032522 000000
 4520 032524 022700 177525 2\$:
 4521 032530 001406

MOV #642,-2(R2) ;MOVE TO MAILBOX # ***** 642 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0!1 IS NOT EQUAL TO 52525
 CMP #177525,R0 ;IS THE RESULT 177525?
 BEQ TST273

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 735 <

032532 012762 000643 177776
 032540 005262 177774
 032544 000000

MOV #643,-2(R2) ;MOVE TO MAILBOX # ***** 643 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;R0 IS NOT EQUAL TO 177525
 ; OR SEQUENCE ERROR

4522
 4523

4524

.SBTTL TEST # 273 - MUL 100000 * #100000 - 40000 0 PS=1

 :TEST 273 - MUL 100000 * #100000 = 40000 0 PS=1

032546 005212
 032550 022712 000273
 032554 001043
 4525 032556 012706 001000
 4526 032562 005037 177776
 4527 032566 012700 100000
 4528 032572 070027 100000
 4529 032576 013767 177776 007736
 4530 032604 122767 000001 007730
 4531 032612 001406

TST273: INC (R2) ;UPDATE TEST NUMBER
 CMP #273,(R2) ;SEQUENCE ERROR?
 BNE TST274-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #STBOT,R6
 CLR @#PS
 MOV #100000,R0 ;LOAD MULTIPLICAN WITH 100000
 MUL #100000,R0 ;MULTIPLY 100000 BY #100000
 MOV @#PS,SPSW ;SAVE PS
 CMPB #1,SPSW ;IS PS = 1
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 760 <

032614 012762 000644 177776
 032622 005262 177774
 032626 000000
 4532 032630 022700 040000 1\$:
 4533 032634 001406

MOV #644,-2(R2) ;MOVE TO MAILBOX # ***** 644 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 1\$: CMP #40000,R0 ;IS HIGH ORDER = 40000
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 747 <====

032636 012762 000645 177776
 032644 005262 177774
 032650 000000
 4534 032652 022701 000000 2\$:
 4535 032656 001406

MOV #645,-2(R2) ;MOVE TO MAILBOX # ***** 645 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;HIGH ORDER IS WRONG
 2\$: CMP #0,R1 ;IS LOWER = 0
 BEQ TST274

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 736 <====

032660 012762 000646 177776
 032666 005262 177774
 032672 000000

MOV #646,-2(R2) ;MOVE TO MAILBOX # ***** 646 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;LOW ORDER IS WRONG
 ; OR SEQUENCE ERROR

4536
 4537
 4538

;TEST MSB OF LOWER PRODUCT CLEAR C-BIT
 ;MODE 2 /REG7

4539

.SBTTL TEST # 274 - MUL 70707 * #70707 = 31221 44261 PS=1

:TEST 274 - MUL 70707 * #70707 = 31221 44261 PS=1

032674 005212
032676 022712 000274
032702 001037
4540 032704 012700 070707
4541 032710 070027 070707
4542 032714 013767 177776 007620
4543 032722 122767 000001 007612
4544 032730 001406

TST274: INC (R2) ;UPDATE TEST NUMBER
CMP #274,(R2) ;SEQUENCE ERROR?
BNE TST275-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #70707,R0 ;LOAD MULTIPLICAN WITH 70707
MUL #70707,R0 ;MULTIPLY 70707 BY #70707
MOV @#PS,SPSW ;SAVE PS = 1
CMPB #1,SPSW ;IS PS = 1
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 764 <

032732 012762 000647 177776
032740 005262 177774
032744 000000
4545 032746 022700 031221 1\$:
4546 032752 001406

MOV #647,-2(R2) ;MOVE TO MAILBOX # ***** 647 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #31221,R0 ;IS HIGH ORDER =31221
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <-
: WHICH FOLLOWS W/ 753 <-

032754 012762 000650 177776
032762 005262 177774
032766 000000
4547 032770 022701 044261 2\$:
4548 032774 001406

MOV #650,-2(R2) ;MOVE TO MAILBOX # ***** 650 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;HIGH ORDER IS WRONG
CMP #44261,R1 ;IS LOWER ORDER =44261
BEQ TST275

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
: CONDITIONAL BRANCH INST. AND <=
: REPLACE THE MOVE INSTRUCTION <=
: WHICH FOLLOWS W/ 742 <=
: <=

032776 012762 000651 177776
033004 005262 177774
033010 000000

MOV #651,-2(R2) ;MOVE TO MAILBOX # ***** 651 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;LOW ORDER IS WRONG
; OR SEQUENCE ERROR

4549
4550
4551

:STORE UPPER PRODUCT AND SET C-BIT NEG #
:MODE 2 /REG 7

4552

.SBTTL TEST # 275 - MUL 107070 * #107070 = 31222 26100 PS=1
 :*****
 :TEST 275 - MUL 107070 * #107070 = 31222 26100 PS=1
 :*****

033012 005212
 033014 022712 000275
 033020 001037
 4553 033022 012700 107070
 4554 033026 070027 107070
 4555 033032 013767 177776 007502
 4556 033040 122767 000001 007474
 4557 033046 001406

TST275: INC (R2) ;UPDATE TEST NUMBER
 CMP #275,(R2) ;SEQUENCE ERROR?
 BNE TST276-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #107070,R0 ;LOAD MULTIPLICAN WITH 107070
 MUL #107070,R0 ;MULTIPLY 107070 BY #107070
 MOV @#1S,SPSW ;SAVE PS
 CPB #1,SPSW ;IS PS = 1
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 764 <====

033050 012762 000652 177776
 033056 005262 177774
 033062 000000
 4558 033064 022700 031222 1\$:
 4559 033070 001406

MOV #652,-2(R2) ;MOVE TO MAILBOX # ***** 652 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #31222,R0 ;IS HIGH ORDER = 31222
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 753 <

033072 012762 000653 177776
 033100 005262 177774
 033104 000000
 4560 033106 022701 026100 2\$:
 4561 033112 001406

MOV #653,-2(R2) ;MOVE TO MAILBOX # ***** 653 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT
 CMP #26100,R1 ;IS LOW ORDER = 26100
 BEQ TST276

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 742 <

033114 012762 000654 177776
 033122 005262 177774
 033126 000000

MOV #654,-2(R2) ;MOVE TO MAILBOX # ***** 654 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;LOW ORDER IS WRONG
 ; OR SEQUENCE ERROR

4562
 4563

;PUT MULTIPLIER IN R12 AND TEST SIGN FOR ZERO
 ;MODE 2 /REG 7

4564

.SBTTL TEST # 276 - MUL 77777 * #100000 = 140000 100000 PS=11

 :TEST 276 - MUL 77777 * #100000 = 140000 100000 PS=11

033130 005212
 033132 022712 000276
 033136 001037
 4565 033140 012700 077777
 4566 033144 070027 100000
 4567 033150 013767 177776 007364
 4568 033156 122767 000011 007356
 4569 033164 001406

TST276: INC (R2)
 CMP #276,(R2)
 BNE TST277-10
 MOV #77777,R0
 MUL #100000,R0
 MOV @#PS,SPSW
 CMPB #11,SPSW
 BEQ 1\$

;UPDATE TEST NUMBER
 ;SEQUENCE ERROR?
 ;BR TO ERROR HALT ON SEQ ERROR
 ;LOAD MULTIPLICAN WITH 77777
 ;MULTIPLY 77777 BY #100000
 ;SAVE PS
 ;IS PS = 11

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 764 <====

033166 012762 000655 177776
 033174 005262 177774
 033200 000000
 4570 033202 022700 140000
 4571 033206 001406

1\$: MOV #655,-2(R2)
 INC -4(R2)
 HALT
 CMP #140000,R0
 BEQ 2\$

;MOVE TO MAILBOX # ***** 655 *****
 ;SET MSGTYP TO FATAL ERROR
 ;PS IS WRONG
 ;IS HIGH ORDER = 140000

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 753 <====

033210 012762 000656 177776
 033216 005262 177774
 033222 000000
 4572 033224 022701 100000
 4573 033230 001406

2\$: MOV #656,-2(R2)
 INC -4(R2)
 HALT
 CMP #100000,R1
 BEQ TST277

;MOVE TO MAILBOX # ***** 656 *****
 ;SET MSGTYP TO FATAL ERROR
 ;HIGH ORDER IS WRONG
 ;IS LOW ORDER = 100000

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 742 <====

033232 012762 000657 177776
 033240 005262 177774
 033244 000000

MOV #657,-2(R2)
 INC -4(R2)
 HALT

;MOVE TO MAILBOX # ***** 657 *****
 ;SET MSGTYP TO FATAL ERROR
 ;LOW ORDER IS WRONG
 ; OR SEQUENCE ERROR

4574
 4575
 4576

;SET Z-BIT R17 = 0
 ;MODE 2 /REG 7

4577

.SBTTL TEST # 277 - MUL -1 * #0 = 0 0 PS=4

:TEST 277 - MUL -1 * #0 = 0 0 PS=4

033246 005212
033250 022712 000277
033254 001037
4578 033256 012700 177777
4579 033262 070027 000000
4580 033266 013767 177776 007246
4581 033274 122767 000004 007240
4582 033302 001406

TST277: INC (R2) ;UPDATE TEST NUMBER
CMP #277,(R2) ;SEQUENCE ERROR?
BNE TST300-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #-1,R0 ;LOAD MULTIPLICAN WITH -1
MUL #0,R0 ;MULTIPLY -1 BY #0
MOV @#PS,SPSW ;SAVE PS
CMPB #4,SPSW ;IS PS = 4
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 764 <=====
:

033304 012762 000660 177776
033312 005262 177774
033316 000000
4583 033320 022700 000000
4584 033324 001406

1\$: MOV #660,-2(R2) ;MOVE TO MAILBOX # ***** 660 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #0,R0 ;IS HIGH ORDER = 0
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 753 <=====
:

033326 012762 000661 177776
033334 005262 177774
033340 000000
4585 033342 022701 000000
4586 033346 001406

2\$: MOV #661,-2(R2) ;MOVE TO MAILBOX # ***** 661 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;HIGH ORDER IS WRONG
CMP #0,R1 ;IS LOW ORDER = 0
BEQ TST300

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 742 <=====
:

033350 012762 000662 177776
033356 005262 177774
033362 000000

MOV #662,-2(R2) ;MOVE TO MAILBOX # ***** 662 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;LOW ORDER IS WRONG
; OR SEQUENCE ERROR

4587
4588
4589

:TEST UPPER PRODUCT FOR ALL ONES,CLEAR C-BIT
:MODE 2 /REG 7

4590

.SBTTL TEST # 300 - MUL -1 * #77777 = 100001 100001 PS=10

:TEST 300 - MUL -1 * #77777 = 100001 100001 PS=10

033364 005212
033366 022712 000300
033372 001037
4591 033374 012701 177777
4592 033400 070127 077777
4593 033404 013767 177776 007130
4594 033412 122767 000010 007122
4595 033420 001406

TST300: INC (R2)
CMP #300,(R2)
BNE TST301-10
MOV #-1,R1
MUL #77777,R1
MOV @#PS,SPSW
CMPB #10,SPSW
BEQ 1\$

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:LOAD MULTIPLICAN WITH -1
:MULTIPLY -1 BY #77777
:SAVE PS
:IS PS = 10

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 764 <====

033422 012752 000663 177776
033430 005262 177774
033434 000000
4596 033436 022701 100001
4597 033442 001406

MOV #663,-2(R2)
INC -4(R2)
HALT
1\$: CMP #100001,R1
BEQ 2\$

:MOVE TO MAILBOX # ***** 663 *****
:SET MSGTYP TO FATAL ERROR
:PS IS WRONG
:IS HIGH ORDER = 100001

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 753 <====

033444 012762 000664 177776
033452 005262 177774
033456 000000
4598 033460 022701 100001
4599 033464 001406

MOV #664,-2(R2)
INC -4(R2)
HALT
2\$: CMP #100001,R1
BEQ TST301

:MOVE TO MAILBOX # ***** 664 *****
:SET MSGTYP TO FATAL ERROR
:HIGH ORDER IS WRONG
:IS LOW ORDER = 100001

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 742 <====

033466 012762 000665 177776
033474 005262 177774
033500 000000

MOV #665,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 665 *****
:SET MSGTYP TO FATAL ERROR
:LOW ORDER IS WRONG
: OR SEQUENCE ERROR

4600

:MODE 2 /REG 7

TEST # 301 - MUL 2 * #2 = 0 4 PS=0

4601

.SBTTL TEST # 301 - MUL 2 * #2 0 4 PS=0

:TEST 301 - MUL 2 * #2 0 4 PS 0

033502 005212
033504 022712 000301
033510 001037
4602 033512 012700 000002
4603 033516 070027 000002
4604 033522 013767 177776 007012
4605 033530 122767 000000 007004
4606 033536 001406

TST301: INC (R2) ;UPDATE TEST NUMBER
CMP #301,(R2) ;SEQUENCE ERROR?
BNE TST302-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #2,R0 ;LOAD MULTIPLICAN WITH 2
MUL #2,R0 ;MULTIPLY 2 BY #2
MOV @#PS,SPSW ;SAVE PS
CMPB #0,SPSW ;IS PS=0
BEQ 1\$

;
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 764 <

033540 012762 000666 177776
033546 005262 177774
033552 000000
4607 033554 022700 000000 1\$:
4608 033560 001406

MOV #666,-2(R2) ;MOVE TO MAILBOX # ***** 666 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #0,R0 ;IS HIGH ORDER =0
BEQ 2\$

;
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 753 <

033562 012762 000667 177776
033570 005262 177774
033574 000000
4609 033576 022701 000004 2\$:
4610 033602 001406

MOV #667,-2(R2) ;MOVE TO MAILBOX # ***** 667 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;HIGH ORDER IS WRONG
CMP #4,R1 ;IS LOW ORDER =4
BEQ TST302

;
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 742 <

033604 012762 000670 177776
033612 005262 177774
033616 000000

MOV #670,-2(R2) ;MOVE TO MAILBOX # ***** 670 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;LOW ORDER IS WRONG
; OR SEQUENCE ERROR

4611
4612

:MODE 2 /REG 7

4613

.SBTTL TEST # 302 - MUL 1 * #-1 -1 -1 PS 10
:.....
:TEST 302 - MUL 1 * #-1 -1 -1 PS-10
:.....

033620 005212
033622 022712 000302
033626 001037
4614 033630 012700 000001
4615 033634 070027 177777
4616 033640 013767 177776 006674
4617 033646 122767 000010 006666
4618 033654 001406

TST302: INC (R2) ;UPDATE TEST NUMBER
CMP #302,(R2) ;SEQUENCE ERROR?
BNE TST303-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #1,R0 ;LOAD MULTIPLICAN WITH 1
MUL #-1,R0 ;MULTIPLY 1 BY #-1
MOV @#PS,SPSW ;SAVE PS
CMPB #10,SPSW ;IS PS =10
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 764 <====

033656 012762 000671 177776
033664 005262 177774
033670 000000
4619 033672 022700 177777 1\$:
4620 033676 001406

MOV #671,-2(R2) ;MOVE TO MAILBOX # ***** 671 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #-1,R0 ;IS HIGH ORDER =-1
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 753 <====

033700 012762 000672 177776
033706 005262 177774
033712 000000
4621 033714 022701 177777 2\$:
4622 033720 001406

MOV #672,-2(R2) ;MOVE TO MAILBOX # ***** 672 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;HIGH ORDER IS WRONG
CMP #-1,R1 ;IS LOW ORDER =-1
BEQ TST303

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 742 <====

033722 012762 000673 177776
033730 005262 177774
033734 000000

MOV #673,-2(R2) ;MOVE TO MAILBOX # ***** 673 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;LOW ORDER IS WRONG
; OR SEQUENCE ERROR

4627

:MODE 2 /REG 7

4624

.SBTTL TEST # 303 - MUL -1 * #100000 = 0 100000 PS=1

:TEST 303 - MUL -1 * #100000 = 0 100000 PS=1

033736 005212
033740 022712 000303
033744 001037
4625 033746 012700 177777
4626 033752 070027 100000
4627 033756 013767 177776 006556
4628 033764 122767 000001 006550
4629 033772 001406

TST303: INC (R2) ;UPDATE TEST NUMBER
CMP #303,(R2) ;SEQUENCE ERROR?
BNE TST304-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #-1,R0 ;LOAD MULTIPLICAN WITH -1
MUL #100000,R0 ;MULTIPLY -1 BY 100000
MOV @#PS,SPSW ;SAVE PS
CMPB #1,SPSW ;IS PS =1
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 764

033774 012762 000674 177776
034002 005262 177774
034006 000000
4630 034010 022700 000000
4631 034014 001406

1\$: MOV #674,-2(R2) ;MOVE TO MAILBOX # ***** 674 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #0,R0 ;IS HIGH ORDER = 0
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 753

034016 012762 000675 177776
034024 005262 177774
034030 000000
4632 034032 022701 100000
4633 034036 001406

2\$: MOV #675,-2(R2) ;MOVE TO MAILBOX # ***** 675 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;HIGH ORDER IS WRONG
CMP #100000,R1 ;IS LOW ORDER =100000
BEQ TST304

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 742

034040 012762 000676 177776
034046 005262 177774
034052 000000

MOV #676,-2(R2) ;MOVE TO MAILBOX # ***** 676 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;LOW ORDER IS WRONG
; OR SEQUENCE ERROR

4634
4635
4636
4637

.SBTTL DIV USING DM6 REG 7

4638

.SBTTL TEST # 304 - DIV 0 52525/S1=25252 REM=1 PS 0

 :TEST 304 - DIV 0 52525/S1=25252 REM=1 PS=0

034054 005212
 034056 022712 000304
 034062 001043
 4639 034064 012700 000000
 4640 034070 012701 052525
 4641 034074 012703 042550
 4642 034100 071067 006444
 4643 034104 013767 177776 006430
 4644 034112 122767 000000 006422
 4645 034120 001406

TST304: INC (R2) ;UPDATE TEST NUMBER
 CMP #304,(R2) ;SEQUENCE ERROR?
 BNE TST305-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD HIGH ORDER WITH R0
 MOV #52525,R1 ;LOAD LOW ORDER WITH 52525
 MOV #S1,R3 ;SET UP R3
 DIV S1,R0 ;DIVIDE BY S1
 MOV @RPS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS PS=0
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 760 <====

034122 012762 000677 177776
 034130 005262 177774
 034134 000000
 4646 034136 022700 025252
 4647 034142 001406

1\$: MOV #677,-2(R2) ;MOVE TO MAILBOX # ***** 677 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #25252,R0 ;IS QUOTIENT =25252
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 747 <====

034144 012762 000700 177776
 034152 005262 177774
 034156 000000
 4648 034160 022701 000001
 4649 034164 001406

2\$: MOV #700,-2(R2) ;MOVE TO MAILBOX # ***** 700 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #1,R1 ;IS REMAINDER =1
 BEQ TST305

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 736 <====

034166 012762 000701 177776
 034174 005262 177774
 034200 000000

MOV #701,-2(R2) ;MOVE TO MAILBOX # ***** 701 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

4650
 4651

.SBTTL DIV USING DM7 REG 7

4652

.SBTTL TEST # 305 - DUV 0 52525/@S2 =25252 REM=1 PS 0
 :*****
 :TEST 305 - DUV 0 52525/@S2 =25252 REM=1 PS=0
 :*****

034202 005212
 034204 022712 000305
 034210 001043
 4653 034212 012700 000000
 4654 034216 012701 052525
 4655 034222 012704 042552
 4656 034226 071077 006320
 4657 034232 013767 177776 006302
 4658 034240 122767 000000 006274
 4659 034246 001406

TST305: INC (R2) ;UPDATE TEST NUMBER
 CMP #305,(R2) ;SEQUENCE ERROR?
 BNE TST306-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD HIGH ORDER WITH R0
 MOV #52525,R1 ;LOAD LOW ORDER WITH 52525
 MOV #S2,R4 ; SET UP R4
 DIV @S2,R0 ;DIVIDE BY @S2
 MOV @#PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS PS=0
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 760 <====

034250 012762 000702 177776
 034256 005262 177774
 034262 000000
 4660 034264 022700 025252 1\$:
 4661 034270 001406

MOV #702,-2(R2) ;MOVE TO MAILBOX # ***** 702 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #25252,R0 ;IS QUOTIENT =25252
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 747 <====

034272 012762 000703 177776
 034300 005262 177774
 034304 000000
 4662 034306 022701 000001 2\$:
 4663 034312 001406

MOV #703,-2(R2) ;MOVE TO MAILBOX # ***** 703 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #1,R1 ;IS REMINDER =1
 BEQ TST306

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 736 <====

034314 012762 000704 177776
 034322 005262 177774
 034326 000000

MOV #704,-2(R2) ;MOVE TO MAILBOX # ***** 704 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMINDR IS WRONG
 ; OR SEQUENCE ERROR

4664
 4665
 4666
 4667

.SBTTL DIV USING DM3 REG 7
 ;CLEAR V-BIT MODE 3 REG 7

4668

.SBTTL TEST # 306 - DIV 0 52525/@#S1 =25252 REM=1 PS=0

:TEST 306 - DIV 0 52525/@#S1 =25252 REM=1 PS=0

034330 005212
034332 022712 000306
034336 001043
4669 034340 012700 000000
4670 034344 012701 052525
4671 034350 012703 042550
4672 034354 071037 042550
4673 034360 013767 177776 006154
4674 034366 122767 000000 006146
4675 034374 001406

TST306: INC (R2) ;UPDATE TEST NUMBER
CMP #306,(R2) ;SEQUENCE ERROR?
BNE TST307-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #0,R0 ;LOAD HIGH ORDER WITH R0
MOV #52525,R1 ;LOAD LOW ORDER WITH 52525
MOV #5,R3 ;SET UP R3
DIV @#S1,R0 ;DIVIDE BY @#S1
MOV @#PS,SPSW ;SAVE PS
CMPB #0,SPSW ;IS PS=0
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 760 <

034376 012762 000705 177776
034404 005262 177774
034410 000000
4676 034412 022700 025252 1\$:
4677 034416 001406

MOV #705,-2(R2) ;MOVE TO MAILBOX # ***** 705 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #25252,R0 ;IS QUOTIENT =25252
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 747 <-

034420 012762 000706 177776
034426 005262 177774
034432 000000
4678 034434 022701 000001 2\$:
4679 034440 001406

MOV #706,-2(R2) ;MOVE TO MAILBOX # ***** 706 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;QUOTIENT IS WRONG
CMP #1,R1 ;IS REMAINDER =1
BEQ TST307

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <-
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 736 <

034442 012762 000707 177776
034450 005262 177774
034454 000000

MOV #707,-2(R2) ;MOVE TO MAILBOX # ***** 707 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REMAINDER IS WRONG
; OR SEQUENCE ERROR

4680
4681
4682

.SBTTL DIV USING DMO REG 4
:MODE 0 REG 2

4683

..SBTTL TEST # 307 - DIV 0 52525 /R4=25252 REM=1 PS=0

 :TEST 307 - DIV 0 52525 /R4=25252 REM=1 PS=0

034456 005212
 034460 022712 000307
 034464 001042
 4684 034466 012700 000000
 4685 034472 012701 052525
 4686 034476 012704 000002
 4687 034502 071004
 4688 034504 013767 177776 006030
 4689 034512 122767 000000 006022
 4690 034520 001406

TST307: INC (R2) ;UPDATE TEST NUMBER
 CMP #307,(R2) ;SEQUENCE ERROR?
 BNE TST310-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD HIGH ORDER WITH 0
 MOV #52525,R1 ;LOAD LOW ORDER WITH 52525
 MOV #2,R4 ;SET UP R4
 DIV R4,R0 ;DIVIDE BY R4
 MOV @#PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS PS=0
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 761 <====

034522 012762 000710 177776
 034530 005262 177774
 034534 000000
 4691 034536 022700 025252
 4692 034542 001406

1\$: MOV #710,-2(R2) ;MOVE TO MAILBOX # ***** 710 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #25252,R0 ;IS QUOTIENT=25252
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 750 <====

034544 012762 000711 177776
 034552 005262 177774
 034556 000000
 4693 034560 022701 000001
 4694 034564 001406

2\$: MOV #711,-2(R2) ;MOVE TO MAILBOX # ***** 711 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #1,R1 ;IS REMAINDER =1
 BEQ TST310

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 737 <====

034566 012762 000712 177776
 034574 005262 177774
 034600 000000

MOV #712,-2(R2) ;MOVE TO MAILBOX # ***** 712 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

4695
 4696
 4697
 4698
 4699
 4700
 4701
 4702
 4703
 4704
 4705
 4706
 4707
 4708
 4709
 4710
 4711

 WARNING!

THE FOLLOWING CIS ABORT TEST MUST BE LOCATED BELOW
 MEMORY LOCATION 40000 DUE TO THE MEMORY MANAGEMENT
 SETUP REQUIRED IN THE TEST.

ALSO THE TEST DESTROYS THE CONTENTS OF MEMORY IN THE
 FOLLOWING RANGES:

57670-60070
 76000-76001

THIS WARNING IS PRIMARILY AIMED AT FUTURE MODIFIERS OF THIS CPU DIAGNOSTIC.

4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745

.....

CIS STACK 'PROBE AHEAD' MEMORY MANAGEMENT ABORT TESTS

THE NEXT THREE SUBTESTS ARE AIMED AT TESTING THE KT
PAGE FAULT ROM (SCHEMATIC PAGE K1-8) AND ASSOCIATED LOGIC.
NOTE: THESE 3 SUBTESTS ARE OPTIONAL AND ARE SELECTED BY SETTING MFM
HARDWARE SWITCH REGISTER BIT 08 TO A 1 IN THE CASE OF
STANDALONE OPERATION OF THE DIAGNOSTIC. IN THE CASE OF
MANUFACTURING APT RUNTIME MODE THEN BIT 08 OF \$SWREG IS SET TO 1
THRU APT SCRIPTING.

EACH OF THESE 3 SUBTESTS SETUP THE STACK POINTER SUCH THAT
WHEN THE CIS PROCESSOR CHECKS TO SEE IF THERE IS ENOUGH
SPACE ON THE STACK (200 BYTES) THE ANSWER FOUND SHOULD BE
NO BECAUSE A PORTION OF THE STACK IS IN PROTECTED MEMORY.

EACH OF THESE 3 SUBTESTS VERIFY THAT THE CIS INSTRUCTION ABORTS
UNDER SEVERAL CONDITIONS OF MEMORY MANAGEMENT PAGE PROTECTION.
ALL OF THE PAGE PROTECTION DATA IS LISTED IN TABLE 'PDRTAB'.

THIS TEST ALLOWS MFG. TO ELIMINATE
HAVING TO RUN THE CIS DIAGNOSTIC DURING QUICK
VERIFY TESTING OF THE CPU MODULE.

000377
076130

FILL=377
MOVCI=76130

4746

.SBTTL TEST # 310 - CIS STACK PROBE AHEAD MEM MGMT ABORT TEST
 :*****
 :TEST 310 - CIS STACK PROBE AHEAD MEM MGMT ABORT TEST
 :*****

034602 005212
 034604 022712 000310
 034610 001020
 4747 034612 132767 000200 143501
 4748 034620 001405
 4749 034622 032767 000400 143472
 4750 034630 001406
 4751 034632 000411
 4752 034634 032737 000400 177570
 4753 034642 001401
 4754 034644 000404
 4755 034646 000167 001102
 4756 034652 000167 001066

TST310: INC (R2) ;UPDATE TEST NUMBER
 CMP #310,(R2) ;SEQUENCE ERROR?
 BNE 6\$;BR TO ERROR HALT ON SEQ ERROR
 BITB #200,\$ENVM ;IS APT SIZING?
 BEQ 4\$;NO; TRY HARDWARE SWITCH REGISTER
 BIT #400,\$SWREG ;YES APT IS SIZING; DOES APT SAY TO DO THIS TEST
 BEQ 5\$;NO ; SKIP TEST
 BR 1\$;YES, DO TEST
 4\$: BIT #400,@#177570 ;DOES HARDWARE SWITCH REGISTER SAY TO DO TEST?
 BEQ 5\$;NO, SKIP TEST
 BR 1\$;YES, DO TEST
 5\$: JMP TST311
 6\$: JMP TST311-10

4757

;SETUP PAR'S FOR DIRECT MAPPING

4758

4759

4760 034656 005037 172516
 4761 034662 010667 001016

1\$: CLR @MMR3 ;CLEAR OUT D-SPACE ENABLES
 MOV SP,STK1 ;SAVE THE STACK POINTER

4762

4763 034666 012737 000000 172340
 4764 034674 012737 000200 172342
 4765 034702 012737 000400 172344
 4766 034710 012737 000600 172346
 4767 034716 012737 001000 172350
 4768 034724 012737 001200 172352
 4769 034732 012737 001400 172354
 4770 034740 012737 177600 172356

MOV #0,@KIPAR0 ;SETUP KERNEL I-SPACE PAR'S
 MOV #200,@KIPAR1
 MOV #400,@KIPAR2
 MOV #600,@KIPAR3
 MOV #1000,@KIPAR4
 MOV #1200,@KIPAR5
 MOV #1400,@KIPAR6
 MOV #177600,@KIPAR7

4771

4772 034746 012737 000000 172240
 4773 034754 012737 000200 172242
 4774 034762 012737 000400 172244
 4775 034770 012737 000600 172246
 4776 034776 012737 001000 172250
 4777 035004 012737 001200 172252
 4778 035012 012737 001400 172254
 4779 035020 012737 177600 172256

MOV #0,@SIPAR0 ;SETUP SUPERVISOR I-SPACE PAR'S
 MOV #200,@SIPAR1
 MOV #400,@SIPAR2
 MOV #600,@SIPAR3
 MOV #1000,@SIPAR4
 MOV #1200,@SIPAR5
 MOV #1400,@SIPAR6
 MOV #177600,@SIPAR7

4780

4781 035026 012737 000000 177640
 4782 035034 012737 000200 177642
 4783 035042 012737 000400 177644
 4784 035050 012737 000600 177646
 4785 035056 012737 001000 177650
 4786 035064 012737 001200 177652
 4787 035072 012737 001400 177654
 4788 035100 012737 177600 177656

MOV #0,@UIPAR0 ;SETUP USER I-SPACE PAR'S
 MOV #200,@UIPAR1
 MOV #400,@UIPAR2
 MOV #600,@UIPAR3
 MOV #1000,@UIPAR4
 MOV #1200,@UIPAR5
 MOV #1400,@UIPAR6
 MOV #177600,@UIPAR7

4789

4790

4791

4792

4793

;SETUP PDR'S FOR R/W ACCESS

4794 035106

4795 035106 012700 172300

4796 035112 012720 177406

SETPDR:
 1\$: MOV #KIPDR0,R0
 MOV #177406,(R0)+

```

4797 035116 020027 172316          CMP R0,#KIPDR7
4798 035122 101773          BLOS 1$
4799
4800 035124 012700 172200          MOV #SIPDR0,R0
4801 035130 012720 177406          2$: MOV #177406,(R0)+
4802 035134 020027 172216          CMP R0,#SIPDR7
4803 035140 101773          BLOS 2$
4804
4805 035142 012700 177600          MOV #UIPDR0,R0
4806 035146 012720 177406          3$: MOV #177406,(R0)+
4807 035152 020027 177616          CMP R0,#UIPDR7
4808 035156 101773          BLOS 3$
4809
4810          ;KERNEL MODE CIS STACK PROBE AHEAD MEM MGMT ABORT SUBTEST
4811          ;
4812
4813 035160          KMTSTS:
4814 035160 012737 035564 000250          MOV #MMHDLR,@MMVEC          ;SETUP MEM MGMT INTERRUPT VECTOR
4815 035166 012737 000340 000252          MOV #PR7,@MMVEC+2
4816 035174 012701 035244          MOV #1$,R1          ;SETUP INTR RETURN ADDRESS
4817 035200 012700 035706          MOV #PDRTAB,R0
4818 035204 012706 060070          MOV #60070,SP
4819
4820 035210 011037 172304          2$: MOV (R0),@#KIPDR2          ;PROTECT PART OF STACK
4821 035214 012737 000001 177572          MOV #1,@MMR0          ;TURN ON MEMORY MGMT
4822
4823 035222 004767 000356          JSR PC,SAVR          ;SAVE REGISTERS
4824
4825 035226 076130          MOVCI          ;EXECUTE THE CIS INSTRUCTION
4826 035230 035574          SRC.PTR
4827 035232 035600          DST.PTR
4828 035234 000377          FILL
4829 035236 000240          NOP
4830 035240 000240          NOP
4831 035242 0C0000          HALT          ;CIS INSTRUCTION SHOULD HAVE ABORTED BUT DIDN'T
4832
4833 035244 004767 000366          1$: JSR PC,RESR          ;RESTORE REGISTERS
4834 035250 062700 000002          ADD #2,R0          ;UPDATE PROTECTION SCHEME TO NEXT TABLE CASE
4835 035254 005710          TST (R0)          ;ANY CASES LEFT TO TRY?
4836 035256 001354          BNE 2$          ;BRANCH IF YES
4837
4838 035260 005037 177572          CLR @MMR0          ;NO - PREPARE TO EXIT TEST
4839 035264 012737 177406 172304          MOV #177406,@#KIPDR2          ;RESTORE R/W ACCESS TO STACK AREA
4840
4841
4842          ;SUPERVISOR MODE CIS STACK PROBEAHEAD MEMORY MGMT ABORT SUBTEST
4843          ;
4844
4845 035272 012737 035564 000250          SMISTS: MOV #MMHDLR,@MMVEC          ;SETUP MEM MGMT INTERRUPT VECTOR
4846 035300 012737 040340 000252          MOV #040340,@MMVEC+2
4847 035306 012701 035372          MOV #1$,R1          ;SETUP INTR RETURN ADDRESS
4848 035312 012700 035706          MOV #PDRTAB,R0
4849 035316 012737 040340 177776          MOV #040340,@#PSW          ;SWITCH TO SUPERVISOR MODE
4850 035324 012706 060070          MOV #60070,SP
4851
4852 035330 011037 172204          2$: MOV (R0),@#SIPDR2          ;PROTECT PART OF STACK
4853 035334 012737 000001 177572          MOV #1,@MMR0          ;TURN ON MEMORY MGMT
    
```

```

4854
4855 035342 004767 000236      JSR PC,SAVR      ;SAVE REGISTERS
4856
4857 035346 076130      MOVCI           ;EXECUTE THE CIS INSTRUCTION
4858 035350 035574      SRC.PTR
4859 035352 035600      DST.PTR
4860 035354 000377      FILL
4861
4862 035356 000240      NOP
4863 035360 000240      NOP
4864 035362 012737 000340 177776    MOV #340,@#PSW  ;SWITCH BACK TO KERNEL MODE BEFORE HALT
4865 035370 000000      HALT           ;CIS INSTRUCTION SHOULD HAVE ABORTED BUT DIDN'T
4866
4867 035372 004767 000240      1$: JSR PC,RESR    ;RESTORE REGISTERS
4868 035376 062700 000002      ADD #2,R0      ;UPDATE PROTECTION SCHEME TO NEXT TABLE CASE
4869 035402 005710      TST (R0)      ;ANY CASES LEFT TO TRY?
4870 035404 001351      BNE 2$        ;BRANCH IF YES
4871
4872 035406 005037 177572      CLR @MMR0     ;NO - PREPARE TO EXIT TEST
4873 035412 012737 177406 172204    MOV #177406,@#SIPDR2 ;RESTORE R/W ACCESS TO STACK AREA
4874 035420 000400      BR UMTSTS     ;GO TO NEXT TEST
4875
4876
4877      ;USER MODE CIS STACK PROBEAHEAD MEM MGMT ABORT SUBTEST
4878      :
4879
4880      UMTSTS:
4881 035422 012737 035564 000250    MOV #MMHDLR,@MMVEC ;SETUP MEM MGMT INTERRUPT VECTOR
4882 035430 012737 140340 000252    MOV #140340,@MMVEC+2
4883 035436 012701 035522      MOV #1$,R1    ;SETUP INTR RETURN ADDRESS
4884 035442 012700 035706      MOV #PDRTAB,R0
4885 035446 012737 140340 177776    MOV #140340,@#PSW  ;SWITCH TO USER MODE
4886 035454 012706 060070      MOV #60070,SP
4887
4888 035460 011037 177604      2$: MOV (R0),@#UIPDR2 ;PROTECT PART OF STACK
4889 035464 012737 000001 177572    MOV #1,@MMR0    ;TURN ON MEMORY MGMT
4890
4891 035472 004767 000106      JSR PC,SAVR   ;SAVE REGISTERS
4892
4893 035476 076130      MOVCI        ;EXECUTE THE CIS INSTRUCTION
4894 035500 035574      SRC.PTR
4895 035502 035600      DST.PTR
4896 035504 000377      FILL
4897
4898 035506 000240      NOP
4899 035510 000240      NOP
4900 035512 012737 000340 177776    MOV #340,@#PSW  ;SWITCH BACK TO KERNEL MODE BEFORE HALT
4901 035520 000000      HALT         ;CIS INSTRUCTION SHOULD HAVE ABORTED BUT DIDN'T
4902
4903 035522 004767 000110      1$: JSR PC,RESR    ;RESTORE REGISTERS
4904 035526 062700 000002      ADD #2,R0    ;UPDATE PROTECTION SCHEME TO NEXT TABLE CASE
4905 035532 005710      TST (R0)    ;ANY CASES LEFT TO TRY?
4906 035534 001351      BNE 2$      ;BRANCH IF YES
4907
4908 035536 005037 177572      CLR @MMR0    ;NO - PREPARE TO EXIT TEST
4909 035542 012737 177406 177604    MOV #177406,@#UIPDR2 ;RESTORE R/W ACCESS TO STACK AREA
4910 035550 012737 000340 177776    MOV #340,@#PSW  ;SWITCH BACK TO KERNEL MODE

```

```

4911 035556 016706 000122          MOV STK1,SP          ;RESTORE THE STACK POINTER
4912 035562 000471          BR ENDABO           ;GO TO NEXT TEST
4913
4914
4915
4916          ;MEMORY MANAGEMENT TRAP HANDLER
4917          ;
4918 035564          ;MMHDLR:
4919 035564 005037 177572          CLR @MMR0           ;TURN OFF MEM MGMT
4920 035570 022626          CMP (S?)+,(SP)+   ;FIX UP STACK
4921 035572 000111          JMP (R1)           ;RETURN VIA R1
4922
4923
4924          ;CIS INSTRUCTION SOURCE AND DESTINATION DESCRIPTORS
4925          ;
4926          SRC.PTR:          .WORD 1
4927 035574 000001          .WORD 76000
4928 035576 076000          DST.PTR:          .WORD 1
4929 035600 000001          .WORD 76001
4930 035602 076001
4931
4932          ;SUBROUTINES
4933          ;
4934 035604 010067 000060          SAVR:  MOV R0,SVR0          ;SAVE REGISTERS
4935 035610 010167 000056          MOV R1,SVR1
4936 035614 010267 000054          MOV R2,SVR2
4937 035620 010367 000052          MOV R3,SVR3
4938 035624 010467 000050          MOV R4,SVR4
4939 035630 010567 000046          MOV R5,SVR5
4940 035634 000207          RTS PC
4941
4942 035636 016700 000026          RESR:  MOV SVR0,R0          ;RESTORE REGISTERS
4943 035642 016701 000024          MOV SVR1,R1
4944 035646 016702 000022          MOV SVR2,R2
4945 035652 016703 000020          MOV SVR3,R3
4946 035656 016704 000016          MOV SVR4,R4
4947 035662 016705 000014          MOV SVR5,R5
4948 035666 000207          RTS PC
4949
4950 035670 000000          SVR0:  .WORD 0
4951 035672 000000          SVR1:  .WORD 0
4952 035674 000000          SVR2:  .WORD 0
4953 035676 000000          SVR3:  .WORD 0
4954 035700 000000          SVR4:  .WORD 0
4955 035702 000000          SVR5:  .WORD 0
4956
4957 035704 000000          STK1:  .WORD 0
4958
4959
4960          ;PROTECTION TABLE (WORD FORMAT = PDR FORMAT)
4961 035706          PDRTAB:
4962 035706 177000          177000          ;ACF=00          ED=0  PLF=176
4963 035710 177410          177410          ;ACF=00          ED=1  PLF=177
4964 035712 177400          177400          ;ACF=00          ED=0  PLF=177
4965 035714 100010          100010          ;ACF=00          ED=1  PLF=0
4966
4967 035716 177002          177002          ;ACF=01          ED=0  PLF=176
  
```

4968	035720	177412	177412	;ACF=01	ED=1	PLF=177
4969	035722	177402	177402	;ACF=01	ED=0	PLF=177
4970	035724	100012	100012	;ACF=01	ED=1	PLF=0
4971						
4972	035726	177004	177004	;ACF=10	ED=0	PLF=176
4973	035730	177414	177414	;ACF=10	ED=1	PLF=177
4974	035732	177404	177404	;ACF=10	ED=0	PLF=177
4975	035734	100014	100014	;ACF=10	ED=1	PLF=0
4976						
4977	035736	177006	177006	;ACF=11	ED=0	PLF=176
4978	035740	177416	177416	;ACF=11	ED=1	PLF=177
4979	035742	000000	0			
4980						
4981						
4982	035744	000000				
4983	035746	000240				
4984	035750	000240				
4985	035752	000240				

ENDABO: HALT ;TEST SEQUENCE ERROR
NOP
NOP
NOP

CKKAABO 11/44 CPU/EIS
DIV USING DM4 REG 3

MACRO M1113 06-APR-81 14:04 PAGE 215

F 5

SEQUENCE 265

4986
4987

.SBTTL DIV USING DM4 REG 3
;MODE 4 REG 3

4988

.SBTTL TEST # 311 - DIV 0 52525 /-(3)=25252 REM=1 PS=0

 :TEST 311 - DIV 0 52525 /-(3)=25252 REM=1 PS=0

035754 005212
 035756 022712 000311
 035762 001044
 4989 035764 005067 142006
 4990 035770 012700 000000
 4991 035774 012701 052525
 4992 036000 012703 042552
 4993 036004 071043
 4994 036006 013767 177776 004526
 4995 036014 122767 000000 004520
 4996 036022 001406

TST311: INC (R2) ;UPDATE TEST NUMBER
 CMP #311,(R2) ;SEQUENCE ERROR?
 BNE TST312-10 ;BR TO ERROR HALT ON SEQ ERROR
 CLR PS ;CLEAR PS FROM LAST TEST
 MOV #0,R0 ;LOAD HIGH ORDER WITH 0
 MOV #52525,R1 ;LOAD LOW ORDER WITH 52525
 MOV #51+2,R3 ;SET UP R3
 DIV -(R3),R0 ;DIVIDE BY -(3)
 MOV @#PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS PS=0
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 757 <====

036024 012762 000713 177776
 036032 005262 177774
 036036 000000
 4997 036040 022700 025252 1\$:
 4998 036044 001406

MOV #713,-2(R2) ;MOVE TO MAILBOX # ***** 713 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #25252,R0 ;IS QUOTIENT =25252
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 746 <====

036046 012762 000714 177776
 036054 005262 177774
 036060 000000
 4999 036062 022701 000001 2\$:
 5000 036066 001406

MOV #714,-2(R2) ;MOVE TO MAILBOX # ***** 714 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #1,R1 ;IS REMAINDER =1
 BEQ TST312

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
 : CONDITIONAL BRANCH INST. AND
 : REPLACE THE MOVE INSTRUCTION
 : WHICH FOLLOWS W/ 735 <====

036070 012762 000715 177776
 036076 005262 177774
 036102 000000

MOV #715,-2(R2) ;MOVE TO MAILBOX # ***** 715 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

5001
 5002

.SBTTL DIV USING DMS REG 4
 :MODE 5 REG 4

5003

.SBTTL TEST # 312 - DIV 0 52525 /@-(4)-25252 REM=1 PS=0

 :TEST 312 - DIV 0 52525 /@-(4)=25252 REM=1 PS=0

036104 005212
 036106 022712 000312
 036112 001042
 5004 036114 012700 000000
 5005 036120 012701 052525
 5006 036124 012704 042554
 5007 036130 071054
 5008 036132 013767 177776 004402
 5009 036140 122767 000000 004374
 5010 036146 001406

TST312: INC (R2) ;UPDATE TEST NUMBER
 CMP #312,(R2) ;SEQUENCE ERROR?
 BNE TST313-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R0 ;LOAD HIGH ORDER WITH 0
 MOV #52525,R1 ;LOAD LOW ORDER WITH 52525
 MOV #S2+2,R4 ;SET UP R4
 DIV @-(R4),R0 ;DIVIDE BY @-(4)
 MOV @#PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS PS = 0
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 761 <====

036150 012762 000716 177776
 036156 005262 177774
 036162 000000
 5011 036164 022700 025252 1\$:
 5012 036170 001406

MOV #716,-2(R2) ;MOVE TO MAILBOX # ***** 716 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #25252,R0 ;IS QUOTIENT =25252
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 750 <====

036172 012762 000717 177776
 036200 005262 177774
 036204 000000
 5013 036206 022701 000001 2\$:
 5014 036212 001406

MOV #717,-2(R2) ;MOVE TO MAILBOX # ***** 717 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #1,R1 ;IS REMAINDER =1
 BEQ TST313

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 737 <====

036214 012762 000720 177776
 036222 005262 177774
 036226 000000

MOV #720,-2(R2) ;MOVE TO MAILBOX # ***** 720 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

5015
 5016

:MODE 5 REG 4

.SBTTL TEST # 313 - DIV 0 52525 /@-(R4)=100000 REM=0 PS 2
:*****
:TEST 313 - DIV 0 52525 /@-(R4)=100000 REM=0 PS=2
:*****

036230 005212
036232 022712 000313
036236 001042
5018 036240 012700 052525
5019 036244 012704 042554
5020 036250 012701 000000
5021 036254 071054
5022 036256 013767 177776 004256
5023 036264 122767 000002 004250
5024 036272 001406

TST313: INC (R2) ;UPDATE TEST NUMBER
CMP #313,(R2) ;SEQUENCE ERROR?
BNE TST314-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #52525,R0 ;LOAD HIGH ORDER WITH 52525
MOV #52+2,R4 ;SET UP R4
MOV #0,R1 ;LOAD LOW ORDER WITH 0
DIV @-(R4),R0 ;DIVIDE BY @-(4)
MOV @#PS,SPSW ;SAVE PS
CMPB #2,SPSW ;IS PS=0
BEQ IS

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 761 <====

036274 012762 000721 177776
036302 005262 177774
036306 000000
5025 036310 022700 052525 1\$:
5026 036314 001406

MOV #721,-2(R2) ;MOVE TO MAILBOX # ***** 721 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
1\$: CMP #52525,R0 ;IS QUOTIENT =52525
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 750 <====

036316 012762 000722 177776
036324 005262 177774
036330 000000
5027 036332 022701 000000 2\$:
5028 036336 001406

MOV #722,-2(R2) ;MOVE TO MAILBOX # ***** 722 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;QUOTIENT IS WRONG
2\$: CMP #0,R1 ;IS REMAINDER =0
BEQ TST314

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 737 <

036340 012762 000723 177776
036346 005262 177774
036352 000000

MOV #723,-2(R2) ;MOVE TO MAILBOX # ***** 723 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REMAINDER IS WRONG
: OR SEQUENCE ERROR

5029
5030
5031

.SBTTL DIV USING DM2 REG 7
:MODE 2 REG 7

5032

.SBTTL TEST # 314 - DIV -1 -4/#2= -2 REM=0 PS-10

:TEST 314 - DIV -1 -4/#2= -2 REM=0 PS=10

036354 005212
036356 022712 000314
036362 001041
5033 036364 012700 177777
5034 036370 012701 177774
5035 036374 071027 000002
5036 036400 013767 177776 004134
5037 036406 122767 000010 004126
5038 036414 001406

TST314: INC (R2) ;UPDATE TEST NUMBER
CMP #314,(R2) ;SEQUENCE ERROR?
BNE TST315-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #-1,R0 ;LOAD HIGH ORDER WITH -1
MOV #-4,R1 ;LOAD LOW ORDER WITH -4
DIV #2,R0 ;DIVIDE BY #2
MOV @#PS,SPSW ;SAVE PS
CMPB #10,SPSW ;IS PS =10
BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 762

036416 012762 000724 177776
036424 005262 177774
036430 000000
5039 036432 022700 177776 1\$:
5040 036436 001406

MOV #724,-2(R2) ;MOVE TO MAILBOX # ***** 724 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
CMP #-2,R0 ;IS QUOTIENT = -2
BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 751

036440 012762 000725 177776
036446 005262 177774
036452 000000
5041 036454 022701 000000 2\$:
5042 036460 001406

MOV #725,-2(R2) ;MOVE TO MAILBOX # ***** 725 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;QUOTIENT IS WRONG
CMP #0,R1 ;IS REMAINDER =0
BEQ TST315

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 740

036462 012762 000726 177776
036470 005262 177774
036474 000000

MOV #726,-2(R2) ;MOVE TO MAILBOX # ***** 726 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REMAINDER IS WRONG
; OR SEQUENCE ERROR

5043
5044

;MODE 2 REG 7

5058

.SBTTL TEST # 316 - DIV 0 52525 / #2=25252 REM 1 PS 0

:TEST 316 - DIV 0 52525 / #2=25252 REM=1 PS=0

036620 005212
036622 022712 000316
036626 001041
5059 036630 012700 000000
5060 036634 012701 052525
5061 036640 071027 000002
5062 036644 013767 177776 003670
5063 036652 122767 000000 003662
5064 036660 001406

TST316: INC (R2)
CMP #316,(R2)
BNE TST317-10
MOV #0,R0
MOV #52525,R1
DIV #2,R0
MOV @#PS,SPSW
CMPB #0,SPSW
BEQ 1\$

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:LOAD HIGH ORDER WITH 0
:LOAD LOW ORDER WITH 52525
:DIVIDE BY #2
:SAVE PS
:IS PS =0

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 762 <=====
:

036662 012762 000732 177776
036670 005262 177774
036674 000000
5065 036676 022700 025252 1\$:
5066 036702 001406

MOV #732,-2(R2)
INC -4(R2)
HALT
CMP #25252,R0
BEQ 2\$

:MOVE TO MAILBOX # ***** 732 *****
:SET MSGTYP TO FATAL ERROR
:PS IS WRONG
:IS QUOTIENT =25252

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 751 <=====
:

036704 012762 000733 177776
036712 005262 177774
036716 000000
5067 036720 022701 000001 2\$:
5068 036724 001406

MOV #733,-2(R2)
INC -4(R2)
HALT
CMP #1,R1
BEQ TST317

:MOVE TO MAILBOX # ***** 733 *****
:SET MSGTYP TO FATAL ERROR
:QUOTIENT IS WRONG
:IS REMAINDER =1

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 740 <=====
:

036726 012762 000734 177776
036734 005262 177774
036740 000000

MOV #734,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 734 *****
:SET MSGTYP TO FATAL ERROR
:REMAINDER IS WRONG
: OR SEQUENCE ERROR

5069
5070

:MODE 2 REG 7

5071

.SBTTL TEST # 317 - DIV 25253 0 / #125252-100000 REM-0 PS-10

 :TEST 317 - DIV 25253 0 / #125252=100000 REM=0 PS 10

036742 005212
 036744 022712 000317
 036750 001041
 5072 036752 012700 025253
 5073 036756 012701 000000
 5074 036762 071027 125252
 5075 036766 013767 177776 003546
 5076 036774 122767 000010 003540
 5077 037002 001406

TST317: INC (R2) ;UPDATE TEST NUMBER
 CMP #317,(R2) ;SEQUENCE ERROR?
 BNE TST320-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #25253,R0 ;LOAD HIGH ORDER WITH 25253
 MOV #0,R1 ;LOAD LOW ORDER WITH 0
 DIV #125252,R0 ;DIVIDE BY #125252
 MOV @#PS,SPSW ;SAVE PS
 CMPB #10,SPSW ;IS PS =10
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 762 <

037004 012762 000735 177776
 037012 005262 177774
 037016 000000
 5078 037020 022700 100000 1\$:
 5079 037024 001406

MOV #735,-2(R2) ;MOVE TO MAILBOX # ***** 735 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #100000,R0 ;IS QUOTIENT =100000
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 751 <

037026 012762 000736 177776
 037034 005262 177774
 037040 000000
 5080 037042 022701 000000 2\$:
 5081 037046 001406

MOV #736,-2(R2) ;MOVE TO MAILBOX # ***** 736 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #0,R1 ;IS REMAINDER =0
 BEQ TST320

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <-
 : WHICH FOLLOWS W/ 740 <

037050 012762 000737 177776
 037056 005262 177774
 037062 000000

MOV #737,-2(R2) ;MOVE TO MAILBOX # ***** 737 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

5082

;MODE 2 REG 7

5083

.SBTTL TEST # 320 - DIV -1 -1 / #-1-1 REM=0 PS=0

 :TEST 320 - DIV -1 -1 / #-1=1 REM=0 PS=0

037064 005212
 037066 022712 000320
 037072 001041
 5084 037074 012700 177777
 5085 037100 012701 177777
 5086 037104 071027 177777
 5087 037110 013767 177776 003424
 5088 037116 122767 000000 003416
 5089 037124 001406

TST320: INC (R2) ;UPDATE TEST NUMBER
 CMP #320,(R2) ;SEQUENCE ERROR?
 BNE TST321-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #-1,R0 ;LOAD HIGH ORDER WITH -1
 MOV #-1,R1 ;LOAD LOW ORDER WITH -1
 DIV #-1,R0 ;DIVIDE BY #-1
 MOV @#PS,SPSW ;SAVE PS
 CMPB #0,SPSW ;IS PS =0
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 762 <

037126 012762 000740 177776
 037134 005262 177774
 037140 000000
 5090 037142 022700 000001 1\$:
 5091 037146 001406

MOV #740,-2(R2) ;MOVE TO MAILBOX # ***** 740 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 CMP #1,R0 ;IS QUOTIENT =1
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < --
 : CONDITIONAL BRANCH INST. AND <---=
 : REPLACE THE MOVE INSTRUCTION <---=
 : WHICH FOLLOWS W/ 751 <---=

037150 012762 000741 177776
 037156 005262 177774
 037162 000000
 5092 037164 022701 000000 2\$:
 5093 037170 001406

MOV #741,-2(R2) ;MOVE TO MAILBOX # ***** 741 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 CMP #0,R1 ;IS REMAINDER =0
 BEQ TST321

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 740 <---=

037172 012762 000742 177776
 037200 005262 177774
 037204 000000

MOV #742,-2(R2) ;MOVE TO MAILBOX # ***** 742 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

5094
 5095

;MODE 2 REG 7

5096

.SBTTL TEST # 321 - DIV 0 177777/#0=0 REM=0 PS=3
:*****
:TEST 321 - DIV 0 177777/#0=0 REM=0 PS=3
:*****

037206 005212
037210 022712 000321
037214 001022
5097 037216 012704 177777
5098 037222 012705 000000
5099 037226 071427 000000
5100 037232 013767 177776 003302
5101 037240 042767 000014 003274
5102 037246 122767 000003 003266
5103 037254 001406

TST321: INC (R2) ;UPDATE TEST NUMBER
CMP #321,(R2) ;SEQUENCE ERROR?
BNE TST322-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #177777,R4 ;LOAD HIGH ORDER WITH 177777
MOV #0,R5 ;LOAD LOW ORDER WITH 0
DIV #0,R4 ;DIVIDE BY #2
MOV @#PS,SPSW ;SAVE PS
BIC #14,SPSW
CMPB #3,SPSW ;IS PS =3
BEQ TST322

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 757 <====

037256 012762 000743 177776
037264 005262 177774
037270 000000

MOV #743,-2(R2) ;MOVE TO MAILBOX # ***** 743 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
; OR SEQUENCE ERROR

5104
5105

:MODE 2 REG 7

5106

.SBTTL TEST # 322 - DIV 0 100000 00000 /#100000 REM=0 PS=2
:*****
:TEST 322 - DIV 0 100000 00000 /#100000 REM=0 PS=2
:*****

037272 005212
037274 022712 000322
037300 001022
5107 037302 012704 100000
5108 037306 012705 000000
5109 037312 071427 000002
5110 037316 013767 177776 003216
5111 037324 042767 000014 003210
5112 037332 122767 000002 003202
5113 037340 001406

TST322: INC (R2) ;UPDATE TEST NUMBER
CMP #322,(R2) ;SEQUENCE ERROR?
BNE TST323-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #100000,R4 ;LOAD HIGH ORDER WITH 100000
MOV #0,R5 ;LOAD LOW ORDER WITH 0
DIV #2,R4 ;DIVIDE BY #2
MOV @#PS,SPSW ;SAVE PS
BIC #14,SPSW
CMPB #2,SPSW ;IS PS=2
BEQ TST323
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 757
: MOVE TO MAILBOX # ***** 744 *****
: SET MSGTYP TO FATAL ERROR
: PS IS WRONG
: OR SEQUENCE ERROR

037342 012762 000744 177776
037350 005262 177774
037354 000000

MOV #744,-2(R2)
INC -4(R2)
HALT

5114
5115

:MODE 2 REG 7

5116

.SBTTL TEST # 323 - DIV 0 77777/#0=0 REM=0 PS=3

 :TEST 323 - DIV 0 77777/#0=0 REM=0 PS=3

037356 005212
 037360 022712 000323
 037364 001061
 5117 037366 012704 000000
 5118 037372 012705 077777
 5119 037376 071427 000000
 5120 037402 013767 177776 003132
 5121 037410 042767 000014 003124
 5122 037416 122767 000003 003116
 5123 037424 001434

TST323: INC (R2) ;UPDATE TEST NUMBER
 CMP #323,(R2) ;SEQUENCE ERROR?
 BNE TST324-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #0,R4 ;LOAD HIGH ORDER WITH 0
 MOV #77777,R5 ;LOAD LOW ORDER WITH 77777
 DIV #0,R4 ;DIVIDE BY 0
 MOV @#PS,SPSW ;SAVE PS
 BIC #14,SPSW
 CMPB #3,SPSW ;IS PS =3
 BEQ 2\$

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 757 <====

037426 012762 000745 177776
 037434 005262 177774
 037440 000000
 5124
 5125 037442 012704 000000
 5126 037446 012705 077777
 5127 037452 071427 000000
 5128 037456 013767 177776 003056
 5129 037464 042767 000014 003050
 5130 037472 122767 000003 003042
 5131 037500 001417

MOV #745,-2(R2) ;MOVE TO MAILBOX # ***** 745 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 MOV #0,R4 ;LOAD HIGH ORDER WITH 0
 MOV #77777,R5 ;LOAD LOW ORDER WITH 77777
 DIV #0,R4 ;DIVIDE BY 0
 MOV @#PS,SPSW ;SAVE PS
 BIC #14,SPSW
 CMPB #3,SPSW ;IS PS =3
 BEQ TST324

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 ; CONDITIONAL BRANCH INST. AND <====
 ; REPLACE THE MOVE INSTRUCTION <====
 ; WHICH FOLLOWS W/ 731 <====

037502 012762 000746 177776
 037510 005262 177774
 037514 000000

MOV #746,-2(R2) ;MOVE TO MAILBOX # ***** 746 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 ; OR SEQUENCE ERROR

5132
 5133 037516 022705 077777 2\$:
 5134 037522 001406

CMP #77777,R5 ;IS REMAINDER =1
 BEQ TST324

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 ; CONDITIONAL BRANCH INST. AND <
 ; REPLACE THE MOVE INSTRUCTION <
 ; WHICH FOLLOWS W/ 720 <

037524 012762 000747 177776
 037532 005262 177774
 037536 000000

MOV #747,-2(R2) ;MOVE TO MAILBOX # ***** 747 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

5135
 5136

;MODE 2 REG 7

513

.SBTTL TEST # 324 - DIV 25253 0 / #125252=100000 REM=0 PS 10

 :TEST 324 - DIV 25253 0 / #125252=100000 REM=0 PS=10

037540 005212
 037542 022712 000324
 037546 001041
 5138 037550 012700 025253
 5139 037554 012701 000000
 5140 037560 071027 125252
 5141 037564 013767 177776 002750
 5142 037572 122767 000010 002742
 5143 037600 001406

TST324: INC (R2) ;UPDATE TEST NUMBER
 CMP #324,(R2) ;SEQUENCE ERROR?
 BNE TST325-10 ;BR TO ERROR HALT ON SEQ ERROR
 MOV #25253,R0 ;LOAD HIGH ORDER WITH 25253
 MOV #0,R1 ;LOAD LOW ORDER WITH 0
 DIV #125252,R0 ;DIVIDE BY #125252
 MOV @#PS,SPSW ;SAVE PS
 CMPB #10,SPSW ;IS PS =10
 BEQ 1\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
 : CONDITIONAL BRANCH INST. AND <
 : REPLACE THE MOVE INSTRUCTION <
 : WHICH FOLLOWS W/ 762 <

037602 012762 000750 177776
 037610 005262 177774
 037614 000000
 5144 037616 022700 100000 1\$:
 5145 037622 001406

MOV #750,-2(R2) ;MOVE TO MAILBOX # ***** 750 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;PS IS WRONG
 1\$: CMP #100000,R0 ;IS QUOTIENT =100000
 BEQ 2\$

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 751 <====

037624 012762 000751 177776
 037632 005262 177774
 037636 000000
 5146 037640 022701 000000 2\$:
 5147 037644 001406

MOV #751,-2(R2) ;MOVE TO MAILBOX # ***** 751 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;QUOTIENT IS WRONG
 2\$: CMP #0,R1 ;IS REMAINDER =0
 BEQ TST325

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
 : CONDITIONAL BRANCH INST. AND <====
 : REPLACE THE MOVE INSTRUCTION <====
 : WHICH FOLLOWS W/ 740 <====

037646 012762 000752 177776
 037654 005262 177774
 037660 000000

MOV #752,-2(R2) ;MOVE TO MAILBOX # ***** 752 *****
 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
 HALT ;REMAINDER IS WRONG
 ; OR SEQUENCE ERROR

5148

;MODE 2 REG 7

5149

.SBTTL TEST # 325 - DIV -1 -1 /#-1=1 REM=0 PS=0

:TEST 325 - DIV -1 -1 /#-1=1 REM=0 PS=0

037662 005212
037664 022712 000325
037670 001041
5150 037672 012700 177777
5151 037676 012701 177777
5152 037702 071027 177777
5153 037706 013767 177776 002626
5154 037714 122767 000000 002620
5155 037722 001406

TST325: INC (R2)
CMP #325,(R2)
BNE TST326-10
MOV #-1,R0
MOV #-1,R1
DIV #-1,R0
MOV @#PS,SPSW
CMPB #0,SPSW
BEQ 1\$

:UPDATE TEST NUMBER
:SEQUENCE ERROR?
:BR TO ERROR HALT ON SEQ ERROR
:LOAD HIGH ORDER WITH -1
:LOAD LOW ORDER WITH -1
:DIVIDE BY #-1
:SAVE PS
:IS PS =0

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 762 <

037724 012762 000753 177776
037732 005262 177774
037736 000000
5156 037740 022700 000001 1\$:
5157 037744 001406

MOV #753,-2(R2)
INC -4(R2)
HALT
CMP #1,R0
BEQ 2\$

:MOVE TO MAILBOX # ***** 753 *****
:SET MSGTYP TO FATAL ERROR
:PS IS WRONG
:IS QUOTIENT =1

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 751 <

037746 012762 000754 177776
037754 005262 177774
037760 000000
5158 037762 022701 000000 2\$:
5159 037766 001406

MOV #754,-2(R2)
INC -4(R2)
HALT
CMP #0,R1
BEQ TST326

:MOVE TO MAILBOX # ***** 754 *****
:SET MSGTYP TO FATAL ERROR
:QUOTIENT IS WRONG
:IS REMAINDER =0

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS
: CONDITIONAL BRANCH INST. AND
: REPLACE THE MOVE INSTRUCTION
: WHICH FOLLOWS W/ 740 <====

037770 012762 000755 177776
037776 005262 177774
040002 000000

MOV #755,-2(R2)
INC -4(R2)
HALT

:MOVE TO MAILBOX # ***** 755 *****
:SET MSGTYP TO FATAL ERROR
:REMAINDER IS WRONG
: OR SEQUENCE ERROR

5160

:MODE 2 REG 7

5169

.SBTTL TEST # 326 - DIV 0 177777/#0=0 REM=0 PS=3

:TEST 326 - DIV 0 177777/#0=0 REM=0 PS=3

040004 005212
040006 022712 000326
040012 001022
5162 040014 012704 177777
5163 040020 012705 000000
5164 040024 071427 000000
5165 040030 013767 177776 002504
5166 040036 042767 000014 002476
5167 040044 122767 000003 002470
5168 040052 001406

TST326: INC (R2) ;UPDATE TEST NUMBER
CMP #326,(R2) ;SEQUENCE ERROR?
BNE TST327-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #177777,R4 ;LOAD HIGH ORDER WITH 177777
MOV #0,R5 ;LOAD LOW ORDER WITH 0
DIV #0,R4 ;DIVE BY #2
MOV @#PS,SPSW ;SAVE PS
BIC #14,SPSW
CMPB #3,SPSW ;IS PS =3
BEQ TST327

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 757 <====

040054 012762 000756 177776
040062 005262 177774
040066 000000

MOV #756,-2(R2) ;MOVE TO MAILBOX # ***** 756 *****
INC -4(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;PS IS WRONG
; OR SEQUENCE ERROR

5169

:MODE 2 REG 7

5170

.SBTTL TEST # 327 - DIV 0 100000 000000 /#2=100000 REM=0 PS=2
:.....
:TEST 327 - DIV 0 100000 000000 /#2=100000 REM=0 PS=2
:.....

040070 005212
040072 022712 000327
040076 001022
5171 040100 012704 100000
5172 040104 012705 000000
5173 040110 071427 000002
5174 040114 013767 177776 002420
5175 040122 042767 000014 002412
5176 040130 122767 000002 002404
5177 040136 001406

TST327: INC (R2) ;UPDATE TEST NUMBER
CMP #327,(R2) ;SEQUENCE ERROR?
BNE TST330-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #100000,R4 ;LOAD HIGH ORDER WITH 100000
MOV #0,R5 ;LOAD LOW ORDER WITH 0
DIV #2,R4 ;DIVIDE BY #2
MOV @#PS,SPSW ;SAVE PS
BIC #14,SPSW
CMPB #2,SPSW ;IS PS=2
BEQ TST330

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--->
: CONDITIONAL BRANCH INST. AND <--->
: REPLACE THE MOVE INSTRUCTION <--->
: WHICH FOLLOWS w/ 757 <--->
: MOVE TO MAILBOX # ***** 757 *****
: SET MSGTYP TO FATAL ERROR
: PS IS WRONG
: OR SEQUENCE ERROR

040140 012762 000757 177776
040146 005262 177774
040152 000000

MOV #757,-2(R2)
INC -4(R2)
HALT

5178

:MODE 2 REG 7

5179

.SBTTL TEST # 330 - DIV 0 77777 /#0=0 REM=0 PS=3

:TEST 330 - DIV 0 77777 /#0=0 REM=0 PS=3

040154 005212
040156 022712 000330
040162 001022
5180 040164 012704 000000
5181 040170 012705 077777
5182 040174 071427 000000
5183 040200 013767 177776 002334
5184 040206 042767 000014 002326
5185 040214 122767 000003 002320
5186 040222 001406

TST330: INC (R2) ;UPDATE TEST NUMBER
CMP #330,(R2) ;SEQUENCE ERROR?
BNE TST331-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #0,R4 ;LOAD HIGH ORDER WITH 0
MOV #77777,R5 ;LOAD LOW ORDER WITH 77777
DIV #0,R4 ;DIVIDE BY 0
MOV @#PS,SPSW ;SAVE PS
BIC #14,SPSW
CMPB #3,SPSW ;IS PS =3
BEQ TST331

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
: CONDITIONAL BRANCH INST. AND <
: REPLACE THE MOVE INSTRUCTION <
: WHICH FOLLOWS W/ 757 <
: MOVE TO MAILBOX # ***** 760 *****
: SET MSGTYP TO FATAL ERROR
: PS IS WRONG
: OR SEQUENCE ERROR

040224 012762 000760 177776 MOV #760,-2(R2)
040232 005262 177774 INC -4(R2)
040236 000000 HALT

5187

.SBTTL TEST # 331 - TEST SUB INSTRUCTION MODES 2-7

 :TEST 331 - TEST SUB INSTRUCTION MODES 2-7

040240	005212			TST331: INC	(R2)	:UPDATE TEST NUMBER
040242	022712	000331		CMP	#331,(R2)	:SEQUENCE ERROR?
040246	001113			BNE	TST332-10	:BR TO ERROR HALT ON SEQ ERROR
5188 040250	012700	177777		MOV	#-1,R0	:SETUP SOURCE TO -1
5189 040254	010037	040336		MOV	R0,@#2\$+2	:SET MODE 2 TO LOCATION TO A -1
5190 040260	012701	042546		MOV	#SUBT+2,R1	:SET R1 TO POINT TO BUFFER AREA +2
5191 040264	012761	177777	177776	MOV	#-1,-2(R1)	:PUT -1 OP INTO DEST. BUFFER AREA
5192 040272	160041			SUB	R0,-(R1)	:MODE-4 SUB SOURCE MODE 0 DEST MODE 4
5193 040274	001406			BEQ	1\$	
						: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
						CONDITIONAL BRANCH INST. AND <
						REPLACE THE MOVE INSTRUCTION <
						WHICH FOLLOWS W/ 764 <
						:MOVE TO MAILBOX # ***** 761 *****
040276	012762	000761	177776	MOV	#761,-2(R2)	
040304	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
040310	000000			HALT		:SUB -1 FROM -1 FAILED TO GIVE ZERO RESULT IN MODE 4
5194 040312	022701	042544	1\$:	CMP	#SUBT,R1	:VARYIFY AUTO-DECREMENT OF R1 IN MODE 4 WORKED
5195 040316	001406			BEQ	2\$	
						: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS < --
						CONDITIONAL BRANCH INST. AND <=
						REPLACE THE MOVE INSTRUCTION <
						WHICH FOLLOWS W/ 753 <
						:MOVE TO MAILBOX # ***** 762 *****
040320	012762	000762	177776	MOV	#762,-2(R2)	
040326	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
040332	000000			HALT		:R1 FAILED TO DECEREMENT IN MODE 4
5196 040334	160027	177777	2\$:	SUB	R0,#-1	:IF PROC HALTS AT PC=32650 THEN AUTO INC MODE FAILED
5197 040340	001406			BEQ	3\$	
						: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <--
						CONDITIONAL BRANCH INST. AND <=
						REPLACE THE MOVE INSTRUCTION <
						WHICH FOLLOWS W/ 742 <
						:MOVE TO MAILBOX # ***** 763 *****
040342	012762	000763	177776	MOV	#763,-2(R2)	
040350	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
040354	000000			HALT		:SUB -1 FROM -1 FAILED TO GIVE ZERO RESULT
5198 040356	012711	177777	3\$:	MOV	#-1,(R1)	:RESET SUBT TO A -1
5199 040362	160037	042544		SUB	R0,@#SUBT	:SET MODE-3
5200 040366	001406			BEQ	4\$	
						: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
						CONDITIONAL BRANCH INST. AND <
						REPLACE THE MOVE INSTRUCTION <
						WHICH FOLLOWS W/ 727 <
						:MOVE TO MAILBOX # ***** 764 *****
040370	012762	000764	177776	MOV	#764,-2(R2)	
040376	005262	177774		INC	-4(R2)	:SET MSGTYP TO FATAL ERROR
040402	000000			HALT		:SUB -1 FROM -1 FAILED TO GIVE A ZERO RESULT
5201 040404	012703	042550	4\$:	MOV	#SUBT+4,R3	:SETUP FOR MODE 5
5202 040410	012711	177777		MOV	#-1,(R1)	:RESET SUBT TO A 1-
5203 040414	160053			SUB	R0,@-(R3)	:SUB -1 FROM -1 USING MODE-5
5204 040416	001406			BEQ	5\$	
						: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
						CONDITIONAL BRANCH INST. AND <
						REPLACE THE MOVE INSTRUCTION <
						WHICH FOLLOWS W/ 713 <
						:MOVE TO MAILBOX # ***** 765 *****
040420	012762	000765	177776	MOV	#765,-2(R2)	

	040426	005262	177774		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
	040432	000000			HALT			:SUB -1 FROM -1 FAILED TO GIVE ZERO RESULT
5205	040434	012711	177777	5\$:	MOV	#-1,(R1)		:RESET SUBT TO A -1
5206	040440	160061	000000		SUB	R0,0(R1)		:MODE-6
5207	040444	001406			BEQ	6\$: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=
								: CONDITIONAL BRANCH INST. AND <- -
								: REPLACE THE MOVE INSTRUCTION <- -
								: WHICH FOLLOWS W/ 700 <== -
	040446	012762	000766	177776	MOV	#766,-2(R2)		:MOVE TO MAILBOX # ***** 766 *****
	040454	005262	177774		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
	040460	000000			HALT			:SUB -1 FROM -1 FAILED TO GIVE ZERO RESULT
5208	040462	010011		6\$:	MOV	R0,(R1)		:RESET SUBT TO A -1
5209	040464	160071	000002		SUB	R0,@2(R1)		:MODE-7
5210	040470	001406			BEQ	TST332		: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <
								: CONDITIONAL BRANCH INST. AND <-
								: REPLACE THE MOVE INSTRUCTION <-
								: WHICH FOLLOWS W/ 666 <- -
	040472	012762	000767	177776	MOV	#767,-2(R2)		:MOVE TO MAILBOX # ***** 767 *****
	040500	005262	177774		INC	-4(R2)		:SET MSGTYP TO FATAL ERROR
	040504	000000			HALT			:SUB -1 FROM -1 FAILED TO GIVE ZERO RESULT
								: OR SEQUENCE ERROR

5211

```

.SBTTL TEST # 332 - TEST RTI KERNAL/USER MODE MICRO FLOW
:*****
:TEST 332 - TEST RTI KERNAL/USER MODE MICRO FLOW
:*****
TST332: INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #332,(R2)    ;SEQUENCE ERROR?
        BNE     TST333-10    ;BR TO ERROR HALT ON SEQ ERROR
        MOV     #STBOT,R6    ;SETUP KERNAL STACK
        MOV     #USRM,PS     ;SETUP USER MODE
        MOV     #USTBOT,R6   ;SETUP USER STACK POINTER
        MOV     #USRM,-(SP)  ;PUSH USER R6 ON STACK
        MOV     #REN,-(SP)   ;SETUP RETURN ADDRESS
        RTI
5218 040546 022767 140000 137222 REN:  CMP     #140000,PS    ;CHECK PSW
5219 040554 001411          BEQ     1$
5220 040556 042767 140000 137212 BIC     #USRM,PS      ;CLEAR USER MODE
5221 040564 012762 000770 177776 MOV     #770,-2(R2)   ;MOVE TO MAILBOX # ***** 770 *****
        INC     -4(R2)     ;SET MSGTYP TO FATAL ERROR
        HALT          ;USER MODE DID NOT SET
5222 040600 042767 140000 137170 1$:  BIC     #USRM,PS      ;SET KERNAL MODE
5223 040606 012746 000340          MOV     #340,-(SP)   ;SET LEVEL 7
5224 040612 012746 040634          MOV     #TST333,-(SP)
        RTI
5226 040620 012762 000771 177776 MOV     #771,-2(R2)   ;MOVE TO MAILBOX # ***** 771 *****
        INC     -4(R2)     ;SET MSGTYP TO FATAL ERROR
        HALT          ;RTI DID NOT HAPPEN
    
```

5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
523
523
524
5241
5242
5243
5244
5245

```

:*****
:* THIS TEST IS OPTIONAL AND IS SELECTED BY SETTING MFM HARDWARE
:* SWITCH REGISTER BIT 08 TO A 1 IN THE CASE OF STANDALONE OPERATION
:* OF THE DIAGNOSTIC. IN THE CASE OF MANUFACTURING APT RUNTIME MODE
:* THEN BIT 08 OF $SWREG IS SET TO 1 THRU APT SCRIPTING.
:*
:* TEST DESCRIPTION:
:* THIS TEST VERIFIES THAT THE SWITCH REGISTER SELECT SIGNAL IS
:* ISSUED FROM THE CPU DATA PATH MODULE TO THE MFM MODULE. WHEN
:* THE HARDWARE SWITCH REGISTER IS READ, THE ADDRESS OF THE SWITCH
:* REGISTER IS DECODED ON THE CPU DATA PATH CAUSING THE 'SR SELECT'
:* SIGNAL TO BE ISSUED TO THE MFM. THE MFM USES THIS SIGNAL TO PLACE
:* THE REGISTER CONTENTS ONTO THE PAX DATA BUS. CONTENTS OF THE SWITCH
:* REGISTER OTHER THAN ALL 1'S SHOULD BE READ.(THERE IS A
:* HI PROBABILITY THAT IF THE 'SR SELECT' SIGNAL IS NOT ISSUED
:* ALL 1'S WILL BE READ ON PAX DATA.) IT IS ASSUMED THAT DATA OTHER
:* THAN ALL 1'S IS LOCATED IN HARDWARE SWITCH REGISTER .
    
```

5246

.SBTTL TEST # 333 - SWITCH REGISTER SELECT TEST

 :TEST 333 - SWITCH REGISTER SELECT TEST

040634 005212
 040636 022712 000333
 040642 001022
 5247
 5248 040644 132767 000200 137447
 5249 040652 001405
 5250 040654 032767 000400 137440
 5251
 5252 040662 001413
 5253 040664 000404
 5254 040666 032737 000400 177570 4\$:
 5255
 5256 040674 001406
 5257
 5258 040676 013700 177570 1\$:
 5259
 5260 040702 022700 177777
 5261 040706 001001
 5262 040710 000000
 5263
 5264
 5265
 5266
 5267
 5268
 5269 040712 000240 10\$:
 5270 040714 000240
 5271 040716 000240
 5272
 5273
 5274
 5275
 5276
 5277
 5278
 5279
 5280
 5281
 5282
 5283
 5284
 5285
 5286
 5287
 5288
 5289
 5290
 5291
 5292
 5293
 5294
 5295

```
TST333: INC      (R2)           ;UPDATE TEST NUMBER
          CMP      #333,(R2)    ;SEQUENCE ERROR?
          BNE      TST334-10    ;BR TO ERROR HALT ON SEQ ERROR

          BITB     #200,$ENVM    ;IS APT SIZING?
          BEQ      4$           ;NO;TRY HARDWARE SWITCH REGISTER
          BIT      #400,$SWREG  ;YES APT IS SIZING;DOES APT SAY TO DO
                                ;THIS TEST
          BEQ      10$          ;NO;SKIP TEST
          BR       1$           ;YES,DO TEST
          BIT      #400,@#177570 ;DOES HARDWARE SWITCH REGISTER SAY TO
                                ;TO DO TEST?
          BEQ      10$          ;NO,SKIP TEST

          MOV      @#177570,R0   ;READ HARDWARE SWITCH REGISTER CONTENTS
                                ;AND SAVE IN R0
          CMP      #-1,R0       ;WERE ALL 1'S RECEIVED
          BNE      10$          ;NO,TEST PASSES
          HALT                ;'SR SELECT' ERROR OR SEQUENCE ERROR
                                ;ALL 1'S WERE RECEIVED WHEN HARDWARE SWITCH
                                ;REGISTER WAS READ. THIS INDICATES THAT THE
                                ;THE SWITCH REGISTER CONTENTS WAS NOT PUT ON
                                ;THE PAX DATA LINES AS A RESULT OF THE 'SR SELECT'
                                ;SIGNAL BEING RECEIVED BY THE MFM FROM THE
                                ;CPU DATA PATH MODULE
                                ;END OF TEST

          NOP
          NOP
          NOP
```

```
*****
: THIS TEST IS OPTIONAL AND IS SELECTED BY SETTING MFM HARDWARE
: SWITCH REGISTER BIT 08 TO A 1 IN THE CASE OF STANDALONE OPERATION
: OF THE DIAGNOSTIC. IN THE CASE OF MANUFACTURING APT RUNTIME MODE
: THEN BIT 08 OF $SWREG IS SET TO 1 THRU APT SCRIPTING.

: THIS TEST VERIFIES THE CACHE RESTART SIGNAL ON THE CPU CONTROL
: MODULE ISSUED FROM THE CACHE.

: THIS TEST ASSUMES THAT ALL MODULES EXCEPT THE CPU DATA PATH/CONTROL
: STORE ARE KNOWN GOOD MODULES.

: THIS TEST TOGETHER WITH OTHER CACHE TESTS,ALLOW MFG. TO ELIMINATE
: HAVING TO RUN THE CACHE DIAGNOSTIC DURING QUICK
: VERIFY TESTING OF THE CPU MODULE.
: TEST DESCRIPTION:
: VERIFY THAT A CACHE READ HIT WILL RESULT IN DATA BEING READ
: FROM CACHE DATA STORE, ASSURING THAT THE CACHE HAS ISSUED A
: A CPU CLOCK RESTART SIGNAL. ASSURE THAT ALL 0'S CAN BE CACHED
: OUT OF CACHE DATA STORE.
:*
```

5296

.SBTTL TEST # 334 - CACHE RESTART SIGNAL TEST

 :TEST 334 - CACHE RESTART SIGNAL TEST

5297	040720	005212				TST334: INC	(R2)		:UPDATE TEST NUMBER
5298	040722	022712	000334			CMP	#334,(R2)		:SEQUENCE ERROR?
5299	040726	001115				BNE	TST335-10		:BR TO ERROR HALT ON SEQ ERROR
5300	040730	132767	000200	137363		BITB	#200,\$ENVM		:IS APT SIZING?
5301	040736	001405				BEQ	4\$:NO;TRY HARDWARE SWITCH REGISTER
5302	040740	032767	000400	137354		BIT	#400,\$SWREG		:YES APT IS SIZING;DOES APT SAY TO DO
5303	040746	001506				BEQ	10\$:THIS TEST
5304	040750	000404				BR	1\$:NO;SKIP TEST
5305	040752	032737	000400	177570	4\$:	BIT	#400,@#177570		:YES,DO TEST
5306	040760	001501				BEQ	10\$:DOES HARDWARE SWITCH REGISTER SAY TO
5307	040762				1\$:				:TO DO TEST?
5308	040762	012701	040000			MOV	#40000,R1		:NO,SKIP TEST
5309	040766	012705	060000			MOV	#60000,R5		:ADDRESS 40000 TO R1
5310	040772	012737	041056	000014		MOV	#3\$,@#14		:ADDRESS 60000 TO R5
5311	041000	012737	000340	000016		MOV	#340,@#16		:SETUP BPT TRAP VECTORS
5312	041006	005003				CLR	R3		:CLEAR ERROR FLAGS
5313	041010	005004				CLR	R4		
5314	041012	012706	060002			MOV	#60002,R6		:STACK POINTER NOW POINTS TO ADDRESS 60002
5315	041016	005037	060000			CLR	@#60000		:PRECONDITION MAIN MEMORY ADDRESS LOCATION
5316	041022	012767	000340	136746		MOV	#340,PS		:60000 WITH ALL 0'S
5317	041030	112737	000002	177750		MOVB	#2,@#177750		:PRECONDITION PS TO 340
5318	041036	012737	000011	177746		MOV	#11,@#177746		: 'HIT ON DESTINATION ONLY' (HODO) ALLOWS
5319	041044	000257				CCC			:CACHE UPDATES AND HITS
5320	041046	005711				TST	(R1)		:ONLY DURING THE DESTINATION MEMORY ACCESS
5321	041050	005715				TST	(R5)		:OF AN INSTRUCTION.
5322	041052	000003				BPT			:NO BYPASS TO ALLOW WRITES TO CACHE STORES.
5323	041054	000000				HALT			:ENABLE LOW CACHE
5324	041056	042737	000002	177750	3\$:	BIC	#2,@#177750		:CLEAR ALL CONDITION CODES
5325	041064	011500				MOV	(R5),R0		:CACHE READ UPDATE. WRITE ALL 0'S FROM
5326									:MAIN MEMORY LOCATION TO CACHE DATA STORE
5327									:LOCATION 0000.
5328									:BREAKPOINT TRAP. DUE TO A TRAP,THE PSW
5329									:WILL BE WRITTEN TO THE STACK, WHICH NOW
5330									:POINTS TO ADDRESS 60000.THE TRAP INSTRUCTION
5331									:IS A NON-DESTINATION ACCESS INSTR.THEREFORE,
5332									:SINCE HODO IS BEING USED, A CACHE UPDATE
5333									:WILL BE INHIBITED. MAIN MEMORY
5334									:ADDRESS 60000 WILL CONTAIN PSW DATA OF 344,AND
5335									:THE LOCATION IN CACHE CORRESPONDING TO ADDRESS
5336									:60000 WILL BE LEFT WITH ALL 0'S DATA.
5337									
5338									
5339									
5340									:BPT TRAP DID NOT OCCUR
5341									
5342									:TRAP TO HERE;DISABLE HODO
5343									: WHEN THIS INSTRUCTION READS
5344									:ADDRESS 60000
5345									:A CACHE READ HIT SHOULD RESULT AND A CPU CLOCK
5346									:RESTART SIGNAL SHOULD BE ISSUED.

```

5347                                     ;THE CPU SHOULD READ DATA FROM CACHE DATA STORE
5348                                     ;RATHER THAN MAIN MEMORY.
5349 041066 000240                       NOP
5350 041070 000240                       NOP
5351 041072 005700                       TST      R0                ;R0 SHOULD CONTAIN ALL 0'S
5352 041074 001406                       BEQ     7$
5353 041076 022700 000344               CMP     #344,R0           ;DID THE CPU READ MAIN MEMORY?
5354 041102 001002                       BNE    6$                ;NO,MUST HAVE READ CACHE BUT BAD DATA WAS RECEIVED.
5355 041104 005203                       INC     R3                ;INDICATE ERROR THAT MAIN MEMORY WAS READ
5356 041106 000401                       BR     7$
5357 041110 005204                       6$:   INC     R4                ;INDICATE ERROR THAT DATA WAS CACHED BUT
5358                                     ;BAD DATA WAS RECEIVED.
5359 041112 105037 177750 177750 7$:   CLRB   @#177750           ;DISABLE MAINTENANCE MODE
5360 041116 012737 000000 177716       MOV     #0,@#177746       ;TURN ON CACHE
5361 041124 012706 001000               MOV     #STBOT,R6        ;RESET STACK POINTER
5362 041130 012737 042432 000014       MOV     #T014,@#14      ;RESTORE BPT VECTORS
5363 041136 005037 000016               CLR     @#16
5364 041142 005703                       TST     R3                ;WAS MAIN MEMORY READ?
5365 041144 001402                       BEQ     8$                ;NO
5366 041146 000000                       HALT
5367                                     ;ERROR
5368                                     ;CPU CLOCK RESTART-CACHED DATA TESTS
5369                                     ;ATTEMPTING TO CAUSE A READ HIT IN ORDER TO
5370                                     ;CACHE DATA RESULTED IN MAIN MEMORY BEING READ
5371 041150 000405                       BR     10$
5372 041152 005704                       8$:   TST     R4                ;NEXT TEST
5373 041154 001403                       BEQ     10$              ;WAS BAD DATA CACHED FROM CACHE DATA STORE?
5374 041156 000000                       HALT
5375                                     ;NO, NEXT TEST
5376                                     ;ERROR
5377                                     ;CPU CLOCK RESTART-CACHED DATA TESTS
5378                                     ;CREATING A READ HIT BY READING ADDRESS 60000
5379 041160 000401                       BR     10$              ;CACHED BAD DATA FROM CACHE DATA STORE
5380 041162 000000                       HALT
5381 041164 000402                       10$:  BR     .+6
5382 041166 000240                       NOP
5383 041170 000240                       NOP
5384
5385                                     ;*****
5386                                     ;* THIS TEST IS OPTIONAL AND IS SELECTED BY SETTING MFM HARDWARE
5387                                     ;* SWITCH REGISTER BIT 08 TO A 1 IN THE CASE OF STANDALONE OPERATION
5388                                     ;* OF THE DIAGNOSTIC. IN THE CASE OF MANUFACTURING APT RUNTIME MODE
5389                                     ;* THEN BIT 08 OF $SWREG IS SET TO 1 THRU APT SCRIPTING.
5390
5391                                     ;*
5392                                     ;* THIS TEST VERIFIES
5393                                     ;* THE CPU CONTROL SIGNAL TO THE CACHE WHICH INDICATES THAT AN ACCESS
5394                                     ;* TO THE UNIBUS IS BEING PERFORMED(PA TOP 128K L).
5395
5396                                     ;*
5397                                     ;* THIS TEST ASSUMES THAT ALL MODULES OTHER THAN CPU CONTROL/DATA PATH
5398                                     ;* ARE KNOWN GOOD MODULES.
5399
5400                                     ;*
5401                                     ;* THIS TEST TOGETHER WITH OTHER CACHE TESTS,ALLOW MFG. TO ELIMINATE
5402                                     ;* HAVING TO RUN THE CACHE DIAGNOSTIC DURING QUICK
5403                                     ;* VERIFY TESTING OF THE CPU MODULE.
5404                                     ;* TEST DESCRIPTION:
5405                                     ;* VERIFY THAT THE CACHE WRITE CONTROL LOGIC WILL INHIBIT A CACHE
5406                                     ;* READ UPDATE TO CACHE TAG STORE DUE TO AN ACCESS TO I/O PAGE.

```

5404

;

5405

.SBITL TEST # 335 - CACHE WRITE CONTROL LOGIC TEST

 :TEST 335 - CACHE WRITE CONTROL LOGIC TEST

5406	041172	005212			TST335: INC	(R2)		;UPDATE TEST NUMBER
	041174	022712	000335		CMP	#335,(R2)		;SEQUENCE ERROR?
	041200	001051			BNE	TST336-10		;BR TO ERROR HALT ON SEQ ERROR
5407	041202	132767	000200	137111	BITB	#200,\$ENVM		;IS APT SIZING?
5408	041210	001405			BEG	4\$;NO;TRY HARDWARE SWITCH REGISTER
5409	041212	032767	000400	137102	BIT	#400,\$SWREG		;YES APT IS SIZING;DOES APT SAY TO DO
								;THIS TEST
5410	041220	001442			BEG	10\$;NO;SKIP TEST
5411	041222	000404			BR	1\$;YES,DO TEST
5412	041224	032737	000400	177570	4\$: BIT	#400,@#177570		;DOES HARDWARE SWITCH REGISTER SAY TO
5413								;TO DO TEST?
5414	041232	001435			BEG	10\$;NO,SKIP TEST
5415								
5416	041234				1\$:			
5417	041234	112737	000002	177750		MOVB #2,@#177750		;ALLOWS CACHE TAG FIELD BITS TO BE
5418								;WRITTEN TO CHR<15:07> ONLY DURING
5419								;THE DESTINATION MEMORY ACCESS
5420								;OF AN INSTRUCTION
5421	041242	012737	000015	177746		MOV #15,@#177746		;NO UCB SO AS TO WRITE ENABLE CACHE STORE
5422	041250	005737	057744			TST @#57744		
5423	041254	005737	077744			TST @#77744		;READ UPDATE;LOAD BIT PATTERN
5424								;000000011 INTO TAG STORE LOCATION
5425								;7762
5426	041260	005737	177744			TST @#177744		;ACCESS I/O PAGE BY READING CME REGISTER.
5427								;THE CACHE COULD DO AN UPDATE TO
5428								;TAG STORE LOCATION 7762 BUT THE ACCESS
5429								;TO I/O PAGE WILL INHIBIT WRITE CONTROL
5430								;LOGIC
5431	041264	005737	057744			TST @#57744		;WRITE TAG STORE DATA FROM LOCATION
5432								;7762 INTO CHR<15:07>.
5433	041270	013700	177752			MOV @#177752,R0		;SAVE CHR DATA
5434	041274	000240				NOP		
5435	041276	000240				NOP		
5436								
5437	041300	105037	177750			CLRB @#177750		;DISABLE MAINTENANCE MODE
5438	041304	012737	000000	177746		MOV #0,@#177746		;TURN ON CACHE
5439	041312	042700	000177			BIC #177,R0		;PREPARE R0 FOR ERROR CHECK
5440	041316	022700	000600			CMP #600,R0		;BITS 15:07 SHOULD BE BIT PATTERN 000000011
5441	041322	001401				BEG 10\$;PASS
5442	041324	000000				HALT		;CACHE WRITE CONTROL CPU LOGIC SIGNAL
5443								;OR SEQUENCE ERROR
5444								;READING TAG DATA BITS<21:13> THRU CACHE HIT REGISTE
5445								;DID NOT RESULT IN BIT PATTERN 000000011.
5446								;THIS INDICATES THAT A READ UPDATE
5447								;MAY HAVE OCCURED TO CACHE TAG STORE
5448								;DUE TO AN ACCESS TO I/O PAGE
5449								;R0 BITS 15:07 CONTAINS DATA RECEIVED
5450								;FROM CACHE HIT REGISTER BITS 15:C7
5451	041326	000240			10\$:	NOP		;END OF TEST
5452	041330	000240				NOP		
5453	041332	000240				NOP		
5454								
5455								

 :* THIS TEST IS OPTIONAL AND IS SELECTED BY SETTING MFM HARDWARE

5456 :* SWITCH REGISTER BIT 08 TO A 1 IN THE CASE OF STANDALONE OPERATION
5457 :* OF THE DIAGNOSTIC. IN THE CASE OF MANUFACTURING APT RUNTIME MODE
5458 :* THEN BIT 08 OF \$SWREG IS SET TO 1 THRU APT SCRIPTING.
5459 :*
5460 :* THIS TEST VERIFIES
5461 :* THE CACHE BYPASS SIGNAL GENERATED FROM THE CPU TO THE CACHE (CACHE BYPASS L).
5462 :* THIS SIGNAL IS GENERATED WHEN THE ASRB INSTRUCTION IS EXECUTED.
5463 :*
5464 :*
5465 :* THIS TEST ASSUMES THAT ALL MODULES EXCEPT CPU DATA PATH/CONTROL STORE
5466 :* ARE KNOWN GOOD MODULES.
5467 :*
5468 :* THIS TEST TOGETHER WITH OTHER CACHE TESTS, ALLOW MFG. TO ELIMINATE
5469 :* HAVING TO RUN THE CACHE DIAGNOSTIC DURING QUICK
5470 :* VERIFY TESTING OF THE CPU MODULE.
5471 :* TEST DESCRIPTION:
5472 :* VERIFY THAT THE ASRB INSTRUCTION WILL CAUSE A CACHE BYPASS
5473 :* UNDER A READ HIT CONDITION.
5474 :*

5475

.SBTTL TEST # 336 - ASRB CACHE BYPASS TEST

 :TEST 336 - ASRB CACHE BYPASS TEST

041334	005212				TST336: INC	(R2)		:UPDATE TEST NUMBER
041336	022712	000336			CMP	#336,(R2)		:SEQUENCE ERROR?
041342	001043				BNE	TST337-10		:BR TO ERROR HALT ON SEQ ERROR
5476	041344	132767	000200	136747	BITB	#200,\$ENVM		:IS APT SIZING?
5477	041352	001405			BEQ	4\$:NO;TRY HARDWARE SWITCH REGISTER
5478	041354	032767	000400	136740	BIT	#400,\$SWREG		:YES APT IS SIZING;DOES APT SAY TO DO
5479								:THIS TEST
5480	041362	001434			BEQ	10\$:NO;SKIP TEST
5481	041364	000404			BR	1\$:YES,DO TEST
5482	041366	032737	000400	177570	4\$: BIT	#400,@#177570		:DOES HARDWARE SWITCH REGISTER SAY TO
5483								:TO DO TEST?
5484	041374	001427			BEQ	10\$:NO,SKIP TEST
5485								
5486	041376				1\$:			
5487	041376	012700	060000			MOV #60000,R0		:SETUP TEST LOCATION ADDRESS
5488								:IN R0
5489	041402	112737	000002	177750		MOVB #2,@#177750		:MODE ALLOWS READ HITS TO BE CACHED,CACHE UPDATES,AN
5490								:CLOCKING OF OUTPUT OF CACHE HIT NAND
5491								:GATE INTO CMR ONLY DURING THE DESTINATION
5492								:ACCESS OF AN INSTRUCTION.
5493	041410	012737	000011	177746		MOV #11,@#177746		:NO UCB SO AS TO WRITE CACHE STORES
5494								:ENABLE LOW CACHE FOR A READ HIT
5495	041416	005710				TST (R0)		:READING LOCATION SPECIFIED BY R0
5496								:WILL ASSURE A READ HIT WHEN THE
5497								:LOCATION IS READ AGAIN
5498	041420	106210				ASRB (R0)		:ASRB INSTRUCTION WILL CAUSE A BYPASS
5499								:TO OCCUR INHIBITING A READ HIT
5500								:TO LOCATION SPECIFIED BY R0.
5501								:THIS SITUATION WILL RESULT IN CMR
5502								:BIT 8 BEING A 1.
5503	041422	013701	177750			MOV @#177750,R1		:SAVE CMR CONTENTS
5504	041426	000240				NOP		
5505	041430	000240				NOP		
5506	041432	105037	177750			CLRB @#177750		:DISABLE MAINT MODE
5507	041436	012737	000000	177746		MOV #0,@#177746		:TURN ON CACHE
5508	041444	032701	000400			BIT #400,R1		:WAS CMR BIT 8 A 1
5509	041450	001001				BNE 10\$:PASS
5510	041452	000000				HALT		:CACHE BYPASS DID NOT OCCUR OR
5511								:SEQUENCE ERROR
5512								:READING OUTPUT OF CACHE HIT NAND GATE
5513								:THRU CMR<8> DID NOT RESULT IN A
5514	041454	000240			10\$:	NOP		:END OF TEST
5515	041456	000240				NOP		
5516	041460	000240				NOP		
5517								
5518								
5519								
5520								
5521								
5522								
5523								
5524								
5525								

 :* THIS TEST IS OPTIONAL AND IS SELECTED BY SETTING MFM HARDWARE
 :* SWITCH REGISTER BIT 08 TO A 1 IN THE CASE OF STANDALONE OPERATION
 :* OF THE DIAGNOSTIC. IN THE CASE OF MANUFACTURING APT RUNTIME MODE
 :* THEN BIT 08 OF \$SWREG IS SET TO 1 THRU APT SCRIPTING.
 :*
 :* THIS TEST VERIFIES THE MICROCODE IN THE CPU CONTROL STORE ASSOCIATED

TEST # 336 - ASRB CACHE BYPASS TEST

5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537

:*
:*
:*
:*
:*
:*
:*
:*
:*
:*
:*
:*

WITH THE CIS INSTRUCTION SET. IT IS ASSUMED THAT ALL MODULES EXCEPT
THE CPU DATA PATH AND CONTROL MODULE ARE ALL 'KNOWN GOOD MODULES'.

THIS TEST ALLOWS MFG. TO ELIMINATE
HAVING TO RUN THE CIS DIAGNOSTIC DURING QUICK
VERIFY TESTING OF THE CPU MODULE.

TEST DESCRIPTION:
THE CPU MPC MICROADDRESSES ASSOCIATED WITH THE CPU ARE
740 THRU 776. IT HAS BEEN DETERMINED THAT BY EXECUTING THIS TEST
ALL BUT MPC ADDRESS 772 AND 773 ARE COVERED.

5538

.SBTTL TEST # 337 - CIS CPU MPC ADDRESS COVERAGE TEST
 :*****
 :TEST 337 - CIS CPU MPC ADDRESS COVERAGE TEST
 :*****

041462 005212
 041464 022712 000337
 041470 001101
 5539 041472 132767 000200 136621
 5540 041500 001405
 5541 041502 032767 000400 136612
 5542
 5543 041510 001472
 5544 041512 000404
 5545 041514 032737 000400 177570 4S:
 5546
 5547 041522 001465
 5548
 5549 041524 012737 041562 000010 1S:
 5550 041532 012737 000340 000012
 5551 041540 076001
 5552
 5553
 5554
 5555
 5556 041542 000240
 5557 041544 012737 042416 000010
 5558 041552 005037 000012
 5559 041556 000000
 5560
 5561
 5562 041560 000406
 5563 041562 022626 2S:
 5564 041564 012737 042416 000010
 5565 041572 005037 000012
 5566
 5567 041576 076150 5S:
 5568
 5569 041600 041610
 5570 041602 041614
 5571 041604 041620
 5572
 5573 041606 000414
 5574
 5575 076150
 5576 041610 000006
 5577 041612 041624
 5578 041614 000006
 5579 041616 041624
 5580 041620 000006
 5581 041622 041632
 5582 041624 001002
 5583 041626 001002
 5584 041630 031002
 5585 041632 000000
 5586 041634 000000
 5587 041636 000000
 5588

TST337: INC (R2) ;UPDATE TEST NUMBER
 CMP #337,(R2) ;SEQUENCE ERROR?
 BNE TST340-10 ;BR TO ERROR HALT ON SEQ ERROR
 BITB #200,\$ENVM ;IS APT SIZING?
 BEQ 4S ;NO;TRY HARDWARE SWITCH REGISTER
 BIT #400,\$SWREG ;YES APT IS SIZING;DOES APT SAY TO DO
 ;THIS TEST
 BEQ ENDMPC ;NO;SKIP TEST
 BR 1S ;YES,DO TEST
 BIT #400,@#177570 ;DOES HARDWARE SWITCH REGISTER SAY TO
 ;TO DO TEST?
 BEQ ENDMPC ;NO,SKIP TEST
 ;SETUP FOR POSSIBLE TRAP
 MOV #2S,@#10 ;THIS CIS INSTRUCTION SHOULD CAUSE A TRAP
 MOV #340,@#12 ;TO OCCUR TO VECTOR 10 INDICATING THAT
 76001 ;SWITCH S1 OF CIS M7092 DATA PATH MODULE
 ;IS OPEN. THE SWITCH MUST BE OPEN TO ALLOW
 ;THIS TEST TO RUN.
 NOP
 MOV #T010,@#10 ;RESTORE VECTORS
 CLR @#12
 HALT ;CHECK TO SEE THAT SWITCH S1 OF M7092 CIS DATA
 ;PATH MODULE IS OPEN.
 BR 5S
 CMP (SP)+,(SP)+ ;READJUST STACK
 MOV #T010,@#10 ;RESTORE VECTORS
 CLR @#12
 ADDNI ;:NOW PERFORM CIS INSTRUCTION WHICH WILL
 ;COVER MOST OF THE CPU MPC MICROADDRESSES
 .WORD A
 .WORD B
 .WORD D
 BR CHECK ;NOW CHECK RESULTS
 ADDNI=76150
 A: .WORD 6
 .WORD SA
 B: .WORD 6
 .WORD SA
 D: .WORD 6
 .WORD DA
 SA: .WORD 1002
 .WORD 1002
 .WORD 31002
 DA: .WORD 0
 .WORD 0
 .WORD 0

```

5589 041640          CHECK:
5590 041640 022737 032064 041632      CMP      #32064,@#DA      ;CHECK RESULTS
5591 041646 001401          BEQ      6$              ;PASS
5592 041650 000000          HALT          ;ERROR
5593 041652 022737 032064 041634 6$:  CMP      #32064,@#DA+2  ;CHECK RESULT
5594 041660 001401          BEQ      7$              ;PASS
5595 041662 000000          HALT          ;ERROR
5596 041664 022737 032064 041636 7$:  CMP      #32064,@#DA+4  ;CHECK RESULT
5597 041672 001401          BEQ      ENDMPC         ;PASS
5598 041674 000000          HALT          ;CIS INSTUCTION OR SEQUENCE ERROR
5599
5600 041676 000240          ENDMPC: NOP
5601 041700 000240          NOP
5602 041702 000240          NOP
5603          .SBTTL TEST #340 - CPU ERROR REGISTER BIT 0 CHECK
5604          :*****
5605          :* TEST 340 CPU ERROR REGISTER BIT 0 CHECK
5606
5607 041704 005212          TST340: INC      (R2)          ;UPDATE TEST NUMBER
5608 041706 022712 000340      CMP      #340,(R2)      ;SEQUENCE ERROR?
5609 041712 001004          BNE      1$              ;BR TO ERROR HALT ON SEQ ERROR
5610 041714 032767 000001 136044      BIT      #BIT0,CPUERR  ;TEST BIT 0 OF THE CPU ERROR REGISTER (BIT 0 SET INDICATES
5611          ;1 OR MORE OF THE POWER SUPPLIES ARE OUT OF SPECIFICATION)
5612 041722 001411          BEQ      ENDPAS         ;BRANCH IF FOUND CLEAR
5613          ;TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <===
5614          ;CONDITIONAL BRANCH INST. AND <===
5615          ;REPLACE THE BIC INSTRUCTION <===
5616          ;WHICH FOLLOWS W/ 775 <===
5617 041724 012762 000772 177776 1$:  MOV      #772,-2(R2)    ;MOVE TO MAILBOX # ***** 772 *****
5618          :*****IMPORTANT - CHANGE THE #772 ABOVE AS APPROPRIATE IF ANY ERRORS ARE EITHER
5619          :ADDED OR REMOVED.*****
5620 041732 005262 177774          INC      -4(R2)          ;SET MSGTYP TO FATAL ERROR
5621 041736 000000          HALT          ;CPU BIT 0 IS SET OR TEST SEQUENCE ERROR
5622 041740 042767 000001 136020      BIC      #BIT0,CPUERR  ;CLEAR THE PWR MON BIT IN THE CPU ERROR REGISTER
    
```

```

5623 :*****
5624 :                               END OF PASS SEQUENCE
5625 :*****
5626
5627 041746 005212      ENDPASS: INC      (R2)          ;UPDATE TEST NUMBER
5628 041750 022712 000341  CMP      #341,(R2)      ;SEQUENCE ERROR?
5629 041754 001042      BNE      EOP1          ;BR TO ERROR HALT ON SEQ ERROR
5630 041756 005237 000306  INC      @#SPASS
5631 041762 105267 000112  INCB    PASSPT        ;SHOULD PRINT HIS PASS?
5632 041766 001033      BNE      GOAGIN       ;NO
5633 041770 132767 000040 136323  BITB    #40,$ENVM     ;WILL APT ALLOW PRINTING?
5634 041776 001017      BNE      ACT          ;NO
5635 042000 023727 000042 042046  CMP      @#42,$SENDAD ;UNDER ACT AUTO ACCEPT?
5636 042006 001413      BEQ     ACT          ;IF SO SKIP PRINTOUT
5637 042010 012700 042102  MOV     #MSG,RO       ;GET MSG ADDR.
5638 042014 105737 177564  WAIT:  TSTB    @#TPS   ;TTY READY
5639 042020 100375      BPL     WAIT         ;NO WAIT
5640 042022 121027 000377  CMPB    (RO),#377    ;IS NEXT CHAR. THE TERMINATOR
5641 042026 001403      BEQ     ACT          ;YES THEN BR
5642 042030 112037 177566  MOVB   (RO)+,@#TPB  ;PRINT CHARACTER
5643 042034 000767      BR     WAIT         ;NEXT IF NOT DONE.
5644 042036 013700 000042  ACT:   MOV     @#42,RO ;CHECK ACT
5645 042042 001405      BEQ     GOAGIN       ;KEEP GOING
5646 042044 000005      RESET
5647 042046 004710  SENDAD: JSR     PC,(RO) ;ACT HOOKS
5648 042050 000240      NOP
5649 042052 000240      NOP
5650 042054 000240      NOP
5651 042056 000167 137040  GOAGIN: JMP     RESTRT ;DO NEXT PASS
5652 042062 000772 177776  EOP1:  MOV     #772,-2(R2) ;MOVE TO MAILBOX # ***** 772 *****
5653 042062 012762 000772 177776  INC     -4(R2)      ;SET MSGTYP TO FATAL ERROR
5654 042070 005262 177774  HALT    ;SEQUENCE ERROR
5655 042076 001100      START
5656
5657 042100 177777  PASSPT: -1
5658 042102 015 012 000  MSG:   .ASCII <15><12><0><0><0><0><0><0>.END OF.
5659 042105 000 000 000
5659 042110 000 000 105
5659 042113 116 104 040
5659 042116 117 106
5659 042120 040 040 103
5659 042123 113 113 101
5659 042126 101 102 060
5659 042131 040 061 061
5659 042134 057 064 064
5659 042137 040 103 120
5659 042142 125 057 105
5659 042145 111 123 000
5659 042150 000 000 377
5660
5661 042154 000402  .EVEN
5662 042156 001002  BRTAB: BR      .+6
5663 042160 001402  BNE     .+6
5664 042162 002002  BEQ    .+6
5664 042162 002002  BGE    .+6
    
```

5665	042164	002402				BLT	+.6		
5666	042166	003002				BGT	+.6		
5667	042170	003402				BLE	+.6		
5668	042172	100002				BPL	+.6		
5669	042174	100402				BMI	+.6		
5670	042176	101002				BHI	+.6		
5671	042200	101402				BLOS	+.6		
5672	042202	102002				BVC	+.6		
5673	042204	102402				BVS	+.6		
5674	042206	103002				BCC	+.6		; SAME AS BHIS
5675	042210	103402				BCS	+.6		; SAME AS BLO
5676	042212	000000				STMP0:	.WORD 0		
5677	042214	000000				STMP1:	.WORD 0		
5678	042216	000000				STMP2:	.WORD 0		
5679	042220	000000				STMP3:	.WORD 0		
5680	042222	000000				SEPIRQ:	.WORD 0		; ERROR PIRQ.
5681	042224	000000				SLPADR:	.WORD 0		
5682	042226	000000				SLPERR:	.WORD 0		; ERROR LOOP
5683		000002				.RADIX	2		
5684	042230	177777				YNTAB:	1111111111111111		; BR
5685	042232	170360					1111000011110000		; BNE: Z=0
5686	042234	007417					0000111100001111		; BEQ: Z=1
5687	042236	146063					1100110000110011		; BGE: N XOR V =0
5688	042240	031714					0011001111001100		; BLT: N XOR V =1
5689	042242	140060					1100000000110000		; BGT: Z+(N XOR V) =0
5690	042244	037717					0011111111001111		; BLE: Z+(N XOR V) -1
5691									
5692	042246	177400					1111111100000000		; BPL: N=0
5693	042250	000377					0000000011111111		; BMI: N=1
5694	042252	120240					1010000010100000		; BHI: C+Z=0
5695	042254	057537					0101111101011111		; BLOS: C+Z=1
5696	042256	146314					1100110011001100		; BVC: V=0
5697	042260	031463					0011001100110011		; BVS: V=1
5698	042262	125252					1010101010101010		; BCC: C=0
5699	042264	052525					0101010101010101		; BCS: C=1
5700		000010				.RADIX	8		
5701									
5702	042266	012737	042276	000024	PWRDN:	MOV	#PWRUP,@#24		; SET UP FOR A POWER UP
5703	042274	000000				HALT			
5704									
5705	042276	012737	042266	000024	PWRUP:	MOV	#PWRDN,@#24		; SET UP FOR A POWER FAIL
5706	042304	012706	001000			MOV	#STBOT,R6		; SET UP STACK POINTER
5707	042310	132767	000040	136003		BITB	#40,\$ENVM		; SHOULD PRINT?
5708	042316	001010				BNE	PWR2		; IF NOT: BR
5709	042320	012700	042344			MOV	#PFMES,R0		; GET POWER FAIL MESSG.
5710	042324	105737	177564		WATE:	TSTB	@#TPS		; TTY READY?
5711	042330	103375				BPL	WATE		; IF NOT: BR
5712	042332	112037	177566			MOVB	(R0)+,@#TPB		; PRINT NEXT CHAR.
5713	042336	001372				BNE	WATE		; IF NOT DONE: BR
5714	042340	000137	001100		PWR2:	JMP	@#START		; START PROGRAM AGAIN
5715									
5716									
5717	042344	012	015	120	PFMES:	.ASCIZ	<12><15>.POWER FAILURE.<12><15>		
	042347	117	127	105					
	042352	122	040	106					
	042355	101	111	114					
	042360	125	122	105					

```

042363 012 015 000
5718 .EVEN
5719 042366 .BLKW 6
5720 042402 USTBOT:
5721 :*****
5722 : THE FOLLOWING ARE SPECIAL CPU TRAP
5723 : HANDLERS TO TRAP AND REPORT SPECIAL TRAPS.
5724 :
5725 :*****
5726
5727 042402 T04:
042402 012762 000773 177776 MOV #773,-2(R2) ;MOVE TO MAILBOX # ***** 773 *****
042410 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042414 000000 HALT ;TRAPPED THRU LOC. 4
5728 042416 T010:
042416 012762 000774 177776 MOV #774,-2(R2) ;MOVE TO MAILBOX # ***** 774 *****
042424 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042430 000000 HALT ;TRAPPED THRU LOC. 10
5729 042432 T014:
042432 012762 000775 177776 MOV #775,-2(R2) ;MOVE TO MAILBOX # ***** 775 *****
042440 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042444 000000 HALT ;TRAPPED THRU LOC. 14
5730 042446 T030:
042446 012762 000776 177776 MOV #776,-2(R2) ;MOVE TO MAILBOX # ***** 776 *****
042454 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042460 000000 HALT ;TRAPPED THRU LOC. 30
5731 042462 T034:
042462 012762 000777 177776 MOV #777,-2(R2) ;MOVE TO MAILBOX # ***** 777 *****
042470 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042474 000000 HALT ;TRAPPED THRU LOC. 34
5732 042476 T0114:
042476 012762 001000 177776 MOV #1000,-2(R2) ;MOVE TO MAILBOX # ***** 1000 *****
042504 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042510 000000 HALT ;TRAPPED THRU LOC. 114
5733 042512 T0244:
042512 012762 001001 177776 MOV #1001,-2(R2) ;MOVE TO MAILBOX # ***** 1001 *****
042520 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042524 000000 HALT ;TRAPPED THRU LOC. 244
5734 042526 T0250:
042526 012762 001002 177776 MOV #1002,-2(R2) ;MOVE TO MAILBOX # ***** 1002 *****
042534 005262 177774 INC -4(R2) ;SET MSGTYP TO FATAL ERROR
042540 000000 HALT ;TRAPPED THRU LOC. 250
5735 042542 000000 SPSW: 0
5736 042544 000000 SUBT: .WORD 0
5737 042546 042544 SUBT1: .WORD SUBT
5738 042550 000002 S1: 2
5739 042552 042550 S2: S1
5740 042554 177771 S3: -7
5741 042556 042554 S4: S3
5742
5743 042560 000000 END: 0
5744 000001 .END
  
```

A	041610	APASS	000000	BIT7	= 000200	DEC5	022216	DOP2	011710
ABASE	= 000000	APRIOR=	000000	BIT8	= 000400	DEC6	022232	DOP4	015706
ACDW1	000000	AROUND	026646	BIT9	= 001000	DEC7	022254	DOP5	015776
ACDW2	000000	ASL1	024454	BPTVEC	= 000014	DISPRE	000174	DST.PT	035600
ACPUOP	= 000000	ASL2	024470	BRCT	026710	DNMB0A	012320	DSWR	= 177570
ACT	042036	ASL3	024506	BRC1	003700	DNMB0B	012334	FIS	030154
ADC1	022764	ASL4	024522	BRC2	003714	DNMB2A	012612	EMTVEC	= 000030
ADC2	023000	ASL5	024536	BRC3	003724	DNMB2B	012626	END	042560
ADC3	023020	ASL6	024552	BRH	026656	DNMB2C	012646	ENDABO	035746
ADC4	023034	ASL7	024576	BRN1	003540	DNMB2D	012662	ENDMPC	041676
ADC5	023052	ASR1	024644	BRN2	003554	DNMB2E	012676	ENDPAS	041746
ADDNI	= 076150	ASR2	024660	BRN3	003564	DNMB2F	012720	EOP1	042062
ADDW0	= 000000	ASR3	024702	BRTAB	042154	DNMB3A	013006	ER	026670
ADDW1	= 000000	ASR4	024716	BRV1	003620	DNMB3B	013022	ERROR	= 104000
ADDW10	= 000000	ASR5	024732	BRV2	003634	DNMB3C	013044	ERRVEC	= 000004
ADDW11	= 000000	ASR6	024746	BRV3	003644	DNMB3D	013062	FILL	= 000377
ADDW12	= 000000	ASR7	024776	BRZ1	003460	DNMB3E	013076	GOAGIN	042056
ADDW13	= 000000	ASWREG=	000000	BRZ2	003474	DNMB4A	013306	HIADRS	= 177742
ADDW14	= 000000	ATESTN=	000000	BRZ3	003504	DNMB4B	013322	HITMIS	= 177752
ADDW15	= 000000	AUNIT	= 000000	BR1	001172	DNMB4C	013344	HT	= 000011
ADDW2	= 000000	AUSWR	= 000000	BR2	001206	DNMB4D	013354	IJMP	020524
ADDW3	= 000000	AVECT1=	000000	BR3	001224	DNMB4E	013370	IJMP4	020246
ADDW4	= 000000	AVECT2=	000000	BR4	001232	DNMB4F	013410	IJMP5	020462
ADDW5	= 000000	B	041614	BR5	001246	DNM03A	011306	INC1	022004
ADDW6	= 000000	BIC1	021604	BTCON	027210	DNM03B	011322	INC2	022020
ADDW7	= 000000	BIC2	021620	BTERR	027170	DNM03C	011332	INC3	022042
ADDW8	= 000000	BIC3	021636	CACHVE=	000114	DNM1	011140	INC4	022056
ADDW9	= 000000	BIS1	021704	CC	026652	DNM1A	012406	INC5	022072
ADD1	022554	BIS2	021720	CCERR	027440	DNM1B	012422	IOTVEC	= 000020
ADD2	022570	BIS3	021740	CC1	027322	DNM2	011160	JMPCK	020526
ADD3	022604	BITCHK	027042	CC2	027336	DNM2A	012474	JMPERR	027244
ADD4	022620	BITCLR	026770	CHECK	041640	DNM2B	012510	JMPSEQ	020552
ADD5	022636	BITCON	027130	CLRCD	027320	DNM2C	012516	JMPT	027234
ADD6	022652	BITSET	027006	CLR1	022316	DNM2D	012532	JMP2	020250
ADD7	022664	BIT0	= 000001	CMP1	023260	DNM3	011202	JMP2A	020272
ADD8	022700	BIT00	= 000001	CMP2	023274	DNM4	011230	JMP3	020166
ADD9	022720	BIT01	= 000002	CMP3	023316	DNM4A	013164	JMP3A	020210
ADEVCT=	000000	BIT02	= 000004	CMP4	023332	DNM4B	013200	JMP3B	020234
ADEVN	= 000000	BIT03	= 000010	CMP5	023356	DNM4C	013220	JMP4	020304
AENV	= 000000	BIT04	= 000020	CMP6	023372	DNM5A	013474	JMP4A	020326
AENVN	= 000000	BIT05	= 000040	CMP7	023412	DNM5B	013510	JMP4B	020352
AFATAL	000000	BIT06	= 000100	COM1	023456	DNM5C	013532	JMP5	020424
AMADR1	= 000000	BIT07	= 000200	CONT	026712	DNM6A	013616	JMP5A	020450
AMADR2	= 000000	BIT08	= 000400	CONTRL=	177746	DNM6B	013632	JMP6	020364
AMADR3	= 000000	BIT09	= 001000	CON1	027356	DNM6C	013654	JMP6A	020410
AMADR4	= 000000	BIT1	= 000002	CON2	027454	DNM7A	013742	JMP7	020464
AMAMS1	= 000000	BIT10	= 002000	CPUERR=	177766	DNM7B	013756	JMP7A	020510
AMAMS2	= 000000	BIT11	= 004000	CR	= 000015	DNM7C	014000	JSRCK	021240
AMAMS3	= 000000	BIT12	= 010000	CRLF	= 000200	DOPB2A	011774	JSRCKA	021234
AMAMS4	= 000000	BIT13	= 020000	D	041620	DOPB2B	012062	JSRCK1	021256
AMSGAD	= 000000	BIT14	= 040000	DA	041632	DOP0A	010646	JSRSEQ	021236
AMSGLG	= 000000	BIT15	= 100000	DAERR	027114	DOP0B	010676	JSRO	020566
AMSGTY	= 000000	BIT2	= 000004	DDISP	= 177570	DOP0C	010722	JSR1	020572
AMTYP1	= 000000	BIT3	= 000010	DEC1	022136	DOP0D	010756	JSR1A	020614
AMTYP2	= 000000	BIT4	= 000020	DEC2	022152	DOP03A	011040	JSR2	020714
AMTYP3	= 000000	BIT5	= 000040	DEC3	022166	DOP03B	011054	JSR2A	020744
AMTYP4	= 000000	BIT6	= 000100	DEC4	022202	DOP1	011562	JSR2B	020760

JSR3 020630
JSR3A 020670
JSR3B 020704
JSR4 021002
JSR4A 021020
JSR4B 021034
JSR5 021116
JSR5A 021142
JSR5B 021156
JSR6 021046
JSR6A 021072
JSR6AD 021232
JSR6B 021106
JSR7 021170
JSR7A 021206
JSR7B 021222
KDPAR0= 172360
KDPAR1= 172362
KDPAR2= 172364
KDPAR3= 172366
KDPAR4= 172370
KDPAR5= 172372
KDPAR6= 172374
KDPAR7= 172376
KDPDR0= 172320
KDPDR1= 172322
KDPDR2= 172324
KDPDR3= 172326
KDPDR4= 172330
KDPDR5= 172332
KDPDR6= 172334
KDPDR7= 172336
KERSTK= 001100
KIPAR0= 172340
KIPAR1= 172342
KIPAR2= 172344
KIPAR3= 172346
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPAR7= 172356
KIPDR0= 172300
KIPDR1= 172302
KIPDR2= 172304
KIPDR3= 172306
KIPDR4= 172310
KIPDR5= 172312
KIPDR6= 172314
KIPDR7= 172316
KMTSTS 035160
KSP = 0000006
LF = 000012
LKS = 177546
LKVEC = 000100
LOADRS= 177740
MAINT = 177750
MAPHO = 170202

MAPH00= 170202
MAPH01= 170206
MAPH02= 170212
MAPH03= 170216
MAPH04= 170222
MAPH05= 170226
MAPH06= 170232
MAPH07= 170236
MAPH1 = 170206
MAPH10= 170242
MAPH11= 170246
MAPH12= 170252
MAPH13= 170256
MAPH14= 170262
MAPH15= 170266
MAPH16= 170272
MAPH17= 170276
MAPH2 = 170212
MAPH20= 170302
MAPH21= 170306
MAPH22= 170312
MAPH23= 170316
MAPH24= 170320
MAPH25= 170326
MAPH26= 170332
MAPH27= 170336
MAPH3 = 170216
MAPH30= 170342
MAPH31= 170346
MAPH32= 170352
MAPH33= 170356
MAPH34= 170362
MAPH35= 170366
MAPH36= 170372
MAPH37= 170376
MAPH4 = 170222
MAPH5 = 170226
MAPH6 = 170232
MAPH7 = 170236
MAPL0 = 170200
MAPL00= 170200
MAPL01= 170204
MAPL02= 170210
MAPL03= 170214
MAPL04= 170220
MAPL05= 170224
MAPL06= 170230
MAPL07= 170234
MAPL1 = 170204
MAPL10= 170240
MAPL11= 170244
MAPL12= 170250
MAPL13= 170254
MAPL14= 170260
MAPL15= 170264
MAPL16= 170270
MAPL17= 170274

MAPL2 = 170210
MAPL20= 170300
MAPL21= 170304
MAPL22= 170310
MAPL23= 170314
MAPL24= 170320
MAPL25= 170324
MAPL26= 170330
MAPL27= 170334
MAPL3 = 170214
MAPL30= 170340
MAPL31= 170344
MAPL32= 170350
MAPL33= 170354
MAPL34= 170360
MAPL35= 170364
MAPL36= 170370
MAPL37= 170374
MAPL4 = 170220
MAPL5 = 170224
MAPL6 = 170230
MAPL7 = 170234
MBDM2A 014244
MBDM2B 014260
MBDM2C 014302
MBDM2D 014314
MBDM2E 014330
MBDM2F 014352
MBDM4A 014756
MBDM4B 015000
MBDM4C 015012
MBDM4D 015026
MBDM4E 015046
MDM1A 014054
MDM1B 014070
MDM2A 014136
MDM2B 014152
MDM2C 014160
MDM2D 014174
MDM3A 014432
MDM3B 014446
MDM3C 014470
MDM3D 014514
MDM3E 014546
MDM4A 014632
MDM4B 014646
MDM4C 014666
MDM5A 015122
MDM5B 015136
MDM5C 015160
MDM5D 015202
MDM5E 015234
MDM6A 015310
MDM6B 015324
MDM6C 015346
MDM6D 015372
MDM6E 015426

MDM7A 015506
MDM7B 015522
MDM7C 015544
MDM7D 015570
MDM7E 015620
MEMERR= 177744
MFPD0 027556
MFPD0A 027610
MFP10 026344
MFP10A 026376
MFPT = 000007
MMHDLR 035564
MMR0 = 177572
MMR1 = 177574
MMR2 = 177576
MMR3 = 172516
MMVEC = 000250
MOVCI = 076130
MOV1 021404
MOV2 021420
MOV3 021436
MRK1 025426
MRK2 025450
MRK3 025464
MRK4 025506
MRK5 025524
MRK6 025540
MSG 042102
MTPD0 027674
MTP10 026466
NBR 026662
NEG00 004774
NEG01 005010
NEG02 005030
NEG03 005044
NEG04 005060
NEG1 023120
NEG10 005130
NEG11 005144
NEG12 005166
NEG13 005202
NEG14 005216
NEG2 023134
NEG20 005270
NEG21 005304
NEG22 005336
NEG3 023156
NEG30 005714
NEG31 005730
NEG32 005750
NEG33 006000
NEG34 006020
NEG4 023172
NEG40 006442
NEG41 006456
NEG42 006476
NEG5 023212

NEG50 006556
NEG51 006572
NEG52 006612
NEG60 006674
NEG61 006710
NEG70 006764
NEG71 007000
PASSPT 042100
PDRTAB 035706
PFMES 042344
PIRQ = 177772
PIROVE= 000240
PRO = 000000
PR1 = 000040
PR2 = 000100
PR3 = 000140
PR4 = 000200
PR5 = 000240
PR6 = 000300
PR7 = 000340
PS = 177776
PSW = 177776
PUSRM = 030000
PURDN 042266
PURUP 042276
PURVEC= 000024
PUR2 042340
REG1 002240
REG1A 002310
REG1E 002252
REG2 002364
REG2A 002416
REG2B 002444
REG2C 002502
REG3 002530
REG3A 002600
REG3E 002542
REG4 002654
REG4A 002724
REG4E 002666
REG5 003000
REG5A 003050
REG5E 003012
REG6 003124
REG6A 003174
REG6E 003136
REN 040546
RESR 035636
REST 025630
RESTR 001122
RESVEC= 000010
ROL1 024104
ROL2 024120
ROL3 024136
ROL4 024152
ROL5 024166
ROL6 024202

ROL7	024224	SB2	017534	SIZELO=	177760	SOP1B	004246	SW08	= 000400
ROR1	024272	SB4	017674	SMTSTS	035272	SOP2B	004530	SW09	= 001000
ROR2	024306	SB5	017772	SNMBOA	007214	SOP3A	005420	SW1	= 000002
ROR3	024324	SB5A	017764	SNMB1A	007330	SOP3B	005434	SW10	= 002000
ROR4	024340	SB5X	020006	SNMB1B	007344	SOP4A	006106	SW11	= 004000
ROR5	024356	SB5XAD	020010	SNMB1C	007366	SOP4B	006126	SW12	= 010000
ROR6	024372	SB6	020050	SNMB2A	007524	SOP5A	006200	SW13	= 020000
ROR7	024406	SB6X	020064	SNMB2B	007540	SOP5B	006214	SW14	= 040000
ROTX	017204	SB7	020124	SNMB2C	007560	SOP6A	006270	SW15	= 100000
ROTXAD	017340	SB7X	020140	SNMB2D	007600	SOP6B	006304	SW2	= 000004
ROTGA	016236	SB7XAD	020142	SNMB2E	007614	SOP7A	006362	SW3	= 000010
ROTOB	016252	SCOPE =	000004	SNMB3A	007772	SOP7B	006376	SW4	= 000020
ROTOC	016274	SC3	027420	SNMB3B	010006	SPSW	042542	SW5	= 000040
ROT1A	016346	SC4	027434	SNMB3C	010024	SRC.PT	035574	SW6	= 000100
ROT1B	016362	SDPAR0=	172260	SNMB3D	010040	SR0	= 177572	SW7	= 000200
ROT1C	016406	SDPAR1=	172262	SNM0A	007150	SR1	= 177574	SW8	= 000400
ROT1D	016422	SDPAR2=	172264	SNM1A	007262	SR2	= 177576	SW9	= 001000
ROT1E	016452	SDPAR3=	172266	SNM2A	007434	SR3	= 172516	SXT0	025050
ROT2A	016530	SDPAR4=	172270	SNM2B	007450	SSP	= 0000006	SXT1	025064
ROT2B	016544	SDPAR5=	172272	SNM3A	007674	SSP1A	025704	SXT2	025114
ROT2C	016574	SDPAR6=	172274	SNM3B	007710	SSP2	025720	SVSTID=	177764
ROT2D	016610	SDPAR7=	172276	SNM4A	010114	STACK	= 001100	S1	042550
ROT2E	016644	SDPDR0=	172220	SNM4B	010130	START	001100	S2	042552
ROT3A	016716	SDPDR1=	172222	SNM5A	010206	STBOT	= 001000	S3	042554
ROT3B	016732	SDPDR2=	172224	SNM5B	010222	STKLMT=	177774	S4	042556
ROT3C	016760	SDPDR3=	172226	SNM6A	010302	STK1	035704	TBITVE=	000014
ROT3D	016774	SDPDR4=	172230	SNM6B	010316	SUBT	042544	TBL1	015732
ROT3E	017022	SDPDR5=	172232	SNM7A	010374	SUBT1	042546	TBL2	016022
ROT4	017102	SDPDR6=	172234	SNM7B	010410	SUB0	010560	TEST1	022360
ROT5	017170	SDPDR7=	172236	SOB1	025272	SUB0A	010574	TEST2	022374
ROT6	017244	SETBR	026542	SOB2	025300	SUB1	023524	TEST3	022412
ROT7	017324	SETCC	026562	SOB3	025314	SUB2	023540	TKVEC	= 000060
RTS1	021334	SETCD	027416	SOB4	025334	SUB3	023562	T010	042416
R1ERR	002326	SETPDR	035106	SOPA	007062	SUB4	023576	T014	042476
R10	=%000000	SETUP	026524	SOPB	007102	SUB5	023614	T014	042432
R11	=%000001	SET2BR	026632	SOPB0A	004156	SUB6	023630	T0244	042512
R12	=%000002	SHL	001654	SOPB0B	004166	SUB7	023650	T0250	042526
R13	=%000003	SHLE	001670	SOPB1A	004320	SUPSTK=	000700	T030	042446
R14	=%000004	SHR	002000	SOPB1B	004336	SVR0	035670	T034	042462
R15	%000005	SHRE	002014	SOPB1C	004412	SVR1	035672	T04	042402
R2ERR	002466	SIPAR0=	172240	SOPB1D	004432	SVR2	035674	TPB	= 177566
R3ERR	002616	SIPAR1=	172242	SOPB2A	004606	SVR3	035676	TPS	= 177564
R4ERR	002742	SIPAR2=	172244	SOPB2B	004626	SVR4	035700	TPVEC	= 000064
R5ERR	003066	SIPAR3=	172246	SOPB2C	004706	SVR5	035702	TRAPVE=	000034
R6	=%000006	SIPAR4=	172250	SOPB2D	004732	SWB1	022456	TRTVEC=	000014
R6ERR	003212	SIPAR5=	172252	SOPB3A	005514	SWB2	022472	TST1	001154
R7	=%000007	SIPAR6=	172254	SOPB3B	005540	SWB3	022510	TST10	001630
SA	041624	SIPAR7=	172256	SOPB3C	005616	SWREG	000176	TST100	007402
SAVR	035604	SIPDR0=	172200	SOPB3D	005640	SW0	= 000001	TST101	007472
SBC1	023716	SIPDR1=	172202	SOPX	007042	SW00	= 000001	TST102	007636
SBC2	023732	SIPDR2=	172204	SOPXAD	007116	SW01	= 000002	TST103	007732
SBC3	023750	SIPDR3=	172206	SOPZA	004510	SW02	= 000004	TST104	010062
SBC4	023764	SIPDR4=	172210	SOP0A	003770	SW03	= 000010	TST105	010150
SBC5	024002	SIPDR5=	172212	SOP0B	004014	SW04	= 000020	TST106	010244
SBC6	024016	SIPDR6=	172214	SOP0C	004066	SW05	= 000040	TST107	010336
SBC7	024036	SIPDR7=	172216	SOP0D	004112	SW06	= 000100	TST11	001704
SBC	017376	SIZEHI=	177762	SOP1A	004234	SW07	000200	TST110	010430

TST111 010470
TST112 010530
TST113 010614
TST114 011002
TST115 011102
TST116 011252
TST117 011346
TST12 001754
TST120 011410
TST121 011452
TST122 011514
TST123 011604
TST124 011650
TST125 011732
TST126 012016
TST127 012104
TST13 002030
TST130 012152
TST131 012220
TST132 012266
TST133 012354
TST134 012442
TST135 012554
TST136 012742
TST137 013122
TST14 002064
TST140 013244
TST141 013432
TST142 013556
TST143 013700
TST144 014024
TST145 014110
TST146 014216
TST147 014376
TST15 002122
TST150 014600
TST151 014710
TST152 015070
TST153 015256
TST154 015452
TST155 015644
TST156 015734
TST157 016024
TST16 002160
TST160 016114
TST161 016204
TST162 016310
TST163 016466
TST164 016660
TST165 017036
TST166 017116
TST167 017206
TST17 002216
TST170 017260
TST171 017342
TST172 017420
TST173 017466

TST174 017556
TST175 017624
TST176 017714
TST177 020012
TST2 001264
TST20 002266
TST200 020066
TST201 020144
TST202 020554
TST203 021272
TST204 021356
TST205 021452
TST206 021552
TST207 021652
TST21 002342
TST210 021754
TST211 022106
TST212 022270
TST213 022332
TST214 022426
TST215 022524
TST216 022734
TST217 023066
TST22 002422
TST220 023226
TST221 023426
TST222 023472
TST223 023664
TST224 024052
TST225 024240
TST226 024422
TST227 024612
TST23 002506
TST230 025012
TST231 025130
TST232 025252
TST233 025350
TST234 025554
TST235 025630
TST236 025750
TST237 026260
TST24 002556
TST240 026376
TST241 026514
TST242 026760
TST243 027134
TST244 027214
TST245 027260
TST246 027474
TST247 027610
TST25 002632
TST250 027722
TST251 027760
TST252 030154
TST253 030260
TST254 030354
TST255 030450

TST256 030544
TST257 030644
TST26 002702
TST260 030744
TST261 031044
TST262 031144
TST263 031272
TST264 031420
TST265 031546
TST266 031676
TST267 032022
TST27 002756
TST270 032146
TST271 032272
TST272 032416
TST273 032546
TST274 032674
TST275 033012
TST276 033130
TST277 033246
TST3 001324
TST30 003026
TST300 033364
TST301 033502
TST302 033620
TST303 033736
TST304 034054
TST305 034202
TST306 034330
TST307 034456
TST31 003102
TST310 034602
TST311 035754
TST312 036104
TST313 036230
TST314 036354
TST315 036476
TST316 036620
TST317 036742
TST32 003152
TST320 037064
TST321 037206
TST322 037272
TST323 037356
TST324 037540
TST325 037662
TST326 040004
TST327 040070
TST33 003226
TST330 040154
TST331 040240
TST332 040506
TST333 040634
TST334 040720
TST335 041172
TST336 041334
TST337 041462

TST34 003272
TST340 041704
TST35 003334
TST36 003376
TST37 003440
TST4 001366
TST40 003520
TST41 003600
TST42 003660
TST43 003740
TST44 004034
TST45 004126
TST46 004202
TST47 004262
TST5 001430
TST50 004352
TST51 004446
TST52 004544
TST53 004642
TST54 004746
TST55 005100
TST56 005240
TST57 005360
TST6 001472
TST60 005450
TST61 005554
TST62 005654
TST63 006050
TST64 006142
TST65 006230
TST66 006320
TST67 006412
TST7 001542
TST70 006516
TST71 006634
TST72 006730
TST73 007030
TST74 007120
TST75 007164
TST76 007230
TST77 007276
UDPAR0= 177660
UDPAR1= 177662
UDPAR2= 177664
UDPAR3= 177666
UDPAR4= 177670
UDPAR5= 177672
UDPAR6= 177674
UDPAR7= 177676
UDPDR0= 177620
UDPDR1= 177622
UDPDR2= 177624
UDPDR3= 177626
UDPDR4= 177630
UDPDR5= 177632
UDPDR6= 177634
UDPDR7= 177636

UIPAR0= 177640
UIPAR1= 177642
UIPAR2= 177644
UIPAR3= 177646
UIPAR4= 177650
UIPAR5= 177652
UIPAR6= 177654
UIPAR7= 177656
UIPDR0= 177600
UIPDR1= 177602
UIPDR2= 177604
UIPDR3= 177606
UIPDR4= 177610
UIPDR5= 177612
UIPDR6= 177614
UIPDR7= 177616
UMTSTS 035422
USESTK= 000600
USP = 2000006
USP1 025654
JSP2 025750
USP2A 026040
USP3 026070
USP4 026120
USP5 026200
USP6 026230
USRM - 140000
USTBOT 042402
WAIT 042014
WATE 042324
XBIT1 021504
XBIT2 021520
XBIT3 021536
XCR1 025174
XCR2 025210
XCR3 025236
YBR 026666
YNTAB 042230
SAPTHD 000330
SCPUOP 000326
SDEVCT 000310
SENDAD 042046
SENV 000320
SENVN 000321
SEPIRO 042222
SERN = 001003
SEERROR= 000302
SETABL 000320
SETEND 000330
SFATAL 000302
SHIBTS 000330
SLPADR 042224
SLPERR 042226
SMAIL 000300
SMBADR 000332
SMSGAD 000314
SMSGLG 000316

SYMBOL TABLE

\$MSGTY	000300	\$SWREG	000322	\$TMP2	042216	\$STSTM=	000304	\$X	-	041472	
\$PASS	000306	\$TESTN	000304	\$TMP3	042220	\$UNIT	000312	\$XX	=	177667	
\$PASTM	000336	\$TMP0	042212	\$TN	=	000340	\$UNITM	000340	\$XXX	=	000666
\$SVPC	=	000200	\$TMP1	042214	\$STSM	000334	\$USWR	000324	.\$X	=	001100
\$SWR	=	000000									

. ABS. 042562 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 20364 WORDS (80 PAGES)

DYNAMIC MEMORY: 20034 WORDS (77 PAGES)

ELAPSED TIME: 00:38:35

CKKAAB.BIN,CKKAAB/CR/-SP/NL:TOC=CKKAAB.MLB/ML,CKKAAB.P11