CONTENTS

1. ABSTRACT
2. REQUIREMENTS
2.1 EQUIPMENT
2.2 STORAGE
2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
4. STARTING PROCEDURE
4.1 STARTING ADDRESS
4.2 PROGRAM AND/OR OPERATOR ACTION
5. OPERATING PROCEDURE
6. ERRORS
6.1 ERROR REPORTING
6.2 ERROR RECOVERY
7. RESTRICTIONS
8. MISCELLANEOUS
8.1 EXECUTION TIME
8.2 STACK POINTER
8.3 PASS COUNTER
8.4 TEST NUMBER
8.5 POWER FAIL
9. PROGRAM DESCRIPTION
1. ABSTRACT

THIS PROGRAM TESTS THE LSI-11 BASIC INSTRUCTION SET IN ALL MODES. THE DIAGNOSTIC IS DESIGNED TO RUN UNDER BOTH APT. AND ACT. SYSTEMS.

2. REQUIREMENTS

2.1 EQUIPMENT

LSI-11 STANDARD COMPUTER AND 4K OF MEMORY

2.2 STORAGE

PROGRAM STORAGE - THE ROUTINES USE MEMORY 0 - 17500

2.3 PRELIMINARY PROGRAMS

NONE

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

4. STARTING PROCEDURE

4.1 STARTING ADDRESS

AFTER LOADING THE PROGRAM IT SHOULD ALWAYS BE STARTED AT 200. IF IT IS DESIRED TO SAVE THE PASS COUNTER THEN CLEAR THE LOCATION $F8TSTN (I.E., LOCATION 102) AND RESTART FROM 450 OTHERWISE THE PROGRAM CAN BE RESTARTED AT 200. IF IT IS DESIRED TO GO TO A TEST OTHER THAN TEST #0 THEN PLACE THE TEST NUMBER IN LOCATION $FSTTN AND RESTART THE PROGRAM AT 450. IN WHICH CASE THE PROGRAM WILL HALT AT LOCATION 666 AND WILL WAIT FOR THE OPERATOR TO PLACE THE STARTING ADDRESS OF THE DESIRED TEST IN PC (R7) AND TYPE A P.
4.2 PROGRAM AND/OR OPERATOR ACTION

3) PLACE L.T.C SWITCH IN OFF POSITION.
4) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
3) TYPE 206G [ THERE ARE NO SWITCH SETTINGS AND THIS DIAGNOSTIC
   DOES NOT USE SOFTWARE SWITCH LOCATION $S_WREG ]
4) THE PROGRAM WILL LOOP AND "END PASS" WILL BE TYPED AFTER
   THE FIRST PASS AND THEN EVERY 377 PASSES, HOWEVER TYPE OUT
   WILL BE SUPPRESSED IF BIT 5 OF LOCATION $SENUM IS HIGH
5) A MINIMUM OF TWO PASSES SHOULD ALWAYS BE RUN.

5. OPERATING PROCEDURE

5.1 OPERATING MODE:

AN 8 BIT BYTE $SENUM [ I.E. LOCATION 117 ] HAS BEEN USED TO DEFINE
THE OPERATING MODE. ALL TYPEOUTS CAN BE SUPPRESSED BY MAKING
BIT 5 OF BYTE $SENUM HIGH, IN OTHER WORDS BY PLACING A 20000 IN
LOCATION 116.

5.2 TRAP CATCHER

A "..+2" - "HALT" SEQUENCE IS REPEATED FROM O-776 TO CATCH
ANY UNEXPECTED TRAPS. THIS WAY UNEXPECTED TRAPS OR
INTERRUPTS WILL HALT AT THE VECTOR +2.

6. ERRORS

6.1 ERROR REPORTING

ON FINDING AN ERROR THE PROCESSOR WILL COME TO A HALT AFTER
PLACING THE ERROR NUMBER IN LOCATION $FATAL [ I.E. LOCATION 100].
IN MOST CASES THE COMMENTS BEHIND THE HALT TELL WHAT WAS BEING
CHECKED. IN SOME CASES THE TEST CAN GET TO A HALT VIA 2 WAYS:

* ) WRONG TEST SEQUENCE
2) ERROR IN ACTUAL TEST

WHEN A HALT DOES OCCUR IT IS RECOMMENDED THAT THE TEST SEQUENCE
LOCATION [ I.E. LOCATION 102 ] BE CHECKED TO VERIFY THAT IT MATCHES
THE PRESENT TEST NUMBER. IF IT DOESN'T, THEN THE HALT OCCURRED
BECAUSE THE TEST SEQUENCE WAS NOT CORRECT OTHERWISE THE HALT IS
DUE TO AN ERROR IN THE TEST.
6.2 ERROR RECOVERY
RESTART AT 200 OR 450 (SEE SEC 4.1)

7. RESTRICTIONS
NONE

8. MISCELLANEOUS

9. EXECUTION TIME
EXECUTION TIME OF THE DIAGNOSTIC IS LESS THAN A SECOND, FIRST
"END PASS" WILL BE TYPED OUT WITHIN A SECOND AND EVERY CONSECUTIVE
"END PASS" WILL BE TYPED OUT WITHIN 20 SECONDS (SEE SEC 4.2)

WHEN RUNNING UNDER APT IN A SCRIPT, THE FIRST PASS RUN TIME
AND SUBSEQUENT PASS RUN TIMES ARE ONE (1) SECOND.

8.2 STACK POINTER
STACK IS INITIALLY SET TO 450

8.3 PASS COUNT
A 16 BIT LOCATION "5PASS" (I.E., LOCATION 104) IS USED TO KEEP
PASS COUNT. IT CAN BE CLEARED BY RESTARTING THE PROGRAM AT 200

8.4 TEST NUMBER
A 16 BIT LOCATION "5TEST" (I.E., LOCATION 102) IS USED TO KEEP TRACK
OF THE TEST NUMBER, UPPER BYTE OF THIS LOCATION GIVES THE ITERATION
NUMBER AND THE LOWER BYTE THE TEST THAT WAS BEING EXECUTED

8.5 POWER FAIL
THE DIAGNOSTIC CAN BE POWER FAILED WITH NO ERRORS. TO USE,
START THE DIAGNOSTIC AS USUAL AND POWER DOWN THEN UP AT ANY TIME.
THE PROGRAM SHOULD TYPE "POWER" AND RESTART AT 450 WITH TEST #b.
HOWEVER THE DIAGNOSTIC WILL NOT RECOVER IF IT IS STORED IN A
MEMORY NOT CAPABLE OF HOLDING DATA WITH POWER DOWN

8. PROGRAM DESCRIPTION
THIS PROGRAM TESTS ALL THE BASIC INSTRUCTIONS OF THE LSI-11 (EXCEPT TRAP-TYPE) WHICH INCLUDES CONTROL CHIP, DATA CHIP, MICROMS, PLA, AND OTHER CIRCUITRY ON THE LSI-11 CPU MODULE. TRAP DIAGNOSTIC SHOULD ALSO BE RUN TO MAKE SURE THAT THE BASIC LSI-11 IS FUNCTIONAL. THIS DIAGNOSTIC DOES NOT MAKE A PASS WITH T-B1 SET.
ABS
:: LSI-11 MACRO INSTRUCTION EXERCISER
:: LIST MC,MD,END
:: LIST ME

TITLEVKAA
:: COPYRIGHT (C) 1975, 1978
:: DIGITAL EQUIPMENT CORP.
:: MAYNARD, MASS. 01754
::
:: PROGRAM BY PERVEZ ZAKI
::
:: THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYMEX
:: PACKAGE (MAINDEC-11-D20A(-C3)), JAN 19, 1977.
::
:: $'N':
:: $WR-16000C :: HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
::: TRAP CATCHERS OF +2 AND HALT IN LOCATIONS 0 THRU 776 [IT IS NOT LISTED]

.SBTL  ACT11 HOOKS

::: HOOKS REQUIRED BY ACT11

;SAVE PC
$VPC=.
.46
$ENAD
;1) SET LOC 46 TO ADDRESS OF $END IN .SEQ.
.52
;2) SET LOC 52 TO ZERO
.WORD  0
.$VPC
; RESTORE PC

<table>
<thead>
<tr>
<th>Location</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>000076</td>
<td>000000</td>
</tr>
<tr>
<td>000077</td>
<td>000000</td>
</tr>
<tr>
<td>000078</td>
<td>000000</td>
</tr>
<tr>
<td>000079</td>
<td>000000</td>
</tr>
<tr>
<td>000080</td>
<td>000000</td>
</tr>
<tr>
<td>000081</td>
<td>000000</td>
</tr>
<tr>
<td>000082</td>
<td>000000</td>
</tr>
<tr>
<td>000083</td>
<td>000000</td>
</tr>
<tr>
<td>000084</td>
<td>000000</td>
</tr>
<tr>
<td>000085</td>
<td>000000</td>
</tr>
<tr>
<td>000086</td>
<td>000000</td>
</tr>
<tr>
<td>000087</td>
<td>000000</td>
</tr>
<tr>
<td>000088</td>
<td>000000</td>
</tr>
<tr>
<td>000089</td>
<td>000000</td>
</tr>
<tr>
<td>000090</td>
<td>000000</td>
</tr>
<tr>
<td>000091</td>
<td>000000</td>
</tr>
<tr>
<td>000092</td>
<td>000000</td>
</tr>
<tr>
<td>000093</td>
<td>000000</td>
</tr>
<tr>
<td>000094</td>
<td>000000</td>
</tr>
<tr>
<td>000095</td>
<td>254</td>
</tr>
<tr>
<td>000096</td>
<td>1</td>
</tr>
<tr>
<td>000097</td>
<td>260</td>
</tr>
<tr>
<td>000098</td>
<td>263</td>
</tr>
<tr>
<td>000099</td>
<td>273</td>
</tr>
<tr>
<td>000100</td>
<td>0</td>
</tr>
<tr>
<td>000101</td>
<td>0</td>
</tr>
</tbody>
</table>
.SBTTL APT MAILBOX-ETABLE

;EVEN
.SBITL APT MAILBOX
.SMALL: ;APT MAILBOX
.SMSGTY: .WORD AMSGTY ;MESSAGE TYPE CODE
.SFATAL: .WORD AFATAL ;FATAL ERROR NUMBER
.STESTN: .WORD ATESTN ;TEST NUMBER
.SPASS: .WORD APASS ;PASS COUNT
.SDEVCT: .WORD ADEVCT ;DEVICE COUNT
.SUNIT: .WORD AUNIT ;I/O UNIT NUMBER
.SMSTG: .WORD AMSGAD ;MESSAGE ADDRESS
.SMSLG: .WORD AMSSLG ;MESSAGE LENGTH
.SETABLE: ;APT ENVIRONMENT TABLE
.SENV: .BYTE AENV ;ENVIRONMENT BYTE
.SENVM: .BYTE AENVM ;ENVIRONMENT MODE BITS
.SSWREG: .WORD ASWREG ;APT SWITCH REGISTER
.SSWR: .WORD AUSRW ;USER SWITCHES
.SCP: .WORD ACPU ;CPU TYPE OPTIONS
.BITS 15-11=CPU TYPE
.BITS 10=REAL TIME CLOCK
.BITS 9=FLOATING POINT PROCESSOR
.BITS 8-MEMORY MANAGEMENT

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

;SBTTL APT PARAMETER BLOCK

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

;S$X=; ;SAVE CURRENT LOCATION
;=24; ;SET POWER FAIL TO POINT TO START OF PROGRAM
;=200; ;FOR APT START UP
;=44; ;POINT TO APT INDIRECT ADDRESS PTR.
;=APTHDR; ;POINT TO APT HEADER BLOCK
;=$X; ;RESET LOCATION COUNTER

;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

;SAPTHDR:
;SIBITS: .WORD 0 ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
;SMABDR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
;STM: .WORD 1 ;RUN TIME OF LONGEST TEST
;SPASTM: .WORD 1 ;RUN TIME IN SECS. OF 1ST PASS ON "UNIT" (QUICK VERIFY)
;SUIN: .WORD 1 ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
;=S$END-$MAIL? ;END MAILBOX-ETABLE (WORDS)
APT PARAMETER BLOCK

385 000126  \_BAPTHD  
387 000130  \_ADR:   
389 000132  \_ADR+2  
391 000134  \_ADR1:  
393 000136  \_ADR+1  
395 000134  \_ADR2:  
397 000134  \_ADR2+2  
399 000136  \_DUMMY:  
401 000136  \_DUMMY+2  
403 000134  \_TEMP:  
405 000134  \_TEMP+2  
407 000134  \_TEMP1:  
409 000134  \_TEMP1+2  
411 000134  \_TEMP2:  
413 000134  \_TEMP2+2  

; OUTPUT TTY STATUS REGISTER
; OUTPUT BUFFER

400 000144  177564  TPS:  177566
402 000146  177566  TPB:  177566
404 000150  006402  MARK2:  MARK  2
406 000150  005015  062440  042526  ENDPAS:  \_ASCIIZ  <15<<12>" END PASS"
408 000160  050040  051581  000123

406 000160  050040  047520  042527  POWER:  \_ASCIIZ  <15<<12>>/POWER/
410 000176  000020  \_EVEN
412 000200  017004  \_BLKW  19.
414 000200  000000  \_20
415 000222  000000  \_TYPE  0
416
: STARTING OF THE PROGRAM

; : SERVICE POWER DOWN ROUTINE ON ANY FUTURE POWER DOWN
MOV #PWRDN,#24
MOV #ETABLE,RO

2%: CLR -(RO)

; : START CLEANING THE STACK
IMP RO,#$MAIL
; FOR INITIALIZATION

; : FOR INITIALIZATION
JMP START

; 450

START:

; : SET THE STACK POINTER
MOV #START,SP

; : PLACE THE ADDRESS OF LOCATION #$TESTN IN R5
MOV #$TESTN,R5

; : CHECK THE SEQUENCE COUNTER
TST (R5)

; : IF THIS IS THE STARTING OF THE TEST THEN
BEQ NOBIT

; : OTHERWISE HALT AND WAIT FOR THE OPERATOR
; TO START AT THE PROPER TEST NUMBER
HALT
TO CHECK BRANCH INSTRUCTIONS WITH ZERO CONDITION CODES

******
**TEST: 0**

CHECK BRANCH INSTRUCTIONS WITH ZERO CONDITION CODES

NOBIT:

CMP (R5), #0
BNE CCO

: IF IN WRONG SEQUENCE GO TO HALT AT END OF THE TEST

1$: INRC (R5)

: ZERO CONDITION CODES, NZVC=0000

1S:

BNE CCO
BCS CCO
BVS CCO
BEQ CCO
BMI CCO
BPL CCO
BLE CCO
BHS CCO
BGE CCO
BGT CCO
BLO CCO
BLE CCO

: CHECK NOP1 INSTRUCTION I.E. OP-CODE 260

ENDCCO: BVC NOBIT

CCO:

MOV #1, -(R5)
INC -(R5)
HALT

: ONE OF THE ABOVE BRANCHES FAILED OR WRONG SEQUENCE

******
**TEST: 1**

CHECK BRANCH INSTRUCTIONS WITH N BIT SET

******

NBIT:

CMP (R5), #1
BNE CCI

: IF IN WRONG SEQUENCE GO TO HALT AT THE END OF THE TEST

1$: INRC (R5)

: NBIT IS SET, NZVC=1000

1S:

BNE CCI
BPL CCI
BEQ CCI
BGE CCI
BGT CCI
BLO CCI
BLE CCI
BHS CCI
BGE CCI
BGT CCI
BLO CCI
BLE CCI

ENDCCI: BVC NB1T

CCI:

MOV #2, -(R5)
INC -(R5)
HALT

: ONE OF THE ABOVE BRANCHES FAILED OR WRONG SEQUENCE

ENDCCI: BNE VBIT

******
CHECK BRANCH INSTRUCTIONS WITH N VF bits set

**TEST: 2**

VBIT:

1. CMP (RS),#2
2. BNE CC2
   : IF IN WRONG SEQUENCE GO TO HALT AT THE END OF THE TEST
3. INC (RS)
4. V AND N BIT SET, NZVC = 1010

CC2:

1. MOV #3, -(RS)
2. INC -(RS)
3. HALT
4. ONE OF THE ABOVE BRANCHES FAILED OR WRONG SEQUENCE

ENDCC2: BGE CBIT

**TEST: 3**

CBIT:

1. CMP (RS),#3
2. BNE CC3
   : IF IN WRONG SEQUENCE GO TO HALT AT THE END OF THE TEST
3. INC (RS)
4. C, V, AND N BITS ARE SET, NZVC=1011

CC3:

1. MOV #4, -(RS)
2. INC -(RS)
3. HALT
4. ONE OF THE ABOVE BRANCHES FAILED OR WRONG SEQUENCE
**TEST: 4**  CHECK BRANCH INSTRUCTIONS WITH N,Z,V&C BITS SET

**Z817:**
- **CMP (R5),#4**
- **BNE CC4** ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
- **INC (R5)**
- **SEI**  ; ALL BITS SET, NZVC=1111
- **BNE CC4**
- **BPL CC4**
- **BVC CC4**
- **BCC CC4**
- **BLT CC4**
- **BGT CC4**
- **BEQ YESCC**

**CL4:**
- **MOV #5,-(R5)**
- **INC -(R5)** ; ONE OF THE ABOVE BRANCHES FAILED
- **HALT** ; OR WRONG SEQUENCE

**YESCC:**
- **CMP (R5),#5** ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
- **BNE CC4**
- **INC (R5)** ; NZ'C-1111
- **SCC**
- **BPL CC6**
- **BNE CC6**
- **BVC CC6**
- **BCC CC6**
- **BPL (C)**
- **BNE CC6**
- **BVC CC6**
- **BCC CC6**
- **BLS NOT(C)**

**C6:**
- **MOV #5,-(R5)**
- **INC -(R5)** ; (1) OR A BRANCH FAILED, OR WRONG SEQUENCE
- **HALT**

**TEST: 5**  CHECK BRANCH INSTRUCTIONS WITH ALL THE CONDITION CODES SET
TEST: 6  CLEAR THE CONDITION CODES

NOTICE:

**CMP**  \( (R5), #6 \)
**BNE**  \( CC5 \)
**INC**  \( (R5) \)
\( ; \text{IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST} \)
**INC**  \( (R5) \)
**SUB**  \( CC5 \)
\( ; \text{NZVC=1111} \)
**BCS**  \( CC5 \)
\( ; \text{NZVC=1110} \)
**CLI**  \( CC5 \)
\( ; \text{NZVC=1100} \)
**VSA**  \( CC5 \)
\( ; \text{NZVC=1000} \)
**LZ**  \( CC5 \)
\( ; \text{NZVC=0000} \)
**BEQ**  \( CC5 \)
**BMI**  \( CC5 \)
**BR**  \( ENDC5 \)

**CC5:**

**MOV**  \( #7,-(R5) \)
**INC**  \( -(R5) \)
\( ; \text{ONE OF THE ABOVE CLEARS FAILED OR WRONG SEQUENCE} \)

ENDCC5: **BPL**  BRANCH

TEST: 7  CHECK FORWARD AND BACKWARD BRANCHES.

BRANCH:

**CMP**  \( (R5), #7 \)
**BEQ**  \( 1\% \)
**MOV**  \( #10,-(R5) \)
\( ; \text{IF IN WRONG SEQUENCE GO TO HLT} \)
**INC**  \( -(R5) \)
**HALT**

\( \star \%: \)

**INC**  \( (R5) \)
**BR**  \( 4\% \)
**MOV**  \( #11,-(R5) \)
\( ; \text{CHECK BRANCH FORWARD AND BACKWARD} \)
**INC**  \( -(R5) \)
**HALT**

\( 2\%: \)

**BR**  \( 3\% \)
**MOV**  \( #12,-(R5) \)
\( ; \text{FORWARD BRANCH FAILED} \)
**INC**  \( -(R5) \)
**HALT**

\( 3\%: \)

**BR**  \( 5\% \)
**MOV**  \( #13,-(R5) \)
\( ; \text{FORWARD BRANCH FAILED} \)
**INC**  \( -(R5) \)
**HALT**

\( 4\%: \)

**BR**  \( 2\% \)
**MOV**  \( #14,-(R5) \)
\( ; \text{FORWARD BRANCH FAILED} \)
**INC**  \( -(R5) \)
**HALT**

\( 5\%: \)

**BR**  \( IMF \)
\( ; \text{BACKWARD BRANCH FAILED} \)
**TEST: 10 CHECK JMP INSTRUCTIONS FOR MODE 1**

---

**JMP1**:

<table>
<thead>
<tr>
<th>Opcode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP</td>
<td>(R5), #10</td>
</tr>
<tr>
<td>BNE</td>
<td>ENDJP1</td>
</tr>
<tr>
<td>JMP</td>
<td>(R0)</td>
</tr>
<tr>
<td>INC</td>
<td>-(R5)</td>
</tr>
<tr>
<td>MOV</td>
<td>#28, RO</td>
</tr>
<tr>
<td>; IF IN WRONG SEQUENCE GO TO HALT AT THE END OF THE TEST</td>
<td></td>
</tr>
</tbody>
</table>

---

**25**:  

- **INC** (R5)  
- **MOV** #28, RO  
- **TEST JUMP INSTRUCTION MODE 1**  
- **JUMP INSTRUCTION FAILED**

---

**35**:  

- **MOV** #16, -(R5)  
- **INC** -(R5)  
- **HALT**  
- **WRONG CC**

---

**55**:  

- **MOV** #17, -(R5)  
- **INC** -(R5)  
- **HALT**  
- **CONTINUE IF RO IS OK**

---

**ENDJP1**:  

- **MOV** #20, -(R5)  
- **INC** -(R5)  
- **HALT**  
- **JUMP INSTRUCTION FAILED OR WRONG SEQUENCE**

---

**TEST: 11 CHECK JMP INSTRUCTIONS FOR MODES 2 AND 3**

---

**JMP2**:

<table>
<thead>
<tr>
<th>Opcode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP</td>
<td>(R5), #11</td>
</tr>
<tr>
<td>BNE</td>
<td>ENDJP3</td>
</tr>
<tr>
<td>JMP</td>
<td>(R0)</td>
</tr>
<tr>
<td>INC</td>
<td>-(R5)</td>
</tr>
<tr>
<td>MOV</td>
<td>#38, RO</td>
</tr>
<tr>
<td>; IF IN WRONG SEQUENCE GO TO HALT AT THE END OF TEST</td>
<td></td>
</tr>
</tbody>
</table>

---

**35**:  

- **INC** (R5)  
- **MOV** #38, RO  
- **TEST JUMP INSTRUCTION MODE 2**  
- **JUMP INSTRUCTION FAILED**

---

**45**:  

- **INC** -(R5)  
- **HALT**  
- **JUMP INSTRUCTION FAILED**
CHECK JMP INSTRUCTIONS FOR MODES 2 AND 3

702 001346 012745 000022 MOV #22,-(R5)  : WRONG CC
703 001352 005245 INC -(R5)  : IS THERE AUTO INC.?
704 001354 000000 HALT
705 001356 002027 001340 5$: CMP R0,#3S+2
706 001362 001404 BEQ 6S
707 001364 012745 MOV #23,-(R5)
708 001370 005245 INC -(R5)
709 001372 000000 HALT
710 001376 012700 001412 6$: MOV #JMP3,R0
711 001400 000012 JMP (RO)+
712 001402 012745 MOV #24,-(R5)
713 001406 005245 INC -(R5)
714 001410 000000 HALT  : JUMP INSTRUCTION FAILED
715 001412 012767 001446 176516 JMP3: MOV #3S,TEMP  : TEST JUMP INSTRUCTION MODE 3
716 001420 012767 004666 17652 MOV #4S,TEMP+2
717 001426 012700 000036 MOV #TEMP,R0
718 001432 000277 SCC
719 001436 012745 MOV A(RO)+
720 001438 000130 INC -(R5)  : JUMP INSTRUCTION FAILED
721 001436 012745 MOV #25,-(R5)
722 001442 005245 INC -(R5)
723 001444 000000 HALT
724 001446 027067 000000 000012 3S: CMP A(RO),4S  : IS THERE AUTO INC.?
725 001446 01404 BEQ 4S
726 001454 012745 MOV #26,-(R5)
727 001462 005245 INC -(R5)
728 001464 000000 HALT
729 001466 012767 001512 176442 4$: MOV #JMP4,TEMP  : JMP INSTRUCTION FAILED IN MODE 2
730 001474 012700 000036 MOV #TEMP,R0
731 001500 000010 JMP A(RO)+
732 001502 012745 000000 ENDJP3: MOV #27,-(R5)  : TEST JUMP INSTRUCTION MODE 3
733 001506 005245 INC -(R5)
734 001510 000000 HALT  : JUMP ERROR OR WRONG SEQUENCE
; TEST: 12 TEST JUMP INSTRUCTION FOR MODE 4, 5

;------------------
JMP4:

CMP       (R5),#12
BNZ       ENDP5
INC       (R5)
MOV       #3$,(R5)
; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

HALT

; JUMP INSTRUCTION FAILED

HALT

BR        4$
; JUMP SHOULD LAND HERE

MOV       #31,-(R5)
INC       -(R5)
; NO AUTO DECREMENT FROM JMP4

HALT

; JUMP INSTRUCTION FAILED

MOV       #35,-(R5)
INC       -(R5)
; NO AUTO DECREMENT FROM JMP5

HALT

; JUMP INSTRUCTION FAILED

MOV       #36,-(R5)
INC       -(R5)
; NO AUTO DECREMENT FROM JMP6

HALT

; JUMP INSTRUCTION FAILED

MOV       #37,-(R5)
INC       -(R5)
; JUMP ERROR OR WRONG SEQUENCE
*TEST: 13 TEST JMP INSTRUCTION FOR MODE 6 AND 7

JMP6:

CMP (RS), #13
BNE ENDJP7
JP (RS)
MOV #1S+6, R3
JMP -6 (RS)
MOV #40, -(RS)
INC -(RS)
HALT

1S:
CMP R3, #1S+6
BEQ 2S
MOV #41, -(RS)
INC -(RS)
HALT

2S:
JMP 3S, -(PC)
MOV #42, -(RS)
INC -(RS)
HALT

3S:
MOV #JMP7, R3
JMP 8(R3)
MOV #43, -(RS)
INC -(RS)
HALT

JMP7:
MOV #TEMP, R3
MOV #15, (R3)
JMP @R3
MOV #44, -(RS)
INC -(RS)
HALT

1S:
MOV #3S, (R3)
MOV #TEMP=4, RO
JMP 96(R0)
MOV #45, -(RS)
INC -(RS)
HALT

3S:
MOV #JSTRST, TEMP
CONTINUE
JMP @R0

ENDJP7:
MOV #46, -(RS)
INC -(RS)
HALT

JUMP ERROR OR SEQUENCE ERROR
**TEST 14: CHECK JSR AND MARK INSTRUCTIONS**

**JSRTST:**

- **CMP** $(R5), #14
- **JSR ENDJSR** $(R5)
- **MOV** #START, SP
- **SETUP STACK POINTER.

**JSR ST:**

- **JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
- **SP WAS NOT RESTORED BY RTS INSTRUCTION

**JMP** $(R5)

- **INC** $(R5)
- **HALT**

**JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED

**JMP** $(R5)

- **INC** $(R5)
- **HALT**

**JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED

**JMP** $(R5)

- **INC** $(R5)
- **HALT**

**JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED

**JMP** $(R5)

- **INC** $(R5)
- **HALT**

**JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED

**JMP** $(R5)

- **INC** $(R5)
- **HALT**

**JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
- **JSR INSTRUCTION FAILED
END JSR:

MOV #67 - (R5)
INC -(R5)
HALT

; RTS INSTRUCTION FAILED OR SEQUENCE ERROR

; TEST: 15 CHECK REGISTER SELECTION

REGS:

CMP (R5), #15
BNE EREGS
INC (R5)
MOV R6, TEMP
MOV #1, R0
MOV #4, R1
MOV #20, R2
MOV #100, R3
MOV #400, R4
CLR R6
ADD R0, R6
ADD R1, R6
ADD R2, R6
ADD R3, R6
ADD R4, R6
ADD R5, R6
CMP #$TESTN+525, R6
BNE 1S
MOV TEMP, R6
BR TS180
TS180
MOV TEMP, R6
RSTORE STACK POINTER

1S: MOV #70, -(R5)
INC -(R5)
HALT

; REGISTER SELECTION FAILURE OR SEQUENCE ERROR
CHECK BYTE INSTRUCTIONS, DESTINATION MODE 0 ONLY

::: TEST: 16  NEW INSTRUCTIONS USED IN THIS SECTION ARE TSTB, CLRB, MOVB :::

TSTB:

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Address</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>CMP (R5), #16</td>
<td>00016</td>
<td></td>
</tr>
<tr>
<td>0002</td>
<td>BEQ 2$</td>
<td>00016</td>
<td>IF IN WRONG SEQUENCE GO TO HLT BELOW</td>
</tr>
<tr>
<td>0003</td>
<td>MOV #71, -(R5)</td>
<td>00016</td>
<td></td>
</tr>
<tr>
<td>0004</td>
<td>INC -(R5)</td>
<td>00016</td>
<td></td>
</tr>
<tr>
<td>0005</td>
<td>JNE (R5)</td>
<td>00016</td>
<td>PROGRAM IS IN WRONG SEQUENCE</td>
</tr>
<tr>
<td>0006</td>
<td>CLRB RO</td>
<td>00016</td>
<td>CLEAR THE REGISTER</td>
</tr>
<tr>
<td>0007</td>
<td>JSR PC + #CC4</td>
<td>00016</td>
<td>CHECK FOR CC = 4</td>
</tr>
<tr>
<td>0008</td>
<td>JSR PC + #CC4</td>
<td>00016</td>
<td>CHECK FOR CC = 4</td>
</tr>
<tr>
<td>0009</td>
<td>MOVB #377, R1</td>
<td>00016</td>
<td>LOAD THE REGISTER</td>
</tr>
<tr>
<td>0010</td>
<td>JSR PC + #CC10</td>
<td>00016</td>
<td>CHECK FOR CC = 10</td>
</tr>
<tr>
<td>0011</td>
<td>TSTB R1</td>
<td>00016</td>
<td>CHECK FOR CC = 10</td>
</tr>
<tr>
<td>0012</td>
<td>JSR PC + #CC10</td>
<td>00016</td>
<td>CHECK FOR CC = 10</td>
</tr>
</tbody>
</table>

::: TEST: 17  NEW INSTRUCTIONS USED IN THIS SECTION ARE CMPB, BISB :::

CMPB:

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Address</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0013</td>
<td>CMP (R5), #17</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>BNE ECMMPB</td>
<td>00017</td>
<td>IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST</td>
</tr>
<tr>
<td>0015</td>
<td>INC (R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SCC</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0017</td>
<td>BISB #377, R2</td>
<td>00017</td>
<td>LOAD REGISTER</td>
</tr>
<tr>
<td>0018</td>
<td>JSR PC + #CC11</td>
<td>00017</td>
<td>CHECK FOR CC = 11</td>
</tr>
<tr>
<td>0019</td>
<td>CMPB #377, R2</td>
<td>00017</td>
<td>CHECK COMPARE</td>
</tr>
<tr>
<td>0020</td>
<td>BEQ 2$</td>
<td>00017</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>0021</td>
<td>MOV #72, -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0022</td>
<td>INC -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0023</td>
<td>HALT</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0024</td>
<td>MOVB #77, R0</td>
<td>00017</td>
<td>BISB OR CMPB INSTRUCTION FAILED</td>
</tr>
<tr>
<td>0025</td>
<td>CMPB RO, R2</td>
<td>00017</td>
<td>CHECK IT AGAIN</td>
</tr>
<tr>
<td>0026</td>
<td>BPL 3$</td>
<td>00017</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>0027</td>
<td>MOV #73, -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0028</td>
<td>INC -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0029</td>
<td>HALT</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0030</td>
<td>CMPB #R2, R0</td>
<td>00017</td>
<td>CMPB INSTRUCTION FAILED (WRONG CC)</td>
</tr>
<tr>
<td>0031</td>
<td>BMI 4$</td>
<td>00017</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>0032</td>
<td>MOV #74, -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0033</td>
<td>INC -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0034</td>
<td>HALT</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0035</td>
<td>CMPB R2, R0</td>
<td>00017</td>
<td>CMPB INSTRUCTION FAILED (WRONG CC)</td>
</tr>
<tr>
<td>0036</td>
<td>BMI 4$</td>
<td>00017</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>0037</td>
<td>MOV #74, -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0038</td>
<td>INC -(R5)</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0039</td>
<td>HALT</td>
<td>00017</td>
<td></td>
</tr>
<tr>
<td>0040</td>
<td>CMPB #77, R2</td>
<td>00017</td>
<td>CMPB INSTRUCTION FAILED (WRONG CC)</td>
</tr>
<tr>
<td>0041</td>
<td>MOVB #77, R2</td>
<td>00017</td>
<td>LOAD REGISTER, SIGN EXTEND</td>
</tr>
<tr>
<td>0042</td>
<td>CMPB #77, R2</td>
<td>00017</td>
<td>CMPB INSTRUCTION FAILED (WRONG CC)</td>
</tr>
</tbody>
</table>
NEW INSTRUCTIONS USED IN THIS SECTION ARE CMPB, BISB

BEQ $5; ; CONTINUE IF OK
MOV #75,-(R5); ; CMPB BECAME CMP INSTRUCTION
INC -(R5) ; ; LOAD REGISTER, SIGN EXTEND
HALT
cmp $; ; CHECK IF BYTE INSTRUCTION
movb #377,$2 ; ; B) CB0
beq $; ; CONTINUE IF OK
ecmpbo;
mov #76,-(R5) ; ; WRONG CC OR WRONG SEQUENCE
inc -(R5)
**TEST: 20**  
NEW INSTRUCTIONS USED IN THIS SECTION ARE BICB, BITB

**BICBO:**

```assembly
CMP  (R5), #20
BEQ  2S
MOV  #77,-(R5)
INC  (R5)
HALT
2S:
INC  (R5)
MOV  #377,R3
MOV  #252,R0
PLACE #252 IN R0
JSR  BICB R0, R3
BITB PC, @#$CC1
BITB R0, R3
BEQ  4S
MOV  #100,-(R5)
INC  (R5)
HALT
4S:
BITB #125,R3
JSR  PC, @#$CC1
BITB R0, R3
BISB PC, @#$CC1
BITB #377,R3
MOV  #101,-(R5)
INC  (R5)
HALT
6S:
BISB #177,R3
JSR  PC, @#$CC1
BITB #377,R3
MOV  #377,R4
INC  R4
SEC
INCB R4
JSR  PC, @#$CC1
BITB #377,R3
MOV  #377,R4
INC  R4
SEC
INCB R4
JMP -177
```

**TEST: 21**  
NEW INSTRUCTIONS USED IN THIS SECTION ARE INCB, DECB

**INCB:**

```assembly
CMP  (R5), #21
BEQ  1S
MOV  #102,-(R5)
INC  (R5)
HALT
1S:
INC  (R5)
MOV  #177,R4
SEC
INCB R4
JSR  PC, @#$CC1
MOV  #377,R4
INCB R4
JSR  PC, @#$CC1
INCB R4
JMP -177
```
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1109</td>
<td>003256</td>
<td>JSR PC, @$CC1</td>
<td>CHECK FOR CC = 1</td>
</tr>
<tr>
<td>1110</td>
<td>003262</td>
<td>CMPB #1, R4</td>
<td>CHECK IT</td>
</tr>
<tr>
<td>1111</td>
<td>003266</td>
<td>SEQ 2$</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>1112</td>
<td>003270</td>
<td>MOV #103, -(R5)</td>
<td></td>
</tr>
<tr>
<td>1113</td>
<td>003274</td>
<td>INC -(R5)</td>
<td>INCB INSTRUCTION FAILED</td>
</tr>
<tr>
<td>1114</td>
<td>003276</td>
<td>HLT</td>
<td></td>
</tr>
<tr>
<td>1115</td>
<td>003300</td>
<td>2$</td>
<td>DECB R4</td>
</tr>
<tr>
<td>1116</td>
<td>003302</td>
<td>SEC</td>
<td>SUBTRACT ONES FROM REG. 4</td>
</tr>
<tr>
<td>1117</td>
<td>003304</td>
<td>JSR PC, @$CC5</td>
<td>CHECK FOR CC = 5</td>
</tr>
<tr>
<td>1118</td>
<td>003310</td>
<td>DECB R4</td>
<td></td>
</tr>
<tr>
<td>1119</td>
<td>003312</td>
<td>JSR PC, @$CC11</td>
<td>CHECK FOR CC = 11</td>
</tr>
<tr>
<td>1120</td>
<td>003316</td>
<td>MOV #200, R4</td>
<td></td>
</tr>
<tr>
<td>1121</td>
<td>003322</td>
<td>DECB R4</td>
<td></td>
</tr>
<tr>
<td>1122</td>
<td>003324</td>
<td>JSR PC, @$CC3</td>
<td>CHECK FOR CC = 3</td>
</tr>
<tr>
<td>1123</td>
<td>003330</td>
<td>DECB R4</td>
<td></td>
</tr>
<tr>
<td>1124</td>
<td>003332</td>
<td>JSR PC, @$CC1</td>
<td>CHECK FOR CC = 1</td>
</tr>
</tbody>
</table>
NEW INSTRUCTION IN THIS SECTION IS NEGB

**NEGB0:**  
MOV $110, -(R5)  
INC -(R5)  
HALT  

; WRONG RESULT IN RO OR WRONG SEQUENCE
**TEST: 26**  NEW INSTRUCTION IN THIS SECTION IS ASLB

```
003726        021527        000026
003732        001164        000115
003740        005245        000000
003742        000000        000000
003744        005215        000000
003746        112703        000040
003752        000164        000000
003756        106303        000000
003760        004737        017300
003764        122703        000200
003770        001164        000116
003772        012745        000000
003776        005245        000000
004000        000000        000000
004002        106303        000000
004004        004737        017214
004010        106303        000000
004012        004737        017150
```

**ASLB0:**
```
CMP       (R5), #26   ; IF IN WRONG SEQUENCE GO TO HLT BELOW
BEQ       2%      
MOV       @115, -(R5)   ; PROGRAM IS IN WRONG SEQUENCE
INC       -(R5)
```
```
MOVB      #40, R3   ; LOAD REGISTER
CCC
ASLB      R3   ; CLEAR FLAGS
R3
SHFT
ASLB      R3
JSR      PC, #WCC12   ; CHECK FOR CC = 12
CMPP      #200, R3   ; CHECK R3
BEQ       4%      ; CONTINUE IF OK
MOV       #16, -(R5)
INC       -(R5)
HALT
```
```
4%:       ASLB      R3   ; ASLB INSTRUCTION FAILED
JSR      PC, #WCC7   ; CHECK FOR CC = 7
ASLB      R3
JSR      PC, #WCC4   ; CHECK FOR CC = 4
```

**TEST: 27**  NEW INSTRUCTION IN THIS SECTION IS ASRB

```
004016        021527        000027
004022        001034        000115
004024        005215        000004
004026        112704        000000
004032        002527        000000
004034        106204        000000
004036        106204        000000
004040        122704        000000
004044        001404        000117
004046        012745        000000
004052        005245        000000
004054        000000        000000
004056        106204        000000
004060        004737        017214
004064        106204        000000
004066        004737        017150
004072        112703        000202
004076        106203        000000
004080        106203        000000
004084        106203        000000
004088        106203        000000
004092        004737        017256
004100        122703        000340
004104        001104        000117
004112        001104        000117
004114        001104        000117
004120        001104        000117
004124        021527        000027
```

**ASRB0:**
```
CMP       (R5), #27   ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
BNE       EASRB0
```
```
1%:       INC       (R5)
MOVB      #4, R4   ; LOAD REGISTER
CCC
ASRB      R4   ; CLEAR FLAGS
R4
SHFT
ASRB      R4
CMPP      #1, R4   ; CHECK IT
BEQ       2%      ; CONTINUE IF OK
MOV       #17, -(R5)
INC       -(R5)
HALT
```
```
2%:       ASRB      R4   ; ASRB INSTRUCTION FAILED
JSR      PC, #WCC7   ; CHECK FOR CC = 7
ASRB      R3
JSR      PC, #WCC4   ; CHECK FOR CC = 4
CMPP      #202, R3   ; LOAD REGISTER
BEQ       4%      ; CONTINUE IF OK
MOV       #16, -(R5)
INC       -(R5)
```

**EASRB:**
```
MOV       #39, -(R5)
INIT
-(R5)
```
NEW INSTRUCTION IN THIS SECTION IS ASRB

HALT
**G 3**

**NEW INSTRUCTION IN THIS SECTION IS ADCB**

---

**ADCBO:**

```
CMP    (R5),#30          ; IF IN WRONG SEQUENCE GO TO HLT BELOW
BEQ    2$                ; PROGRAM IS IN WRONG SEQUENCE
MOV    #127.1,-(R5)
INC    -(R5)
HALT
```

---

**SBCBO:**

```
CMP    (R5),#31          ; IF IN WRONG SEQUENCE GO TO HLT BELOW
BEQ    1$                ; PROGRAM IS IN WRONG SEQUENCE
MOV    #127.6,-(R5)
INC    -(R5)
HALT
```

---

**TEST: 30** NEW INSTRUCTION IN THIS SECTION IS ADCB

---

**TEST: 31** NEW INSTRUCTION IN THIS SECTION IS SBCB
NEW INSTRUCTION IN THIS SECTION IS SBCB

```
554 000331 122701 000003
555 000332 001404
556 000320 012753 000125
557 000324 003255
558 000326 000000
559 000330 000265
560 000332 105601
561 000334 000261
562 000336 105601
563 000330 000207 217046
564 000334 000001
565 000330 001404
566 000332 012745 000126
567 000336 000265
568 000330 000261
569 000332 105601
570 000334 000261
571 000336 000237 017150
572 000332 000261
573 000334 105601
574 000336 000237 017256
575 000340 122701 000377
576 000340 001404
577 000340 012745 000127
578 000340 000265
579 000340 000000
580 000340 112701 000200
581 000340 000261
582 000340 105601
583 000340 004737 C
```

CMPB #3,PI : CHECK IT
BEQ 2$ : CONTINUE IF OK
MOV #125,-(R5)
INC -(R5)
SBCB R1 : SBCB INSTRUCTION FAILED
SEC
SBCB R1
JSR PC, #PC+CC0
CMPB #1, R1
BEQ 3$: SUBTRACT C BIT = 0
INC -(R5)
SBCB R1
JSR PC, #PC+CC4
CHECK FOR CC = 0
CHECK IT
CONTINUE IF OK
SBCB R1
JSR PC, #PC+CC1
CHECK FOR CC = 1
CHECK IT
CONTINUE IF OK
MOV #127,-(R5)
INC -(R5)
SBCB R1
JSR PC, #PC+CC2
CHECK FOR CC = 2
LOAD RT
C=1
SBCB R1
SUBTRACT C BIT = 1
SEC
SBCB R1
SUBTRACT C BIT = 1
JSR PC, #PC+CC0
CHECK IT
CHECK WORD INSTRUCTIONS, DESTINATION MODE 0 ONLY

; TEST: 32 NEW INSTRUCTIONS USED IN THIS SECTION ARE TST, CLR, MOV

TSIO:

CMP (R5), #32 ; IF IN WRONG SEQUENCE GO TO HLT
BNE E3
MOV #130, -(R5)
INC -(R5)
HALT ; TEST IS IN WRONG SEQUENCE

*5:
INC (R5)

CLC
CLR RO ; CLEAR THE REGISTER
TST RO ; CHECK FOR CC = 4
JSR PC, #CC4 ; CHECK IT
MOV #17777, R4 ; LOAD THE REGISTER
MOV R4, R1 ; CHECK FOR CC = 4
TST R1 ; LOAD THE REGISTER
JSR PC, #CC10 ; CHECK FOR CC = 10
TST R1 ; CHECK FOR CC = 10
JSR PC, #CC10 ; CHECK FOR CC = 10
CMP R4, R1 ; CHECK FOR CC = 10
BEQ 2S ; CONTAIN PROPER DATA
JSR PC, #CC10
MOV #131, -(R5)
INC -(R5)
HALT ; SET V & C BITS

25:
SEVC
MOV RO, RO
JSR PC, #CC5

; TEST: 33 NEW INSTRUCTIONS USED IN THIS SECTION ARE CMP, BIS

CMPO:

CMP (R5), #33 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
BNE E5
MOV #17777, R0
BIS R0, R2 ; CHECK THE BIS INSTRUCTION
JSR PC, #CC10 ; CHECK FOR CC = 10
CMP R0, R2 ; CHECK COMPARE
BFG 2S ; CONTINUE IF OK
MOV #132, -(R5)
INC -(R5)
HALT ; BIS OR CMP INSTRUCTION FAILED
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>440</td>
<td>CMP 77.R2</td>
<td>CHECK IT AGAIN</td>
</tr>
<tr>
<td>441</td>
<td>BPL 58</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>442</td>
<td>MOV 133,-(R5)</td>
<td>CMP INSTRUCTION FAILED [WRONG CC]</td>
</tr>
<tr>
<td>443</td>
<td>INC -(R5)</td>
<td>ONCE MORE</td>
</tr>
<tr>
<td>444</td>
<td>HALT</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>445</td>
<td>CMP R2,#77</td>
<td>WRONG CC OR WRONG SEQUENCE</td>
</tr>
<tr>
<td>446</td>
<td>BMI B100</td>
<td></td>
</tr>
<tr>
<td>447</td>
<td>CMPO:</td>
<td></td>
</tr>
<tr>
<td>448</td>
<td>MOV 134,-(R5)</td>
<td></td>
</tr>
<tr>
<td>449</td>
<td>INC -(R5)</td>
<td></td>
</tr>
<tr>
<td>450</td>
<td>HALT</td>
<td></td>
</tr>
</tbody>
</table>
BICU:  
CMP *(R5),#34  ; IF IN WRONG SEQUENCE GO TO HLT ABOVE
BNE EBICO  
INC (R5)  
MOV #17777,(R3)  ; LOAD REGISTER
MOV #TEMP,R0  ; PLACE THE ADDRESS OF LOCATION TEMP IN R0
MOV #125252,(R0)  ; SET (R0)
SCE  
BIC *(R3)  ; CLEAR EVERY OTHER BIT
JSR PC,#CC1  ; CHECK FOR CC = 1
BIT *(R0),R3  ; CHECK IT
BEG 1S  ; CONTINUE IF OK
MOV #135,-(R5)  
INC -(R5)  ; BIT OR BIT INSTRUCTION FAILED
HALT 1S:  
BIT #52525,R3  ; CHECK IT
JSR PC,#CC1  ; CHECK FOR CC = 1
BIS #125252,R3  ; SET THE BITS THAT WERE CLEARED
BMI 2S  ; CONTINUE IF OK
MOV #176,-(R5)  
INC -(R5)  ; BIT OR BIT INSTRUCTION FAILED
HALT 2S:  
BIT #77777,R3  ; CLEAR ALL THE BITS EXCEPT FOR SIGN
JSR PC,#CC1  ; CHECK FOR CC = 11
MOV #177777,R0  ; CHECK IT
BIT R0,R3  ; CHECK FOR CC = 11
JSR PC,#CC11  ; SET V & C BITS
BMI R0,R0  
SEVC  
BIC R0,R0  
JSR PC,#CC5  ; CHECK CC = 5
BMI R0,R0  
TST R0  ; CHECK R0 TO CONTAIN 0
BEG INCO  
EBICO:  
MOV #137,-(R5)  ; BIT FAILED OR SEQUENCE ERROR
INC -(R5)
HALT  ; BIT FAILED OR SEQUENCE ERROR
NEW INSTRUCTIONS USED IN THIS SECTION ARE BIC, BIT
NEW INSTRUCTIONS USED IN THIS SECTION ARE BIC, BIT

*TEST: 35  NEW INSTRUCTIONS USED IN THIS SECTION ARE INC, DEC

<table>
<thead>
<tr>
<th>INCO:</th>
<th>(RS),#35</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEQ</td>
<td>28:</td>
</tr>
<tr>
<td>MOV</td>
<td>#160-(RS)</td>
</tr>
<tr>
<td>INC</td>
<td>(RS)</td>
</tr>
<tr>
<td>HALT</td>
<td></td>
</tr>
<tr>
<td>28:</td>
<td>INC</td>
</tr>
<tr>
<td>MOV</td>
<td>#77777,R4</td>
</tr>
</tbody>
</table>

: IF IN WRONG SEQUENCE GO TO HALT BELOW

: PROGRAM IS IN WRONG SEQUENCE

: R4=77777
ADD ONES INTO REG. 4
CHECK FOR CC = 13
CHECK FOR CC = 11
CHECK FOR CC = 5
CHECK FOR CC = 1
CHECK IT
FAILED
INC INSTRUCTION FAILED
SUBTRACT ONES FROM REG. 4
CHECK FOR CC = 5
CHECK FOR CC = 11
CHECK FOR CC = 3
CHECK FOR CC = 1
**TEST: 36**

NEW INSTRUCTION IN THIS SECTION IS COM

```
COM:  
  CMP (R5),#36 ; IF IN WRONG SEQUENCE GO TO HLT BELOW
  BEQ 1S
  MOV #142,-(R5) ; TEST IS IN WRONG SEQUENCE
  INC -(R5)
  HALT
  INC (R5)
  MOV #125252,R3 ; LOAD EVERY OTHER BIT
  SCC
  COM R3 ; 1'S COMPLEMENT
  SW PC,#CC13
  SHR $10,PC
  CHECK CC = 1
  MOV -2,-(R5)
  INC -(R5)
  HALT
  MOV #125252,R3 ; CHECK FOR CC = 1
  INC -(R5)
  HALT
```

**TEST: 37**

NEW INSTRUCTION IN THIS SECTION IS NEG

```
NEGO:  
  CMP (R5),#37 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
  BNE ENEGO
  INC (R5)
  MOV #1,R0 ; LOAD THE REGISTER
  NEG R0 ; 2'S COMPLEMENT
  MOV PC,#CC11 ; CHECK FOR CC = 1
  JSR PC
  CMP #177777,R0 ; CHECK IT
  BEQ 2S
  MOV #1-5,-(R5) ; CONTINUE IF OK
  INC -(R5)
  HALT
  MOV #100000,R0 ; NEG INSTRUCTION FAILED
  NEG R0 ; 2'S COMPLEMENT
  MOV PC,#CC13 ; CHECK FOR CC = 12
  JSR PC
  CMP #100000,R0
  BEQ 2S
  INC -(R5)
  SEG KOCC
  CONTINUE IF OK
```

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction 1</th>
<th>Address</th>
<th>Instruction 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1587</td>
<td>005326</td>
<td>1588</td>
<td>005326</td>
</tr>
<tr>
<td></td>
<td>012745 000146</td>
<td>1589</td>
<td>005332 065245</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1590</td>
<td>005332 000000</td>
</tr>
</tbody>
</table>

ENEGO:

MOV    #146,-(RS)
INC    -(RS)
HALT

: WRONG RESULT IN RO OR WRONG SEQUENCE
NEW INSTRUCTION IN THIS SECTION IS ROR

HALT : WRONG RESULT IN R2 OR WRONG SEQUENCE
NEW INSTRUCTION IN THIS SECTION IS ASR

HALT: WRONG RESULT IN R3 OR WRONG SEQUENCE
H 4

1705 NEW INSTRUCTION IN THIS SECTION IS ADC

1706 TEST: 44

1707 NEW INSTRUCTION IN THIS SECTION IS ADC

1709 ADC0:

1710 CMP (R5),#44

1711 BEQ 2%

1712 MOV #157, -(R5)

1713 INC -(R5)

1714 HALT

1715 ; IF IN WRONG SEQUENCE GO TO HLT BELOW

1716 2%

1717 INC (R5)

1718 CLR RO

1719 ADC RC

1720 JSR PC, @$CC4

1721 SEC

1722 ADDC 40

1723 ADDC C=1

1724 AGAIN

1725 JSR PC, @$CC0

1726 MOV #160, -(R5)

1727 INC -(R5)

1728 4%

1729 HALT

1730 MOV #77777, RO

1731 SEC

1732 ADC RO

1733 JSR PC, @$CC4

1734 CMP #100000, RO

1735 INC -(R5)

1736 6%

1737 MOV #161, -(R5)

1738 INC -(R5)

1739 HALT

1740 MOV #-1, RO

1741 SEC

1742 ADC RO

1743 JSR PC, @$CC5

1744 ; CHECK FOR CC = 5

1745 ;

1746 TEST: 45

1747 NEW INSTRUCTION IN THIS SECTION IS SBC

1748

1750 SBC0:

1751 CMP (R5), #45

1752 BEQ 1%

1753 MOV #162, -(R5)

1754 INC -(R5)

1755 HALT

1756 1%

1757 INC (R5)

1758 MOV #3, R1

1759 SEC

1760 SBC R

1761 JSR PC, @$CC5

1762 ; CHECK FOR CC = 0
**Test: 46**  NEW INSTRUCTION IN THIS SECTION IS SXT

SXT0:

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>006230</td>
<td>CMP   (R5), #46</td>
<td>If in wrong sequence go to HLT at the end of the test</td>
</tr>
<tr>
<td>006231</td>
<td>BNE   ESXTO</td>
<td></td>
</tr>
<tr>
<td>006232</td>
<td>INC   (R5)</td>
<td></td>
</tr>
<tr>
<td>006233</td>
<td>CLR   R2</td>
<td>Clear register</td>
</tr>
<tr>
<td>006234</td>
<td>SCC   R2</td>
<td></td>
</tr>
<tr>
<td>006235</td>
<td>CLNZ  R2</td>
<td></td>
</tr>
<tr>
<td>006236</td>
<td>SXT   R2</td>
<td></td>
</tr>
<tr>
<td>006237</td>
<td>JSR   PC, #&amp;$CC5</td>
<td>Check for CC = 5</td>
</tr>
<tr>
<td>006238</td>
<td>TEST  R2</td>
<td>Reg. 2 should still be 0</td>
</tr>
<tr>
<td>006239</td>
<td>BEQ   2$</td>
<td>Continue if OK</td>
</tr>
<tr>
<td>00623A</td>
<td>MOV   #166, -(R5)</td>
<td></td>
</tr>
<tr>
<td>00623B</td>
<td>INC   -(R5)</td>
<td></td>
</tr>
<tr>
<td>00623C</td>
<td>HALT</td>
<td>SXT instruction failed</td>
</tr>
<tr>
<td>00623D</td>
<td>SENC  R2</td>
<td>Set N, V &amp; C bits</td>
</tr>
<tr>
<td>00623E</td>
<td>SXT   R2</td>
<td></td>
</tr>
<tr>
<td>00623F</td>
<td>JSR   PC, #&amp;$CC11</td>
<td>Check for CC = 11</td>
</tr>
<tr>
<td>006240</td>
<td>CMP   #-1, R2</td>
<td>Reg. 2 should now have -1</td>
</tr>
<tr>
<td>006241</td>
<td>BEO   SWABD</td>
<td>Continue if OK</td>
</tr>
<tr>
<td>006242</td>
<td>MOV   #167, -(R5)</td>
<td></td>
</tr>
<tr>
<td>006243</td>
<td>INC   -(R5)</td>
<td></td>
</tr>
<tr>
<td>006244</td>
<td>HALT</td>
<td>Wrong result in R2 or wrong sequence</td>
</tr>
</tbody>
</table>

**Test: 47**  NEW INSTRUCTION IN THIS SECTION IS SWAB

SWAB0:

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>006316</td>
<td>CMP   (R5), #47</td>
<td>If in wrong sequence go to HLT at the end of the test</td>
</tr>
<tr>
<td>006317</td>
<td>BNE   ESWABO</td>
<td></td>
</tr>
<tr>
<td>006318</td>
<td>INC   (R5)</td>
<td></td>
</tr>
<tr>
<td>006319</td>
<td>MOV   #125125, R3</td>
<td>Load bit pattern into register</td>
</tr>
<tr>
<td>00631A</td>
<td>SWAB  R3</td>
<td>Swap bytes of register</td>
</tr>
<tr>
<td>00631B</td>
<td>JSR   PC, #&amp;$CC10</td>
<td>Check for CC = 10</td>
</tr>
<tr>
<td>00631C</td>
<td>CMP   #52652, R3</td>
<td>Check it</td>
</tr>
<tr>
<td>00631D</td>
<td>BEQ   1$</td>
<td>Continue if OK</td>
</tr>
<tr>
<td>00631E</td>
<td>MOV   #170, -(R5)</td>
<td></td>
</tr>
<tr>
<td>00631F</td>
<td>INC   -(R5)</td>
<td></td>
</tr>
<tr>
<td>006320</td>
<td>HALT</td>
<td>SWAB instruction failed</td>
</tr>
<tr>
<td>006321</td>
<td>MOV   #377, R3</td>
<td></td>
</tr>
<tr>
<td>006322</td>
<td>BEO   0</td>
<td></td>
</tr>
<tr>
<td>006323</td>
<td>MOV   #172000, R3</td>
<td></td>
</tr>
<tr>
<td>006324</td>
<td>BEO   0</td>
<td></td>
</tr>
<tr>
<td>006325</td>
<td>SWAB  R3</td>
<td></td>
</tr>
<tr>
<td>006326</td>
<td>JSR   PC, #&amp;$CC4</td>
<td>Check for CC = 4</td>
</tr>
<tr>
<td>006327</td>
<td>CMP   #172000, R3</td>
<td></td>
</tr>
<tr>
<td>006328</td>
<td>BEO   0</td>
<td></td>
</tr>
<tr>
<td>006329</td>
<td>ESWABO</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>Instruction</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>847</td>
<td>MOV #171, -(RS)</td>
<td></td>
</tr>
<tr>
<td>1849</td>
<td>INC -(RS)</td>
<td></td>
</tr>
<tr>
<td>1849</td>
<td>HALT</td>
<td></td>
</tr>
</tbody>
</table>

; WRONG RESULT IN R3 OR WRONG SEQUENCE
**TEST: 50**

NEW INSTRUCTION IN THIS SECTION IS XOR

---

XOR:

```
CMP (R5), #50 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
BNE EXORO    ; LOAD REGISTERS
INC (R5)      ;
MOV #1, R4     ;
MOV #1, R3     ;
SRE R4, R3     ; SHOULD PRODUCE 0'S IN REG. 3
JSR PC, @PCCS ; CHECK FOR CC = 5
MOV #17777777,R3
MOV R4, R0     ; PLACE A -1 IN R0
SEVC          ; SET V & C BITS
CLZ R0         ;
XOR A0, R3     ; CHECK FOR CC = 11
JSR PC, @PCC11 ; LOAD REGISTERS
MOV #525252,R2
MOV #525252,R4
SSE R0, R2     ; SHOULD PRODUCE ALL 1'S IN REG. 4
JSR PC, @PCC11 ; CHECK FOR CC = 11
CMP #1, R4     ; CHECK IT
BEQ ADDO       ; CONTINUE IF OK
```

EXORO:

```
MOV #172,-(R5) ;
INC -R5        ;
HALT           ; WRONG RESULT IN R4 OR WRONG SEQUENCE
```

---

**TEST: 51**

NEW INSTRUCTION IN THIS SECTION IS ADD

---

ADD:

```
CMP (R5), #51 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
BNE EADDO     ; LOAD REGISTERS
INC (R5)      ;
MOV #21421,R1
ADD R1, R1     ;
SRE R1, R1     ; ADD
JSR PC, @PCC0  ; CHECK FOR CC = 0
CMP #43042,R1
BEQ 1S         ; CHECK IT
MOV #173,-(R5)
INC -R5        ; CONTINUE IF OK
HALT R0        ;
```

```
MOV #-21421, R0 ; ADD INSTRUCTION FAILED
ADD R0, R0
JSR PC, @PCC1  ; ADD
CMP #43042, R0
BEQ 2S
MOV #2,-(R5)
INC -R5
```

---
NEW INSTRUCTION IN THIS SECTION IS ADD

```
26: MOV #100000,R2
ADD R2,R2
JSR PC,#$CC7
MOV #21421,R4
MOV #-21421,R1
ADD R4,R1
BEQ 3$
MOV #175,-(R5)
INC -(R5)
3$: NEG R4
MOV #21421,R1
ADD R1,R4
BEQ SUB0
4ADD: MOV #176,-(R5)
INC -(R5)
HALT
```

; ADD INSTRUCTION FAILED
; LOAD REGISTERS
; ADD SHOULD RESULT AS 0'S
; CHECK FOR CC = 7
; LOAD REGISTERS
; ADD SHOULD RESULT AS 0'S
; CONTINUE IF OK
; ADD INSTRUCTION FAILED
; SWITCH SOURCE AND DESTINATION
; SHOULD RESULT AS 0'S
; CONTINUE IF OK
; WRONG RESULT IN R1 OR WRONG SEQUENCE
NEW INSTRUCTIONS IN THERE SECTION IS MIPS & MFPS

BEQ 2$  ; CONTINUE IF BIT 8 OF PSW WAS EXTENDED IN R1
MOV #203,-(R5)
INC -(R5)
JSR PC,#$CC4  ; MIPS OR MFPS INSTRUCTION FAIiLED
MOV #377,RO  ; CHECK FOR CC = 4
MIPS RO     
MIPS R0     ; SET PSW TO 357 SINCE MIPS DOES NOT SET T BIT
.MWORD 106400:...C
JSR PC,#$CC17  ; CHECK FOR CC = 17
MIPS R1     ; MOVE PSW TO R1
MIPS R1     
.MWORD 106700:...C
JSR PC,#$CC11  ; CHECK FOR CC = 11 CC BIT SHOULD NOT BE EFFECTED BY MFPS
CMP #177757,R1 ; CHECK TO SEE IF BIT 8 OF PSW WAS EXTENDED THRU R1
BEQ ModE0  
MIPS Mode0  
MO, #204,-(R5)
INC -(R5)    ; MIPS OR MFPS INSTRUCTION FAILED OR WRONG SEQUENCE
MIPS R1
LSI-11 INSTRUCTIONS NOT MODE 0

**MODE 0:**

```
CMP (R5), #54
BEQ EMODE
JNE (R5)
MLVB #252, R0
MOVB R0, R1
MOVB R1, R2
CMPB #252, R2
BEQ 1$
MOV #205, -(R5)
INC -(R5)
HALT
```

*IF IN WRONG SEQUENCE GO TO HALT AT THE END OF THE TEST

**MODE 1:**

```
MOV #125252, R0
LOAD REGISTERS
MOV R0, R1
MOV R1, R2
CMP #125252, R2
CHECK IT
BEQ MODE1
MOV #206, -(R5)
INC -(R5)
HALT
```

*MOV INSTRUCTION FAILED IN MODE 0

**MODE 2:**

```
MOV #TEMP, R0
LOAD ADDRESSES INTO REGS.
MOV #TEMP1, R1
MOV #TEMP2, R2
CLR TEMP2
MOV #125, (R0)
LOAD THE LOCATIONS
MOV (R0), (R1)
MOV (R1), (R2)
CMPB #125, TEMP2
CHECK IT
BEQ 1$
MOV #207, -(R5)
INC -(R5)
HALT
```

*MOV INSTRUCTION FAILED IN MODE 1

**EMODE:**

```
MOV #210, -(R5)
INC -(R5)
HALT
```

*MOV INSTRUCTION FAILED IN MODE 1 OR WRONG SEQUENCE
135  CHECK MODE 2 USING THE MOVB AND MOV INSTRUCTIONS

*:TEST: 55  CHECK MODE 2 USING THE MOVB AND MOV INSTRUCTIONS

:------------------------------------------:

MODE2:

155  CHECK MODE 2 USING THE MOVB AND MOV INSTRUCTIONS

SECTION: 0055

COPY

0055:

007326 021527 000055
007326 001050
007334 005215
007336 012700 000136
007342 012701 000140
007346 012702 000142
007352 105022
007354 112710 002522
007360 112022
007362 105201
007364 111167 170546
007370 105200
007372 112021
007374 132227 002522
007376 001050
007402 105767 170530
007406 001404
007410 012745 000211
007416 005245
007416 000000
007420 005741
007424 005023
007424 012740 125252
007430 012020
007432 011067 170500
007436 012121
007440 024227 125252
007444 001003
007446 005767 170464
007446 001404
007452 012745 000212
007460 005245
007462 000000

COPY

0055:

007326 021527 000055
007332 001050
007334 005215
007336 012700 000136
007342 012701 000140
007346 012702 000142
007352 105022
007354 112710 002522
007360 112022
007362 105201
007364 111167 170546
007370 105200
007372 112021
007374 132227 002522
007376 001050
007402 105767 170530
007406 001404
007410 012745 000211
007416 005245
007416 000000
007420 005741
007424 005023
007424 012740 125252
007430 012020
007432 011067 170500
007436 012121
007440 024227 125252
007444 001003
007446 005767 170464
007446 001404
007452 012745 000212
007460 005245
007462 000000

COPY
TEST: 60 CHECK MODE 5 USING THE MOVB AND MOV INSTRUCTIONS

MODES:

IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

START CLEAN

LOAD ADDRESSES

LOAD ADDRESSES OF ADDRESSES

ADJUST THE POINTERS

TEMP2 ---> TEMP1

ADJUST THE POINTERS

TEMP ---> TEMP2

ADJUST THE POINTERS

CHECK IT

ADJUST THE POINTERS

CHECK IT

OK, CONTINUE

INSTRUCTIONS FAILED IN MODE 5

START CLEAN

LOAD ADDRESSES OF ADDRESSES

ADJUST THE POINTERS

LOAD TEMP2

ADJUST THE POINTERS

TEMP2 ---> TEMP1

ADJUST THE POINTERS

TEMP ---> TEMP2

ADJUST THE POINTERS

CHECK IT

CHECK IT

OK, CONTINUE
HALT : INSTRUCTIONS FAILED IN MODE 5
; OR WRONG SEQUENCE

; TEST: 61 CHECK MODE 6 USING THE MOV.B AND MOV INSTRUCTIONS

--- MODE 6 ---

CMP (R5), #61
BNE EMODE6
INC (R5)
CLR TEMP2
MOV #TEMP1, RO
MOV #TEMP1, R1
MOV #TEMP2, R2
MOVB #252, (RO)
MOVB #252, (RO)
CMP #25252, TEMP
BEQ 25

1$: MOV #255, -(R5)
INC -(R5)
HALT

2$: CLR TEMP1
MOV #25252, (RO)
LOAD TEMP
MOV #6, (R2), (R0)
TEMP ----> TEMP1
CHECK IT
BEQ MODE7
OK, CONTINUE

EMODE6:
MOV #226, -(R5)
INC -(R5)
HALT

: INSTRUCTIONS FAILED IN MODE 6
: OR WRONG SEQUENCE
CHECK MODE 7 USING THE MOVB AND MOV INSTRUCTIONS

; TEST: 62

; MODE 7:

CMP (RS), #62
BNE EMODE7
INC (RS)

; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

CLR TEMP1

; START CLEAN

MOV #TEMP1,ADR1
MOV #TEMP1,ADR2
MOV #ADR1,R0
MOV #ADR1,R1
MOV #ADR2,R2

; LOAD ADDRESSES

MOV #252,a0,(RO)
MOV a-4(R2),a2,(RO)
CMP #252,TEMP1
BEQ 1$
MOV #227,-(RS)
INC -(RS)

; LOAD TEMP

MOV #125252,a0,(RO)
MOV a-4(R2),a2,(RO)
CMP #125252,TEMP1
BEQ TSTB1
INC -(RS)

; MODE 7 IS FAILING

MOV #230,-(RS)
HALT

; INSTRUCTIONS FAILED IN MODE 7
; OR WRONG SEQUENCE
; CHECK BYTE INSTRUCTIONS, NOT DESTINATION MODE 0

; TEST: 63  NEW INSTRUCTIONS USED IN THIS SECTION ARE TSTB, CLRb, MOVb

TSTB:

CMP (R5), #63
BNE TESTB1 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF TEST

2$: INC (R5)
MOV #TEMP, RO ; LOAD ADDRESSES

MOV #TEMP1, R1

Scc
CLRb (RO) ; CLEAR THE LOCATION

JSR PC, #Scc4 ; CHECK FOR CC = 4

TSTB (RO) ; CHECK JT

JSR PC, #Scc4 ; CHECK FOR CC = 4

MOVb #377, (R1) ; LOAD THE LOCATION

JSR PC, #Scc10 ; CHECK FOR CC = 10

TSTb (R1) ; CHECK IT

JSR PC, #Scc10 ; CHECK FOR CC = 10

MOV R0, R2 ; R2 IS NOW POINTING TO LOCATION TEMP

MOVb #200,0(R2) ; PLACE #200 IN LOCATION TEMP

MOVb (R2)+, -(R1) ; MOVE #200 TO LOCATION TEMP+1

CMP -(R1), #100200 ; CHECK THE DATA IN LOCATION TEMP

BEQ 4$ ; MOVb INSTRUCTION FAILED

MOV #251, -(R5)
INC -(R5)

HALT 4$; MOVb INSTRUCTION FAILED OR WRONG SEQUENCE

4$: CMP R1, R2 ; CHECK THE REGISTERS FOR PROPER VALUE

BEQ TSTB ; MOVb INSTRUCTION FAILED OR WRONG SEQUENCE
**TEST: 64**

NEW INSTRUCTIONS USED IN THIS SECTION ARE CMPB, BISB

---

**CMPPB:**

CMP (R5), #64

BNE ECMPPB1

; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

INC (R5)

MOV #TEMP2, R1

MOV #TEMP, R2

LOAD ADDRESS

MOV #77, (R1)

PLACE 77 IN LOCATION TEMP2

MOV #37, R4

R4 SHOULD CONTAIN #77777

BISB R4, (R2)

LOAD LOCATION

JR PC, BMS10

CHECK FOR CC = 10

CMPB R4, (R2)

CHECK COMPARE

BEQ 2$

CONTINUE IF OK

MOV #233, -(R5)

HALT

Halt

2$: CMPB (R1), (R2)

BPL 3$

CHECK IT AGAIN

CONTINUE IF OK

MOV #234, -(R5),

INC - (R5)

HALT

3$: CMPB (R2), (R1)

BMI BICB1

ONCE MORE

CONTINUE IF OK

ECMPBP1:

MOV #235, -(R5)

HALT

HALT

---

**TEST: 65**

NEW INSTRUCTIONS USED IN THIS SECTION ARE BICB, BITB

---

**BICB1:**

CMP (R5), #65

BEQ 2$

; IF IN WRONG SEQUENCE GO TO HLT BELOW

MOV #236, -(R5)

INC - (R5)

HALT

; PROGRAM IS IN WRONG SEQUENCE

2$: INC (R5)

MOV #TEMP, R3

LOAD ADDRESS

MOV #377, (R3)

LOAD LOCATION

MOV #TEMP1, R0

PLACE THE ADDRESS OF LOCATION TEMP1 IN R0

AND R1

MOV #252, (R11)$

PLACE #252 IN TEMP1

BITB 0(R0), (R5)

CLEAR EVERY OTHER BIT

JSR PC, BMS10

CHECK FOR CC = 7

BITB -1(R1), (R3)

CHECK IT

BEQ 4$

CONTINUE IF OK

MOV #37, -(R5)

INT - (R5)

---
NEW INSTRUCTIONS USED IN THIS SECTION ARE BICB, BITB

4$:      HALT
          BITB  #125,(R3)
          JSR  PC,#$CC1
          BISB  -(R1),(R3)
          BMI  6$:
          MOV  #260,-(R5)
          INC  -(R5)

6$:      MOV  #177,-(SP)
          BICB  (SP)+,(R3)
          JSR  PC,#$CC11
          BITB  #377,(R3)
          JSR  PC,#$CC11
          MOV  R3,RO
          MOV  #TEMP1,(RO)
          MOV  #377,(RO)+
          STC  V & C

BITB OR BISB INSTRUCTION FAILED
STORE #177 ON THE STACK
CLEAR ALL THE BITS EXCEPT SIGN BIT
CHECK FOR CC = 11
CHECK IT
CHECK FOR CC = 11
PLACE THE ADDRESS OF LOCATION TEMP IN RO
PLACE THE ADDRESS OF LOCATION TEMP1 IN TEMP
WRITE A 377 IN LOCATION TEMP1
SET V & C BITS
BIT CLEAR THE CONTENTS
OF TEMP1 TO THE CONTENTS OF TEMP1
CHECK FOR CC = 5
MAKE SURE THAT (RO) IS POINTING TO LOCATION TEMP1

8$:      HALT
          BICB  -(RO),@(RO)
          INC  -(R5)

BITC OR CMP INSTRUCTION FAILED IN THE SPECIFIC MODE
TEST LOCATION TEMP1

10$:     HALT
          BICB  -(RO),@(RO)

BITC INSTRUCTION FAILED
CLEAR THE LOCATION TEMP
NEW INSTRUCTIONS USED IN THIS SECTION ARE INCB, DECB

; TEST: 66 NEW INSTRUCTIONS USED IN THIS SECTION ARE INCB, DECB

; INCB1:

CMP (R5),#66
BNE EINCB1 : IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

INC (R5)
MOV #TEMP,R4 ; LOAD ADDRESS
MOV #177,(R4) ; TEMP LOCATION = 177
SEC
INCB (R4) ; ADD ONES INTO LOCATION
JSR PC, @$CC13 ; CHECK FOR CC = 13
MOV #376,(R4)
INCB (R4)+ ; MAKE R0 POINT TO CHECKING ROUTINE FOR CC = 11
JSR PC, @$CC11,RO ; CHECK FOR CC = 11
INCB (R4)+ ; DECORATE R4 BY 1
TSTB -R4) ; AND SP BY 2
MOV R4,(SP)+ ; PLACE THE ADDRESS OF TEMP ON THE STACK
CLC
INCB @-SP) ; CLEAR C BIT
JSR PC,@$CC4 ; INCREMENT THE CONTENTS OF LOCATION TEMP
CMPB @-SP)+ @R4) ; CHECK FOR CC = 4
BEG 2$: ; RESTORE STACK POINTER
SEC
INCB -1(R4) ; SET C BIT
JSR PC, @$CC1) ; CHECK FOR CC = 1
CMPS @-R4),#7 ; CHECK IT
BEG 2$: ; CONTINUE IF OK
MVB #263,-(R5) ; INCB INSTRUCTION FAILED
INC -(R5)
HALT

2$: SEC
DECB (R4) ; SUBTRACT ONES FROM LOCATION
JSR PC, @$CC5 ; CHECK FOR CC = 5
DECB (R4)+ ; CHECK FOR CC = 11
JSR PC,-(R0) ; CHECK FOR CC = 11
MOV #200,-(R4)
DECB -(R4)
JSR PC,$C13-$CC11(R0) ; CHECK FOR CC = 3
DECB 0(R4)
JSR PC, @$CC1) ; CHECK FOR CC = 1
CMPS @R0)+ @R16) ; CHECK FOR CC = 1
BEG EINCB1
MOV #244,-(R5) ; DECB INSTRUCTION FAILED OR SEQUENCE ERROR
INC -(R5)
HALT
**TEST: 67**  
**NEW INSTRUCTION IN THIS SECTION IS COMB**

**COMB1:**
- `CMP (R5), #67`
- `BEQ 1%`
- `MOV #245, -(R5)`  
  ; IF IN WRONG SEQUENCE GO TO HLT
- `INC -(R5)`
- `HALT`
  ; TEST IS IN WRONG SEQUENCE

**1%:**
- `INC (R5)`
- `MOV #TEMP, R5`
  ; LOAD ADDRESS
- `MOV #TEMP1, R4`
- `MOV #252, (R6)`
- `MOVB (R4)+, (R3)`
  ; LOAD EVERY OTHER BIT
- `SCC`
- `CMDB (R3)`
  ; 1'S COMPLEMENT
- `JSR PC, @SEC1`
  ; CHECK FOR C = 1
- `CMDB #125, (R3)`
  ; CHECK IT
- `BEQ 2%`
  ; CONTINUE IF OK
- `MOV #246, -(R5)`
- `INC -(R3)`
  ; COMB INSTRUCTION FAILED

**2%:**
- `SCC`
- `CMDB (R3)`
  ; COMPLEMENT BACK
- `JSR PC, @SEC11`
  ; CHECK FOR CC = 1
- `MOV R4, R0`
- `CMDB -(R0), (R3)`
  ; CHECK IT
- `BEQ 3%`
  ; CONTINUE IF OK
- `MOV #247, -(R5)`
- `INC -(R5)`
  ; COMB INSTRUCTION FAILED

**3%:**
- `MOV #377, (R4)+`
- `MOV -(R4)+, (R3)`
  ; PLACE #377 IN (R3)
- `SEC`
- `CMDB (R3)`
  ; CHECK FOR CC = 5
- `JSR PC, @SEC5`
**NEGB1:**

```
NEGB1:
CMP (R5), #70
BNE ENNEGB1 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
INC (R5)
MOV #TEMP, R0 ; LOAD ADDRESS
MOVB #7, (R0) ; LOAD THE LOCATION
NEG (R0) ; 2'S COMPLEMENT
JSR PC, #&CC11 ; CHECK FOR CC = 11
CMPB #377, (R0) ; CHECK IT
BEG 2$ ; CONTINUE IF OK
MOV #250, -(R5) ; CONTINUE
INC -(R5)
HLT
```

**NEGB1:**

```
2$:
MOV #200, (R0) ; NEGB INSTRUCTION FAILED
NEG (R0) ; 2'S COMPLEMENT
JSR PC, #&CC13 ; CHECK FOR CC = 13
CMPB #200, (R0) ; CHECK IT
BEG ROLB1 ; CONTINUE IF OK
```

**ENNEGB1:**

```
MOV #251, -(R5) ; WRONG RESULT AT TEMP OR WRONG SEQUENCE
INC -(R5)
HLT
```

**TEST: 71 NEW INSTRUCTION IN THIS SECTION IS ROLB**

**ROLB1:**

```
ROLB1:
CMP (R5), #71
BNE EROLB1 ; IF IN WRONG SEQUENCE GO TO HLT ABOVE
INC (R5)
MOV #TEMP1, R1 ; LOAD ADDRESS
MOVB #40, (R1) ; LOAD LOCATION
CCC
ROLB (R1) ; CLEAR FLAGS
ROLB (R1) ; SHIFT
ROLB (R1) ; SHIFT
JSR PC, #&CC12 ; CHECK FOR CC = 2
CMPB #200, (R1) ; CHECK IT
BEG 1$ ; CONTINUE IF OK
MOV #252, -(R5) ; CONTINUE
INC -(R5)
```

**1$:**

```
ROLB (R1) ; ROLB INSTRUCTION FAILED
SHIFT
JSR PC, #&CC7 ; CHECK FOR CC = 7
ROLB (R1) ; SHIFT
ROLB (R1) ; SHIFT
CMPPB #1, (R1) ; CHECK IT
BEG ROLB1 ; CONTINUE IF OK
```

**EROLB1:**

```
MOV #253, -(R5)
INF -(R5)
```

**NEW INSTRUCTION IN THIS SECTION IS NRB**

**NEGB:**

```
NEGB:
CMP (R5), #70
BNE ENNEGB0 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
INC (R5)
MOV #TEMP, R0 ; LOAD ADDRESS
MOVB #7, (R0) ; LOAD THE LOCATION
NEG (R0) ; 2'S COMPLEMENT
JSR PC, #&CC11 ; CHECK FOR CC = 11
CMPB #377, (R0) ; CHECK IT
BEG 2$ ; CONTINUE IF OK
MOV #250, -(R5) ; CONTINUE
INC -(R5)
HLT
```

**NEGB:**

```
2$:
MOV #200, (R0) ; NEGB INSTRUCTION FAILED
NEG (R0) ; 2'S COMPLEMENT
JSR PC, #&CC13 ; CHECK FOR CC = 13
CMPB #200, (R0) ; CHECK IT
BEG ROLB1 ; CONTINUE IF OK
```

**ENNEGB0:**

```
MOV #251, -(R5) ; WRONG RESULT AT TEMP OR WRONG SEQUENCE
INC -(R5)
HLT
```

**TEST: 70 NEW INSTRUCTION IN THIS SECTION IS NRB**
NEW INSTRUCTION IN THIS SECTION IS ROLB

HALT : WRONG RESULT AT TEMP1 OR WRONG SEQUENCE
**TEST: 72** NEW INSTRUCTION IN THIS SECTION IS RORB

```
2617    CMP   (R5),#72
2618    BNE    ERORB1
2619    INC   (R5)
2620    MOV   #TEMP1,R2   ; LOAD ADDRESS
2621    MOVB  #4,(R2)   ; LOAD LOCATION
2622    CCC   (R2)   ; CLEAR FLAGS
2623    ROTB  (R2)   ; SHIFT
2624    CMPB  #1,(R2)   ; CHECK IT
2625    BEQ   1$   ; CONTINUE IF OK
2626    MOV   #254,-(R5)   ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
2627    INC   -(R5)
2628    HALT   ; RORB INSTRUCTION FAILED
1$:    RORB  (R2)   ; SHIFT
2629    JSR   PC,=CC17   ; CHECK FOR CC = 7
2630    JSR   PC,=CC7   ; SHIFT
2631    JSR   PC,=CC12   ; CHECK FOR CC = 12
2632    CMPB  #200,(R2)   ; CHECK IT
2633    BEQ   1$   ; CONTINUE IF OK
2634    MOV   #255,-(R5)   ; WRONG RESULT AT TEMP1 OR WRONG SEQUENCE
2635    INC   -(R5)
2636    HALT
```

**TEST: 73** NEW INSTRUCTION IN THIS SECTION IS ASLB

```
2651    CMP   (R5),#73
2652    BEQ   2$   ; IF IN WRONG SEQUENCE GO TO HLT BELOW
2653    MOV   #256,-(R5)
2654    INC   -(R5)
2655    HALT   ; PROGRAM IS IN WRONG SEQUENCE
2$:    INC   (R5)
2657    MOV   #TEMP1,R3   ; LOAD ADDRESS
2658    MOVB  #40,(R3)   ; LOAD LOCATION
2659    CCC   (R3)   ; CLEAR FLAGS
2660    ASLB  (R3)   ; SHIFT
2661    JSR   PC,=CC17   ; CHECK FOR CC = 7
2662    JSR   PC,=CC7   ; SHIFT
2663    JSR   PC,=CC12   ; CHECK FOR CC = 12
2664    CMPB  #200,(R3)   ; CHECK IT
2665    BEQ   2$   ; CONTINUE IF OK
2666    MOV   #257,-(R5)   ; ASLB INSTRUCTION FAILED
2667    INC   -(R5)
2668    HALT
4$:    ASLB  (R3)   ; SHIFT
2670    JSR   PC,=CC17   ; CHECK FOR CC = 7
2671    ASLB  (R3)   ; SHIFT
2672    JSR   PC,=CC7   ; SHIFT
```

NEW INSTRUCTION IN THIS SECTION IS ASLB

JSR PC, #C
; CHECK FOR CC = 4
**TEST: 74**

NEW INSTRUCTION IN THIS SECTION IS ASRB

-------------------------------

**ASRB1:**

1$:
- CMP (R5), #74
- BNE EASRB1

; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

- INC (R5)
- MOV #TEMP1, R4
- MOV #TEMP2, R3
- MOVB #4, (R4)
- CCC
- ASRB (R4)
- ASRB (R4)
- ASRB (R4)
- CMPB #1, (R4)
- BEQ 2$
- MOV #260, -(R5)
- INC -(R5)

; LOAD ADDRESSES

2$:
- ASRB (R4)
- JSR PC, #14CC
- ASRB (R4)
- ASRB (R4)
- ASRB (R4)
- MOV #202, (R3)
- ASRB (R3)
- ASRB (R3)
- JSR PC, #14CC
- CMPB #340, (R3)
- BEQ ADCB1

; CHECK FOR CC = 7

EASRB1:

- MOV #261, -(R5)
- INC -(R5)
- HALT

; ASRB INSTRUCTION FAILED

; CONTINUE IF OK

**TEST: 75**

NEW INSTRUCTION IN THIS SECTION IS ADCB

-------------------------------

**ADCB1:**

1$:
- CMP (R5), #75
- BEQ 2$
- MOV #262, -(R5)
- INC -(R5)

; IF IN WRONG SEQUENCE GO TO HLT BELOW

2$:
- INC (R5)
- MOV #TEMP2, R0
- MOV (R0)
- CLR (R0)
- ADCB (R0)
- JSR PC, #014CC
- SEC
- ADCB (R0)
- ADCB (R0)

; PROGRAM IS IN WRONG SEQUENCE

; LOAD ADDRESS

; CLEAR THE LOCATION

; CLEAR FLAGS

; ADD C BIT = 0

; CHECK FOR CC = 4

; C = 1

; C = 1

; AGAIN
NEW INSTRUCTION IN THIS SECTION IS ADCB
**SBCB1:**

CMP (R5), #76 ; IF IN WRONG SEQUENCE GO TO HLT

BEQ 1% ; TEST IS IN WRONG SEQUENCE

MOV #265, -(R5)

INC -(R5)

HALT

1%: INC (R5) ; LOAD ADDRESS

MOV #TEMP2, R1 ; LOAD LOCATION

MOV R1, #3, (R1) ; CLEAR FLAGS

SUB R1, #8, #CC0

JMP #3, (R1) ; CHECK FOR CC = 0

BEQ 2% ; CHECK IT

MOV #246, -(R5) ; CONTINUE IF OK

INC -(R5)

HALT

2%: SEC ; SBCB INSTRUCTION FAILED

SBCB (R1) ; C = 1

SBCB (R1) ; C = 1

JSR PC, #CC0 ; CHECK FOR CC = 0

CMPB #1, (R1) ; CHECK IT

BEQ 3% ; CONTINUE IF OK

MOV #267, -(R5)

INC -(R5)

HALT

3%: SEC ; SBCB INSTRUCTION FAILED

SBCB (R1) ; C = 1

JSR PC, #CC4 ; CHECK FOR CC = 4

SEC ; C = 1

SBCB (R1) ; C = 1

JSR PC, #CC11 ; CHECK FOR CC = 11

CMPB #377, (R1) ; CHECK IT

BEQ 4% ; CONTINUE IF OK

MOV #270, -(R5)

INC -(R5)

HALT

4%: MOV R1, #200, (R1) ; SBCB INSTRUCTION FAILED

HALT
NEW INSTRUCTION IN THIS SECTION IS SBCB

SEC
SBCB (R1)
JSR PC, @W$CC2

; C=1
; SUBTRACT C Bit = 1
; CHECK FOR CC = 2
**Check Word Instructions, Not Destination Mode 0**

----------

*Test: 77 New Instructions Used in This Section Are TST, CLR, MOV*

----------

**TST1:**

```
CMP   (R5), #77 ; IF IN WRONG SEQUENCE GO TO HLT
BEQ   1$ ;
MOV   #271, -(R5)
INC   -(R5)
HALT
1$:  INC   (R5)
```

; TEST IS IN A WRONG SEQUENCE

```
MOV   #TEMP, R1 ; LOAD ADDRESSES
MOV   #TEMP1, RO ;
CLR   (RO) ; CLEAR THE LOCATION
JSR   PC, @MSCC4 ; CHECK FOR CC = 4
TST   RO, -(RO) ;
JSR   PC, @MSCC4 ; CHECK FOR CC = 4
MOV   RO, -(RO) ;
MOV   @177777, @RO+ ; LOAD THE LOCATION
JSR   PC, @MSCC10 ; CHECK FOR CC = 10
TST   (R1) ; CHECK IT
JSR   PC, @MSCC10 ; CHECK FOR CC = 10
```
CMP: 013012 CMP (R5), #100
013016 BNE EEMP1 1$: INC (R5)
013017 MOV #TEMP1,R2 2$: LOAD ADDRESS
013018 MOV #TEMP,RO PLACE THE ADDRESS OF TEMP IN RO
013019 MOV #177777,(R0)+ PLACE #177777 IN LOCATION TEMP AND INC. RO BY 2
01301A BIS -(R0),(R2) LOAD LOCATION temp
01301B JSR PC,#8CC10 CHECK FOR CC = 10
01301C CMP (R2)+,#177777 1$: CONTINUE IF OK
01301D BEQ 2$ CMP OR BIS INSTRUCTION FAILED
01301E MOV #272,-(R5) CHECK R2 TO CONTAIN ADDRESS OF TEMP+2
01301F INC -(R5)
013020 HALT
013021 013062 020227 000142 3$: NO AUTO INCREMENT
013025 MOV #77,-(R2) CHECK IT AGAIN
013026 JSR PC,#85CC1 CHECK FOR CC = 1
013027 CMP #777777,(R2)+ ONCE MORE
013028 JSR PC,#85CC13 CHECK FOR CC = 13
013029 CMP -(R2),#777777 PLACE THE ADDRESS OF TEMP IN LOCATION TEMP
01302A JSR PC,#8CC10 AND INCREMENT R2 BY 2
01302B MOV #ADR1,(R4) PLACE THE ADDRESS OF ADR1 IN ADR POINTED BY R4
01302C MOV #125252,(R4)+ PLACE THE #125252 IN LOCATION ADR1
01302D BIS @-2(R4),#(R2)+ SET EVERY OTHER BIT AT LOCATION TEMP2
01302E MOV #ADR1,(R4) AND INCREMENT R2 BY 2
01302F MOV @-2(R4),#177777 PLACE ADDRESS OF TEMP2 IN RO
013030 CMP R2,RO TEMP2 SHOULD CONTAIN ALL 1'SS
013031 BEQ 4$ PLACE ADDRESS OF TEMP2 IN RO
013032 MOV #272,-(R5) TEMP2 SHOULD CONTAIN ALL 1'SS
013033 INC -(R5)
013034 HALT
013035 013060 057432 177777
013039 MOV R2,RO
01303A CMP @-(R0),#177777
01303B BEQ 4$ CMP OR BIS INSTRUCTIONS FAILED IN MODES OTHER THAN 0
01303C MOV #272,-(R5) R2 SHOULD CONTAIN THE ADDRESS FOR TEMP2
01303D INC -(R5) I.E. TEMP1+2
01303E 013064 010200
01303F 013066 025027 177777
013041 MOV @-(R0),#177777 MODE 5 IS FILLING
013042 BEQ 5$ PLACE A 0 IN LOCATION TEMP
013043 MOV #275,-(R5) PLACE ADDRESS OF TEMP IN LOCATION TEMP2
013044 INC -(R5)
013045 HALT
013046 013072 000000
013049 MOV #272,-(R5)
01304A INC -(R5)
01304B HALT
01304C 013073 020227 000142
01304E MOV R2,TEMP1+2
BIC:

CMP (R5), #101
INC (R5)
MOV #TEMP1,R3
MOV #177777,(R3)
MOV #ADR1,(R4)
MOV #ADR1,(R4)
MOV #ADR1,ADR1+177777
MOV TEMP1,R0
MOV #125252,(R0)
SET EVERY OTHER BIT AT LOCATION TEMP1

HALT

BIC (R0)+,(R3)
JSR PC,+PC
BIT (R0),(R3)
CHECK IF CC = 1
CONTINUE IF OK

MOV #27, -(R5)
INC -(R5)

HALT

BIT #52525,(R3)
PC,+PC
CHECK FOR CC = 1
SET THE BITS THAT WERE CLEARED
CONTINUE IF OK

MOV #300, -(R5)
INC -(R5)

HALT

MOV #77777,(R0)+
MOV RO,R0
BIC -2(R2),(R3)
JSR PC,+PC
TIMES RO,#TEMP1+2
BEO 3
MOV #301, -(R5)
INC -(R5)

HALT

MOV RO,(R0)+
JSR PC,+PC
SET THE 00 Bits
CLEAR RO
BIT (R0),RO
CHECK FOR CC = 5
CONTINUE IF OK

MOV RO,(R0)+
BIC 0(SP), - (R3)
MOV (SP), - (R3)
SUB RO,#52525
TEMP SHOULD CONTAIN #52525
NEW INSTRUCTIONS USED IN THIS SECTION ARE BIC, BIT

E843 013472 012745 000302  MOV  #302,-(R5)
2944 013476 005245  INC  -(R5)
2945 013500 000000  HALT
2946 013502 012700 000144 48:  MOV  #TEMP2+2,RO
2947 013506 010340  MOV  R5,-(RO)
2948 013510 014330  MOV  -(R5),3(R0)+
2949 013512 000265  SEVC
2950 013514 035026  BIT  a-(RO),3(SP)+
2951 013516 004757 017172  JSR  PC,3#8CCS
2952 013522 020627 000456  CMP  SP,#START
2953 013526 001404  BEQ  INC1
2954 013530 000000  FB1C:  MOV  #303,-(R5)
2955 013530 013745 000303  INC  -(R5)
2956 013534 005245  HALT
2957 013536 000000  ; STACK POINTER FOUL ED UP OR SEQUENCE ERROR
**TEST: 102** NEW INSTRUCTIONS USED IN THIS SECTION ARE INC, DEC

**INCl:**
- CMP (R5), #102
  
  *IF IN WRONG SEQUENCE GO TO HLT BELOW*
- BEQ 2%
- MOV #306, -(R5)
- INC -(R5)

2%
- INC (R5)
  
  *PROGRAM IS IN WRONG SEQUENCE*
- MOV #TEMP1, R4
  
  *TEMP1 = 77777*
- MOV #77777, (R4)
- SEC
- INC (R4)
  
  *ADD ONES INTO LOCATION*
- JSR PC, #5C13
  
  *CHECK FOR CC = 13*
- MOV #17777, (R4)
- MOV #TEMP, R0
  
  *RO IS POINTING TO LOCATION TEMP*
- MOV #5C11, (R0)
  
  *PLACE ADDRESS OF SUBROUTINE TO CHECK CC = 11 IN LOCATION TEMP*
- INC (R4)
  
  *CHECK FOR CC = 11*
- JSR PC, @ (R0) +
  
  *CHECK FOR CC = 5*
- INC (R4)
- JSR PC, #5C5
  
  *CHECK FOR CC = 5*
- INC (R4)
- JSR PC, #5C1
  
  *CHECK FOR CC = 1*
- CMP 0(R4), #1
  
  *CHECK IF OK*
- BEQ 4%
  
  *CONTINUE IF OK*
- MOV #305, -(R5)
- INC -(R5)
  
  *INC INSTRUCTION FAILED*
- SEC
- DEC (R4)
  
  *SUBTRACT ONES FROM LOCATION*
- JSR PC, #5C5
  
  *CHECK FOR CC = 5*
- DEC (R4)
- JSR PC, @2(R0)
  
  *CHECK FOR CC = 11*
- MOV #100000, (R4)
- DEC (R4)
- JSR PC, #5C3
  
  *CHECK FOR CC = 3*
- DEC (R4)
- JSR PC, #5C1
  
  *CHECK FOR CC = 1*

**TEST: 103** NEW INSTRUCTION IN THIS SECTION IS COM

**COM1:**
- CMP (R5), #103
  
  *IF IN WRONG SEQUENCE GO TO HLT*
- BEQ 1%
- MOV #706, -(R5)
- INC -(R5)
- HLT
- INC (R5)
  
  *TEST IS IN WRONG SEQUENCE*
- MOV #TEMP, R3
  
  *LOAD ADDRESS*
<table>
<thead>
<tr>
<th>Line</th>
<th>Address 1</th>
<th>Address 2</th>
<th>Address 3</th>
<th>Instruction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>5014</td>
<td>013736</td>
<td>012713</td>
<td>125252</td>
<td>MOV #125252,(R3)</td>
<td>LOAD EVERY OTHER BIT</td>
</tr>
<tr>
<td>5015</td>
<td>013742</td>
<td>000277</td>
<td></td>
<td>SCL</td>
<td></td>
</tr>
<tr>
<td>5016</td>
<td>013744</td>
<td>005163</td>
<td>200000</td>
<td>COM 0(R3)</td>
<td>1'S COMPLEMENT</td>
</tr>
<tr>
<td>5017</td>
<td>013750</td>
<td>004737</td>
<td>017066</td>
<td>JSR PC,#$CC1</td>
<td>CHECK FOR CC = 1</td>
</tr>
<tr>
<td>5018</td>
<td>013744</td>
<td>027713</td>
<td>052525</td>
<td>CMP #52525,(R3)</td>
<td>CHECK IT</td>
</tr>
<tr>
<td>5019</td>
<td>013750</td>
<td>004737</td>
<td>012525</td>
<td>BEQ 2#</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>5020</td>
<td>013760</td>
<td>004737</td>
<td>000307</td>
<td>MOV #307,-(R5)</td>
<td></td>
</tr>
<tr>
<td>5021</td>
<td>013766</td>
<td>005245</td>
<td></td>
<td>INC -(R5)</td>
<td></td>
</tr>
<tr>
<td>5022</td>
<td>013770</td>
<td>000000</td>
<td></td>
<td>HALT</td>
<td>COM INSTRUCTION FAILED</td>
</tr>
<tr>
<td>5023</td>
<td>013772</td>
<td>000277</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5024</td>
<td>013774</td>
<td>005123</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5025</td>
<td>013776</td>
<td>004737</td>
<td>012756</td>
<td>JSR PC,#$CC1</td>
<td>CHECK FOR CC = 11</td>
</tr>
<tr>
<td>5026</td>
<td>014002</td>
<td>022743</td>
<td>125252</td>
<td>CMP #125252,-(R3)</td>
<td>CHECK IT</td>
</tr>
<tr>
<td>5027</td>
<td>014006</td>
<td>001404</td>
<td></td>
<td>BEQ 3#</td>
<td>CONTINUE IF OK</td>
</tr>
<tr>
<td>5028</td>
<td>014010</td>
<td>012745</td>
<td>000310</td>
<td>MOV #310,-(R5)</td>
<td></td>
</tr>
<tr>
<td>5029</td>
<td>014014</td>
<td>005245</td>
<td></td>
<td>INC -(R5)</td>
<td></td>
</tr>
<tr>
<td>5030</td>
<td>014016</td>
<td>000000</td>
<td></td>
<td>HALT</td>
<td>COM INSTRUCTION FAILED</td>
</tr>
<tr>
<td>5031</td>
<td>014020</td>
<td>010300</td>
<td></td>
<td></td>
<td>RO IS NOW POINTING TO LOCATION TEMP1</td>
</tr>
<tr>
<td>5032</td>
<td>014022</td>
<td>02770</td>
<td>177777</td>
<td>MOV R3,RO</td>
<td></td>
</tr>
<tr>
<td>5033</td>
<td>014026</td>
<td>000277</td>
<td></td>
<td>MOV #177777,(R0)</td>
<td></td>
</tr>
<tr>
<td>5034</td>
<td>014030</td>
<td>005110</td>
<td></td>
<td>SCL (RO)</td>
<td></td>
</tr>
<tr>
<td>5035</td>
<td>014032</td>
<td>004737</td>
<td>017172</td>
<td>JSR PC,#$CC5</td>
<td>CHECK FOR CC = 5</td>
</tr>
</tbody>
</table>
**TEST: 104**  NEW INSTRUCTION IN THIS SECTION IS NEG

```
NEG1:

CMP (R5), #104
BNE ENEG1 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

1$: INC (R5) ; LOAD ADDRESS
MOV #TEMP1,R4 ; LOAD THE LOCATION
MOV #1,(R4)+
MOV R4,R2
MOV #1000000,0(R2)
NEG -(R4)
JSR PC,#$C11 ; CHECK FOR CC = 11
CMP #177777,(R4)+ ; CHECK IT
BEQ 2$ ; CONTINUE IF OK
MOV #311,-(R5)
INC -(R5)
HALT ; NEG INSTRUCTION FAILED

2$: MOV 0(R4), -(R4) ; TEMP2 CONTAINS THE LARGEST NEGATIVE NUMBER
NEG (R4) ; 2'S COMPLEMENT
JSR PC,#$C13 ; CHECK FOR CC = 13
CMP 0(R2), (R4) ; CHECK IT
BEQ ROL1 ; CONTINUE IF OK

ENEG1:

MOV #312,-(R5)
INC -(R5)
HALT ; WRONG RESULT IN TEMP2 OR WRONG SEQUENCE
```

**TEST: 105**  NEW INSTRUCTION IN THIS SECTION IS ROL

```
ROL1:

CMP (R5), #105
BNE EROL1 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
INC (R5)
MOV #TEMP2,R1
MOV #200000,(R1)
MOV #1000000,(R1)+ ; LOAD ADDRESS
ROL (R1)+ ; LOAD LOCATION
ROL -(R1)
ROL (R1)
JSR PC,#$C12 ; CHECK FOR CC = 12
CMP #1000000,(R1) ; CHECK IT
BEQ 1$ ; CONTINUE IF OK
MOV #313,-(R5)
INC -(R5)
HALT ; ROL INSTRUCTION FAILED

1$: ROL 0(R1) ; SHIF
JSR PC,#$C7 ; CHECK FOR CC = 7
MOV R1,R2 ; R2 IS NOW POINTING TO LOCATION TEMP2
ROL (R2) ; SHIF
CMP #7,(R1) ; CHECK IT
BEQ ROL1 ; CONTINUE IF OK
```

000104
EROL1: MOV     #314,-(RS)
       INC     -(RS)
       HALT

; WRONG RESULT AT TEMP2 OR WRONG SEQUENCE
**TEST: 106 NEW INSTRUCTION IN THIS SECTION IS ROR**

```
ROR:
  CMP (R5), #106
  BNE ERROR1 ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
  INC (R5)
  MOV #TEMP2, R2 ; LOAD ADDRESS
  MOV #4, (R2) ; LOAD LOCATION
  CCC
  CLR FLAGS
  ROR (R2)
  ROR (R2) ;
  CMP #1, (R2) ; CHECK IT
  BEQ 1$ ; CONTINUE IF OK
  MOV #315, -(R5)
  INC -(R5)
  HALT

1$:
  ROR (R2) ; ROR INSTRUCTION FAILED
  SHFT
  JSR PC, #CC7 ; CHECK FOR CC = 7
  ROR (R2) ;
  JSR PC, #CC12 ; CHECK FOR CC = 12
  CMP #100000, (R2) ; CHECK IT
  BEQ ASL1 ; CONTINUE IF OK

ERROR:
  MOV #316, -(R5)
  INC -(R5)
  HALT ; WRONG RESULT AT TEMP2 OR WRONG SEQUENCE
```
Test: 110 New instruction in this section is ASR

ASR1:

1:
- CMP (R5), #110
  - BNE EASR1
  - INC (R5)
  - IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST

2:
- MOV #TMP2, R4
- MOV #TMP, R3
- MOV #4, (R4)
- CCC (R4)
- ASR (R4)
- ASR (R4)
- CMP #7, (R4)
- BEO 2%
- MOV #321, -(R5)
- INC -(R5)
- HALT
- ASR (R4)
- JSR PC, #CC7
- JSR (R4)
- JSR PC, #CC4
- MOV #100002,(R3)
- HALT
- ASR (R3)
- ASR (R3)
- JSR PC, #CC11
- CMP #160000,(R3)
- BEO ADC1
- INC -(R5)
- HALT

EASR1:
- MOV #322, -(R5)
- ASR INSTRUCTION FAILED
- CONTINUE IF OK

Test: 111 New instruction in this section is ADC

ADC1:

1:
- CMP (R5), #111
- BEO 2%
- MOV #323, -(R5)
- INC -(R5)
- BEO ADC1

2:
- MOV #TMP, R0
- CLR (R0)
- CCC (R0)
- ADD (R0)
- JSR PC, #CC4
- SEC (R0)
- SEC (R0)
- SEC (R0)
- AGIAN

Program is in wrong sequence

Wrong result in TEMP or wrong sequence
NEW INSTRUCTION IN THIS SECTION IS ADC

3209 014622 004737 017046  JSR PC,#$E00 ; CHECK FOR CC = 0
3211 014626 022710 000002  CMP #2,(R0) ; CHECK IT
3212 014632 001404  BEQ 4$ ; CONTINUE IF OK
3213 014634 012765 000324  MOV #324,-(R5)
3214 014660 005245  INC -(R5)
3215 014662 000000  HALT               ; ADC INSTRUCTION FAILED
3216 014664 012710 077777  MOV #77777,(R0) ; LOAD LARGEST POSITIVE NUMBER
3217 014665 000261  SEC               ; <|=1
3218 014665 005510  ADC (R0)          ; ADD C BIT=1
3219 014666 004737 173000  JSR PC,#$C12 ; CHECK FOR CC = 12
3220 014666 022710 100000  CMP #$100000,(R0) ; CHECK IT
3222 014667 001404  BEQ 6$ ; CONTINUE IF OK
3223 014667 007265  MOV #325,-(R5)
3224 014672 005245  INC -(R5)
3225 014674 000000  HALT               ; ADC INSTRUCTION FAILED
3226 014676 012710 177777  MOV #-1,(R0) ; LOAD -1
3227 014704 000261  SEC               ; C=1
3228 014706 005510  ADC (R0)          ; ADD C BIT=1
3229 014706 004737 017777  JSR PC,#$C5 ; CHECK FOR CC = 5
SBC:

CMP (R5), #112
BEQ 1%
MOV #326, -(R5)
INC -(R5)

1%:

INC (R5)
MOV #160, -R1
LOAD ADDRESS
MOV #5, (R1)
LOAD LOCATION
CC
SUBTRACT C BIT=0
CLEAR FLAGS
SUBRACT C BIT=0
CHECK FOR CC = 0
CHECK IT

BEQ 2$
MOV #332, -R5
INC -(R5)

2$

SEC
SBC (R1)
SUBTRACT C BIT=1
C-1
SEC
SBC (R1)
SUBTRACT C BIT=1
C-1
JMP
PC, #0000
CHECK FOR CC = 0
CHECK IT
BEQ 3$
MOV #333, -(R5)
INC -(R5)

3$

SEC
SBC (R1)
SUBTRACT C BIT=1
C-1
SEC
SBC (R1)
SUBTRACT C BIT=1
C-1
JMP
PC, #0000
CHECK FOR CC = 0
CHECK IT
BEQ 4$
MOV #100000, (R1)
LOAD R1
C-1
SEC
SUBRACT C BIT = 1
C-1
JMP
PC, #0000
CHECK FOR CC = 0
CHECK IT

4$

HALT
SBC INSTRUCTION FAILED

SXT:

CMP (R5), #113
BNF ESXT1

ESXT1:

SBC INSTRUCTION FAILED

IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Opcode 1</th>
<th>Opcode 2</th>
<th>Immediate</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>5284</td>
<td>015102</td>
<td>005215</td>
<td>000140</td>
<td></td>
<td>INC (R5) LOAD ADDRESS</td>
</tr>
<tr>
<td>5285</td>
<td>015104</td>
<td>012702</td>
<td>00001012</td>
<td></td>
<td>CLN (R2) CLEAR LOCATIONS</td>
</tr>
<tr>
<td>5286</td>
<td>015110</td>
<td>000027</td>
<td>000027</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5287</td>
<td>015112</td>
<td>000254</td>
<td>000254</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5288</td>
<td>015114</td>
<td>006712</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5289</td>
<td>015116</td>
<td>006712</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5290</td>
<td>015120</td>
<td>006733</td>
<td>017172</td>
<td></td>
<td>SXT (R2) SIGN EXTEND</td>
</tr>
<tr>
<td>5291</td>
<td>015124</td>
<td>005712</td>
<td>005712</td>
<td></td>
<td>JSR PC, #5CC5 CHECK FOR CC = 5</td>
</tr>
<tr>
<td>5292</td>
<td>015126</td>
<td>001404</td>
<td>001404</td>
<td></td>
<td>TST (R2) LOCATION SHOULD STILL BE 0</td>
</tr>
<tr>
<td>5293</td>
<td>015130</td>
<td>012765</td>
<td>00032</td>
<td>3232</td>
<td>MOV #332,-(R5) CONTINUE IF OK</td>
</tr>
<tr>
<td>5294</td>
<td>015134</td>
<td>005245</td>
<td></td>
<td></td>
<td>INC -(R5)</td>
</tr>
<tr>
<td>5295</td>
<td>015136</td>
<td>000000</td>
<td></td>
<td></td>
<td>HLT</td>
</tr>
<tr>
<td>5296</td>
<td>015140</td>
<td>00273</td>
<td>00273</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5297</td>
<td>015142</td>
<td>004733</td>
<td>017256</td>
<td></td>
<td>SENVC (R2) SXT INSTRUCTION FAILED</td>
</tr>
<tr>
<td>5298</td>
<td>015144</td>
<td>004733</td>
<td>017256</td>
<td></td>
<td>SET N, V &amp; C BITS</td>
</tr>
<tr>
<td>5299</td>
<td>015150</td>
<td>022712</td>
<td>022712</td>
<td></td>
<td>JSR PC, #5CC11 SIGN EXTEND</td>
</tr>
<tr>
<td>5300</td>
<td>015154</td>
<td>001404</td>
<td>001404</td>
<td></td>
<td>CMP #1, (R2) CHECK FOR CC = 11</td>
</tr>
<tr>
<td>5301</td>
<td>015156</td>
<td>012745</td>
<td>000333</td>
<td></td>
<td>BEQ #1, (R2) LOCATION SHOULD NOT HAVE -1</td>
</tr>
<tr>
<td>5302</td>
<td>015156</td>
<td>012745</td>
<td>000333</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5303</td>
<td>015162</td>
<td>005245</td>
<td>005245</td>
<td></td>
<td>MOV #333,-(R5) CONTINUE IF OK</td>
</tr>
<tr>
<td>5304</td>
<td>015164</td>
<td>000000</td>
<td>000000</td>
<td></td>
<td>INC +(R5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HLT</td>
</tr>
</tbody>
</table>

**ESXT:** MOV #333,-(R5) INC +(R5) HLT

; WRONG RESULT IN TEMP1 OR WRONG SEQUENCE
NEW INSTRUCTION IN THIS SECTION IS SWAB

**SWAB**:  
```
CMP (R5), #114   ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
BNE ESWAB1
INC (R5)
MOV #15362, R3  ; LOAD ADDRESS
MOV #125125, R3 ; LOAD BIT PATTERN INTO LOCATION
SPL CLN
CLN SWAB (R3)  ; SWAP BYTES OF LOCATIONS
JSR PC @$$CC10 ; CHECK FOR CC = 10
CLR #2652, R3  ; CHECK IT
BEQ 1$ ; CONTINUE IF OK
MOV #334, -(R5)  ; SWAB INSTRUCTION FAILED
INC -(R5)
HALT 1$:  
```
```
MOV #377, R3
SPL SCC
CLZ SWAB 0(R3)  ; CHECK FOR LC = 4
JSR PC @$$CC4
CMP #177400, R3
BEQ XOR1
ESWAB1:  
MOV #335, -(R5)  ; WRONG RESULT IN TEMP2 OR WRONG SEQUENCE
INC -(R5)
HALT
```

**TEST: 115**  
NEW INSTRUCTION IN THIS SECTION IS XOR

**XOR**:  
```
CMP (R5), #185  ; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
BNE EXOR1
INC (R5)
MOV #17252, R4  ; LOAD LOCATIONS
MOV #1, R4
MOV #1, TEMP1
SPL SCC
CLZ XOR R4, TEMP1  ; SHOULD PRODUCE 0'S IN TEMP1
JSR PC @$$CC5
MOV #77777, TEMP1
MOV TEMP1, R0  ; PLACE THE ADDRESS OF TEMP1 IN R0
SEVCC
SEVC
CLZ XOR R4, (R0)  ; CHECK FOR CC = 11
JSR PC @$$CC11
MOV #152525, R1  ; LOAD LOCATIONS
MOV #52525, (R0)+
SPL SCC
XOR R1, -(R0)  ; SHOULD PRODUCE ALL 1'S IN TEMP1
```
NEW INSTRUCTION IN THIS SECTION IS XOR

JMP PC,#CC11 : CHECK FOR CC = 11
CMP #-1,#TEMP1 : CHECK IT
BEQ ADD1 : CONTINUE IF OK

EXOR1:
MOV #336,-(R5) : WRONG RESULT IN TEMP1 OR WRONG SEQUENCE
INC -(R5)
HALT
ADD1:

CMP (R5), #116
BNE EADD1
INC (R5)
MOV #12700, 000142
MOV #TEMP2, R0
MOV #TEMP, R1
MOV #21421, TEMP2
MOV (R0), (R1)
ADD (R0), (R1)
JSR PC @#CC0
CMP #43042, TEMP
BEQ 1$
MOV #337, -(R5)
INC -(R5)

ADD INSTRUCTION FAILED
CLR (R0)
ADD R0, (R0) +
CMP -(RO), TEMP2
BEQ 2$
MOV #360, -(R5)
INC -(R5)

ADD INSTRUCTION FAILED IN MODE 2
MOV #21421, TEMP2
MOV (RO), (R1)
ASR (RO), (R1)
JR PC @#CC11
CMP #43042, TEMP
BEQ 3$
MOV #361, -(R5)
INC -(R5)

ADD INSTRUCTION FAILED
MOV #100000, TEMP2
MOV (RO), 0(R1)
ADD O(RO), (RT)
JMP PC @#CC7
MOV #21421, TEMP1
MOV (RO), (R1)
LOAD LOCATION TEMP1
JMP PC @#CC1
JMP 000000
MOV #TEMP1, 0(RU)
MOV #21421, (RT)
LOAD LOCATION TEMP
MOV R4, R6
LOAD LOCATION TEMP2
ADD @0(RG), (R1)
ADD SHOULD RESULT AS 0'S
MOV R0, R6
MOV #21421, -(SP)
MOV -(RO), 0(SP)
NEG (RO)
NEGATE THE CONTENTS OF TEMP1
TST (SP) +
CHECK THE STACK TO CONTAIN 0, ALSO
ADD, SHOULD=0'S
MOV #137777, TEMP2
ADD INSTRUCTION FAILED IN MODE 5
NEW INSTRUCTION IN THIS SECTION IS ADD

EADD1:
MOV #343,-(R5)
INC -(R5)
HALT
; WRONG RESULT AT TEMP OR WRONG SEQUENCE

; TEST: 117 NEW INSTRUCTION IN THIS SECTION IS SUB

SUB1:
CMP (R5),#117
BNE ESUB1
; IF IN WRONG SEQUENCE GO TO HLT AT THE END OF THE TEST
INC (R5)
MOV #21421,TEMP1
; LOAD ADDRESSES
MOV #21421,TEMP1
; LOAD LOCATIONS
SUB (R2),#3
; RESULT SHOULD =-43042
JSR PC,#65536
; CHECK FOR CC = 10
CMP #43042,TEMP1
; CHECK IT
BEQ 1$
; CONTINUE IF OK
MOV #346,-(R5)
INC -(R5)
HALT
; SUB INSTRUCTION FAILED
MOV #21421,TEMP1
; LOAD LOCATION
SUB (R2),#3
; RESULT SHOULD = 0
BEQ 2$
MOV #345,-(R5)
INC -(R5)
HALT
; SUB INSTRUCTION FAILED
MOV #1,TEMP1
; LOAD LOCATIONS
MOV #77777,TEMP1
; LOAD LOCATIONS
SUB (R3),#2
; RESULT SHOULD GIVE 100000 AND OVERFLOW
JSR PC,#65536
; CHECK FOR CC = 13
CMP #100000,TEMP1
; CHECK IT
BEO 3$
; CONTINUE IF OK
MOV #346,-(R5)
INC -(R5)
HALT
; SUB INSTRUCTION FAILED
MOV #1,(R2)
SUB (R3),#2
; CHECK FOR CC = 4
JSR PC,#65536
MOV #77777,TEMP1
; CHECK FOR CC = 4
SUB (R3),#4
; TEMP SHOULD BE 0
MOV #77777,TEMP1
INT - (R5)
NEW INSTRUCTION IN THIS SECTION IS SUB

HALT ; SUB INSTRUCTION FAILED OR SEQUENCE ERROR
**TEST: 120 NEW INSTRUCTION IN THIS SECTION IS SOB**

---

**SOB:**

```assembly
CMP (R5),#120
BNE ESOB
INC (R5)
MOV #10,.R0
CLR R1
INC R1
CMP R1,#10.
BLE 2S
MOV #350,-(R5)
INC -(R5)
HALT
SCC
SOB R0,1S
JSR PC,#$CC17
TST R0
REG. O = 0 ?
NO, FAILED
MOV #351,-(R5)
INC -(R5)
HALT
SOB INSTRUCTION FAILED
MOV #10,.R1
DIGIT GO THRU 10 TIMES?
CONTINUE IF OK
MOV #352,-(R5)
INC -(R5)
HALT
SOB INSTRUCTION FAILED
MOV #10,.R4
PLACE #10 IN R4
STAY HERE UNTILL R4 = 0
TST R4
BEQ PSONO
CONTINUE IF OK
MP553
MOV #353,-(R5)
INC -(R5)
HALT
SOB FAILED OR WRONG SEQUENCE
```

---

**TEST: 121 NEW INSTRUCTIONS IN THIS SECTION ARE MP5S & MP5Ps**

---

**PSWNO:**

```assembly
CMP (R5),#121
BNE EPSWNO
INC (R5)
MOV #TEMP,RO
PUT THE ADDRESS OF TEMP IN RO
MOV #TEMP1,R1
PUT THE ADDRESS OF TEMP1 IN R1
MOV #177777777777,(R1)
CLR (R0)
MP5S (R0)
PSW = 0
.MWORD 104400,...
JSR PC,#$CC0
CHECK FOR CF = 0
```
NEW INSTRUCTIONS IN THIS SECTION ARE MFPS & MFPS

E 8

NEW INSTRUCTIONS IN THIS SECTION ARE MFPS & MFPS

SEQ 0095

3538 016314
3538 016314 106711
3539 016316 004737 117150
3540 016322 022711 177490
3541 016326 001404
3543 016330 012765 000354
3544 016334 005245
3545 016336 000000
3546 016340 005011
3547 016342
3548 016342 106427
3549 016346 004737 017340
3550 016352
3551 016352 106767
3552 016356 006737 017256
3553 016362 022767 000357 061550
3554 016370 001404
3555 016372
3556 016372 012745 000355
3557 016376 005245
3558 016400 000000
3559

MFPS (R1)
WORD 106700.C
JSR PC @$9C4
CMP #177400,(R1)
BEO 1S
MOV #354,-(R5)
INC -(R5)
HALT
CLR (R1)
MFPS #377
WORD 106400.C
JSR PC @$CC17
MFPS TEMP1
WORD 106700.C
JSR PC @$CC11
CMP #357,TEMP1
BEQ BTWRD
EPSWNO:
MOV #355,-(R5)
INC -(R5)
HALT

; MOVE PSW TO TEMP1
; CHECK FOR CC = 4
; CHECK TEMP1 TO MAKE SURE THAT ONLY
; THE LOWER BYTE WAS AFFECTED BY MFPS
; MFPS OR MFPS INSTRUCTION FAILED
; SET PSW = 357 SINCE T BIT CAN NOT BE SET BY MFPS
; CHECK FOR CC = 17
; MOVE PSW TO TEMP1
; CHECK FOR CC = 17 CC BIT SHOULD NOT BE EFFECTED BY MFPS
; MFPS INSTRUCTION FAILED IN MODE 6
; OR SEQUENCE ERROR
TEST: 122 BYTE INSTRUCTIONS REQUIRING WORD INST. TO CHECK

; BTWRI:
; CMP (R5), #122
; BNE BTWRI
; INC (R5)
; CLR RO
; MOV #200, RO
; JSR PC, @S$CC11
; CMP #177600, RO
; BEQ 1$
; MOV #556, -(R5)
; INC -(R5)
; HALT
; SCC
; MOV #177777, RO
; JSR PC, @S$CC5
; TST RO
; BEQ 2$
; MOV #357, -(R5)
; INC -(R5)
; HALT
; MOV #17204, 000162
; MOV #577, (R4)
; MOV #12706, 000456
; MOV O(R4). (R6)+
; CMP #START, R6
; BEQ 3$
; MOV #360, -(R5)
; INC -(R5)
; HALT
; CMPB -(R6), #377
; BEQ 4$
; MOV #361, -(R5)
; INC -(R5)
; HALT
; CMP #START-2, R6
; BEQ 5$
; MOV #362, -(R5)
; INC -(R5)
; HALT
; MOV O(R4), TEMP
; TST (R6)+
; SCC
; JSR PC, @S$CC11
; CMP #177777, TEMP
; BEQ 6$
; R6 WAS NOT DECREMENTED
; SET THE LOWER BYTE OF LOCATION TEMP
; RESTORE STACK POINTER
; CHECK FOR CC=11
; CHECK TEMP FOR THE CORRECT VALUE
<table>
<thead>
<tr>
<th>Byte</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3616</td>
<td>MOV #363,-(R5)</td>
<td>MOVE IMMEDIATE DATA (363) TO REGISTER (R5)</td>
</tr>
<tr>
<td>3617</td>
<td>INC -(R5)</td>
<td>INCREASE REGISTER (R5)</td>
</tr>
<tr>
<td>3618</td>
<td>HALT</td>
<td>Halt Operation</td>
</tr>
<tr>
<td>3619</td>
<td>CLR TEMP</td>
<td>Clear Register TEMP</td>
</tr>
<tr>
<td>3620</td>
<td>CLC</td>
<td>Clear Carry Flag</td>
</tr>
<tr>
<td>3621</td>
<td>COMB TEMP+1</td>
<td>Combine TEMP and TEMP+1</td>
</tr>
<tr>
<td>3622</td>
<td>JSR PC @#CC11</td>
<td>Jump to Memory Location CC11</td>
</tr>
<tr>
<td>3623</td>
<td>CMP #177400,TEMP</td>
<td>Compare TEMP with Immediate Data (177400)</td>
</tr>
<tr>
<td>3624</td>
<td>BEQ NEXT</td>
<td>Branch if Equal to (R5)</td>
</tr>
<tr>
<td>3625</td>
<td>016660</td>
<td>016660</td>
</tr>
<tr>
<td>3626</td>
<td>MOV #364,-(R5)</td>
<td>MOVE IMMEDIATE DATA (364) TO REGISTER (R5)</td>
</tr>
<tr>
<td>3627</td>
<td>INC -(R5)</td>
<td>INCREASE REGISTER (R5)</td>
</tr>
<tr>
<td>3628</td>
<td>HALT</td>
<td>Halt Operation</td>
</tr>
</tbody>
</table>

Notes:
- TEMP FOULED UP
- WRITE 1'S IN THE HIGHER BYTE OF TEMP
- CHECK FOR CC=11
- WRONG VALUE IN TEMP OR WRONG SEQUENCE
; END OF PASS
;
***********

NEXT:
CMP (R5),#123  ; IF IN WRONG SEQUENCE GO TO HLT BELOW
BEQ 2$       
MOV #365,-(R5)
INC -(R5)     ; PROGRAM IS IN WRONG SEQUENCE
;
INC $PASS     ; ALLOW THE TYPE OUT OF END OF
CMPB $PASS,#1 ; PASS EVERY 377 PASSES
;
2$:           
BNE DOAGN    ; TYPE END OF PASS MESSAGE

DOAGN:       
CLR $TESTN    ; PREPARE TO START FROM TEST 0
RETUR:N:     
JMP START     ; START TEST OVER AT BEGINNING
;
-------------

;**********************************************************************

;**********************************************************************

;SBLTL POWER FAIL ROUTINE

PWRDN:       
MOV #PWRUP,#24 ; GO TO POWER UP ROUTINE AFTER THE POWER COMES BACK
HALT         

PWRUP:       
MOV #START,SP
MOV #PWRDN,#24

TYPE POWER
BR DOAGN
.SBITL TYPE ROUTINE

80, #ENVN

: HAS THE CONSOLE OUTPUTS BEEN SUPPRESSED?

: IF SO THEN GO TO 4$

: GET ADDRESS OF MESSAGE

: END OF MESSAGE:

: YES, GO WRAP IT UP

: READY FOR NEXT CHARACTER?

: NO, WAIT

: LOAD AND TYPE THE CHARACTER

: YES, GET THE NEXT CHARACTER

: ADJUST THE RETURN PC

: RETURN
ROUTINES TO CHECK CONDITION CODES

HALT

RTS  PC

;WRONG CC, IT SHOULD HAVE BEEN 17

01735E: CO00
01736E: CO09

01735E: CO00
01736E: CO09
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABASE</td>
<td>000000</td>
</tr>
<tr>
<td>ACDF7</td>
<td>339</td>
</tr>
<tr>
<td>ACDF8</td>
<td>339</td>
</tr>
<tr>
<td>ACDF10</td>
<td>339</td>
</tr>
<tr>
<td>ACDF10</td>
<td>339</td>
</tr>
<tr>
<td>ADCB0</td>
<td>354</td>
</tr>
<tr>
<td>ADCB1</td>
<td>1302#</td>
</tr>
<tr>
<td>ADCB2</td>
<td>319</td>
</tr>
<tr>
<td>ADCB3</td>
<td>319</td>
</tr>
<tr>
<td>ADCB4</td>
<td>319</td>
</tr>
<tr>
<td>ADCB5</td>
<td>319</td>
</tr>
<tr>
<td>ADCB6</td>
<td>319</td>
</tr>
<tr>
<td>ADCB7</td>
<td>319</td>
</tr>
<tr>
<td>ADCB8</td>
<td>319</td>
</tr>
<tr>
<td>ADCB9</td>
<td>319</td>
</tr>
<tr>
<td>ADD0</td>
<td>1825</td>
</tr>
<tr>
<td>ADD1</td>
<td>1827#</td>
</tr>
<tr>
<td>ADD2</td>
<td>3363</td>
</tr>
<tr>
<td>ADD3</td>
<td>3372#</td>
</tr>
<tr>
<td>ADD4</td>
<td>339</td>
</tr>
<tr>
<td>ADD5</td>
<td>339</td>
</tr>
<tr>
<td>ADD6</td>
<td>339</td>
</tr>
<tr>
<td>ADD7</td>
<td>339</td>
</tr>
<tr>
<td>ADD8</td>
<td>339</td>
</tr>
<tr>
<td>ADD9</td>
<td>339</td>
</tr>
<tr>
<td>ADDO</td>
<td>006524</td>
</tr>
<tr>
<td>ADD1</td>
<td>015414</td>
</tr>
<tr>
<td>ADEVCY</td>
<td>000000</td>
</tr>
<tr>
<td>ADEVM</td>
<td>000000</td>
</tr>
<tr>
<td>ADR1</td>
<td>388#</td>
</tr>
<tr>
<td>ADR2</td>
<td>390#</td>
</tr>
<tr>
<td>AENV</td>
<td>350</td>
</tr>
<tr>
<td>AENVM</td>
<td>351</td>
</tr>
<tr>
<td>AFATAL</td>
<td>000000</td>
</tr>
<tr>
<td>AMADR1</td>
<td>000000</td>
</tr>
<tr>
<td>AMADR2</td>
<td>000000</td>
</tr>
<tr>
<td>AMADR3</td>
<td>339</td>
</tr>
<tr>
<td>AMADR4</td>
<td>339</td>
</tr>
<tr>
<td>AMAPS1</td>
<td>339</td>
</tr>
<tr>
<td>AMAPS2</td>
<td>339</td>
</tr>
<tr>
<td>AMAPS3</td>
<td>339</td>
</tr>
<tr>
<td>AMAPS4</td>
<td>339</td>
</tr>
<tr>
<td>AMSGAD</td>
<td>36#</td>
</tr>
<tr>
<td>AMSLG</td>
<td>346</td>
</tr>
<tr>
<td>AMSVG1</td>
<td>346</td>
</tr>
<tr>
<td>AMYP1</td>
<td>339</td>
</tr>
<tr>
<td>AMYP2</td>
<td>339</td>
</tr>
<tr>
<td>AMYP3</td>
<td>339</td>
</tr>
<tr>
<td>AMYP4</td>
<td>339</td>
</tr>
<tr>
<td>APASS</td>
<td>339</td>
</tr>
<tr>
<td>APR10R</td>
<td>339</td>
</tr>
<tr>
<td>ASL0</td>
<td>1236</td>
</tr>
<tr>
<td>ASL01</td>
<td>1245#</td>
</tr>
<tr>
<td>ASL02</td>
<td>2640</td>
</tr>
<tr>
<td>ASL03</td>
<td>2652#</td>
</tr>
<tr>
<td>ASL04</td>
<td>1643</td>
</tr>
<tr>
<td>ASL05</td>
<td>1652#</td>
</tr>
<tr>
<td>ASL06</td>
<td>315#</td>
</tr>
<tr>
<td>ASR00</td>
<td>004616</td>
</tr>
<tr>
<td>USER SYMBOL</td>
<td>ADDRESS</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td>ASMB1</td>
<td>0122628</td>
</tr>
<tr>
<td>ASRO</td>
<td>0056123</td>
</tr>
<tr>
<td>ASR1</td>
<td>0144336</td>
</tr>
<tr>
<td>ASWREG</td>
<td>0000000</td>
</tr>
<tr>
<td>ATEST</td>
<td>0000000</td>
</tr>
<tr>
<td>AUNIT</td>
<td>0000000</td>
</tr>
<tr>
<td>AUSWR</td>
<td>0000000</td>
</tr>
<tr>
<td>AVECT1</td>
<td>0000000</td>
</tr>
<tr>
<td>AVECT2</td>
<td>0000000</td>
</tr>
<tr>
<td>BITBO</td>
<td>0030660</td>
</tr>
<tr>
<td>BICB1</td>
<td>0111223</td>
</tr>
<tr>
<td>BICO</td>
<td>0465300</td>
</tr>
<tr>
<td>BIT1</td>
<td>0132568</td>
</tr>
<tr>
<td>BRANCH</td>
<td>0011110</td>
</tr>
<tr>
<td>BTWRD</td>
<td>0164002</td>
</tr>
<tr>
<td>BIT6</td>
<td>0066000</td>
</tr>
<tr>
<td>CGO</td>
<td>0005328</td>
</tr>
<tr>
<td>CCO</td>
<td>0005768</td>
</tr>
<tr>
<td>CCO</td>
<td>0006464</td>
</tr>
<tr>
<td>CCO</td>
<td>0007144</td>
</tr>
<tr>
<td>CCO</td>
<td>0007648</td>
</tr>
<tr>
<td>CCO</td>
<td>0010768</td>
</tr>
<tr>
<td>CCO</td>
<td>0010328</td>
</tr>
<tr>
<td>CLN2</td>
<td>0002534</td>
</tr>
<tr>
<td>CMPBO</td>
<td>0027220</td>
</tr>
<tr>
<td>CMPB1</td>
<td>0102010</td>
</tr>
<tr>
<td>CMP2</td>
<td>0465328</td>
</tr>
<tr>
<td>CMP1</td>
<td>0301128</td>
</tr>
<tr>
<td>COME0</td>
<td>0333328</td>
</tr>
<tr>
<td>COME1</td>
<td>0115444</td>
</tr>
<tr>
<td>COMO</td>
<td>0061328</td>
</tr>
<tr>
<td>COM1</td>
<td>0137212</td>
</tr>
<tr>
<td>DOAGN</td>
<td>0167444</td>
</tr>
<tr>
<td>DUMMY</td>
<td>0001328</td>
</tr>
<tr>
<td>EADD0</td>
<td>0066664</td>
</tr>
<tr>
<td>EADD1</td>
<td>0157108</td>
</tr>
<tr>
<td>EASRBO</td>
<td>0011144</td>
</tr>
<tr>
<td>EASR1</td>
<td>0123708</td>
</tr>
<tr>
<td>EASR2</td>
<td>0057108</td>
</tr>
<tr>
<td>EASR3</td>
<td>0145448</td>
</tr>
<tr>
<td>EBICO</td>
<td>0047644</td>
</tr>
<tr>
<td>EBIC1</td>
<td>0133350</td>
</tr>
<tr>
<td>EBTWRD</td>
<td>0666664</td>
</tr>
<tr>
<td>ECMPB0</td>
<td>0030528</td>
</tr>
<tr>
<td>ECMPB1</td>
<td>0111112</td>
</tr>
<tr>
<td>ECMPB2</td>
<td>0466228</td>
</tr>
<tr>
<td>ECMP1</td>
<td>0132428</td>
</tr>
<tr>
<td>EINCBO</td>
<td>0115328</td>
</tr>
<tr>
<td>EMODE0</td>
<td>0073168</td>
</tr>
<tr>
<td>EMODE2</td>
<td>0074568</td>
</tr>
<tr>
<td>EMODE3</td>
<td>0074664</td>
</tr>
<tr>
<td>EMODE4</td>
<td>0101224</td>
</tr>
<tr>
<td>EMODE5</td>
<td>0103248</td>
</tr>
<tr>
<td>EMODE6</td>
<td>0105248</td>
</tr>
<tr>
<td>EMODE7</td>
<td>0106664</td>
</tr>
<tr>
<td>ENFCCO</td>
<td>0005416</td>
</tr>
<tr>
<td>Symbol</td>
<td>Value</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>ESOB</td>
<td>016250</td>
</tr>
<tr>
<td>ESUB1</td>
<td>016126</td>
</tr>
<tr>
<td>ESUB1B</td>
<td>006406</td>
</tr>
<tr>
<td>ESUB2A</td>
<td>015264</td>
</tr>
<tr>
<td>ESX10</td>
<td>063006</td>
</tr>
<tr>
<td>ESX11</td>
<td>015156</td>
</tr>
<tr>
<td>ESX1B1</td>
<td>011010</td>
</tr>
<tr>
<td>EXOR0</td>
<td>065514</td>
</tr>
<tr>
<td>EXOR1</td>
<td>015404</td>
</tr>
<tr>
<td>GET42</td>
<td>064800</td>
</tr>
<tr>
<td>INCB0</td>
<td>065200</td>
</tr>
<tr>
<td>INCB1</td>
<td>011350</td>
</tr>
<tr>
<td>INCQ</td>
<td>004774</td>
</tr>
<tr>
<td>INC1</td>
<td>013540</td>
</tr>
<tr>
<td>JMP1</td>
<td>001202</td>
</tr>
<tr>
<td>JMP2</td>
<td>001306</td>
</tr>
<tr>
<td>JMR1</td>
<td>001142</td>
</tr>
<tr>
<td>JMR2</td>
<td>001517</td>
</tr>
<tr>
<td>JMR5</td>
<td>001610</td>
</tr>
<tr>
<td>JMR6</td>
<td>001722</td>
</tr>
<tr>
<td>JMR7</td>
<td>002024</td>
</tr>
<tr>
<td>JSRM</td>
<td>002264</td>
</tr>
<tr>
<td>JSRST1</td>
<td>002122</td>
</tr>
<tr>
<td>MARK2</td>
<td>000150</td>
</tr>
<tr>
<td>MODE0</td>
<td>007140</td>
</tr>
<tr>
<td>MODE1</td>
<td>007226</td>
</tr>
<tr>
<td>MODE2</td>
<td>007326</td>
</tr>
<tr>
<td>MODE3</td>
<td>007464</td>
</tr>
<tr>
<td>MODE4</td>
<td>006566</td>
</tr>
<tr>
<td>MODF1</td>
<td>010134</td>
</tr>
<tr>
<td>MODF6</td>
<td>010364</td>
</tr>
<tr>
<td>MODF7</td>
<td>010534</td>
</tr>
<tr>
<td>NEBI1</td>
<td>000542</td>
</tr>
<tr>
<td>NEBO</td>
<td>003452</td>
</tr>
<tr>
<td>NEGB1</td>
<td>016764</td>
</tr>
<tr>
<td>NEG0</td>
<td>05246</td>
</tr>
<tr>
<td>NEG1</td>
<td>014036</td>
</tr>
<tr>
<td>NEXT</td>
<td>016670</td>
</tr>
<tr>
<td>NOBIT</td>
<td>000466</td>
</tr>
<tr>
<td>NOP1</td>
<td>000260</td>
</tr>
<tr>
<td>NOTCC</td>
<td>001042</td>
</tr>
<tr>
<td>POWER</td>
<td>000166</td>
</tr>
<tr>
<td>PSW</td>
<td>007040</td>
</tr>
<tr>
<td>PSWNO</td>
<td>016260</td>
</tr>
<tr>
<td>PURDN</td>
<td>016764</td>
</tr>
<tr>
<td>PURUP</td>
<td>002536</td>
</tr>
<tr>
<td>REGS</td>
<td>016750</td>
</tr>
<tr>
<td>RETURN</td>
<td>36553</td>
</tr>
</tbody>
</table>

**Cross Reference Table -- User Symbols**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>3169</td>
<td>3172#</td>
</tr>
<tr>
<td>3183</td>
<td>3186#</td>
</tr>
<tr>
<td>3196</td>
<td>3199#</td>
</tr>
<tr>
<td>3212</td>
<td>3215#</td>
</tr>
<tr>
<td>3221</td>
<td>3224#</td>
</tr>
<tr>
<td>3235</td>
<td>3238#</td>
</tr>
<tr>
<td>3246</td>
<td>3249#</td>
</tr>
<tr>
<td>3256</td>
<td>3259#</td>
</tr>
<tr>
<td>3267</td>
<td>3270#</td>
</tr>
<tr>
<td>3293</td>
<td>3296#</td>
</tr>
<tr>
<td>3302</td>
<td>3305#</td>
</tr>
<tr>
<td>3312</td>
<td>3315#</td>
</tr>
<tr>
<td>3321</td>
<td>3324#</td>
</tr>
<tr>
<td>3332</td>
<td>3335#</td>
</tr>
<tr>
<td>3342</td>
<td>3345#</td>
</tr>
<tr>
<td>3352</td>
<td>3355#</td>
</tr>
<tr>
<td>3362</td>
<td>3365#</td>
</tr>
<tr>
<td>3375</td>
<td>3378#</td>
</tr>
<tr>
<td>3384</td>
<td>3387#</td>
</tr>
<tr>
<td>3391</td>
<td>3394#</td>
</tr>
<tr>
<td>3400</td>
<td>3403#</td>
</tr>
<tr>
<td>3412</td>
<td>3415#</td>
</tr>
<tr>
<td>3424</td>
<td>3427#</td>
</tr>
<tr>
<td>3434</td>
<td>3437#</td>
</tr>
<tr>
<td>3478</td>
<td>3481#</td>
</tr>
<tr>
<td>3489</td>
<td>3492#</td>
</tr>
<tr>
<td>3502</td>
<td>3505#</td>
</tr>
<tr>
<td>3510#</td>
<td>3514#</td>
</tr>
<tr>
<td>3531</td>
<td>3534#</td>
</tr>
<tr>
<td>3556</td>
<td>3559#</td>
</tr>
<tr>
<td>3575</td>
<td>3578#</td>
</tr>
<tr>
<td>3595</td>
<td>3600#</td>
</tr>
<tr>
<td>3606</td>
<td>3609#</td>
</tr>
<tr>
<td>3616</td>
<td>3619#</td>
</tr>
<tr>
<td>3626</td>
<td>3629#</td>
</tr>
<tr>
<td>3660</td>
<td>3663#</td>
</tr>
<tr>
<td>3732</td>
<td>3735#</td>
</tr>
<tr>
<td>3775</td>
<td>3781#</td>
</tr>
<tr>
<td>3784</td>
<td>3790#</td>
</tr>
<tr>
<td>3793</td>
<td>3796#</td>
</tr>
<tr>
<td>3800</td>
<td>3803#</td>
</tr>
<tr>
<td>3847</td>
<td>3851#</td>
</tr>
<tr>
<td>3441</td>
<td>3477#</td>
</tr>
<tr>
<td>1827</td>
<td>1846#</td>
</tr>
<tr>
<td>3311</td>
<td>3333#</td>
</tr>
<tr>
<td>1797</td>
<td>1814#</td>
</tr>
<tr>
<td>3283</td>
<td>3301#</td>
</tr>
<tr>
<td>2347</td>
<td>2370#</td>
</tr>
<tr>
<td>1856</td>
<td>1876#</td>
</tr>
<tr>
<td>3344</td>
<td>3364#</td>
</tr>
<tr>
<td>3648#</td>
<td>3684#</td>
</tr>
<tr>
<td>1092#</td>
<td></td>
</tr>
<tr>
<td>2470#</td>
<td></td>
</tr>
<tr>
<td>1486</td>
<td>1498#</td>
</tr>
<tr>
<td>2953</td>
<td>2962#</td>
</tr>
<tr>
<td>645</td>
<td>650#</td>
</tr>
<tr>
<td>674</td>
<td>688#</td>
</tr>
<tr>
<td>710</td>
<td>717#</td>
</tr>
<tr>
<td>729</td>
<td>740#</td>
</tr>
<tr>
<td>760</td>
<td>766#</td>
</tr>
<tr>
<td>784</td>
<td>795#</td>
</tr>
<tr>
<td>814</td>
<td>820#</td>
</tr>
<tr>
<td>855</td>
<td>884#</td>
</tr>
<tr>
<td>883#</td>
<td></td>
</tr>
<tr>
<td>402#</td>
<td>887</td>
</tr>
<tr>
<td>1994</td>
<td>2008#</td>
</tr>
<tr>
<td>2024</td>
<td>2029#</td>
</tr>
<tr>
<td>2045</td>
<td>2058#</td>
</tr>
<tr>
<td>2090</td>
<td>2100#</td>
</tr>
<tr>
<td>2132</td>
<td>214#</td>
</tr>
<tr>
<td>2204</td>
<td>2214#</td>
</tr>
<tr>
<td>2262</td>
<td>2274#</td>
</tr>
<tr>
<td>2298</td>
<td>2308#</td>
</tr>
<tr>
<td>468</td>
<td>475#</td>
</tr>
<tr>
<td>1164#</td>
<td></td>
</tr>
<tr>
<td>2564#</td>
<td></td>
</tr>
<tr>
<td>1570#</td>
<td></td>
</tr>
<tr>
<td>304#</td>
<td></td>
</tr>
<tr>
<td>3624</td>
<td>3637#</td>
</tr>
<tr>
<td>437</td>
<td>445#</td>
</tr>
<tr>
<td>331#</td>
<td>455</td>
</tr>
<tr>
<td>587</td>
<td>596#</td>
</tr>
<tr>
<td>406#</td>
<td>3668</td>
</tr>
<tr>
<td>1970#</td>
<td></td>
</tr>
<tr>
<td>3513</td>
<td>3526#</td>
</tr>
<tr>
<td>423</td>
<td>3663#</td>
</tr>
<tr>
<td>3663</td>
<td>3666#</td>
</tr>
<tr>
<td>924</td>
<td>960#</td>
</tr>
<tr>
<td>3655#</td>
<td></td>
</tr>
</tbody>
</table>

**Symbol Definitions**

- `#` denotes a symbol with a specific meaning in the context of the table.
- Others represent standard symbols without additional notes.
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROLBO</td>
<td>003542</td>
</tr>
<tr>
<td>ROLBI</td>
<td>071772</td>
</tr>
<tr>
<td>ROLI</td>
<td>005736</td>
</tr>
<tr>
<td>ROL1</td>
<td>014142</td>
</tr>
<tr>
<td>RORBO</td>
<td>003634</td>
</tr>
<tr>
<td>RORBI</td>
<td>012036</td>
</tr>
<tr>
<td>ROR1</td>
<td>005430</td>
</tr>
<tr>
<td>ROR2</td>
<td>014244</td>
</tr>
<tr>
<td>SEVCB</td>
<td>004256</td>
</tr>
<tr>
<td>SB2B</td>
<td>025358</td>
</tr>
<tr>
<td>SBCO</td>
<td>006058</td>
</tr>
<tr>
<td>SBC1</td>
<td>014712</td>
</tr>
<tr>
<td>SENVC</td>
<td>000273</td>
</tr>
<tr>
<td>SELVC</td>
<td>000263</td>
</tr>
<tr>
<td>SUEB</td>
<td>016156</td>
</tr>
<tr>
<td>SUB1</td>
<td>000450</td>
</tr>
<tr>
<td>SUBO</td>
<td>006674</td>
</tr>
<tr>
<td>SUB2</td>
<td>015720</td>
</tr>
<tr>
<td>SWABO</td>
<td>006316</td>
</tr>
<tr>
<td>SWAB1</td>
<td>015166</td>
</tr>
<tr>
<td>SXT0</td>
<td>006230</td>
</tr>
<tr>
<td>SXT1</td>
<td>015072</td>
</tr>
<tr>
<td>TEMP</td>
<td>000136</td>
</tr>
<tr>
<td>TEP1</td>
<td>000140</td>
</tr>
<tr>
<td>TEM2</td>
<td>000142</td>
</tr>
<tr>
<td>TPB</td>
<td>000146</td>
</tr>
<tr>
<td>TPS</td>
<td>000144</td>
</tr>
<tr>
<td>TSBO</td>
<td>002644</td>
</tr>
<tr>
<td>TSIB</td>
<td>010676</td>
</tr>
<tr>
<td>TSTO</td>
<td>004436</td>
</tr>
<tr>
<td>TST1</td>
<td>012270</td>
</tr>
<tr>
<td>TYPE</td>
<td>017004</td>
</tr>
<tr>
<td>VBIT</td>
<td>006610</td>
</tr>
<tr>
<td>XORO</td>
<td>006416</td>
</tr>
<tr>
<td>XOR1</td>
<td>015274</td>
</tr>
<tr>
<td>YESC</td>
<td>000774</td>
</tr>
<tr>
<td>YSEP</td>
<td>000724</td>
</tr>
<tr>
<td>SAPTHD</td>
<td>000126</td>
</tr>
<tr>
<td>SCCO</td>
<td>017046</td>
</tr>
<tr>
<td>SCC1</td>
<td>017066</td>
</tr>
<tr>
<td>SCC2</td>
<td>017236</td>
</tr>
</tbody>
</table>

Note: The table contains numerical values, possibly representing codes or identifiers, but the specific context or meaning is not clear from the image alone.
<p>| $\text{STSM}$ | 000132 | 381# |
| $\text{SUNIT}$ | 000110 | 366# |
| $\text{SUNITM}$ | 000136 | 383# |
| $\text{SUSWR}$ | 000122 | 353# |
| $\text{=}$ | 017562 | |
| $\text{TYPE}$ | 000004 | 335# |
| $\text{BX}$ | 000126 | 368# |
| $\text{=}$ | 016352 | 368# |
| $\text{=}$ | 016356 | 3549 |
| $\text{=}$ | 000067 | 1977# |
| $\text{=}$ | 0988# |
| $\text{=}$ | 351# |
| $\text{=}$ | 352# |
| $\text{=}$ | 353# |
| $\text{=}$ | 354# |
| $\text{=}$ | 355# |
| $\text{=}$ | 356# |
| $\text{=}$ | 357# |
| $\text{=}$ | 358# |
| $\text{=}$ | 359# |
| $\text{=}$ | 360# |
| $\text{=}$ | 361# |
| $\text{=}$ | 362# |
| $\text{=}$ | 363# |
| $\text{=}$ | 364# |
| $\text{=}$ | 365# |
| $\text{=}$ | 366# |
| $\text{=}$ | 367# |</p>
<table>
<thead>
<tr>
<th>SETPRI</th>
<th>1#</th>
<th>SETUP</th>
<th>1#</th>
<th>SKIP</th>
<th>1#</th>
<th>SLASH</th>
<th>1#</th>
<th>STARS</th>
<th>1#</th>
</tr>
</thead>
<tbody>
<tr>
<td>519</td>
<td>521</td>
<td>542</td>
<td>544</td>
<td>569</td>
<td>571</td>
<td>593</td>
<td>594</td>
<td>618</td>
<td>620</td>
</tr>
<tr>
<td>738</td>
<td>791</td>
<td>793</td>
<td>839</td>
<td>841</td>
<td>956</td>
<td>958</td>
<td>990</td>
<td>1013</td>
<td>1015</td>
</tr>
<tr>
<td>1125</td>
<td>127</td>
<td>127</td>
<td>160</td>
<td>162</td>
<td>185</td>
<td>187</td>
<td>194</td>
<td>204</td>
<td>206</td>
</tr>
<tr>
<td>1341</td>
<td>139</td>
<td>139</td>
<td>174</td>
<td>176</td>
<td>174</td>
<td>174</td>
<td>207</td>
<td>1746</td>
<td>1748</td>
</tr>
<tr>
<td>1621</td>
<td>165</td>
<td>165</td>
<td>165</td>
<td>165</td>
<td>165</td>
<td>165</td>
<td>1531</td>
<td>1533</td>
<td>1533</td>
</tr>
<tr>
<td>1852</td>
<td>1883</td>
<td>1883</td>
<td>1925</td>
<td>1927</td>
<td>1966</td>
<td>1968</td>
<td>2004</td>
<td>2054</td>
<td>2096</td>
</tr>
<tr>
<td>22010</td>
<td>2212</td>
<td>2270</td>
<td>2273</td>
<td>2275</td>
<td>1966</td>
<td>2004</td>
<td>2006</td>
<td>2054</td>
<td>2096</td>
</tr>
<tr>
<td>2525</td>
<td>2560</td>
<td>2562</td>
<td>2589</td>
<td>2591</td>
<td>2617</td>
<td>2619</td>
<td>2648</td>
<td>2650</td>
<td>2674</td>
</tr>
<tr>
<td>2804</td>
<td>2806</td>
<td>2831</td>
<td>2833</td>
<td>2893</td>
<td>2895</td>
<td>2898</td>
<td>2908</td>
<td>3004</td>
<td>3038</td>
</tr>
<tr>
<td>3015</td>
<td>3129</td>
<td>3195</td>
<td>3153</td>
<td>3155</td>
<td>3189</td>
<td>3191</td>
<td>3258</td>
<td>3230</td>
<td>3277</td>
</tr>
<tr>
<td>3368</td>
<td>3370</td>
<td>3435</td>
<td>3437</td>
<td>3481</td>
<td>3483</td>
<td>3521</td>
<td>3523</td>
<td>3560</td>
<td>3562</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SWRSH</th>
<th>1#</th>
<th>TYPBIN</th>
<th>1#</th>
<th>TYPDEC</th>
<th>1#</th>
<th>TYPNAM</th>
<th>1#</th>
<th>TYPNLM</th>
<th>1#</th>
<th>TYPDCS</th>
<th>1#</th>
<th>TYPDCT</th>
<th>1#</th>
<th>TYPDTXT</th>
<th>1#</th>
</tr>
</thead>
<tbody>
<tr>
<td>3015</td>
<td>3017</td>
<td>660</td>
<td>696</td>
<td>859</td>
<td>894</td>
<td>519</td>
<td>542</td>
<td>569</td>
<td>592</td>
<td>618</td>
<td>646</td>
<td>682</td>
<td>736</td>
<td>791</td>
<td>839</td>
</tr>
<tr>
<td>990</td>
<td>1013</td>
<td>1053</td>
<td>1088</td>
<td>1125</td>
<td>1160</td>
<td>1185</td>
<td>1214</td>
<td>1241</td>
<td>1267</td>
<td>1298</td>
<td>1339</td>
<td>1323</td>
<td>1424</td>
<td>1451</td>
<td></td>
</tr>
<tr>
<td>1497</td>
<td>1531</td>
<td>1566</td>
<td>1591</td>
<td>1621</td>
<td>1648</td>
<td>1674</td>
<td>1705</td>
<td>1746</td>
<td>1791</td>
<td>1821</td>
<td>1850</td>
<td>1883</td>
<td>1929</td>
<td>1966</td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>2054</td>
<td>2096</td>
<td>2140</td>
<td>2210</td>
<td>2270</td>
<td>2304</td>
<td>2341</td>
<td>2377</td>
<td>2410</td>
<td>2466</td>
<td>2528</td>
<td>2560</td>
<td>2617</td>
<td>2675</td>
<td></td>
</tr>
<tr>
<td>3268</td>
<td>3277</td>
<td>3305</td>
<td>3338</td>
<td>3368</td>
<td>3435</td>
<td>3481</td>
<td>3521</td>
<td>3560</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SESCA</th>
<th>1#</th>
<th>SNEET</th>
<th>1#</th>
<th>$SKIP</th>
<th>1#</th>
<th>$EQUT</th>
<th>1#</th>
<th>$KT11</th>
<th>1#</th>
<th>$SETUP</th>
<th>1#</th>
<th>$SWRH</th>
<th>1#</th>
<th>$SACT</th>
<th>1#</th>
<th>$SAPTB</th>
<th>1#</th>
<th>$SAPTH</th>
<th>1#</th>
<th>$SAPTY</th>
<th>1#</th>
<th>$SASTA</th>
<th>1#</th>
<th>$SCATC</th>
<th>1#</th>
</tr>
</thead>
<tbody>
<tr>
<td>272#</td>
<td>272#</td>
<td>296</td>
<td></td>
<td>575</td>
<td>536</td>
<td>272#</td>
<td>272#</td>
<td>296</td>
<td></td>
<td>575</td>
<td>536</td>
<td>272#</td>
<td>272#</td>
<td>296</td>
<td></td>
<td>575</td>
<td>536</td>
<td>272#</td>
<td>272#</td>
<td>296</td>
<td></td>
<td>575</td>
<td>536</td>
<td>272#</td>
<td>272#</td>
</tr>
</tbody>
</table>

<p>| CROSS REFERENCE TABLE -- MACRO NAMES | SEU U11 |</p>
<table>
<thead>
<tr>
<th>Macro Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MTA</td>
<td></td>
</tr>
<tr>
<td>$DB2</td>
<td></td>
</tr>
<tr>
<td>$DB2C</td>
<td></td>
</tr>
<tr>
<td>$DV</td>
<td></td>
</tr>
<tr>
<td>$EOP</td>
<td></td>
</tr>
<tr>
<td>$ERRO</td>
<td></td>
</tr>
<tr>
<td>$ERRT</td>
<td></td>
</tr>
<tr>
<td>$MULT</td>
<td></td>
</tr>
<tr>
<td>$POWE</td>
<td></td>
</tr>
<tr>
<td>$RND</td>
<td></td>
</tr>
<tr>
<td>$RODE</td>
<td></td>
</tr>
<tr>
<td>$RDCC</td>
<td></td>
</tr>
<tr>
<td>$RFA</td>
<td></td>
</tr>
<tr>
<td>$R2A</td>
<td></td>
</tr>
<tr>
<td>$SAVE</td>
<td></td>
</tr>
<tr>
<td>$SB2D</td>
<td></td>
</tr>
<tr>
<td>$SB2O</td>
<td></td>
</tr>
<tr>
<td>$SCOP</td>
<td></td>
</tr>
<tr>
<td>$SIZE</td>
<td></td>
</tr>
<tr>
<td>$SUPR</td>
<td></td>
</tr>
<tr>
<td>$STRP</td>
<td></td>
</tr>
<tr>
<td>$TYPB</td>
<td></td>
</tr>
<tr>
<td>$TYPD</td>
<td></td>
</tr>
<tr>
<td>$TYPE</td>
<td></td>
</tr>
<tr>
<td>$TYPQ</td>
<td></td>
</tr>
<tr>
<td>$40CA</td>
<td></td>
</tr>
<tr>
<td>$1170</td>
<td></td>
</tr>
</tbody>
</table>

ABS. 017362 000

ERRORS DETECTED: 0

CVKAAC.BIN CVKAAC.LST CRF SOL/NL TGC=CVKAAC.SML CVKAAC.P11
RUN-TIME: 13 18 1 SECONDS
RUN-TIME RATIO: 103/33=3.0
ORE USED: 32k (63 PAGES)