

Testing

1. MOVL, MOVLZ, MOVLS, and MOVH instructions work as expected

Purpose: Checks for successful execution of MOVL, MOVLZ, MOVLS, and MOVH instructions.

Configuration: The following program is loaded into the emulator. It performs MOVL, MOVLS, MOVLZ, and MOVH operations.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: MOVLtoMOVH.txt
Time of assembly: Sun 23 Jun 2024 23:24:16
1          org #1000
2          MAIN
3      1000    64C8      movl    #99,R0          ;set low bytes
4      1002    7FF8      movh    #FF00,R0       ;set high bytes
5      1004    6C40      movlz   #88,R0         ;set low bytes without changing high bytes
6      1006    73B8      movls   #77,R0         ;set low bytes and clear high bytes
7          DONE
8      1008    3FFF      bra      DONE          ; loop
9          end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1002, MOVL is performed:</p> <ul style="list-style-type: none">DST = 0xEE00DST.lowbyte <- 0x99 <p>This will produce:</p> <ul style="list-style-type: none">DST = 0xFF99	<pre>R0: EE00 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1002 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1002: MOVL BYTE: 99 DST: R0 R0: EE99 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>
<p>At address 1004, MOVH is performed:</p> <ul style="list-style-type: none">DST = 0xEE99DST.highbyte <- 0xAA <p>This will produce:</p> <ul style="list-style-type: none">DST = 0xAA99	<pre>R0: EE99 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1004: MOVH BYTE: AA DST: R0 R0: AA99 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>

<p>At address 1006, MOVLZ is performed:</p> <ul style="list-style-type: none"> DST = 0xAA99 DST.highbyte <- 0x00 DST.lowbyte <- 0x88 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0xFF88 	<pre> R0: AA99 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1006: MOVLZ BYTE: 88 DST: R0 R0: 0088 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>
<p>At memory address 1008, MOVLS is performed:</p> <ul style="list-style-type: none"> DST = 0xFF88 DST.highbyte <- 0xFF DST.lowbyte <- 0x77 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0xFF77 	<pre> 1006: MOVLZ BYTE: 88 DST: R0 R0: 0088 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1008: MOVLS BYTE: 77 DST: R0 R0: FF77 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>

The MOVL, MOVLZ, MOVLS, and MOVH operations perform as expected. They do not set or clear any flags.

Pass/Fail: PASS

2. ADD and ADDC instructions work as expected

Purpose: Checks for successful execution of ADD and ADDC instructions.

Configuration: The following program is loaded into the emulator. ADD and ADDC are executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: ADDtoADDC.txt
Time of assembly: Mon 24 Jun 2024 01:36:03
1          org #1000
2          MAIN
3      1000      7C80      movh      #9000,R0
4      1002      7C81      movh      #9000,R1
5      1004      4008      add       R1,R0      ;producing carry and overflow flag using word addition
6      1006      4180      addc      $0,R0      ;adding 0 to a register with carry of 1, clearing flags
7      1008      6008      movl      $1,R0
8      100A      67F9      movl      $-1,R1
9      100C      4048      add.b     R1,R0      ;producing a zero flag using byte addition
10     100E      4048      add.b     R1,R0      ;producing a negative flag
11          DONE
12     1010      3FFF      bra       DONE      ; loop
13          end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1004, ADD is performed using values of size word and a register source:</p> <ul style="list-style-type: none">DST <- 0x9000 + 0x9000 <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x2000SET overflow flagCLEAR negative flagCLEAR zero flagSET carry flag	<pre>R0: 9000 R1: 9000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1004: ADD RC: 0 WB: 0 SRC: R1 DST: R0 R0: 2000 R1: 9000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 1 slp: 0 n: 0 z: 0 c: 1</pre>

<p>At address 1006, ADDC is performed using values of size word with a constant value and a constant source:</p> <ul style="list-style-type: none"> DST <- 0x2000 + \$0 + CARRY <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x2001 CLEAR overflow flag CLEAR negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: 2000 R1: 9000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 1 slp: 0 n: 0 z: 0 c: 1 1006: ADDC RC: 1 WB: 0 CON: 0 DST: R0 R0: 2001 R1: 9000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>
<p>At memory address 100A, ADD is performed using values of size byte and a register source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 0x01 + 0xFF <p>This will produce:</p> <ul style="list-style-type: none"> DST.lowbyte = 0x00 CLEAR overflow flag CLEAR negative flag SET zero flag SET carry flag 	<pre> 100A: MOVL BYTE: 00FF DST: R1 R0: 2001 R1: 90FF R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 100C: ADD RC: 0 WB: 1 SRC: R0 DST: R1 R0: 2001 R1: 9000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100E prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1 </pre>
<p>At memory address 100E, ADD is performed using values of size byte and a register source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 0x00 + 0xFF <p>This will produce:</p> <ul style="list-style-type: none"> DST.lowbyte = 0xFF CLEAR overflow flag SET negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: 2000 R1: 90FF R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100E prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1 100E: ADD RC: 0 WB: 1 SRC: R1 DST: R0 R0: 20FF R1: 90FF R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1010 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 </pre>

The ADD and ADDC instructions perform as expected. They produce the correct results when adding word or byte, and when there is a carry for ADDC. They also set the flags as expected.

Pass/Fail: PASS

3. SUB and SUBC instructions work as expected

Purpose: Checks for successful execution of SUB and SUBC

Configuration: The following program is loaded into the emulator. SUB and SUBC are executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: SUBtoSUBC.txt
Time of assembly: Sun 23 Jun 2024 23:17:12
1
2                               org #1000
3                               MAIN
4 1000 7FF8 movh #FF00,R0
5 1002 7FF9 movh #FF00,R1
6 1004 4208 sub R1,R0 ;producing zero and carry flags using word subtraction
7 1006 4380 subc $0,R0 ;adding constant 0 to a register with carry of 1, producing a zero and negative flag
8 1008 6180 movl #30,R0
9 100A 6401 movl #80,R1
10 100C 4248 sub.b R1,R0 ;producing an overflow and negative flag using byte subtraction
11 100E 3FFF bra DONE ; loop
12 end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1004, SUB is performed using values of size word and register source:</p> <ul style="list-style-type: none">DST <- 0x8000 - 0x8000 <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x0000CLEAR overflow flagCLEAR negative flagSET zero flagSET carry flag	<pre>R0: FF00 R1: FF00 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1004: SUB RC: 0 WB: 0 SRC: R1 DST: R0 R0: 0000 R1: FF00 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1</pre>
<p>At address 1006, SUBC is performed using values of size word and a constant source:</p> <ul style="list-style-type: none">DST <- 0x2000 - \$0 + CARRY <p>This will produce:</p> <ul style="list-style-type: none">RESULT = 0x2001CLEAR overflow flagCLEAR negative flagSET zero flagSET carry flag	<pre>R0: 0000 R1: FF00 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1 1006: SUBC RC: 1 WB: 0 CON: 0 DST: R0 R0: 0000 R1: FF00 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1</pre>

At address 100C, SUB is performed using values of size byte and a register source:

- DST.lowbyte <- 0x30 – 0x80

This will produce:

- DST.lowbyte = 0x00B0
- SET overflow flag
- SET negative flag
- CLEAR zero flag
- CLEAR carry flag

```

R0: 0030
R1: FF80
R2: 0000
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100C
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 1

100C: SUB    RC: 0 WB: 1 SRC: R1 DST: R0
R0: 00B0
R1: FF80
R2: 0000
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100E
prev: 3 flt: 0 curr: 7 v: 1 slp: 0 n: 1 z: 0 c: 0

```

The SUB and SUBC instructions perform as expected. They produce the correct results when subtracting word or byte, and when there is a carry for SUBC. They also set the flags as expected.

Pass/Fail: PASS

4. DADD instruction works as expected

Purpose: Checks for successful execution of DADD instructions

Configuration: The following program is loaded into the emulator. DADD is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```

X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: DADD.txt
Time of assembly: Mon 24 Jun 2024 01:19:53
 1                                org #1000
 2                                MAIN
 3      1000      7890            movh  #1200,R0
 4      1002      61A0            movl  #34,R0
 5      1004      7C39            movh  #8700,R1
 6      1006      6331            movl  #66,R1
 7      1008      4401            DADD  R0,R1      ;DADD with word using a register, producing a carry flag
 8      100A      44D8            DADD.b $4,R0     ;DADD with byte using a constant
 9                                DONE
10      100C      3FFF            bra   DONE      ; loop
11                                end MAIN
Successful completion of assembly - 1P

```

Expected Results	Actual Results
<p>At address 1008, DADD is performed using values of size word with a register source:</p> <ul style="list-style-type: none"> DST <- 1234 + 8766 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0000 CLEAR overflow flag CLEAR negative flag CLEAR zero flag SET carry flag 	<pre> R0: 1234 R1: 8766 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1008: DADD RC: 0 WB: 0 SRC: R0 DST: R1 R0: 1234 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 1 </pre>
<p>At address 100A, DADD is performed using values of size word with a constant source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 34 + \$4 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x1238 CLEAR overflow flag CLEAR negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: 1234 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 1 100A: DADD RC: 1 WB: 1 CON: 4 DST: R0 R0: 1238 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>

The DADD instruction works as expected. It can perform binary coded decimal addition with both sources and words, as well as with register and constant sources. It also sets the flags as expected.

Pass/Fail: PASS

5. CMP instruction works as expected

Purpose: Checks for successful execution of CMP

Configuration: The following program is loaded into the emulator. CMP is executed with byte and word, register and constant sources, and related flags each set in at least one operation. Tests are conducted for when DST and SRC have a difference of 0, more than 0, and less than 0.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: CMP.txt
Time of assembly: Mon 24 Jun 2024 02:39:09
1      org #1000
2
3      MAIN
4      1000 6042      MOVL #08,R2
5      1002 6041      MOVL #08,R1
6      1004 454A      CMP.b R1,R2      ;0x08-0x08
7      1006 7841      MOVH #0800,R1
8      1008 6001      MOVL #00,R1
9      100A 783A      MOVH #0700,R2      ;0x0700-0x0800
10     100C 450A      CMP R1,R2
11     100E 604B      MOVL #09,R3
12     1010 458B      CMP $1,R3      ;0x0009-$1
13     1012 3FFF      DONE bra DONE      ; loop
14     end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1004, CMP is performed using values of size byte and register source:</p> <ul style="list-style-type: none">0x08 - 0x08 <p>This will produce:</p> <ul style="list-style-type: none">CLEAR overflow flagCLEAR negative flagSET zero flagSET carry flag	<pre>R0: 0000 R1: 0008 R2: 0008 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1004: CMP RC: 0 WB: 1 SRC: R1 DST: R2 R0: 0000 R1: 0008 R2: 0008 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1</pre>
<p>At address 100C, CMP is performed using values of size word and a register source:</p> <ul style="list-style-type: none">0x0708 – 0x0800 <p>This will produce:</p> <ul style="list-style-type: none">CLEAR overflow flagSET negative flagCLEAR zero flagCLEAR carry flag	<pre>R0: 0000 R1: 0800 R2: 0708 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1 100C: CMP RC: 0 WB: 0 SRC: R1 DST: R2 R0: 0000 R1: 0800 R2: 0708 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100E prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0</pre>

At address 10010, CMP is performed using values of size word and a constant source:

- 0x0009 - \$1

This will produce:

- CLEAR overflow flag
- CLEAR negative flag
- CLEAR zero flag
- SET carry flag

```
R0: 0000
R1: 0800
R2: 0708
R3: 0009
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 1010
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0

1010: CMP    RC: 1 WB: 0 CON: 1 DST: R3
R0: 0000
R1: 0800
R2: 0708
R3: 0009
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 1012
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 1
```

The CMP instructions perform as expected. They also set the flags as expected.

Pass/Fail: PASS

6. XOR instruction works as expected

Purpose: Checks for successful execution of XOR instruction.

Configuration: The following program is loaded into the emulator. XOR is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: XOR.txt
Time of assembly: Mon 24 Jun 2024 04:23:39
 1                                org #1000
 2                                MAIN
 3      1000      784A            MOVH    #0900,R2
 4      1002      6042            MOVL    #08,R2
 5      1004      6041            MOVL    #08,R1
 6      1006      464A            XOR.b   R1,R2          ;produce zero flag using byte and register source
 7      1008      7FF8            MOVH    #FFFF,R0
 8      100A      4680            XOR     $0,R0          ;produce negative flag using word and constant source
 9                                DONE
10      100C      3FFF            bra     DONE          ; loop
11                                end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1006, XOR is performed using values of size byte with a register source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 0x08 ^ 0x08 <p>This will produce:</p> <ul style="list-style-type: none"> DST.lowbyte = 0x00 CLEAR overflow flag CLEAR negative flag SET zero flag CLEAR carry flag 	<pre> R0: 0000 R1: 0008 R2: 0908 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1006: XOR RC: 0 WB: 1 SRC: R1 DST: R2 R0: 0000 R1: 0008 R2: 0900 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 </pre>
<p>At address 100A, XOR is performed using values of size word with a constant source:</p> <ul style="list-style-type: none"> DST <- 0xFF00 ^ \$0 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0xFF00 CLEAR overflow flag SET negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: FF00 R1: 0008 R2: 0900 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 100A: XOR RC: 1 WB: 0 CON: 0 DST: R0 R0: FF00 R1: 0008 R2: 0900 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 </pre>

The XOR instruction works as expected. It can execute with both sources and words, as well as with register and constant sources. It also sets the flags as expected.

Pass/Fail: PASS

7. AND instruction works as expected

Purpose: Checks for successful execution of AND instruction.

Configuration: The following program is loaded into the emulator. AND is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17

Input file name: AND.txt

Time of assembly: Mon 24 Jun 2024 04:36:55

```
1          org #1000
2          MAIN
3      1000    7FF8      MOVH    #FFFF,R0
4      1002    67F8      MOVL    #FF,R0
5      1004    6041      MOVL    #08,R1
6      1006    4701      AND     R0,R1          ;produce no flags using byte and register source
7      1008    6782      MOVL    #F0,R2
8      100A    4742      AND.B   R0,R2          ;produce negative flag using word and register source
9      100C    4780      AND     $0,R0          ;produce zero flag using word and constant source
10         DONE
11      100E    3FFF      bra     DONE          ; loop
12         end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1006, AND is performed using values of size byte with a register source:</p> <ul style="list-style-type: none">DST <- 0x008 & 0xFFFF <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x0008CLEAR overflow flagCLEAR negative flagCLEAR zero flagCLEAR carry flag	<pre>R0: FFFF R1: 0008 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1006: AND RC: 0 WB: 0 SRC: R0 DST: R1 R0: FFFF R1: 0008 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>
<p>At address 100A, AND is performed using values of size byte with a register source:</p> <ul style="list-style-type: none">DST.lowbyte <- 0xF0 & 0xFF <p>This will produce:</p> <ul style="list-style-type: none">DST.lowbyte = 0xF0CLEAR overflow flagSET negative flagCLEAR zero flagCLEAR carry flag	<pre>R0: FFFF R1: 0008 R2: 00F0 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 100A: AND RC: 0 WB: 1 SRC: R0 DST: R2 R0: FFFF R1: 0008 R2: 00F0 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0</pre>

At address 100A, AND is performed using values of size word with a constant source:

- $DST.lowbyte \leftarrow 0xFFFF \& \0

This will produce:

- $RESULT = 0x0000$
- CLEAR overflow flag
- CLEAR negative flag
- SET zero flag
- CLEAR carry flag

```
R0: FFFF
R1: 0008
R2: 00F0
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100C
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0

100C: AND    RC: 1 WB: 0 CON: 0 DST: R0
R0: 0000
R1: 0008
R2: 00F0
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100E
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0
```

The AND instruction works as expected. It can execute with both sources and words, as well as with register and constant sources. It also sets the flags as expected.

Pass/Fail: PASS

8. OR instruction works as expected

Purpose: Checks for successful execution of OR instruction.

Configuration: The following program is loaded into the emulator. OR is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: BIT.txt
Time of assembly: Mon 24 Jun 2024 05:36:20
 1                                org #1000
 2                                MAIN
 3      1000    7FF8                movh    #FFFF,R0
 4      1002    67F8                movl    #FF,R0
 5      1004    49A0                bit     $8,R0                ;produce no flag with word and constant source
 6      1006    6011                movl    #02,R1
 7      1008    7F71                movh    #EE00,R1
 8      100A    494A                bit.b   R1,R2                ;produce zero flag with byte and register source
 9                                DONE
10      100C    3FFF                bra     DONE                ; loop
11                                end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1004, BIT is performed using values of size word with a constant source:</p> <ul style="list-style-type: none"> DST <- 0xFFFF & (1 << \$8) <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x0100 CLEAR overflow flag CLEAR negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: FFFF R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1004: BIT RC: 1 WB: 0 CON: 8 DST: R0 R0: 0100 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>
<p>At address 100A, BIT is performed using values of size byte with a register source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 0x00 & (1 << 0x02) <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x00 CLEAR overflow flag CLEAR negative flag SET zero flag CLEAR carry flag 	<pre> R0: 0100 R1: EE02 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 100A: BIT RC: 0 WB: 1 SRC: R1 DST: R2 R0: 0100 R1: EE02 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 </pre>

The OR instruction works as expected. It can execute with both sources and words, as well as with register and constant sources. It also sets the flags as expected.

Pass/Fail: PASS

9. BIT instruction works as expected

Purpose: Checks for successful execution of BIT instruction.

Configuration: The following program is loaded into the emulator. BIT is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: BIS.txt
Time of assembly: Mon 24 Jun 2024 04:05:54
1          org #1000
2          MAIN
3      1000  67FA      movl    #FF,R2
4      1002  4B51      bis.b   R2,R1      ;produce result of 0 with byte and register source, set zero flag
5      1004  6041      movl    #08,R1
6      1006  4BA1      bis     $8,R1      ;produce result of 0x0108 with word and constant source
7      1008  603A      movl    #07,R2
8      100A  4B53      bis.b   R2,R3      ;produce negative result, negative flag
9          DONE
10     100C  3FFF      bra     DONE      ; loop
11          end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1002, BIS is performed using values of size byte with a register source:</p> <ul style="list-style-type: none">DST.lowbyte <- 0x00 (1 << 0xFF) <p>This will produce:</p> <ul style="list-style-type: none">DST.lowbyte = 0x00CLEAR overflow flagCLEAR negative flagSET zero flagCLEAR carry flag	<pre>R0: 0000 R1: 0000 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1002 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1002: BIS RC: 0 WB: 1 SRC: R2 DST: R1 R0: 0000 R1: 0000 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0</pre>
<p>At address 1006, BIS is performed using values of size word with a constant source:</p> <ul style="list-style-type: none">DST <- 0x0008 (1 << \$8) <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x0108CLEAR overflow flagCLEAR negative flagSET zero flagCLEAR carry flag	<pre>R0: 0000 R1: 0008 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 1006: BIS RC: 1 WB: 0 CON: 8 DST: R1 R0: 0000 R1: 0108 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>

At address 1008, OR is performed using values of size byte with a register source:

- $DST.lowbyte \leftarrow 0x0000 \mid (1 \ll 0x07)$

This will produce:

- $DST = 0x80$
- CLEAR overflow flag
- SET negative flag
- CLEAR zero flag
- CLEAR carry flag

```
R0: 0000
R1: 0108
R2: 0007
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100A
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0

100A: BIS    RC: 0 WB: 1 SRC: R2 DST: R3
R0: 0000
R1: 0108
R2: 0007
R3: 0080
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100C
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0
```

The BIT instruction works as expected. It can execute with both sources and words, as well as with register and constant sources. It also sets the flags as expected.

Pass/Fail: PASS

10. BIC instructions work as expected

Purpose: Checks for successful execution of BIC instruction.

Configuration: The following program is loaded into the emulator. BIC is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: BIC.txt
Time of assembly: Mon 24 Jun 2024 06:03:35
1          org #1000
2          MAIN
3      1000    7FF8          movh    #FFFF,R0
4      1002    6078          movl    #0F,R0
5      1004    4AA0          bic      $8,R0          ;produce negative flag with word and constant source
6      1006    4AD0          bic.b    $2,R0          ;produces no flags with word and register source
7      1008    6011          movl    #02,R1
8      100A    6008          movl    #01,R0
9      100C    4A41          bic.b    R0,R1          ;produce zero flag with byte and register source
10         DONE
11      100E    3FFF          bra      DONE          ; loop
12         end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1004, BIC is performed using values of size word with a constant source:</p> <ul style="list-style-type: none"> DST <- 0xFF0F & ~(1 << \$8) <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0xFE0F CLEAR overflow flag SET negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: FF0F R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1004: BIC RC: 1 WB: 0 CON: 8 DST: R0 R0: FE0F R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 </pre>
<p>At address 1006, BIC is performed using values of size byte with a constant source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 0x0F (1 << \$2) <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x0B CLEAR overflow flag CLEAR negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: FE0F R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 1006: BIC RC: 1 WB: 1 CON: 2 DST: R0 R0: FE0B R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>
<p>At address 1006, BIC is performed using values of size byte with a register source:</p> <ul style="list-style-type: none"> DST.lowbyte <- 0x02 (1 << 0x01) <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x00 CLEAR overflow flag CLEAR negative flag SET zero flag CLEAR carry flag 	<pre> R0: FE01 R1: 0002 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 100C: BIC RC: 0 WB: 1 SRC: R0 DST: R1 R0: FE01 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100E prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 </pre>

The BIC instruction works as expected. It can execute with both sources and words, as well as with register and constant sources. It also sets the flags as expected.

Pass/Fail: PASS

11. BIS instructions work as expected

Purpose: Checks for successful execution of BIS instruction.

Configuration: The following program is loaded into the emulator. BIS is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: BIS.txt
Time of assembly: Mon 24 Jun 2024 04:05:54
 1                               org #1000
 2                               MAIN
 3      1000    67FA            movl    #FF,R2
 4      1002    4B51            bis.b   R2,R1            ;produce result of 0 with byte and register source, set zero flag
 5      1004    6041            movl    #08,R1
 6      1006    4BA1            bis     $8,R1            ;produce result of 0x0108 with word and constant source
 7      1008    603A            movl    #07,R2
 8      100A    4B53            bis.b   R2,R3            ;produce negative result, negative flag
 9
10      100C    3FFF            bra     DONE            ; loop
11                               end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1002, BIS is performed using values of size byte with a register source:</p> <ul style="list-style-type: none">DST.lowbyte <- 0x00 (1 << 0xFF) <p>This will produce:</p> <ul style="list-style-type: none">DST.lowbyte = 0x00CLEAR overflow flagCLEAR negative flagSET zero flagCLEAR carry flag	<pre>R0: 0000 R1: 0000 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1002 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1002: BIS RC: 0 WB: 1 SRC: R2 DST: R1 R0: 0000 R1: 0000 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0</pre>
<p>At address 1006, BIS is performed using values of size word with a constant source:</p> <ul style="list-style-type: none">DST <- 0x0008 (1 << \$8) <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x0108CLEAR overflow flagCLEAR negative flagSET zero flagCLEAR carry flag	<pre>R0: 0000 R1: 0008 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 1006: BIS RC: 1 WB: 0 CON: 8 DST: R1 R0: 0000 R1: 0108 R2: 00FF R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>

At address 1006, BIS is performed using values of size byte with a register source:

- DST.lowbyte <- 0x0000 | (1 << 0x07)

This will produce:

- DST = 0x80
- CLEAR overflow flag
- SET negative flag
- CLEAR zero flag
- CLEAR carry flag

```
R0: 0000
R1: 0108
R2: 0007
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100A
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0

100A: BIS    RC: 0 WB: 1 SRC: R2 DST: R3
R0: 0000
R1: 0108
R2: 0007
R3: 0080
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100C
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0
```

The BIS instruction works as expected. It can execute with both sources and words, as well as with register and constant sources. It also sets the flags as expected and does not change the destination value when source or constant value is larger than max number of bits.

Pass/Fail: PASS

12. MOV instructions work as expected

Purpose: Checks for successful execution of BIS instruction.

Configuration: The following program is loaded into the emulator. BIS is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: MOV.txt
Time of assembly: Mon 24 Jun 2024 06:16:51
1      org #1000
2      MAIN
3      1000    7FF8      movh    #FFFF,R0
4      1002    6078      movl    #0F,R0
5      1004    6329      movl    #65,R1
6      1006    4C01      mov     R0,R1
7      1008    6329      movl    #65,R1
8      100A    4C48      mov.b   R1,R0
9      DONE
10     100C    3FFF      bra     DONE      ; loop
11     end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1006, MOV is performed using values of size word:</p> <ul style="list-style-type: none"> DST = 0x0065 DST = FF0F <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0xFF0F 	<pre> R0: FF0F R1: 0065 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1006: MOV WB: 0 SRC: R0 DST: R1 R0: FF0F R1: FF0F R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>
<p>At address 100A, MOV is performed using values of size byte:</p> <ul style="list-style-type: none"> DST.lowbyte = 0x0F DST.lowbyte <- 0x65 <p>This will produce:</p> <ul style="list-style-type: none"> DST.lowbyte = 0x65 	<pre> R0: FF0F R1: FF65 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 100A: MOV WB: 1 SRC: R1 DST: R1 R0: 0065 R1: FF65 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100C prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>

The MOV instruction works as expected. It can execute with both bytes and words.

Pass/Fail: PASS

13. SWAP instructions work as expected

Purpose: Checks for successful execution of SWAP instruction.

Configuration: The following program is loaded into the emulator. SWAP is executed.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: SWAP.txt
Time of assembly: Mon 24 Jun 2024 06:27:26
 1                               org #1000
 2                               MAIN
 3      1000      7FF8          movh    #FFFF,R0
 4      1002      6078          movl    #0F,R0
 5      1004      6329          movl    #65,R1
 6      1006      4C81          swap     R0,R1
 7                               DONE
 8      1008      3FFF          bra     DONE          ; loop
 9                               end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1006, SWAP is performed:</p> <ul style="list-style-type: none">SRC = 0x0065DST = 0xFF0F <p>This will produce:</p> <ul style="list-style-type: none">SRC = 0xFF0FDST = 0x0065	<pre>R0: FF0F R1: 0065 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1006: SWAP SRC: R0 DST: R1 R0: 0065 R1: FF0F R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>

The SWAP instruction works as expected.

Pass/Fail: PASS

14. SRA instructions work as expected

Purpose: Checks for successful execution of SRA instruction.

Configuration: The following program is loaded into the emulator. SRA is executed with byte and word.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: SRA.txt
Time of assembly: Mon 24 Jun 2024 06:42:47
1                               org #1000
2                               MAIN
3      1000      7FF8                movh    #FFFF,R0
4      1002      4D00                SRA     R0
5      1004      6009                movl    #1,R1
6      1006      4D41                SRA.b   R1
7
8      1008      3FFF                bra     DONE        ; loop
9                               end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At address 1002, SRA is performed using values of size word:</p> <ul style="list-style-type: none">DST = 0xFF00 >> 1 <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x7F80CLEAR overflow flagSET negative flagCLEAR zero flagCLEAR carry flag	<pre>R0: FF00 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1002 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1002: SRA WB: 0 DST: R0 R0: FF80 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0</pre>
<p>At address 1006, SRA is performed using values of size byte.</p> <ul style="list-style-type: none">DST.lowbyte <- 0x01 >> 1 <p>This will produce:</p> <ul style="list-style-type: none">RESULT = 0x00CLEAR overflow flagCLEAR negative flagSET zero flagSET carry flag	<pre>R0: FF80 R1: 0001 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 1006: SRA WB: 1 DST: R1 R0: FF80 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1</pre>

The SRA instruction works as expected. It can execute with both bytes and words, and sets the flags as expected.

Pass/Fail: PASS

15. RRC instructions work as expected

Purpose: Checks for successful execution of RRC instruction.

Configuration: The following program is loaded into the emulator. RRC is executed with byte and word.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: BIS.txt
Time of assembly: Mon 24 Jun 2024 04:05:54
1                               org #1000
2                               MAIN
3      1000    67FA      movl    #FF,R2
4      1002    4B51      bis.b   R2,R1      ;produce result of 0 with byte and register source, set zero flag
5      1004    6041      movl    #08,R1
6      1006    4BA1      bis     $8,R1      ;produce result of 0x0108 with word and constant source
7      1008    603A      movl    #07,R2
8      100A    4B53      bis.b   R2,R3      ;produce negative result, negative flag
9
10     100C    3FFF      bra     DONE      ; loop
11                               end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At memory address 1006, RRC is performed using values of size word:</p> <ul style="list-style-type: none">DST = 0x7FFDDST <- 0x7FFD >> 1 through carry(0) <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x3FFECLEAR overflow flagCLEAR negative flagCLEAR zero flagSET zero flagSET carry flag	<pre>R0: 7FFD R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1006: RRC WB: 0 DST: R0 R0: 3FFE R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 1</pre>
<p>At address 1008, RRC is performed using values of size byte:</p> <ul style="list-style-type: none">DST.lowbyte = 0xFEDST.lowbyte <- 0xFE >> 1 through carry (1) <p>This will produce:</p> <ul style="list-style-type: none">DST.lowbyte = 0xFFCLEAR overflow flagSET negative flagCLEAR zero flagCLEAR carry flag	<pre>R0: 3FFE R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 1 1008: RRC WB: 1 DST: R0 R0: 3FFF R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0</pre>

At address 100C, RRC is performed using values of size word:

- DST = 0x0001
- DST <- 0xF001 >> 1 through carry (0)

This will produce:

- DST = 0x0000
- CLEAR overflow flag
- CLEAR negative flag
- SET zero flag
- SET carry flag

```
R0: 3FFF
R1: 0001
R2: 0000
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100C
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0

100C: RRC    WB: 0 DST: R1
R0: 3FFF
R1: 0000
R2: 0000
R3: 0000
R4 (BP): 0000
R5 (LR): 0000
R6 (SP): 0000
R7 (PC): 100E
prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 1
```

The RRC instruction works as expected. It can execute with both bytes and words, and sets the flags as expected.

Pass/Fail: PASS

16. SWPB instructions work as expected

Purpose: Checks for successful execution of SWPB instruction.

Configuration: The following program is loaded into the emulator. SWPB is executed.

X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17

Input file name: SWPB.txt

Time of assembly: Mon 24 Jun 2024 07:58:59

```
1          org #1000
2          MAIN
3      1000      4D18      swpb      R0
4      1002      67D0      movl      #FA,R0
5      1004      4D18      swpb      R0
6      1006      6048      movl      #09,R0
7      1008      4D18      swpb      R0
8          DONE
9      100A      3FFF      bra        DONE          ; loop
10         end MAIN
```

Successful completion of assembly - 1P

Expected Results	Actual Results
<p>At address 1000, SWPB is performed:</p> <ul style="list-style-type: none"> DST = 0x0000 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x0000 CLEAR overflow flag CLEAR negative flag SET zero flag CLEAR carry flag 	<pre> R0: 0000 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1000 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1000: SWPB DST: R0 R0: 0000 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1002 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 </pre>
<p>At address 1004, SWPB is performed:</p> <ul style="list-style-type: none"> DST = 0x00FA <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0xFA00 CLEAR overflow flag SET negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: 00FA R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 1004: SWPB DST: R0 R0: FA00 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 </pre>
<p>At address 1008, SWPB is performed:</p> <ul style="list-style-type: none"> DST = 0xFA09 <p>This will produce:</p> <ul style="list-style-type: none"> DST = 0x09FA CLEAR overflow flag CLEAR negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: FA09 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 1008: SWPB DST: R0 R0: 09FA R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>

The SWPB instruction works as expected. It sets registers as expected

Pass/Fail: PASS

17. SXT instructions work as expected

Purpose: Checks for successful execution of SXT instruction.

Configuration: The following program is loaded into the emulator. SXT is executed.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: SXT.txt
Time of assembly: Mon 24 Jun 2024 08:22:27
1                                     org #1000
2             MAIN
3     1000    4D20          sxt      R0
4     1002    67D0          movl    #FA,R0
5     1004    4D20          sxt      R0
6     1006    6051          movl    #0A,R1
7     1008    4D21          sxt      R1
8             DONE
9     100A    3FFF          bra     DONE      ; loop
10            end MAIN
Successful completion of assembly - 1P
```

Expected Results	Actual Results
<p>At memory address 1000, SXT is performed:</p> <ul style="list-style-type: none">DST = 0x0000 <p>This will produce:</p> <ul style="list-style-type: none">DST = 0x0000CLEAR overflow flagCLEAR negative flagSET zero flagCLEAR carry flag	<pre>R0: 0000 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1000 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1000: SXT DST: R0 R0: 0000 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1002 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0</pre>
<p>At address 1004, SXT is performed:</p> <ul style="list-style-type: none">DST = 0x00FA <p>This will produce:</p> <ul style="list-style-type: none">RESULT = 0xFFFFACLEAR overflow flagSET negative flagCLEAR zero flagCLEAR carry flag	<pre>R0: 00FA R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1004 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 1 c: 0 1004: SXT DST: R0 R0: FFFA R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1006 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0</pre>

<p>At address 1008, SXT is performed:</p> <ul style="list-style-type: none"> DST = 0x000A <p>This will produce:</p> <ul style="list-style-type: none"> RESULT = 0x000A CLEAR overflow flag CLEAR negative flag CLEAR zero flag CLEAR carry flag 	<pre> R0: FFFA R1: 000A R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1008 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 1 z: 0 c: 0 1008: SXT DST: R1 R0: FFFA R1: 000A R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 100A prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 </pre>
---	---

The SXT instruction works as expected. It sets registers as expected.

Pass/Fail: PASS

18. Encounter instruction not a part of A2 requirements

Purpose: Checks for successful execution of BIS instruction.

Configuration: The following program is loaded into the emulator. BIS is executed with byte and word, register and constant sources, and related flags each set in at least one operation.

```

X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: nota2.txt
Time of assembly: Mon 24 Jun 2024 08:37:38
 1          org #1000
 2          MAIN
 3      1000      4DB0      SETCC      v
 4          DONE
 5      1002      3FFF      bra      DONE      ; loop
 6          end MAIN
Successful completion of assembly - 1P

```

Expected Results	Actual Results
<ul style="list-style-type: none"> The emulator will not execute any commands. 	<pre> R0: 0000 R1: 0000 R2: 0000 R3: 0000 R4 (BP): 0000 R5 (LR): 0000 R6 (SP): 0000 R7 (PC): 1000 prev: 3 flt: 0 curr: 7 v: 0 slp: 0 n: 0 z: 0 c: 0 1000: 4DB0 1002: 3FFF End: PC: 1004 Clk: 4 </pre>

The emulator behaved as expected.

Pass/Fail: PASS