



# PDP11/34

BASIC INSTRUCTION TEST  
CFKAAC0

AH-8042C-MC

COPYRIGHT © 75-78

FIGHE 2 OF 2

JAN 1979

**digital**

MADE IN USA

IDENTIFICATION

PRODUCT CODE: AC-8041C-MC  
PRODUCT NAME: CFKAACO 11/34 BSC INST TST  
PRODUCT DATE: 30-OCT-78  
MAINTAINER: DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES

COPYRIGHT (C) 1975, 1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL  
DEC

PDP  
DECUS

UNIBUS  
DECTAPE

MASSBUS

\*\*\*\*\*  
\* SUMMARY OF OPERATING INSTRUCTIONS \*  
\*\*\*\*\*

THE FOLLOWING PROCEDURE CAN BE USED TO RUN THIS DIAGNOSTIC IN A STANDARD CONFIGURATION WITH AT LEAST 4K 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 THE TITLE WITHIN THE 1ST SECOND AND END PASS REPEATABLY THEREAFTER AT APPROX. 10 SEC. INTERVALS UNTIL EXTERNALLY HALTED.
4. IF THE PROGRAM DOES NOT RUN AS DESCRIBED ABOVE, CONSULT THE FULL OPERATING INSTRUCTIONS WHICH FOLLOW.

1.0 GENERAL PROGRAM INFORMATION

1.1 PROGRAM PURPOSE

THIS DIAGNOSTIC PROGRAM IS DESIGNED TO BE A COMPREHENSIVE CHECK OF THE PDP-11/34 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 11/34 (E.G. RTT, RT1, WAIT, RESET, TRAP, EMT).

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE

PDP-11/34 PROCESSOR  
8K MEMORY -- THE PROGRAM USES LOCATIONS 0 - 26520

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-11/34 MICROCODE LISTING

PDP-11/34 ELECTRICAL SCHEMATICS

DIAGNOSTIC ENGINEERING PROJECT PLAN FOR 11/34

DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS PROGRAMMING PRACTICES  
DOCUMENT NO. 175-003-009-00

APT INTERFACE SPECIFICATION, REVISION 9.

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

NONE

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).

LOAD ADDRESS = 200  
START

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. LOAD ADDRESS = 204
4. START

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.

2.3 PROGRAM OPTIONS

THIS PROGRAM IS INTENDED TO BE A BASIC PROCESSOR TEST.  
IT IS INTENDED TO BE THE LOWEST LEVEL DIAGNOSTIC RUN.  
IT PROVIDES FOR NO SELECTABLE OPTIONS.

IN ORDER THAT THE TEST BE RUNNABLE ON A PROCESSOR WITHOUT A  
TELETYPE, IT IS POSSIBLE TO SUPPRESS THE END OF PASS MESSAGE.  
IF NO TELETYPE IS AVAILABLE, ALTER THE BYTE, \$ENVN, WHICH  
IS LOCATED IN THE APT MAILBOX. SETTING \$ENVN TO 40(8) WILL  
SUPPRESS ALL CONSOLE OUTPUT.  
THE EXACT LOCATION OF THIS BYTE CAN BE FOUND IN THE SYMBOL  
TABLE AT THE END OF THE LISTING.

2.4 EXECUTION TIMES

THE DIAGNOSTIC COMPLETES THE FIRST PASS IN LESS THAN 1 SEC.  
SUBSEQUENT PASSES REQUIRE APPROXIMATELY 10 SECS. EACH.  
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 TESTS 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  
TO IDENTIFY THE ERROR DETECTED.

CERTAIN FAILURES WILL CAUSE THE PROESSOR 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.)

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 TO DIAGNOSTIC INTERFACE SPECIFICATION. IT IS THE PRIMARY MEDIUM FOR IDENTIFYING ERRORS.

3.2.1 \$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.

3.2.2 \$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.

3.2.3 \$PASS

THIS LOCATION IS INCREMENTED FOR EVERY COMPLETE PASS OF THE DIAGNOSTIC. MONITORING THIS 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.

3.2.4 \$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.



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 THE SUBTEST IN WHICH THEY WERE DISCOVERED.

4.0 PROGRESS REPORT

AT THE END OF EACH SUCCESSFUL PASS (THE EQUIVALENT OF 400 (8) PROGRAM PASSES, EXCEPT THE FIRST PASS WHICH IS ONLY ONE PROGRAM PASS) THE PROGRAM INCREMENTS THE LOCATION \$PASS WHICH IS IN THE APT MAILBOX. THIS LOCATION WILL ALWAYS CONTAIN THE NUMBER OF SUCCESSFUL PASSES COMPLETED. \$PASS IS RESET WITH EVERY RETART FROM LOC. 200.

ADDITIONALLY, THE TITLE AND END PASS MESSAGE IS PRINTED ON THE CONSOLE TELETYPE AFTER THE FIRST PASS. THE END PASS MESSAGE IS REPEATED EVERY SUBSEQUENT PASS (400 PROGRAM LOOPS) THEREAFTER.

IF NO TELETYPE IS AVAILABLE, THE CONSOLE OUTPUT MUST BE SUPPRESSED. (SEE SECTION 2.3).

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 HAS BEEN 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 2. 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 NOT TELETYPE IS AVAILABLE TO REPORT THE PRINTED PROGRESS REPORTS.

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.

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).

14	ACT11 HOOKS
25	APT MAILBOX-ETABLE
52	APT PARAMETER BLOCK
130	T1 CHECK BRANCHES ON Z BIT
177	DATA PATH TESTS
193	T2 TEST OF ZEROES IN THE DATA PATH
213	T3 TEST OF PATTERN 125252 IN DATA PATH
233	T4 TEST OF PATTERN 052525 IN DATA PATH
253	T5 TEST OF ALL ONES IN DATA PATH
270	B-REGISTER TEST
287	T6 SHIFT BIT 0 TO BIT 1
308	T7 SHIFT CARRY INTO BIT 0
338	T10 LEFT SHIFT FROM BIT 0 TO C-BIT
363	T11 SHIFT BIT 15 TO BIT 14
384	T12 RIGHT SHIFT FROM BIT 15 TO C-BIT
407	SCRATCH PAD TESTS
436	T13 TEST IF R0 CAN HOLD ALL ZEROES
456	T14 TEST IF R0 CAN HOLD ONES AND ZEROES
475	T15 TEST IF R0 CAN HOLD ZEROES AND ONES
494	T16 TEST IF R0 CAN HOLD ALL ONES
513	T17 TEST IF R1 CAN HOLD A ONE IN ALL BITS
538	T20 TEST IF R1 CAN HOLD A ZERO IN ALL BITS
563	T21 TEST IF R2 CAN HOLD A ONE IN ALL BITS
588	T22 TEST IF R2 CAN HOLD A ZERO IN ALL BITS
611	T23 TEST IF R3 CAN HOLD A ONE IN ALL BITS
636	T24 TEST IF R3 CAN HOLD A ZERO IN ALL BITS
662	T25 TEST IF R4 CAN HOLD A ONE IN ALL BITS
687	T26 TEST IF R4 CAN HOLD A ZERO IN ALL BITS
714	T27 TEST IF R5 CAN HOLD A ONE IN ALL BITS
739	T30 TEST IF R5 CAN HOLD A ZERO IN ALL BITS
765	T31 TEST IF R6 CAN HOLD A ONE IN ALL BITS
790	T32 TEST IF R6 CAN HOLD A ZERO IN ALL BITS
814	PSW TESTS
831	T33 TEST IF PSW WILL HOLD ZEROES
851	T34 TEST IF PSW WILL HOLD ONES AND ZEROES
870	T35 TEST IF PSW (EXCEPT T-BIT) WILL HOLD ZEROES AND ONES
889	T36 TEST IF PSW (EXCEPT T-BIT) WILL HOLD ALL ONES
904	CONDITION CODE TEST
922	T37 TEST BRANCHES AROUND Z-BIT
970	T40 TEST BRANCHES AROUND N-BIT
1018	T41 TEST BRANCHES AROUND V-BIT
1066	T42 TEST BRANCHES AROUND C-BIT
1099	MICROCODE TESTS
1136	T43 TEST MODE 0 USING SOP INST.
1184	T44 TEST REMAINDER OF SOP INSTS IN MODE 0
1227	T45 TEST MODE 0 EVEN BYTE USING SOP INST
1265	T46 TEST MODE 1 USING SOP INST.
1304	T47 TEST MODE 1 EVEN BYTE USING SOP INST
1350	T50 TEST MODE 1 ODD BYTE USING SOP INST
1398	T51 TEST MODE 2 USING SOP INST.
1447	T52 TEST MODE 2 EVEN BYTE USING SOP INST.
1491	T53 TEST MODE 2 ODD BYTE USING SOP INST.
1538	T54 TEST MODE 0 USING NEGATE INSTRUCTION
1595	T55 TEST MODE 1 USING NEGATE INST.
1652	T56 TEST MODE 2 USING NEGATE INSTRUCTION
1714	T57 TEST MODE 3 USING SOP INST.

1762	T60	TEST MODE 3 EVEN BYTE USING SOP INST.
1817	T61	TEST MODE 3 ODD BYTE USING SOP INST.
1856	T62	TEST MODE 3 USING NEGATE INSTRUCTION
1933	T63	TEST MODE 4 USING SOP INSTS
1985	T64	TEST MODE 5 USING SOP INSTS
2028	T65	TEST MODE 6 USING SOP INSTS
2070	T66	TEST MODE 7 USING SOP INST.
2104	T67	TEST MODE 4 WITH NEGATE INSTRUCTION
2146	T70	TEST MODE 5 WITH NEGATE INSTRUCTION
2193	T71	TEST MODE 6 WITH NEGATE
2229	T72	TEST MODE 7 W/ NEGATE
2275	T73	TEST SOP INSTRUCTIONS MODES 2,3,6,7 WITH REGISTER 7
2316	T74	TEST MODE 0 SOP NON-MODIFYING
2349	T75	TEST MODE 0 EVEN BYTE W/ SOP NON-MODIFYING
2382	T76	TEST MODE 1 SOP NON-MODIFYING
2415	T77	TEST MODE 1 BYTE INST. NON-MODIFYING
2465	T100	TEST MODE 2 WITH SOP NON-MODIFYING
2509	T101	TEST MODE 2 - BYTE W/ SOP NON-MODIFYING
2577	T102	TEST MODE 3 W/ SOP NON-MODIFYING INSTS
2624	T103	TEST MODE 3 - BYTES W/ SOP NON-MODIFYING INSTS.
2685	T104	TEST MODE 4 W/ SOP NON-MODIFYING INSTS
2727	T105	TEST MODE 5 W/ SOP NON-MODIFYING INSTS
2772	T106	TEST MODE 6 W/ SOP NON-MODIFYING INSTS
2815	T107	TEST MODE 7 W/ SOP NON-MODIFYING INSTS.
2857	T110	TEST MODE 0 DOUBLE-OPERAND (DOP) INSTS.
2885	T111	MOV MODE 0 TO MODE 0
2913	T112	TEST SUB MODE 0,0
2955	T113	TEST ALL THE DOP INSTRUCTIONS W/ SOURCE MODE 0,0
3029	T114	TEST MODE 0,X DOUBLE-OPERAND INSTRUCTIONS
3071	T115	TEST DOP NON-MODIFYING INST. W/ SOURCE MODE 0,0
3137	T116	TEST MODE 0,X DOUBLE-OPERAND NON-MODIFYING INSTS.
3181	T117	TEST MODE 1 W/ DOP INST.
3210	T120	TEST MODE 1 - EVEN BYTE W/ DOP INSTS.
3240	T121	TEST MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING INST.
3274	T122	TEST MOV INSTRUCTION MODE 1,0 EVEN BYTE
3316	T123	TEST MODE 1-ODD BYTE W/ DOP INSTS.
3347	T124	TEST MODE 2 W/ DOP INSTS.
3388	T125	TEST MODE 2 - EVEN BYTE W/ DOP INST.
3425	T126	TEST MODE 2 - ODD BYTE W/ DOP INST.
3466	T127	TEST MODE 3 W/ DOP INSTS.
3493	T130	TEST MODE 3 - EVEN BYTE W/ DOP INSTS.
3520	T131	TEST MODE 3 - ODD BYTE W/ DOP INSTS.
3541	T132	TEST DEST. MODE 0-BYTE W/ DOP NON-MODIFYING INST
3575	T133	TEST DEST. MODE 1 W/ DOP NON-MODIFYING INST
3609	T134	TEST DEST. MODE 2 W/ DOP NON-MODIFYING INST.
3653	T135	TEST DEST. MODE 2-BYTE W/DOP NON-MODIFYING INST
3721	T136	TEST DEST. MODE 3-BYTES W/DOP NON-MODIFYING INST.
3783	T137	TEST DEST. MODE 4 W/DOP NON-MODIFYING INST.
3828	T140	TEST DEST. MODE 4-BYTE W/ DOP NON-MODIFYING INST.
3893	T141	TEST DEST. MODE 5 W/DOP NON-MODIFYING INST.
3938	T142	TEST DEST. MODE 6 W/DOP NON-MODIFYING INST.
3982	T143	TEST DEST. MODE 7 W/DOP NON-MODIFYING INST.
4032	T144	TEST MOV DESTINATION MODE 1
4072	T145	TEST MOV DESTINATION MODE 2
4121	T146	TEST MOV-BYTE DESTINATION MODE 2
4188	T147	TEST MOV(B) DESTINATION MODE 3

4256	T150	TEST MOV DESTINATION MODE 4
4306	T151	TEST MOVB DESTINATION MODE 4
4376	T152	TEST MOV DESTINATION MODE 5
4446	T153	TEST MOV DESTINATION MODE 6
4515	T154	TEST MOV DESTINATION MODE 7
4588	T155	TEST MODE 4 W/ DOP INSTS.
4627	T156	TEST MODE 5 W/ DOP INSTS.
4665	T157	TEST MODE 6 W/ DOP INSTS.
4697	T160	TEST MODE 7 W/ DOP INSTS.
4727	T161	TEST ROTATE INSTRUCTIONS OF MODE 0
4775	T162	TEST ROTATE INSTRUCTIONS W/ MODE 1
4839	T163	TEST ROTATE INSTRUCTIONS W/ MODE 2
4909	T164	TEST ROTATE INSTRUCTIONS /W MODE 3
4968	T165	TEST MODE 4 W/ ROTATE INSTRUCTIONS
5004	T166	TEST MODE 5 W/ ROTATE INSTRUCTIONS
5039	T167	TEST MODE 6 W/ ROTATE INSTRUCTIONS
5069	T170	TEST MODE 7 W/ ROTATE INSTRUCTIONS
5102	T171	TEST MODE 0 W/ SWAB INST.
5137	T172	TEST MODE 1 W/ SWAB INST
5166	T173	TEST MODE 2 W/ SWAB INST
5204	T174	TEST MODE 3 W/SWAB INST.
5232	T175	TEST MODE 4 W/ SWAB INST
5272	T176	TEST MODE 5 W/ SWAB INST.
5315	T177	TEST MODE 6 W/ SWAB INST.
5349	T200	TEST MODE 7 W/ SWAB INST.
5405	T201	TEST THE JMP INSTRUCTION IN ALL MODES
5541	T202	TEST JSR INSTRUCTION W/ ALL MODES
5698	T203	TEST RTS INSTRUCTION
5738	T204	TEST MOV INSTRUCTION
5775	T205	TEST BIT INSTRUCTION
5813	T206	TEST BIC INSTRUCTION
5850	T207	TEST BIS INSTRUCTION
5901	T210	TEST INC INSTRUCTION
5956	T211	TEST DEC INSTRUCTION
6034	T212	TEST CLR INSTRUCTION
6058	T213	TEST TST INSTRUCTION
6096	T214	TEST SWAB INSTRUCTION
6144	T215	TEST ADD INSTRUCTION
6222	T216	TEST ADC INSTRUCTION
6286	T217	TEST NEG INSTRUCTION
6343	T220	TEST CMP INSTRUCTION
6412	T221	TEST COM INSTRUCTION
6447	T222	TEST SUB INSTRUCTION
6515	T223	TEST SBC INSTRUCTION
6595	T224	TEST ROL INSTRUCTION
6664	T225	TEST ROR INSTRUCTION
6732	T226	TEST ASL INSTRUCTION
6802	T227	TEST ASR INSTRUCTION
6886	T230	TEST THE SXT INSTRUCTION
6940	T231	TEST THE XOR INSTRUCTION
6992	T232	TEST SOB INSTRUCTION
7037	T233	TEST MARK INSTRUCTION
7101	T234	TEST MTPS INSTRUCTION
7138	T235	TEST MTPS MODE 2
7169	T236	TEST MTPS MODE 3
7201	T237	TEST MTPS MODE 4

7232	T240	TEST MTPS MODE 5
7263	T241	TEST MTPS MODE 6
7294	T242	TEST MTPS MODE 7
7333	T243	TEST MFPS INSTRUCTION
7367	T244	TEST MFPS MODE 2
7410	T245	TEST MFPS MODE 3
7453	T246	TEST MFPS MODE 4
7496	T247	TEST MFPS MODE 5
7539	T250	TEST MFPS MODE 6
7582	T251	TEST MFPS MODE 7
7633	T252	TEST THAT RESET DOES NOT CLEAR PSW
7661	T253	TEST USER MODE R6 CAN HOLD A ONE IN EVERY POSITION
7687	T254	TEST INDEPENDENCE OF USER AND KERNEL MODE R6'S
7728	T255	TEST MFPI WITH R6 IN MODE 0
7753	T256	TEST MTPI WITH R6 IN MODE 0
7808	T257	TEST THE BRANCH ROM
7866	T260	DUAL REGISTER ADDRESSING TEST
7917	T261	TEST BYTE INSTRUCTION ON PSW
7941	T262	TEST THAT JMP INSTRUCTION DOES NOT AFFECT CONDITION CODES
7978	T263	TEST SET CC AND CLEAR CC INSTRUCTIONS
8030	T264	END OF PASS SEQUENCE

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

```
.TITLE CFKAACO 1.1/34 BSC INST TST
.ENABLE ABS
STBOT=500
.NLIST CND,MC,MD
.LIST ME
SCOPE=NOP
R7=%7
R6=%6
PS=177776
TPS=177564
TPB=177566
USRM=140000
PUSRM=30000
.SBTTL ACT11 HOOKS

:*****
:HOOKS REQUIRED BY ACT11
$SVPC=. ;SAVE PC
.=46
$ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
.=52 ;;2)SET LOC.52 TO ZERO
.$SVPC ;; RESTORE PC
.=300
.SBTTL APT MAILBOX-ETABLE

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

:*****
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
:*****
```



```
57          000330          .SX=      ;;SAVE CURRENT LOCATION
58          000024          .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
59 000024    000200          200        ;;FOR APT START UP
60          000044          .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
61 000044    000330          $APTHDR  ;;POINT TO APT HEADER BLOCK
62          000330          .=.SX     ;;RESET LOCATION COUNTER
63          *****
64          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
65          ;INTERFACE SPEC.
66
67 000330    $APTHD:
68 000330    000000    $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
69 000332    000300    $MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
70 000334    000010    $TSTM:  .WORD 10   ;;RUN TIM OF LONGEST TEST
71 000336    000010    $PASTM: .WORD 10   ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
72 000340    000000    $UNITM: .WORD 0    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
73 000342    000014    .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
74          *****
75          ;SOME POINTERS TO CPU TRAP HANDLERS
76          *****
77          .=4
78 000004    026424    T04
79 000006    000000    0
80 000010    026434    T010
81 000012    000000    0
82 000014    026444    T014
83          000030    .=30
84 000030    026454    T030
85 000032    000000    0
86 000034    026464    T034
87 000036    000000    0
88          000114    .=114
89 000114    026474    T0114
90 000116    000000    0
91          000244    .=244
92 000244    026504    T0244
93 000246    000000    0
94 000250    026514    T0250
95 000252    000000    0
96
97          *****
98          ;DATA TABLE FOR USE IN ADDRESSING MODE TESTS
99          *****
100         .=370
101 000370    000000    000000    000000    0,0,0,0,0,0
102 000376    000000    000000    000000
103 000404    000001    000001    177777    1,1,-1
104         .=500
105          *****
106          ;SET UP STARTING ADDRESS
107         .SX=
108         .=200
109 000200    000167    000274    JMP      START
110
111 000204    012706    000500    MOV     #JTBOT,R6      ;SET STACK POINTER
112 000210    012702    000304    MOV     #$TESTN,R2     ;SET MAILBOX POINTER
```

```
113 000214 000137          JMP    @ (PC)+          ; JUMP TO SUBTEST
114 000216 000000          0                      ; ADDR. OF SUBTEST GOES HERE
115
116          000500          . = . $X
117          000302          $ERROR=$FATAL
118          000304          $TSTNM=$TESTN
119 000500 012737 026310 000024  START:  MOV    #PWRDN,@#24          ; SET UP FOR POWER FAIL
120 000506 012737 000000 000306          MOV    #0,@#$PASS          ; CLEAR PASS COUNT
121 000514 012737 177777 026060          MOV    #-1,@#PASSPT        ; SET PRINT COUNTER
122 000522 012706 000500          RESTRT: MOV   #STBOT,R6      ; INITIALIZE STACK POINTER
123 000526 012702 000304          MOV    #$TESTN,R2         ; SET UP POINTER TO MESSAGE TYPE
124 000532 012737 000000 000304          MOV    #0,@#$TSTNM        ; CLEAR TEST NUMBER
125 000540 012737 000000 000302          MOV    #0,@#$ERROR        ; CLEAR ERROR NUMBER
126 000546 012737 000000 000300          MOV    #0,@#$MSGTY        ; CLEAR MESSAGE TYPE (FOR APT)
```

```
127 :*****  
128 :TEST 1 CHECK BRANCHES ON Z BIT  
129 :*****  
130 000554 005212 TST1: INC (R2) ;UPDATE TEST NUMBER  
131 000556 022712 000001 CMP #1,(R2) ;SEQUENCE ERROR?  
132 000562 001024 BNE TST2-10 ;BR TO ERROR HALT ON SEQ ERROR  
133 000564 000257 CCC ;CLEAR ALL CONDITION CODES  
134 000566 001401 BEQ BR1 ;SHOULD BRANCH  
135 000570 000404 BR BR2 ;BAD BRANCH OF Z-BIT  
136 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
137 : BRANCH INSTRUCTION AND <=====  
138 : REPLACE THE MOVE INSTRUCTION <=====  
139 : FOLLOWING W/ 774 <=====  
140 000572 BR1: MOV #1,-(R2) ;MOVE TO MAILBOX # ***** 1 *****  
141 000572 012742 000001 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
142 000576 005242 HALT ;SHOULD HAVE BRANCHED: Z=0  
143 000600 000000 BR2: BNE BR3  
144 000602 001004 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
145 000602 001004 : CONDITIONAL BRANCH INST. AND <=====  
146 : REPLACE THE MOVE INSTRUCTION <=====  
147 : WHICH FOLLOWS W/ 770 <=====  
148 000604 012742 000002 BR3: MOV #2,-(R2) ;MOVE TO MAILBOX # ***** 2 *****  
149 000610 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
150 000612 000000 HALT  
151 000614 000264 SEZ  
152 000616 001001 BNE BR4  
153 000620 000404 BR BR5  
154 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
155 : BRANCH INSTRUCTION AND <=====  
156 : REPLACE THE MOVE INSTRUCTION <=====  
157 : FOLLOWING W/ 760 <=====  
158 000622 BR4: MOV #3,-(R2) ;MOVE TO MAILBOX # ***** 3 *****  
159 000622 012742 000003 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
160 000626 005242 HALT ;SHOULD NOT HAVE BRANCHED HERE ON Z=1  
161 000630 000000 BR5: BEQ TST2  
162 000632 001404 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
163 : CONDITIONAL BRANCH INST. AND <=====  
164 : REPLACE THE MOVE INSTRUCTION <=====  
165 : WHICH FOLLOWS W/ 754 <=====  
166 000634 012742 000004 BR5: MOV #4,-(R2) ;MOVE TO MAILBOX # ***** 4 *****  
167 000640 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
168 000642 000000 HALT ;SHOULD HAVE BRANCHED ON Z=1  
169 : OR SEQUENCE ERROR  
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  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225

\*\*\*\*\*  
:SBTTL 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  
: DATA TRANSCIEVERS, AND AMUX CONTROL FOR ALU AND UBUS 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.

\*\*\*\*\*  
:TEST 2 TEST OF ZEROES IN THE DATA PATH

\*\*\*\*\*  
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/ 772 <====  
MOV #5,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 5 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;DATA INCORRECT  
; OR SEQUENCE ERROR

\*\*\*\*\*  
:TEST 3 TEST OF PATTERN 125252 IN DATA PATH

\*\*\*\*\*  
TST3: INC (R2) ;UPDATE TEST NUMBER  
CMP #3,(R2) ;SEQUENCE ERROR?  
BNE TST4-10 ;BR TO ERROR HALT ON SEQ ERROR  
MOV #125252,@#0 ;MOVE ALTERNATING ONES AND ZEROES  
;THRU DATA PATHS  
CMP #125252,@#0 ;SUCCESSFUL  
BEQ TST4  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 771 <====  
MOV #6,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 6 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;DATA INCORRECT  
; OR SEQUENCE ERROR

```
226  
227  
228 :*****  
229 :TEST 4 TEST OF PATTERN 052525 IN DATA PATH  
230 :*****  
230 000736 005212 TST4: INC (R2) ;UPDATE TEST NUMBER  
231 000740 022712 000004 CMP #4,(R2) ;SEQUENCE ERROR?  
232 000744 001007 BNE TST5-10 ;BR TO ERROR HALT ON SEQ ERROR  
233 000746 012737 052525 000000 MOV #052525,@#0 ;MOVE ALTERNATING ZEROES AND ONES  
234 ;THRU DATA PATH  
235 000754 022737 052525 000000 CMP #052525,@#0 ;SUCCESSFUL?  
236 000762 001404 BEQ TST5  
237 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
238 ; CONDITIONAL BRANCH INST. AND <====  
239 ; REPLACE THE MOVE INSTRUCTION <====  
240 ; WHICH FOLLOWS W/ 771 <====  
241 000764 012742 000007 MOV #7,-(R2) ;MOVE TO MAILBOX # ***** 7 *****  
242 000770 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
243 000772 000000 HALT ;DATA INCORRECT  
244 ; OR SEQUENCE ERROR  
245  
246 :*****  
247 :TEST 5 TEST OF ALL ONES IN DATA PATH  
248 :*****  
249 000774 005212 TST5: INC (R2) ;UPDATE TEST NUMBER  
250 000776 022712 000005 CMP #5,(R2) ;SEQUENCE ERROR?  
251 001002 001007 BNE TST6-10 ;BR TO ERROR HALT ON SEQ ERROR  
252 001004 012737 177777 000000 MOV #177777,@#0 ;MOVE ONES THRU DATA PATH  
253 001012 022737 177777 000000 CMP #177777,@#0 ;SUCCESSFUL  
254 001020 001404 BEQ TST6  
255 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
256 ; CONDITIONAL BRANCH INST. AND <====  
257 ; REPLACE THE MOVE INSTRUCTION <====  
258 ; WHICH FOLLOWS W/ 771 <====  
259 001022 012742 000010 MOV #10,-(R2) ;MOVE TO MAILBOX # ***** 10 *****  
260 001026 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
261 001030 000000 HALT ;DATA INCORRECT  
262 ; OR SEQUENCE ERROR
```

263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318

001032 005212  
001034 022712 000006  
001040 001012  
001042 000241  
001044 012737 000001 000000  
001052 006137 000000  
001056 022737 000002 000000  
001064 001404  
  
001066 012742 000011  
001072 005242  
001074 000000  
  
001076 005212  
001100 022712 000007  
001104 001017  
001106 012737 000000 000000  
001114 000261  
001116 006137 000000  
001122 103014  
  
001124 012742 000012  
001130 005242  
001132 000000  
  
001134 022737 000001 000000  
001142 001404

```
*****  
: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.  
*****  
:TEST 6 SHIFT BIT 0 TO BIT 1  
*****  
TST6: INC (R2) :UPDATE TEST NUMBER  
CMP #6,(R2) :SEQUENCE ERROR?  
BNE TST7-10 :BR TO ERROR HALT ON SEQ ERROR  
CLC :CLEAR CARRY BIT  
MOV #1,@#0 :LOAD A 1  
ROL @#0 :SHIFT LEFT  
CMP #2,@#0 :SUCCESSFUL  
BEQ TST7  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 766 <=====  
MOV #11,-(R2) :MOVE TO MAILBOX # ***** 11 *****  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :BIT 1 NOT SET  
: OR SEQUENCE ERROR  
  
*****  
:TEST 7 SHIFT CARRY INTO BIT 0  
*****  
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/ 771 <=====  
MOV #12,-(R2) :MOVE TO MAILBOX # ***** 12 *****  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :CARRY CLEAR  
: OR SEQUENCE ERROR  
CMP #1,@#0 :BIT 0 SET  
BEQ TST10
```

319 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
320 ; CONDITIONAL BRANCH INST. AND <====  
321 ; REPLACE THE MOVE INSTRUCTION <====  
322 ; WHICH FOLLOWS W/ 761 <====  
323 001144 012742 000013 MOV #13,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 13 \*\*\*\*\*  
324 001150 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
325 001152 000000 HALT ;BIT 0 NOT SET  
326 ; OR SEQUENCE ERROR

327  
328 :\*\*\*\*\*  
329 :TEST 10 LEFT SHIFT FROM BIT 0 TO C-BIT  
330 :\*\*\*\*\*

331 001154 005212 TST10: INC (R2) ;UPDATE TEST NUMBER  
332 001156 022712 000010 CMP #10,(R2) ;SEQUENCE ERROR?  
333 001162 001014 BNE TST11-10 ;BR TO ERROR HALT ON SEQ ERROR  
334 001164 012737 000001 000000 MOV #1,@#0 ;SET BIT 0  
335 001172 012700 177757 MOV #-21,R0 ;SET BIT COUNTER  
336 001176 000241 CLC ;CLEAR C-BIT  
337 001200 005200 SHL: INC R0 ;INCREMENT BIT COUNTER  
338 001202 001404 BEQ SHLE ;BR TO ERROR HALT IF BIT IS LOST  
339 001204 006137 000000 ROL @#0 ;SHIFT LEFT ONE POSITION  
340 001210 103373 BCC SHL ;BRANCH IF C-BIT NOT SET  
341 001212 001404 BEQ TST11

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

346 001214 SHLE: MOV #14,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 14 \*\*\*\*\*  
347 001214 012742 000014 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
348 001220 005242 HALT ;LEFT SHIFTING LOGIC FAILED  
349 001222 000000 ; OR SEQUENCE ERROR

350  
351 :\*\*\*\*\*  
352 :TEST 11 SHIFT BIT 15 TO BIT 14  
353 :\*\*\*\*\*  
354 :\*\*\*\*\*

355 001224 005212 TST11: INC (R2) ;UPDATE TEST NUMBER  
356 001226 022712 000011 CMP #11,(R2) ;SEQUENCE ERROR?  
357 001232 001012 BNE TST12-10 ;BR TO ERROR HALT ON SEQ ERROR  
358 001234 012737 100000 000000 MOV #100000,@#0 ;SET BIT 15  
359 001242 000241 CLC ;CLEAR CARRY  
360 001244 006037 000000 ROR @#0 ;SHIFT BIT 15 TO BIT 14  
361 001250 022737 040000 000000 CMP #40000,@#0 ;SUCCESSFUL  
362 001256 001404 BEQ TST12

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

367 001260 012742 000015 MOV #15,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 15 \*\*\*\*\*  
368 001264 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
369 001266 000000 HALT ;BIT 14 NOT SET  
370 ; OR SEQUENCE ERROR

371 :\*\*\*\*\*  
372 :TEST 12 RIGHT SHIFT FROM BIT 15 TO C-BIT  
373 :\*\*\*\*\*  
374 :\*\*\*\*\*

375	001270	005212		TST12:	INC	(R2)	:UPDATE TEST NUMBER	
376	001272	022712	000012		CMP	#12,(R2)	:SEQUENCE ERROR?	
377	001276	001014			BNE	TST13-10	:BR TO ERROR HALT ON SEQ ERROR	
378	001300	012737	100000	000000	MOV	#100000,@#0	:SET BIT 15	
379	001306	012700	177757		MOV	#-21,R0	:SET BIT COUNTER	
380	001312	000241			CLC		:CLEAR C-BIT	
381	001314	005200		SHR:	INC	R0	:INCREMENT BIT COUNTER	
382	001316	001404			BEQ	SHRE	:BR TO ERROR HALT IF BIT IS LOST	
383	001320	006037	000000		ROR	@#0	:ROTATE RIGHT ONE POSITION	
384	001324	103373			BCC	SHR	:BRANCH IF C-BIT CLEAR	
385	001326	001404			BEQ	TST13		
386							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS	<====
387							: CONDITIONAL BRANCH INST. AND	<====
388							: REPLACE THE MOVE INSTRUCTION	<====
389							: WHICH FOLLOWS W/ 764	<====
390	001330			SHRE:	MOV	#16,-(R2)	:MOVE TO MAILBOX # ***** 16 *****	
391	001330	012742	000016		INC	-(R2)	:SET MSGTYP TO FATAL ERROR	
392	001334	005242			HALT		:RIGHT SHIFT LOGIC FAILED	
393	001336	000000					: OR SEQUENCE ERROR	
394								



395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450

\*\*\*\*\*  
: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.

\*\*\*\*\*  
:TEST 13 TEST IF R0 CAN HOLD ALL ZEROES

\*\*\*\*\*  
TST13: INC (R2) ;UPDATE TEST NUMBER  
CMP #13,(R2) ;SEQUENCE ERROR?  
BNE TST14-10 ;BR TO ERROR HALT ON SEQ ERROR  
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/ 774 <====  
MOV #17,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 17 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;R0 NOT 0  
: OR SEQUENCE ERROR

\*\*\*\*\*  
:TEST 14 TEST IF R0 CAN HOLD ONES AND ZEROES

\*\*\*\*\*  
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

```
451  
452  
453  
454  
455 001412 012742 000020      MOV    #20,-(R2)      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
456 001416 005242              INC    -(R2)          ;          CONDITIONAL BRANCH INST. AND <====  
457 001420 000000              HALT                   ;          REPLACE THE MOVE INSTRUCTION <====  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473 001444 012742 000021      MOV    #21,-(R2)      ;          WHICH FOLLOWS W/ 773 <====  
474 001450 005242              INC    -(R2)          ;          MOVE TO MAILBOX # ***** 20 ***** <====  
475 001452 000000              HALT                   ;          SET MSGTYP TO FATAL ERROR <====  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491 001476 012742 000022      MOV    #22,-(R2)      ;          RO NOT 125252 <====  
492 001502 005242              INC    -(R2)          ;          OR SEQUENCE ERROR <====  
493 001504 000000              HALT                   ;
```

```
*****  
:TEST 15      TEST IF RO CAN HOLD ZEROES AND ONES  
*****
```

```
463 001422 005212 000015      TST15: INC    (R2)          ;UPDATE TEST NUMBER  
464 001424 022712              CMP    #15,(R2)       ;SEQUENCE ERROR?  
465 001430 001005              BNE   TST16-10        ;BR TO ERROR HALT ON SEQ ERROR  
466 001432 012700 052525      MOV    #052525,R0     ;MOVE ALTERNATING ZEROES AND ONES TO RO  
467 001436 020027 052525      CMP    R0,#052525     ;SUCCESSFUL?  
468 001442 001404              BEQ   TST16
```

```
469  
470  
471  
472  
473 001444 012742 000021      MOV    #21,-(R2)      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
474 001450 005242              INC    -(R2)          ;          CONDITIONAL BRANCH INST. AND <====  
475 001452 000000              HALT                   ;          REPLACE THE MOVE INSTRUCTION <====  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491 001476 012742 000022      MOV    #22,-(R2)      ;          WHICH FOLLOWS W/ 773 <====  
492 001502 005242              INC    -(R2)          ;          MOVE TO MAILBOX # ***** 21 ***** <====  
493 001504 000000              HALT                   ;          SET MSGTYP TO FATAL ERROR <====  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506
```

```
*****  
:TEST 16      TEST IF RO CAN HOLD ALL ONES  
*****
```

```
481 001454 005212 000016      TST16: INC    (R2)          ;UPDATE TEST NUMBER  
482 001456 022712              CMP    #16,(R2)       ;SEQUENCE ERROR?  
483 001462 001005              BNE   TST17-10        ;BR TO ERROR HALT ON SEQ ERROR  
484 001464 012700 177777      MOV    #177777,R0     ;MOVE ALL ONES TO RO  
485 001470 020027 177777      CMP    R0,#177777     ;SUCCESSFUL?  
486 001474 001404              BEQ   TST17
```

```
487  
488  
489  
490  
491 001476 012742 000022      MOV    #22,-(R2)      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
492 001502 005242              INC    -(R2)          ;          CONDITIONAL BRANCH INST. AND <====  
493 001504 000000              HALT                   ;          REPLACE THE MOVE INSTRUCTION <====  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506
```

```
*****  
:TEST 17      TEST IF R1 CAN HOLD A ONE IN ALL BITS  
*****
```

```
499 001506 005212 000017      TST17: INC    (R2)          ;UPDATE TEST NUMBER  
500 001510 022712              CMP    #17,(R2)       ;SEQUENCE ERROR?  
501 001514 001012              BNE   TST20-10        ;BR TO ERROR HALT ON SEQ ERROR  
502 001516 012701 000001      MOV    #1,R1          ;SET BIT 0  
503 001522 012700 177757      MOV    #-21,R0        ;SET BIT COUNTER  
504 001526 000241              CLC                   ;CLEAR C-BIT  
505 001530 005200      REG1: INC    R0          ;INCREMENT BIT COUNTER  
506 001532 001403              BEQ   REG1E           ;BR TO ERROR HALT IF BIT IS LOST
```

```
507 001534 006101 ROL R1 ;ROTATE 1 POSITION
508 001536 103374 BCC REG1 ;ALL DONE
509 001540 001404 BEQ TST20
510 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
511 ; CONDITIONAL BRANCH INST. AND <=====
512 ; REPLACE THE MOVE INSTRUCTION <=====
513 ; WHICH FOLLOWS W/ 766 <=====
514 001542 REG1E:
515 001542 012742 000023 MOV #23,-(R2) ;MOVE TO MAILBOX # ***** 23 *****
516 001546 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
517 001550 000000 HALT ;FAILURE WITH R1
518 ; OR SEQUENCE ERROR
519
520 ;*****
521 ;TEST 20 TEST IF R1 CAN HOLD A ZERO IN ALL BITS
522 ;*****
523 001552 005212 ST20: INC (R2) ;UPDATE TEST NUMBER
524 001554 022712 000020 CMP #20,(R2) ;SEQUENCE ERROR?
525 001560 001014 BNE TST21-10 ;BR TO ERROR HALT ON SEQ ERROR
526 001562 012701 177776 MOV #-2,R1 ;SET ALL ONES IN R1 EXCEPT FOR BIT 0
527 001566 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
528 001572 000261 SEC ;SET C-BIT
529 001574 005200 REG1A: INC R0 ;INCREMENT COUNTER
530 001576 001405 BEQ R1ERR ;BR TO ERROR HALT IF COUNTER=0
531 001600 006101 ROL R1 ;ROTATE 1 POSITION
532 001602 103774 BCS REG1A ;CONTINUE UNTIL C-BIT IS CLEAR
533 001604 022701 177777 CMP #-1,R1 ;CHECK DATA IN R1
534 001610 001404 BEQ TST21
535 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
536 ; CONDITIONAL BRANCH INST. AND <=====
537 ; REPLACE THE MOVE INSTRUCTION <=====
538 ; WHICH FOLLOWS W/ 764 <=====
539 001612 R1ERR:
540 001612 012742 000024 MOV #24,-(R2) ;MOVE TO MAILBOX # ***** 24 *****
541 001616 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
542 001620 000000 HALT ;FAILURE WITH R1
543 ; OR SEQUENCE ERROR
544
545 ;*****
546 ;TEST 21 TEST IF R2 CAN HOLD A ONE IN ALL BITS
547 ;*****
547 001622 005212 TST21: INC (R2) ;UPDATE TEST NUMBER
548 001624 022712 000021 CMP #21,(R2) ;SEQUENCE ERROR?
549 001630 001012 BNE REG2A-14 ;BR TO ERROR HALT ON SEQ ERROR
550 001632 012702 000001 MOV #1,R2 ;SET BIT 0
551 001636 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
552 001642 000241 CLC ;CLEAR C-BIT
553 001644 005200 REG2: INC R0 ;INCREMENT BIT COUNTER
554 001646 001403 BEQ REG2A-14 ;BR TO ERROR HALT IF BIT IS LOST
555 001650 006102 ROL R2 ;ROTATE 1 POSITION
556 001652 103374 BCC REG2 ;ALL DONE
557 001654 001406 BEQ REG2A
558 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
559 ; BRANCH INSTRUCTION AND <=====
560 ; REPLACE THE MOVE INSTRUCTION <=====
561 ; FOLLOWING W/ 771 <=====
562 001656 012702 000304 MOV #STESTN,R2 ;RESTORE POINTER
```

563 001662 012742 000025       MOV     #25,-(R2)       ;MOVE TO MAILBOX # \*\*\*\*\* 25 \*\*\*\*\*  
564 001666 005242       INC     -(R2)       ;SET MSGTYP TO FATAL ERROR  
565 001670 000000       HALT               ;FAILURE WITH R2  
566 001672 012702 000304   REG2A: MOV     #\$TESTN,R2       ;RESTORE POINTER

567  
568  
569       ;\*\*\*\*\*  
570       ;TEST 22       TEST IF R2 CAN HOLD A ZERO IN ALL BITS  
571       ;\*\*\*\*\*

571 001676 005212       TST22: INC     (R2)       ;UPDATE TEST NUMBER  
572 001700 022712 000022       CMP     #22,(R2)       ;SEQUENCE ERROR?  
573 001704 001020       BNE     TST23-10       ;BR TO ERROR HALT ON SEQ ERROR  
574 001706 012702 177776       MOV     #-2,R2       ;SET ALL ONES IN R2 EXCEPT FOR BIT 0  
575 001712 012700 177757       MOV     #-21,R0       ;SET BIT COUNTER  
576 001716 000261       SEC               ;SET C-BIT  
577 001720 005200       REG2B: INC     R0       ;INCREMENT BIT COUNTER  
578 001722 001407       BEQ     R2ERR       ;BR TO ERROR HALT IF COUNTER=0  
579 001724 006102       ROL     R2       ;ROTATE 1 POSITION  
580 001726 103774       BCS     REG2B       ;CONTINUE UNTIL C-BIT IS CLEAR  
581 001730 022702 177777       CMP     #-1,R2       ;CHECK DATA IN R2  
582 001734 001406       BEQ     REG2C  
583 001736 012702 000304       MOV     #\$TESTN,R2       ;RESTORE POINTER

584 001742  
585 001742 012742 000026       R2ERR: MOV     #26,-(R2)       ;MOVE TO MAILBOX # \*\*\*\*\* 26 \*\*\*\*\*  
586 001746 005242       INC     -(R2)       ;SET MSGTYP TO FATAL ERROR  
587 001750 000000       HALT               ;FAILURE WITH R2  
588 001752 012702 000304   REG2C: MOV     #\$TESTN,R2       ;RESTORE POINTER

589  
590       ;\*\*\*\*\*  
591       ;TEST 23       TEST IF R3 CAN HOLD A ONE IN ALL BITS  
592       ;\*\*\*\*\*

593 001756 005212       TST23: INC     (R2)       ;UPDATE TEST NUMBER  
594 001760 022712 000023       CMP     #23,(R2)       ;SEQUENCE ERROR?  
595 001764 001012       BNE     TST24-10       ;BR TO ERROR HALT ON SEQ ERROR  
596 001766 012703 000001       MOV     #1,R3       ;SET BIT 0  
597 001772 012700 177757       MOV     #-21,R0       ;SET BIT COUNTER  
598 001776 000241       CLC               ;CLEAR C-BIT  
599 002000 005200       REG3:  INC     R0       ;INCREMENT BIT COUNTER  
600 002002 001403       BEQ     REG3E       ;BR TO ERROR HALT IF BIT IS LOST  
601 002004 006103       ROL     R3       ;ROTATE 1 POSITION  
602 002006 103374       BCC     REG3       ;ALL DONE  
603 002010 001404       BEQ     TST24

604  
605       ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS       <====  
606       ;       CONDITIONAL BRANCH INST. AND       <====  
607       ;       REPLACE THE MOVE INSTRUCTION       <====  
608       ;       WHICH FOLLOWS W/ 766       <====

609 002012       REG3E: MOV     #27,-(R2)       ;MOVE TO MAILBOX # \*\*\*\*\* 27 \*\*\*\*\*  
610 002016 012742 000027       INC     -(R2)       ;SET MSGTYP TO FATAL ERROR  
611 002020 000000       HALT               ;FAILURE WITH R3  
612       ; OR SEQUENCE ERROR

613  
614       ;\*\*\*\*\*  
615       ;TEST 24       TEST IF R3 CAN HOLD A ZERO IN ALL BITS  
616       ;\*\*\*\*\*

617 002022 005212       TST24: INC     (R2)       ;UPDATE TEST NUMBER  
618 002024 022712 000024       CMP     #24,(R2)       ;SEQUENCE ERROR?

```
619 002030 001014 BNE TST25-10 ;BR TO ERROR HALT ON SEQ ERROR
620 002032 012703 177776 MOV #-2,R3 ;SET ALL ONES IN R3 EXCEPT FOR BIT 0
621 002036 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
622 002042 000261 SEC ;SET C-BIT
623 002044 005200 REG3A: INC R0 ;INCREMENT BIT COUNTER
624 002046 001405 BEQ R3ERR ;BR TO ERROR HALT IF COUNTER=0
625 002050 006103 ROL R3 ;ROTATE 1 POSITION
626 002052 103774 BCS REG3A ;CONTINUE UNTIL C-BIT IS CLEAR
627 002054 022703 177777 CMP #-1,R3 ;CHECK DATA
628 002060 001404 BEQ TST25
```

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

```
633 002062 R3ERR:
634 002062 012742 000030 MOV #30,-(R2) ;MOVE TO MAILBOX # ***** 30 *****
635 002066 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
636 002070 000000 HALT ;FAILURE WITH R3
637 ; OR SEQUENCE ERROR
```

```
638
639
640 ;*****
641 ;TEST 25 TEST IF R4 CAN HOLD A ONE IN ALL BITS
642 ;*****
```

```
642 002072 005212 TST25: INC (R2) ;UPDATE TEST NUMBER
643 002074 022712 000025 CMP #25,(R2) ;SEQUENCE ERROR?
644 002100 001012 BNE TST26-10 ;BR TO ERROR HALT ON SEQ ERROR
645 002102 012704 000001 MOV #1,R4 ;SET BIT 0
646 002106 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
647 002112 000241 CLC ;CLEAR C-BIT
648 002114 005200 REG4: INC R0 ;INCREMENT BIT COUNTER
649 002116 001403 BEQ REG4E ;BR TO ERROR HALT IF BIT IS LOST
650 002120 006104 ROL R4 ;ROTATE 1 POSITION
651 002122 103374 BCC REG4 ;ALL DONE
652 002124 001404 BEQ TST26
```

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

```
657 002126 REG4E:
658 002126 012742 000031 MOV #31,-(R2) ;MOVE TO MAILBOX # ***** 31 *****
659 002132 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
660 002134 000000 HALT ;FAILURE WITH R4
661 ; OR SEQUENCE ERROR
```

```
662
663 ;*****
664 ;TEST 26 TEST IF R4 CAN HOLD A ZERO IN ALL BITS
665 ;*****
```

```
666 002136 005212 TST26: INC (R2) ;UPDATE TEST NUMBER
667 002140 022712 000026 CMP #26,(R2) ;SEQUENCE ERROR?
668 002144 001014 BNE TST27-10 ;BR TO ERROR HALT ON SEQ ERROR
669 002146 012704 177776 MOV #-2,R4 ;SET ALL ONES IN R4 EXCEPT FOR BIT 0
670 002152 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
671 002156 000261 SEC ;SET C-BIT
672 002160 005200 REG4A: INC R0 ;INCREMENT BIT COUNTER
673 002162 001405 BEQ R4ERR ;BR TO ERROR HALT IF COUNTER=0
674 002164 006104 ROL R4 ;ROTATE 1 POSITION
```

```
675 002166 103774 BCS REG4A ;CONTINUE UNTIL C-BIT IS CLEAR
676 002170 022704 177777 CMP #-1,R4 ;CHECK DATA
677 002174 001404 BEQ TST27
678 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
679 ; CONDITIONAL BRANCH INST. AND <====
680 ; REPLACE THE MOVE INSTRUCTION <====
681 ; WHICH FOLLOWS W/ 764 <====
682 002176 R4ERR:
683 002176 012742 000032 MOV #32,-(R2) ;MOVE TO MAILBOX # ***** 32 *****
684 002202 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
685 002204 000000 HALT ;FAILURE WITH R4
686 ; OR SEQUENCE ERROR
687
688
689
```

```
690 ;*****
691 ;TEST 27 TEST IF R5 CAN HOLD A ONE IN ALL BITS
692 ;*****
693 002206 005212 TST27: INC (R2) ;UPDATE TEST NUMBER
694 002210 022712 000027 CMP #27,(R2) ;SEQUENCE ERROR?
695 002214 001012 BNE TST30-10 ;BR TO ERROR HALT ON SEQ ERROR
696 002216 012705 000001 MOV #1,R5 ;SET BIT 0
697 002222 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
698 002226 000241 ;CLEAR C-BIT
699 002230 005200 REG5: INC R0 ;INCREMENT BIT COUNTER
700 002232 001403 BEQ REG5E ;BR TO ERROR HALT IF BIT IS LOST
701 002234 006105 ROL R5 ;ROTATE 1 POSITION
702 002236 103374 BCC REG5 ;ALL DONE
703 002240 001404 BEQ TST30
```

```
704 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
705 ; CONDITIONAL BRANCH INST. AND <====
706 ; REPLACE THE MOVE INSTRUCTION <====
707 ; WHICH FOLLOWS W/ 766 <====
708 002242 REG5E:
709 002242 012742 000033 MOV #33,-(R2) ;MOVE TO MAILBOX # ***** 33 *****
710 002246 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
711 002250 000000 HALT ;FAILURE WITH R5
712 ; OR SEQUENCE ERROR
713
```

```
714 ;*****
715 ;TEST 30 TEST IF R5 CAN HOLD A ZERO IN ALL BITS
716 ;*****
717 002252 005212 TST30: INC (R2) ;UPDATE TEST NUMBER
718 002254 022712 000030 CMP #30,(R2) ;SEQUENCE ERROR?
719 002260 001014 BNE TST31-10 ;BR TO ERROR HALT ON SEQ ERROR
720 002262 012705 177776 MOV #-2,R5 ;SET ALL ONES IN R5 EXCEPT FOR BIT 0
721 002266 012700 177757 MOV #-21,R0 ;SET BIT COUNTER
722 002272 000261 SEC ;SET C-BIT
723 002274 005200 REG5A: INC R0 ;INCREMENT BIT COUNTER
724 002276 001405 BEQ R5ERR ;BR TO ERROR HALT IF COUNTER=0
725 002300 006105 ROL R5 ;ROTATE 1 POSITION
726 002302 103774 BCS REG5A ;CONTINUE UNTIL C-BIT IS CLEAR
727 002304 022705 177777 CMP #-1,R5 ;CHECK DATA
728 002310 001404 BEQ TST31
```

```
729 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
730 ; CONDITIONAL BRANCH INST. AND <====
731 ; REPLACE THE MOVE INSTRUCTION <====
```



786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837

\*\*\*\*\*  
: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.

\*\*\*\*\*  
:TEST 33 TEST IF PSW WILL HOLD ZEROES

\*\*\*\*\*

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 @#PS ;SUCCESSFUL  
BEQ TST34

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 770 <=====  
: \*\*\*\*\* 37 \*\*\*\*\*

MOV #37,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 37 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;PSW NOT 0  
; OR SEQUENCE ERROR

\*\*\*\*\*  
:TEST 34 TEST IF PSW WILL HOLD ONES AND ZEROES

\*\*\*\*\*

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 TST35

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 771 <=====  
: \*\*\*\*\* 40 \*\*\*\*\*

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



838  
839  
840  
841  
842 002534 005212  
843 002536 022712 000035  
844 002542 001007  
845 002544 012737 000105 177776  
846 002552 023727 177776 000105  
847 002560 001404  
848  
849  
850  
851  
852 002562 012742 000041  
853 002566 005242  
854 002570 000000  
855  
856  
857  
858  
859  
860 002572 005212  
861 002574 022712 000036  
862 002600 001007  
863 002602 012737 000357 177776  
864 002610 023727 177776 000357  
865 002616 001404  
866  
867  
868  
869  
870 002620 012742 000042  
871 002624 005242  
872 002626 000000  
873

```
*****  
:TEST 35 TEST IF PSW (EXCEPT T-BIT) WILL HOLD ZEROES AND ONES  
*****  
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 TST36  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 771 <=====  
MOV #41,-(R2) ;MOVE TO MAILBOX # ***** 41 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;PSW NOT 105  
: OR SEQUENCE ERROR
```

```
*****  
:TEST 36 TEST IF PSW (EXCEPT T-BIT) WILL HOLD ALL ONES  
*****  
TST36: INC (R2) ;UPDATE TEST NUMBER  
CMP #36,(R2) ;SEQUENCE ERROR?  
BNE TST37-10 ;BR TO ERROR HALT ON SEQ ERROR  
MOV #357,@#PS ;MOVE ONES TO PSW  
CMP @#PS,#357 ;SUCCESSFUL  
BEQ TST37  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 771 <=====  
MOV #42,-(R2) ;MOVE TO MAILBOX # ***** 42 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;PSW NOT 357  
: OR SEQUENCE ERROR
```

874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921

.SBTTL CONDITION CODE TEST

\*\*\*\*\*

THIS TEST CHECKS THE CONDITIONAL BRANCHES INVOLVING THE Z-BIT.  
THE Z-BIT IS SET WITH ALL OTHER CC BITS ZERO AND BOTH CONDITIONS  
BEQ AND BNE ARE TESTED FOR PROPER EXECUTION. THEN THE Z-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.

\*\*\*\*\*

TEST 37 TEST BRANCHES AROUND Z-BIT

\*\*\*\*\*

```
TST37:  INC      (R2)           ;UPDATE TEST NUMBER
        CMP      #37,(R2)     ;SEQUENCE ERROR?
        BNE      TST40-10    ;BR TO ERROR HALT ON SEQ ERROR
        ;FIRST WITH Z-BIT ON
        CCC
        SEZ
        BNE      BRZ1        ;CHECK OPPOSITE CONDITION
        BEQ      BRZ2
        ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
        ; CONDITIONAL BRANCH INST. AND <====
        ; REPLACE THE MOVE INSTRUCTION <====
        ; WHICH FOLLOWS W/ 774 <====

BRZ1:   MOV      #43,-(R2)    ;MOVE TO MAILBOX # ***** 43 *****
        INC      -(R2)
        HALT
        ;CHECK WITH Z-BIT OFF
        SCC
        BRZ2:  SCC           ;CC=1011: ALL BUT Z-BIT
        CLZ
        BEQ      BRZ3
        BNE      TST40
        ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
        ; CONDITIONAL BRANCH INST. AND <====
        ; REPLACE THE MOVE INSTRUCTION <====
        ; WHICH FOLLOWS W/ 764 <====

BRZ3:   MOV      #44,-(R2)    ;MOVE TO MAILBOX # ***** 44 *****
        INC      -(R2)
        HALT
        ;IMPROPER BR W/ Z=0
        ; OR SEQUENCE ERROR
```

```
002630 005212
002632 022712 000037
002636 001014
002640 000257
002642 000264
002644 001001
002646 001404
002650
002650 012742 000043
002654 005242
002656 000000
002660 000277
002662 000244
002664 001401
002666 001004
002670
002670 012742 000044
002674 005242
002676 000000
```

922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968

002700 005212  
002702 022712 000040  
002706 001014  
002710 000257  
002712 000270  
002714 100001  
002716 100404  
002720  
002720 012742 000045  
002724 005242  
002726 000000  
002730 000277  
002732 000250  
002734 100401  
002736 100004  
002740  
002740 012742 000046  
002744 005242  
002746 000000

```
*****
:
:   THIS TEST CHECKS THE CONDITIONAL BRANCHES INVOLVING THE N-BIT.
: THE N-BIT IS SET WITH ALL OTHER CC BITS ZERO AND BOTH CONDITIONS
: BMI AND BPL ARE TESTED FOR PROPER EXECUTION. THEN THE N-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.
:
: *****
: TEST 40          TEST BRANCHES AROUND N-BIT
: *****
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
        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/ 774 <====
BRN1:   MOV      #45,-(R2)   ;MOVE TO MAILBOX # ***** 45 *****
        INC      -(R2)
        HALT
        ;CHECK WITH N-BIT OFF
        ;SET MSGTYP TO FATAL ERROR
        ;IMPROPER BR W/ N=1
BRN2:   SCC
        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/ 764 <====
BRN3:   MOV      #46,-(R2)   ;MOVE TO MAILBOX # ***** 46 *****
        INC      -(R2)
        HALT
        ;SET MSGTYP TO FATAL ERROR
        ;IMPROPER BR W/ N=0
        ; OR SEQUENCE ERROR
```

969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986 002750 005212  
987 002752 022712 000041  
988 002756 001014  
989  
990 002760 000257  
991 002762 000262  
992 002764 102001  
993 002766 102404  
994  
995  
996  
997  
998 002770  
999 002770 012742 000047  
1000 002774 005242  
1001 002776 000000  
1002  
1003 003000 000277  
1004 003002 000242  
1005 003004 102401  
1006 003006 102004  
1007  
1008  
1009  
1010  
1011 003010  
1012 003010 012742 000050  
1013 003014 005242  
1014 003016 000000  
1015

```
*****  
: THIS TEST CHECKS THE CONDITIONAL BRANCHES INVOLVING 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.  
*****  
: TEST 41 TEST BRANCHES AROUND V-BIT  
*****  
TST41: INC (R2) ;UPDATE TEST NUMBER  
CMP #41,(R2) ;SEQUENCE ERROR?  
BNE TST42-10 ;BR TO ERROR HALT ON SEQ ERROR  
;FIRST WITH V-BIT ON  
CCC ;CC=0010: JUST V-BIT  
SEV  
BVC BRV1 ;CHECK OPPOSITE CONDITION  
BVS BRV2  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 774 <====  
BRV1: MOV #47,-(R2) ;MOVE TO MAILBOX # ***** 47 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;IMPROPER BR W/ V=1  
;CHECK WITH V-BIT OFF  
BRV2: SCC ;CC=1101: ALL BVT V-BIT  
CLV  
BVS BRV3 ;CHECK OPPOSITE CONDITION  
BVC TST42  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 764 <====  
BRV3: MOV #50,-(R2) ;MOVE TO MAILBOX # ***** 50 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;IMPROPER BR W/ V=0  
: OR SEQUENCE ERROR
```

1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062

003020 005212  
003022 022712 000042  
003026 001014  
003030 000257  
003032 000261  
003034 103001  
003036 103404  
003040  
003040 012742 000051  
003044 005242  
003046 000000  
003050 000277  
003052 000241  
003054 103401  
003056 100404  
003060  
003060 012742 000052  
003064 005242  
003066 000000

```
*****
:
:   THIS TEST CHECKS THE CONDITIONAL BRANCHES INVOLVING THE C-BIT.
: THE C-BIT IS SET WITH ALL OTHER CC BITS ZERO AND BOTH CONDITIONS
: BCS AND BCC ARE TESTED FOR PROPER EXECUTION. THEN THE C-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.
:
: *****
: TEST 42      TEST BRANCHES AROUND C-BIT
: *****
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/ 774 <====
BRC1:  MOV      #51,-(R2)      ;MOVE TO MAILBOX # ***** 51 *****
        INC      -(R2)
        HALT
        ;CHECK WITH C-BIT OFF
BRC2:  SCC
        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/ 764 <====
BRC3:  MOV      #52,-(R2)      ;MOVE TO MAILBOX # ***** 52 *****
        INC      -(R2)
        HALT
        ;IMPROPER BR W/ C=0
        ; OR SEQUENCE ERROR
```

1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118

```
*****  
: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.  
:*****
```

```
*****  
:  
: 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.  
:*****
```

```
TEST 43 TEST MODE 0 USING SOP INST.  
*****
```

```
TST43: INC (R2) ;UPDATE TEST NUMBER  
CMP #43,(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/ 776 <====  
MOV #53,-(R2) ;MOVE TO MAILBOX # ***** 53 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CLR DID NOT SET Z-BIT  
SOP0A: INC R0 ;TRY THE INCREMENT INST.  
COM R0 ;TRY COMPLEMENT  
INC R0  
BMI SOP0B  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
```

```
1119          :          CONDITIONAL BRANCH INST. AND <====
1120          :          REPLACE THE MOVE INSTRUCTION <====
1121          :          WHICH FOLLOWS W/ 766 <====
1122 003124 012742 000054      MOV #54,-(R2)  ;MOVE TO MAILBOX # ***** 54 *****
1123 003130 005242          INC -(R2)      ;SET MSGTYP TO FATAL ERROR
1124 003132 000000          HALT          ;NEGATE DID NOT SET N-BIT
1125 003134 005100      SOPOB: COM R0      ;TRY COMPLEMENT INST.
1126 003136 001404          BEQ TST44
1127          :          : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
1128          :          CONDITIONAL BRANCH INST. AND <====
1129          :          REPLACE THE MOVE INSTRUCTION <====
1130          :          WHICH FOLLOWS W/ 760 <====
1131 003140 012742 000055      MOV #55,-(R2)  ;MOVE TO MAILBOX # ***** 55 *****
1132 003144 005242          INC -(R2)      ;SET MSGTYP TO FATAL ERROR
1133 003146 000000          HALT          ;CUMMULATIVE RESULT OF CLR,INC,NEG AND COM INSTS. FAILED
1134          :          : OR SEQUENCE ERROR
1135
1136
1137          :*****
1138          :
1139          :          THIS TEST INTRODUCES THE REMAINING SOP INSTRUCTIONS AND TESTS
1140          :          THEM IN MODE 0. THE PURPOSE IS TO PROVIDE A BASELINE OF
1141          :          INSTRUCTIONS FOR USE IN THE SUBSEQUENT TESTS. SINCE THE MICROCODE FOR
1142          :          THESE INSTRUCTIONS IS IDENTICAL TO THAT ALREADY TESTED, ANY TROUBLE
1143          :          SHOOTING EFFORTS SHOULD BE AIMED AT THE ACTUAL IR DECODE AND ALU
1144          :          FUNCTIONING.
1145          :
1146          :*****
1147          :TEST 44          TEST REMAINDER OF SOP INSTS IN MODE 0
1148          :*****
1149 003150 005212          TST44: INC (R2)          ;UPDATE TEST NUMBER
1150 003152 022712 000044      CMP #44,(R2)      ;SEQUENCE ERROR?
1151 003156 001021          BNE TST45-10    ;BR TO ERROR HALT ON SEQ ERROR
1152 003160 005000          CLR R0          ;INITIALIZE
1153 003162 005300          DEC R0          ;TRY DECREMENT INST.
1154 003164 100404          BMI SOPOC
1155          :          : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
1156          :          CONDITIONAL BRANCH INST. AND <====
1157          :          REPLACE THE MOVE INSTRUCTION <====
1158          :          WHICH FOLLOWS W/ 775 <====
1159 003166 012742 000056      MOV #56,-(R2)  ;MOVE TO MAILBOX # ***** 56 *****
1160 003172 005242          INC -(R2)      ;SET MSGTYP TO FATAL ERROR
1161 003174 000000          HALT          ;N-BIT NOT SET ON DEC
1162 003176 000261      SOPOC: SEC          ;INITIALIZE CARRY
1163 003200 005500          ADC R0          ;TRY ADD CARRY INST
1164 003202 001007          BNE SOPOD
1165 003204 000261          SEC          ;INITIALIZE CARRY
1166 003206 005600          SBC R0          ;TRY SUBTRACT-CARRY INST
1167 003210 100004          BPL SOFOD
1168 003212 005100          COM R0
1169 003214 005200          INC R0
1170 003216 005300          DEC R0
1171 003220 001404          BEQ TST45
1172          :          : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
1173          :          CONDITIONAL BRANCH INST. AND <====
1174          :          REPLACE THE MOVE INSTRUCTION <====
```

1175  
1176 003222  
1177 003222 012742 000057  
1178 003226 005242  
1179 003230 000000  
1180

SOP0D: MOV #57, -(R2)  
INC -(R2)  
HALT

; WHICH FOLLOWS W/ 757 <=====  
; MOVE TO MAILBOX # \*\*\*\*\* 57 \*\*\*\*\*  
; SET MSGTYP TO FATAL ERROR  
; CUMMULATIVE RESULT OF ADC, SBC, COM, INC AND DEC INSTS. F  
; OR SEQUENCE ERROR



1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191 003232 005212  
1192 003234 022712 000045  
1193 003240 001012  
1194 003242 105000  
1195 003244 001404  
1196  
1197  
1198  
1199  
1200 003246 012742 000060  
1201 003252 005242  
1202 003254 000000  
1203 003256 105100  
1204 003260 100002  
1205 003262 105200  
1206 003264 001404  
1207  
1208  
1209  
1210  
1211 003266  
1212 003266 012742 000061  
1213 003272 005242  
1214 003274 000000  
1215

```
*****
:
:   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.
:
:*****
:TEST 45          TEST MODE 0 EVEN BYTE USING SOP INST
:*****
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/ 776 <====
:
:             MOV      #60,-(R2) :MOVE TO MAILBOX # ***** 60 *****
:             INC      -(R2)     :SET MSGTYP TO FATAL ERROR
:             HALT
SOPBOA: COMB    R0           :CLRB DID NOT SET Z-BIT
        BPL     SOPBOB        :TRY SETTING EVEN BYTE OF REGISTER
        INCB    R0           :TRY INCREMENTING EVEN BYTE OF REGISTER>>
        BEQ     TST46
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:             CONDITIONAL BRANCH INST. AND <====
:             REPLACE THE MOVE INSTRUCTION <====
:             WHICH FOLLOWS W/ 766 <====
:
:             MOV      #61,-(R2) :MOVE TO MAILBOX # ***** 61 *****
:             INC      -(R2)     :SET MSGTYP TO FATAL ERROR
:             HALT
SOPBOB:          :TEST CUMMULATIVE RESULT OF ABOVE BYTE INST.
:             OR SEQUENCE ERROR
```

1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254

003276 005212  
003300 022712 000046  
003304 001014  
003306 005000  
003310 005010  
003312 001404  
  
003314 012742 000062  
003320 005242  
003322 000000  
003324 005310  
003326 100003  
003330 000261  
003332 005510  
003334 001404  
  
003336 012742 000063  
003342 005242  
003344 000000

```
*****
: THIS TEST USES THE CLR INSTRUCTION TO INTRODUCE AND TEST
: SINGLE OPERAND MODE 1 INSTRUCTIONS. AGAIN, THE CLR INSTRUCTION
: IS USED TO INTRODUCE THE MICROCODE AND TO TEST THAT THE PROPER
: CONDITION CODES ARE SET. OTHER SOP INSTRUCTIONS ARE USED TO MANIPULATE
: COMMON DATA TO VERIFY THAT THE CORRECT DATA IS PRODUCED.
*****
: TEST 46 TEST MODE 1 USING SOP INST.
*****
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/ 775 <====
: MOV #62,-(R2) ;MOVE TO MAILBOX # ***** 62 *****
: INC -(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 ;INITIALIZE CARRY
: ADC (R0) ;TRY ADD-CARRY W/MODE 1
: BEQ TST47
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 764 <====
SOP1B: MOV #63,-(R2) ;MOVE TO MAILBOX # ***** 63 *****
: INC -(R2) ;SET MSGTYP TO FATAL ERROR
: HALT ;TEST CUMMULATIVE RESULT OF ABOVE INST
: ; OR SEQUENCE ERROR
```

1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297

```
*****  
: 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.  
*****
```

```
TEST 47 TEST MODE 1 EVEN BYTE USING SOP INST
```

```
*****  
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/ 773 <====  
MOV #64,-(R2) ;MOVE TO MAILBOX # ***** 64 *****  
INC -(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) ;INCREMENT ODD BYTE=0  
BEQ TST50  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 760 <====  
SOPB1B: MOV #65,-(R2) ;MOVE TO MAILBOX # ***** 65 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CHECK CUMMULATIVE RESULT OF ABOVE INST  
: OR SEQUENCE ERROR
```

```
003346 005212  
003350 022712 000047  
003354 001020  
003356 005000  
003360 005010  
003362 005110  
003364 105010  
003366 001404  
  
003370 012742 000064  
003374 005242  
003376 000000  
003400 005210  
003402 100005  
003404 105110  
003406 105210  
003410 100002  
003412 105210  
003414 001404  
  
003416  
003416 012742 000065  
003422 005242  
003424 000000
```

1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311 003426 005212  
1312 003430 022712 000050  
1313 003434 001022  
1314 003436 005000  
1315 003440 005010  
1316 003442 005110  
1317 003444 005200  
1318 003446 105010  
1319 003450 001404  
1320  
1321  
1322  
1323  
1324 003452 012742 000066  
1325 003456 005242  
1326 003460 000000  
1327 003462 005300  
1328 003464 005210  
1329 003466 005200  
1330 003470 105110  
1331 003472 105210  
1332 003474 100002  
1333 003476 105210  
1334 003500 001404  
1335  
1336  
1337  
1338  
1339 003502  
1340 003502 012742 000067  
1341 003506 005242  
1342 003510 000000  
1343

```
*****  
: 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.  
*****  
: TEST 50 TEST MODE 1 ODD BYTE USING SOP INST  
*****  
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/ 772 <====  
MOV #66,-(R2) ;MOVE TO MAILBOX # ***** 66 *****  
INC -(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) ;TRY TO INCREMENT BYTE 1  
BEQ TST51  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 756 <====  
SOPB1D: MOV #67,-(R2) ;MOVE TO MAILBOX # ***** 67 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;TEST CUMMULATIVE RESULT OF ABOVE INST.  
: OR SEQUENCE ERROR
```

1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391

003512 005212  
003514 022712 000051  
003520 001023  
003522 005000  
003524 105100  
003526 005200  
003530 005010  
003532 005110  
003534 005020  
003536 001404  
  
003540 012742 000070  
003544 005242  
003546 000000  
003550 005300  
003552 005300  
003554 005120  
003556 100004  
003560 005300  
003562 005300  
003564 005220  
003566 001404  
  
003570  
003570 012742 000071  
003574 005242  
003576 000000

```
*****  
: 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.  
*****  
: TEST 51 TEST MODE 2 USING SOP INST.  
*****  
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/ 771 <====  
MOV #70,-(R2) ;MOVE TO MAILBOX # ***** 70 *****  
INC -(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/ 755 <====  
SOP2B: MOV #71,-(R2) ;MOVE TO MAILBOX # ***** 71 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CHECK CUMMULATIVE RESULT OF ABOVE INST  
: OR SEQUENCE ERROR
```

1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439

003600 005212  
003602 022712 000052  
003606 001023  
003610 005000  
003612 105100  
003614 005200  
003616 005010  
003620 005110  
003622 105020  
003624 001404  
  
003626 012742 000072  
003632 005242  
003634 000000  
003636 005300  
003640 005210  
003642 105110  
003644 105220  
003646 100003  
003650 005300  
003652 105220  
003654 001404  
  
003656 012742 000073  
003662 005242  
003664 000000

```
*****  
: 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.  
*****  
: TEST 52 TEST MODE 2 EVEN BYTE USING SOP INST.  
*****  
TST52: INC (R2) ;UPDATE TEST NUMBER  
CMP #52,(R2) ;SEQUENCE ERROR?  
BNE TST53-10 ;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  
CLRB (R0)+ ;TRY TO CLEAT 400 W/MODE 2  
BEQ SOPB2A  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 771 <====  
MOV #72,-(R2) ;MOVE TO MAILBOX # ***** 72 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CLR DID NOT SET Z-BIT  
SOPB2A: DEC R0 ;RESULT R0=400  
INC (R0) ;INC 400 TO TEST WORD  
COMB (R0)  
INCB (R0)+ ;TRY TO INC EVEN BYTE  
BPL SOPB2B  
DEC R0 ;RESET R0=400  
INCB (R0)+ ;TRY INCREMENT OF EVEN BYTE  
BEQ TST53  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 755 <====  
SOPB2B: MOV #73,-(R2) ;MOVE TO MAILBOX # ***** 73 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;TEST CUMMULATIVE RESULT OF ABOVE INST.  
: OR SEQUENCE ERROR
```

1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449 003666 005212  
1450 003670 022712 000053  
1451 003674 001026  
1452 003676 005000  
1453 003700 105100  
1454 003702 005200  
1455 003704 005010  
1456 003706 005110  
1457 003710 005200  
1458 003712 105020  
1459 003714 001404  
1460  
1461  
1462  
1463  
1464 003716 012742 000074  
1465 003722 005242  
1466 003724 000000  
1467 003726 005300  
1468 003730 005300  
1469 003732 005220  
1470 003734 005300  
1471 003736 105110  
1472 003740 105220  
1473 003742 100003  
1474 003744 005300  
1475 003746 105220  
1476 003750 001404  
1477  
1478  
1479  
1480  
1481 003752  
1482 003752 012742 000075  
1483 003756 005242  
1484 003760 000000  
1485  
1486

```
*****  
: THIS TEST FOLLOWS THE SAME PROCEDURE DESCRIBED IN THE PREVIOUS  
: TEST. HERE, THE BYTE INSTRUCTION IS USED TO ADDRESS AN ODD BYTE.  
*****  
: TEST 53 TEST MODE 2 ODD BYTE USING SOP INST.  
*****  
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/ 770 <====  
: MOVE TO MAILBOX # ***** 74 *****  
: SET MSGTYP TO FATAL ERROR  
: CLRB DID NOT SET Z-BIT  
: R0=WORD ADDR.  
  
SOPB2C: DEC R0  
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/ 752 <====  
: MOVE TO MAILBOX # ***** 75 *****  
: SET MSGTYP TO FATAL ERROR  
: TEST CUMMULATIVE RESULT OF ABOVE INST.  
: OR SEQUENCE ERROR
```

1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542

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

: TEST 54 - TEST MODE 0 USING NEGATE INSTRUCTION  
\*\*\*\*\*

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=-1  
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/ 771 <====

NEG00: MOV #76,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 76 \*\*\*\*\*  
INC -(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/ 763 <====

NEG02: MOV #77,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 77 \*\*\*\*\*  
INC -(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/ 751 <====

NEG03: MOV #100,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 100 \*\*\*\*\*  
INC -(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 <====



```
1543
1544 004064 012742 000101      MOV    #101,-(R2)      ;          WHICH FOLLOWS W/ 743      <=====
1545 004070 005242              INC    -(R2)          ;MOVE TO MAILBOX # ***** 101 *****
1546 004072 000000              HALT                   ;SET MSGTYP TO FATAL ERROR
1547                                     ;DATA RESULT OF NEGB INCORRECT
1548                                     ;OR SEQUENCE ERROR
1549
1550      ;*****
1551      ;TEST 55      TEST MODE 1 USING NEGATE INST.
1552      ;*****
1553      TST55:      INC    (R2)          ;UPDATE TEST NUMBER
1554      004074 005212 000055      CMP    #55,(R2)       ;SEQUENCE ERROR?
1555      004076 022712              BNE   TST56-10        ;BR TO ERROR HALT ON SEQ ERROR
1556      004102 001040              CLR   R0              ;POINT TO LOC. 0
1557      004104 005000              CLR   (R0)           ;CLEAR LOC. 0
1558      004106 005010              INC   (R0)           ;LOC. 0=1
1559      004110 005210              NEG   (R0)           ;TRY NEG. LOC. 0=-1
1560      004112 005410              BPL   NEG10          ;CC=1001
1561      004114 100003              BEQ   NEG10
1562      004116 001402              BVS   NEG10
1563      004120 102401              BCS   NEG10
1564      004122 103404              BCS   NEG11
1565
1566      NEG10:
1567      004124 012742 000102      MOV    #102,-(R2)     ;          TO SCOPE: CLEAR THE RIGHT BYTE OF THIS      <=====
1568      004130 005242              INC    -(R2)         ;          CONDITIONAL BRANCH INST. AND      <=====
1569      004132 000000              HALT                   ;          REPLACE THE MOVE INSTRUCTION      <=====
1570                                     ;          WHICH FOLLOWS W/ 770      <=====
1571      004134 005237 000000      NEG11: INC    @#0      ;TEST DATA RESULT
1572      004140 001404              BEQ   NEG12
1573
1574      NEG12:
1575      004142 012742 000103      MOV    #103,-(R2)     ;          TO SCOPE: CLEAR THE RIGHT BYTE OF THIS      <=====
1576      004146 005242              INC    -(R2)         ;          CONDITIONAL BRANCH INST. AND      <=====
1577      004150 000000              HALT                   ;          REPLACE THE MOVE INSTRUCTION      <=====
1578      004152 105110              COMB  (R0)           ;          WHICH FOLLOWS W/ 761      <=====
1579      004154 105410              NEGB  (R0)           ;MOVE TO MAILBOX # ***** 103 *****
1580      004156 100403              BMI   NEG13          ;SET MSGTYP TO FATAL ERROR
1581      004160 001402              BEQ   NEG13          ;DATA RESULT OF NEGATE INCORRECT
1582      004162 102401              BVS   NEG13          ;LOC. 0=377
1583      004164 103404              BCS   NEG14          ;TRY NEGB LOC. 0=1
1584                                     ;CC=0001?
1585
1586      NEG13:
1587      004166 012742 000104      MOV    #104,-(R2)     ;          TO SCOPE: CLEAR THE RIGHT BYTE OF THIS      <=====
1588      004172 005242              INC    -(R2)         ;          CONDITIONAL BRANCH INST. AND      <=====
1589      004174 000000              HALT                   ;          REPLACE THE MOVE INSTRUCTION      <=====
1590      004176 005337 000000      NEG14: DEC    @#0      ;          WHICH FOLLOWS W/ 747      <=====
1591      004202 001404              BEQ   TST56          ;MOVE TO MAILBOX # ***** 104 *****
1592                                     ;SET MSGTYP TO FATAL ERROR
1593                                     ;NEGB DID NOT SET CC'S CORRECTLY
1594                                     ;TEST DATA RESULT
1595
1596      NEG14:
1597      004202 001404              BEQ   TST56          ;          TO SCOPE: CLEAR THE RIGHT BYTE OF THIS      <=====
1598                                     ;          CONDITIONAL BRANCH INST. AND      <=====
1599                                     ;          REPLACE THE MOVE INSTRUCTION      <=====
```

CFKAACO 11/34 BSC INST TST  
CFKAAC.P11 18-OCT-78 11:01

MACY11 30A(1052) 18-OCT-78 11:06 PAGE 36  
T55 TEST MODE 1 USING NEGATE INST.

SEQ 0048

1599  
1600 004204 012742 000105  
1601 004210 005242  
1602 004212 000000  
1603

MOV #105, -(R2)  
INC -(R2)  
HALT

WHICH FOLLOWS W/ 740 <====  
: MOVE TO MAILBOX # \*\*\*\*\* 105 \*\*\*\*\*  
: SET MSGTYP TO FATAL ERROR  
: DATA RESULT OF NEGB INCORRECT  
: OR SEQUENCE ERROR

1604  
1605  
1606  
1607 004214 005212  
1608 004216 022712 000056  
1609 004222 001032  
1610 004224 005000  
1611 004226 005010  
1612 004230 005210  
1613 004232 005420  
1614 004234 100003  
1615 004236 001402  
1616 004240 102401  
1617 004242 103404  
1618  
1619  
1620  
1621  
1622 004244  
1623 004244 012742 000106  
1624 004250 005242  
1625 004252 000000  
1626 004254 105300  
1627 004256 105300  
1628 004260 105420  
1629 004262 105420  
1630 004264 105340  
1631 004266 005300  
1632 004270 001404  
1633  
1634  
1635  
1636  
1637 004272 012742 000107  
1638 004276 005242  
1639 004300 000000  
1640 004302 005337 000000  
1641 004306 001404  
1642  
1643  
1644  
1645  
1646 004310 012742 000110  
1647 004314 005242  
1648 004316 000000  
1649

```
*****  
:TEST 56 TEST MODE 2 USING NEGATE INSTRUCTION  
*****  
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/ 770 <====  
  
NEG20: MOV #106,-(R2) ;MOVE TO MAILBOX # ***** 106 *****  
INC -(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/ 755 <====  
  
MOV #107,-(R2) ;MOVE TO MAILBOX # ***** 107 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;REGISTER NOT INCREMENTED CORRECTLY  
NEG22: DEC @#0 ;LOC. 0=0  
BEQ TST57  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 746 <====  
  
MOV #110,-(R2) ;MOVE TO MAILBOX # ***** 110 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;NEG BYTE INSTRUCTIONS FAILED  
: OR SEQUENCE ERROR
```

1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698

004320 005212  
004322 022712 000057  
004326 001020  
004330 005000  
004332 105100  
004334 005200  
004336 005010  
004340 005030  
004342 001404  
  
004344 012742 000111  
004350 005242  
004352 000000  
004354 005300  
004356 005300  
004360 005130  
004362 100002  
004364 005230  
004366 001404  
  
004370  
004370 012742 000112  
004374 005242  
004376 000000

```
*****
: 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
: 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).
*****
: TEST 57 TEST MODE 3 USING SOP INST.
*****
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/ 772 <====
MOV #111,-(R2) ;MOVE TO MAILBOX # ***** 111 *****
INC -(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/ 760 <====
SOP3B: MOV #112,-(R2) ;MOVE TO MAILBOX # ***** 112 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;CUMMULATIVE RESULT OF ABOVE INST FAILED
; OR SEQUENCE ERROR
```

1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751

004400 005212  
004402 022712 000060  
004406 001026  
004410 005004  
004412 105104  
004414 005204  
004416 005000  
004420 005010  
004422 005110  
004424 105034  
004426 001404  
  
004430 012742 000113  
004434 005242  
004436 000000  
004440 005304  
004442 005304  
004444 005234  
004446 100006  
004450 105434  
004452 100004  
004454 005304  
004456 005304  
004460 105234  
004462 001404  
  
004464  
004464 012742 000114  
004470 005242  
004472 000000

```
*****  
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).  
*****  
TEST 60 TEST MODE 3 EVEN BYTE USING SOP INST.  
*****  
ST60: 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=402  
BEQ SOPB3A  
  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
; CONDITIONAL BRANCH INST. AND <=====  
; REPLACE THE MOVE INSTRUCTION <=====  
; WHICH FOLLOWS W/ 770 <=====  
MOV #113,-(R2) ;MOVE TO MAILBOX # ***** 113 *****  
INC -(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=402  
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/ 752 <=====  
SOPB3B: MOV #114,-(R2) ;MOVE TO MAILBOX # ***** 114 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CUMMULATIVE RESULT OF ABOVE INST FAILED  
; OR SEQUENCE ERROR
```

1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803

004474 005212  
004476 022712 000061  
004502 001024  
004504 005000  
004506 105100  
004510 005200  
004512 005030  
004514 005130  
004516 105030  
004520 001404  
  
004522 012742 000115  
004526 005242  
004530 000000  
004532 005300  
004534 005300  
004536 005300  
004540 005300  
004542 005230  
004544 105430  
004546 100002  
004550 105230  
004552 001404  
  
004554 012742 000116  
004560 005242  
004562 000000

```
*****  
: 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.  
*****  
: TEST 61 TEST MODE 3 ODD BYTE USING SOP INST.  
*****  
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/ 771 <=====  
MOV #115,-(R2) ;MOVE TO MAILBOX # ***** 115 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
SOPB3C: HALT ;CLRB DID NOT SET Z-BIT  
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/ 754 <=====  
SOPB3D: MOV #116,-(R2) ;MOVE TO MAILBOX # ***** 116 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CUMMULATIVE RESULT OF ABOVE INSTS FAILED  
: OR SEQUENCE ERROR
```

```
1804
1805
1806
1807 004564 005212
1808 004566 022712 000062
1809 004572 001054
1810 004574 005000
1811 004576 105100
1812 004600 005200
1813 004602 005010
1814 004604 005004
1815 004606 005014
1816 004610 005214
1817 004612 005430
1818 004614 100003
1819 004616 001402
1820 004620 102401
1821 004622 103404
1822
1823
1824
1825
1826 004624
1827 004624 012742 000117
1828 004630 005242
1829 004632 000000
1830 004634 005214
1831 004636 001404
1832
1833
1834
1835
1836 004640 012742 000120
1837 004644 005242
1838 004646 000000
1839 004650 105137 000001
1840 004654 005237 000000
1841 004660 105430
1842 004662 100404
1843
1844
1845
1846
1847 004664 012742 000121
1848 004670 005242
1849 004672 000000
1850 004674 105430
1851 004676 100004
1852
1853
1854
1855
1856 004700 012742 000122
1857 004704 005242
1858 004706 000000
1859 004710 105137 000001
```

```
*****
:TEST 62 TEST MODE 3 USING NEGATE INSTRUCTION
*****
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/ 764 <====

NEG30: MOV #117,-(R2) ;MOVE TO MAILBOX # ***** 117 *****
INC -(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/ 756 <====

MOV #120,-(R2) ;MOVE TO MAILBOX # ***** 120 *****
INC -(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/ 744 <====

MOV #121,-(R2) ;MOVE TO MAILBOX # ***** 121 *****
INC -(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/ 736 <====

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

CFKAACO 11/34 BSC INST TST  
CFKAAC.P11 18-OCT-78 11:01

MACY11 30A(1052) 18-OCT-78 11:06 PAGE 42  
T62 TEST MODE 3 USING NEGATE INSTRUCTION

SEQ 0054

1860 004714 105237 000001  
1861 004720 005214  
1862 004722 001404  
1863  
1864  
1865  
1866  
1867 004724 012742 000123  
1868 004730 005242  
1869 004732 000000  
1870

INCB @#1  
INC (R4)  
BEQ TST63  
  
MOV #123, -(R2)  
INC -(R2)  
HALT

:LOC. 0=177777  
:LOC. 0=0  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 724 <====  
:MOVE TO MAILBOX # \*\*\*\*\* 123 \*\*\*\*\*  
:SET MSGTYP TO FATAL ERROR  
:DATA RESULT OF NEGB'S INCORRECT  
: OR SEQUENCE ERROR



1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883 004734 005212  
1884 004736 022712 000063  
1885 004742 001021  
1886 004744 005000  
1887 004746 105100  
1888 004750 005200  
1889 004752 005040  
1890 004754 001404  
1891  
1892  
1893  
1894  
1895 004756 012742 000124  
1896 004762 005242  
1897 004764 000000  
1898 004766 005200  
1899 004770 005200  
1900 004772 005140  
1901 004774 100004  
1902 004776 005200  
1903 005000 005200  
1904 005002 005240  
1905 005004 001404  
1906  
1907  
1908  
1909  
1910 005006  
1911 005006 012742 000125  
1912 005012 005242  
1913 005014 000000  
1914

```
*****
:
: THIS TEST VERIFIES MODE 4 SINGLE OPERAND INSTRUCTIONS.
: R0 IS SET TO 400. A CLR INSTRUCTION IS EXECUTED IN MODE 4 TO CLEAR
: LOC. 376. R0 IS RESET TO 400 AND A COM INSTRUCTION USING MODE 4
: COMPLEMENTS LOC.376.
: TWO INC INSTRUCTIONS AND A MODE 4 INSTRUCTION ARE EXECUTED
: TO COMPLETE THE TEST.
:
:*****
: TEST 63 TEST MODE 4 USING SOP INSTS
:*****
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/ 773 <====
      MOV #124,-(R2) ;MOVE TO MAILBOX # ***** 124 *****
      INC -(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/ 757 <====
SOP4B: MOV #125,-(R2) ;MOVE TO MAILBOX # ***** 125 *****
      INC -(R2) ;SET MSGTYP TO FATAL ERROR
      HALT ;CHECK CUMMULATIVE RESULT OF ABOVE INST.
: OR SEQUENCE ERROR
```

1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934 005016 005212  
1935 005020 022712 000064  
1936 005024 001017  
1937 005026 005000  
1938 005030 005020  
1939 005032 105400  
1940 005034 005050  
1941 005036 001404  
1942  
1943  
1944  
1945  
1946 005040 012742 000126  
1947 005044 005242  
1948 005046 000000  
1949 005050 005200  
1950 005052 005200  
1951 005054 005150  
1952 005056 100002  
1953 005060 005250  
1954 005062 001404  
1955  
1956  
1957  
1958  
1959 005064  
1960 005064 012742 000127  
1961 005070 005242  
1962 005072 000000  
1963

\*\*\*\*\*

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).

\*\*\*\*\*

TEST 64 TEST MODE 5 USING SOP INSTS

\*\*\*\*\*

TST64: INC (R2) ;UPDATE TEST NUMBER  
CMP #64,(R2) ;SEQUENCE ERROR?  
BNE TST65-10 ;BR TO ERROR HALT ON SEQ ERROR  
CLR R0 ;SET R0=376  
CLR (R0)+  
NEGB R0  
CLR @-(R0) ;TRY TO CLEAR LOC 0 W/MODE 5  
BEQ SOP5A  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 773 <====  
MOV #126,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 126 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CLR DID NOT SET Z-BIT  
SOP5A: INC R0 ;RESET R0  
INC R0  
COM @-(R0) ;TRY TO COMPLEMENT LOC. 0 W/MODE 5  
BPL SOP5B  
INC @-(R0) ;TRY TO INCREMENT LOC. 0 W/MODE 5  
BEQ TST65  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 761 <====  
SOP5B: MOV #127,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 127 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;TEST CUMMULATIVE RESULT OF ABOVE INSTS  
; OR SEQUENCE ERROR

1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976 005074 005212  
1977 005076 022712 000065  
1978 005102 001020  
1979 005104 005000  
1980 005106 105100  
1981 005110 005200  
1982 005112 005060 177400  
1983 005116 001404  
1984  
1985  
1986  
1987  
1988 005120 012742 000130  
1989 005124 005242  
1990 005126 000000  
1991 005130 005160 177400  
1992 005134 100003  
1993 005136 005260 177400  
1994 005142 001404  
1995  
1996  
1997  
1998  
1999 005144  
2000 005144 012742 000131  
2001 005150 005242  
2002 005152 000000  
2003

```
*****
:
:   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.
:
:*****
:TEST 65          TEST MODE 6 USING SOP INSTS
:*****
TST65:  INC      (R2)          ;UPDATE TEST NUMBER
        CMP      #65,(R2)     ;SEQUENCE ERROR?
        BNE     TST66-10     ;BR TO ERROR HALT ON SEQ ERROR
        CLR     R0           ;SET R0=400
        COMB    R0
        INC     R0
        CLR     -400(R0)     ;TRY TO CLEAR LOCATION 0 W/MODE 6
        BEQ     SOP6A
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
:   CONDITIONAL BRANCH INST. AND <=====
:   REPLACE THE MOVE INSTRUCTION <=====
:   WHICH FOLLOWS W/ 772 <=====
        MOV     #130,-(R2)    ;MOVE TO MAILBOX # ***** 130 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT
SOP6A:  COM     -400(R0)     ;CLR DID NOT SET Z-BIT
        BPL     SOP6B
        INC     -400(R0)     ;TRY TO INCREMENT LOCATION 0 W/MODE 6
        BEQ     TST66
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
:   CONDITIONAL BRANCH INST. AND <=====
:   REPLACE THE MOVE INSTRUCTION <=====
:   WHICH FOLLOWS W/ 0 <=====
SOP6B:  MOV     #131,-(R2)    ;MOVE TO MAILBOX # ***** 131 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT                ;TEST CUMMULATIVE RESULT OF ABOVE INSTS
:                               ; OR SEQUENCE ERROR
```

2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017 005154 005212  
2018 005156 022712 000066  
2019 005162 001021  
2020 005164 005000  
2021 005166 105100  
2022 005170 005200  
2023 005172 005210  
2024 005174 005070 000002  
2025 005200 001404  
2026  
2027  
2028  
2029  
2030 005202 012742 000132  
2031 005206 005242  
2032 005210 000000  
2033 005212 005170 000002  
2034 005216 100003  
2035 005220 005270 000002  
2036 005224 001404  
2037  
2038  
2039  
2040  
2041 005226  
2042 005226 012742 000133  
2043 005232 005242  
2044 005234 000000  
2045  
2046

```
*****
:
: THIS TEST VERIFIES MODE 7 SINGLE OPERAND INSTRUCTIONS. IT USES
: THE POINTER TO LOC. 0 WHICH IS STORED AT LOC. 402.
: R0 IS SET TO 400 AND A MODE 7 CLR INSTRUCTION IS
: EXECUTED WITH A +2 OFFSET TO CLEAR LOC. 0.
: SEVERAL OTHER MODE 7 INSTRUCTIONS ARE THEN USED ON THE COMMON
: LOCATION TO VERIFY THE DATA RESULTS.
:
:*****
:TEST 66 TEST MODE 7 USING SOP INST.
:*****
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
      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
      ; CONDITIONAL BRANCH INST. AND <====
      ; REPLACE THE MOVE INSTRUCTION <====
      ; WHICH FOLLOWS W/ 771 <====
      MOV #132,-(R2) ;MOVE TO MAILBOX # ***** 132 *****
      INC -(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
      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
      ; CONDITIONAL BRANCH INST. AND <====
      ; REPLACE THE MOVE INSTRUCTION <====
      ; WHICH FOLLOWS W/ 757 <====
SOP7B: MOV #133,-(R2) ;MOVE TO MAILBOX # ***** 133 *****
      INC -(R2) ;SET MSGTYP TO FATAL ERROR
      HALT ;TEST CUMMULATIVE RESULT OF ABOVE INSTS.
      ; OR SEQUENCE ERROR
```

```
2047 :*****
2048 :TEST 67 TEST MODE 4 WITH NEGATE INSTRUCTION
2049 :*****
2050 005236 005212 TST67: INC (R2) ;UPDATE TEST NUMBER
2051 005240 022712 000067 CMP #67,(R2) ;SEQUENCE ERROR?
2052 005244 001024 BNE TST70-10 ;BR TO ERROR HALT ON SEQ ERROR
2053 005246 005000 CLR R0
2054 005250 005010 CLR (R0)
2055 005252 005120 COM (R0)+ ;LOC. 0=177777, R0=2
2056 005254 005440 NEG -(R0) ;TRY NEGATE, LOC. 0=1
2057 005256 100403 BMI NEG40 ;CC=0001?
2058 005260 001402 BEQ NEG40
2059 005262 102401 BVS NEG40
2060 005264 103404 BCS NEG41
2061 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
2062 : CONDITIONAL BRANCH INST. AND <=====
2063 : REPLACE THE MOVE INSTRUCTION <=====
2064 : WHICH FOLLOWS W/ 770 <=====
2065 005266 NEG40:
2066 005266 012742 000134 MOV #134,-(R2) ;MOVE TO MAILBOX # ***** 134 *****
2067 005272 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
2068 005274 000000 HALT ;NEG DID NOT SET CC'S CORRECTLY
2069 005276 005400 NEG41: NEG R0 ;TST R0 WITH A NEG.
2070 005300 001404 BEQ NEG42
2071 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
2072 : CONDITIONAL BRANCH INST. AND <=====
2073 : REPLACE THE MOVE INSTRUCTION <=====
2074 : WHICH FOLLOWS W/ 762 <=====
2075 005302 012742 000135 MOV #135,-(R2) ;MOVE TO MAILBOX # ***** 135 *****
2076 005306 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
2077 005310 000000 HALT ;R0 NOT DECREMENTED PROPERLY
2078 005312 005310 NEG42: DEC (R0) ;TEST DTA RESULT OF NEG
2079 005314 001404 BEQ TST70
2080 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
2081 : CONDITIONAL BRANCH INST. AND <=====
2082 : REPLACE THE MOVE INSTRUCTION <=====
2083 : WHICH FOLLOWS W/ 754 <=====
2084 005316 012742 000136 MOV #136,-(R2) ;MOVE TO MAILBOX # ***** 136 *****
2085 005322 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
2086 005324 000000 HALT ;DATA RESULT OF NEG INCORRECT
2087 : OR SEQUENCE ERROR
```

```
2088 :*****
2089 :TEST 70 TEST MODE 5 WITH NEGATE INSTRUCTION
2090 :*****
2091 005326 005212 TST70: INC (R2) ;UPDATE TEST NUMBER
2092 005330 022712 000070 CMP #70,(R2) ;SEQUENCE ERROR?
2093 005334 001031 BNE TST71-10 ;BR TO ERROR HALT ON SEQ ERROR
2094 005336 005000 CLR R0 ;R0=0
2095 005340 005010 CLR (R0) ;LOC. 0=0
2096 005342 105100 COMB R0 ;R0=377
2097 005344 005200 INC R0 ;R0=400
2098 005346 005010 CLR (R0) ;SET 400 = 0
2099 005350 005004 CLR R4 ;R4=0
2100 005352 005314 DEC (R4) ;LOC. 0=177777
2101 005354 005450 NEG @-(R0) ;TRY NEGATE: LOC. 0=1
2102 005356 100403 BMI NEG50 ;CC=0001?
2103 005360 001402 BEQ NEG50
2104 005362 102401 BVS NEG50
2105 005364 103404 BCS NEG51
2106 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
2107 : CONDITIONAL BRANCH INST. AND <====
2108 : REPLACE THE MOVE INSTRUCTION <====
2109 : WHICH FOLLOWS W/ 764 <====
2110 005366 NEG50:
2111 005366 012742 000137 MOV #137,-(R2) ;MOVE TO MAILBOX # ***** 137 *****
2112 005372 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
2113 005374 000000 HALT ;NEG DID NOT SET CC'S CORRECTLY
2114 005376 005314 NEG51: DEC (R4)
2115 005400 001404 BEQ NEG52
2116 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
2117 : CONDITIONAL BRANCH INST. AND <====
2118 : REPLACE THE MOVE INSTRUCTION <====
2119 : WHICH FOLLOWS W/ 756 <====
2120 005402 012742 000140 MOV #140,-(R2) ;MOVE TO MAILBOX # ***** 140 *****
2121 005406 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
2122 005410 000000 HALT ;DATA RESULT OF NEG INCORRECT
2123 005412 105100 NEG52: COMB R0
2124 005414 005300 DEC R0
2125 005416 001404 BEQ TST71
2126 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
2127 : CONDITIONAL BRANCH INST. AND <====
2128 : REPLACE THE MOVE INSTRUCTION <====
2129 : WHICH FOLLOWS W/ 747 <====
2130 005420 012742 000141 MOV #141,-(R2) ;MOVE TO MAILBOX # ***** 141 *****
2131 005424 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
2132 005426 000000 HALT ;REGISTER NOT DECREMENTED PROPERLY
2133 : OR SEQUENCE ERROR
```

```
2134 :*****  
2135 :TEST 71 TEST MODE 6 WITH NEGATE  
2136 :*****  
2137 005430 005212 TST71: INC (R2) ;UPDATE TEST NUMBER  
2138 005432 022712 000071 CMP #71,(R2) ;SEQUENCE ERROR?  
2139 005436 001922 BNE TST72-10 ;BR TO ERROR HALT ON SEQ ERROR  
2140 005440 005000 CLR R0 ;R0=0  
2141 005442 005004 CLR R4 ;R4=0  
2142 005444 105100 COMB R0 ;R0=377  
2143 005446 005014 CLR (R4) ;LOC. 0=0  
2144 005450 105024 CLRB (R4)+ ;LOC. 0=177777, R4=1  
2145 005452 105114 COMB (R4) ;LOC. 0=177400  
2146 005454 005460 177401 NEG -377(R0) ;LOC. 0=400  
2147 005460 100403 BMI NEG60 ;CC=0001  
2148 005462 001402 BEQ NEG60  
2149 005464 102401 BVS NEG60  
2150 005466 103404 BCS NEG61  
2151 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
2152 : CONDITIONAL BRANCH INST. AND <====  
2153 : REPLACE THE MOVE INSTRUCTION <====  
2154 : WHICH FOLLOWS W/ 764 <====  
2155 005470 NEG60:  
2156 005470 012742 000142 MOV #142,-(R2) ;MOVE TO MAILBOX # ***** 142 *****  
2157 005474 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
2158 005476 000000 HALT ;NEG DID NOT SET CC'S CORRECTLY  
2159 005500 NEG61: DECB (R4)  
2160 005502 001404 BEQ TST72  
2161 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
2162 : CONDITIONAL BRANCH INST. AND <====  
2163 : REPLACE THE MOVE INSTRUCTION <====  
2164 : WHICH FOLLOWS W/ 756 <====  
2165 005504 012742 000143 : MOV #143,-(R2) ;MOVE TO MAILBOX # ***** 143 *****  
2166 005510 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
2167 005512 000000 HALT ;DATA RESULT OF NEG INCORRECT  
2168 : OR SEQUENCE ERROR
```

2169  
2170  
2171  
2172 005514 005212  
2173 005516 022712 000072  
2174 005522 001024  
2175 005524 005000  
2176 005526 005010  
2177 005530 005110  
2178 005532 105100  
2179 005534 105470 000005  
2180 005540 100403  
2181 005542 001402  
2182 005544 102401  
2183 005546 103404  
2184  
2185  
2186  
2187  
2188 005550  
2189 005550 012742 000144  
2190 005554 005242  
2191 005556 000000  
2192 005560 105100  
2193 005562 105120  
2194 005564 105310  
2195 005566 005467 172206  
2196 005572 001404  
2197  
2198  
2199  
2200  
2201 005574 012742 000145  
2202 005600 005242  
2203 005602 000000  
2204

```
*****  
:TEST 72 TEST MODE 7 W/ NEGATE  
*****  
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/ 766 <====  
  
NEG70: MOV #144,-(R2) :MOVE TO MAILBOX # ***** 144 *****  
INC -(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/ 754 <====  
  
MOV #145,-(R2) :MOVE TO MAILBOX # ***** 145 *****  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :DATA RESULT OF NEG WAS INCORRECT  
: OR SEQUENCE ERROR
```



2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216  
2217  
2218  
2219  
2220  
2221  
2222  
2223  
2224  
2225  
2226  
2227  
2228  
2229  
2230  
2231  
2232  
2233  
2234  
2235  
2236  
2237  
2238  
2239  
2240  
2241  
2242  
2243  
2244

005604 005212  
005606 022712 000073  
005612 001017  
005614 005027  
005616 177777  
005620 001404  
  
005622 012742 000146  
005626 005242  
005630 000000  
005632 005237 005616  
005636 005467 177754  
005642 100003  
005644 005277 000012  
005650 001405  
  
005652  
005652 012742 000147  
005656 005242  
005660 000000  
005662 005616

```
*****
:
:   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.
:
: *****
: TEST 73      TEST SOP INSTRUCTIONS MODES 2,3,6,7 WITH REGISTER 7
: *****
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
        BEQ     SOPA         ;USE MODE 27
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:   CONDITIONAL BRANCH INST. AND <====
:   REPLACE THE MOVE INSTRUCTION <====
:   WHICH FOLLOWS W/ 775 <====
:
:   MOV      #146,-(R2)      ;MOVE TO MAILBOX # ***** 146 *****
:   INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
:   HALT
SOPA:   INC      @#SOPX      ;CLR DID NOT SET Z-BIT
        NEG     SOPX        ;INC SOPX W/MODE 37
        BPL     SOPB        ;NEGATE SOPX W/MODE 67
        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/ 761 <====
:
:   MOV      #147,-(R2)      ;MOVE TO MAILBOX # ***** 147 *****
:   INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
:   HALT
:   OR      SEQUENCE ERROR
SOPXAD: SOPX ;INDIRECT ADDRESS OF SOPX
```

2245  
2246  
2247  
2248  
2249  
2250  
2251  
2252  
2253  
2254  
2255  
2256  
2257  
2258  
2259  
2260  
2261  
2262  
2263  
2264  
2265  
2266  
2267  
2268  
2269  
2270  
2271  
2272  
2273  
2274  
2275  
2276

005664 005212  
005666 022712 000074  
005672 001010  
005674 005000  
005676 000277  
005700 000244  
005702 005700  
005704 102403  
005706 100402  
005710 103401  
005712 001404  
  
005714  
005714 012742 000150  
005720 005242  
005722 000000

```
*****
: 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.
*****
: TEST 74 TEST MODE 0 SOP NON-MODIFYING
*****
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/ 770 <====

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

2277  
2278  
2279  
2280  
2281  
2282  
2283  
2284  
2285  
2286  
2287  
2288  
2289  
2290  
2291  
2292  
2293  
2294  
2295  
2296  
2297  
2298  
2299  
2300  
2301  
2302  
2303  
2304  
2305  
2306  
2307  
2308

005724 005212  
005726 022712 000075  
005732 001010  
005734 005000  
005736 105100  
005740 000277  
005742 000250  
005744 105700  
005746 102402  
005750 101401  
005752 100404  
  
005754  
005754 012742 000151  
005760 005242  
005762 000000

\*\*\*\*\*  
: THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING BYTE INSTRUCTIONS WITH MODE C.  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 75 TEST MODE 0 EVEN BYTE W/ SOP NON-MODIFYING  
\*\*\*\*\*

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/ 770 <====  
  
SNMBOA: MOV #151,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 151 \*\*\*\*\*  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :CONDITION CODES NOT SET PROPERLY  
: OR SEQUENCE ERROR

2309  
2310  
2311  
2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325  
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341

005764 005212  
005766 022712 000076  
005772 001011  
005774 005000  
005776 005010  
006000 000277  
006002 000244  
006004 005710  
006006 102403  
006010 103402  
006012 100401  
006014 001404  
  
006016  
006016 012742 000152  
006022 005242  
006024 000000

```
*****
:
: THIS TEST VERIFIES SINGLE OPERAND INSTRUCTIONS WITH MODE 1.
: R0 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 R0 AND CONDITIONAL BRANCHES TEST
: THE RESULTS.
:
: *****
: TEST 76 TEST MODE 1 SOP NON-MODIFYING
: *****
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/ 767 <====

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

2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363  
2364  
2365  
2366  
2367  
2368  
2369  
2370  
2371  
2372  
2373  
2374  
2375  
2376  
2377  
2378  
2379  
2380  
2381  
2382  
2383  
2384  
2385  
2386  
2387  
2388  
2389  
2390

006026 005212  
006030 022712 000077  
006034 001026  
006036 005000  
006040 005010  
006042 105110  
006044 000277  
006046 000250  
006050 105710  
006052 102402  
006054 101401  
006056 100404  
  
006060  
006060 012742 000153  
006064 005242  
006066 000000  
006070 005000  
006072 005200  
006074 000277  
006076 000244  
006100 105710  
006102 102403  
006104 103402  
006106 100401  
006110 001404  
  
006112  
006112 012742 000154  
006116 005242  
006120 000000

```
*****  
: 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.  
*****  
: TEST 77 TEST MODE 1 BYTE INST. NON-MODIFYING  
*****  
TST77: INC (R2) ;UPDATE TEST NUMBER  
CMP #77,(R2) ;SEQUENCE ERROR?  
BNE TST100-10 ;BR TO ERROR HALT ON SEQ ERROR  
CLR R0 ;POINT TO LOC 0  
CLR (R0) ;CLEAR LOC 0  
COMB (R0) ;COMPLEMENT BYTE 0  
SCC ;SET CC=0111  
CLN  
TSTB (R0) ;TRY TST ON EVEN BYTE  
BVS SNMB1A  
BLOS SNMB1A  
BMI SNMB1B  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 767 <====  
  
SNMB1A: MOV #153,-(R2) ;MOVE TO MAILBOX # ***** 153 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT CORRECT  
  
SNMB1B: CLR R0  
INC R0  
SCC ;SET CC=1011  
CLZ  
TSTB (R0) ;TRY TO TST AN ODD BYTE  
BVS SNMB1C  
BCS SNMB1C  
BMI SNMB1C  
BEQ TST100  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 752 <====  
  
SNMB1C: MOV #154,-(R2) ;MOVE TO MAILBOX # ***** 154 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT CORRECT  
; OR SEQUENCE ERROR
```

2391  
2392  
2393  
2394  
2395  
2396  
2397  
2398  
2399  
2400  
2401  
2402 006122 005212  
2403 006124 022712 000100  
2404 006130 001020  
2405 006132 005000  
2406 006134 005010  
2407 006136 000277  
2408 006140 000244  
2409 006142 005720  
2410 006144 102403  
2411 006146 103402  
2412 006150 100401  
2413 006152 001404  
2414  
2415  
2416  
2417  
2418 006154  
2419 006154 012742 000155  
2420 006160 005242  
2421 006162 000000  
2422 006164 005300  
2423 006166 005300  
2424 006170 001404  
2425  
2426  
2427  
2428  
2429 006172 012742 000156  
2430 006176 005242  
2431 006200 000000  
2432

```
*****
: THIS TEST VERIFIES THE SINGLE-OPERAND NON-MODIFYING INSTRUCTIONS
: USING MODE 2. IT USES THE IDENTICAL PROCEDURE EMPLOYED IN THE
: MODE 1 TESTS. ADDITIONALLY, THE REGISTER IS CHECKED TO ASSURE THAT
: IT IS INCREMENTED PROPERLY.
*****
: TEST 100 TEST MODE 2 WITH SOP NON-MODIFYING
*****
TST100: INC (R2) ;UPDATE TEST NUMBER
        CMP #100,(R2) ;SEQUENCE ERROR?
        BNE TST101-10 ;BR TO ERROR HALT ON SEQ ERROR
        CLR R0 ;INITIALIZE R0=0
        CLR (R0) ;CLEAR LOC 0
        SCC ;SET CC=1011
        CLZ
        TST (R0)+ ;TRY TST W/ MODE 2
        BVS SNM2A ;CHECK CC=0100
        BCS SNM2A
        BMI SNM2A
        BEQ SNM2B

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

SNM2A: MOV #155,-(R2) ;MOVE TO MAILBOX # ***** 155 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;CC'S NOT CORRECT
SNM2B: DEC R0 ;RESET R0
        DEC R0
        BEQ TST101

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

        MOV #156,-(R2) ;MOVE TO MAILBOX # ***** 156 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;MODE 2 DID NOT INC REQ CORRECTLY
; OR SEQUENCE ERROR
```

2433  
2434  
2435  
2436  
2437  
2438  
2439  
2440  
2441  
2442  
2443  
2444  
2445  
2446  
2447  
2448  
2449  
2450  
2451  
2452  
2453  
2454  
2455  
2456  
2457  
2458  
2459  
2460  
2461  
2462  
2463  
2464  
2465  
2466  
2467  
2468  
2469  
2470  
2471  
2472  
2473  
2474  
2475  
2476  
2477  
2478  
2479  
2480  
2481  
2482  
2483  
2484  
2485  
2486  
2487  
2488

006202 005212  
006204 022712 000101  
006210 001042  
006212 005000  
006214 005010  
006216 105110  
006220 000277  
006222 000250  
006224 105720  
006226 102402  
006230 101401  
006232 100404  
  
006234 012742 000157  
006240 005242  
006242 000000  
006244 005300  
006246 001404  
  
006250 012742 000160  
006254 005242  
006256 000000  
006260 005200  
006262 000277  
006264 000244  
006266 105720  
006270 102403  
006272 103402  
006274 100401  
006276 001404  
  
006300 012742 000161  
006304 005242

```
*****  
: 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.  
*****  
: TEST 101 TEST MODE 2 - BYTE W/ SOP NON-MODIFYING  
*****  
TST101: INC (R2) ;UPDATE TEST NUMBER  
CMP #101,(R2) ;SEQUENCE ERROR?  
BNE TST102-10 ;BR TO ERROR HALT ON SEQ ERROR  
CLR R0 ;CLEAR R0  
CLR (R0) ;CLEAR LOC 0  
COMB (R0) ;SET LOC 0=377  
SCC ;SET CC=0111  
CLN  
TSTB (R0)+ ;TRY TST OF EVEN BYTE  
BVS SNMB2A  
BLOS SNMB2A  
BMI SNMB2B  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 767 <====  
  
SNMB2A: MOV #157,-(R2) ;MOVE TO MAILBOX # ***** 157 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT SET CORRECTLY  
SNMB2B: DEC R0 ;DECREMENT R0  
BEQ SNMB2C  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 761 <====  
  
SNMB2C: MOV #160,-(R2) ;MOVE TO MAILBOX # ***** 160 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;MODE 2 DID NOT INC REG CORRECTLY  
SNMB2C: INC R0 ;POINT TO ODD BYTE  
SCC ;SET CC=0111  
CLZ  
TSTB (R0)+ ;TRY TST OF ODD BYTE  
BVS SNMB2D ;CHECK CC'S=0100  
BCS SNMB2D  
BMI SNMB2D  
BEQ SNMB2E  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 745 <====  
  
SNMB2D: MOV #161,-(R2) ;MOVE TO MAILBOX # ***** 161 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR
```

2489 006306 000000  
2490 006310 005300  
2491 006312 005300  
2492 006314 001404  
2493  
2494  
2495  
2496  
2497 006316 012742 000162  
2498 006322 005242  
2499 006324 000000  
2500

SNMB2E: HALT  
DEC R0  
DEC R0  
BEQ TST102  
  
MOV #162, -(R2)  
INC -(R2)  
HALT

;CC'S NOT CORRECT  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 736 <====  
: MOVE TO MAILBOX # \*\*\*\*\* 162 \*\*\*\*\*  
: SET MSGTYP TO FATAL ERROR  
: R0 DID NOT INCREMENT PROPERLY  
: OR SEQUENCE ERROR



2501  
2502  
2503  
2504  
2505  
2506  
2507  
2508  
2509  
2510  
2511  
2512 006326 005212  
2513 006330 022712 000102  
2514 006334 001022  
2515 006336 005000  
2516 006340 005010  
2517 006342 105100  
2518 006344 005300  
2519 006346 000277  
2520 006350 000244  
2521 006352 005730  
2522 006354 102403  
2523 006356 103402  
2524 006360 100401  
2525 006362 001404  
2526  
2527  
2528  
2529  
2530 006364  
2531 006364 012742 000163  
2532 006370 005242  
2533 006372 000000  
2534 006374 005300  
2535 006376 105100  
2536 006400 001404  
2537  
2538  
2539  
2540  
2541 006402 012742 000164  
2542 006406 005242  
2543 006410 000000  
2544

```
*****
: 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.
*****
: TEST 102 TEST MODE 3 W/ SOP NON-MODIFYING INSTS
*****
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/ 765 <====

SNM3A: MOV #163,-(R2) ;MOVE TO MAILBOX # ***** 163 *****
INC -(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/ 756 <====

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

2545  
2546  
2547  
2548  
2549  
2550  
2551  
2552  
2553  
2554  
2555  
2556  
2557  
2558  
2559  
2560  
2561  
2562  
2563  
2564  
2565  
2566  
2567  
2568  
2569  
2570  
2571  
2572  
2573  
2574  
2575  
2576  
2577  
2578  
2579  
2580  
2581  
2582  
2583  
2584  
2585  
2586  
2587  
2588  
2589  
2590  
2591  
2592  
2593  
2594  
2595  
2596  
2597  
2598  
2599  
2600

006412 005212  
006414 022712 000103  
006420 001036  
006422 005000  
006424 005010  
006426 105110  
006430 105100  
006432 005200  
006434 005720  
006436 000277  
006440 000250  
006442 105730  
006444 102402  
006446 101401  
006450 100404  
  
006452 012742 000165  
006456 005242  
006460 000000  
006462 000277  
006464 000244  
006466 105730  
006470 102403  
006472 103402  
006474 100401  
006476 001404  
  
006500 012742 000166  
006504 005242  
006506 000000  
006510 005720  
006512 005710  
006514 100404

```

*****
: 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.
*****
: TEST 103 TEST MODE 3 - BYTES W/ SOP NON-MODIFYING INSTS.
*****
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/ 764 <====
SNMB3A: MOV #165,-(R2) ;MOVE TO MAILBOX # ***** 165 *****
        INC -(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
        RCS SNMB3C
        BMI SNMB3C
        BEQ SNMB3D
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 751 <====
SNMB3C: MOV #166,-(R2) ;MOVE TO MAILBOX # ***** 166 *****
        INC -(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 <====

```

```
2601  
2602  
2603 006516 012742 000167  
2604 006522 005242  
2605 006524 000000  
2606  
2607  
2608  
2609  
2610  
2611  
2612  
2613  
2614  
2615  
2616  
2617
```

REPLACE THE MOVE INSTRUCTION WHICH FOLLOWS W/ 742  
: MOVE TO MAILBOX # \*\*\*\*\* 167 \*\*\*\*\*  
: SET MSGTYP TO FATAL ERROR  
: TSTB DID NOT INCREMENT RO CORRECTLY  
: OR SEQUENCE ERROR

```
*****  
: 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  
: 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.  
*****
```

```
2618 006526 005212  
2619 006530 022712 000104  
2620 006534 001017  
2621 006536 005000  
2622 006540 005010  
2623 006542 005120  
2624 006544 000277  
2625 006546 000244  
2626 006550 005740  
2627 006552 102402  
2628 006554 101401  
2629 006556 100404  
2630  
2631  
2632  
2633  
2634
```

TEST 104 TEST MODE 4 W/ SOP NON-MODIFYING INSTS  
\*\*\*\*\*

```
TST104: INC (R2) :UPDATE TEST NUMBER  
CMP #104,(R2) :SEQUENCE ERROR?  
BNE TST105-10 :BR TO ERROR HALT ON SEQ ERROR  
CLR RO :RO=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
```

```
2635 006560 012742 000170  
2636 006564 005242  
2637 006566 000000  
2638 006570 005700  
2639 006572 001404  
2640  
2641  
2642  
2643  
2644 006574 012742 000171  
2645 006600 005242  
2646 006602 000000  
2647
```

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

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

```
2648  
2649  
2650  
2651  
2652  
2653  
2654  
2655  
2656  
2657  
2658  
2659  
2660  
2661  
2662  
2663  
2664  
2665  
2666  
2667  
2668  
2669  
2670  
2671  
2672  
2673  
2674  
2675  
2676  
2677  
2678  
2679  
2680  
2681  
2682  
2683  
2684  
2685  
2686  
2687  
2688  
2689  
2690  
2691  
2692  
2693  
2694  
2695  
2696  
2697  
2698  
2699  
2700
```

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

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

2648  
2649  
2650  
2651  
2652  
2653  
2654  
2655  
2656  
2657  
2658  
2659 006604 005212  
2660 006606 022712 000105  
2661 006612 001022  
2662 006614 005000  
2663 006616 005010  
2664 006620 005110  
2665 006622 105100  
2666 006624 005200  
2667 006626 000277  
2668 006630 000250  
2669 006632 005750  
2670 006634 102402  
2671 006636 101401  
2672 006640 100404  
2673  
2674  
2675  
2676  
2677 006642  
2678 006642 012742 000172  
2679 006646 005242  
2680 006650 000000  
2681 006652 005200  
2682 006654 105100  
2683 006656 001404  
2684  
2685  
2686  
2687  
2688 006660 012742 000173  
2689 006664 005242  
2690 006666 000000  
2691

```
*****
: 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.
*****
: TEST 105 TEST MODE 5 W/ SOP NON-MODIFYING INSTS
*****
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/ 765 <====

SNM5A: MOV #172,-(R2) ;MOVE TO MAILBOX # ***** 172 *****
INC -(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/ 756 <====

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

2692  
2693  
2694  
2695  
2696  
2697  
2698  
2699  
2700  
2701  
2702  
2703 006670 005212  
2704 006672 022712 000106  
2705 006676 001021  
2706 006700 005000  
2707 006702 005010  
2708 006704 005110  
2709 006706 105100  
2710 006710 000277  
2711 006712 000250  
2712 006714 005760 177401  
2713 006720 102402  
2714 006722 101401  
2715 006724 100404  
2716  
2717  
2718  
2719  
2720 006726  
2721 006726 012742 000174  
2722 006732 005242  
2723 006734 000000  
2724 006736 105100  
2725 006740 001404  
2726  
2727  
2728  
2729  
2730 006742 012742 000175  
2731 006746 005242  
2732 006750 000000  
2733

```
*****
: THIS TEST VERIFIES 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.
*****
: TEST 106 TEST MODE 6 W/ SOP NON-MODIFYING INSTS
*****
TST106: 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/ 765 <====
SNM6A: MOV #174,-(R2) ;MOVE TO MAILBOX # ***** 174 *****
        INC -(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/ 757 <====
        MOV #175,-(R2) ;MOVE TO MAILBOX # ***** 175 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;TST MODE 6 INCORRECTLY CHANGED RO
; OR SEQUENCE ERROR
```

2734  
2735  
2736  
2737  
2738  
2739  
2740  
2741  
2742  
2743  
2744  
2745  
2746  
2747  
2748  
2749  
2750  
2751  
2752  
2753  
2754  
2755  
2756  
2757  
2758  
2759  
2760  
2761  
2762  
2763  
2764  
2765  
2766  
2767  
2768  
2769  
2770  
2771  
2772  
2773  
2774  
2775

006752 005212  
006754 022712 000107  
006760 001021  
006762 005000  
006764 005010  
006766 005110  
006770 105100  
006772 000277  
006774 000250  
006776 005770 000001  
007002 102402  
007004 101401  
007006 100404  
  
007010  
007010 012742 000176  
007014 005242  
007016 000000  
007020 105100  
007022 001404  
  
007024 012742 000177  
007030 005242  
007032 000000

```

*****
: THIS TEST VERIFIES MODE 7 SOP NON-MODIFYING INSTRUCTIONS.
: IT USES A POINTER TO LOC. 0 STORED AT LOC. 400 TO TST LOC. 0.
: R0 IS SET TO 377 AND LOC. 0 IS TESTED THRU THE POINTER AT 400 USING
: R0 AND AN OFFSET OF 1.
*****
: TEST 107 TEST MODE 7 W/ SOP NON-MODIFYING INSTS.
*****
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/ 765 <====
SNM7A: MOV #176,-(R2) ;MOVE TO MAILBOX # ***** 176 *****
        INC -(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/ 757 <====
        MOV #177,-(R2) ;MOVE TO MAILBOX # ***** 177 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;TST MODE 7 INCORRECTLY CHANGED R0
; OR SEQUENCE ERROR
    
```

2776  
2777  
2778  
2779  
2780  
2781  
2782  
2783  
2784  
2785  
2786  
2787  
2788  
2789  
2790  
2791  
2792  
2793  
2794  
2795  
2796  
2797  
2798  
2799  
2800  
2801  
2802  
2803  
2804  
2805  
2806  
2807  
2808  
2809  
2810  
2811  
2812  
2813  
2814  
2815  
2816  
2817  
2818  
2819  
2820  
2821  
2822  
2823  
2824  
2825  
2826  
2827  
2828  
2829  
2830  
2831

000110

000200

000111

000201

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

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

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/ 772 <====  
MOV #200,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 200 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;ADD INST. FAILED W/ MODE 0  
: OR SEQUENCE ERROR

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

\*\*\*\*\*  
: TEST 111 MOV MODE 0 TO MODE 0  
\*\*\*\*\*

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/ 772 <====  
MOV #201,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 201 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;MOVE FAILED MODE 0 TO MODE 0  
: OR SEQUENCE ERROR

\*\*\*\*\*

2832  
2833  
2834  
2835  
2836  
2837  
2838  
2839  
2840 007124 005212  
2841 007126 022712 000112  
2842 007132 001016  
2843 007134 005000  
2844 007136 005004  
2845 007140 005204  
2846 007142 160400  
2847 007144 100003  
2848 007146 001402  
2849 007150 102401  
2850 007152 103404  
2851  
2852  
2853  
2854  
2855 007154  
2856 007154 012742 000202  
2857 007160 005242  
2858 007162 000000  
2859 007164 005200  
2860 007166 001404  
2861  
2862  
2863  
2864  
2865 007170 012742 000203  
2866 007174 005242  
2867 007176 000000  
2868

```

:
:      THIS TEST VERIFIES THE SUBTRACT INSTRUCTION WITH MODE 0,0.
: THIS TEST IS NECESSARY BECAUSE THIS PARTICULAR INSTRUCTION UTILIZES SOME
: UNIQUE MICROCODE.
:
:*****
:TEST 112      TEST SUB MODE 0,0
:*****
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=10C1
        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/ 770 <====

SUB0:   MOV     #202,-(R2)    ;MOVE TO MAILBOX # ***** 202 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT                    ;CONDITION CODE FAILED ON SUB

SUB0A:  INC     R0
        BEQ     TST113

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

        MOV     #203,-(R2)    ;MOVE TO MAILBOX # ***** 203 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT                    ;DATA RESULT OF SUB FAILED
: OR SEQUENCE ERROR
```



2869  
2870  
2871  
2872  
2873  
2874  
2875  
2876  
2877  
2878  
2879  
2880  
2881 007200 005212  
2882 007202 022712 000113  
2883 007206 001051  
2884 007210 005000  
2885 007212 010004  
2886 007214 001404  
2887  
2888  
2889  
2890  
2891 007216 012742 000204  
2892 007222 005242  
2893 007224 000000  
2894 007226 005200  
2895 007230 005100  
2896 007232 005104  
2897 007234 040004  
2898 007236 005304  
2899 007240 001404  
2900  
2901  
2902  
2903  
2904 007242 012742 000205  
2905 007246 005242  
2906 007250 000000  
2907 007252 050004  
2908 007254 005204  
2909 007256 005204  
2910 007260 001404  
2911  
2912  
2913  
2914  
2915 007262 012742 000206  
2916 007266 005242  
2917 007270 000000  
2918 007272 005000  
2919 007274 105100  
2920 007276 005004  
2921 007300 005104  
2922 007302 040004  
2923 007304 060004  
2924 007306 005204

```
*****
:
: 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.
:
: *****
: TEST 113 TEST ALL THE DOP INSTRUCTIONS W/ SOURCE MODE 0,0
: *****
TST113: INC (R2) ;UPDATE TEST NUMBER
CMP #113,(R2) ;SEQUENCE ERROR?
BNE TST114-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
MOV R0,R4 ;TRY MOVE MODE 0,0
BEQ DOP0A
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 775 <====
MOV #204,-(R2) ;MOVE TO MAILBOX # ***** 204 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;Z-BIT NOT SET
DOP0A: INC R0 ;R0=1
COM R0 ;R0=177776
COM R4 ;R4=177777
BIC R0,R4 ;TRY BIC: R4=1
DEC R4 ;R4=0
BEQ DOP0B
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 763 <====
MOV #205,-(R2) ;MOVE TO MAILBOX # ***** 205 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;BIC CLEAR RESULT INCORRECT
DOP0B: BIS R0,R4 ;TRY BIS: R4=177777
INC R4
INC R4 ;R4=0
BEQ DOP0C
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 753 <====
MOV #206,-(R2) ;MOVE TO MAILBOX # ***** 206 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF BIS INCORRECT
DOP0C: CLR R0 ;R0=0
COMB R0 ;R0=377
CLR R4 ;R4=0
COM R4 ;R4=177777
BIC R0,R4 ;R4=177400
ADD R0,R4 ;TRY ADD: R4=177777
INC R4 ;R4=0
```



2946  
2947  
2948  
2949  
2950  
2951  
2952  
2953  
2954 007342 005212  
2955 007344 022712 000114  
2956 007350 001024  
2957 007352 005000  
2958 007354 005010  
2959 007356 105110  
2960 007360 005220  
2961 007362 005400  
2962 007364 060037 000000  
2963 007370 100403  
2964 007372 001402  
2965 007374 102401  
2966 007376 103404  
2967  
2968  
2969  
2970  
2971 007400  
2972 007400 012742 000211  
2973 007404 005242  
2974 007406 000000  
2975 007410 105137 000000  
2976 007414 005337 000000  
2977 007420 001404  
2978  
2979  
2980  
2981  
2982 007422 012742 000212  
2983 007426 005242  
2984 007430 000000  
2985

```
*****
: THIS TEST VERIFIES MODE 0,X DOUBLE OPERAND INSTRUCTIONS. IT SETS
: DATA IN R0 AND LOCATION 0 AND OPERATES UPON IT USING DOP INSTRUCTIONS.
*****
: TEST 114 TEST MODE 0,X DOUBLE-OPERAND INSTRUCTIONS
*****
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/ 765 <====

DOP03A: MOV #211,-(R2) ;MOVE TO MAILBOX # ***** 211 *****
        INC -(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/ 754 <====

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

```
2986  
2987  
2988  
2989  
2990  
2991  
2992  
2993  
2994  
2995 007432 005212  
2996 007434 022712 000115  
2997 007440 001042  
2998 007442 005000  
2999 007444 005004  
3000 007446 005204  
3001 007450 020400  
3002 007452 003004  
3003  
3004  
3005  
3006  
3007 007454 012742 000213  
3008 007460 005242  
3009 007462 000000  
3010 007464 020004  
3011 007466 002404  
3012  
3013  
3014  
3015  
3016 007470 012742 000214  
3017 007474 005242  
3018 007476 000000  
3019 007500 005200  
3020 007502 020400  
3021 007504 001404  
3022  
3023  
3024  
3025  
3026 007506 012742 000215  
3027 007512 005242  
3028 007514 000000  
3029 007516 005000  
3030 007520 005100  
3031 007522 005004  
3032 007524 030004  
3033 007526 001404  
3034  
3035  
3036  
3037  
3038 007530 012742 000216  
3039 007534 005242  
3040 007536 000000  
3041 007540 005304
```

\*\*\*\*\*  
: 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.  
\*\*\*\*\*  
: TEST 115 TEST DOP NON-MODIFYING INST. W/ SOURCE MODE 0,0  
\*\*\*\*\*  
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/ 773 <====  
MOV #213,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 213 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT CORRECT FOR CMP  
DNM1: 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/ 765 <====  
MOV #214,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 214 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT CORRECT FOR CMP  
DNM2: 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/ 756 <====  
MOV #215,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 215 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT CORRECT (Z=1) FOR CMP  
DNM3: 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/ 745 <====  
MOV #216,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 216 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CC'S NOT CORRECT FOR BIT  
DNM4: DEC R4 ;R4=177777

```
3042 007542 030004 BIT R0,R4 ;TRY BIT AGAIN
3043 007544 100404 BMI TST116
3044
3045 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3046 ; CONDITIONAL BRANCH INST. AND <====
3047 ; REPLACE THE MOVE INSTRUCTION <====
3048 007546 012742 000217 MOV #217,-(R2) ;MOVE TO MAILBOX # ***** 217 *****
3049 007552 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3050 007554 000000 HALT ;CC'S NOT CORRECT FOR BIT
3051 ; OR SEQUENCE ERROR
3052
3053
```

```
*****
: 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.
*****
```

```
3054
3055 :TEST 116 TEST MODE 0,X DOUBLE-OPERAND NON-MODIFYING INSTS.
3056
3057 *****
```

```
3058 :TST116: INC (R2) ;UPDATE TEST NUMBER
3059 TST116: INC (R2) ;SEQUENCE ERROR?
3060 007556 005212 000116 CMP #116,(R2) ;BR TO ERROR HALT ON SEQ ERROR
3061 007560 022712 BNE TST117-10 ;R0=0
3062 007564 001022 CLR R0 ;LOC. 0=0
3063 007566 005000 CLR (R0) ;LOC. 0=177777
3064 007570 005010 COM (R0) ;R0=1
3065 007572 005110 INC R0 ;TRY CMP MODE 0,3
3066 007574 005200 CMP R0,#0 ;CC=0001
3067 007576 020037 000000 BMI DNM03A
3068 007602 100403 BEQ DNM03A
3069 007604 001402 BVS DNM03A
3070 007606 102401 BCS DNM03B
3071 007610 103404
```

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

```
3072
3073 DNM03A: MOV #220,-(R2) ;MOVE TO MAILBOX # ***** 220 *****
3074 007612 012742 000220 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3075 007616 005242 000220 HALT ;CC'S NOT SET CORRECTLY
3076 007620 000000
3077 007622 005300 DNM03B: DEC R0
3078 007624 001002 BNE DNM03C
3079 007626 005210 INC (R0)
3080 007630 001404 BEQ TST117
3081
3082
3083
3084
```

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

```
3085
3086 DNM03C: MOV #221,-(R2) ;MOVE TO MAILBOX # ***** 221 *****
3087 007632 012742 000221 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3088 007636 005242 000221 HALT ;DATA INCORRECTLY MODIFIED BY CMP
3089 007640 000000 ; OR SEQUENCE ERROR
3090
3091
3092
```

3093  
3094  
3095  
3096  
3097  
3098  
3099  
3100  
3101  
3102  
3103 007642 005212  
3104 007644 022712 000117  
3105 007650 001007  
3106 007652 005000  
3107 007654 005100  
3108 007656 005004  
3109 007660 005014  
3110 007662 005214  
3111 007664 061400  
3112 007666 001404  
3113  
3114  
3115  
3116  
3117 007670 012742 000222  
3118 007674 005242  
3119 007676 000000  
3120

```
*****
:
:   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.
:
:*****
:TEST 117      TEST MODE 1 w/ DOP INST.
:*****
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/ 771 <====
: MOVE TO MAILBOX # ***** 222 *****
: SET MSGTYP TO FATAL ERROR
: RESULT QF ADD INCORRECT
: OR SEQUENCE ERROR
```

3121  
3122  
3123  
3124  
3125  
3126  
3127  
3128  
3129  
3130  
3131 007700 005212  
3132 007702 022712 000120  
3133 007706 001007  
3134 007710 005000  
3135 007712 005010  
3136 007714 005110  
3137 007716 005004  
3138 007720 151004  
3139 007722 105104  
3140 007724 001404  
3141  
3142  
3143  
3144  
3145 007726 012742 000223  
3146 007732 005242  
3147 007734 000000  
3148

```
*****  
: 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.  
*****  
: TEST 120 TEST MODE 1 - EVEN BYTE W/ DOP INSTS.  
*****  
TST120: 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/ 771 <====  
MOV #223,-(R2) ;MOVE TO MAILBOX # ***** 223 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;RESULT OF BISB IS INCORRECT  
: OR SEQUENCE ERROR
```

3149  
3150  
3151  
3152  
3153  
3154  
3155  
3156  
3157  
3158  
3159  
3160 007736 005212  
3161 007740 022712 000121  
3162 007744 001007  
3163 007746 005000  
3164 007750 005010  
3165 007752 005110  
3166 007754 005004  
3167 007756 105104  
3168 007760 121004  
3169 007762 001404  
3170  
3171  
3172  
3173  
3174 007764 012742 000224  
3175 007770 005242  
3176 007772 000000  
3177

```
*****
: 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 CMPB INSTRUCTION IS USED THE RESULTS VERIFIED.
*****
: TEST 121 TEST MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING INST.
*****
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/ 771 <====
: MOVE TO MAILBOX # ***** 224 *****
: SET MSGTYP TO FATAL ERROR
: RESULT OF CMPB INCORRECT
: OR SEQUENCE ERROR
```



3178  
3179  
3180  
3181  
3182  
3183  
3184  
3185  
3186  
3187  
3188  
3189  
3190  
3191  
3192  
3193  
3194  
3195  
3196  
3197  
3198  
3199  
3200  
3201  
3202  
3203  
3204  
3205  
3206  
3207  
3208  
3209  
3210  
3211  
3212  
3213  
3214  
3215  
3216  
3217  
3218  
3219  
3220  
3221  
3222

007774 005212  
007776 022712 000122  
010002 001020  
010004 005000  
010006 005010  
010010 105110  
010012 005110  
010014 005004  
010016 005104  
010020 111004  
010022 005704  
010024 001404  
  
010026 012742 000225  
010032 005242  
010034 000000  
010036 005110  
010040 111004  
010042 100404  
  
010044 012742 000226  
010050 005242  
010052 000000

```
*****  
: THIS TEST VERIFIES MODE 1,0 MOV B INSTRUCTIONS  
: WHICH ADDRESS EVEN BYTES. LOC. 0 IS SET TO 177400, R0 IS CLEARED AND  
: R4 IS SET TO -1. MOV B 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.  
*****  
: TEST 122 TEST MOV INSTRUCTION MODE 1,0 EVEN BYTE  
*****  
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/ 767 <====  
: MOVE TO MAILBOX # ***** 225 *****  
: SET MSGTYP TO FATAL ERROR  
: MOV B SHOULD SIGN X-TEND  
: LOC 0=177777  
: DO MOV B W/ EVEN BYTE  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 760 <====  
: MOVE TO MAILBOX # ***** 226 *****  
: SET MSGTYP TO FATAL ERROR  
: MOV B SHOULD SIGN X-TEND  
: OR SEQUENCE ERROR
```

```
TST122: INC (R2)  
CMP #122,(R2)  
BNE TST123-10  
CLR R0  
CLR (R0)  
COMB (R0)  
COM (R0)  
CLR R4  
COM R4  
MOVB (R0),R4  
TST R4  
BEQ DOP1  
  
DOP1: COM (R0)  
MOVB (R0),R4  
BMI TST123
```

3223  
3224  
3225  
3226  
3227  
3228  
3229  
3230  
3231  
3232  
3233  
3234 010054 005212  
3235 010056 022712 000123  
3236 010062 001010  
3237 010064 005000  
3238 010066 005010  
3239 010070 005004  
3240 010072 005204  
3241 010074 105114  
3242 010076 151410  
3243 010100 005210  
3244 010102 001404  
3245  
3246  
3247  
3248  
3249 010104 012742 000227  
3250 010110 005242  
3251 010112 000000  
3252

```
*****
: 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 BISB INSTRUCTION USES THE DATA IN BYTE 1 TO SET BYTE 0.
: THE RESULT IS CHECKED BY INCREMENTING THE WORD (LOC. 0) TO ZERO.
*****
: TEST 123 TEST MODE 1-ODD BYTE W/ DOP INSTS.
*****
TST123: 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/ 770 <====
        MOV #227,-(R2) ;MOVE TO MAILBOX # ***** 227 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;RESULT OF BISB INCORRECT
; OR SEQUENCE ERROR
```

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

010114 005212  
010116 022712 000124  
010122 001015  
010124 005000  
010126 005010  
010130 005110  
010132 012004  
010134 005204  
010136 001404  
  
010140 012742 000230  
010144 005242  
010146 000000  
010150 005300  
010152 005300  
010154 001404  
  
010156 012742 000231  
010162 005242  
010164 000000

```
*****
: 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.
*****
: TEST 124 TEST MODE 2 W/ DOP INSTS.
*****
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/ 772 <====
        MOV #230,-(R2) ;MOVE TO MAILBOX # ***** 230 *****
        INC -(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/ 763 <====
        MOV #231,-(R2) ;MOVE TO MAILBOX # ***** 231 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;REGISTER NOT INCREMENTED IN MODE 2
; OR SEQUENCE ERROR
```

3291  
3292  
3293  
3294  
3295  
3296  
3297  
3298  
3299  
3300  
3301  
3302  
3303  
3304 010166 005212  
3305 010170 022712 000125  
3306 010174 001016  
3307 010176 005000  
3308 010200 010010  
3309 010202 005110  
3310 010204 142010  
3311 010206 105737 000001  
3312 010212 001404  
3313  
3314  
3315  
3316  
3317 010214 012742 000232  
3318 010220 005242  
3319 010222 000000  
3320 010224 105137 000000  
3321 010230 001404  
3322  
3323  
3324  
3325  
3326 010232 012742 000233  
3327 010236 005242  
3328 C10240 000000  
3329

```
*****
:
:   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.
:
:*****
:TEST 125      TEST MODE 2 - EVEN BYTE W/ DOP INST.
:*****
TST125: INC      (R2)          ;UPDATE TEST NUMBER
        CMP      #125,(R2)    ;SEQUENCE ERROR?
        BNE     TST126-10     ;BR TO ERROR HALT ON SEQ ERROR
        CLR     R0            ;R0=0
        MOV     R0,(R0)       ;LOC. 0=0
        COM     (R0)          ;LOC. 0=177777
        BICB   (R0)+,(R0)    ;TRY TO CLEAR BYTE 1 FROM BYTE 0 W/ BICB
        TSTB   @#1           ;CHECK RESULT
        BEQ    DOPB2A
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:   CONDITIONAL BRANCH INST. AND <====
:   REPLACE THE MOVE INSTRUCTION <====
:   WHICH FOLLOWS W/ 771 <====
:
:   MOV     #232,-(R2)        ;MOVE TO MAILBOX # ***** 232 *****
:   INC     -(R2)            ;SET MSGTYP TO FATAL ERROR
:   HALT                                ;BICB DESTINATION INCORRECT
DOPB2A: COMB   @#0           ;CHECK BICB SOURCE
        BEQ    TST126
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:   CONDITIONAL BRANCH INST. AND <====
:   REPLACE THE MOVE INSTRUCTION <====
:   WHICH FOLLOWS W/ 762 <====
:
:   MOV     #233,-(R2)        ;MOVE TO MAILBOX # ***** 233 *****
:   INC     -(R2)            ;SET MSGTYP TO FATAL ERROR
:   HALT                                ;BICB SOURCE INCORRECTLY CHANGED
:   OR SEQUENCE ERROR
```

3330  
3331  
3332  
3333  
3334  
3335  
3336  
3337  
3338  
3339  
3340 010242 005212  
3341 010244 022712 000126  
3342 010250 001017  
3343 010252 005000  
3344 010254 005004  
3345 010256 005010  
3346 010260 005110  
3347 010262 105120  
3348 010264 112004  
3349 010266 005204  
3350 010270 001404  
3351  
3352  
3353  
3354  
3355 010272 012742 000234  
3356 010276 005242  
3357 010300 000000  
3358 010302 005740  
3359 010304 005700  
3360 010306 001404  
3361  
3362  
3363  
3364  
3365 010310 012742 000235  
3366 010314 005242  
3367 010316 000000  
3368

```
*****
:
:   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 MOV BUSES R0 TO MOVE BYTE 1 TO R4. AN INCREMENT
: IS USED TO CHECK THAT THE PROPER BYTE WAS MOVED AND SIGN X-TENDED.
:
: *****
: TEST 126      TEST MODE 2 - ODD BYTE W/ DOP INST.
: *****
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
        MOV B   (R0)+,R4    ;TRY DOP MODE 2 W/ ODD BYTE
        INC     R4          ;CHECK RESULT OF MOV B
        BEQ     DOPB2B
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:   CONDITIONAL BRANCH INST. AND <====
:   REPLACE THE MOVE INSTRUCTION <====
:   WHICH FOLLOWS W/ 770 <====
:
:   MOVE TO MAILBOX # ***** 234 *****
:   SET MSGTYP TO FATAL ERROR
:   RESULT OF MOV B INCORRECT
:   BUMP R0 DOWN BY 2
:   CHECK R0
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:   CONDITIONAL BRANCH INST. AND <====
:   REPLACE THE MOVE INSTRUCTION <====
:   WHICH FOLLOWS W/ 761 <====
:
:   MOVE TO MAILBOX # ***** 235 *****
:   SET MSGTYP TO FATAL ERROR
:   MODE 2 BYTE DID NOT INCREMENT REG. CORRECTLY
:   OR SEQUENCE ERROR
: *****
```

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

010320 005212  
010322 022712 000127  
010326 001011  
010330 012737 052525 000000  
010336 012700 125252  
010342 053700 000000  
010346 005200  
010350 001404  
  
010352 012742 000236  
010356 005242  
010360 000000  
  
010362 005212  
010364 022712 000130  
010370 001011  
010372 012737 052652 000000  
010400 005000  
010402 153700 000000  
010406 022700 000252  
010412 001404  
  
010414 012742 000237  
010420 005242  
010422 000000

\*\*\*\*\*  
: 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.  
\*\*\*\*\*  
: TEST 127 TEST MODE 3 W/ DOP INSTS.  
\*\*\*\*\*  
TST127: INC (R2) ;UPDATE 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/ 767 <=====  
MOV #236,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 236 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;BIS W/ MODE 3 INCORRECT RESULT  
; OR SEQUENCE ERROR

\*\*\*\*\*  
: 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.  
\*\*\*\*\*  
: TEST 130 TEST MODE 3 - EVEN BYTE W/ DOP INSTS.  
\*\*\*\*\*  
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/ 767 <=====  
MOV #237,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 237 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;BISB W/ MODE 3 - EVEN BYTE FAILED  
; OR SEQUENCE ERROR

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

010424 005212  
010426 022712 000131  
010432 001011  
010434 012737 052652 000000  
010442 005000  
010444 153700 000001  
010450 022700 000125  
010454 001404  
  
010456 012742 000240  
010462 005242  
010464 000000  
  
010466 005212  
010470 022712 000132  
010474 001017  
010476 005000  
010500 105100  
010502 000263  
010504 132700 000200  
010510 001403  
010512 102402  
010514 103001  
010516 100404  
  
010520  
010520 012742 000241  
010524 005242  
010526 000000  
010530 105100  
010532 001404  
  
010534 012742 000242

```
*****
:
: 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.
:
: *****
: TEST 131 TEST MODE 3 - ODD BYTE W/ DOP INSTS.
: *****
TST131: INC (R2) ;UPDATE TEST NUMBER
        CMP #131,(R2) ;SEQUENCE ERROR?
        BNE TST132-10 ;BR TO ERROR HALT ON SEQ ERROR
        MOV #52652,@#0 ;MOVE 1'S AND 0'S PATTERN TO LOC 0
        CLR R0 ;R0=0
        BISB @#1,R0 ;TRY R0=152 W/ MODE 3 - ODD BYTE
        CMP #125,R0 ;R0=125?
        BEQ TST132
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 767 <====
        MOV #240,-(R2) ;MOVE TO MAILBOX # ***** 240 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;BISB W/ MODE 3 - ODD BYTE FAILED
: OR SEQUENCE ERROR
:
: *****
: TEST 132 TEST DEST. MODE 0-BYTE W/ DOP NON-MODIFYING MST
: *****
TST132: 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 #200,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/ 767 <====
DNMB0A: MOV #241,-(R2) ;MOVE TO MAILBOX # ***** 241 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;CC'S INCORRECT
DNMB0B: COMB R0 ;CHECK DESTINATION DATA
        BEQ TST133
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 761 <====
        MOV #242,-(R2) ;MOVE TO MAILBOX # ***** 242 *****
```

```
3478 010540 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
3479 010542 000000          HALT                    ;DEST. DATA MODIFIED
3480                                     ; OR SEQUENCE ERROR
3481
3482
3483 :*****
3484 :TEST 133      TEST DEST. MODE 1 W/ DOP NON-MODIFYING INST
3485 :*****
3485 010544 005212          TST133: INC      (R2)          ;UPDATE TEST NUMBER
3486 010546 022712 000133    CMP      #133,(R2)       ;SEQUENCE ERROR?
3487 010552 001017          BNE     TST134-10      ;BR TO ERROR HALT ON SEQ ERROR
3488 010554 005000          CLR     R0             ;R0=0
3489 010556 005010          CLR     (R0)           ;LOC. 0=0
3490 010560 000241          CLC                    ;CLEAR C BIT
3491 010562 032710 177777    BIT     #177777,(R0)    ;TRY DOPNM DEST. MODE 1
3492 010566 100403          BMI     DNM1A          ;BR TO ERROR IF N BIT SET
3493 010570 102402          BVS     DNM1A          ;BR TO ERROR IF V BIT SET
3494 010572 103401          BCS     DNM1A          ;BR TO ERROR IF C BIT SET
3495 010574 001404          BEQ     DNM1B
3496
3497                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3498                                     ; CONDITIONAL BRANCH INST. AND <====
3499                                     ; REPLACE THE MOVE INSTRUCTION <====
3500                                     ; WHICH FOLLOWS W/ 767 <====
3500 010576          DNM1A:
3501 010576 012742 000243    MOV     #243,-(R2)      ;MOVE TO MAILBOX # ***** 243 *****
3502 010602 005242          INC     -(R2)          ;SET MSGTYP TO FATAL ERROR
3503 010604 000000          HALT                    ;COND. CODES INCORRECT
3504 010606 005710          DNM1B: TST     (R0)       ;CHECK TEST DATA
3505 010610 001404          BEQ     TST134
3506
3507                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3508                                     ; CONDITIONAL BRANCH INST. AND <====
3509                                     ; REPLACE THE MOVE INSTRUCTION <====
3510                                     ; WHICH FOLLOWS W/ 761 <====
3510 010612 012742 000244    MOV     #244,-(R2)      ;MOVE TO MAILBOX # ***** 244 *****
3511 010616 005242          INC     -(R2)          ;SET MSGTYP TO FATAL ERROR
3512 010620 000000          HALT                    ;DESTINATION DATA MODIFIED
3513                                     ; OR SEQUENCE ERROR
3514
3515 :*****
3516 :TEST 134      TEST DEST, MODE 2 W/ DOP NON-MODIFYING INST.
3517 :*****
3518 010622 005212          TST134: INC      (R2)          ;UPDATE TEST NUMBER
3519 010624 022712 000134    CMP     #134,(R2)       ;SEQUENCE ERROR?
3520 010630 001027          BNE     TST135-10      ;BR TO ERROR HALT ON SEQ ERROR
3521 010632 005000          CLR     R0             ;R0=0
3522 010634 005010          CLR     (R0)           ;LOC. 0=0
3523 010636 052710 125252    BIS     #125252,(R0)    ;LCC. 0=125252
3524 010642 032720 077777    BIT     #77777,(R0)+   ;TRY DOPNM INST W/ MODE 2
3525 010646 102402          BVS     DNM2A          ;BR TO ERROR IF V BIT SET
3526 010650 001401          BEQ     DNM2A          ;BR TO ERROR IF Z-BIT SET
3527 010652 100004          BPL     DNM2B
3528
3529                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3530                                     ; CONDITIONAL BRANCH INST. AND <====
3531                                     ; REPLACE THE MOVE INSTRUCTION <====
3532                                     ; WHICH FOLLOWS W/ 767 <====
3532 010654          DNM2A:
3533 010654 012742 000245    MOV     #245,-(R2)      ;MOVE TO MAILBOX # ***** 245 *****
```





3590 011002 005200  
3591 011004 132720 000201  
3592 011010 001402  
3593 011012 102401  
3594 011014 100004  
3595  
3596  
3597  
3598  
3599 011016  
3600 011016 012742 000252  
3601 011022 005242  
3602 011024 000000  
3603 011026 005300  
3604 011030 005300  
3605 011032 001404  
3606  
3607  
3608  
3609  
3610 011034 012742 000253  
3611 011040 005242  
3612 011042 000000  
3613 011044 022710 052652  
3614 011050 001404  
3615  
3616  
3617  
3618  
3619 011052 012742 000254  
3620 011056 005242  
3621 011060 000000  
3622  
3623  
3624  
3625  
3626  
3627  
3628 011062 005212  
3629 011064 022712 000136  
3630 011070 001050  
3631 011072 005000  
3632 011074 005010  
3633 011076 052710 125125  
3634 011102 105100  
3635 011104 005200  
3636 011106 005010  
3637 011110 000263  
3638 011112 132730 000201  
3639 011116 001403  
3640 011120 102402  
3641 011122 103001  
3642 011124 100004  
3643  
3644  
3645

DNMB2C: INC R0 ;R0=1  
BITB #201,(R0)+ ;TRY DOPNM INST. W/MODE 2-ODD BYTE  
BEQ DNMB2D ;BR TO ERROR IF Z-BIT SET  
BVS DNMB2D ;BR TO ERROR IF V-BIT SET  
BPL DNMB2E  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 745 <====  
DNMB2D: MOV #252,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 252 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;COND. CODES INCORRECT  
DNMB2E: DEC R0 ;DEC R0 TO CHECK IT.  
DEC R0  
BEQ DNMB2F  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 736 <====  
MOV #253,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 253 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;DEST. REGISTER NOT INCREMENTED BY 1  
DNMB2F: CMP #52652,(R0) ;CHECK DEST. DATA IS UNMODIFIED  
BEQ TST136  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 727 <====  
MOV #254,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 254 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;DEST. DATA WAS MODIFIED.  
; OR SEQUENCE ERROR  
\*\*\*\*\*  
:TEST 136 TEST DEST. MODE 3-BYTES W/DOP NON-MODIFYING INST.  
\*\*\*\*\*  
TST136: INC (R2) ;UPDATE TEST NUMBER  
CMP #136,(R2) ;SEQUENCE ERROR?  
BNE TST137-10 ;BR TO ERROR HALT ON SEQ ERROR  
CLR R0 ;R0=0  
CLR (R0) ;LOC. 0=0  
BIS #125125,(R0) ;LOC. 0=125125  
COMB R0 ;R0=377  
INC R0 ;RC=400  
CLR (R0) ;LOC. 400=0  
+SEC!SEV ;C-BIT=V-BIT=1  
BITB #201,@(R0)+ ;TRY DOPNM W/MODE 3-EVEN BYTE  
BEQ DNMB3A ;BR TO ERROR IF Z BIT SET  
BVS DNMB3A ;BR TO ERROR IF V BIT SET  
BCC DNMB3A ;BR TO ERROR IF C BIT CLEAR  
BPL DNMB3B  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====

```
3646  
3647 011126  
3648 011126 012742 000255  
3649 011132 005242  
3650 011134 000000  
3651 011136 022700 000402  
3652 011142 001404  
3653  
3654  
3655  
3656  
3657 011144 012742 000256  
3658 011150 005242  
3659 011152 000000  
3660 011154 005200  
3661 011156 005200  
3662 011160 132730 000201  
3663 011164 001402  
3664 011166 102401  
3665 011170 100404  
3666  
3667  
3668  
3669  
3670 011172  
3671 011172 012742 000257  
3672 011176 005242  
3673 011200 000000  
3674 011202 005004  
3675 011204 022714 125125  
3676 011210 001404  
3677  
3678  
3679  
3680  
3681 011212 012742 000260  
3682 011216 005242  
3683 011220 000000  
3684  
3685  
3686  
3687  
3688  
3689 011222 005212  
3690 011224 022712 000137  
3691 011230 001033  
3692 011232 005000  
3693 011234 005010  
3694 011236 052710 125252  
3695 011242 052700 000002  
3696 011246 000277  
3697 011250 032740 020000  
3698 011254 100403  
3699 011256 102402  
3700 011260 103001  
3701 011262 001004
```

DNMB3A: ; WHICH FOLLOWS W/ 762 <====  
MOV #255, -(R2) ; MOVE TO MAILBOX # \*\*\*\*\* 255 \*\*\*\*\*  
INC -(R2) ; SET MSGTYP TO FATAL ERROR  
HALT ; COND. CODES INCORRECT  
DNMB3B: CMP #402, R0 ; CHECK DEST. REGISTER INC. BY 2 AND INC BY 2 AGAIN  
BEQ DNMB3C ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 753 <====  
MOV #256, -(R2) ; MOVE TO MAILBOX # \*\*\*\*\* 256 \*\*\*\*\*  
INC -(R2) ; SET MSGTYP TO FATAL ERROR  
HALT ; DEST. REGISTER NOT INCREMENTED BY 2  
DNMB3C: INC R0 ; R0=404  
INC R0 ; TRY DOPNM DEST MODE 3-BYTE (ODD)  
BITB #201, @ (R0)+ ; BR TO ERROR IF Z BIT SET  
BEQ DNMB3D ; BR TO ERROR IF V BIT SET  
BVS DNMB3D  
BMI DNMB3E ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 740 <====  
DNMB3D: MOV #257, -(R2) ; MOVE TO MAILBOX # \*\*\*\*\* 257 \*\*\*\*\*  
INC -(R2) ; SET MSGTYP TO FATAL ERROR  
HALT ; COND. CODES INCORRECT  
DNMB3E: CLR R4 ; R4=0  
CMP #125125, (R4) ; CHECK DEST. DATA  
BEQ TST137 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 730 <====  
MOV #260, -(R2) ; MOVE TO MAILBOX # \*\*\*\*\* 260 \*\*\*\*\*  
INC -(R2) ; SET MSGTYP TO FATAL ERROR  
HALT ; DEST. DATA MODIFIED  
; OR SEQUENCE ERROR

\*\*\*\*\*  
: TEST 137 TEST DEST. MODE 4 W/DOP NON-MODIFYING INST.  
\*\*\*\*\*  
TST137: INC (R2) ; UPDATE TEST NUMBER  
CMP #137, (R2) ; SEQUENCE ERROR?  
BNE TST140-10 ; BR TO ERROR HALT ON SEQ ERROR  
CLR R0 ; R0=0  
CLR (R0) ; LOC. 0=0  
BIS #125252, (R0) ; LOC. 0=125125  
BIS #2, R0 ; R0=2  
SCC ; SET ALL COND. CODE BITS  
BIT #20000, -(R0) ; TRY DOPNM W/ MODE 4  
BMI DNMB4A ; BR TO ERROR IF N-BIT SET  
BVS DNMB4A ; BR TO ERROR IF V-BIT SET  
BCC DNMB4A ; BR TO ERROR IF C-BIT CHAR  
BNE DNMB4B

```
3702 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3703 ; CONDITIONAL BRANCH INST. AND <====
3704 ; REPLACE THE MOVE INSTRUCTION <====
3705 ; WHICH FOLLOWS W/ 763 <====
3706 011264 DNM4A:
3707 011264 012742 000261 MOV #261,-(R2) ;MOVE TO MAILBOX # ***** 261 *****
3708 011270 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3709 011272 000000 HALT ;COND. CODES INCORRECT
3710 011274 005700 DNM4B: TST R0 ;CHECK DEST. REGISTER
3711 011276 001404 BEQ DNM4C
3712 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3713 ; CONDITIONAL BRANCH INST. AND <====
3714 ; REPLACE THE MOVE INSTRUCTION <====
3715 ; WHICH FOLLOWS W/ 755 <====
3716 011300 012742 000262 MOV #262,-(R2) ;MOVE TO MAILBOX # ***** 262 *****
3717 011304 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3718 011306 000000 HALT ;DEST. REGISTER NOT DECREMENTED BY 2
3719 011310 022737 125252 000000 DNM4C: CMP #125252,@#0 ;CHECK DEST. DATA
3720 011316 001404 BEQ TST140
3721 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3722 ; CONDITIONAL BRANCH INST. AND <====
3723 ; REPLACE THE MOVE INSTRUCTION <====
3724 ; WHICH FOLLOWS W/ 745 <====
3725 011320 012742 000263 MOV #263,-(R2) ;MOVE TO MAILBOX # ***** 263 *****
3726 011324 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3727 011326 000000 HALT ;DEST. DATA MODIFIED
3728 ; OR SEQUENCE ERROR
3729
3730 *****
3731 ;TEST 140 TEST DEST. MODE 4-BYTE W/ DOP NON-MODIFYING INST.
3732 *****
3733 011330 005212 TST140: INC (R2) ;UPDATE TEST NUMBER
3734 011332 022712 000140 CMP #140,(R2) ;SEQUENCE ERROR?
3735 011336 001051 BNE TST141-10 ;BR TO ERROR HALT ON SEQ ERROR
3736 011340 005000 CLR R0 ;R0=0
3737 011342 005010 CLR (R0) ;LOC. 0=0
3738 011344 052710 052652 BIS #52652,(R0) ;LOC. 0=52652
3739 011350 052700 000002 BIS #2,R0 ;R0=2
3740 011354 000257 CCC ;COND. CODES=0
3741 011356 132740 000201 BITB #201,-(R0) ;TRY DOPNM INST W/MODE 4 ODD BYTE
3742 011362 102403 BVS DNMB4A ;BR TO ERROR IF V BIT SET
3743 011364 001402 BEQ DNMB4A ;BR TO ERROR IF Z BIT SET
3744 011366 103401 BCS DNMB4A ;BR TO ERROR IF C BIT SET
3745 011370 001004 BNE DNMB4B
3746 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3747 ; CONDITIONAL BRANCH INST. AND <====
3748 ; REPLACE THE MOVE INSTRUCTION <====
3749 ; WHICH FOLLOWS W/ 763 <====
3750 011372 DNMB4A:
3751 011372 012742 000264 MOV #264,-(R2) ;MOVE TO MAILBOX # ***** 264 *****
3752 011376 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3753 011400 000000 HALT ;COND. CODES INCORRECT
3754 011402 022700 000001 DNMB4B: CMP #1,R0 ;CHECK DEST. REGISTER
3755 011406 001404 BEQ DNMB4C
3756 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3757 ; CONDITIONAL BRANCH INST. AND <====
```

```
3758  
3759  
3760 011410 012742 000265      MOV      #265,-(R2)      ;  
3761 011414 005242      INC      -(R2)          ;   REPLACE THE MOVE INSTRUCTION <=====  
3762 011416 000000      HALT                    ;   WHICH FOLLOWS W/ 754      <=====  
3763 011420 132740 000201      DNMB4C: BITB      #201,-(R0) ;  
3764 011424 001401      BEQ      DNMB4D        ;MOVE TO MAILBOX # ***** 265 *****  
3765 011426 100404      BMI      DNMB4E        ;SET MSGTYP TO FATAL ERROR  
3766  
3767  
3768  
3769  
3770 011430      DNMB4D:                ;   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
3771 011430 012742 000266      MOV      #266,-(R2)      ;   CONDITIONAL BRANCH INST. AND <=====  
3772 011434 005242      INC      -(R2)          ;   REPLACE THE MOVE INSTRUCTION <=====  
3773 011436 000000      HALT                    ;   WHICH FOLLOWS W/ 744      <=====  
3774 011440 005700      DNMB4E: TST      R0      ;  
3775 011442 001404      BEQ      DNMB4F        ;MOVE TO MAILBOX # ***** 266 *****  
3776  
3777  
3778  
3779  
3780 011444 012742 000267      MOV      #267,-(R2)      ;  
3781 011450 005242      INC      -(R2)          ;SET MSGTYP TO FATAL ERROR  
3782 011452 000000      HALT                    ;COND. CODES INCORRECT  
3783 011454 022710 052652      DNMB4F: CMP      #52652,(R0) ;CHECK DEST. REGISTER  
3784 011460 001404      BEQ      TST141        ;  
3785  
3786  
3787  
3788  
3789 011462 012742 000270      MOV      #270,-(R2)      ;   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
3790 011466 005242      INC      -(R2)          ;   CONDITIONAL BRANCH INST. AND <=====  
3791 011470 000000      HALT                    ;   REPLACE THE MOVE INSTRUCTION <=====  
3792  
3793  
3794  
3795  
3796  
3797 011472 005212      TST141: INC      (R2)      ;   WHICH FOLLOWS W/ 727      <=====  
3798 011474 022712 000141      CMP      #141,(R2)      ;MOVE TO MAILBOX # ***** 270 *****  
3799 011500 001034      BNE      TST142-10      ;SET MSGTYP TO FATAL ERROR  
3800 011502 005000      CLR      R0            ;DEST. DATA MODIFIED  
3801 011504 005010      CLR      (R0)          ;OR SEQUENCE ERROR  
3802 011506 052710 100000      BIS      #100000,(R0)    ;  
3803 011512 052700 000402      BIS      #402,R0        ;  
3804 011516 000277      SCC                    ;  
3805 011520 032750 100000      BIT      #100000,@-(R0) ;SET ALL COND. CODE BITS  
3806 011524 102403      BVS      DN15A         ;TRY DOPNM W/MODE 5  
3807 011526 103002      BCC      DN15A         ;BR TO ERROR IF V-BIT SET  
3808 011530 001401      BEQ      DN15A         ;BR TO ERROR IF C-BIT CLEAR  
3809 011532 100404      BMI      DN15B         ;BR TO ERROR IF Z-BIT SET  
3810  
3811  
3812  
3813
```

```
3814 011534          DNM5A:
3815 011534 012742 000271      MOV #271,-(R2)      ;MOVE TO MAILBOX # ***** 271 *****
3816 011540 005242          INC -(R2)           ;SET MSGTYP TO FATAL ERROR
3817 011542 000000          HALT                ;COND, CODES INCORRECT
3818 011544 022700 000400      DNM5B: CMP #400,R0    ;CHECK DEST. REGISTER
3819 011550 001404          BEQ DNM5C           ;
3820                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3821                                     ; CONDITIONAL BRANCH INST. AND <====
3822                                     ; REPLACE THE MOVE INSTRUCTION <====
3823                                     ; WHICH FOLLOWS W/ 754 <====
3824 011552 012742 000272      MOV #272,-(R2)      ;MOVE TO MAILBOX # ***** 272 *****
3825 011556 005242          INC -(R2)           ;SET MSGTYP TO FATAL ERROR
3826 011560 000000          HALT                ;DEST. REGISTER NOT DECREMENTED BY 2
3827 011562 022737 100000 000000 DNM5C: CMP #100000,@#0    ;CHECK DESTINATION DATA
3828 011570 001404          BEQ TST142         ;
3829                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3830                                     ; CONDITIONAL BRANCH INST. AND <====
3831                                     ; REPLACE THE MOVE INSTRUCTION <====
3832                                     ; WHICH FOLLOWS W/ 744 <====
3833 011572 012742 000273      MOV #273,-(R2)      ;MOVE TO MAILBOX # ***** 273 *****
3834 011576 005242          INC -(R2)           ;SET MSGTYP TO FATAL ERROR
3835 011600 000000          HALT                ;DEST. DATA INCORRECTLY MODIFIED
3836                                     ; OR SEQUENCE ERROR
3837
3838 :*****
3839 :TEST 142 TEST DEST. MODE 6 W/DOP NON-MODIFYING INST.
3840 :*****
3841 011602 005212          TST142: INC (R2)      ;UPDATE TEST NUMBER
3842 011604 022712 000142      CMP #142,(R2)       ;SEQUENCE ERROR?
3843 011610 001033          BNE TST143-10      ;BR TO ERROR HALT ON SEQ ERROR
3844 011612 005000          CLR R0             ;R0=0
3845 011614 005010          CLR (R0)           ;LOC> 0=0
3846 011616 052710 000001      BIS #1,(R0)         ;LOC. 0=1
3847 011622 005100          COM R0             ;R0=-1 C-BIT=1
3848 011624 032760 000001 000001 BIT #1,1(R0)         ;TRY DOPNM W/MODE 6
3849 011632 001403          BEQ DNM6A          ;BR TO ERROR IF Z-BIT SET
3850 011634 102402          BVS DNM6A          ;BR TO ERROR IF V-BIT SET
3851 011636 103001          BCC DNM6A          ;BR TO ERROR IF C-BIT CLEAR
3852 011640 100004          BPL DNM6B          ;
3853                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3854                                     ; CONDITIONAL BRANCH INST. AND <====
3855                                     ; REPLACE THE MOVE INSTRUCTION <====
3856                                     ; WHICH FOLLOWS W/ 764 <====
3857 011642          DNM6A:
3858 011642 012742 000274      MOV #274,-(R2)      ;MOVE TO MAILBOX # ***** 274 *****
3859 011646 005242          INC -(R2)           ;SET MSGTYP TO FATAL ERROR
3860 011650 000000          HALT                ;COND CODES INCORRECT
3861 011652 022700 177777      DNM6B: CMP #-1,R0    ;CHECK DEST. REGISTER
3862 011656 001404          BEQ DNM6C          ;
3863                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3864                                     ; CONDITIONAL BRANCH INST. AND <====
3865                                     ; REPLACE THE MOVE INSTRUCTION <====
3866                                     ; WHICH FOLLOWS W/ 755 <====
3867 011660 012742 000275      MOV #275,-(R2)      ;MOVE TO MAILBOX # ***** 275 *****
3868 011664 005242          INC -(R2)           ;SET MSGTYP TO FATAL ERROR
3869 011666 000000          HALT                ;DEST. REGISTER MODIFIED
```

```

3870 011670 022737 000001 000000 DNM6C: CMP #1,@#0 ;CHECK DEST. DATA
3871 011676 001404 BEQ TST143
3872 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
3873 ; CONDITIONAL BRANCH INST. AND <=====
3874 ; REPLACE THE MOVE INSTRUCTION <=====
3875 ; WHICH FOLLOWS W/ 745 <=====
3876 011700 012742 000276 MOV #276,-(R2) ;MOVE TO MAILBOX # ***** 276 *****
3877 011704 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3878 011706 000000 HALT ;DEST. DATA MODIFIED
3879 ; OR SEQUENCE ERROR
3880

```

\*\*\*\*\*  
 :TEST 143 TEST DEST MODE 7 W/DOP NON-MODIFYING INST.  
 \*\*\*\*\*

```

3884 011710 005212 TST143: INC (R2) ;UPDATE TEST NUMBER
3885 011712 022712 000143 CMP #143,(R2) ;SEQUENCE ERROR?
3886 011716 001034 BNE TST144-10 ;BR TO ERROR HALT ON SEQ ERROR
3887 011720 005000 CLR R0 ;R0=0
3888 011722 005010 CLR (R0) ;LOC. 0=0 C-BIT=0
3889 011724 052710 125125 BIS #125125,(R0) ;LOC. 0=125125
3890 011730 052700 000001 BIS #1,R0 ;R0=1
3891 011734 132770 000125 000403 BITB #125,@403(R0) ;TRY DOPNM W/MODE 7
3892 011742 102403 BVS DNM7A ;BR TO ERROR IF V-BIT SET
3893 011744 100402 BMI DNM7A ;BR TO ERROR IF N-BIT SET
3894 011746 103401 BCS DNM7A ;BR TO ERROR IF C-BIT SET
3895 011750 001404 BEQ DNM7B

```

```

3896 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
3897 ; CONDITIONAL BRANCH INST. AND <=====
3898 ; REPLACE THE MOVE INSTRUCTION <=====
3899 ; WHICH FOLLOWS W/ 763 <=====

```

```

3900 011752 DNM7A:
3901 011752 012742 000277 MOV #277,-(R2) ;MOVE TO MAILBOX # ***** 277 *****
3902 011756 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3903 011760 000000 HALT ;COND. CODES INCORRECT
3904 011762 022700 000001 DNM7B: CMP #1,R0 ;CHECK DEST. REGISTER
3905 011766 001404 BEQ DNM7C

```

```

3906 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
3907 ; CONDITIONAL BRANCH INST. AND <=====
3908 ; REPLACE THE MOVE INSTRUCTION <=====
3909 ; WHICH FOLLOWS W/ 754 <=====

```

```

3910 011770 012742 000300 MOV #300,-(R2) ;MOVE TO MAILBOX # ***** 300 *****
3911 011774 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3912 011776 000000 HALT ;DESTINATION REGISTER MODIFIED
3913 012000 022737 125125 000000 DNM7C: CMP #125125,@#0 ;CHECK DEST. DATA
3914 012006 001404 BEQ TST144

```

```

3915 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
3916 ; CONDITIONAL BRANCH INST. AND <=====
3917 ; REPLACE THE MOVE INSTRUCTION <=====
3918 ; WHICH FOLLOWS W/ 744 <=====

```

```

3919 012010 012742 000301 MOV #301,-(R2) ;MOVE TO MAILBOX # ***** 301 *****
3920 012014 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3921 012016 000000 HALT ;DEST. DATA INCORRECT
3922 ; OR SEQUENCE ERROR

```

\*\*\*\*\*

3926  
3927  
3928  
3929  
3930  
3931  
3932  
3933 012020 005212  
3934 012022 022712 000144  
3935 012026 001016  
3936 012030 005000  
3937 012032 005010  
3938 012034 005100  
3939 012036 005004  
3940 012040 010014  
3941 012042 102402  
3942 012044 001401  
3943 012046 100404  
3944  
3945  
3946  
3947  
3948 012050  
3949 012050 012742 000302  
3950 012054 005242  
3951 012056 000000  
3952 012060 005704  
3953 012062 001404  
3954  
3955  
3956  
3957  
3958 012064 012742 000303  
3959 012070 005242  
3960 012072 000000  
3961  
3962  
3963  
3964  
3965  
3966  
3967  
3968  
3969  
3970  
3971  
3972 012074 005212  
3973 012076 022712 000145  
3974 012102 001025  
3975 012104 005000  
3976 012106 005010  
3977 012110 005110  
3978 012112 010020  
3979 012114 100402  
3980 012116 102401  
3981 012120 001404

: 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.

\*\*\*\*\*  
: TEST 144 TEST MOV DESTINATION MODE 1  
\*\*\*\*\*

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/ 770 <====

MDM1A: MOV #302,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 302 \*\*\*\*\*  
INC -(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/ 762 <====

MOV #303,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 303 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;DESTINATION REGISTER INCORRECTLY ALTERED  
; OR SEQUENCE ERROR

\*\*\*\*\*  
: 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, DEST. MODE 1.

\*\*\*\*\*  
: TEST 145 TEST MOV DESTINATION MODE 2  
\*\*\*\*\*

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= 1  
MOV R0,(R0)+ ;TRY MOVE MODE 0,2  
BMI MDM2A ;BR TO ERROR IF N SET  
BVS MDM2A ;BR TO ERROR IF V SET  
BEQ MDM2B



```
3982 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3983 ; CONDITIONAL BRANCH INST. AND <====
3984 ; REPLACE THE MOVE INSTRUCTION <====
3985 ; WHICH FOLLOWS W/ 771 <====
3986 012122 MDM2A:
3987 012122 012742 000304 MOV #304,-(R2) ;MOVE TO MAILBOX # ***** 304 *****
3988 012126 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
3989 012130 000000 HALT ;CC'S INCORRECT
3990 012132 005300 MDM2B: DEC R0
3991 012134 005300 DEC R0
3992 012136 001404 BEQ MDM2D
3993 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
3994 ; CONDITIONAL BRANCH INST. AND <====
3995 ; REPLACE THE MOVE INSTRUCTION <====
3996 ; WHICH FOLLOWS W/ 762 <====
3997 012140 MDM2C:
3998 012140 012742 000305 MOV #305,-(R2) ;MOVE TO MAILBOX # ***** 305 *****
3999 012144 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4000 012146 000000 HALT ;DESTINATION REGISTER NOT INCREMENTED PROPERLY
4001 012150 005737 000000 MDM2D: TST @#0
4002 012154 001404 BEQ TST146
4003 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4004 ; CONDITIONAL BRANCH INST. AND <====
4005 ; REPLACE THE MOVE INSTRUCTION <====
4006 ; WHICH FOLLOWS W/ 753 <====
4007 012156 012742 000306 MOV #306,-(R2) ;MOVE TO MAILBOX # ***** 306 *****
4008 012162 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4009 012164 000000 HALT ;DESTINATION DATA INCORRECT
4010 ; OR SEQUENCE ERROR
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
```

\*\*\*\*\*  
: 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.  
: \*\*\*\*\*  
: TEST 146 TEST MOV-BYTE DESTINATION MODE 2  
: \*\*\*\*\*  
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

```
4033 012214 MBDM2A:
4034 012214 012742 000307 MOV #307,-(R2) ;MOVE TO MAILBOX # ***** 307 *****
4035 012220 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4036 012222 000000 HALT ;CC'S INCORRECT
4037 012224 022700 000001 MBDM2B: CMP #1,R0
```

```

4038 012230 001404          BEQ      MBDM2C          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4039                                     ;          CONDITIONAL BRANCH INST. AND <====
4040                                     ;          REPLACE THE MOVE INSTRUCTION <====
4041                                     ;          WHICH FOLLOWS W/ 762 <====
4042                                     ;
4043 012232 012742 000310    MOV      #310,-(R2)    ;MOVE TO MAILBOX # ***** 310 *****
4044 012236 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4045 012240 000000          HALT     ;REGISTER NOT INCREMENTED BY ONE
4046 012242 112720 000252    MBDM2C: MOVB   #252,(R0)+ ;TRY DESTINATION MODE 2 W/ODD BYTE
4047 012246 102402          BVS     MBDM2D
4048 012250 001401          BEQ     MBDM2D
4049 012252 100404          BMI     MBDM2E
4050                                     ;
4051                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4052                                     ;          CONDITIONAL BRANCH INST. AND <====
4053                                     ;          REPLACE THE MOVE INSTRUCTION <====
4054                                     ;          WHICH FOLLOWS W/ 751 <====
4054 012254          MBDM2D:
4055 012254 012742 000311    MOV      #311,-(R2)    ;MOVE TO MAILBOX # ***** 311 *****
4056 012260 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4057 012262 000000          HALT     ;CC'S NOT SET CORRECT
4058 012264 022700 000002    MBDM2E: CMP     #2,R0
4059 012270 001404          BEQ     MBDM2F
4060                                     ;
4061                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4062                                     ;          CONDITIONAL BRANCH INST. AND <====
4063                                     ;          REPLACE THE MOVE INSTRUCTION <====
4064                                     ;          WHICH FOLLOWS W/ 742 <====
4064 012272 012742 000312    MOV      #312,-(R2)    ;MOVE TO MAILBOX # ***** 312 *****
4065 012276 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4066 012300 000000          HALT     ;REGISTER NOT INCREMENTED BY ONE
4067 012302 022737 125125 000000 MBDM2F: CMP     #125125,@#0
4068 012310 001404          BEQ     TST147
4069                                     ;
4070                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4071                                     ;          CONDITIONAL BRANCH INST. AND <====
4072                                     ;          REPLACE THE MOVE INSTRUCTION <====
4073                                     ;          WHICH FOLLOWS W/ 732 <====
4073 012312 012742 000313    MOV      #313,-(R2)    ;MOVE TO MAILBOX # ***** 313 *****
4074 012316 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4075 012320 000000          HALT     ;DESTINATION DATA INCORRECT
4076                                     ; OR SEQUENCE ERROR
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086 012322 005212          TST147: INC     (R2)        ;UPDATE TEST NUMBER
4087 012324 022712 000147    CMP     #147,(R2)     ;SEQUENCE ERROR?
4088 012330 001057          BNE     TST150-10     ;BR TO ERROR HALT ON SEQ ERROR
4089 012332 012700 000400    MOV     #400,R0       ;R0=400
4090 012336 005010          CLR     (R0)         ;LOC. 400 POINTS TO LOC. 0
4091 012340 005037 000000    CLR     @#0          ;LOC. 0=0
4092 012344 012730 125252    MOV     #125252,@(R0)+ ;TRY MOV DESTINATION MODE 2
4093 012350 102402          BVS     MDM3A        ;BR TO ERROR IF V SET
    
```

```

:*****
:
: 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. ADJR.. ALSO, MOV(B)
: INST. ARE USED W/ EVEN AND ODD BYTES TO CHECK MOV BYTES INST AND MODE 37 DESTINATIONS.
:*****
:TEST 147      TEST MOV(B) DESTINATION MODE 3
:*****
    
```

```

4094 012352 001401      BEQ   MDM3A      ;BR TO ERROR IF Z SET
4095 012354 100404      BMI   MDM3B
4096                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4097                                     ;                               CONDITIONAL BRANCH INST. AND <=====
4098                                     ;                               REPLACE THE MOVE INSTRUCTION <=====
4099                                     ;                               WHICH FOLLOWS W/ 766 <=====
4100 012356                                     MDM3A:
4101 012356 012742 000314      MOV   #314,-(R2)  ;MOVE TO MAILBOX # ***** 314 *****
4102 012362 005242      INC   -(R2)      ;SET MSGTYP TO FATAL ERROR
4103 012364 000000      HALT                                     ;CC'S INCORRECT
4104 012366 022700 000402      MDM3B:  CMP   #402,R0  ;CHECK DEST. MODE REGISTER
4105 012372 001404      BEQ   MDM3C
4106                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4107                                     ;                               CONDITIONAL BRANCH INST. AND <=====
4108                                     ;                               REPLACE THE MOVE INSTRUCTION <=====
4109                                     ;                               WHICH FOLLOWS W/ 757 <=====
4110 012374 012742 000315      MOV   #315,-(R2)  ;MOVE TO MAILBOX # ***** 315 *****
4111 012400 005242      INC   -(R2)      ;SET MSGTYP TO FATAL ERROR
4112 012402 000000      HALT                                     ;REGISTER NOT INCREMENTED BY 2
4113 012404 022737 125252 000000  MDM3C:  CMP   #125252,@#0  ;CHECK DESTINATION DATA
4114 012412 001404      BEQ   MDM3D
4115                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4116                                     ;                               CONDITIONAL BRANCH INST. AND <=====
4117                                     ;                               REPLACE THE MOVE INSTRUCTION <=====
4118                                     ;                               WHICH FOLLOWS W/ 747 <=====
4119 012414 012742 000316      MOV   #316,-(R2)  ;MOVE TO MAILBOX # ***** 316 *****
4120 012420 005242      INC   -(R2)      ;SET MSGTYP TO FATAL ERROR
4121 012422 000000      HALT                                     ;DESTINATION DATA INCORRECT
4122 012424 112737 000125 000000  MDM3D:  MOVB  #125,@#0    ;TRY MOVB DESTINATION MODE Z EVEN BYTE
4123 012432 022737 125125 000000      CMP   #125125,@#0 ;CHECK DATA
4124 012440 001404      BEQ   MDM3E
4125                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4126                                     ;                               CONDITIONAL BRANCH INST. AND <=====
4127                                     ;                               REPLACE THE MOVE INSTRUCTION <=====
4128                                     ;                               WHICH FOLLOWS W/ 734 <=====
4129 012442 012742 000317      MOV   #317,-(R2)  ;MOVE TO MAILBOX # ***** 317 *****
4130 012446 005242      INC   -(R2)      ;SET MSGTYP TO FATAL ERROR
4131 012450 000000      HALT                                     ;DESTINATION DATA INCORRECT
4132 012452 112737 000525 000001  MDM3E:  MOVB  #525,@#1    ;TRY MOVB DESTINATION MODE 2 ODD BYTE
4133 012460 022737 052525 000000      CMP   #52525,@#0 ;CHECK DATA
4134 012466 001404      BEQ   TST150
4135                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4136                                     ;                               CONDITIONAL BRANCH INST. AND <=====
4137                                     ;                               REPLACE THE MOVE INSTRUCTION <=====
4138                                     ;                               WHICH FOLLOWS W/ 721 <=====
4139 012470 012742 000320      MOV   #320,-(R2)  ;MOVE TO MAILBOX # ***** 320 *****
4140 012474 005242      INC   -(R2)      ;SET MSGTYP TO FATAL ERROR
4141 012476 000000      HALT

```

```

:*****
:
: 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.
:

```

4150  
4151  
4152  
4153 012500 005212  
4154 012502 022712 000150  
4155 012506 001026  
4156 012510 005000  
4157 012512 005010  
4158 012514 012704 000002  
4159 012520 012744 012345  
4160 012524 102402  
4161 012526 001401  
4162 012530 100004  
4163  
4164  
4165  
4166  
4167 012532  
4168 012532 012742 000321  
4169 012536 005242  
4170 012540 000000  
4171 012542 005704  
4172 012544 001404  
4173  
4174  
4175  
4176  
4177 012546 012742 000322  
4178 012552 005242  
4179 012554 000000  
4180 012556 022710 012345  
4181 012562 001404  
4182  
4183  
4184  
4185  
4186 012564 012742 000323  
4187 012570 005242  
4188 012572 000000  
4189  
4190  
4191  
4192  
4193  
4194  
4195  
4196  
4197  
4198  
4199  
4200  
4201  
4202 012574 005212  
4203 012576 022712 000151  
4204 012602 001046  
4205 012604 005004

```
*****  
:TEST 150      TEST MOV DESTINATION MODE 4  
*****  
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/ 767 <=====  
  
MDM4A:  MOV     #321,-(R2)     ;MOVE TO MAILBOX # ***** 321 *****  
        INC     -(R2)         ;SET MSGTYP TO FATAL ERROR  
        HALT  
MDM4B:  TST     R4             ;CC'S NOT CORRECT  
        BEQ     MDM4C         ;CHECK DECREMENTING OF MODE 4 REG.  
  
        ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
        ;          CONDITIONAL BRANCH INST. AND <=====  
        ;          REPLACE THE MOVE INSTRUCTION <=====  
        ;          WHICH FOLLOWS W/ 761 <=====  
  
        MOV     #322,-(R2)     ;MOVE TO MAILBOX # ***** 322 *****  
        INC     -(R2)         ;SET MSGTYP TO FATAL ERROR  
        HALT  
MDM4C:  CMP     #12345,(R0)    ;DESTINATION MODE REGISTER NOT DECREMENTED BY 2  
        BEQ     TST151        ;CHECK DESTINATION DATA  
  
        ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
        ;          CONDITIONAL BRANCH INST. AND <=====  
        ;          REPLACE THE MOVE INSTRUCTION <=====  
        ;          WHICH FOLLOWS W/ 752 <=====  
  
        MOV     #323,-(R2)     ;MOVE TO MAILBOX # ***** 323 *****  
        INC     -(R2)         ;SET MSGTYP TO FATAL ERROR  
        HALT  
        ;DESTINATION DATA INCORRECT  
        ; OR SEQUENCE ERROR
```

```
*****  
: 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.  
*****
```

```
*****  
:TEST 151      TEST MOV(B) DESTINATION MODE 4  
*****  
TST151: INC      (R2)          ;UPDATE TEST NUMBER  
        CMP     #151,(R2)     ;SEQUENCE ERROR?  
        BNE     TST152-10      ;BR TO ERROR HALT ON SEQ ERROR  
        CLR     R4             ;R4=0
```

```

4206 012606 005014          CLR      (R4)          ;LOC. 0=0
4207 012610 012700 000002  MOV      #2,R0        ;R0 = 2
4208 012614 112740 125125  MOV B    #125125,-(R0) ;TRY MOV B DEST. MODE 4-ODD BYTE
4209 012620 020027 000001  CMP      R0,#1        ;CHECK THAT DEST. REG. WAS DECREMENTED
4210 012624 001404          BEQ      MBDM4A
4211          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4212          ;          CONDITIONAL BRANCH INST. AND <=====
4213          ;          REPLACE THE MOVE INSTRUCTION <=====
4214          ;          WHICH FOLLOWS W/ 767 <=====
4215 012626 012742 000324          MOV      #324,-(R2)   ;MOVE TO MAILBOX # ***** 324 *****
4216 012632 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4217 012634 000000          HALT
4218 012636 021427 052400  MBDM4A: CMP      (R4),#52400 ;DESTINATION REG. NOT DECREMENTED BY 1
4219 012642 001404          BEQ      MBDM4B      ;CHECK DEST. DATA
4220          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4221          ;          CONDITIONAL BRANCH INST. AND <=====
4222          ;          REPLACE THE MOVE INSTRUCTION <=====
4223          ;          WHICH FOLLOWS W/ 760 <=====
4224 012644 012742 000325          MOV      #325,-(R2)   ;MOVE TO MAILBOX # ***** 325 *****
4225 012650 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4226 012652 000000          HALT
4227 012654 112740 125125  MBDM4B: MOV B    #125125,-(R0) ;DEST. DATA NOT CORRECT
4228 012660 102402          BVS     MBDM4C        ;TRY MOV B DEST. MODE 4--EVEN BYTE
4229 012662 001401          BEQ     MBDM4C        ;BR. TO ERROR IF V-BIT SET
4230 012664 100004          BPL     MBDM4D        ;BR TO ERROR IF Z-BIT SET
4231          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4232          ;          CONDITIONAL BRANCH INST. AND <=====
4233          ;          REPLACE THE MOVE INSTRUCTION <=====
4234          ;          WHICH FOLLOWS W/ 747 <=====
4235 012666          MBDM4C:
4236 012666 012742 000326          MOV      #326,-(R2)   ;MOVE TO MAILBOX # ***** 326 *****
4237 012672 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4238 012674 000000          HALT
4239 012676 005700          MBDM4D: TST     R0     ;COND. CODES INCORRECT
4240 012700 001404          BEQ     MBDM4E      ;CHECK MODE 4 DEST. REGISTER
4241          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4242          ;          CONDITIONAL BRANCH INST. AND <=====
4243          ;          REPLACE THE MOVE INSTRUCTION <=====
4244          ;          WHICH FOLLOWS W/ 741 <=====
4245 012702 012742 000327          MOV      #327,-(R2)   ;MOVE TO MAILBOX # ***** 327 *****
4246 012706 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4247 012710 000000          HALT
4248 012712 021427 052525  MBDM4E: CMP      (R4),#52525 ;DESTINATION REG NOT DECREMENTED BY 1
4249 012714 001404          BEQ     TST152      ;CHECK DEST. DATA
4250          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4251          ;          CONDITIONAL BRANCH INST. AND <=====
4252          ;          REPLACE THE MOVE INSTRUCTION <=====
4253          ;          WHICH FOLLOWS W/ 732 <=====
4254 012720 012742 000330          MOV      #330,-(R2)   ;MOVE TO MAILBOX # ***** 330 *****
4255 012724 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
4256 012726 000000          HALT
4257          ; DESTINATION DATA INCORRECT
4258          ; OR SEQUENCE ERROR
4259
4260
4261

```

\*\*\*\*\*  
 : THIS TEST VERIFIES THE MOV DESTINATION MODE 5 AND THE MOV B  
 :

```
4262 ;DESTINATION MODE 5 - EVEN BYTE INSTRUCTIONS. R4 IS A
4263 ;POINTER TO TARGET LOCATION 0 AND R0 IS SETUP TO
4264 ;POINT TO LOCATION 376 FOR THE MOV, AND LOCATION 404 FOR
4265 ;THE MOV B INSTRUCTIONS. CMP INSTRUCTIONS ARE USED TO VERIFY
4266 ;PROPER ADDRESSING AND DATA.
4267
4268
4269 ;*****
4270 ;TEST 152 TEST MOV DESTINATION MODE 5
4271 ;*****
4271 012730 005212 TST152: INC (R2) ;UPDATE TEST NUMBER
4272 012732 022712 000152 CMP #152,(R2) ;SEQUENCE ERROR?
4273 012736 001051 BNE TST153-10 ;BR TO ERROR HALT ON SEQ ERROR
4274 012740 005004 CLR R4 ;R4=0
4275 012742 005014 CLR (R4) ;LOC. 0 = 0
4276 012744 012700 000400 MOV #400,R0 ;R0=400
4277 012750 012750 004321 MOV #4321,@-(R0) ;TRY MOV DEST. MODE 5
4278 012754 102402 BVS MDM5A ;BR TO ERROR IF V-BIT SET
4279 012756 001401 BEQ MDM5A ;BR TO ERROR IF Z-BIT SET
4280 012760 100004 BPL MDM5B
4281
4282 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4283 ; CONDITIONAL BRANCH INST. AND <=====
4284 ; REPLACE THE MOVE INSTRUCTION <=====
4285 ; WHICH FOLLOWS W/ 767 <=====
4285 012762 MDM5A: MOV #331,-(R2) ;MOVE TO MAILBOX # ***** 331 *****
4286 012762 012742 000331 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4287 012766 005242 HALT ;COND. CODES INCORRECT
4288 012770 000000 MDM5B: CMP #376,R0 ;CHECK MODE 5 REG. WAS DECREMENTED
4289 012772 022700 000376 BEQ MDM5C
4290 012776 001404
4291
4292 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4293 ; CONDITIONAL BRANCH INST. AND <=====
4294 ; REPLACE THE MOVE INSTRUCTION <=====
4295 ; WHICH FOLLOWS W/ 760 <=====
4295 013000 012742 000332 MOV #332,-(R2) ;MOVE TO MAILBOX # ***** 332 *****
4296 013004 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4297 013006 000000 HALT ;MODE 5 REGISTER NOT DECREMENTED BY 2
4298 013010 022714 004321 MDM5C: CMP #4321,(R4) ;CHECK DEST. DATA
4299 013014 001404 BEQ MDM5D
4300
4301 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4302 ; CONDITIONAL BRANCH INST. AND <=====
4303 ; REPLACE THE MOVE INSTRUCTION <=====
4304 ; WHICH FOLLOWS W/ 751 <=====
4304 013016 012742 000333 MOV #333,-(R2) ;MOVE TO MAILBOX # ***** 333 *****
4305 013022 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4306 013024 000000 HALT ;DEST. DATA INCORRECT
4307 013026 012700 000406 MDM5D: MOV #406,R0 ;RC=406
4308 013032 112750 000377 MOV B #377,@-(R0) ;TRY MOV DEST. MODE 5 --EVEN BYTE
4309 013036 022700 000404 CMP #404,R0 ;CHECK MODE 5 REG.
4310 013042 001404 BEQ MDM5E
4311
4312 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4313 ; CONDITIONAL BRANCH INST. AND <=====
4314 ; REPLACE THE MOVE INSTRUCTION <=====
4315 ; WHICH FOLLOWS W/ 736 <=====
4315 013044 012742 000334 MOV #334,-(R2) ;MOVE TO MAILBOX # ***** 334 *****
4316 013050 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4317 013052 000000 HALT ;MODE 5 REGISTER NOT DECREMENTED BY 2
```

```
4318 013054 022714 177721 MDM5E: CMP #177721,(R4) ;CHECK DEST. DATA
4319 013060 001404 BEQ TST153 ;
4320 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
4321 ; CONDITIONAL BRANCH INST. AND <=====
4322 ; REPLACE THE MOVE INSTRUCTION <=====
4323 ; WHICH FOLLOWS W/ 727 <=====
4324 013062 012742 000335 MOV #335,-(R2) ;MOVE TO MAILBOX # ***** 335 *****
4325 013066 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4326 013070 000000 HALT ;DEST. DATA INCORRECT
4327 ; OR SEQUENCE ERROR
4328
4329
4330
```

```
*****
: THIS TEST VERIFIES THE MOV DESTINATION MODE 6 AND MOV B - 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.
*****
```

```
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340 013072 005212
4341 013074 022712 000153
4342 013100 001054
4343 013102 005000
4344 013104 005010
4345 013106 005200
4346 013110 012760 052525 177777
4347 013116 102402
4348 013120 001401
4349 013122 100004
4350
4351
4352
4353
4354 013124
4355 013124 012742 000336
4356 013130 005242
4357 013132 000000
4358 013134 022700 000001
4359 013140 001404
4360
4361
4362
4363
4364 013142 012742 000337
4365 013146 005242
4366 013150 000000
4367 013152 022737 052525 000000 MDM6C: CMP #52525,@#0
4368 013160 001404 BEQ MDM6D
4369
4370
4371
4372
4373 013162 012742 000340 MOV #340,-(R2) ;MOVE TO MAILBOX # ***** 340 *****
```

```
*****
: TEST 153 TEST MOV DESTINATION MODE 6
*****
```

```
TST153: INC (R2) ;UPDATE TEST NUMBER
CMP #153,(R2) ;SEQUENCE ERROR?
BNE TST154-10 ;BR TO ERROR HALT ON SEQ ERROR
CLR R0 ;R0=0
CLR (R0) ;LOC. 0=0
INC R0 ;R0=1
MOV #052525,-1(R0) ;TRY MOV DEST. MODE 6
BVS MDM6A ;BR TO ERROR IF V-BIT SET
BEQ MDM6A ;BR TO ERROR IF Z-BIT SET
BPL MDM6B
```

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

```
MDM6A: MOV #336,-(R2) ;MOVE TO MAILBOX # ***** 336 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;COND. CODES INCORRECT
MDM6B: CMP #1,R0 ;CHECK DEST. REGISTER UNALTERED
BEQ MDM6C
```

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

```
MDM6C: MOV #337,-(R2) ;MOVE TO MAILBOX # ***** 337 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;DEST. REGISTER INCORRECTLY ALTERED
MDM6C: CMP #52525,@#0 ;CHECK DEST. DATA
BEQ MDM6D
```

```
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
: CONDITIONAL BRANCH INST. AND <=====
: REPLACE THE MOVE INSTRUCTION <=====
: WHICH FOLLOWS W/ 750 <=====
```

```
4374 013166 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
4375 013170 000000          HALT                    ;DEST. DATA INCORRECT
4376 013172 012700 000002          MOV      #2,R0          ;R0=2
4377 013176 112760 000377 177777 MDM6D: MOVB     #377,-1(R0)    ;TRY MOVB DEST. MODE 6
4378 013204 022700 000002          CMP      #2,R0          ;CHECK DEST. REGISTER UNALTERED
4379 013210 001404          BEQ      MDM6E          ;
4380                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4381                                     ;          CONDITIONAL BRANCH INST. AND <====
4382                                     ;          REPLACE THE MOVE INSTRUCTION <====
4383                                     ;          WHICH FOLLOWS W/ 734 <====
4384 013212 012742 000341          MOV      #341,-(R2)    ;MOVE TO MAILBOX # ***** 341 *****
4385 013216 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
4386 013220 000000          HALT                    ;DEST. REGISTER INCORRECTLY ALTERED
4387 013222 022737 177525 000000 MDM6E: CMP      #177525,@#0 ;CHECK DEST. DATA
4388 013230 001404          BEQ      TST154        ;
4389                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4390                                     ;          CONDITIONAL BRANCH INST. AND <====
4391                                     ;          REPLACE THE MOVE INSTRUCTION <====
4392                                     ;          WHICH FOLLOWS W/ 724 <====
4393 013232 012742 000342          MOV      #342,-(R2)    ;MOVE TO MAILBOX # ***** 342 *****
4394 013236 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
4395 013240 000000          HALT                    ;DEST. DATA INCORRECT
4396                                     ; OR SEQUENCE ERROR
4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408 013242 005212          TST154: INC      (R2)          ;UPDATE TEST NUMBER
4409 013244 022712 000154          CMP      #154,(R2)     ;SEQUENCE ERROR?
4410 013250 001053          BNE     TST155-10      ;BR TO ERROR HALT ON SEQ ERROR
4411 013252 005004          CLR     R4             ;R4=0
4412 013254 005014          CLR     (R4)           ;LOC.0=0
4413 013256 012700 000403          MOV      #403,R0       ;R0=403
4414 013262 012770 070707 177777 MDM7A: MOV      #70707,@-1(R0) ;TRY MOV W/DEST MODE 7
4415 013270 102402          BVS     MDM7A          ;BR. TO ERROR IF V-BIT SET
4416 013272 001401          BEQ     MDM7A          ;BR TO ERROR IF Z-BIT SET
4417 013274 100004          BPL     MDM7B          ;
4418                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4419                                     ;          CONDITIONAL BRANCH INST. AND <====
4420                                     ;          REPLACE THE MOVE INSTRUCTION <====
4421                                     ;          WHICH FOLLOWS W/ 766 <====
4422 013276          MDM7A:
4423 013276 012742 000343          MOV      #343,-(R2)    ;MOVE TO MAILBOX # ***** 343 *****
4424 013302 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
4425 013304 000000          HALT                    ;COND. CODES INCORRECT
4426 013306 022700 000403 MDM7B: CMP      #403,R0    ;CHECK DEST. REGISTER
4427 013312 001404          BEQ     MDM7C          ;
4428                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4429                                     ;          CONDITIONAL BRANCH INST. AND <====
```



```
4430  
4431  
4432 013314 012742 000344      MOV    #344,-(R2)      ; REPLACE THE MOVE INSTRUCTION <=====  
4433 013320 005242      INC    -(R2)          ; WHICH FOLLOWS W/ 757 <=====  
4434 013322 000000      HALT                                ; MOVE TO MAILBOX # ***** 344 *****  
4435 013324 022737 070707 000000 MDM7C: CMP    #70707,@#0      ; SET MSGTYP TO FATAL ERROR  
4436 013332 001404      BEQ    MDM7D          ; DEST. REGISTER INCORRECTLY ALTERED  
4437  
4438  
4439  
4440  
4441 013334 012742 000345      MOV    #345,-(R2)      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
4442 013340 005242      INC    -(R2)          ; CONDITIONAL BRANCH INST. AND <=====  
4443 013342 000000      HALT                                ; REPLACE THE MOVE INSTRUCTION <=====  
4444 013344 112770 107070 000001 MDM7D: MOVB  #107070,@1(R0)    ; WHICH FOLLOWS W/ 747 <=====  
4445 013352 022700 000403      CMP    #403,R0        ; MOVE TO MAILBOX # ***** 345 *****  
4446 013356 001404      BEQ    MDM7E          ; SET MSGTYP TO FATAL ERROR  
4447  
4448  
4449  
4450  
4451 013360 012742 000346      MOV    #346,-(R2)      ; DEST. DATA INCORRECT  
4452 013364 005242      INC    -(R2)          ; TRY MOVW W/DEST MODE 7--ODD BYTE  
4453 013366 000000      HALT                                ; CHECK MODE 7 DEST. REG.  
4454 013370 022737 034307 000000 MDM7E: CMP    #34307,@#0      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
4455 013376 001404      BEQ    TST155         ; CONDITIONAL BRANCH INST. AND <=====  
4456  
4457  
4458  
4459  
4460 013400 012742 000347      MOV    #347,-(R2)      ; REPLACE THE MOVE INSTRUCTION <=====  
4461 013404 005242      INC    -(R2)          ; WHICH FOLLOWS W/ 725 <=====  
4462 013406 000000      HALT                                ; MOVE TO MAILBOX # ***** 347 *****  
4463  
4464  
4465  
4466  
4467  
4468  
4469  
4470  
4471  
4472  
4473  
4474  
4475  
4476  
4477  
4478  
4479
```

```
*****  
: 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.  
: *****  
: TEST 155 TEST MODE 4 W/ DOP INSTS.  
: *****
```

```
4480 013410 005212      TST155: INC    (R2)          ; UPDATE TEST NUMBER  
4481 013412 022712 000155      CMP    #155,(R2)      ; SEQUENCE ERROR?  
4482 013416 001015      BNE    DOP4           ; BR TO ERROR HALT ON SEQ ERROR  
4483 013420 012700 013472      MOV    #TBL1,R0       ; INITIALIZE R0  
4484 013424 014037 013472      MOV    -(R0),@#TBL1   ; TBL1=125252  
4485 013430 064037 013472      ADD    -(R0),@#TBL1   ; TBL1=000377
```

4486 013434 144037 013472  
4487 013440 154037 013473  
4488 013444 024037 013472  
4489 013450 001411

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/ 763 <====

4494 013452  
4495 013452 012742 000350  
4496 013456 005242  
4497 013460 000000  
4498  
4499

DOP4: MOV #350,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 350 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;RESULT OF MODE 4 INSTS. INCORRECT  
; OR SEQUENCE ERROR

4500 013462 125252  
4501 013464 052652  
4502 013466 053125  
4503 013470 125252  
4504 013472 000000  
4505

125252  
52652  
53125  
125252  
TBL1: 0

4506  
4507  
4508  
4509  
4510  
4511  
4512  
4513  
4514  
4515

\*\*\*\*\*  
: 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).  
\*\*\*\*\*

4516  
4517

TEST 156 TEST MODE 5 W/ DOP INSTS.

4518 013474 005212  
4519 013476 022712 000156  
4520 013502 001015  
4521 013504 012700 013560  
4522 013510 015037 013472  
4523 013514 065037 013472  
4524 013520 145037 013472  
4525 013524 155037 013473  
4526 013530 025037 013472  
4527 013534 001411

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/ 763 <====

4532 013536  
4533 013536 012742 000351  
4534 013542 005242  
4535 013544 000000  
4536

DOP5: MOV #351,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 351 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;RESULT OF MODE 5 INSTS. INCORRECT  
; OR SEQUENCE ERROR

4537 013546 013462  
4538 013550 013464  
4539 013552 013465  
4540 013554 013466  
4541 013556 013470

TBL1-10  
TBL1-6  
TBL1-5  
TBL1-4  
TBL2: TBL1-2

4542  
4543  
4544  
4545  
4546  
4547  
4548  
4549  
4550  
4551  
4552  
4553  
4554  
4555 013560 005212  
4556 013562 022712 000157  
4557 013566 001022  
4558 013570 012700 013466  
4559 013574 016037 000002 013472  
4560 013602 066037 000000 013472  
4561 013610 146037 177777 013472  
4562 013616 156037 177776 013473  
4563 013624 026037 177774 013472  
4564 013632 001404  
4565  
4566  
4567  
4568  
4569 013634 012742 000352  
4570 013640 005242  
4571 013642 000000  
4572  
4573  
4574  
4575  
4576  
4577  
4578  
4579  
4580  
4581  
4582  
4583  
4584  
4585  
4586 013644 0 212  
4587 013646 0 712 000160  
4588 013652 0 022  
4589 013654 0 2700 013552  
4590 013660 017037 000004 013472  
4591 013666 067037 000002 013472  
4592 013674 147037 000000 013472  
4593 013702 157037 177776 013473  
4594 013710 027037 177774 013472  
4595 013716 001404  
4596  
4597

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 157 TEST MODE 6 W/ DOP INSTS.  
\*\*\*\*\*

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/ 756 <====  
MOV #352,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 352 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;RESULT OF MODE 6 INSTS. INCORRECT  
: OR SEQUENCE ERROR

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 160 TEST MODE 7 W/ DOP INSTS.  
\*\*\*\*\*

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 <====

```
4598  
4599  
4600 013720 012742 000353      MOV    #353,-(R2)      ;REPLACE THE MOVE INSTRUCTION <====  
4601 013724 005242              INC    -(R2)           ;WHICH FOLLOWS W/ 756 <====  
4602 013726 000000              HALT                   ;MOVE TO MAILBOX # ***** 353 *****  
4603                                ;SET MSGTYP TO FATAL ERROR  
4604                                ;RESULT OF MODE 7 INSTS INCORRECT  
4605                                ;OR SEQUENCE ERROR
```

```
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
```

```
*****  
: THIS TEST VERIFIES THE ROTATE MODE 0 INSTRUCTIONS.  
: R0 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.  
*****
```

```
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
```

```
*****  
: TEST 161 TEST ROTATE INSTRUCTIONS OF MODE 0  
*****
```

```
4615 013730 005212  
4616 013732 022712 000161  
4617 013736 001026  
4618 013740 012700 125252  
4619 013744 000261  
4620 013746 006100  
4621 013750 102004  
4622 013752 103003  
4623 013754 022700 052525  
4624 013760 001404  
4625  
4626  
4627  
4628  
4629  
4630  
4631  
4632  
4633  
4634  
4635  
4636  
4637  
4638  
4639  
4640  
4641  
4642  
4643  
4644  
4645  
4646  
4647  
4648
```

```
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   ROT0A          ;CC=0011  
        BCC   ROT0A  
        CMP   #052525,R0      ;CHECK DATA  
        BEQ   ROT0B  
  
        ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
        ; CONDITIONAL BRANCH INST. AND <====  
        ; REPLACE THE MOVE INSTRUCTION <====  
        ; WHICH FOLLOWS W/ 767 <====
```

```
4629 013762  
4630 013762 012742 000354  
4631 013766 005242  
4632 013770 000000  
4633 013772 012700 125252  
4634 013776 000261  
4635 014000 106100  
4636 014002 102004  
4637 014004 103003  
4638 014006 022700 125125  
4639 014012 001404  
4640  
4641  
4642  
4643  
4644  
4645  
4646  
4647  
4648
```

```
ROT0A: MOV    #354,-(R2)      ;MOVE TO MAILBOX # ***** 354 *****  
        INC    -(R2)         ;SET MSGTYP TO FATAL ERROR  
        HALT                   ;ROL MODE 0 FAILED  
ROT0B: MOV   #125252,R0      ;INITIALIZE DATA  
        SEC   ;SET C-BIT  
        ROLB  R0             ;TRY ROL W/ MODE 0 EVEN BYTE  
        BVC   ROT0C          ;CC=0011  
        BCC   ROT0C  
        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/ 752 <====
```

```
4644 014014  
4645 014014 012742 000355  
4646 014020 005242  
4647 014022 000000  
4648
```

```
ROT0C: MOV    #355,-(R2)      ;MOVE TO MAILBOX # ***** 355 *****  
        INC    -(R2)         ;SET MSGTYP TO FATAL ERROR  
        HALT                   ;ROLB MODE 0 FAILED  
        ; OR SEQUENCE ERROR
```

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

014024 005212  
014026 022712 000162  
014032 001051  
014034 005000  
014036 012710 052525  
014042 000241  
014044 006110  
014046 102005  
014050 103404  
014052 023727 000000 125252  
014060 001404  
  
014062  
014062 012742 000356  
014066 005242  
014070 000000  
014072 000261  
014074 012710 125252  
014100 106110  
014102 102005  
014104 103004  
014106 022737 125125 000000  
014114 001404  
  
014116  
014116 012742 000357  
014122 005242  
014124 000000  
014126 012710 125252  
014132 005000  
014134 005200  
014136 000261  
014140 106110  
014142 102005  
014144 103004  
014146 022737 052652 000000  
014154 001404

```

:*****
:
:   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.
:*****
:TEST 162      TEST ROTATE INSTRUCTIONS W/ MODE 1
:*****
TST162: INC      (R2)          ;UPDATE TEST NUMBER
        CMP      #162,(R2)    ;SEQUENCE ERROR?
        BNE     TST163-10    ;BR TO ERROR HALT ON SEQ ERROR
        CLR     R0           ;POINT TO LOC. 0
        MOV     #52525,(R0)   ;INITIALIZE DATA
        CLC     ;CLEAR C-BIT
        ROL     (R0)         ;TRY ROL W/ MODE 1
        BVC     ROT1A        ;CC=1010
        BCS     ROT1A
        CMP     @#0,#125252   ;CHECK RESULT
        BEQ     ROT1B

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

ROT1A:  MOV     #356,-(R2)    ;MOVE TO MAILBOX # ***** 356 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT    ;ROL MODE 1 FAILED

ROT1B:  SEC
        MOV     #125252,(R0) ;INITIALIZE DATA
        ROLB   (R0)         ;TRY ROLB W/ MODE 1 EVEN BYTE
        BVC     ROT1C        ;CC=1011
        BCC     ROT1C
        CMP     #125125,@#0  ;TEST RESULT
        BEQ     ROT1D

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

ROT1C:  MOV     #357,-(R2)    ;MOVE TO MAILBOX # ***** 357 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT    ;ROLB W/ MODE 1 EVEN BYTE FAILED

ROT1D:  MOV     #125252,(R0) ;INITIALIZE DATA
        CLR     R0           ;POINT TO ODD BYTE
        INC     R0
        SEC     ;SET C-BIT
        ROLB   (R0)         ;TRY ROLB W/ MODE 1 ODD BYTE
        BVC     ROT1E        ;CC=0011
        BCC     ROT1E
        CMP     #052652,@#0  ;CHECK DATA
        BEQ     TST163
    
```

```

4705 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4706 ; CONDITIONAL BRANCH INST. AND <====
4707 ; REPLACE THE MOVE INSTRUCTION <====
4708 ; WHICH FOLLOWS W/ 727 <====

```

```

4709 014156 ROT1E: MOV #360,-(R2) ;MOVE TO MAILBOX # ***** 360 *****
4710 014156 012742 000360 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4711 014162 005242 HALT ;ROLB W/ MODE 1 ODD BYTE FAILED
4712 014164 000000 ; OR SEQUENCE ERROR

```

```

:*****
: THIS TEST VERIFIES MODE 2 ROTATE INSTRUCTIONS.
: THE SAME PROCEDURE AS IN THE OTHER ROTATE TESTS ARE USED. R0
: IS USED AS THE ADDRESSING REGISTER AND IS CHECKED FOR PROPER
: INCREMENTING. BYTE INSTRUCTIONS ARE ALSO CHECKED.
:*****

```

```

4723 :TEST 163 TEST ROTATE INSTRUCTIONS W/ MODE 2
4724 :*****

```

```

4725 014166 005212 TST163: INC (R2) ;UPDATE TEST NUMBER
4726 014170 022712 000163 CMP #163,(R2) ;SEQUENCE ERROR?
4727 014174 001057 BNE TST164-10 ;BR TO ERROR HALT ON SEQ ERROR
4728 014176 005000 CLR R0 ;POINT TO LOC 0
4729 014200 012710 173737 MOV #173737,(R0) ;INITIALIZE DATA
4730 014204 000241 CLC ;CLEAR C-BIT
4731 014206 006120 ROL (R0)+ ;TRY ROL W/ MODE 2
4732 014210 103007 BCC ROT2A ;CHECK C-BIT
4733 014212 022737 167676 000000 CMP #167676,@#0 ;CHECK DATA
4734 014220 001003 BNE ROT2A ;BRANCH IF RESULT INCORRECT
4735 014222 005300 DEC R0 ;TEST R0
4736 014224 005300 DEC R0
4737 014226 001404 BEQ ROT2B

```

```

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

```

```

4742 014230 ROT2A: MOV #361,-(R2) ;MOVE TO MAILBOX # ***** 361 *****
4743 014230 012742 000361 INC -(R2) ;SET MSGTYP TO FATAL ERROR
4744 014234 005242 HALT ;ROL W/ MODE 2 FAILED

```

```

4746 014240 005000 ROT2B: CLR R0 ;POINT TO LOC 0
4747 014242 012710 004040 MOV #4040,(R0) ;INITIALIZE DATA
4748 014246 000241 CLC ;CLEAR C-BIT
4749 014250 106120 ROLB (R0)+ ;TRY ROLB W/ MODE 2 EVEN BYTE
4750 014252 103406 BCS ROT2C ;CHECK C-BIT
4751 014254 022737 004100 000000 CMP #4100,@#0 ;CHECK DATA
4752 014262 001002 BNE ROT2C ;BRANCH IF DATA INCORRECT
4753 014264 005300 DEC R0 ;CHECK R0
4754 014266 001404 BEQ ROT2D

```

```

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 743 <====

```

```

4759 014270 ROT2C: MOV #362,-(R2) ;MOVE TO MAILBOX # ***** 362 *****
4760 014270 012742 000362 INC -(R2)

```

```

4761 014274 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
4762 014276 000000          HALT                    ;ROLB W/ MODE 2 EVEN BYTE FAILED
4763 014300 005000          ROT2D: CLR      R0          ;POINT TO LOC 0
4764 014302 012710 004040  MOV      #4040,(R0)    ;INITIALIZE DATA
4765 014306 005200          INC      R0          ;POINT TO ODD BYTE OF DATA
4766 014310 000261          SEC                    ;SET C-BIT
4767 014312 106120          ROLB     (R0)+        ;TRY ROL W/ MODE 2 ODD BYTE
4768 014314 103407          BCS     ROT2E        ;CHECK C-BIT
4769 014316 022737 010440 000000 CMP      #10440,@#0   ;CHECK DATA
4770 014324 001003          BNE     ROT2E        ;BRANCH IF DATA INCORRECT
4771 014326 005300          DEC      R0          ;CHECK R0
4772 014330 005300          DEC      R0
4773 014332 001404          BEQ     TST164
4774
4775
4776
4777
4778 014334          ROT2E:
4779 014334 012742 000363  MOV      #363,-(R2)   ;MOVE TO MAILBOX # ***** 363 *****
4780 014340 005242          INC      -(R2)
4781 014342 000000          HALT                    ;SET MSGTYP TO FATAL ERROR
4782
                                     ;ROLB W/ MODE 2 ODD BYTE FAILED
                                     ; OR SEQUENCE ERROR

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

```

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

014344 005212  
014346 022712 000164  
014352 001051  
014354 012737 052525 000000  
014362 000261  
014364 006137 000000  
014370 103404  
014372 022737 125253 000000  
014400 001404  
  
014402  
014402 012742 000364  
014406 005242  
014410 000000  
014412 012737 125252 000000  
014420 000241  
014422 106137 000000  
014426 103004  
014430 023727 000000 125124 4\$:  
014436 001404  
  
014440  
014440 012742 000365  
014444 005242  
014446 000000  
014450 012737 125252 000000  
014456 000261  
014460 106137 000001  
014464 103004  
014466 022737 052652 000000  
014474 001404  
  
014476  
014476 012742 000366  
014502 005242  
014504 000000

```
*****
: 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 ADDRESSED USING
: MODE 37. BYTE ADDRESSING IS ALSO CHECKED FOR EVEN AND ODD BYTES.
*****
: TEST 164 TEST ROTATE INSTRUCTIONS /W MODE 3
*****
TST164: INC (R2) ;UPDATE TEST NUMBER
        CMP #164,(R2) ;SEQUENCE ERROR?
        BNE TST165-10 ;BR TO ERROR HALT ON SEQ ERROR
        MOV #52525,@#0 ;INITIALIZE DATA IN LOC 0
        SEC ;SET C-BIT
        ROL @#0 ;TR0 ROL W/ MODE 3
        BCS ROT3A ;CHECK C-BIT
        CMP #125253,@#0 ;CHECK DATA
        BEQ ROT3B

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

ROT3A: MOV #364,-(R2) ;MOVE TO MAILBOX # ***** 364 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;ROL W/ MODE 3 FAILED
ROT3B: MOV #125252,@#0 ;INITIALIZE DATA
        CLC ;CLEAR C-BIT
        ROLB @#0 ;TRY ROL W/ MODE 3 EVEN BYTE
        BCC ROT3C ;CHECK C-BIT
        CMP @#0,#125124 ;CHECK DATA
        BEQ ROT3D

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

ROT3C: MOV #365,-(R2) ;MOVE TO MAILBOX # ***** 365 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;ROL W/ MODE 3 EVEN BYTE FAILED
ROT3D: MOV #125252,@#0 ;INITIALIZE DATA IN LOC. 0
        SEC ;SET C-BIT
        ROLB @#1 ;TRY ROL W/ MODE 3 ODD BYTE
        BCC ROT3E ;CHECK C-BIT
        CMP #052652,@#0 ;CHECK DATA
        BEQ TST165

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

ROT3E: MOV #366,-(R2) ;MOVE TO MAILBOX # ***** 366 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;ROL W/ MODE 3 ODD BYTE FAILED
```



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

014506 005212  
014510 022712 000165  
014514 001016  
014516 012737 070707 000000  
014524 012700 000002  
014530 000261  
014532 006140  
014534 103406  
014536 022737 161617 000000  
014544 001002  
014546 005700  
014550 001404  
  
014552  
014552 012742 000367  
014556 005242  
014560 000000  
  
014562 005212  
014564 022712 000166  
014570 001021  
014572 012737 014644 000000  
014600 012700 000002  
014604 012767 107070 000032  
014612 000241  
014614 006150

```

; OR SEQUENCE ERROR
:*****
:
:   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
: R0 IS VERIFIED.
:*****
: TEST 165      TEST MODE 4 W/ ROTATE INSTRUCTIONS
:*****
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/ 762 <====
:
ROT4:   MOV     #367,-(R2)     ;MOVE TO MAILBOX # ***** 367 *****
        INC     -(R2)         ;SET MSGTYP TO FATAL ERROR
        HALT    ;ROL MODE 4 FAILED
:              ; OR SEQUENCE ERROR
:*****
:
:   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.
:*****
: TEST 166      TEST MODE 5 W/ ROTATE INSTRUCTIONS
:*****
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
    
```

```

4895 014616 103006
4896 014620 022737 016160 014644
4897 014626 001002
4898 014630 005700
4899 014632 001405
4900
4901
4902
4903
4904 014634
4905 014634 012742 000370
4906 014640 005242
4907 014642 000000
4908
4909 014644 000000
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921 014646 005212
4922 014650 022712 000167
4923 014654 001013
4924 014656 012737 125252 014644
4925 014664 000261
4926 014666 006167 177752
4927 014672 103004
4928 014674 022737 052525 014644
4929 014702 001404
4930
4931
4932
4933
4934 014704
4935 014704 012742 000371
4936 014710 005242
4937 014712 000000
4938
    
```

```

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/ 757 <====
ROT5:
MOV #370,-(R2) ;MOVE TO MAILBOX # ***** 370 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ROL MODE 5 FAILED
; OR SEQUENCE ERROR
ROT5: 0
ROT5: 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).
:
:*****
    
```

```

:*****
: TEST 167 TEST MODE 6 W/ ROTATE INSTRUCTIONS
:*****
    
```

```

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/ 765 <====
ROT6:
MOV #371,-(R2) ;MOVE TO MAILBOX # ***** 371 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;ROL W/ MODE 6 FAILED
; OR SEQUENCE ERROR
ROT6: 0
ROT6: 0
    
```

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

014714 005212  
014716 022712 000170  
014722 001016  
014724 012737 052525 014644  
014732 012737 014644 014770  
014740 000241  
014742 006177 000022  
014746 103404  
014750 023727 014644 125252  
014756 001405  
  
014760  
014760 012742 000372  
014764 005242  
014766 000000  
  
014770 000000  
  
014772 005212  
014774 022712 000171  
015000 001013  
015002 012700 177400  
015006 000300  
015010 100404  
  
015012 012742 000373  
015016 005242  
015020 000000

```
*****
: 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.
*****
: TEST 170 TEST MODE 7 W/ ROTATE INSTRUCTIONS
*****
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/ 762 <====
ROT7: MOV #372,-(R2) ;MOVE TO MAILBOX # ***** 372 *****
      INC -(R2) ;SET MSGTYP TO FATAL ERROR
      HALT ;ROL W/ MODE 7 FAILED
          ; OR SEQUENCE ERROR
ROTXAD: 0
```

```
*****
: 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.
*****
: TEST 171 TEST MODE 0 W/ SWAB INST.
*****
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 SBC
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 774 <====
        MOV #373,-(R2) ;MOVE TO MAILBOX # ***** 373 *****
        INC -(R2) ;SET MSGTYP TO FATAL ERROR
        HALT ;SWAB DID NOT SET CC'S CORRECT
```

```
4995 015022 022700 000377 SBO: CMP #377,R0 ;CHECK RESULT
4996 015026 001404 BEQ TST172 ;
4997 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
4998 ; CONDITIONAL BRANCH INST. AND <====
4999 ; REPLACE THE MOVE INSTRUCTION <====
5000 ; WHICH FOLLOWS W/ 765 <====
5001 015030 012742 000374 MOV #374,-(R2) ;MOVE TO MAILBOX # ***** 374 *****
5002 015034 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
5003 015036 000000 HALT ;RESULT OF SWAB MODE 0 FAILED
5004 ; OR SEQUENCE ERROR
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
```

```
*****
: 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
*****
```

```
5016 015040 005212 TST172: INC (R2) ;UPDATE TEST NUMBER
5017 015042 022712 000172 CMP #172,(R2) ;SEQUENCE ERROR?
5018 015046 001011 BNE TST173-10 ;BR TO ERROR HALT ON SEQ ERROR
5019 015050 012737 125652 000000 MOV #125652,@#0 ;MOVE TEST PATTERN TO LOC. 0
5020 015056 005000 CLR R0 ;R0=0
5021 015060 000310 SWAB (R0) ;TRY SWAB MODE 1
5022 015062 022737 125253 000000 CMP #125253,@#0 ;CHECK RESULT
5023 015070 001404 BEQ TST173 ;
5024 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5025 ; CONDITIONAL BRANCH INST. AND <====
5026 ; REPLACE THE MOVE INSTRUCTION <====
5027 ; WHICH FOLLOWS W/ 767 <====
5028 015072 012742 000375 MOV #375,-(R2) ;MOVE TO MAILBOX # ***** 375 *****
5029 015076 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
5030 015100 000000 HALT ;RESULT OF SWAB MODE 1 FAILED
5031 ; OR SEQUENCE ERROR
```

5032  
5033  
5034  
5035  
5036  
5037  
5038  
5039  
5040  
5041  
5042  
5043  
5044 015102 005212  
5045 015104 022712 000173  
5046 015110 001020  
5047 015112 012737 125152 000000  
5048 015120 005000  
5049 015122 000320  
5050 015124 022737 065252 000000  
5051 015132 001404  
5052  
5053  
5054  
5055  
5056 015134 012742 000376  
5057 015140 005242  
5058 015142 000000  
5059 015144 162700 000002  
5060 015150 001404  
5061  
5062  
5063  
5064  
5065 015152 012742 000377  
5066 015156 005242  
5067 015160 000000  
5068  
5069  
5070  
5071  
5072  
5073  
5074  
5075  
5076  
5077  
5078  
5079  
5080  
5081 015162 005212  
5082 015164 022712 000174  
5083 015170 001011  
5084 015172 012737 000377 000000  
5085 015200 000337 000000  
5086 015204 022737 177400 000000  
5087 015212 001404

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 173 TEST MODE 2 W/ SWAB INST  
\*\*\*\*\*

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/ 767 <====  
MOV #376,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 376 \*\*\*\*\*  
INC -(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/ 760 <====  
MOV #377,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 377 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;REGISTER VALUE INCORRECT  
: OR SEQUENCE ERROR

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 174 TEST MODE 3 W/SWAB INST.  
\*\*\*\*\*

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

5088  
5089  
5090  
5091  
5092 015214 012742 000400  
5093 015220 005242  
5094 015222 000000  
5095

MOV #400,-(R2)  
INC -(R2)  
HALT

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 767 <====  
: MOVE TO MAILBOX # \*\*\*\*\* 400 \*\*\*\*\*  
: SET MSGTYP TO FATAL ERROR  
: RESULT OF SWAB INCORRECT  
: OR SEQUENCE ERROR

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

015224 005212  
015226 022712 000175  
015232 001020  
015234 012737 125652 000000  
015242 012700 000002  
015246 000340  
015250 022737 125253 000000  
015256 001404  
  
  
  
  
  
  
  
  
  
015260 012742 000401  
015264 005242  
015266 000000  
015270 005700  
015272 001404  
  
  
  
  
  
015274 012742 000402  
015300 005242  
015302 000000

```
*****
: 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.
*****
: TEST 175 TEST MODE 4 W/ SWAB INST
*****
TST175: INC (R2) ;UPDATE TEST NUMBER
CMP #175,(R2) ;SEQUENCE ERROR?
BNE TST176-10 ;BR TO ERROR HALT ON SEQ ERROR
MOV #125652,@#0 ;MOVE TEST PATTERN TO LOC. 0
MOV #2,R0 ;SET UP REGISTER POINTER
SWAB -(R0) ;TRY SWAB MODE 4
CMP #125253,@#0 ;CHECK RESULT
BEQ SB4

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

MOV #401,-(R2) ;MOVE TO MAILBOX # ***** 401 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;RESULT OF SWAB INCORRECT
TST R0 ;CHECK EFFECT ON REG.
BEQ TST176

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

MOV #402,-(R2) ;MOVE TO MAILBOX # ***** 402 *****
INC -(R2) ;SET MSGTYP TO FATAL ERROR
HALT ;REGISTER VALUE INCORRECT
; OR SEQUENCE ERROR
```

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

015304 005212  
015306 022712 000176  
015312 001021  
015314 012700 015372  
015320 012767 125125 000040  
015326 000350  
015330 022767 052652 000030  
015336 001404  
  
015340 012742 000403  
015344 005242  
015346 000000  
015350 020027 015370  
015354 001406  
  
015356  
015356 012742 000404  
015362 005242  
015364 000000  
  
015366 000000  
015370 015366

```
*****
: 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.
*****
: TEST 176 TEST MODE 5 W/ SWAB INST.
*****
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
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 766 <====
:
: MOV #403,-(R2) ;MOVE TO MAILBOX # ***** 403 *****
: INC -(R2) ;SET MSGTYP TO FATAL ERROR
: HALT ;RESULT OF SWAB INCORRECT
SB5A: CMP R0,#SB5XAD ;CHECK RESULT OF REG.
: BEQ TST177
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 757 <====
:
SB5: MOV #404,-(R2) ;MOVE TO MAILBOX # ***** 404 *****
: INC -(R2) ;SET MSGTYP TO FATAL ERROR
: HALT ;REGISTER VALUE INCORRECT
: ; OR SEQUENCE ERROR
SB5X: 0 ;WORK LOCATION
SB5XAD: SB5X
```



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

015372 005212  
015374 022712 000177  
015400 001013  
015402 012767 125125 000030  
015410 012700 015432  
015414 000360 000006  
015420 022760 052652 000006  
015426 001405  
  
015430  
015430 012742 000405  
015434 005242  
015436 000000  
  
015440 000000

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

TEST 177 TEST MODE 6 W/ SWAB INST.

\*\*\*\*\*  
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/ 765 <====  
  
SB6: MOV #405,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 405 \*\*\*\*\*  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :RESULT OF SWAB INCORRECT  
: OR SEQUENCE ERROR  
SB6X: 0 :WORK LOCATION

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

015442 005212  
015444 022712 000200  
015450 001013  
015452 012767 177400 000030  
015460 012700 015420  
015464 000370 000072  
015470 027027 000072 000377  
015476 001406  
  
015500  
015500 012742 000406  
015504 005242  
015506 000000  
  
015510 000000  
015512 015510

```
*****  
: 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.  
: *****  
: TEST 200 TEST MODE 7 W/ SWAB INST.  
: *****  
TST200: INC (R2) ;UPDATE TEST NUMBER  
CMP #200,(R2) ;SEQUENCE ERROR?  
BNE SB7 ;BR TO ERROR HALT ON SEQ ERROR  
MOV #177400,SB7X ;MOVE PATTERN TO WORK LOCATION  
MOV #SB7XAD-72,R0 ;MOVE OFFSET POINTER TO R0  
SWAB @72(R0) ;TRY SWAB MODE 7  
CMP @72(R0),#377 ;CHECK RESULTS  
BEQ TST201  
  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 765 <====  
  
SB7: MOV #406,--(R2) ;MOVE TO MAILBOX # ***** 406 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;RESULT OF SWAB INCORRECT  
: OR SEQUENCE ERROR  
SB7X: 0 ;WORK LOCATION  
SB7XAD: SB7X ;POINTER TO WORK LOCATION
```

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

\*\*\*\*\*

: 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  
: 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.)

\*\*\*\*\*

: TEST 201 TEST THE JMP INSTRUCTION IN ALL MODES

\*\*\*\*\*

```
TST201: INC      (R2)           ;UPDATE TEST NUMBER
          CMP      #201,(R2)   ;SEQUENCE ERROR?
          BNE      JMPCK+6     ;BR TO ERROR HALT ON SEQ ERROR
          CLR      JMPSEQ      ;ESTABLISH A SEQUENCE CHECKER
          MOV      #JMP2,R0    ;SET R0=JUMP TARGET
          JMP      (R0)        ;TRY JMP MODE 1
JMP3:    CMP      #.+2,R0     ;CHECK RESULT OF MODE 2 JUMP
          BEQ      JMP3A

          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
          ;          CONDITIONAL BRANCH INST. AND <====
          ;          REPLACE THE MOVE INSTRUCTION <====
          ;          WHICH FOLLOWS W/ 770 <====
          MOV      #407,-(R2)  ;MOVE TO MAILBOX # ***** 407 *****
          INC      -(R2)      ;SET MSGTYP TO FATAL ERROR
          HALT                               ;REGISTER VALUE AFTER JMP MODE 2 INCORRECT
JMP3A:   CMP      JMPSEQ,#1   ;MAKE SURE JUMPS ARE IN SEQUENCE: JMPSEQ=1?
          BEQ      JMP3B

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

N 10

CFKAACO 11/34 BSC INST TST MACY11 30A(1052) 18-OCT-78 11:06 PAGE 118  
 CFKAAC.P11 18-OCT-78 11:01 T201 TEST THE JMP INSTRUCTION IN ALL MODES SEQ 0130

```

5298 015564 012742 000410      MOV    #410,-(R2)      ;MOVE TO MAILBOX # ***** 410 *****
5299 015570 005242      INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
5300 015572 000000      HALT                    ;SHOULD BE HERE FROM JMP MODE 2 ONLY
5301 015574 012700 015606      JMP3B: MOV    #IJMP4,R0 ;POINT R0 TO INDIRECT JMP ADDR.
5302 015600 005267 000252      INC    JMPSEQ         ;UPDATE SEQUENCE CHECKER
5303 015604 000130      JMP    @ (R0)+        ;TRY JMP MODE 3
5304 015606 015640      IJMP4: JMP4          ;ADDRESS INDIRECT JUMP
5305
5306 015610 005767 000242      JMP2:  TST    JMPSEQ   ;CHECK THAT JMPs ARE IN SEQUENCE: JMPSEQ=0?
5307 015614 001404      BEQ    JMP2A
5308
5309
5310
5311
5312 015616 012742 000411      MOV    #411,-(R2)      ;MOVE TO MAILBOX # ***** 411 *****
5313 015622 005242      INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
5314 015624 000000      HALT                    ;SHOULD BE HERE FROM JMP MODE 1 ONLY
5315 015626 005267 000224      JMP2A: INC    JMPSEQ   ;UPDATE SEQUENCE CHECKER
5316 015632 012700 015536      MOV    #JMP3,R0       ;SET R0=JUMP TARGET
5317 015636 000120      JMP    (R0)+          ;TRY A JUMP MODE 2 TO 'JMP3'
5318 015640 022700 015610      JMP4:  CMP    #IJMP4+2,R0 ;CHECK RESULT OF REGISTER IN MODE 3 JUMP
5319 015644 001404      BEQ    JMP4A
5320
5321
5322
5323
5324 015646 012742 000412      MOV    #412,-(R2)      ;MOVE TO MAILBOX # ***** 412 *****
5325 015652 005242      INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
5326 015654 000000      HALT                    ;REGISTER VALUE AFTER MODE 3 JUMP INCORRECT
5327 015656 022767 000002 000172  JMP4A: CMP    #2,JMPSEQ  ;CHECK JUMP SEQUENCE: JMPSEQ=2?
5328 015664 001404      BEQ    JMP4B
5329
5330
5331
5332
5333 015666 012742 000413      MOV    #413,-(R2)      ;MOVE TO MAILBOX # ***** 413 *****
5334 015672 005242      INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
5335 015674 000000      HALT                    ;SHOULD BE ONLY FROM MODE 3 JUMP
5336 015676 012700 015746      JMP4B: MOV    #JMP5+2,R0 ;SET UP POINTER TO JUMP TARGET
5337 015702 005267 000150      INC    JMPSEQ         ;UPDATE SEQUENCE CHECKER
5338 015706 000140      JMP    -(R0)         ;TRY JUMP MODE 4 TO 'JMP4'
5339
5340 015710 022767 000004 000140  JMP6:  CMP    #4,JMPSEQ  ;CHECK THAT JUMPS ARE IN SEQUENCE: JMPSEQ=4?
5341 015716 001404      BEQ    JMP6A
5342
5343
5344
5345
5346 015720 012742 000414      MOV    #414,-(R2)      ;MOVE TO MAILBOX # ***** 414 *****
5347 015724 005242      INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
5348 015726 000000      HALT                    ;SHOULD BE HERE ONLY FROM MODE 5 JUMP
5349 015730 012700 016376      JMP6A: MOV    #JMP7+376,R0 ;SET UP OFFSET POINTER TO JUMP TARGET
5350 015734 005267 000116      INC    JMPSEQ         ;UPDATE JUMP SEQUENCE
5351 015740 000160 177402      JMP    -376(R0)       ;TRY MODE 6 JUMP
5352
5353 015744 022767 000003 000104  JMP5:  CMP    #3,JMPSEQ  ;CHECK THAT JUMPS ARE IN SEQUENCE: JMPSEQ=3?

```

```
5354 015752 001404          BEQ      JMP5A          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5355                                     ;          CONDITIONAL BRANCH INST. AND <====
5356                                     ;          REPLACE THE MOVE INSTRUCTION <====
5357                                     ;          WHICH FOLLOWS W/ 664 <====
5358                                     ;          ***** 415 *****
5359 015754 012742 000415    MOV      #415,-(R2)      ;MOVE TO MAILBOX # ***** 415 *****
5360 015760 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5361 015762 000000          HALT                    ;SHOULD ONLY BE HERE FROM MODE 4 JUMP
5362 015764 012700 016000    JMP5A:  MOV      #IJMP5+2,R0 ;SET UP POINTER TO INDIRECT JUMP ADDR.
5363 015770 005267 000062          INC      JMPSEQ        ;UPDATE JUMP SEQUENCE
5364 015774 000150          JMP      @-(R0)        ;TRY JUMP MODE 5 TO "JMP6"
5365 015776 015710          IJMP5:  JMP6          ;INDIRECT ADDRESS POINTER
5366
5367 016000 022767 000005 000050 JMP7:   CMP      #5,JMPSEQ   ;CHECK JUMPS IN SEQUENCE: JMPSEQ=5?
5368 016006 001404          BEQ      JMP7A
5369                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5370                                     ;          CONDITIONAL BRANCH INST. AND <====
5371                                     ;          REPLACE THE MOVE INSTRUCTION <====
5372                                     ;          WHICH FOLLOWS W/ 646 <====
5373 016010 012742 000416    MOV      #416,-(R2)      ;MOVE TO MAILBOX # ***** 416 *****
5374 016014 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5375 016016 000000          HALT                    ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
5376 016020 012700 016044    JMP7A:  MOV      #IJMP+10,R0 ;SET UP OFFSET POINTER TO INDIRECT ADDR.
5377 016024 005267 000026          INC      JMPSEQ        ;UPDATE JUMP SEQUENCE
5378 016030 000170 177770          JMP      @-10(R0)      ;TRY MODE 7 JUMP
5379 016034 016036          IJMP:   JMPCK
5380
5381 016036 026727 000014 000006 JMPCK:  CMP      JMPSEQ,#6   ;CHECK JUMPS IN SEQUENCE: JMPSEQ
5382 016044 001405          BEQ      TST202
5383                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5384                                     ;          CONDITIONAL BRANCH INST. AND <====
5385                                     ;          REPLACE THE MOVE INSTRUCTION <====
5386                                     ;          WHICH FOLLOWS W/ 627 <====
5387 016046 012742 000417    MOV      #417,-(R2)      ;MOVE TO MAILBOX # ***** 417 *****
5388 016052 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5389 016054 000000          HALT                    ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
5390
5391 016056 000000          JMPSEQ: 0                ; OR SEQUENCE ERROR
```

5392  
5393  
5394  
5395  
5396  
5397  
5398  
5399  
5400  
5401  
5402  
5403  
5404  
5405  
5406  
5407  
5408  
5409  
5410  
5411  
5412 016060 005212  
5413 016062 022712 000202  
5414 016066 001001  
5415 016070 000402  
5416 016072 000137 016526  
5417  
5418 016076 012706 000500  
5419 016102 012700 016210  
5420 016106 005037 016506  
5421 016112 005001  
5422 016114 005101  
5423 016116 004110  
5424  
5425  
5426 016120  
5427 016120 012742 000420  
5428 016124 005242  
5429 016126 000000  
5430  
5431 016130 022737 000001 016506  
5432 016136 001014  
5433 016140 020127 016272  
5434 016144 001011  
5435 016146 022706 000476  
5436 016152 001006  
5437 016154 022716 125252  
5438 016160 001003  
5439 016162 022700 016132  
5440 016166 001404  
5441  
5442  
5443  
5444  
5445 016170  
5446 016170 012742 000421  
5447 016174 005242

```
*****  
: 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).  
*****  
: TEST 202 TEST JSR INSTRUCTION W/ ALL MODES  
*****  
TST202: INC (R2) ;UPDATE TEST NUMBER  
CMP #202,(R2) ;SEQUENCE ERROR?  
BNE JSR0 ;BR TO ERROR HALT ON SEQ ERROR  
BR JSR1  
JSR0: JMP @#JSRCK1  
JSR1: MOV #STBOT,R6 ;SET STACK POINTER  
MOV #JSR2,R0 ;SET TARGET ADDRESS  
CLR @#JSRSEQ ;INITIALIZE SEQUENCE CHECKER  
CLR R1 ;INITIALIZE R1  
COM R1  
JSR R1,(R0) ;TRY JSR MODE 1  
; TO SCOPE: REPLACE THE MOVE INSTRUCTION <=====  
; FOLLOWING W/ 774 <=====  
JSR1A: MOV #420,-(R2) ;MOVE TO MAILBOX # ***** 420 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;JSR MODE 1 FAILED  
JSR3: CMP #1,@#JSRSEQ ;CHECK SEQUENCE: JSRSEQ=1?  
BNE JSR3A ;BRANCH IF OUT OF SEQUENCE  
CMP R1,#JSR4 ;PROPER PC SAVED?  
BNE JSR3A ;BRANCH IF PC WRONG  
CMP #STBOT-2,R6 ;STACK POINTER DECREMENTED?  
BNE JSR3A ;BRANCH IF SP WRONG  
CMP #125252,(R6) ;REG SAVED ON STACK?  
BNE JSR3A ;BRANCH IF REG. NOT SAVED  
CMP #JSR3+2,R0 ;MODE 2 INCREMENT CORRECT?  
BEQ JSR3B  
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
; CONDITIONAL BRANCH INST. AND <=====  
; REPLACE THE MOVE INSTRUCTION <=====  
; WHICH FOLLOWS W/ 740 <=====  
JSR3A: MOV #421,-(R2) ;MOVE TO MAILBOX # ***** 421 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR
```

```
5448 016176 000000  
5449 016200 005237 016506 JSR3B: HALT ;JSR MODE 3 MALFUNCTIONED  
5450 016204 004137 016272 INC @#JSRSEQ ;UPDATE SEQUENCE CHECKER  
5451 JSR R1,@#JSR4 ;TRY JSR MODE 4  
5452 016210 005737 016506 JSR2: TST @#JSRSEQ ;CHECK SEQUENCE: JSRSEQ=0?  
5453 016214 001011 BNE JSR2A ;BRANCH IF OUT OF SEQUENCE  
5454 016216 020127 016120 CMP R1,#JSR1A ;PROPER PC SAVED?  
5455 016222 001006 BNE JSR2A ;BRANCH IF PC WRONG  
5456 016224 022706 000476 CMP #STBOT-2,R6 ;R6 DECREMENT?  
5457 016230 001003 BNE JSR2A ;BRANCH IF R6 IS INCORRECT  
5458 016232 021627 177777 CMP (R6),#-1 ;REGISTER SAVED?  
5459 016236 001404 BEQ JSR2B  
5460 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
5461 ; CONDITIONAL BRANCH INST. AND <====  
5462 ; REPLACE THE MOVE INSTRUCTION <====  
5463 ; WHICH FOLLOWS W/ 714 <====  
5464 016240 JSR2A:  
5465 016240 012742 000422 MOV #422,-(R2) ;MOVE TO MAILBOX # ***** 422 *****  
5466 016244 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
5467 016246 000000 HALT ;JSR MODE 1 MALFUNCTIONED  
5468 016250 012706 000500 JSR2B: MOV #STBOT,R6 ;INITIALIZE R6  
5469 016254 012701 125252 MOV #125252,R1 ;INITIALIZE R1  
5470 016260 005237 016506 INC @#JSRSEQ ;UPDATE SEQUENCE CHECKER  
5471 016264 012700 016130 MOV #JSR3,R0 ;SET TARGET ADDRESS  
5472 016270 004120 JSR R1,(R0)+ ;TRY JSR MODE 2  
5473  
5474 016272 022737 000002 016506 JSR4: CMP #2,@#JSRSEQ ;CHECK SEQUENCE: JSRSEQ=2?  
5475 016300 001003 BNE JSR4A ;BRANCH IF OUT OF SEQUENCE  
5476 016302 022701 016210 CMP #JSR2,R1 ;PROPER PC SAVED?  
5477 016306 001404 BEQ JSR4B  
5478 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
5479 ; CONDITIONAL BRANCH INST. AND <====  
5480 ; REPLACE THE MOVE INSTRUCTION <====  
5481 ; WHICH FOLLOWS W/ 670 <====  
5482 016310 JSR4A:  
5483 016310 012742 000423 MOV #423,-(R2) ;MOVE TO MAILBOX # ***** 423 *****  
5484 016314 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
5485 016316 000000 HALT ;JSR MODE 3 MALFUNCTIONED  
5486 016320 005237 016506 JSR4B: INC @#JSRSEQ ;UPDATE SEQUENCE CHECKER  
5487 016324 012700 016400 MOV #JSR5+2,R0 ;SET TARGET ADDRESS  
5488 016330 004140 JSR R1,-(R0) ;TRY JSR MODE 4  
5489  
5490 016332 022767 000004 000146 JSR6: CMP #4,JSRSEQ ;CHECK SEQUENCE: JSRSEQ=4?  
5491 016340 001006 BNE JSR6A ;BRANCH IF OUT OF SEQUENCE  
5492 016342 022701 016444 CMP #JSR7,R1 ;PROPER PC SAVED?  
5493 016346 001003 BNE JSR6A ;BRANCH IF PC WRONG  
5494 016350 022700 016502 CMP #JSR6AD,R0 ;MODE 5 REGISTER CORRECT?  
5495 016354 001404 BEQ JSR6B  
5496 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
5497 ; CONDITIONAL BRANCH INST. AND <====  
5498 ; REPLACE THE MOVE INSTRUCTION <====  
5499 ; WHICH FOLLOWS W/ 645 <====  
5500 016356 JSR6A:  
5501 016356 012742 000424 MOV #424,-(R2) ;MOVE TO MAILBOX # ***** 424 *****  
5502 016362 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
5503 016364 000000 HALT ;JSR MODE 5 FAILED
```





5558  
5559  
5560  
5561  
5562  
5563  
5564  
5565  
5566  
5567  
5568 016536 005212  
5569 016540 022712 000203  
5570 016544 001016  
5571 016546 012706 000500  
5572 016552 012746 052525  
5573 016556 012700 016574  
5574 016562 000200  
5575  
5576  
5577 016564 012742 000430  
5578 016570 005242  
5579 016572 000000  
5580 016574 022700 052525  
5581 016600 001404  
5582  
5583  
5584  
5585  
5586 016602 012742 000431  
5587 016606 005242  
5588 016610 000000  
5589

```
*****  
: 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.  
*****  
: TEST 203 TEST RTS INSTRUCTION  
*****  
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 #RTS1,R0 :INITIALIZE RETURN REGISTER  
RTS R0 :TRY RTS THROUGH R0  
: TO SCOPE: REPLACE THE MOVE INSTRUCTION <====  
: FOLLOWING W/ 770 <====  
MOV #430,-(R2) :MOVE TO MAILBOX # ***** 430 *****  
INC -(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/ 762 <====  
MOV #431,-(R2) :MOVE TO MAILBOX # ***** 431 *****  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :RTS MALFUNCTIONED  
: OR SEQUENCE ERROR
```

5590  
5591  
5592  
5593  
5594  
5595  
5596  
5597  
5598  
5599  
5600  
5601  
5602  
5603  
5604  
5605  
5606  
5607 016612 005212  
5608 016614 022712 000204  
5609 016620 001022  
5610 016622 000277  
5611 016624 000251  
5612 016626 012700 100000  
5613 016632 101402  
5614 016634 102401  
5615 016636 100404  
5616  
5617  
5618  
5619  
5620 016640  
5621 016640 012742 000432  
5622 016644 005242  
5623 016646 000000  
5624  
5625 016650 000277  
5626 016652 000244  
5627 016654 012700 000000  
5628 016660 101002  
5629 016662 102401  
5630 016664 100004  
5631  
5632  
5633  
5634  
5635 016666  
5636 016666 012742 000433  
5637 016672 005242  
5638 016674 000000  
5639  
5640  
5641  
5642  
5643 016676 005212  
5644 016700 022712 000205  
5645 016704 001024

```
*****  
: 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.  
*****  
: TEST 204 TEST MOV INSTRUCTION  
*****  
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/ 771 <====  
MOV1: MOV #432,-(R2) ;MOVE TO MAILBOX # ***** 432 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;MOV DID NOT SET CC'S CORRECTLY  
MOV2: SCC ;CC=1011  
CLZ  
MOV #0,R0 ;CC=0101  
RHI 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/ 756 <====  
MOV3: MOV #433,-(R2) ;MOVE TO MAILBOX # ***** 433 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;MOV DID NOT SET CC'S CORRECTLY  
: OR SEQUENCE ERROR  
*****  
: TEST 205 TEST BIT INSTRUCTION  
*****  
TST205: INC (R2) ;UPDATE TEST NUMBER  
CMP #205,(R2) ;SEQUENCE ERROR?  
BNE TST206-10 ;BR TO ERROR HALT ON SEQ ERROR
```

```

5646 016706 012700 100001      MOV      #100001,R0
5647 016712 000277      SCC
5648 016714 000251      +CLN:CLC      ;CC=0110
5649 016716 032700 100000      BIT      #100000,R0      ;CC=1000
5650 016722 101402      BLOS     BIT1
5651 016724 102401      BVS     BIT1
5652 016726 100404      BMI     BIT2
5653
5654      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5655      ; CONDITIONAL BRANCH INST. AND <====
5656      ; REPLACE THE MOVE INSTRUCTION <====
5657      ; WHICH FOLLOWS W/ 767 <====
5657 016730      BIT1:
5658 016730 012742 000434      MOV      #434,-(R2)      ;MOVE TO MAILBOX # ***** 434 *****
5659 016734 005242      INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5660 016736 000000      HALT     ;BIT DID NOT SET CC'S CORRECTLY
5661
5662 016740 000277      BIT2:  SCC              ;CC=1011
5663 016742 000244      CLZ
5664 016744 032700 077776      BIT      #77776,R0      ;CC=0101
5665 016750 101002      BHI     BIT3
5666 016752 102401      BVS     BIT3
5667 016754 100004      BPL     TST206
5668
5669      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5670      ; CONDITIONAL BRANCH INST. AND <====
5671      ; REPLACE THE MOVE INSTRUCTION <====
5672      ; WHICH FOLLOWS W/ 754 <====
5672 016756      BIT3:
5673 016756 012742 000435      MOV      #435,-(R2)      ;MOVE TO MAILBOX # ***** 435 *****
5674 016762 005242      INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5675 016764 000000      HALT     ;BIT DID NOT SET CC'S CORRECTLY
5676      ; OR SEQUENCE ERROR
5677
5678      ;*****
5679      ;TEST 206      TEST BIC INSTRUCTION
5680 016766 005212      TST206: INC      (R2)          ;UPDATE TEST NUMBER
5681 016770 022712 000206      CMP      #206,(R2)      ;SEQUENCE ERROR?
5682 016774 001024      BNE     TST207-10      ;BR TO ERROR HALT ON SEQ ERROR
5683 016776 012700 177777      MOV      #177777,R0
5684 017002 000277      SCC              ;CC=0110
5685 017004 000251      +CLN:CLC
5686 017006 042700 077777      BIC      #77777,R0      ;CC=1000
5687 017012 101402      BLOS     BIC1
5688 017014 102401      BVS     BIC1
5689 017016 100404      BMI     BIC2
5690
5691      ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5692      ; CONDITIONAL BRANCH INST. AND <====
5693      ; REPLACE THE MOVE INSTRUCTION <====
5694      ; WHICH FOLLOWS W/ 767 <====
5694 017020      BIC1:
5695 017020 012742 000436      MOV      #436,-(R2)      ;MOVE TO MAILBOX # ***** 436 *****
5696 017024 005242      INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5697 017026 000000      HALT     ;BIC DID NOT SET CC'S CORRECTLY
5698 017030 000277      BIC2:  SCC              ;CC=1011
5699 017032 000244      CLZ
5700 017034 042700 100000      BIC      #100000,R0      ;CC=0101
5701 017040 101002      BHI     BIC3

```

5702 017042 102401  
5703 017044 100004  
5704  
5705  
5706  
5707  
5708 017046  
5709 017046 012742 000437  
5710 017052 005242  
5711 017054 000000  
5712  
5713  
5714  
5715  
5716 017056 005212  
5717 017060 022712 000207  
5718 017064 001025  
5719 017066 005000  
5720 017070 000277  
5721 017072 000251  
5722 017074 052700 000000  
5723 017100 103403  
5724 017102 102402  
5725 017104 100401  
5726 017106 001404  
5727  
5728  
5729  
5730  
5731 017110  
5732 017110 012742 000440  
5733 017114 005242  
5734 017116 000000  
5735 017120 000277  
5736 017122 000250  
5737 017124 052700 177777  
5738 017130 103003  
5739 017132 102402  
5740 017134 001401  
5741 017136 100404  
5742  
5743  
5744  
5745  
5746 017140  
5747 017140 012742 000441  
5748 017144 005242  
5749 017146 000000  
5750

BVS BIC3  
BPL TST207  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 754 <====  
BIC3: MOV #437, -(R2) :MOVE TO MAILBOX # \*\*\*\*\* 437 \*\*\*\*\*  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :BIC DID NOT SET CC'S CORRECTLY  
: OR SEQUENCE ERROR  
:\*\*\*\*\*  
:TEST 207 TEST BIS INSTRUCTION  
:\*\*\*\*\*  
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 :CC=0100 R0=0  
BIS #0, R0  
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/ 767 <====  
BIS1: MOV #440, -(R2) :MOVE TO MAILBOX # \*\*\*\*\* 440 \*\*\*\*\*  
INC -(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/ 753 <====  
BIS3: MOV #441, -(R2) :MOVE TO MAILBOX # \*\*\*\*\* 441 \*\*\*\*\*  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :BIS DID NOT SET CC'S CORRECTLY  
: OR SEQUENCE ERROR

5751  
5752  
5753  
5754  
5755  
5756  
5757  
5758  
5759  
5760  
5761  
5762  
5763  
5764  
5765  
5766 017150 005212  
5767 017152 022712 000210  
5768 017156 001037  
5769 017160 012700 077777  
5770 017164 000257  
5771 017166 000264  
5772 017170 005200  
5773 017172 101402  
5774 017174 100001  
5775 017176 102404  
5776  
5777  
5778  
5779  
5780 017200  
5781 017200 012742 000442  
5782 017204 005242  
5783 017206 000000  
5784 017210 052700 077777  
5785 017214 000261  
5786 017216 000244  
5787 017220 005200  
5788 017222 100403  
5789 017224 102402  
5790 017226 103001  
5791 017230 001404  
5792  
5793  
5794  
5795  
5796 017232  
5797 017232 012742 000443  
5798 017236 005242  
5799 017240 000000  
5800  
5801 017242 000277  
5802 017244 000241  
5803 017246 005200  
5804 017250 101402  
5805 017252 100401  
5806 017254 100004

```
*****
:
: 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.
:
: *****
: TEST 210 TEST INC INSTRUCTION
: *****
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
: INC R0 ;CC=1010 R0=10000
: 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/ 770 <====
:
INC1: MOV #442,-(R2) ;MOVE TO MAILBOX # ***** 442 *****
: INC -(R2) ;SET MSGTYP TO FATAL ERROR
: HALT ;INC DID NOT SET CC'S CORRECTLY
INC2: BIS #77777,R0 ;R0=177777
: SEC ;CC=1011
: CLZ
: INC R0 ;CC=0101 R0=0
: BMI INC3
: BVS INC3
: BCC INC3
: BEQ INC4
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 753 <====
:
INC3: MOV #443,-(R2) ;MOVE TO MAILBOX # ***** 443 *****
: INC -(R2) ;SET MSGTYP TO FATAL ERROR
: HALT ;INC DID NOT SET CC'S CORRECTLY
INC4: SCC ;CC=1110
: CLC
: INC R0 ;CC=0000 R0=1
: BLOS INC5
: BMI INC5
: BPL TST211
```

```
5807                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5808                                     :                                     <====
5809                                     :                                     <====
5810                                     :                                     <====
5811 017256                               INC5: MOV #444,-(R2)      ;MOVE TO MAILBOX # ***** 444 *****
5812 017256 012742 000444                INC  -(R2)      ;SET MSGTYP TO FATAL ERROR
5813 017262 005242                        HALT              ;INC DID NOT SET CC'S CORRECTLY
5814 017264 000000                        ; OR SEQUENCE ERROR
5815
5816
5817
5818
5819
5820 017266 005212                               :*****
5821 017270 022712 000211                :TEST 211      TEST DEC INSTRUCTION
5822 017274 001051                               :*****
5823 017276 012700 000002                TST211: INC (R2)      ;UPDATE TEST NUMBER
5824 017302 000277                               CMP #211,(R2)      ;SEQUENCE ERROR?
5825 017304 005300                               BNE TST212-10     ;BR TO ERROR HALT ON SEQ ERROR
5826 017306 100403                               MOV #2,R0         ;R0=2
5827 017310 001402                               SCC              ;CC=1111
5828 017312 102401                               DEC R0           ;CC=0001 R0=1
5829 017314 103404                               BMI DEC1
5830                                     BEQ DEC1
5831                                     BVS DEC1
5832                                     BCS DEC2
5833
5834                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5835                                     :                                     <====
5836                                     :                                     <====
5837                                     :                                     <====
5838 017316                               DEC1: MOV #445,-(R2)      ;MOVE TO MAILBOX # ***** 445 *****
5839 017322 005242 000445                INC  -(R2)      ;SET MSGTYP TO FATAL ERROR
5840 017324 000000                               HALT              ;DEC DID NOT SET CC'S CORRECTLY
5841 017326 000261                               DEC2: SEC         ;CC=1011
5842 017330 000244                               CLZ
5843 017332 005300                               DEC R0           ;CC=0101 R0=0
5844 017334 101002                               BHI DEC3
5845 017336 100401                               BMI DEC3
5846 017340 102004                               BVC DEC4
5847
5848                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5849                                     :                                     <====
5850                                     :                                     <====
5851                                     :                                     <====
5852 017342                               DEC3: MOV #446,-(R2)      ;MOVE TO MAILBOX # ***** 446 *****
5853 017342 012742 000446                INC  -(R2)      ;SET MSGTYP TO FATAL ERROR
5854 017350 000000                               HALT              ;DEC DID NOT SET CC'S CORRECTLY
5855 017352 000277                               DEC4: SCC         ;CC=0110
5856 017354 000251                               +CLN!CLC
5857 017356 005300                               DEC R0           ;CC=1000 R0=177777
5858 017360 101402                               BLOS DEC5
5859 017362 102401                               BVS DEC5
5860 017364 100404                               BMI DEC6
5861
5862 017366                               DEC5:
5863                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
5864                                     :                                     <====
5865                                     :                                     <====
5866                                     :                                     <====
```

```
5863 017366 012742 000447      MOV      #447,-(R2)      ;MOVE TO MAILBOX # ***** 447 *****
5864 017372 005242              INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5865 017374 000000              HALT                    ;DEC DID NOT SET CC'S CORRECTLY
5866 017376 042700 077777      DEC6:  BIC      #77777,R0 ;R0=100000
5867 017402 000277              SCC                    ;CC=0101
5868 017404 000252              +CLN!CLV
5869 017406 005300              DFC      R0            ;CC=1011  R0=77777
5870 017410 100403              BMI      DEC7          ;CC=0011
5871 017412 001402              BEQ      DEC7
5872 017414 102001              BVC      DEC7
5873 017416 103404              BCS      TST212
5874
5875
5876
5877
5878 017420
5879 017420 012742 000450      DEC7:  MOV      #450,-(R2) ;MOVE TO MAILBOX # ***** 450 *****
5880 017424 005242              INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
5881 017426 000000              HALT                    ;DEC DID NOT SET CC'S CORRECTLY
5882
5883
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
;          CONDITIONAL BRANCH INST. AND <====
;          REPLACE THE MOVE INSTRUCTION <====
;          WHICH FOLLOWS W/ 727 <====
```

5884  
5885  
5886  
5887  
5888  
5889  
5890  
5891  
5892  
5893  
5894  
5895  
5896  
5897 017430 005212  
5898 017432 022712 000212  
5899 017436 001007  
5900 017440 000277  
5901 017442 000244  
5902 017444 005000  
5903 017446 100403  
5904 017450 102402  
5905 017452 103401  
5906 017454 001404  
5907  
5908  
5909  
5910  
5911 017456  
5912 017456 012742 000451  
5913 017462 005242  
5914 017464 000000  
5915  
5916  
5917  
5918  
5919  
5920 017466 005212  
5921 017470 022712 000213  
5922 017474 001022  
5923 017476 000277  
5924 017500 000244  
5925 017502 005700  
5926 017504 100403  
5927 017506 102402  
5928 017510 103401  
5929 017512 001404  
5930  
5931  
5932  
5933  
5934 017514  
5935 017514 012742 000452  
5936 017520 005242  
5937 017522 000000  
5938 017524 005300  
5939 017526 000277

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 212 TEST CLR INSTRUCTION  
\*\*\*\*\*

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/ 771 <=====

CLR1: MOV #451,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 451 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;CLR DID NOT SET CC'S CORRECTLY  
; OR SEQUENCE ERROR

\*\*\*\*\*  
: TEST 213 TEST TST INSTRUCTION  
\*\*\*\*\*

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/ 771 <=====

TEST1: MOV #452,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 452 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;TEST DID NOT SET CC'S CORRECTLY  
TEST2: DEC R0 ;MAKE R0 NEGATIVE  
SCC ;CC=0111



```
5940 017530 000250      CLN
5941 017532 005700      TST      R0      ;CC=1000
5942 017534 101402      BLOS     TEST3
5943 017536 102401      BVS     TEST3
5944 017540 100404      BMI     TST214
5945
5946
5947
5948
5949 017542
5950 017542 012742 000453      TEST3:  MOV     #453,-(R2)      ;MOVE TO MAILBOX # ***** 453 *****
5951 017546 005242      INC     -(R2)      ;SET MSGTYP TO FATAL ERROR
5952 017550 000000      HALT
5953
5954
5955
5956
5957 017552 005212
5958 017554 022712 000214      TST214: INC     (R2)      ;UPDATE TEST NUMBER
5959 017560 001023      CMP     #214,(R2)      ;SEQUENCE ERROR?
5960 017562 012700 170000      BNE     TST215-10      ;BR TO ERROR HALT ON SEQ ERROR
5961 017566 000277      MOV     #170000,R0      ;R0=170000
5962 017570 000250      SCC
5963 017572 000300      CLN
5964 017574 101402      SWAB    R0      ;CC=1000  R0=360
5965 017576 102401      BLOS    SWB1
5966 017600 100404      BVS     SWB1
5967
5968
5969
5970
5971 017602
5972 017602 012742 000454      SWB1:  MOV     #454,-(R2)      ;MOVE TO MAILBOX # ***** 454 *****
5973 017606 005242      INC     -(R2)      ;SET MSGTYP TO FATAL ERROR
5974 017610 000000      HALT
5975 017612 000277      SWB2:  SCC
5976 017614 000244      CLZ
5977 017616 000300      SWAB    R0      ;CC=0100  R0=170000
5978 017620 102403      BVS     SWB3
5979 017622 103402      BCS     SWB3
5980 017624 100401      BMI     SWB3
5981 017626 001404      BEQ     TST215
5982
5983
5984
5985
5986 017630
5987 017630 012742 000455      SWB3:  MOV     #455,-(R2)      ;MOVE TO MAILBOX # ***** 455 *****
5988 017634 005242      INC     -(R2)      ;SET MSGTYP TO FATAL ERROR
5989 017636 000000      HALT

;*****
;TEST 214      TEST SWAB INSTRUCTION
;*****
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 756 <====

; 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 BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 755 <====
```

5990  
5991  
5992  
5993  
5994  
5995  
5996  
5997  
5998  
5999  
6000  
6001  
6002  
6003  
6004 017640 005212  
6005 017642 022712 000215  
6006 017646 001062  
6007 017650 012700 040000  
6008 017654 000277  
6009 017656 062700 030000  
6010 017662 101402  
6011 017664 102401  
6012 017666 100004  
6013  
6014  
6015  
6016  
6017 017670  
6018 017670 012742 000456  
6019 017674 005242  
6020 017676 000000  
6021 117700 000264  
6022  
6023 017702 062700 010000  
6024 017706 101402  
6025 017710 102001  
6026 017712 100404  
6027  
6028  
6029  
6030  
6031 017714  
6032 017714 012742 000457  
6033 017720 005242  
6034 017722 000000  
6035 017724 000257  
6036 017726 000270  
6037 017730 062700 100000  
6038 017734 101002  
6039 017736 102001  
6040 017740 100004  
6041  
6042  
6043  
6044  
6045 017742

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 215 TEST ADD INSTRUCTION  
\*\*\*\*\*

TST215: INC (R2) ;UPDATE TEST NUMBER  
CMP #215,(R2) ;SEQUENCE ERROR?  
BNE TST216-10 ;BR TO ERROR HALT ON SEQ ERROR  
MOV #40000,R0 ;R0=40000  
SCC ;CC=1111  
ADD #30000,R0 ;CC=0000 R0=70000  
BLOS ADD1  
BVS ADD1  
BPL ADD2

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

ADD1: MOV #456,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 456 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
ADD2: HALT ;ADD DID NOT SET CC'S CORRECTLY  
SEZ ;CC=0100  
ADD #10000,R0 ;CC=1010 40=100000  
BLOS ADD3  
BVC ADD3  
BMI ADD4

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

ADD3: MOV #457,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 457 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
ADD4: HALT ;ADD DID NOT SET CC'S CORRECTLY  
CCC ;CC=1000  
SEN  
ADD #100000,R0 ;CC=0111 R0=0  
BHI ADD5  
BVC ADD5  
BPL ADD6

: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 743 <====

ADD5:

```
6046 017742 012742 000460      MOV    #460,-(R2)      ;MOVE TO MAILBOX # ***** 460 *****
6047 017746 005242              INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
6048 017750 000000              HALT                    ;ADD DID NOT SET CC'S CORRECTLY
6049 017752 062700 177777      ADD6:  ADD    #177777,R0 ;CC=1000  R0=177777
6050 017756 101402              BLOS  ADD7
6051 017760 102401              BVS   ADD7
6052 017762 100404              BMI   ADD8
6053                                ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
6054                                ;                               CONDITIONAL BRANCH INST. AND <=====
6055                                ;                               REPLACE THE MOVE INSTRUCTION <=====
6056                                ;                               WHICH FOLLOWS W/ 732 <=====
6057 017764                                ADD7:
6058 017764 012742 000461      MOV    #461,-(R2)      ;MOVE TO MAILBOX # ***** 461 *****
6059 017770 005242              INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
6060 017772 000000              HALT                    ;ADD DID NOT SET CC'S CORRECTLY
6061 017774 000277      ADD8:  SCC
6062 017776 000245              +CLC!CLZ
6063 020000 062700 000001      ADD    #1,R0           ;CC=0101  R=0
6064 020004 102403              BVS   ADD9
6065 020006 103002              BCC   ADD9
6066 020010 100401              BMI   ADD9
6067 020012 001404              BEQ   TST216
6068                                ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
6069                                ;                               CONDITIONAL BRANCH INST. AND <=====
6070                                ;                               REPLACE THE MOVE INSTRUCTION <=====
6071                                ;                               WHICH FOLLOWS W/ 716 <=====
6072 020014                                ADD9:
6073 020014 012742 000462      MOV    #462,-(R2)      ;MOVE TO MAILBOX # ***** 462 *****
6074 020020 005242              INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
6075 020022 000000              HALT                    ;ADD DID NOT SET CC'S CORRECTLY
6076                                ; OR SEQUENCE ERROR
6077
6078
6079
6080
6081 020024 005212                                ;*****
6082 020026 022712 000216      ;TEST 216 TEST ADC INSTRUCTION
6083 020032 001037                                ;*****
6084 020034 012700 077777      TST216: INC    (R2)          ;UPDATE TEST NUMBER
6085 020040 000277              CMP    #216,(R2)      ;SEQUENCE ERROR?
6086 020042 000252              BNE   TST217-10      ;BR TO ERROR HALT ON SEQ ERROR
6087 020044 005500              MOV    #077777,R0
6088 020046 101402              SCC
6089 020050 102001              +CLN!CLV
6090 020052 100404              ADC    R0             ;CC=0101
6091                                ;CC=1010
6092                                ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
6093                                ;                               CONDITIONAL BRANCH INST. AND <=====
6094                                ;                               REPLACE THE MOVE INSTRUCTION <=====
6095                                ;                               WHICH FOLLOWS W/ 770 <=====
6096 020054                                ADC1:
6097 020054 012742 000463      MOV    #463,-(R2)      ;MOVE TO MAILBOX # ***** 463 *****
6098 020060 005242              INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
6099 020062 000000              HALT                    ;ADC DID NOT SET CC'S CORRECTLY
6100 020070 000277      ADC2:  BIS    #77777,R0
6101 020072 000244              SCC
              CLZ
              ;CC=1011
```

```
6102 020074 005500          ADC      R0          ;CC=0101  R0=0
6103 020076 101002          BHI     ADC3
6104 020100 102401          BVS     ADC3
6105 020102 100004          BPL     ADC4
6106
6107
6108
6109
6110 020104          ADC3:
6111 020104 012742 000464          MOV     #464,-(R2)      ;MOVE TO MAILBOX # ***** 464 *****
6112 020110 005242          INC     -(R2)          ;SET MSGTYP TO FATAL ERROR
6113 020112 000000          HALT
6114 020114 000277          ADC4:  SCC           ;ADC DID NOT SET CC'S CORRECTLY
6115 020116 000245          +CLZ!CLC              ;CC=1010
6116 020120 005500          ADC     R0          ;CC=0100
6117 020122 102403          BVS     ADC5
6118 020124 103402          BCS     ADC5
6119 020126 100401          BMI     ADC5
6120 020130 001404          BEQ     TST217
6121
6122
6123
6124
6125 020132          ADC5:
6126 020132 012742 000465          MOV     #465,-(R2)      ;MOVE TO MAILBOX # ***** 465 *****
6127 020136 005242          INC     -(R2)          ;SET MSGTYP TO FATAL ERROR
6128 020140 000000          HALT           ;ADC DID NOT SET CC'S CORRECTLY
6129
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 754 <====
; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
; CONDITIONAL BRANCH INST. AND <====
; REPLACE THE MOVE INSTRUCTION <====
; WHICH FOLLOWS W/ 741 <====
; OR SEQUENCE ERROR
```

6130  
6131  
6132  
6133  
6134  
6135  
6136  
6137  
6138  
6139  
6140  
6141  
6142  
6143  
6144 020142 005212  
6145 020144 022712 000217  
6146 020150 001042  
6147 020152 012700 000001  
6148 020156 000277  
6149 020160 000251  
6150 020162 005400  
6151 020164 103003  
6152 020166 102402  
6153 020170 001401  
6154 020172 100404  
6155  
6156  
6157  
6158  
6159 020174  
6160 020174 012742 000466  
6161 020200 005242  
6162 020202 000000  
6163 020204 042700 077777  
6164 020210 000257  
6165 020212 000264  
6166 020214 005400  
6167 020216 102003  
6168 020220 103002  
6169 020222 001401  
6170 020224 100404  
6171  
6172  
6173  
6174  
6175 020226  
6176 020226 012742 000467  
6177 020232 005242  
6178 020234 000000  
6179 020236 005000  
6180 020240 000277  
6181 020242 000244  
6182 020244 005400  
6183 020246 102403  
6184 020250 103402  
6185 020252 001001

```
*****  
: THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE NEG,  
: 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.  
*****  
: TEST 217 TEST NEG INSTRUCTION  
*****  
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/ 767 <====  
  
NEG1: MOV #466,-(R2) ;MOVE TO MAILBOX # ***** 466 *****  
INC -(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 NEG3  
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/ 752 <====  
  
NEG3: MOV #467,-(R2) ;MOVE TO MAILBOX # ***** 467 *****  
INC -(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
```

```
6186 020254 100004          BPL      ST220
6187
6188                          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6189                          ;          CONDITIONAL BRANCH INST. AND <====
6190                          ;          REPLACE THE MOVE INSTRUCTION <====
6191                          ;          WHICH FOLLOWS W/ 736 <====
6191 020256
6192 020256 012742 000470    NEG5:    MOV      #470,-(R2)      ;MOVE TO MAILBOX # ***** 470 *****
6193 020262 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6194 020264 000000          HALT                    ;NEG DID NOT SET CC'S CORRECTLY
6195
6196
6197
6198
6199
6200 020266 005212          TST220: INC      (R2)          ;UPDATE TEST NUMBER
6201 020270 022712 000220    CMP      #220,(R2)      ;SEQUENCE ERROR?
6202 020274 001060          BNE     TST221-10      ;BR TO ERROR HALT ON SEQ ERROR
6203 020276 012700 000005    MOV      #5,R0
6204 020302 000257          CCC
6205 020304 000271          +SEN!SEC              ;CC=1010
6206 020306 022700 000005    CMP      #5,R0          ;CC=0101
6207 020312 101002          BHI     CMP1
6208 020314 102401          BVS     CMP1
6209 020316 100004          BPL     CMP2
6210
6211                          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6212                          ;          CONDITIONAL BRANCH INST. AND <====
6213                          ;          REPLACE THE MOVE INSTRUCTION <====
6214                          ;          WHICH FOLLOWS W/ 767 <====
6214 020320
6215 020320 012742 000471    CMP1:    MOV      #471,-(R2)      ;MOVE TO MAILBOX # ***** 471 *****
6216 020324 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6217 020326 000000          HALT                    ;CMP DID NOT SET CC'S CORRECTLY
6218 020330 012700 100000    CMP2:    MOV      #100000,R0
6219 020334 000277          SCC
6220 020336 000242          CLV
6221 020340 020027 077777    CMP      R0,#77777      ;CC=0010
6222 020344 101402          BLOS    CMP3
6223 020346 102001          BVC     CMP3
6224 020350 100004          BPL     CMP4
6225
6226                          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6227                          ;          CONDITIONAL BRANCH INST. AND <====
6228                          ;          REPLACE THE MOVE INSTRUCTION <====
6229                          ;          WHICH FOLLOWS W/ 752 <====
6229 020352
6230 020352 012742 000472    CMP3:    MOV      #472,-(R2)      ;MOVE TO MAILBOX # ***** 472 *****
6231 020356 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6232 020360 000000          HALT                    ;CMP DID NOT SET CC'S CORRECTLY
6233 020362 052700 040000    CMP4:    BIS      #40000,R0      ;R0=140000
6234 020366 000257          CCC
6235 020370 000264          SEZ
6236 020372 022700 040000    CMP      #40000,R0      ;CC=1011
6237 020376 102003          BVC     CMP5
6238 020400 103002          BCC     CMP5
6239 020402 001401          BEQ     CMP5
6240 020404 100404          BMI     CMP6
6241
                          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
```



6288  
6289  
6290  
6291  
6292  
6293  
6294  
6295  
6296  
6297  
6298  
6299  
6300  
6301  
6302 020506 005212  
6303 020510 022712 000222  
6304 020514 001055  
6305 020516 012700 125252  
6306 020522 000257  
6307 020524 000271  
6308 020526 162700 125252  
6309 020532 101002  
6310 020534 102401  
6311 020536 100004  
6312  
6313  
6314  
6315  
6316 020540  
6317 020540 012742 000476  
6318 020544 005242  
6319 020546 000000  
6320 020550 052700 100000  
6321 020554 000277  
6322 020556 000242  
6323 020560 162700 077777  
6324 020564 101402  
6325 020566 102001  
6326 020570 100004  
6327  
6328  
6329  
6330  
6331 020572  
6332 020572 012742 000477  
6333 020576 005242  
6334 020600 000000  
6335 020602 005100  
6336 020604 000277  
6337  
6338 020606 162700 100000  
6339 020612 101402  
6340 020614 102401  
6341 020616 100004  
6342  
6343

```
*****
:
: 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.
:
: *****
: TEST 222 TEST SUB INSTRUCTION
: *****
TST222: INC (R2) ;UPDATE TEST NUMBER
        CMP #222,(R2) ;SEQUENCE ERROR?
        BNE TST223-10 ;BR TO ERROR HALT ON SEQ ERROR
        MOV #125252,R0
        CCC ;CC=1010
        +SEN!SEC
        SUB #125252,R0 ;CC=0101 R0=0
        BHI SUB1
        BVS SUB1
        BPL SUB2
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 767 <====
:
SUB1: MOV #476,-(R2) ;MOVE TO MAILBOX # ***** 476 *****
      INC -(R2) ;SET MSGTYP TO FATAL ERROR
      HALT ;SUB DID NOT SET CC'S CORRECTLY
SUB2: BIS #100000,R0
      SCC ;CC=1101
      CLV
      SUB #77777,R0 ;CC=0010 R0=1
      BLOS SUB3
      BVC SUB3
      BPL SUB4
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 752 <====
:
SUB3: MOV #477,-(R2) ;MOVE TO MAILBOX # ***** 477 *****
      INC -(R2) ;SET MSGTYP TO FATAL ERROR
      HALT
SUB4: COM R0 ;R0=177777
      SCC ;CC=11111
:
SUB5: SUB #100000,R0 ;CC=0000 R0=77777
      BLOS SUB5
      BVS SUB5
      BPL SUB6
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
```



```
6344                                     :                               <====
6345                                     :                               <====
6346 020620                               SUB5:  MOV    #500,-(R2)      ;MOVE TO MAILBOX # ***** 500 *****
6347 020620 012742 000500                INC    -(R2)      ;SET MSGTYP TO FATAL ERROR
6348 020624 005242                        HALT                    ;SUB DID NOT SET CC'S CORRECTLY
6349 020626 000000                        SUB6:  CCC                    ;CC=0100
6350 020630 000257                        SEZ                    ;
6351 020632 000264                        SUB    #140000,R0     ;CC=1011
6352 020634 162700 140000                BVC    SUB7
6353 020640 102003                        BCC    SUB7
6354 020642 103002                        BEQ    SUB7
6355 020644 001401                        BMI    TST223
6356 020646 100404
6357                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6358                                     :                               <====
6359                                     :                               <====
6360                                     :                               <====
6361 020650                               SUB7:  MOV    #501,-(R2)      ;MOVE TO MAILBOX # ***** 501 *****
6362 020650 012742 000501                INC    -(R2)      ;SET MSGTYP TO FATAL ERROR
6363 020654 005242                        HALT                    ;
6364 020656 000000
6365
6366 :*****
6367 :TEST 223 TEST SBC INSTRUCTION
6368 :*****
6369 020660 005212                               TST223: INC    (R2)      ;UPDATE TEST NUMBER
6370 020662 022712 000223                CMP    #223,(R2)     ;SEQUENCE ERROR?
6371 020666 001053                        BNE    TST224-10     ;BR TO ERROR HALT ON SEQ ERROR
6372 020670 012700 000001                MOV    #1,R0
6373 020674 000277                        SCC                    ;CC=1011
6374 020676 000244                        CLZ
6375 020700 005600                        SBC    R0            ;CC=0100 R=0
6376 020702 103403                        BCS    SBC1
6377 020704 102402                        BVS    SBC1
6378 020706 100401                        BMI    SBC1
6379 020710 001404                        BEQ    SBC2
6380
6381                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6382                                     :                               <====
6383                                     :                               <====
6384                                     :                               <====
6385 020712                               SBC1:  MOV    #502,-(R2)      ;MOVE TO MAILBOX # ***** 502 *****
6386 020712 012742 000502                INC    -(R2)      ;SET MSGTYP TO FATAL ERROR
6387 020720 000000                        HALT                    ;SBC DID NOT SET CC'S CORRECTLY
6388 020722 000277                        SBC2:  SCC                    ;CC=1010
6389 020724 000245                        +CLZ!CLC
6390 020726 005600                        SBC    R0            ;CC=0100 R=0
6391 020730 103403                        BCS    SBC3
6392 020732 102402                        BVS    SBC3
6393 020734 100401                        BMI    SBC3
6394 020736 001404                        BEQ    SBC4
6395
6396                                     : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6397                                     :                               <====
6398                                     :                               <====
6399 020740                               SBC3:                                     :                               <====
```

6400	020740	012742	000503		MOV	#503,-(R2)	:MOVE TO MAILBOX # ***** 503 *****
6401	020744	005242			INC	-(R2)	:SET MSGTYP TO FATAL ERROR
6402	020746	000000			HALT		:SBC DID NOT SET CC'S CORRECTLY
6403	020750	000277		SBC4:	SCC		:CC=0111
6404	020752	000250			CLN		
6405	020754	005600			SBC	R0	:CC=1001 R0=177777
6406	020756	103003			BCC	SBC5	
6407	020760	102402			BVS	SBC5	
6408	020762	001401			BEQ	SBC5	
6409	020764	100404			BMI	SBC6	
6410							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6411							: CONDITIONAL BRANCH INST. AND <====
6412							: REPLACE THE MOVE INSTRUCTION <====
6413							: WHICH FOLLOWS W/ 741 <====
6414	020766			SBC5:			
6415	020766	012742	000504		MOV	#504,-(R2)	:MOVE TO MAILBOX # ***** 504 *****
6416	020772	005242			INC	-(R2)	:SET MSGTYP TO FATAL ERROR
6417	020774	000000			HALT		:SBC DID NOT SET CC'S CORRECTLY
6418	020776	042700	077777	SBC6:	BIC	#77777,R0	:R0=100000
6419	021002	000277			SCC		:CC=1101
6420	021004	000242			CLV		
6421	021006	005600			SBC	R0	:CC=0010
6422	021010	101402			BLOS	SBC7	
6423	021012	102001			BVC	SBC7	
6424	021014	100004			BPL	TST224	
6425							: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6426							: CONDITIONAL BRANCH INST. AND <====
6427							: REPLACE THE MOVE INSTRUCTION <====
6428							: WHICH FOLLOWS W/ 725 <====
6429	021016			SBC7:			
6430	021016	012742	000505		MOV	#505,-(R2)	:MOVE TO MAILBOX # ***** 505 *****
6431	021022	005242			INC	-(R2)	:SET MSGTYP TO FATAL ERROR
6432	021024	000000			HALT		:SBC DID NOT SET CC'S CORRECTLY
6433							: OR SEQUENCE ERROR
6434							

6435  
6436  
6437  
6438  
6439  
6440  
6441  
6442  
6443  
6444  
6445  
6446  
6447  
6448 021026 005212  
6449 021030 022712 000224  
6450 021034 001053  
6451 021036 012700 144000  
6452 021042 000257  
6453 021044 000266  
6454 021046 006100  
6455 021050 103003  
6456 021052 102402  
6457 021054 001401  
6458 021056 100404  
6459  
6460  
6461  
6462  
6463 021060  
6464 021060 012742 000506  
6465 021064 005242  
6466 021066 000000  
6467 021070 000277  
6468 021072 000243  
6469 021074 006100  
6470 021076 103003  
6471 021100 102002  
6472 021102 001401  
6473 021104 100004  
6474  
6475  
6476  
6477  
6478 021106  
6479 021106 012742 000507  
6480 021112 005242  
6481 021114 000000  
6482 021116 000277  
6483 021120 000250  
6484 021122 006100  
6485 021124 101402  
6486 021126 102401  
6487 021130 100004  
6488  
6489  
6490

\*\*\*\*\*  
: 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 COMMULATIVE DATA RESULT. THE DATA PATTERNS HAVE  
: BEEN SELECTED TO PRODUCE ALL COMBINATIONS OF THE C AND V BITS.  
\*\*\*\*\*

\*\*\*\*\*  
: TEST 224 TEST ROL INSTRUCTION  
\*\*\*\*\*

```
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/ 767 <=====  
:

```
ROL1: MOV #506,-(R2) ;MOVE TO MAILBOX # ***** 506 *****  
INC -(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/ 754 <=====  
:

```
ROL3: MOV #507,-(R2) ;MOVE TO MAILBOX # ***** 507 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT  
ROL4: SCC ;ROL DID NOT SET CC'S CORRECTLY  
CLN ;CC=0111  
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 <=====  
:

```
6491 ; WHICH FOLLOWS W/ 742 <====
6492 021132 ROL5: MOV #510,-(R2) ;MOVE TO MAILBOX # ***** 510 *****
6493 021132 012742 000510 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6494 021136 005242 HALT ;ROL DID NOT SET CC'S CORRECTLY
6495 021140 000000 ROL6: CCC ;CC=0101
6496 021142 000257 +SEZ!SEC ;CC=1010 R0=100003
6497 021144 000265 ROL R0
6498 021146 006100 BLOS ROL7
6499 021150 101405 BVC ROL7
6500 021152 102004 BPL ROL7
6501 021154 100003 CMP #100003,R0
6502 021156 022700 100003 BEQ TST225
6503 021162 001404 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6504 ; CONDITIONAL BRANCH INST. AND <====
6505 ; REPLACE THE MOVE INSTRUCTION <====
6506 ; WHICH FOLLOWS W/ 725 <====
6507
6508 021164 ROL7: MOV #511,-(R2) ;MOVE TO MAILBOX # ***** 511 *****
6509 021164 012742 000511 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6510 021170 005242 HALT ;ROL MALFUNCTIONED
6511 021172 000000 ; OR SEQUENCE ERROR
6512
6513 ;*****
6514 ;TEST 225 TEST ROR INSTRUCTION
6515 ;*****
6516 021174 005212 TST225: INC (R2) ;UPDATE TEST NUMBER
6517 021176 022712 000225 CMP #225,(R2) ;SEQUENCE ERROR?
6518 021202 001051 BNE TST226-10 ;BR TO ERROR HALT ON SEQ ERROR
6519 021204 012700 000023 MOV #23,R0 ;R0=23
6520 021210 000277 SCC ;CC=0111
6521 021212 000250 CLN
6522 021214 006000 ROR R0 ;CC=1001 R0=100011
6523 021216 102403 BVS ROR1
6524 021220 103002 BCC ROR1
6525 021222 001401 BEQ ROR1
6526 021224 100404 BMI ROR2
6527 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6528 ; CONDITIONAL BRANCH INST. AND <====
6529 ; REPLACE THE MOVE INSTRUCTION <====
6530 ; WHICH FOLLOWS W/ 767 <====
6531 021226 ROR1: MOV #512,-(R2) ;MOVE TO MAILBOX # ***** 512 *****
6532 021226 012742 000512 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6533 021232 005242 HALT ;ROR DID NOT SET CC'S CORRECTLY
6534 021234 000000 ROR2: CCC ;CC=1100
6535 021236 000257 +SEN!SEZ ;CC=0011 R0=040004
6536 021240 000274 ROR R0
6537 021242 006000 BVC ROR3
6538 021244 102003 BCC ROR3
6539 021246 103002 BEQ ROR3
6540 021250 001401 BPL ROR4
6541 021252 100004 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6542 ; CONDITIONAL BRANCH INST. AND <====
6543 ; REPLACE THE MOVE INSTRUCTION <====
6544 ; WHICH FOLLOWS W/ 754 <====
6545
6546 021254 ROR3:
```

```
6547 021254 012742 000513      MOV      #513,-(R2)      ;MOVE TO MAILBOX # ***** 513 *****
6548 021260 005242              INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6549 021262 000000              HALT                    ;ROR DID NOT SET CC'S CORRECTLY
6550 021264 000277      ROR4:   SCC            ;CC=1110
6551 021266 000241              CLC
6552 021270 006000              ROR      R0            ;CC=0000  R0=020002
6553 021272 101403              BLOS     ROR5
6554 021274 102402              BVS      ROR5
6555 021276 001401              BEQ      ROR5
6556 021300 100004              BPL      ROR6
6557                                ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6558                                ;          CONDITIONAL BRANCH INST. AND <====
6559                                ;          REPLACE THE MOVE INSTRUCTION <====
6560                                ;          WHICH FOLLOWS W/ 741 <====
6561 021302              ROR5:   MOV      #514,-(R2)      ;MOVE TO MAILBOX # ***** 514 *****
6562 021302 012742 000514      INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6563 021306 005242              HALT                    ;ROR DID NOT SET CC'S CORRECTLY
6564 021310 000000      ROR6:   CCC            ;CC=0101
6565 021312 000257              CCC
6566 021314 000265              +SEC!SEZ
6567 021316 006000              ROR      R0            ;CC=1010  R0=110001
6568 021320 101402              BLOS     ROR7
6569 021322 102001              BVC      ROR7
6570 021324 100404              BMI      TST226
6571                                ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6572                                ;          CONDITIONAL BRANCH INST. AND <====
6573                                ;          REPLACE THE MOVE INSTRUCTION <====
6574                                ;          WHICH FOLLOWS W/ 727 <====
6575 021326              ROR7:   MOV      #515,-(R2)      ;MOVE TO MAILBOX # ***** 515 *****
6576 021326 012742 000515      INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6577 021332 005242              HALT                    ;ROR DID NOT PRODUCE CORRECT RESULTS
6578 021334 000000              ; OR SEQUENCE ERROR
6579
6580      ;*****
6581      ;TEST 226      TEST ASL INSTRUCTION
6582      ;*****
6583 021336 005212      TST226: INC      (R2)          ;UPDATE TEST NUMBER
6584 021340 022712 000226      CMP      #226,(R2)      ;SEQUENCE ERROR?
6585 021344 001054              BNE      TST227-10      ;BR TO ERROR HALT ON SEQ ERROR
6586 021346 012700 144000      MOV      #144000,R0     ;R0=14000
6587 021352 000257              CCC            ;CC=0110
6588 021354 000271              +SEN!SEC
6589 021356 006300              ASL      R0            ;CC=1001  R0=110000
6590 021360 103003              BCC      ASL1
6591 021362 102402              BVS      ASL1
6592 021364 001401              BEQ      ASL1
6593 021366 100404              BMI      ASL2
6594                                ;
6595                                ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6596                                ;          CONDITIONAL BRANCH INST. AND <====
6597                                ;          REPLACE THE MOVE INSTRUCTION <====
6598                                ;          WHICH FOLLOWS W/ 767 <====
6599 021370              ASL1:   MOV      #516,-(R2)      ;MOVE TO MAILBOX # ***** 516 *****
6600 021374 005242              INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
6601 021376 000000              HALT
6602 021400 000277      ASL2:   SCC            ;CC=1100
```

```
6603 021402 000243 +CLV!CLC
6604 021404 006300 ASL R0 ;CC=0011 R0=020000
6605 021406 103003 BCC ASL3
6606 021410 102002 BVC ASL3
6607 021412 001401 BEQ ASL3
6608 021414 100004 BPL ASL4
6609
6610 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6611 ; CONDITIONAL BRANCH INST. AND <====
6612 ; REPLACE THE MOVE INSTRUCTION <====
6613 ; WHICH FOLLOWS W/ 754 <====
6614 021416 012742 000517 ASL3: MOV #517,-(R2) ;MOVE TO MAILBOX # ***** 517 *****
6615 021422 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6616 021424 000000 HALT ;ASL DID NOT SET CC'S CORRECTLY
6617 021426 000277 ASL4: SCC ;CC=0111
6618 021430 0C0250 CLN
6619 021432 006300 ASL R0 ;CC=0000 R0=040000
6620 021434 101402 BLOS ASL5
6621 021436 102401 BVS ASL5
6622 021440 100004 BPL ASL6
6623
6624 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6625 ; CONDITIONAL BRANCH INST. AND <====
6626 ; REPLACE THE MOVE INSTRUCTION <====
6627 ; WHICH FOLLOWS W/ 742 <====
6628 021442 012742 000520 ASL5: MOV #520,-(R2) ;MOVE TO MAILBOX # ***** 520 *****
6629 021446 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6630 021450 000000 HALT ;ASL DID NOT SET CC'S CORRECTLY
6631 021452 000257 ASL6: CCC ;CC=0101
6632 021454 000265 +SEZ!SEC
6633 021456 006300 ASL R0 ;CC=1010 R0=100000
6634 021460 103406 BCS ASL7
6635 021462 001405 BEQ ASL7
6636 021464 102004 BVC ASL7
6637 021466 100003 BPL ASL7
6638 021470 022700 100000 CMP #100000,R0
6639 021474 001404 BEQ TST227
6640
6641 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6642 ; CONDITIONAL BRANCH INST. AND <====
6643 ; REPLACE THE MOVE INSTRUCTION <====
6644 ; WHICH FOLLOWS W/ 724 <====
6645 021476 012742 000521 ASL7: MOV #521,-(R2) ;MOVE TO MAILBOX # ***** 521 *****
6646 021502 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6647 021504 000000 HALT ;ASL MALFUNCTIONED
6648 ; OR SEQUENCE ERROR
```

```
6649  
6650  
6651  
6652 021506 005212  
6653 021510 022712 000227  
6654 021514 001060  
6655 021516 012700 100023  
6656 021522 000277  
6657 021524 000250  
6658 021526 006200  
6659 021530 102403  
6660 021532 103002  
6661 021534 001401  
6662 021536 100404  
6663  
6664  
6665  
6666  
6667 021540  
6668 021540 012742 000522  
6669 021544 005242  
6670 021546 000000  
6671 021550 042700 100000  
6672 021554 000277  
6673 021556 000243  
6674 021560 006200  
6675 021562 102003  
6676 021564 103002  
6677 021566 001401  
6678 021570 100004  
6679  
6680  
6681  
6682  
6683 021572  
6684 021572 012742 000523  
6685 021576 005242  
6686 021600 000000  
6687 021602 000277  
6688  
6689 021604 006200  
6690 021606 101403  
6691 021610 102402  
6692 021612 001401  
6693 021614 100004  
6694  
6695  
6696  
6697  
6698 021616  
6699 021616 012742 000524  
6700 021622 005242  
6701 021624 000000  
6702 021626 052700 100000  
6703 021632 000257  
6704 021634 000265
```

```
*****  
:TEST 227 TEST ASR INSTRUCTION  
*****  
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/ 767 <====  
  
ASR1: MOV #522,-(R2) :MOVE TO MAILBOX # ***** 522 *****  
INC -(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/ 752 <====  
  
ASR3: MOV #523,-(R2) :MOVE TO MAILBOX # ***** 523 *****  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :ASR DID NOT SET CC'S CORRECTLY  
ASR4: SCC :CC=1111  
  
ASR 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/ 740 <====  
  
ASR5: MOV #524,-(R2) :MOVE TO MAILBOX # ***** 524 *****  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :ASR DID NOT SET CC'S CORRECTLY  
ASR6: BIS #100000,R0 :R0=110002  
CCC :CC=0101  
+SFZ!SEC
```

```

6705 021636 006200 ASR R0 ;C=1010 R0=144001
6706 021640 101406 BLOS ASR7
6707 021642 102005 BVC ASR7
6708 021644 100004 BPL ASR7
6709 021646 001403 BEQ ASR7
6710 021650 022700 144001 CMP #144001,R0 ;CHECK RESULT OF ASR'S
6711 021654 001404 BEQ TST230
6712 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6713 : CONDITIONAL BRANCH INST. AND <====
6714 : REPLACE THE MOVE INSTRUCTION <====
6715 : WHICH FOLLOWS W/ 720 <====
6716 021656 ASR7:
6717 021656 012742 000525 MOV #525,--(R2) ;MOVE TO MAILBOX # ***** 525 *****
6718 021662 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6719 021664 000000 HALT ;ASR DID NOT FUNCTION CORRECTLY
6720 ; OR SEQUENCE ERROR
6721
6722
6723
6724
6725
6726
6727
6728
6729
6730
6731
6732
6733
6734

```

```

:*****
: 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 ZEROES. THE DATA
: IS VERIFIED BY CONDITIONAL BRANCHES.
:*****

```

```

6735 021666 005212 TST230: INC (R2) ;UPDATE TEST NUMBER
6736 021670 022712 000230 CMP #230,(R2) ;SEQUENCE ERROR?
6737 021674 001033 BNE TST231-10 ;BR TO ERROR HALT ON SEQ ERROR
6738 021676 005000 CLR R0
6739 021700 000277 SCC ;SET CC=1011
6740 021702 000244 CLZ
6741 021704 006700 SXT R0 ;TRY SXT
6742 021706 100006 BPL SXT0 ;TEST CC=1001
6743 021710 001405 BEQ SXT0
6744 021712 102404 BVS SXT0
6745 021714 103003 BCC SXT0
6746 021716 022700 177777 CMP #-1,R0 ;CHECK DATA RESULT
6747 021722 001404 BEQ SXT1
6748 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6749 : CONDITIONAL BRANCH INST. AND <====
6750 : REPLACE THE MOVE INSTRUCTION <====
6751 : WHICH FOLLOWS W/ 765 <====
6752 021724 SXT0:
6753 021724 012742 000526 MOV #526,--(R2) ;MOVE TO MAILBOX # ***** 526 *****
6754 021730 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
6755 021732 000000 HALT ;RESULTS OF SXT INCORRECT
6756 021734 005000 SXT1: CLR R0 ;R0=0
6757 021736 005010 CLR (R0) ;LOC. 0=0
6758 021740 005110 COM (R0) ;LOC. 0=177777
6759 021742 000257 CCC ;SET CC=0110
6760 021744 000266 +SEZ:SEV

```



6761 021746 006710  
6762 021750 001005  
6763 021752 103404  
6764 021754 102403  
6765 021756 100402  
6766 021760 005710  
6767 021762 001404  
6768  
6769  
6770  
6771  
6772 021764  
6773 021764 012742 000527  
6774 021770 005242  
6775 021772 000000  
6776

SXT (R0)  
BNE SXT2  
BCS SXT2  
BVS SXT2  
BMI SXT2  
TST (R0)  
BEQ TST231

;TEST CC=0100

; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
; CONDITIONAL BRANCH INST. AND <====  
; REPLACE THE MOVE INSTRUCTION <====  
; WHICH FOLLOWS W/ 745 <====

SXT2:

MOV #527, -(R2)  
INC -(R2)  
HALT

;MOVE TO MAILBOX # \*\*\*\*\* 527 \*\*\*\*\*  
;SET MSGTYP TO FATAL ERROR  
;RESULTS OF SXT INCORRECT  
; OR SEQUENCE ERROR

6777  
6778  
6779  
6780  
6781  
6782  
6783  
6784  
6785  
6786  
6787  
6788 021774 005212  
6789 021776 022712 000231  
6790 022002 001035  
6791 022004 012700 007463  
6792 022010 012701 031525  
6793 022014 000277  
6794 022016 000241  
6795 022020 074100  
6796 022022 101406  
6797 022024 102405  
6798 022026 001404  
6799 022030 100403  
6800 022032 022700 036146  
6801 022036 001404  
6802  
6803  
6804  
6805  
6806 022040  
6807 022040 012742 000530  
6808 022044 005242  
6809 022046 000000  
6810 022050 010104  
6811 022052 000261  
6812 022054 000241  
6813 022056 074400  
6814 022060 101406  
6815 022062 102405  
6816 022064 001404  
6817 022066 100403  
6818 022070 022700 007463  
6819 022074 001404  
6820  
6821  
6822  
6823  
6824 022076  
6825 022076 012742 000531  
6826 022102 005242  
6827 022104 000000  
6828

```
*****
:
:   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.
:
: *****
: TEST 231      TEST THE XOR INSTRUCTION
: *****
TST231: INC      (R2)          ;UPDATE TEST NUMBER
        CMP      #231,(R2)    ;SEQUENCE ERROR?
        BNE     TST232-10    ;BR TO ERROR HALT ON SEQ ERROR
        MOV     #7463,R0     ;SET UP R0
        MOV     #31525,R1    ;SET UP R1
        SCC     ;SET CC=1110
        CLC
        XOR     R1,R0        ;TRY XOR
        BLOS   XOR1         ;CC=0000?
        BVS   XOR1
        BEQ   XOR1
        BMI   XOR1
        CMP   #36146,R0     ;DATA RESULT CORRECT?
        BEQ   XOR2
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:             CONDITIONAL BRANCH INST. AND <====
:             REPLACE THE MOVE INSTRUCTION <====
:             WHICH FOLLOWS W/ 762 <====
XOR1:  MOV     #530,-(R2)    ;MOVE TO MAILBOX # ***** 530 *****
        INC     -(R2)
        HALT
:
XOR2:  MOV     R1,R4
        SEC     ;CC=1110
        CLC
        XOR     R4,R0        ;TRY XOR MODE 0,0
        BLOS   XOR3         ;CC=0000?
        BVS   XOR3
        BEQ   XOR3
        BMI   XOR3
        CMP   #7463,R0
        BEQ   TST232
:
:   TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:             CONDITIONAL BRANCH INST. AND <====
:             REPLACE THE MOVE INSTRUCTION <====
:             WHICH FOLLOWS W/ 743 <====
XOR3:  MOV     #531,-(R2)    ;MOVE TO MAILBOX # ***** 531 *****
        INC     -(R2)
        HALT
: RESULT OF XOR INCORRECT
: OR SEQUENCE ERROR
```

6829  
6830  
6831  
6832  
6833  
6834  
6835  
6836  
6837  
6838  
6839 022106 005212  
6840 022110 022712 000232  
6841 022114 001023  
6842 022116 012700 000525  
6843 022122 010004  
6844 022124 000277  
6845 022126 101002  
6846 022130 100001  
6847 022132 102404  
6848  
6849  
6850  
6851  
6852 022134  
6853 022134 012742 000532  
6854 022140 005242  
6855 022142 000000  
6856 022144 005304  
6857 022146 000277  
6858 022150 077012  
6859 022152 101004  
6860 022154 100003  
6861 022156 102002  
6862 022160 005704  
6863 022162 001404  
6864  
6865  
6866  
6867  
6868 022164  
6869 022164 012742 000533  
6870 022170 005242  
6871 022172 000000  
6872

```
*****
: 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.
*****
: TEST 232      TEST SOB INSTRUCTION
*****
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/ 771 <====
SOB2:   MOV     #532,-(R2)    ;MOVE TO MAILBOX # ***** 532 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT
SOB3:   DEC     R4           ;COUNT ITERATIONS
        SCC          ;CC=1111
        SOB     R0,SOB1     ;DO SOB W/ R0
        BHI     SOB4        ;CHECK CC=1111
        BPL     SOB4
        BVC     SOB4
        TST     R4
        BEQ     TST233
: ITERATION COUNT OK?
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
:           CONDITIONAL BRANCH INST. AND <====
:           REPLACE THE MOVE INSTRUCTION <====
:           WHICH FOLLOWS W/ 755 <====
SOB4:   MOV     #533,-(R2)    ;MOVE TO MAILBOX # ***** 533 *****
        INC     -(R2)        ;SET MSGTYP TO FATAL ERROR
        HALT                ;INCORRECT # OF BRANCHES OR CC'S CHANGED
: OR SEQUENCE ERROR
```

6873  
6874  
6875  
6876  
6877  
6878  
6879  
6880  
6881  
6882  
6883 022174 005212  
6884 022176 022712 000233  
6885 022202 001062  
6886 022204 012706 000500  
6887 022210 012746 125252  
6888 022214 162706 000074  
6889 022220 012705 022246  
6890 022224 012746 006436  
6891 022230 000277  
6892 022232 000137 000400  
6893 022236 012742 000534  
6894 022242 005242  
6895 022244 000000  
6896 022246 101010  
6897 022250 100007  
6898 022252 102006  
6899 022254 020527 125252  
6900 022260 001003  
6901 022262 022706 000500  
6902 022266 001404  
6903  
6904  
6905  
6906  
6907 022270  
6908 022270 012742 000535  
6909 022274 005242  
6910 022276 000000  
6911 022300 012746 052525  
6912 022304 012746 006400  
6913 022310 010605  
6914 022312 004737 022322  
6915 022316 000137 022334  
6916 022322 000205  
6917 022324 012742 000536  
6918 022330 005242  
6919 022332 000000  
6920 022334 022706 000500  
6921 022340 001003  
6922 022342 022705 052525  
6923 022346 001404  
6924  
6925  
6926  
6927  
6928 022350

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

TEST 233 TEST MARK INSTRUCTION  
\*\*\*\*\*

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 @#400 ;XFER CONTL TO MARK 36 INST. ON STACK  
MOV #534,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 534 \*\*\*\*\*  
INC -(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/ 746 <====  
MRK2: MOV #535,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 535 \*\*\*\*\*  
INC -(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,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 536 \*\*\*\*\*  
INC -(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/ 716 <====  
MRK6:

CFKAACO 11/34 BSC INST TST  
CFKAAC.P11 18-OCT-78 11:01

MACY11 30A(1052) 18-OCT-78 11:06 PAGE 151  
T233 TEST MARK INSTRUCTION

H 13

SEQ 0163

6929 022350 012742 000537  
6930 022354 005242  
6931 022356 000000  
6932

MOV #537, -(R2)  
INC -(R2)  
HALT

:MOVE TO MAILBOX # \*\*\*\*\* 537 \*\*\*\*\*  
:SET MSGTYP TO FATAL ERROR  
:RESULTS OF MARK INCORRECT  
: OR SEQUENCE ERROR

6933 177776  
6934  
6935  
6936  
6937  
6938  
6939  
6940  
6941  
6942  
6943  
6944  
6945  
6946 022360 005212  
6947 022362 022712 000234  
6948 022366 001024  
6949 022370 012700 000377  
6950 022374 000257  
6951 022376 106400  
6952 022400 022767 000357 155370  
6953 022406 001404  
6954  
6955  
6956  
6957  
6958 022410 012742 000540  
6959 022414 005242  
6960 022416 000000  
6961 022420 005000  
6962 022422 005010  
6963 022424 000277  
6964 022426 106410  
6965 022430 100403  
6966 022432 102402  
6967 022434 103401  
6968 022436 001004  
6969  
6970  
6971  
6972  
6973 022440  
6974 022440 012742 000541  
6975 022444 005242  
6976 022446 000000  
6977  
6978  
6979  
6980  
6981  
6982 022450 005212  
6983 022452 022712 000235  
6984 022456 001021  
6985 022460 005000  
6986 022462 012710 177777  
6987 022466 005037 177776  
6988 022472 106420

PS=177776

\*\*\*\*\*  
THESE NEXT SEVEN TESTS VERIFY THE MTPS INSTRUCTION IN ALL  
MODES. THE PSW IS DEFINED BY AN EQUATE STATEMENT BEFORE THE  
FIRST MTPS TEST. IN EACH TEST A PATTERN OF ONES AND  
ZEROS IS SET IN A DATA REGISTER AND MOVED TO THE PSW.  
THE DATA IN THE PSW, AND THE DATA REGISTER ADDRESS,  
ARE CHECKED TO VERIFY PROPER EXECUTION OF THE INSTRUCTION.  
\*\*\*\*\*

TEST 234 TEST MTPS INSTRUCTION

\*\*\*\*\*  
TST234: INC (R2) :UPDATE TEST NUMBER  
CMP #234,(R2) :SEQUENCE ERROR?  
BNE TST235-10 :BR TO ERROR HALT ON SEQ ERROR  
MOV #377,R0  
CCC  
MTPS R0  
CMP #357,PS  
BEQ MTPS1  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 770 <=====  
MOV #540,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 540 \*\*\*\*\*  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :MTPS FAILED  
MTPS1: CLR R0  
CLR (R0)  
SCC :CC=1111  
MTPS (R0) :TRY MTPS MODE 1  
BMI MTPS1A :CHECK PS  
BVS MTPS1A  
BCS MTPS1A  
BNE TST235  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====  
: CONDITIONAL BRANCH INST. AND <=====  
: REPLACE THE MOVE INSTRUCTION <=====  
: WHICH FOLLOWS W/ 754 <=====  
MTPS1A: MOV #541,-(R2) :MOVE TO MAILBOX # \*\*\*\*\* 541 \*\*\*\*\*  
INC -(R2) :SET MSGTYP TO FATAL ERROR  
HALT :MTPS FAILED  
: OR SEQUENCE ERROR

TEST 235 TEST MTPS MODE 2

\*\*\*\*\*  
TST235: INC (R2) :UPDATE TEST NUMBER  
CMP #235,(R2) :SEQUENCE ERROR?  
BNE TST236-10 :BR TO ERROR HALT ON SEQ ERROR  
CLR R0 :R0=0  
MOV #-1,(R0) :LOC. 0=-1  
CLR @PS :PS=0  
MTPS (R0) :TRY MTPS W/MODE 2

```
6989 022474 022737 000357 177776      CMP      #357,@#PS      ;CHECK DATA
6990 022502 001404                      BEQ      MTPS2
6991                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
6992                                     ;         CONDITIONAL BRANCH INST. AND <====
6993                                     ;         REPLACE THE MOVE INSTRUCTION <====
6994                                     ;         WHICH FOLLOWS W/ 766 <====
6995 022504 012742 000542      MOV      #542,-(R2)    ;MOVE TO MAILBOX # ***** 542 *****
6996 022510 005242      INC      -(R2)      ;SET MSGTYP TO FATAL ERROR
6997 022512 000000      HALT
6998 022514 022700 000001      MTPS2:  CMP      #1,R0  ;DEST. DATA INCORRECT
6999 022520 001404      BEQ      TST236     ;CHECK DEST. REGISTER.
7000                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7001                                     ;         CONDITIONAL BRANCH INST. AND <====
7002                                     ;         REPLACE THE MOVE INSTRUCTION <====
7003                                     ;         WHICH FOLLOWS W/ 757 <====
7004 022522 012742 000543      MOV      #543,-(R2)    ;MOVE TO MAILBOX # ***** 543 *****
7005 022526 005242      INC      -(R2)      ;SET MSGTYP TO FATAL ERROR
7006 022530 000000      HALT
7007                                     ;DEST REGISTER NOT INCREMENTED BY 1
7008                                     ; OR SEQUENCE ERROR
```

```
*****
:TEST 236      TEST MTPS MODE 3
*****
```

```
7011 TST236: INC      (R2)      ;UPDATE TEST NUMBER
7012 022532 005212      CMP      #236,(R2)    ;SEQUENCE ERROR?
7013 022534 022712 000236      BNE      TST237-10   ;BR TO ERROR HALT ON SEQ ERROR
7014 022540 001024      MOV      #402,R0     ;R0=402
7015 022542 012700 000402      CLR      (R0)       ;LOC. 402=0
7016 022546 005010      MOV      #52652,@#0 ;LOC. 0=52652
7017 022550 012737 052652 000000      CLR      @#PS       ;PS=0
7018 022556 005037 177776      MTPS    @(R0)+      ;TRY MTPS W/MODE 3
7019 022562 106430      CMP      #252,@#PS  ;CHECK DEST. DATA
7020 022564 022737 000252 177776      BEQ      MTPS3
7021 022572 001404
7022                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7023                                     ;         CONDITIONAL BRANCH INST. AND <====
7024                                     ;         REPLACE THE MOVE INSTRUCTION <====
7025                                     ;         WHICH FOLLOWS W/ 763 <====
7026 022574 012742 000544      MOV      #544,-(R2)    ;MOVE TO MAILBOX # ***** 544 *****
7027 022600 005242      INC      -(R2)      ;SET MSGTYP TO FATAL ERROR
7028 022602 000000      HALT
7029 022604 022700 000404      MTPS3:  CMP      #404,RC ;CHECK MODE 3 REGISTER.
7030 022610 001404      BEQ      TST237
7031                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7032                                     ;         CONDITIONAL BRANCH INST. AND <====
7033                                     ;         REPLACE THE MOVE INSTRUCTION <====
7034                                     ;         WHICH FOLLOWS W/ 754 <====
7035 022612 012742 000545      MOV      #545,-(R2)    ;MOVE TO MAILBOX # ***** 545 *****
7036 022616 005242      INC      -(R2)      ;SET MSGTYP TO FATAL ERROR
7037 022620 000000      HALT
7038                                     ;MODE 3 REGISTER INCORRECT
7039                                     ; OR SEQUENCE ERROR
```

```
*****
:TEST 237      TEST MTPS MODE 4
*****
7041 TST237: INC      (R2)      ;UPDATE TEST NUMBER
7042 022622 005212      CMP      #237,(R2)    ;SEQUENCE ERROR?
7043 022624 022712 000237
```

```
7045 022630 001022          BNE    TST240-10      ;BR TO ERROR HALT ON SEQ ERROR
7046 022632 012700 000001    MOV    #1,R0          ;R0=1
7047 022636 012737 125125 000000    MOV    #125125,@#0   ;LOC. 0 = 125125
7048 022644 005037 177776          CLR    @#PS           ;PS=0
7049 022650 106440          MTPS  -(R0)          ;TRY MTPS W/MODE 4
7050 022652 022737 000105 177776    CMP    #105,@#PS     ;CHECK DEST. DATA
7051 022660 001404          BEQ    MTPS4
7052                               ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7053                               ;          CONDITIONAL BRANCH INST. AND <=====
7054                               ;          REPLACE THE MOVE INSTRUCTION <=====
7055                               ;          WHICH FOLLOWS W/ 764 <=====
7056 022662 012742 000546          MOV    #546,-(R2)    ;MOVE TO MAILBOX # ***** 546 *****
7057 022666 005242          INC    -(R2)         ;SET MSGTYP TO FATAL ERROR
7058 022670 000000          HALT                    ;DEST. DATA INCORRECT
7059 022672 005700          MTPS4: TST    R0      ;CHECK MODE 4 REGISTER
7060 022674 001404          BEQ    TST240
7061                               ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7062                               ;          CONDITIONAL BRANCH INST. AND <=====
7063                               ;          REPLACE THE MOVE INSTRUCTION <=====
7064                               ;          WHICH FOLLOWS W/ 756 <=====
7065 022676 012742 000547          MOV    #547,-(R2)    ;MOVE TO MAILBOX # ***** 547 *****
7066 022702 005242          INC    -(R2)         ;SET MSGTYP TO FATAL ERROR
7067 022704 000000          HALT                    ;MODE 4 REGISTER NOT DECREMENTED BY 1
7068                               ; OR SEQUENCE ERROR
7069
```

```
7070 :*****
7071 :TEST 240      TEST MTPS MODE 5
7072 :*****
```

```
7073 022706 005212          TST240: INC    (R2)      ;UPDATE TEST NUMBER
7074 022710 022712 000240    CMP    #240,(R2)     ;SEQUENCE ERROR?
7075 022714 001021          BNE    TST241-10     ;BR TO ERROR HALT ON SEQ ERROR
7076 022716 012700 000404    MOV    #404,R0       ;R0=404
7077 022722 012737 177400 000000    MOV    #177400,@#0   ;LOC. 0=177400
7078 022730 000277          SCC                    ;SET ALL COND. CODES
7079 022732 106450          MTPS  @-(R0)        ;TRY MTPS W/MODE 5
7080 022734 005737 177776    TST    @#PS          ;CHECK DEST. DATA.
7081 022740 001404          BEQ    MTPS5
7082                               ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7083                               ;          CONDITIONAL BRANCH INST. AND <=====
7084                               ;          REPLACE THE MOVE INSTRUCTION <=====
7085                               ;          WHICH FOLLOWS W/ 766 <=====
7086 022742 012742 000550          MOV    #550,-(R2)    ;MOVE TO MAILBOX # ***** 550 *****
7087 022746 005242          INC    -(R2)         ;SET MSGTYP TO FATAL ERROR
7088 022750 000000          HALT                    ;DESTINATION DATA INCORRECT
7089 022752 022700 000402    MTPS5: CMP    #402,R0  ;CHECK MODE 5 REGISTER
7090 022756 001404          BEQ    TST241
7091                               ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7092                               ;          CONDITIONAL BRANCH INST. AND <=====
7093                               ;          REPLACE THE MOVE INSTRUCTION <=====
7094                               ;          WHICH FOLLOWS W/ 757 <=====
7095 022760 012742 000551          MOV    #551,-(R2)    ;MOVE TO MAILBOX # ***** 551 *****
7096 022764 005242          INC    -(R2)         ;SET MSGTYP TO FATAL ERROR
7097 022766 000000          HALT                    ;MODE 5 REGISTER NOT DECREMENTED BY 2
7098                               ; OR SEQUENCE ERROR
7099
```

```
7100 :*****
```



```
7101 ;TEST 241 TEST MTPS MODE 6
7102 :*****
7103 TST241: INC (R2) ;UPDATE TEST NUMBER
7104 022770 005212 000241 CMP #241,(R2) ;SEQUENCE ERROR?
7105 022776 001024 BNE TST242-10 ;BR TO ERROR HALT ON SEQ ERROR
7106 023000 012737 052652 000000 MOV #52652,@#0 ;LOC. 0=52652
7107 023006 012700 000406 MOV #406,R0 ;R0=406
7108 023012 005037 177776 CLR @#PS ;PS=0
7109 023016 106460 177372 MTPS -406(R0) ;TRY MTPS W/MODE 6
7110 023022 022737 000252 177776 CMP #252,@#PS ;CHECK DEST. DATA
7111 023030 001404 BEQ MTPS6
7112 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7113 ; CONDITIONAL BRANCH INST. AND <====
7114 ; REPLACE THE MOVE INSTRUCTION <====
7115 ; WHICH FOLLOWS W/ 763 <====
7116 023032 012742 000552 MOV #552,-(R2) ;MOVE TO MAILBOX # ***** 552 *****
7117 023036 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
7118 023040 000000 HALT ;DEST. DATA INCORRECT
7119 023042 022700 000406 MTPS6: CMP #406,R0 ;CHECK MODE 6 REGISTER
7120 023046 001404 BEQ TST242
7121 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7122 ; CONDITIONAL BRANCH INST. AND <====
7123 ; REPLACE THE MOVE INSTRUCTION <====
7124 ; WHICH FOLLOWS W/ 754 <====
7125 023050 012742 000553 MOV #553,-(R2) ;MOVE TO MAILBOX # ***** 553 *****
7126 023054 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
7127 023056 000000 HALT ;MODE 6 REGISTER MODIFIED
7128 ; OR SEQUENCE ERROR
7129
7130 :*****
7131 ;TEST 242 TEST MTPS MODE 7
7132 :*****
7133 TST242: INC (R2) ;UPDATE TEST NUMBER
7134 023060 005212 000242 CMP #242,(R2) ;SEQUENCE ERROR?
7135 023066 001024 BNE TST243-10 ;BR TO ERROR HALT ON SEQ ERROR
7136 023070 012737 052652 000000 MOV #52652,@#0 ;LOC. 0=52652
7137 023076 012700 000410 MOV #410,R0 ;R0=410
7138 023102 005037 177776 CLR @#PS ;PS=0
7139 023106 106470 177776 MTPS @-2(R0) ;TRY MTPS W/MODE 7
7140 023112 022737 000105 177776 CMP #105,@#PS ;CHECK DEST. DATA
7141 023120 001404 BEQ MTPS7
7142 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7143 ; CONDITIONAL BRANCH INST. AND <====
7144 ; REPLACE THE MOVE INSTRUCTION <====
7145 ; WHICH FOLLOWS W/ 763 <====
7146 023122 012742 000554 MOV #554,-(R2) ;MOVE TO MAILBOX # ***** 554 *****
7147 023126 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
7148 023130 000000 HALT ;DESTINATION DATA INCORRECT
7149 023132 022700 000410 MTPS7: CMP #410,R0 ;CHECK MODE 7 REGISTER
7150 023136 001404 BEQ TST243
7151 ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7152 ; CONDITIONAL BRANCH INST. AND <====
7153 ; REPLACE THE MOVE INSTRUCTION <====
7154 ; WHICH FOLLOWS W/ 754 <====
7155 023140 012742 000555 MOV #555,-(R2) ;MOVE TO MAILBOX # ***** 555 *****
7156 023144 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR
```

CFKAACO 11/34 BSC INST TST  
CFKAAC.P11 18-OCT-78 11:01

MACY11 30A(1052) 18-OCT-78 11:06 M 13 PAGE 156  
T242 TEST MTPS MODE 7

SEQ 0168

7157 023146 000000  
7158  
7159

HALT

:MODE 7 REGISTER MODIFIED  
: OR SEQUENCE ERROR

7160  
7161  
7162  
7163  
7164  
7165  
7166  
7167  
7168  
7169  
7170  
7171 023150 005212  
7172 023152 022712 000243  
7173 023156 001025  
7174 023160 012737 000377 177776  
7175 023166 106700  
7176 023170 022700 177757  
7177 023174 001404  
7178  
7179  
7180  
7181  
7182 023176 012742 000556  
7183 023202 005242  
7184 023204 000000  
7185  
7186 023206 005000  
7187 023210 012737 177777 000000  
7188 023216 005037 177776  
7189 023222 106710  
7190 023224 105737 000000  
7191 023230 001404  
7192  
7193  
7194  
7195  
7196 023232 012742 000557  
7197 023236 005242  
7198 023240 000000  
7199  
7200  
7201  
7202  
7203  
7204 023242 005212  
7205 023244 022712 000244  
7206 023250 001031  
7207 023252 005000  
7208 023254 005010  
7209 023256 012737 000377 177776  
7210 023264 106720  
7211 023266 103003  
7212 023270 102402  
7213 023272 001401  
7214 023274 100404  
7215

```
*****  
: THESE NEXT SEVEN TESTS VERIFY THE MFPS INSTRUCTION IN ALL  
: MODES. IN EACH TEST, A PATTERN OF ONES AND ZEROES IS MOVED TO THE  
: PSW, AND AN MFPS INSTRUCTION MOVES THE DATA TO A LOCATION SETUP  
: BY R0, EITHER DIRECTLY OR INDIRECTLY. CONDITIONAL BRANCHES ARE  
: USED TO CHECK PROPER ADDRESSING AND DATA.  
:*****  
: TEST 243 TEST MFPS INSTRUCTION  
:*****  
TST243: INC (R2) ;UPDATE TEST NUMBER  
CMP #243,(R2) ;SEQUENCE ERROR?  
BNE TST244-10 ;BR TO ERROR HALT ON SEQ ERROR  
MOV #377,@#PS  
MFPS R0  
CMP #177757,R0  
BEQ MFPS1  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 771 <====  
MOV #556,-(R2) ;MOVE TO MAILBOX # ***** 556 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;MFPS FAILED  
MFPS1: CLR R0  
MOV #-1,@#0  
CLR @#PS  
MFPS (R0)  
TSTB @#0  
BEQ TST244  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 753 <====  
MOV #557,-(R2) ;MOVE TO MAILBOX # ***** 557 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;MFPS FAILED  
: OR SEQUENCE ERROR  
:*****  
: TEST 244 TEST MFPS MODE 2  
:*****  
TST244: INC (R2) ;UPDATE TEST NUMBER  
CMP #244,(R2) ;SEQUENCE ERROR?  
BNE TST245-10 ;BR TO ERROR HALT ON SEQ ERROR  
CLR R0 ;R0=0  
CLR (R0) ;LOC. 0=0  
MOV #377,@#PS ;SET PS=357  
MFPS (R0)+ ;TRY MFPS W/MODE 2  
BCC MFPS2A ;BR TO ERROR IF C BIT CLEAR  
BVS MFPS2A ;BR TO ERROR IF V BIT SET  
BEQ MFPS2A ;BR TO ERROR IF Z BIT SET  
BMI MFPS2B  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
```

```
7216 : CONDITIONAL BRANCH INST. AND <====  
7217 : REPLACE THE MOVE INSTRUCTION <====  
7218 : WHICH FOLLOWS W/ 766 <====  
7219 023276 MFPS2A: MOV #560,-(R2) ;MOVE TO MAILBOX # ***** 560 *****  
7220 023276 012742 000560 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7221 023302 005242 HALT ;COND. CODES INCORRECT  
7222 023304 000000 MFPS2B: CMP #357,@#0 ;CHECK DEST. DATA  
7223 023306 022737 000357 000000 BEQ MFPS2C  
7224 023314 001404  
7225 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
7226 : CONDITIONAL BRANCH INST. AND <====  
7227 : REPLACE THE MOVE INSTRUCTION <====  
7228 : WHICH FOLLOWS W/ 756 <====  
7229 023316 012742 000561 MOV #561,-(R2) ;MOVE TO MAILBOX # ***** 561 *****  
7230 023322 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7231 023324 000000 HALT ;DEST. DATA INCORRECT  
7232 023326 022700 000001 MFPS2C: CMP #1,R0 ;CHECK MODE Z REGISTER  
7233 023332 001404 BEQ TST245  
7234 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
7235 : CONDITIONAL BRANCH INST. AND <====  
7236 : REPLACE THE MOVE INSTRUCTION <====  
7237 : WHICH FOLLOWS W/ 747 <====  
7238 023334 012742 000562 MOV #562,-(R2) ;MOVE TO MAILBOX # ***** 562 *****  
7239 023340 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7240 023342 000000 HALT ;MODE 2 REGISTER NOT INCREMENTED 1  
7241 : OR SEQUENCE ERROR  
7242  
7243 :*****  
7244 :TEST 245 TEST MFPS MODE 3  
7245 :*****  
7246 023344 005212 TST245: INC (R2) ;UPDATE TEST NUMBER  
7247 023346 022712 000245 CMP #245,(R2) ;SEQUENCE ERROR?  
7248 023352 001033 BNE TST246-10 ;BR TO ERROR HALT ON SEQ ERROR  
7249 023354 012700 000406 MOV #406,R0 ;R0=406  
7250 023360 005037 000000 CLR @#0 ;LOC. 0=0  
7251 023364 012737 000252 177776 MOV #252,@#PS ;PS=252  
7252 023372 106730 MFPS @ (R0)+ ;TRY MFPS WITH MODE 3  
7253 023374 103403 BCS MFPS3A ;BR TO ERROR IF C-BIT SET  
7254 023376 102402 BVS MFPS3A ;BR TO ERROR IF V-BIT SET  
7255 023400 001401 BEQ MFPS3A ;BR TO ERROR IF Z-BIT SET  
7256 023402 100404 BMI MFPS3B  
7257 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
7258 : CONDITIONAL BRANCH INST. AND <====  
7259 : REPLACE THE MOVE INSTRUCTION <====  
7260 : WHICH FOLLOWS W/ 764 <====  
7261 023404 MFPS3A: MOV #563,-(R2) ;MOVE TO MAILBOX # ***** 563 *****  
7262 023404 012742 000563 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7263 023410 005242 HALT ;CONDITION CODES INCORRECT  
7264 023412 000000 MFPS3B: CMP #125000,@#0 ;CHECK DEST. DATA  
7265 023414 022737 125000 000000 BEQ MFPS3C  
7266 023422 001404  
7267 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
7268 : CONDITIONAL BRANCH INST. AND <====  
7269 : REPLACE THE MOVE INSTRUCTION <====  
7270 : WHICH FOLLOWS W/ 754 <====  
7271 023424 012742 000564 MOV #564,-(R2) ;MOVE TO MAILBOX # ***** 564 *****
```



```
7328 :TEST 247 TEST MFPS MODE 5
7329 :*****
7330 023560 005212 :TST247: INC (R2) :UPDATE TEST NUMBER
7331 023562 022712 000247 :CMP #247,(R2) :SEQUENCE ERROR?
7332 023566 001033 :BNE TST250-10 :BR TO ERROR HALT ON SEQ ERROR
7333 023570 012700 000410 :MOV #410,R0 :R0=410
7334 023574 012737 177777 000000 :MOV #-1,@#0 :LOC. 0=-1
7335 023602 005037 177776 :CLR @#PS :PS=0
7336 023606 106750 :MFPS @-(R0) :TRY MFPS W/MODE 5
7337 023610 103403 :BCS MFPS5A :BR TO ERROR IF C-BIT SET
7338 023612 102402 :BVS MFPS5A :BR TO ERROR IF V-BIT SET
7339 023614 100401 :BMI MFPS5A :BR TO ERROR IF N-BIT SET
7340 023616 001404 :BEQ MFPS5B
7341 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7342 : CONDITIONAL BRANCH INST. AND <====
7343 : REPLACE THE MOVE INSTRUCTION <====
7344 : WHICH FOLLOWS W/ 764 <====
7345 023620 MFPS5A:
7346 023620 012742 000571 :MOV #571,-(R2) :MOVE TO MAILBOX # ***** 571 *****
7347 023624 005242 :INC -(R2) :SET MSGTYP TO FATAL ERROR
7348 023626 000000 :HALT :COND. CODES INCORRECT
7349 023630 022737 000377 000000 MFPS5B: :CMP #377,@#0 :CHECK DEST. DATA
7350 023636 001404 :BEQ MFPS5C
7351 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7352 : CONDITIONAL BRANCH INST. AND <====
7353 : REPLACE THE MOVE INSTRUCTION <====
7354 : WHICH FOLLOWS W/ 754 <====
7355 023640 012742 000572 :MOV #572,-(R2) :MOVE TO MAILBOX # ***** 572 *****
7356 023644 005242 :INC -(R2) :SET MSGTYP TO FATAL ERROR
7357 023646 000000 :HALT :DEST DATA INCORRECT
7358 023650 020027 000406 MFPS5C: :CMP R0,#406 :CHECK MODE 5 REGISTER
7359 023654 001404 :BEQ TST250
7360 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7361 : CONDITIONAL BRANCH INST. AND <====
7362 : REPLACE THE MOVE INSTRUCTION <====
7363 : WHICH FOLLOWS W/ 745 <====
7364 023656 012742 000573 :MOV #573,-(R2) :MOVE TO MAILBOX # ***** 573 *****
7365 023662 005242 :INC -(R2) :SET MSGTYP TO FATAL ERROR
7366 023664 000000 :HALT :MODE 5 REGISTER NOT DECREMENTED BY 2
7367 : OR SEQUENCE ERROR
7368 :
7369 :*****
7370 :TEST 250 TEST MFPS MODE 6
7371 :*****
7372 023666 005212 :TST250: INC (R2) :UPDATE TEST NUMBER
7373 023670 022712 000250 :CMP #250,(R2) :SEQUENCE ERROR?
7374 023674 001034 :BNE TST251-10 :BR TO ERROR HALT ON SEQ ERROR
7375 023676 012700 000401 :MOV #401,R0 :R0=410
7376 023702 005037 000000 :CLR @#C :LOC. 0=0
7377 023706 012737 000252 177776 :MOV #252,@#PS :PS=252
7378 023714 106760 177377 :MFPS -401(R0) :TRY MFPS W/MODE 6
7379 023720 102403 :BVS MFPS6A :BR TO ERROR IF V-BIT SET
7380 023722 103402 :BCS MFPS6A :BR TO ERROR IF C-BIT SET
7381 023724 001401 :BEQ MFPS6A :BR TO ERROR IF Z-BIT SET
7382 023726 100404 :BMI MFPS6B
7383 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
```

```

7384                                     :          CONDITIONAL BRANCH INST. AND <=====
7385                                     :          REPLACE THE MOVE INSTRUCTION <=====
7386                                     :          WHICH FOLLOWS W/ 763 <=====
7387 023730                               MFPS6A:
7388 023730 012742 000574                 MOV    #574,-(R2)      ;MOVE TO MAILBOX # ***** 574 *****
7389 023734 005242                         INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
7390 023736 000000                         HALT                                     ;COND. CODES INCORRECT
7391 023740 022737 000252 000000 MFPS6B: CMP    #252,@#0      ;CHECK DEST. DATA
7392 023746 001404                         BEQ    MFPS6C
7393                                     :          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7394                                     :          CONDITIONAL BRANCH INST. AND <=====
7395                                     :          REPLACE THE MOVE INSTRUCTION <=====
7396                                     :          WHICH FOLLOWS W/ 753 <=====
7397 023750 012742 000575                 MOV    #575,-(R2)      ;MOVE TO MAILBOX # ***** 575 *****
7398 023754 005242                         INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
7399 023756 000000                         HALT                                     ;DEST. DATA INCORRECT
7400 023760 022700 000401                 MFPS6C: CMP    #401,R0  ;CHECK DEST. REGISTER
7401 023764 001404                         BEQ    TST251
7402                                     :          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7403                                     :          CONDITIONAL BRANCH INST. AND <=====
7404                                     :          REPLACE THE MOVE INSTRUCTION <=====
7405                                     :          WHICH FOLLOWS W/ 744 <=====
7406 023766 012742 000576                 MOV    #576,-(R2)      ;MOVE TO MAILBOX # ***** 576 *****
7407 023772 005242                         INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
7408 023774 000000                         HALT                                     ;DEST. DATA INCORRECT
7409                                     :          ; OR SEQUENCE ERROR
7410
7411
7412 :*****
7413 :TEST 251 TEST MFPS MODE 7
7414 :*****
7414 023776 005212 TST251: INC    (R2)      ;UPDATE TEST NUMBER
7415 024000 022712 000251                 CMP    #251,(R2)      ;SEQUENCE ERROR?
7416 024004 001034                         BNE    TST252-10      ;BR TO ERROR HALT ON SEQ ERROR
7417 024006 012700 000777                 MOV    #777,R0        ;R0=777
7418 024012 005037 000000                 CLR    @#0            ;LOC. 0=0
7419 024016 012737 000125 177776         MOV    #125,@#PS      ;PS=125
7420 024024 106770 177407                 MFPS   @-371(R0)      ;TRY MFPS W/MODE 7
7421 024030 102403                         BVS    MFPS7A         ;BR TO ERROR IF V-BIT SET
7422 024032 103002                         BCC    MFPS7A         ;BR TO ERROR IF C-BIT SET
7423 024034 001401                         BEQ    MFPS7A         ;BR TO ERROR IF Z-BIT SET
7424 024036 100004                         BPL    MFPS7B
7425                                     :          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7426                                     :          CONDITIONAL BRANCH INST. AND <=====
7427                                     :          REPLACE THE MOVE INSTRUCTION <=====
7428                                     :          WHICH FOLLOWS W/ 763 <=====
7429 024040                               MFPS7A:
7430 024040 012742 000577                 MOV    #577,-(R2)      ;MOVE TO MAILBOX # ***** 577 *****
7431 024044 005242                         INC    -(R2)          ;SET MSGTYP TO FATAL ERROR
7432 024046 000000                         HALT                                     ;CONDITION CODE INCORRECT
7433 024050 022737 042400 000000 MFPS7B: CMP    #42400,@#0   ;CHECK DESTINATION DATA
7434 024056 001404                         BEQ    MFPS7C
7435                                     :          ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <=====
7436                                     :          CONDITIONAL BRANCH INST. AND <=====
7437                                     :          REPLACE THE MOVE INSTRUCTION <=====
7438                                     :          WHICH FOLLOWS W/ 753 <=====
7439 024060 012742 000600                 MOV    #600,-(R2)      ;MOVE TO MAILBOX # ***** 600 *****
    
```

```
7440 024064 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
7441 024066 000000          HALT                    ;DEST. DATA INCORRECT
7442 024070 022700 000777  MFPS7C: CMP      #777,R0    ;CHECK MODE 7 REGISTER
7443 024074 001404          BEQ      TST252
7444                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7445                                     ;          CONDITIONAL BRANCH INST. AND <====
7446                                     ;          REPLACE THE MOVE INSTRUCTION <====
7447                                     ;          WHICH FOLLOWS W/ 744 <====
7448 024076 012742 000601          MOV      #601,-(R2)    ;MOVE TO MAILBOX # ***** 601 *****
7449 024102 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
7450 024104 000000          HALT                    ;MODE 7 REGISTER MODIFIED
7451                                     ; OR SEQUENCE ERROR
```

```
*****
: THIS TEST VERIFIES THAT RESET DOES NOT CLEAR THE PSW.
: 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.
```

```
7460
7461
7462 :TEST 252 TEST THAT RESET DOES NOT CLEAR PSW
7463 *****
7464 024106 005212          TST252: INC      (R2)          ;UPDATE TEST NUMBER
7465 024110 022712 000252          CMP      #252,(R2)    ;SEQUENCE ERROR?
7466 024114 001014          BNE     TST253-10    ;BR TO ERROR HALT ON SEQ ERROR
7467 024116 123727 026060 000377  CMPB    @#PASSPT,#377 ;ONLY DUE RESET EVERY 256. PASSES
7468 024124 001014          BNE     REST          ;BR IF TO SKIP TEST
7469 024126 012737 000357 177776  MOV     #357,@#PS     ;MOV ONES TO PSW
7470 024134 000005          RESET
7471 024136 022737 000357 177776  CMP     #357,@#PS     ;PSW CORRECT?
7472 024144 001404          BEQ     TST253
```

```
7473                                     ; TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
7474                                     ;          CONDITIONAL BRANCH INST. AND <====
7475                                     ;          REPLACE THE MOVE INSTRUCTION <====
7476                                     ;          WHICH FOLLOWS W/ 764 <====
7477 024146 012742 000602          MOV      #602,-(R2)    ;MOVE TO MAILBOX # ***** 602 *****
7478 024152 005242          INC      -(R2)          ;SET MSGTYP TO FATAL ERROR
7479 024154 000000          HALT                    ;RESET ALTERED PSW
7480                                     ; OR SEQUENCE ERROR
```

REST:

```
*****
: THE FOLLOWING TEST CHECKS THE INDEPENDENT FUNCTIONING OF BASIC
: DATA PATH COMPONENTS WITH USER MODE SET.
```

```
7481 024156
7482
7483 :TEST 253 TEST USER MODE R6 CAN HOLD A ONE IN EVERY POSITION
7484 *****
7485
7486
7487
7488
7489 :TEST 253 TEST USER MODE R6 CAN HOLD A ONE IN EVERY POSITION
7490 *****
7491 024156 005212          TST253: INC      (R2)          ;UPDATE TEST NUMBER
7492 024160 022712 000253          CMP      #253,(R2)    ;SEQUENCE ERROR?
7493 024164 001014          BNE     TST254-10    ;BR TO ERROR HALT ON SEQ ERROR
7494 024166 052767 140000 153602  BIS     #USR,PS       ;SET USER MODE
7495 024174 012706 000001          MOV     #1,R6         ;SET BIT0
```



```
7496 024200 000241          CLC          ;CLEAR C-BIT
7497 024202 006106          USP1: ROL      R6          ;ROTATE 1 POSITION
7498 024204 103376          BCC      USP1          ;BR IF NOT ALL DONE
7499 024206 001407          BEQ      USP1A         ;BR IF NO BITS PICKED
7500 024210 042767 140000 153560 BIC      #USRM,PS      ;CLEAR USER MODE
7501 024216 012742 000603    MOV      #603,-(R2)    ;MOVE TO MAILBOX # ***** 603 *****
7502 024222 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
7503 024224 000000          HALT          ;USER MODE R6 PICKED A BIT
7504 024226
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515
7516 024226 005212          TST254: INC     (R2)    ;UPDATE TEST NUMBER
7517 024230 022712 000254    CMP      #254,(R2)    ;SEQUENCE ERROR?
7518 024234 001036          BNE      USP4-14     ;BR TO ERROR HALT ON SEQ ERROR
7519 024236 052767 140000 153532 BIS      #USRM,PS      ;SET USER MODE
7520 024244 012706 177777    MOV      #-1,R6      ;SET USER R6 TO ALL ONES
7521 024250 022706 177777    CMP      #-1,R6      ;READ AND CHECK USER R6
7522 024254 001407          BEQ      USP2        ;BR IF NO ERROR
7523 024256 042767 140000 153512 BIC      #USRM,PS      ;CLEAR USER MODE
7524 024264 012742 000604    MOV      #604,-(R2)  ;MOVE TO MAILBOX # ***** 604 *****
7525 024270 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
7526 024272 000000          HALT          ;USER R6 WILL NOT HOLD ALL ONES
7527 024274 042767 140000 153474 USP2: BIC      #USRM,PS      ;SET KERNEL MODE
7528 024302 022706 177777    CMP      #-1,R6      ;KERNEL MODE R6 ADDR. FROM USER MODE?>>
7529 024306 001004          BNE      USP3
7530
7531
7532
7533
7534 024310 012742 000605          MOV      #605,-(R2)  ;MOVE TO MAILBOX # ***** 605 *****
7535 024314 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
7536 024316 000000          HALT          ;DUAL ADDRESSING ERROR USER/KERNEL R6
7537 024320 005006          USP3: CLR      R6          ;CLEAR KERNEL MODE SP
7538 024322 052767 140000 153446 BIS      #USRM,PS      ;SET USER MODE
7539 024330 022706 177777    CMP      #-1,R6      ;CHECK USER R6 NOT ADDR. FROM KERNEL MODE
7540 024334 001404          BEQ      USP4        ;BR IF NO ERROR
7541 024336 012742 000606          MOV      #606,-(R2)  ;MOVE TO MAILBOX # ***** 606 *****
7542 024342 005242          INC      -(R2)        ;SET MSGTYP TO FATAL ERROR
7543 024344 000000          HALT          ;DUAL ADDRESSING ERROR OR SEQUENCE ERROR
7544 024346 012706 000500          USP4: MOV      #STBOT,R6 ;RESTORE SP USER
7545 024352 042767 140000 153416 BIC      #USRM,PS      ;SET KERNEL MODE
7546 024360 012706 000500          MOV      #STBOT,R6   ;RESTORE SP KERNEL
7547
7548
7549
7550
7551
```

```
*****
:
: THIS TEST CHECKS THE INDEPENDENT FUNCTIONING OF THE USER
: 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.
:
*****
```

```
*****
: TEST 254 TEST INDEPENDENCE OF USER AND KERNEL MODE R6'S
:
*****
```

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

```
*****
:
: THESE NEXT TWO TESTS VERIFY MFPI AND MTP1 INSTRUCTIONS
: WITH R6 IN MODE 0.
:
*****
```

```
7552  
7553  
7554 :*****  
7555 :TEST 255 TEST MFPI WITH R6 IN MODE 0  
7556 :*****  
7556 024364 005212 TST255: INC (R2) ;UPDATE TEST NUMBER  
7557 024366 022712 000255 CMP #255,(R2) ;SEQUENCE ERROR?  
7558 024372 001032 BNE TST256-10 ;BR TO ERROR HALT ON SEQ ERROR  
7559 024374 012706 000500 MOV #STBOT,R6 ;INITIALIZE KERNEL STACK POINTER  
7560 024400 012767 140000 153370 MOV #USRM,PS ;SET USER MODE/PREVIOUS KERNEL  
7561 024406 012706 026424 MOV #USTBOT,R6 ;INITIALIZE USER STACK POINTER  
7562 024412 006506 MFPI R6 ;TRY MFPI WITH MODE 0  
7563 024414 022767 140000 153354 CMP #140000,PS ;CHECK PSW  
7564 024422 001407 BEQ MFPI0 ;BR IF NO ERROR  
7565 024424 042767 140000 153344 BIC #USRM,PS ;CLEAR USER MODE  
7566 024432 012742 000607 MOV #607,-(R2) ;MOVE TO MAILBOX # ***** 607 *****  
7567 024436 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7568 024440 000000 HALT ;INCORRECT PSW FROM MFPI  
7569 024442 022767 000500 001752 MFPI0: CMP #STBOT,USTBOT-2 ;CHECK DATA ON STACK  
7570 024450 001407 BEQ MFPI0A ;BR IF NO ERROR  
7571 024452 042767 140000 153316 BIC #USRM,PS ;CLEAR USER MODE  
7572 024460 012742 000610 MOV #610,-(R2) ;MOVE TO MAILBOX # ***** 610 *****  
7573 024464 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7574 024466 000000 HALT ;INCORRECT DATA FROM MFPI  
7575 024470 MFPI0A:  
7576  
7577 :*****  
7578 :TEST 256 TEST MTPI WITH R6 IN MODE 0  
7579 :*****  
7580 024470 005212 TST256: INC (R2) ;UPDATE TEST NUMBER  
7581 024472 022712 000256 CMP #256,(R2) ;SEQUENCE ERROR?  
7582 024476 001033 BNE TST257-10 ;BR TO ERROR HALT ON SEQ ERROR  
7583 024500 005067 153272 CLR PS ;SET KERNEL MODE  
7584 024504 005006 CLR R6 ;INITIALIZE KERNEL R6  
7585 024506 012767 140000 153262 MOV #USRM,PS ;SET USER MODE/PREVIOUS KERNEL  
7586 024514 012706 026424 MOV #USTBOT,R6 ;INITIALIZE USER STACK POINTER  
7587 024520 012746 000500 MOV #STBOT,-(R6) ;SET UP TARGET DATA  
7588 024524 006606 MTPI R6 ;TRY MODE 0 MTPI  
7589 024526 022767 140000 153242 CMP #USRM,PS ;CHECK PSW  
7590 024534 001407 BEQ MTPI0 ;BR IF NO ERROR  
7591 024536 042767 140000 153232 BIC #USRM,PS ;CLEAR USER MODE  
7592 024544 012742 000611 MOV #611,-(R2) ;MOVE TO MAILBOX # ***** 611 *****  
7593 024550 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7594 024552 000000 HALT ;PS INCORRECT FOLLOWING MTPI  
7595 024554 005067 153216 MTPI0: CLR PS ;SET KERNEL MODE  
7596 024560 020627 000500 CMP R6,#STBOT ;CHECK TARGET DATA  
7597 024564 001404 BEQ TST257  
7598  
7599 : TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
7600 : CONDITIONAL BRANCH INST. AND <====  
7601 : REPLACE THE MOVE INSTRUCTION <====  
7602 : WHICH FOLLOWS W/ 745 <====  
7602 024566 012742 000612 MOV #612,-(R2) ;MOVE TO MAILBOX # ***** 612 *****  
7603 024572 005242 INC -(R2) ;SET MSGTYP TO FATAL ERROR  
7604 024574 000000 HALT ;DATA INCORRECT FOLLOWING MTPI  
7605 : OR SEQUENCE ERROR  
7606
```

7607  
7608  
7609  
7610  
7611  
7612  
7613  
7614  
7615  
7616  
7617  
7618  
7619  
7620  
7621  
7622  
7623  
7624  
7625  
7626  
7627  
7628  
7629  
7630  
7631  
7632  
7633  
7634  
7635  
7636  
7637  
7638  
7639  
7640  
7641  
7642  
7643  
7644  
7645  
7646  
7647  
7648  
7649  
7650  
7651  
7652  
7653  
7654  
7655  
7656  
7657  
7658  
7659  
7660  
7661  
7662

024576 005212  
024600 022712 000257  
024604 001062  
024606 012700 026214  
024612 012704 026252  
024616 012767 000017 000142  
024624 012067 000110  
024630 012401  
024632 012767 177777 000074  
024640 012703 000020  
024644 005267 000064  
024650 032701 100000  
024654 013705 177776  
024660 042705 177773  
024664 000165 024670  
024670 000167 000020  
024674 012767 024770 000042  
024702 012767 024752 000040  
024710 000167 000014  
024714 012767 024752 000022  
024722 012767 024770 000020  
024730 006101  
024732 012737  
024734 000000  
024736 177776  
024740 000000  
024742 000137  
024744 000000

```
*****
: 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.
*****
: TEST 257, TEST THE BRANCH ROM
*****
TST257: INC (R2) ;UPDATE TEST NUMBER
        CMP #257,(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 ; (JUMP NOT EQUAL)
        JMP +4(R5) ; TO SET2BR
        JMP SET2BR
        MOV #CONT,NBR ;SET TO CONTINUE IF NO BRANCH
        MOV #ER,YBR ;SET TO REPORT ERROR IF BRANCH
        JMP AROUND ;GC 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
    177776
BRH: 0 ;BRANCH INST. GOES HERE
    JMP @ (PC)+ ;THIS JUMP IF NO BRANCH
NBR: 0 ;WHERE TO GO IF NO BRANCH OCCURS
```

7663	024746	000137	
7664	024750	000000	
7665	024752	012702	000304
7666	024756	012742	000613
7667	024762	005242	
7668	024764	000000	
7669	024766	000000	
7670	024770	005303	
7671	024772	013705	177776
7672	024776	042705	177773
7673	025002	000165	025006
7674	025006	000167	177632
7675	025012	005367	177750
7676	025016	013705	177776
7677	025022	042705	177773
7678	025026	000165	025032
7679	025032	000167	177566

YBR:	JMP	@(PC)+	:THIS JUMP IF BRANCH OCCURS
ER:	0		:WHERE TO GO IF BRANCH OCCURS
	MOV	#\$TESTN,R2	:RESTORE POINTER
	MOV	#613,-(R2)	:MOVE TO MAILBOX # ***** 613 *****
	INC	-(R2)	:SET MSGTYP TO FATAL ERROR
	HALT		:
BRCT:	0		:
CONT:	DEC	R3	:CC'S DONE?
	MOV	@#177776,R5	:SIMULATE A JNE
	BIC	#177773,R5	: (JUMP NOT EQUAL)
	JMP	+.4(R5)	: TO SETCC
	JMP	SETCC	:
	DEC	BRCT	:BR'S DONE?
	MOV	@#177776,R5	:SIMULATE A JNE
	BIC	#177773,R5	: (JUMP NOT EQUAL)
	JMP	+.4(R5)	: TO SETBR
	JMP	SETBR	:

7680  
7681  
7682  
7683  
7684  
7685  
7686  
7687  
7688  
7689  
7690  
7691 025036 005212  
7692 025040 022712 000260  
7693 025044 001052  
7694 025046 005000  
7695 025050 005001  
7696 025052 005002  
7697 025054 005003  
7698 025056 005004  
7699 025060 005005  
7700 025062 005006  
7701 025064 052700 000001  
7702 025070 052701 000002  
7703 025074 052702 000004  
7704 025100 052703 000010  
7705 025104 052704 000020  
7706 025110 052705 000040  
7707 025114 052706 000100  
7708 025120 022706 000100  
7709 025124 001022  
7710 025126 022705 000040  
7711 025132 001017  
7712 025134 022704 000020  
7713 025140 001014  
7714 025142 022703 000010  
7715 025146 001011  
7716 025150 022702 000004  
7717 025154 001006  
7718 025156 022701 000002  
7719 025162 001003  
7720 025164 022700 000001  
7721 025170 001404  
7722  
7723  
7724  
7725  
7726 025172  
7727 025172 012742 000614  
7728 025176 005242  
7729 025200 000000  
7730 025202 012702 000304  
7731 025206 012706 000500

```
*****  
: 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.  
*****  
: TEST 260 DUAL REGISTER ADDRESSING TEST  
*****  
TST260: INC (R2) ;UPDATE TEST NUMBER  
CMP #260,(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/ 726 <====  
  
DAERR: MOV #614,-(R2) ;MOVE TO MAILBOX # ***** 614 *****  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;DUAL ADDRESSING ERROR  
BITCON: MOV # $TESTN,R2 ;RESTORE POINTER  
MOV # $STBOT,R6 ;RESET STACK
```

7732  
7733  
7734  
7735  
7736  
7737  
7738  
7739  
7740  
7741 025212 005212  
7742 025214 022712 000261  
7743 025220 001012  
7744 025222 052737 170357 177776  
7745 025230 105037 177776  
7746 025234 013700 177776  
7747 025240 032700 170000  
7748 025244 001006  
7749 025246 005037 177776  
7750 025252 012742 000615  
7751 025256 005242  
7752 025260 000000  
7753 025262 005037 177776  
7754  
7755  
7756  
7757  
7758  
7759  
7760  
7761  
7762  
7763  
7764 025266 005212  
7765 025270 022712 000262  
7766 025274 001010  
7767 025276 000277  
7768 025300 000252  
7769 025302 000167 000000  
7770 025306 100403  
7771 025310 001002  
7772 025312 102401  
7773 025314 103404  
7774  
7775  
7776  
7777  
7778 025316  
7779 025316 012742 000616  
7780 025322 005242  
7781 025324 000000  
7782

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

TEST 261 TEST BYTE INSTRUCTION ON PSW

\*\*\*\*\*  
TST261: INC (R2) ;UPDATE TEST NUMBER  
CMP #261,(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 #615,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 615 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;BYTE INSTRUCTION ALTERED PSW  
BTCON: CLR @#PS ;RETURN TO KERNEL MODE  
\*\*\*\*\*

\*\*\*\*\*  
: 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.  
\*\*\*\*\*

TEST 262 TEST THAT JMP INSTRUCTION DOES NOT AFFECT CONDITION CODES

\*\*\*\*\*  
TST262: INC (R2) ;UPDATE TEST NUMBER  
CMP #262,(R2) ;SEQUENCE ERROR?  
BNE TST263-10 ;BR TO ERROR HALT ON SEQ ERROR  
SCC  
+CLN!CLV ;CC=0101  
JMPT: JMP JMPT ;JUMP TO TEST PSW  
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 IF SET  
BCS TST263  
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====  
: CONDITIONAL BRANCH INST. AND <====  
: REPLACE THE MOVE INSTRUCTION <====  
: WHICH FOLLOWS W/ 770 <====  
JMPTERR: MOV #616,-(R2) ;MOVE TO MAILBOX # \*\*\*\*\* 616 \*\*\*\*\*  
INC -(R2) ;SET MSGTYP TO FATAL ERROR  
HALT ;JMP INSTRUCTION AFFECTED CC'S  
: OR SEQUENCE ERROR

7783  
7784  
7785  
7786  
7787  
7788  
7789  
7790  
7791  
7792  
7793  
7794  
7795  
7796  
7797  
7798  
7799  
7800 025326 005212  
7801 025330 022712 000263  
7802 025334 001062  
7803 025336 012767 000240 000024  
7804 025344 012767 000017 000032  
7805 025352 012767 000261 000102  
7806 025360 012767 000001 000110  
7807 025366 000277  
7808 025370 000000  
7809 025372 013704 177776  
7810 025376 042704 177760  
7811 025402 022704  
7812 025404 000000  
7813 025406 001404  
7814  
7815  
7816  
7817  
7818 025410 012742 000617  
7819 025414 005242  
7820 025416 000000  
7821 025420 005367 177760  
7822 025424 005267 177740  
7823 025430 026727 177734 000257  
7824 025436 003753  
7825 025440 026727 177724 000260  
7826 025446 001004  
7827 025450 012767 000017 177726  
7828 025456 000743  
7829 025460 000257  
7830 025462 000000  
7831 025464 013704 177776  
7832 025470 042704 177760  
7833 025474 022704  
7834 025476 000000  
7835 025500 001404  
7836  
7837  
7838

```
*****
:
: 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 NOP'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.
:
: *****
: TEST 263 TEST SET CC AND CLEAR CC INSTRUCTIONS
: *****
TST263: INC (R2) ;UPDATE TEST NUMBER
        CMP #263,(R2) ;SEQUENCE ERROR?
        BNE CCERR ;BR TO ERROR HALT ON SEQ ERROR
        MOV #240,CC1 ;INITIALIZE CLR CC INSTRUCTION CODES
        MCV #17,CC2 ;INITIALIZE OCTAL MAP
        MOV #261,SC3 ;INITIALIZE SET CC INSTRUCTION CODES
        MOV #1,SC4 ;INITIALIZE OCTAL MAP
CLRCD: SCC ;SET ALL CONDITION CODES
CC1: 0 ;CONDITION CODE INSTRUCTION
      MOV @#PS,R4 ;COPY THE PSW
      BIC #177760,R4 ;ISOLATE CONDITION CODES
      CMP (PC)+,R4 ;CHECK THAT PROPER CC'S WERE CLEARED
CC2: 0 ;OCTAL REPRESENTATION OF CC'S
      BEQ CON1
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
: WHICH FOLLOWS W/ 753 <====
:
: MOVE TO MAILBOX # ***** 617 *****
CON1: DEC CC2 ;SET MSGTYP TO FATAL ERROR
      INC CC1 ;CLEAR CC INSTRUCTION FAILED
      CMP CC1,#257 ;SET NEXT OCTAL MAP OF CC'S
      BLE CLRCD ;GET NEXT CLEAR CC INSTRUCTION
      CMP CC1,#260 ;TEST FOR CCC INSTRUCTION
      BNE SETCD ;GO TEST NEXT INSTRUCTION IF NOT FOUND
      MOV #17,CC2 ;CHECK FOR NOP=260
      BR CLRCD ;GO TEST SET CC INSTRUCTIONS
SETCD: CCC ;SET OCTAL MAP TO TEST NOP
SC3: 0 ;GC TEST NOP
      MOV @#PS,R4 ;CLEAR ALL CONDITION CODES
      BIC #177760,R4 ;CONDITION CODE INSTRUCTION
      CMP (PC)+,R4 ;COY PSW
      BEQ CON2 ;CLEAR AWAY UNWANTED BITS
:
: TO SCOPE: CLEAR THE RIGHT BYTE OF THIS <====
: CONDITIONAL BRANCH INST. AND <====
: REPLACE THE MOVE INSTRUCTION <====
```

```
7839  
7840 025502                CCERR:                    ;           WHICH FOLLOWS W/ 716        <=====  
7841 025502 012742 000620      MOV     #620,-(R2)        ;MOVE TO MAILBOX # ***** 620 *****  
7842 025506 005242              INC     -(R2)            ;SET MSGTYP TO FATAL ERROR  
7843 025510 000000              HALT                               ;SET CC FAILED OR SEQUENCE ERROR  
7844 025512 005267 177760      CON2:   INC     SC4        ;SET NEXT OCTAL MAP  
7845 025516 005267 177740      INC     SC3              ;PREPARE NEXT SET CC INSTRUCTION  
7846 025522 026727 177734 000277  CMP     SC3,#277        ;FINISHED?  
7847 025530 003753              BLE     SETCD            ;BR IF NO
```



```
7848 :*****  
7849 ;TEST 264 END OF PASS SEQUENCE  
7850 :*****  
7851 TST264: INC (R2) ;UPDATE TEST NUMBER  
7852 CMP #264,(R2) ;SEQUENCE ERROR?  
7853 BNE EOP1 ;BR TO ERROR HALT ON SEG ERROR  
7854 INCB PASSPT ;SHOULD PRINT THIS PASS?  
7855 BNE GOAGIN ;NO  
7856 INC @#SPASS  
7857 BITB #40,$ENVM ;WILL APT ALLOW PRINTING?  
7858 BNE ACT ;NO  
7859 CMP @#42,$SENDAD ;UNDER ACT AUTO ACCEPT?  
7860 BEQ ACT ;IF SO SKIP PRINTOUT  
7861 CMP @#SPASS,#1 ;IS THIS 1ST PASS?  
7862 BNE 1$  
7863 MOV #TITLE,R0 ;THEN PRINT TITLE  
7864 JSR PC,@#WAIT  
7865 1$: MOV #MSG,R0 ;NOW PRINT END PASS  
7866 JSR PC,@#WAIT  
7867 MOV #BUFF,R0 ;SET UP TO BUILD EOP#  
7868 MOVB #377,-(R0) ;MOV TERM INTO BOT OF PSNUM  
7869 MOVB #0,-(R0) ;MOVE THREE  
7870 MOVB #0,-(R0) ;NULL BYTES  
7871 MOVB #0,-(R0) ;ON TOP OF TERMINATOR  
7872 JSR PC,@#BUILD ;GO BUILD ASCII NUMBER  
7873 MOVB #0,-(R0) ;MOVE THREE  
7874 MOVB #0,-(R0) ;NULL BYTES  
7875 MOVB #0,-(R0) ;ON TOP OF ASCII NUMBER  
7876 JSR PC,@#WAIT ;GO PRINT PSNUM (PASSNUMBER)  
7877 BR ACT ;SERVICE ACT  
7878  
7879 WAIT: TSTB @#TPS ;ROUTINE TO PRINT MSG  
7880 BPL WAIT ;WAIT FOR TTY READY  
7881 CMPB (R0),#377 ;CHECK FOR TERMINATOR  
7882 BEQ 1$  
7883 MOVB (R0)+,@#TPB ;NOT TERM, PRINT CHAR  
7884 BR WAIT ;GET NEXT CHARACTER  
7885 1$: RTS PC ;CHAR STRING DONE, RETURN  
7886  
7887 BUILD: MOV @#SPASS,@#OCTPSS ;ROUTINE TO CONV OCTAL TO ASCII  
7888 1$: MOV #60,@#ASCPSS ;MOVE ZERO, ASCII FORMAT  
7889 ASR @#OCTPSS ;MOVE LOWEST BIT INTO CARRY  
7890 BCC 2$ ;CHECK CARRY  
7891 ADD #1,@#ASCPSS ;AND ADD VALUE TO ZERO  
7892 CLC ;CLEAR CARRY  
7893 2$: ASR @#OCTPSS ;REPEAT FOR 2ND BIT  
7894 BCC 3$  
7895 ADD #2,@#ASCPSS  
7896 CLC  
7897 3$: ASR @#OCTPSS ;REPEAT FOR 3RD BIT  
7898 BCC 4$  
7899 ADD #4,@#ASCPSS  
7900 CLC  
7901 4$: MOVB @#ASCPSS,-(R0) ;STORE ASCII DIGIT  
7902 TST @#OCTPSS ;CHECK FOR MORE BITS  
7903 BNE 1$ ;REPEAT UNTIL OCTPSS=0
```

```
7904 026022 000207          RTS      PC          ;THEN RETURN
7905
7906 026024 013700 000042    ACT:     MOV      @#42,R0 ;CHECK ACT
7907 026030 001405          BEQ      GOAGIN   ;KEEP GOING
7908 026032 000005          RESET
7909 026034 004710    $ENDAD: JSR      PC,(R0) ;ACT HOOKS
7910 026036 000240          NOP
7911 026040 000240          NOP
7912 026042 000240          NOP
7913 026044 000167 152452    GOAGIN: JMP      RESTRT   ;DO NEXT PASS
7914 026050          EOP1:
7915 026050 012742 000621    MOV      #621,-(R2) ;MOVE TO MAILBOX # ***** 621 *****
7916 026054 005242          INC      -(R2)    ;SET MSGTYP TO FATAL ERROR
7917 026056 000000          HALT      ;SEQUENCE ERROR
7918 026060 177777    PASSPT: -1
7919 026062 000000    OCTPSS: .WORD 0 ;PASSCOUNT, OCTAL, STORED HERE
7920 026064 000000    ASCPSS: .WORD 0 ;PASSCOUNT, ASCII, BUILT HERE
7921 026066 005015 000000 000000 TITLE: .ASCII <15><12><0><0><0><0><0><0><0>.CFKAACO 11/34 BSC INST TST.<0><0><0><0><0><0>
7922 026074 000000 043103 040513
7923 026102 041501 020060 030461
7924 026110 031457 020064 051502
7925 026116 020103 047111 052123
7926 026124 052040 052123 000000
7927 026132 000000 000000 177400
7928
7929 026140 005015 000000 000000 MSG:     .EVEN
7930 026146 000000 047105 020104 MSG:     .ASCII <15><12><0><0><0><0><0><0><0>.END PASS .<0><0><0><0><0><0><377>
7931 026154 040520 051523 000040
7932 026162 000000 000000 177400
7933
7934
7935          .EVEN
7936          ;*****
7937          ;THESE ARE A UNIT, INSERT NO CODE BETWEEN THEM *
7938 PSNUM: .WORD 0,0,0 ;
7939          .WORD 0,0,0 ;
7940          .WORD 0,0,0 ;
7941          .WORD 0 ;
          ;*****
```

```
7942 026214 000402          BRTAB: BR      .+6
7943 026216 001002          BNE      .+6
7944 026220 001402          BEQ      .+6
7945 026222 002002          BGE      .+6
7946 026224 002402          BLT      .+6
7947 026226 003002          BGT      .+6
7948 026230 003402          BLE      .+6
7949 026232 100002          BPL      .+6
7950 026234 100402          BMI      .+6
7951 026236 101002          BHI      .+6
7952 026240 101402          BLOS     .+6
7953 026242 102002          BVC      .+6
7954 026244 102402          BVS      .+6
7955 026246 103002          BCC      .+6          ;SAME AS BHIS
7956 026250 103402          BCS      .+6          ;SAME AS BLO
7957
7958          000002          .RADIX 2
7959 026252 177777          YNTAB: 1111111111111111
7960 026254 170360          1111000011110000
7961 026256 007417          0000111100001111
7962 026260 146063          1100110000110011
7963 026262 031714          0011001111001100
7964 026264 140060          1100000000110000
7965 026266 037717          0011111111001111
7966 026270 177400          1111111100000000
7967 026272 000377          0000000011111111
7968 026274 120240          1010000010100000
7969 026276 057537          0101111101011111
7970 026300 146314          1100110011001100
7971 026302 031463          0011001100110011
7972 026304 125252          1010101010101010
7973 026306 052525          0101010101010101
7974          000010          .RADIX 8
7975
7976 026310 012737 026320 000024 PWRDN: MOV      #PWRUP,@#24          ;SET UP FOR A POWER UP
7977 026316 000000          HALT
7978
7979 026320 012737 026310 000024 PWRUP: MOV      #PWRDN,@#24          ;SET UP FOR A POWER FAIL
7980 026326 012706 000500          MOV      #STBOT,R6          ;SET UP STACK POINTER
7981 026332 132767 000040 151761          BITB     #40,$ENVM          ;SHOULD PRINT?
7982 026340 001010          BNE      PWR2              ;IF NOT: BR
7983 026342 012700 026366          MOV      #PFMES,R0          ;GET POWER FAIL MESSG.
7984 026346 105737 177564          WATE:   TSTB    @#TPS          ;TTY READY?
7985 026352 100375          BPL      WATE              ;IF NOT: BR
7986 026354 112037 177566          MOVB    (R0)+,@#TPB          ;PRINT NEXT CHAR.
7987 026360 001372          BNE      WATE              ;IF NOT DONE: BR
7988 026362 000137 000500          PWR2:   JMP      @#START          ;START PROGRAM AGAIN
7989
7990 026366 006412 047520 042527 PFMES: .ASCIZ <12><15>.POWER FAILURE.<12><15>
7991 026374 020122 040506 046111
7992 026402 051125 005105 000015
7993          .EVEN
7994 026410 000006          .BLKW   6
7995 026424          JSTBOT:
7996          ;*****
7997          ; THE FOLLOWING ARE SPECIAL CPU TRAP
```



ABASE = 000000	28			
ACDW1 = 000000	28			
ACDW2 = 000000	28			
ACPUOP= 000000	28	43		
ACT 026024	7858	7860	7877	7906#
ADC1 020054	6088	6089	6095#	
ADC2 020064	6090	6099#		
ADC3 020104	6103	6104	6110#	
ADC4 020114	6105	6114#		
ADC5 020132	6117	6118	6119	6125#
ADDW0 = 000000	28			
ADDW1 = 000000	28			
ADDW10= 000000	28			
ADDW11= 000000	28			
ADDW12= 000000	28			
ADDW13= 000000	28			
ADDW14= 000000	28			
ADDW15= 000000	28			
ADDW2 = 000000	28			
ADDW3 = 000000	28			
ADDW4 = 000000	28			
ADDW5 = 000000	28			
ADDW6 = 000000	28			
ADDW7 = 000000	28			
ADDW8 = 000000	28			
ADDW9 = 000000	28			
ADD1 017670	6010	6011	6017#	
ADD2 017700	6012	6021#		
ADD3 017714	6024	6025	6031#	
ADD4 017724	6026	6035#		
ADD5 017742	6038	6039	6045#	
ADD6 017752	6040	6049#		
ADD7 017764	6050	6051	6057#	
ADD8 017774	6052	6061#		
ADD9 020014	6064	6065	6066	6072#
ADEVCT= 000000	28	34		
ADEVN = 000000	28			
AENV = 000000	28	39		
AENVN = 000000	28	40		
AFATAL= 000000	28	31		
AMADR1= 000000	28			
AMADR2= 000000	28			
AMADR3= 000000	28			
AMADR4= 000000	28			
AMAMS1= 000000	28			
AMAMS2= 000000	28			
AMAMS3= 000000	28			
AMAMS4= 000000	28			
AMSGAD= 000000	28	36		
AMSGLG= 000000	28	37		
AMSGTY= 000000	28	30		
AMTYP1= 000000	28			
AMTYP2= 000000	28			
AMTYP3= 000000	28			
AMTYP4= 000000	28			
APASS = 000000	28	33		

APRIOR=	000000	28					
AROUND	024730	7652	7655#				
ASCPSS	026064	7888*	7891*	7895*	7899*	7901	7920#
ASL1	021370	6590	6591	6592	6598#		
ASL2	021400	6593	6602#				
ASL3	021416	6605	6606	6607	6613#		
ASL4	021426	6608	6617#				
ASL5	021442	6620	6621	6627#			
ASL6	021452	6622	6631#				
ASL7	021476	6634	6635	6636	6637	6644#	
ASR1	021540	6659	6660	6661	6667#		
ASR2	021550	6662	6671#				
ASR3	021572	6675	6676	6677	6683#		
ASR4	021602	6678	6687#				
ASR5	021616	6690	6691	6692	6698#		
ASR6	021626	6693	6702#				
ASR7	021656	6706	6707	6708	6709	6716#	
ASWREG=	000000	28	41				
ATESTN=	000000	28	32				
AUNIT =	000000	28	35				
AUSWR =	000000	28	42				
AVECT1=	000000	28					
AVECT2=	000000	28					
BIC1	017020	5687	5688	5694#			
BIC2	017030	5689	5698#				
BIC3	017046	5701	5702	5708#			
BIS1	017110	5723	5724	5725	5731#		
BIS2	017120	5726	5735#				
BIS3	017140	5738	5739	5740	5746#		
BITCHK	025120	7708#					
BITCLR	025046	7694#					
BITCON	025202	7721	7730#				
BITSET	025064	7701#					
BIT1	016730	5650	5651	5657#			
BIT2	016740	5652	5662#				
BIT3	016756	5665	5666	5672#			
BRCT	024766	7639*	7669#	7675*			
BRC1	003040	1039	1045#				
BRC2	003050	1040	1050#				
BRC3	003060	1052	1058#				
BRH	024740	7640*	7660#				
BRN1	002720	945	951#				
BRN2	002730	946	956#				
BRN3	002740	958	964#				
BRTAB	026214	7637	7942#				
BRV1	002770	992	998#				
BRV2	003000	993	1003#				
BRV3	003010	1005	1011#				
BRZ1	002650	898	904#				
BRZ2	002660	899	909#				
BRZ3	002670	911	917#				
BR1	000572	134	140#				
BR2	000602	135	144#				
BR3	000614	145	153#				
BR4	000622	154	160#				
BR5	000632	155	164#				

BTCON	025262	7748	7753#						
BTERR	025246	7743	7749#						
BUFF	026212	7867	7939#						
BUILD	025722	7872	7887#						
CC	024734	7642*	7644*	7658#					
CCERR	025502	7802	7840#						
CC1	025370	7803*	7808#	7822*	7823	7825			
CC2	025404	7804*	7812#	7821*	7827*				
CLRCD	025366	7807#	7824	7828					
CLR1	017456	5903	5904	5905	5911#				
CMP1	020320	6207	6208	6214#					
CMP2	020330	6209	6218#						
CMP3	020352	6222	6223	6229#					
CMP4	020362	6224	6233#						
CMP5	020406	6237	6238	6239	6245#				
CMP6	020416	6240	6249#						
CMP7	020436	6252	6253	6259#					
COM1	020476	6275	6276	6282#					
CONT	024770	7650	7654	7670#					
CON1	025420	7813	7821#						
CON2	025512	7835	7844#						
DAERR	025172	7693	7709	7711	7713	7715	7717	7719	7726#
DEC1	017316	5826	5827	5828	5834#				
DEC2	017326	5829	5838#						
DEC3	017342	5841	5842	5848#					
DEC4	017352	5843	5852#						
DEC5	017366	5855	5856	5862#					
DEC6	017376	5857	5866#						
DEC7	017420	5870	5871	5872	5878#				
DNMBOA	010520	3459	3460	3461	3467#				
DNMBOB	010530	3462	3471#						
DNMB2A	010756	3569	3570	3571	3577#				
DNMB2B	010766	3572	3581#						
DNMB2C	011002	3582	3590#						
DNMB2D	011016	3592	3593	3599#					
DNMB2E	011026	3594	3603#						
DNMB2F	011044	3605	3613#						
DNMB3A	011126	3639	3640	3641	3647#				
DNMB3B	011136	3642	3651#						
DNMB3C	011154	3652	3660#						
DNMB3D	011172	3663	3664	3670#					
DNMB3E	011202	3665	3674#						
DNMB4A	011372	3742	3743	3744	3750#				
DNMB4B	011402	3745	3754#						
DNMB4C	011420	3755	3763#						
DNMB4D	011430	3764	3770#						
DNMB4E	011440	3765	3771#						
DNMB4F	011454	3775	3783#						
DNM03A	007612	3068	3069	3070	3076#				
DNM03B	007622	3071	3080#						
DNM03C	007632	3081	3088#						
DNM1	007464	3002	3010#						
DNM1A	010576	3492	3493	3494	3500#				
DNM1B	010606	3495	3504#						
DNM2	007500	3011	3019#						
DNM2A	010654	3525	3526	3532#					







MDM5B	012772	4280	4289#		
MDM5C	013010	4290	4298#		
MDM5D	013026	4299	4307#		
MDM5E	013054	4310	4318#		
MDM6A	013124	4347	4348	4354#	
MDM6B	013134	4349	4358#		
MDM6C	013152	4359	4367#		
MDM6D	013172	4368	4376#		
MDM6E	013222	4379	4387#		
MDM7A	013276	4415	4416	4422#	
MDM7B	013306	4417	4426#		
MDM7C	013324	4427	4435#		
MDM7D	013344	4436	4444#		
MDM7E	013370	4446	4454#		
MFP10	024442	7564	7569#		
MFP10A	024470	7570	7575#		
MFPS1	023206	7177	7186#		
MFPS2A	023276	7211	7212	7213	7219#
MFPS2B	023306	7214	7223#		
MFPS2C	023326	7224	7232#		
MFPS3A	023404	7253	7254	7255	7261#
MFPS3B	023414	7256	7265#		
MFPS3C	023434	7266	7274#		
MFPS4A	023512	7295	7296	7297	7303#
MFPS4B	023522	7298	7307#		
MFPS4C	023542	7308	7316#		
MFPS5A	023620	7337	7338	7339	7345#
MFPS5B	023630	7340	7349#		
MFPS5C	023650	7350	7358#		
MFPS6A	023730	7379	7380	7381	7387#
MFPS6B	023740	7382	7391#		
MFPS6C	023760	7392	7400#		
MFPS7A	024040	7421	7422	7423	7429#
MFPS7B	024050	7424	7433#		
MFPS7C	024070	7434	7442#		
MOV1	016640	5613	5614	5620#	
MOV2	016650	5615	5625#		
MOV3	016666	5628	5629	5635#	
MRK1	022246	6889	6896#		
MRK2	022270	6896	6897	6898	6900 6907#
MRK3	022300	6902	6911#		
MRK4	022322	6914	6916#		
MRK5	022334	6915	6920#		
MRK6	022350	6921	6928#		
MSG	026140	7865	7929#		
MTP10	024554	7590	7595#		
MTPS1	022420	6953	6961#		
MTPS1A	022440	6965	6966	6967	6973#
MTPS2	022514	6990	6998#		
MTPS3	022604	7021	7029#		
MTPS4	022672	7051	7059#		
MTPS5	022752	7081	7089#		
MTPS6	023042	7111	7119#		
MTPS7	023132	7141	7149#		
NBR	024744	7650*	7653*	7662#	
NEG00	004010	1501	1502	1503	1509#



REG4A	002160	672#	675								
REG4E	002126	649	657#								
REG5	002230	698#	701								
REG5A	002274	722#	725								
REG5E	002242	699	707#								
REG6	002344	747#	750								
REG6A	002410	771#	774								
REG6E	002356	748	756#								
REST	024156	7468	7481#								
RESTR1	000522	122#	7913								
ROL1	021060	6455	6456	6457	6463#						
ROL2	021070	6458	6467#								
ROL3	021106	6470	6471	6472	6478#						
ROL4	021116	6473	6482#								
ROL5	021132	6485	6486	6492#							
ROL6	021142	6487	6496#								
ROL7	021164	6499	6500	6501	6508#						
ROR1	021226	6523	6524	6525	6531#						
ROR2	021236	6526	6535#								
ROR3	021254	6538	6539	6540	6546#						
ROR4	021264	6541	6550#								
ROR5	021302	6553	6554	6555	6561#						
ROR6	021312	6556	6565#								
ROR7	021326	6568	6569	6575#							
ROTX	014644	4890	4892*	4896	4909#	4924*	4926*	4928	4953*	4954	4958
ROTXAD	014770	4954*	4956*	4969#							
ROTOA	013762	4621	4622	4629#							
ROTOB	013772	4624	4633#								
ROTOC	014014	4636	4637	4644#							
ROT1A	014062	4669	4670	4677#							
ROT1B	014072	4672	4681#								
ROT1C	014116	4684	4685	4692#							
ROT1D	014126	4687	4696#								
ROT1E	014156	4701	4702	4709#							
ROT2A	014230	4732	4734	4742#							
ROT2B	014240	4737	4746#								
ROT2C	014270	4750	4752	4759#							
ROT2D	014300	4754	4763#								
ROT2E	014334	4768	4770	4778#							
ROT3A	014402	4800	4807#								
ROT3B	014412	4802	4811#								
ROT3C	014440	4814	4821#								
ROT3D	014450	4816	4825#								
ROT3E	014476	4828	4835#								
ROT4	014552	4859	4861	4868#							
ROT5	014634	4889	4895	4897	4904#						
ROT6	014704	4927	4934#								
ROT7	014760	4952	4957	4964#							
RTS1	016574	5573	5580#								
R1ERR	001612	530	539#								
R2ERR	001742	578	584#								
R3ERR	002062	624	633#								
R4ERR	002176	673	682#								
R5ERR	002312	723	732#								
R6ERR	002426	772	781#								
SBC1	020712	6376	6377	6378	6384#						

SBC2	020722	6379	6388#		
SBC3	020740	6391	6392	6393	6399#
SBC4	020750	6394	6403#		
SBC5	020766	6406	6407	6408	6414#
SBC6	020776	6409	6418#		
SBC7	021016	6422	6423	6429#	
SB0	015022	4987	4995#		
SB2	015144	5051	5059#		
SB4	015270	5115	5123#		
SB5	015356	5149	5168#		
SB5A	015350	5154	5162#		
SB5X	015366	5151*	5153	5173#	5174
SB5XAD	015370	5150	5162	5174#	
SB6	015430	5191	5201#		
SB6X	015440	5192*	5193	5206#	
SB7	015500	5224	5234#		
SB7X	015510	5225*	5239#	5240	
SB7XAD	015512	5226	5240#		
SCOPE =	000240				
SC3	025462	7805*	7830#	7845*	7846
SC4	025476	7806*	7834#	7844*	
SETBR	024624	7640#	7679		
SETCC	024644	7644#	7674		
SETCD	025460	7826	7829#	7847	
SETUP	024606	7637#			
SET2BR	024714	7649	7653#		
SHL	001200	337#	340		
SHLE	001214	338	346#		
SHR	001314	381#	384		
SHRE	001330	382	390#		
SNMBOA	005754	2297	2298	2304#	
SNMB1A	006060	2362	2363	2369#	
SNMB1B	006070	2364	2373#		
SNMB1C	006112	2378	2379	2380	2386#
SNMB2A	006234	2454	2455	2461#	
SNMB2B	006244	2456	2465#		
SNMB2C	006260	2466	2474#		
SNMB2D	006300	2478	2479	2480	2486#
SNMB2E	006310	2481	2490#		
SNMB3A	006452	2570	2571	2577#	
SNMB3B	006462	2572	2581#		
SNMB3C	006500	2584	2585	2586	2592#
SNMB3D	006510	2587	2596#		
SNM0A	005714	2264	2265	2266	2272#
SNM1A	006016	2329	2330	2331	2337#
SNM2A	006154	2410	2411	2412	2418#
SNM2B	006164	2413	2422#		
SNM3A	006364	2522	2523	2524	2530#
SNM3B	006374	2525	2534#		
SNM4A	006560	2627	2628	2634#	
SNM4B	006570	2629	2638#		
SNM5A	006642	2670	2671	2677#	
SNM5B	006652	2672	2681#		
SNM6A	006726	2713	2714	2720#	
SNM6B	006736	2715	2724#		
SNM7A	007010	2755	2756	2762#	





TST136	011062	3563	3614	3628#
TST137	011222	3630	3676	3689#
TST14	001370	428	432	445#
TST140	011330	3691	3720	3733#
TST141	011472	3735	3784	3797#
TST142	011602	3799	3828	3841#
TST143	011710	3843	3871	3884#
TST144	012020	3886	3914	3933#
TST145	012074	3935	3953	3972#
TST146	012166	3974	4002	4020#
TST147	012322	4022	4068	4086#
TST15	001422	447	450	463#
TST150	012500	4088	4134	4153#
TST151	012574	4155	4181	4202#
TST152	012730	4204	4249	4271#
TST153	013072	4273	4319	4340#
TST154	013242	4342	4388	4408#
TST155	013410	4410	4455	4480#
TST156	013474	4489	4518#	
TST157	013560	4527	4555#	
TST16	001454	465	468	481#
TST160	013644	4557	4564	4586#
TST161	013730	4588	4595	4615#
TST162	014024	4617	4639	4662#
TST163	014166	4664	4704	4725#
TST164	014344	4727	4773	4794#
TST165	014506	4796	4830	4852#
TST166	014562	4854	4863	4887#
TST167	014646	4899	4921#	
TST17	001506	483	486	499#
TST170	014714	4923	4929	4950#
TST171	014772	4959	4982#	
TST172	015040	4984	4996	5016#
TST173	015102	5018	5023	5044#
TST174	015162	5046	5060	5081#
TST175	015224	5083	5087	5108#
TST176	015304	5110	5124	5147#
TST177	015372	5163	5189#	
TST2	000644	132	165	192#
TST20	001552	501	509	523#
TST200	015442	5196	5222#	
TST201	015514	5229	5277#	
TST202	016060	5382	5412#	
TST203	016536	5546	5568#	
TST204	016612	5570	5581	5607#
TST205	016676	5609	5630	5643#
TST206	016766	5645	5667	5680#
TST207	017056	5682	5703	5716#
TST21	001622	525	534	547#
TST210	017150	5718	5741	5766#
TST211	017266	5768	5806	5820#
TST212	017430	5822	5873	5897#
TST213	017466	5899	5906	5920#
TST214	017552	5922	5944	5957#
TST215	017640	5959	5981	6004#
TST216	020024	6006	6067	6081#



TST217	020142	6083	6120	6144#
TST22	001676	571#		
TST220	020266	6146	6186	6200#
TST221	020446	6202	6254	6268#
TST222	020506	6270	6277	6302#
TST223	020660	6304	6356	6369#
TST224	021026	6371	6424	6448#
TST225	021174	6450	6503	6516#
TST226	021336	6518	6570	6583#
TST227	021506	6585	6639	6652#
TST23	001756	573	593#	
TST230	021666	6654	6711	6735#
TST231	021774	6737	6767	6788#
TST232	022106	6790	6819	6839#
TST233	022174	6841	6863	6883#
TST234	022360	6885	6923	6946#
TST235	022450	6948	6968	6982#
TST236	022532	6984	6999	7012#
TST237	022622	7014	7030	7043#
TST24	002022	595	603	617#
TST240	022706	7045	7060	7073#
TST241	022770	7075	7090	7103#
TST242	023060	7105	7120	7133#
TST243	023150	7135	7150	7171#
TST244	023242	7173	7191	7204#
TST245	023344	7206	7233	7246#
TST246	023452	7248	7275	7288#
TST247	023560	7290	7317	7330#
TST25	002072	619	628	642#
TST250	023666	7332	7359	7372#
TST251	023776	7374	7401	7414#
TST252	024106	7416	7443	7464#
TST253	024156	7466	7472	7491#
TST254	024226	7493	7516#	
TST255	024364	7556#		
TST256	024470	7558	7580#	
TST257	024576	7582	7597	7634#
TST26	002136	644	652	666#
TST260	025036	7691#		
TST261	025212	7741#		
TST262	025266	7764#		
TST263	025326	7766	7773	7800#
TST264	025532	7851#		
TST27	002206	668	677	692#
TST3	000700	194	198	211#
TST30	002252	694	702	716#
TST31	002322	718	727	741#
TST32	002366	743	751	765#
TST33	002436	767	776	805#
TST34	002476	807	811	824#
TST35	002534	826	829	842#
TST36	002572	844	847	860#
TST37	002630	862	865	892#
TST4	000736	213	217	230#
TST40	002700	894	912	939#
TST41	002750	941	959	986#

CFKAACO 11/34 BSC INST TST  
CFKAAC.P11 18-OCT-78 11:01

MACY11 30A(1052) 18-OCT-78 11:06 PAGE 189  
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0200

TST42	003020	988	1006	1033#																
TST43	003070	1035	1053	1102#																
TST44	003150	1104	1126	1149#																
TST45	003232	1151	1171	1191#																
TST46	003276	1193	1206	1228#																
TST47	003346	1230	1245	1266#																
TST5	000774	232	236	249#																
TST50	003426	1268	1287	1311#																
TST51	003512	1313	1334	1358#																
TST52	003600	1360	1382	1406#																
TST53	003666	1408	1430	1449#																
TST54	003762	1451	1476	1495#																
TST55	004074	1497	1539	1551#																
TST56	004214	1553	1595	1607#																
TST57	004320	1609	1641	1668#																
TST6	001032	251	254	282#																
TST60	004400	1670	1689	1715#																
TST61	004474	1717	1742	1769#																
TST62	004564	1771	1794	1807#																
TST63	004734	1809	1862	1883#																
TST64	005016	1885	1905	1934#																
TST65	005074	1936	1954	1976#																
TST66	005154	1978	1994	2017#																
TST67	005236	2019	2036	2050#																
TST7	001076	284	289	302#																
TST70	005326	2052	2079	2091#																
TST71	005430	2093	2125	2137#																
TST72	005514	2139	2160	2172#																
TST73	005604	2174	2196	2217#																
TST74	005664	2234	2257#																	
TST75	005724	2259	2267	2289#																
TST76	005764	2291	2299	2321#																
TST77	006026	2323	2332	2353#																
USP1	024202	7497#	7498																	
USP1A	024226	7499	7504#																	
USP2	024274	7522	7527#																	
USP3	024320	7529	7537#																	
USP4	024346	7518	7540	7544#																
USRM = 140000		12#	7494	7500	7519	7523	7527	7538	7545	7560	7565	7571	7585	7589						
		7591																		
USTBOT	026424	7561	7569	7586	7995#															
WAIT	025676	7864	7866	7876	7879#	7880	7884													
WATE	026346	7984#	7985	7987																
XOR1	022040	6796	6797	6798	6799	6806#														
XOR2	022050	6801	6810#																	
XOR3	022076	6814	6815	6816	6817	6824#														
YBR	024750	7651*	7654*	7664#																
YNTAB	026252	7638	7959#																	
\$APTHD	000330	61	67#																	
\$CPUOP	000326	43#																		
\$DEVCT	000310	34#																		
\$ENDAD	026034	20	7859	7909#																
\$ENV	000320	39#																		
\$ENVM	000321	40#	7857	7981																
\$ERN = 000632		1#	141	142#	150	151#	161	162#	170	171#	203	204#	222	223#						
		241	242#	259	260#	294	295#	313	314#	323	324#	347	348#	367						

368#	391	392#	437	438#	455	456#	473	474#	491	492#	515	516#
540	541#	563	564#	585	586#	609	610#	634	635#	658	659#	683
684#	708	709#	733	734#	757	758#	782	783#	816	817#	834	835#
852	853#	870	871#	905	906#	918	919#	952	953#	965	966#	999
1000#	1012	1013#	1046	1047#	1059	1060#	1111	1112#	1122	1123#	1131	1132#
1159	1160#	1177	1178#	1200	1201#	1212	1213#	1238	1239#	1251	1252#	1278
1279#	1293	1294#	1324	1325#	1340	1341#	1372	1373#	1388	1389#	1420	1421#
1436	1437#	1464	1465#	1482	1483#	1510	1511#	1520	1521#	1535	1536#	1544
1545#	1567	1568#	1577	1578#	1591	1592#	1600	1601#	1623	1624#	1637	1638#
1646	1647#	1681	1682#	1695	1696#	1730	1731#	1748	1749#	1783	1784#	1800
1801#	1827	1828#	1836	1837#	1847	1848#	1856	1857#	1867	1868#	1895	1896#
1911	1912#	1946	1947#	1960	1961#	1988	1989#	2000	2001#	2030	2031#	2042
2043#	2066	2067#	2075	2076#	2084	2085#	2111	2112#	2120	2121#	2130	2131#
2156	2157#	2165	2166#	2189	2190#	2201	2202#	2227	2228#	2240	2241#	2273
2274#	2305	2306#	2338	2339#	2370	2371#	2387	2388#	2419	2420#	2429	2430#
2462	2463#	2471	2472#	2487	2488#	2497	2498#	2531	2532#	2541	2542#	2578
2579#	2593	2594#	2603	2604#	2635	2636#	2644	2645#	2678	2679#	2688	2689#
2721	2722#	2730	2731#	2763	2764#	2772	2773#	2799	2800#	2826	2827#	2856
2857#	2865	2866#	2891	2892#	2904	2905#	2915	2916#	2930	2931#	2941	2942#
2972	2973#	2982	2983#	3007	3008#	3016	3017#	3026	3027#	3038	3039#	3048
3049#	3077	3078#	3089	3090#	3117	3118#	3145	3146#	3174	3175#	3209	3210#
3219	3220#	3249	3250#	3277	3278#	3287	3288#	3317	3318#	3326	3327#	3355
3356#	3365	3366#	3392	3393#	3418	3419#	3444	3445#	3468	3469#	3477	3478#
3501	3502#	3510	3511#	3533	3534#	3544	3545#	3553	3554#	3578	3579#	3587
3588#	3600	3601#	3610	3611#	3619	3620#	3648	3649#	3657	3658#	3671	3672#
3681	3682#	3707	3708#	3716	3717#	3725	3726#	3751	3752#	3760	3761#	3771
3772#	3780	3781#	3789	3790#	3815	3816#	3824	3825#	3833	3834#	3858	3859#
3867	3868#	3876	3877#	3901	3902#	3910	3911#	3919	3920#	3949	3950#	3958
3959#	3987	3988#	3998	3999#	4007	4008#	4034	4035#	4043	4044#	4055	4056#
4064	4065#	4073	4074#	4101	4102#	4110	4111#	4119	4120#	4129	4130#	4139
4140#	4168	4169#	4177	4178#	4186	4187#	4215	4216#	4224	4225#	4236	4237#
4245	4246#	4254	4255#	4286	4287#	4295	4296#	4304	4305#	4315	4316#	4324
4325#	4355	4356#	4364	4365#	4373	4374#	4384	4385#	4393	4394#	4423	4424#
4432	4433#	4441	4442#	4451	4452#	4460	4461#	4495	4496#	4533	4534#	4569
4570#	4600	4601#	4630	4631#	4645	4646#	4678	4679#	4693	4694#	4710	4711#
4743	4744#	4760	4761#	4779	4780#	4808	4809#	4822	4823#	4836	4837#	4869
4870#	4905	4906#	4935	4936#	4965	4966#	4992	4993#	5001	5002#	5028	5029#
5056	5057#	5065	5066#	5092	5093#	5120	5121#	5129	5130#	5159	5160#	5169
5170#	5202	5203#	5235	5236#	5289	5290#	5298	5299#	5312	5313#	5324	5325#
5333	5334#	5346	5347#	5359	5360#	5373	5374#	5387	5388#	5427	5428#	5446
5447#	5465	5466#	5483	5484#	5501	5502#	5517	5518#	5533	5534#	5552	5553#
5577	5578#	5585	5587#	5621	5622#	5636	5637#	5658	5659#	5673	5674#	5695
5696#	5709	5710#	5732	5733#	5747	5748#	5781	5782#	5797	5798#	5812	5813#
5835	5836#	5849	5850#	5863	5864#	5879	5880#	5912	5913#	5935	5936#	5950
5951#	5972	5973#	5987	5988#	6018	6019#	6032	6033#	6046	6047#	6058	6059#
6073	6074#	6096	6097#	6111	6112#	6126	6127#	6160	6161#	6176	6177#	6192
6193#	6215	6216#	6230	6231#	6246	6247#	6260	6261#	6283	6284#	6317	6318#
6332	6333#	6347	6348#	6362	6363#	6385	6386#	6400	6401#	6415	6416#	6430
6431#	6464	6465#	6479	6480#	6493	6494#	6509	6510#	6532	6533#	6547	6548#
6562	6563#	6576	6577#	6599	6600#	6614	6615#	6628	6629#	6645	6646#	6668
6669#	6684	6685#	6699	6700#	6717	6718#	6753	6754#	6773	6774#	6807	6808#
6825	6826#	6853	6854#	6869	6870#	6893	6894#	6908	6909#	6917	6918#	6929
6930#	6958	6959#	6974	6975#	6995	6996#	7004	7005#	7026	7027#	7035	7036#
7056	7057#	7065	7066#	7086	7087#	7095	7096#	7116	7117#	7125	7126#	7146
7147#	7155	7156#	7182	7183#	7196	7197#	7220	7221#	7229	7230#	7238	7239#
7262	7263#	7271	7272#	7280	7281#	7304	7305#	7313	7314#	7322	7323#	7346

7347#	7355	7356#	7364	7365#	7388	7389#	7397	7398#	7406	7407#	7430	7431#
7439	7440#	7448	7449#	7477	7478#	7501	7502#	7524	7525#	7534	7535#	7541
7542#	7566	7567#	7572	7573#	7592	7593#	7602	7603#	7666	7667#	7727	7728#
7750	7751#	7779	7780#	7818	7819#	7841	7842#	7915	7916#	8003	8004#	8007
8008#	8011	8012#	8015	8016#	8019	8020#	8023	8024#	8027	8028#	8031	8032#
117#	125*											
38#	73											
50#	117											
31#												
68#												
29#	69	73										
69#												
36#												
37#												
30#	126*											
33#	120*	7856*	7861	7887								
71#												
18#	23											
1#												
41#												
32#	112	118	123	562	566	583	588	7665	7730			
1#	127	133#	165	189	195#	198	208	214#	217	227	233#	236
246	252#	254	279	285#	289	299	305#	308	318	328	334#	341
352	358#	362	372	378#	385	423	429#	432	442	448#	450	460
466#	468	478	484#	486	496	502#	509	520	526#	534	544	550#
568	574#	590	596#	603	614	620#	628	639	645#	652	663	669#
677	689	695#	702	713	719#	727	738	744#	751	762	768#	776
802	808#	811	821	827#	829	839	845#	847	857	863#	865	889
895#	912	936	942#	959	983	989#	1006	1030	1036#	1053	1099	1105#
1126	1146	1152#	1171	1188	1194#	1206	1225	1231#	1245	1263	1269#	1287
1308	1314#	1334	1355	1361#	1382	1403	1409#	1430	1446	1452#	1476	1492
1498#	1539	1548	1554#	1595	1604	1610#	1641	1665	1671#	1689	1712	1718#
1742	1766	1772#	1794	1804	1810#	1862	1880	1886#	1905	1931	1937#	1954
1973	1979#	1994	2014	2020#	2036	2047	2053#	2079	2088	2094#	2125	2134
2140#	2160	2169	2175#	2196	2214	2220#	2234	2254	2260#	2267	2286	2292#
2299	2318	2324#	2332	2350	2356#	2381	2399	2405#	2424	2442	2448#	2492
2509	2515#	2536	2555	2561#	2598	2615	2621#	2639	2656	2662#	2683	2700
2706#	2725	2742	2748#	2767	2783	2789#	2794	2810	2816#	2821	2837	2843#
2860	2878	2884#	2936	2951	2957#	2977	2992	2998#	3043	3057	3063#	3083
3100	3106#	3112	3128	3134#	3140	3157	3163#	3169	3190	3196#	3214	3231
3237#	3244	3261	3267#	3282	3301	3307#	3321	3337	3343#	3360	3377	3383#
3387	3403	3409#	3413	3429	3435#	3439	3449	3455#	3472	3482	3488#	3505
3515	3521#	3548	3558	3564#	3614	3625	3631#	3676	3686	3692#	3720	3730
3736#	3784	3794	3800#	3828	3838	3844#	3871	3881	3887#	3914	3930	3936#
3953	3969	3975#	4002	4017	4023#	4068	4083	4089#	4134	4150	4156#	4181
4199	4205#	4249	4268	4274#	4319	4337	4343#	4388	4405	4411#	4455	4477
4483#	4489	4515	4521#	4527	4552	4558#	4564	4583	4589#	4595	4612	4618#
4639	4659	4665#	4704	4722	4728#	4773	4791	4797#	4830	4849	4855#	4863
4884	4890#	4899	4918	4924#	4929	4947	4953#	4959	4979	4985#	4996	5013
5019#	5023	5041	5047#	5060	5078	5084#	5087	5105	5111#	5124	5144	5150#
5163	5186	5192#	5196	5219	5225#	5229	5274	5280#	5382	5409	5415#	5546
5565	5571#	5581	5604	5610#	5630	5640	5646#	5667	5677	5683#	5703	5713
5719#	5741	5763	5769#	5806	5817	5823#	5873	5894	5900#	5906	5917	5923#
5944	5954	5960#	5981	6001	6007#	6067	6078	6084#	6120	6141	6147#	6186
6197	6203#	6254	6265	6271#	6277	6299	6305#	6356	6366	6372#	6424	6445
6451#	6503	6513	6519#	6570	6580	6586#	6639	6649	6655#	6711	6732	6738#

\$ERROR= 000302  
\$ETABL 000320  
\$ETEND 000330  
\$FATAL 000302  
\$HIBTS 000330  
\$MAIL 000300  
\$MBADR 000332  
\$MSGAD 000314  
\$MSGLG 000316  
\$MSGTY 000300  
\$PASS 000306  
\$PASTM 000336  
\$SVPC = 000400  
\$SWR = 000000  
\$SWREG 000322  
\$TESTN 000304  
\$TN = 000265

\$TSTM 000334  
\$TSTM= 000304  
\$UNIT 000312  
\$UNITM 000340  
\$USWR 000324  
\$X = 025542

6767	6785	6791#	6819	6836	6842#	6863	6880	6886#	6923	6943	6949#	6968
6979	6985#	6999	7009	7015#	7030	7040	7046#	7060	7070	7076#	7090	7100
7106#	7120	7130	7136#	7150	7168	7174#	7191	7201	7207#	7233	7243	7249#
7275	7285	7291#	7317	7327	7333#	7359	7369	7375#	7401	7411	7417#	7443
7461	7467#	7472	7488	7494#	7513	7519#	7553	7559#	7577	7583#	7597	7631
7637#	7688	7694#	7738	7744#	7761	7767#	7773	7797	7803#	7848	7854#	
70#												
118#	124*											
35#												
72#												
42#												
133#	148	168	195#	201	214#	220	233#	239	252#	257	285#	292
305#	311	321	334#	344	358#	365	378#	388	429#	435	448#	453
466#	471	484#	489	502#	512	526#	537	550#	574#	596#	606	620#
631	645#	655	669#	680	695#	705	719#	730	744#	754	768#	779
808#	814	827#	832	845#	850	863#	868	895#	902	915	942#	949
962	989#	996	1009	1036#	1043	1056	1105#	1109	1120	1129	1152#	1157
1174	1194#	1198	1209	1231#	1236	1248	1269#	1276	1290	1314#	1322	1337
1361#	1370	1385	1409#	1418	1433	1452#	1462	1479	1498#	1507	1518	1532
1542	1554#	1564	1575	1588	1598	1610#	1620	1635	1644	1671#	1679	1692
1718#	1728	1745	1772#	1781	1797	1810#	1824	1834	1845	1854	1865	1886#
1893	1908	1937#	1944	1957	1979#	1986	1997	2020#	2028	2039	2053#	2063
2073	2082	2094#	2108	2118	2128	2140#	2153	2163	2175#	2186	2199	2220#
2225	2237	2260#	2270	2292#	2302	2324#	2335	2356#	2367	2384	2405#	2416
2427	2448#	2459	2469	2484	2495	2515#	2528	2539	2561#	2575	2590	2601
2621#	2632	2642	2662#	2675	2686	2706#	2718	2728	2748#	2760	2770	2789#
2797	2816#	2824	2843#	2853	2863	2884#	2889	2902	2913	2928	2939	2957#
2969	2980	2998#	3005	3014	3024	3036	3046	3063#	3074	3086	3106#	3115
3134#	3143	3163#	3172	3196#	3207	3217	3237#	3247	3267#	3275	3285	3307#
3315	3324	3343#	3353	3363	3383#	3390	3409#	3416	3435#	3442	3455#	3465
3475	3488#	3498	3508	3521#	3530	3541	3551	3564#	3575	3585	3597	3608
3617	3631#	3645	3655	3668	3679	3692#	3704	3714	3723	3736#	3748	3758
3768	3778	3787	3800#	3812	3822	3831	3844#	3855	3865	3874	3887#	3898
3908	3917	3936#	3946	3956	3975#	3984	3995	4005	4023#	4031	4041	4052
4062	4071	4089#	4098	4108	4117	4127	4137	4156#	4165	4175	4184	4205#
4213	4222	4233	4243	4252	4274#	4283	4293	4302	4313	4322	4343#	4352
4362	4371	4382	4391	4411#	4420	4430	4439	4449	4458	4483#	4492	4521#
4530	4558#	4567	4589#	4598	4618#	4627	4642	4665#	4675	4690	4707	4728#
4740	4757	4776	4797#	4805	4819	4833	4855#	4866	4890#	4902	4924#	4932
4953#	4962	4985#	4990	4999	5019#	5026	5047#	5054	5063	5084#	5090	5111#
5118	5127	5150#	5157	5166	5192#	5199	5225#	5232	5280#	5287	5296	5310
5322	5331	5344	5357	5371	5385	5415#	5443	5462	5480	5498	5514	5530
5549	5571#	5584	5610#	5618	5633	5646#	5655	5670	5683#	5692	5706	5719#
5729	5744	5769#	5778	5794	5809	5823#	5832	5846	5860	5876	5900#	5909
5923#	5932	5947	5960#	5969	5984	6007#	6015	6029	6043	6055	6070	6084#
6093	6108	6123	6147#	6157	6173	6189	6203#	6212	6227	6243	6257	6271#
6280	6305#	6314	6329	6344	6359	6372#	6382	6397	6412	6427	6451#	6461
6476	6490	6506	6519#	6529	6544	6559	6573	6586#	6596	6611	6625	6642
6655#	6665	6681	6696	6714	6738#	6750	6770	6791#	6804	6822	6842#	6850
6866	6886#	6905	6926	6949#	6956	6971	6985#	6993	7002	7015#	7024	7033
7046#	7054	7063	7076#	7084	7093	7106#	7114	7123	7136#	7144	7153	7174#
7180	7194	7207#	7217	7227	7236	7249#	7259	7269	7278	7291#	7301	7311
7320	7333#	7343	7353	7362	7375#	7385	7395	7404	7417#	7427	7437	7446
7467#	7475	7494#	7519#	7532	7559#	7583#	7600	7637#	7694#	7724	7744#	7767#
7776	7803#	7816	7838	7854#								
148#	168#	201#	220#	239#	257#	292#	311#	321#	344#	365#	388#	435#

\$XX = 177716

453#	471#	489#	512#	537#	606#	631#	655#	680#	705#	730#	754#	779#
814#	832#	850#	868#	902#	915#	949#	962#	996#	1009#	1043#	1056#	1109#
1120#	1129#	1157#	1174#	1198#	1209#	1236#	1248#	1276#	1290#	1322#	1337#	1370#
1385#	1418#	1433#	1462#	1479#	1507#	1518#	1532#	1542#	1564#	1575#	1588#	1598#
1620#	1635#	1644#	1679#	1692#	1728#	1745#	1781#	1797#	1824#	1834#	1845#	1854#
1865#	1893#	1908#	1944#	1957#	1986#	1997#	2028#	2039#	2063#	2073#	2082#	2108#
2118#	2128#	2153#	2163#	2186#	2199#	2225#	2237#	2270#	2302#	2335#	2367#	2384#
2416#	2427#	2459#	2469#	2484#	2495#	2528#	2539#	2575#	2590#	2601#	2632#	2642#
2675#	2686#	2718#	2728#	2760#	2770#	2797#	2824#	2853#	2863#	2889#	2902#	2913#
2928#	2939#	2969#	2980#	3005#	3014#	3024#	3036#	3046#	3074#	3086#	3115#	3143#
3172#	3207#	3217#	3247#	3275#	3285#	3315#	3324#	3353#	3363#	3390#	3416#	3442#
3465#	3475#	3498#	3508#	3530#	3541#	3551#	3575#	3585#	3597#	3608#	3617#	3645#
3655#	3668#	3679#	3704#	3714#	3723#	3748#	3758#	3768#	3778#	3787#	3812#	3822#
3831#	3855#	3865#	3874#	3898#	3908#	3917#	3946#	3956#	3984#	3995#	4005#	4031#
4041#	4052#	4062#	4071#	4098#	4108#	4117#	4127#	4137#	4165#	4175#	4184#	4213#
4222#	4233#	4243#	4252#	4283#	4293#	4302#	4313#	4322#	4352#	4362#	4371#	4382#
4391#	4420#	4430#	4439#	4449#	4458#	4492#	4530#	4567#	4598#	4627#	4642#	4675#
4690#	4707#	4740#	4757#	4776#	4805#	4819#	4833#	4866#	4902#	4932#	4962#	4990#
4999#	5026#	5054#	5063#	5090#	5118#	5127#	5157#	5166#	5199#	5232#	5287#	5296#
5310#	5322#	5331#	5344#	5357#	5371#	5385#	5443#	5462#	5480#	5498#	5514#	5530#
5549#	5584#	5618#	5633#	5655#	5670#	5692#	5706#	5729#	5744#	5778#	5794#	5809#
5832#	5846#	5860#	5876#	5909#	5932#	5947#	5969#	5984#	6015#	6029#	6043#	6055#
6070#	6093#	6108#	6123#	6157#	6173#	6189#	6212#	6227#	6243#	6257#	6280#	6314#
6329#	6344#	6359#	6382#	6397#	6412#	6427#	6461#	6476#	6490#	6506#	6529#	6544#
6559#	6573#	6596#	6611#	6625#	6642#	6665#	6681#	6696#	6714#	6750#	6770#	6804#
6822#	6850#	6866#	6905#	6926#	6956#	6971#	6993#	7002#	7024#	7033#	7054#	7063#

\$XXX = 000716







	6625	6642	6665	6681	6696	6714	6750	6770	6804	6822	6850	6866	6905	6926	6956
	6971	6993	7002	7024	7033	7054	7063	7084	7093	7114	7123	7144	7153	7180	7194
	7217	7227	7236	7259	7269	7278	7301	7311	7320	7343	7353	7362	7385	7395	7404
	7427	7437	7446	7475	7532	7600	7724	7776	7816	7838					
MULT	1#														
NEWTST	1#														
	496	127	189	208	227	246	279	299	328	352	372	423	442	460	478
	857	520	544	568	590	614	639	663	689	713	738	762	802	821	839
	1548	889	936	983	1030	1099	1146	1188	1225	1263	1308	1355	1403	1446	1492
	2254	1604	1665	1712	1766	1804	1880	1931	1973	2014	2047	2088	2134	2169	2214
	2878	2286	2318	2350	2399	2442	2509	2555	2615	2656	2700	2742	2783	2810	2837
	3449	2951	2992	3057	3100	3128	3157	3190	3231	3261	3301	3337	3377	3403	3429
	4199	3482	3515	3558	3625	3686	3730	3794	3838	3881	3930	3969	4017	4083	4150
	4947	4268	4337	4405	4477	4515	4552	4583	4612	4659	4722	4791	4849	4884	4918
	5713	4979	5013	5041	5078	5105	5144	5186	5219	5274	5409	5565	5604	5640	5677
	6580	5763	5817	5894	5917	5954	6001	6078	6141	6197	6265	6299	6366	6445	6513
	7243	6649	6732	6785	6836	6880	6943	6979	7009	7040	7070	7100	7130	7168	7201
	7848	7285	7327	7369	7411	7461	7488	7513	7553	7577	7631	7688	7738	7761	7797
POP	1#														
PUSH	1#														
REPORT	1#														
SETPRI	1#														
SETUP	1#														
SKIP	1#														
SLASH	1#														
STARS	1#														
		16	27	54	56	63	74	76	97	99	105	127	129	175	189
	191	208	210	227	229	246	248	264	279	281	299	301	328	330	352
	354	372	374	396	423	425	442	444	460	462	478	480	496	498	520
	522	544	546	568	570	590	592	614	616	639	641	663	665	689	691
	713	715	738	740	762	764	787	802	804	821	823	839	841	857	859
	876	889	891	923	936	938	970	983	985	1017	1030	1032	1064	1082	1086
	1099	1101	1137	1146	1148	1182	1188	1190	1217	1225	1227	1256	1263	1265	1299
	1308	1310	1345	1355	1357	1393	1403	1405	1441	1446	1448	1487	1492	1494	1548
	1550	1604	1606	1651	1665	1667	1700	1712	1714	1753	1766	1768	1804	1806	1872
	1880	1882	1916	1931	1933	1965	1973	1975	2005	2014	2016	2047	2049	2088	2090
	2134	2136	2169	2171	2206	2214	2216	2246	2254	2256	2278	2286	2288	2310	2318
	2320	2343	2350	2352	2392	2399	2401	2434	2442	2444	2502	2509	2511	2546	2555
	2557	2607	2615	2617	2649	2656	2658	2693	2700	2702	2735	2742	2744	2777	2783
	2785	2804	2810	2812	2831	2837	2839	2870	2878	2880	2946	2951	2953	2986	2992
	2994	3052	3057	3059	3093	3100	3102	3122	3128	3130	3150	3157	3159	3179	3190
	3192	3224	3231	3233	3254	3261	3263	3292	3301	3303	3330	3337	3339	3369	3377
	3379	3396	3403	3405	3422	3429	3431	3449	3451	3482	3484	3515	3517	3558	3560
	3625	3627	3686	3688	3730	3732	3794	3796	3838	3840	3881	3883	3924	3930	3932
	3963	3969	3971	4012	4017	4019	4078	4083	4085	4143	4150	4152	4191	4199	4201
	4259	4268	4270	4329	4337	4339	4398	4405	4407	4465	4477	4479	4506	4515	4517
	4542	4552	4554	4573	4583	4585	4604	4612	4614	4650	4659	4661	4715	4722	4724
	4784	4791	4793	4841	4849	4851	4874	4884	4886	4911	4918	4920	4940	4947	4949
	4972	4979	4981	5006	5013	5015	5034	5041	5043	5071	5078	5080	5098	5105	5107
	5135	5144	5146	5177	5186	5188	5209	5219	5221	5243	5274	5276	5393	5409	5411
	5558	5565	5567	5590	5604	5606	5640	5642	5677	5679	5713	5715	5752	5763	5765
	5817	5819	5885	5894	5896	5917	5919	5954	5956	5991	6001	6003	6078	6080	6131
	6141	6143	6197	6199	6265	6267	6289	6299	6301	6366	6368	6435	6445	6447	6513
	6515	6580	6582	6649	6651	6724	6732	6734	6777	6785	6787	6829	6836	6838	6873
	6880	6882	6934	6943	6945	6979	6981	7009	7011	7040	7042	7070	7072	7100	7102
	7130	7132	7160	7168	7170	7201	7203	7243	7245	7285	7287	7327	7329	7369	7371
	7411	7413	7453	7461	7463	7483	7488	7490	7506	7513	7515	7548	7553	7555	7577

	7579	7608	7631	7633	7681	7688	7690	7732	7738	7740	7755	7761	7763	7783	7797
	7799	7848	7850	7996	8000										
SWRSU	1#														
TYPBIN	1#														
TYPDEC	1#														
TYPNAM	1#														
TYPNUM	1#														
TYPOCS	1#														
TYP OCT	1#														
TYP TXT	1#														
\$SERCD	1#	141	150	161	170	203	222	241	259	294	313	323	347	367	391
	437	455	473	491	515	540	563	585	609	634	658	683	708	733	757
	782	816	834	852	870	905	918	952	965	999	1012	1046	1059	1111	1122
	1131	1159	1177	1200	1212	1238	1251	1278	1293	1324	1340	1372	1388	1420	1436
	1464	1482	1510	1520	1535	1544	1567	1577	1591	1600	1623	1637	1646	1681	1695
	1730	1748	1783	1800	1827	1836	1847	1856	1867	1895	1911	1946	1960	1988	2000
	2030	2042	2066	2075	2084	2111	2120	2130	2156	2165	2189	2201	2227	2240	2273
	2305	2338	2370	2387	2419	2429	2462	2471	2487	2497	2531	2541	2578	2593	2603
	2635	2644	2678	2688	2721	2730	2763	2772	2799	2826	2856	2865	2891	2904	2915
	2930	2941	2972	2982	3007	3016	3026	3038	3048	3077	3089	3117	3145	3174	3209
	3219	3249	3277	3287	3317	3326	3355	3365	3392	3418	3444	3468	3477	3501	3510
	3533	3544	3553	3578	3587	3600	3610	3619	3648	3657	3671	3681	3707	3716	3725
	3751	3760	3771	3780	3789	3815	3824	3833	3858	3867	3876	3901	3910	3919	3949
	3958	3987	3998	4007	4034	4043	4055	4064	4073	4101	4110	4119	4129	4139	4168
	4177	4186	4215	4224	4236	4245	4254	4286	4295	4304	4315	4324	4355	4364	4373
	4384	4393	4423	4432	4441	4451	4460	4495	4533	4569	4600	4630	4645	4678	4693
	4710	4743	4760	4779	4808	4822	4836	4869	4905	4935	4965	4992	5001	5028	5056
	5065	5092	5120	5129	5159	5169	5202	5235	5289	5298	5312	5324	5333	5346	5359
	5373	5387	5427	5446	5465	5483	5501	5517	5533	5552	5577	5586	5621	5636	5658
	5673	5695	5709	5732	5747	5781	5797	5812	5835	5849	5863	5879	5912	5935	5950
	5972	5987	6018	6032	6046	6058	6073	6096	6111	6126	6160	6176	6192	6215	6230
	6246	6260	6283	6317	6332	6347	6362	6385	6400	6415	6430	6464	6479	6493	6509
	6532	6547	6562	6576	6599	6614	6628	6645	6668	6684	6699	6717	6753	6773	6807
	6825	6853	6869	6893	6908	6917	6929	6958	6974	6995	7004	7026	7035	7056	7065
	7086	7095	7116	7125	7146	7155	7182	7196	7220	7229	7238	7262	7271	7280	7304
	7313	7322	7346	7355	7364	7388	7397	7406	7430	7439	7448	7477	7501	7524	7534
	7541	7566	7572	7592	7602	7666	7727	7750	7779	7818	7841	7915	8003	8007	8011
	8015	8019	8023	8027	8031										
\$SERNU	1#	141	150	161	170	203	222	241	259	294	313	323	347	367	391
	437	455	473	491	515	540	563	585	609	634	658	683	708	733	757
	782	816	834	852	870	905	918	952	965	999	1012	1046	1059	1111	1122
	1131	1159	1177	1200	1212	1238	1251	1278	1293	1324	1340	1372	1388	1420	1436
	1464	1482	1510	1520	1535	1544	1567	1577	1591	1600	1623	1637	1646	1681	1695
	1730	1748	1783	1800	1827	1836	1847	1856	1867	1895	1911	1946	1960	1988	2000
	2030	2042	2066	2075	2084	2111	2120	2130	2156	2165	2189	2201	2227	2240	2273
	2305	2338	2370	2387	2419	2429	2462	2471	2487	2497	2531	2541	2578	2593	2603
	2635	2644	2678	2688	2721	2730	2763	2772	2799	2826	2856	2865	2891	2904	2915
	2930	2941	2972	2982	3007	3016	3026	3038	3048	3077	3089	3117	3145	3174	3209
	3219	3249	3277	3287	3317	3326	3355	3365	3392	3418	3444	3468	3477	3501	3510
	3533	3544	3553	3578	3587	3600	3610	3619	3648	3657	3671	3681	3707	3716	3725
	3751	3760	3771	3780	3789	3815	3824	3833	3858	3867	3876	3901	3910	3919	3949
	3958	3987	3998	4007	4034	4043	4055	4064	4073	4101	4110	4119	4129	4139	4168
	4177	4186	4215	4224	4236	4245	4254	4286	4295	4304	4315	4324	4355	4364	4373
	4384	4393	4423	4432	4441	4451	4460	4495	4533	4569	4600	4630	4645	4678	4693
	4710	4743	4760	4779	4808	4822	4836	4869	4905	4935	4965	4992	5001	5028	5056
	5065	5092	5120	5129	5159	5169	5202	5235	5289	5298	5312	5324	5333	5346	5359

5373	5387	5427	5446	5465	5483	5501	5517	5533	5552	5577	5586	5621	5636	5658
5673	5695	5709	5732	5747	5781	5797	5812	5835	5849	5863	5879	5912	5935	5950
5972	5987	6018	6032	6046	6058	6073	6096	6111	6126	6160	6176	6192	6215	6230
6246	6260	6283	6317	6332	6347	6362	6385	6400	6415	6430	6464	6479	6493	6509
6532	6547	6562	6576	6599	6614	6628	6645	6668	6684	6699	6717	6753	6773	6807
6825	6853	6869	6893	6908	6917	6929	6958	6974	6995	7004	7026	7035	7056	7065
7086	7095	7116	7125	7146	7155	7182	7196	7220	7229	7238	7262	7271	7280	7304
7313	7322	7346	7355	7364	7388	7397	7406	7430	7439	7448	7477	7501	7524	7534
7541	7566	7572	7592	7602	7666	7727	7750	7779	7818	7841	7915	8003	8007	8011
8015	8019	8023	8027	8031										
1#	165	198	217	236	254	289	308	318	341	362	385	432	450	468
486	509	534	603	628	652	677	702	727	751	776	811	829	847	865
912	959	1006	1053	1126	1171	1206	1245	1287	1334	1382	1430	1476	1539	1595
1641	1689	1742	1794	1862	1905	1954	1994	2036	2079	2125	2160	2196	2234	2267
2299	2332	2381	2424	2492	2536	2598	2639	2683	2725	2767	2794	2821	2860	2936
2977	3043	3083	3112	3140	3169	3214	3244	3282	3321	3360	3387	3413	3439	3472
3505	3548	3614	3676	3720	3784	3828	3871	3914	3953	4002	4068	4134	4181	4249
4319	4388	4455	4489	4527	4564	4595	4639	4704	4773	4830	4863	4899	4929	4959
4996	5023	5060	5087	5124	5163	5196	5229	5382	5546	5581	5630	5667	5703	5741
5806	5873	5906	5944	5981	6067	6120	6186	6254	6277	6356	6424	6503	6570	6639
6711	6767	6819	6863	6923	6968	6999	7030	7060	7090	7120	7150	7191	7233	7275
7317	7359	7401	7443	7472	7597	7773								
1#	148	168	201	220	239	257	292	311	321	344	365	388	435	453
471	489	512	537	606	631	655	680	705	730	754	779	814	832	850
868	902	915	949	962	996	1009	1043	1056	1109	1120	1129	1157	1174	1198
1209	1236	1248	1276	1290	1322	1337	1370	1385	1418	1433	1462	1479	1507	1518
1532	1542	1564	1575	1588	1598	1620	1635	1644	1679	1692	1728	1745	1781	1797
1824	1834	1845	1854	1865	1893	1908	1944	1957	1986	1997	2028	2039	2063	2073
2082	2108	2118	2128	2153	2163	2186	2199	2225	2237	2270	2302	2335	2367	2384
2416	2427	2459	2469	2484	2495	2528	2539	2575	2590	2601	2632	2642	2675	2686
2718	2728	2760	2770	2797	2824	2853	2863	2889	2902	2913	2928	2939	2969	2980
3005	3014	3024	3036	3046	3074	3086	3115	3143	3172	3207	3217	3247	3275	3285
3315	3324	3353	3363	3390	3416	3442	3465	3475	3498	3508	3530	3541	3551	3575
3585	3597	3608	3617	3645	3655	3668	3679	3704	3714	3723	3748	3758	3768	3778
3787	3812	3822	3831	3855	3865	3874	3898	3908	3917	3946	3956	3984	3995	4005
4031	4041	4052	4062	4071	4098	4108	4117	4127	4137	4165	4175	4184	4213	4222
4233	4243	4252	4283	4293	4302	4313	4322	4352	4362	4371	4382	4391	4420	4430
4439	4449	4458	4492	4530	4567	4598	4627	4642	4675	4690	4707	4740	4757	4776
4805	4819	4833	4866	4902	4932	4962	4990	4999	5026	5054	5063	5090	5118	5127
5157	5166	5199	5232	5287	5296	5310	5322	5331	5344	5357	5371	5385	5443	5462
5480	5498	5514	5530	5549	5584	5618	5633	5655	5670	5692	5706	5729	5744	5778
5794	5809	5832	5846	5860	5876	5909	5932	5947	5969	5984	6015	6029	6043	6055
6070	6093	6108	6123	6157	6173	6189	6212	6227	6243	6257	6280	6314	6329	6344
6359	6382	6397	6412	6427	6461	6476	6490	6506	6529	6544	6559	6573	6596	6611
6625	6642	6665	6681	6696	6714	6750	6770	6804	6822	6850	6866	6905	6926	6956
6971	6993	7002	7024	7033	7054	7063	7084	7093	7114	7123	7144	7153	7180	7194
7217	7227	7236	7259	7269	7278	7301	7311	7320	7343	7353	7362	7385	7395	7404
7427	7437	7446	7475	7532	7600	7724	7776	7816	7838					
1#	127	189	208	227	246	279	299	328	352	372	423	442	460	478
496	520	544	568	590	614	639	663	689	713	738	762	802	821	839
857	889	936	983	1030	1099	1146	1188	1225	1263	1308	1355	1403	1446	1492
1548	1604	1665	1712	1766	1804	1880	1931	1973	2014	2047	2088	2134	2169	2214
2254	2286	2318	2350	2399	2442	2509	2555	2615	2656	2700	2742	2783	2810	2837
2878	2951	2992	3057	3100	3128	3157	3190	3231	3261	3301	3337	3377	3403	3429
3449	3482	3515	3558	3625	3686	3730	3794	3838	3881	3930	3969	4017	4083	4150

4199	4268	4337	4405	4477	4515	4552	4583	4612	4659	4722	4791	4849	4884	4918
4947	4979	5013	5041	5078	5105	5144	5186	5219	5274	5409	5565	5604	5640	5677
5713	5763	5817	5894	5917	5954	6001	6078	6141	6197	6265	6299	6366	6445	6513
6580	6649	6732	6785	6836	6880	6943	6979	7009	7040	7070	7100	7130	7168	7201
7243	7285	7327	7369	7411	7461	7488	7513	7553	7577	7631	7688	7738	7761	7797
7848														

- \$\$\$SKIP 1#
- .EQUAT 1#
- .HEADE 1#
- .KT11 1#
- .SETUP 1#
- .SWRHI 1#
- .\$ACT1 1#
- .\$APT8 1#
- .\$APTH 1#
- .\$APTY 1#
- .\$ASTA 1#
- .\$CATC 1#
- .\$CMTA 1#
- .\$DB2D 1#
- .\$DB20 1#
- .\$DIV 1#
- .\$EOP 1#
- .\$ERRO 1#
- .\$ERRT 1#
- .\$MULT 1#
- .\$POWE 1#
- .\$RAND 1#
- .\$RDDE 1#
- .\$RDOG 1#
- .\$READ 1#
- .\$R2AZ 1#
- .\$SAVE 1#
- .\$SB2D 1#
- .\$SB20 1#
- .\$SCOP 1#
- .\$SIZE 1#
- .\$SUPR 1#
- .\$TRAP 1#
- .\$TYPB 1#
- .\$TYPD 1#
- .\$TYPE 1#
- .\$TYPO 1#
- .\$4OCA 1#
- .1170 1#

14#  
14# 25  
14# 52

. ABS. 026524 000

ERRORS DETECTED: 0

CFKAAC.BIN,CFKAAC.LST/CRF/SOL=CFKAAC.SML,CFKAAC.P11  
 RUN-TIME: 30 40 3 SECONDS  
 RUN-TIME RATIO: 170/74=2.2  
 CORE USED: 33K (65 PAGES)

CFKAACO 11/34 BSC INST TST  
CFKAAC.P11 18-OCT-78 11:01

MACY11 30A(1052) 18-OCT-78 11:06 PAGE 201  
E 1  
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0211