

Testing

1. Prints first instruction as NOP

Purpose: Checks for successful recognition of first instruction as NOP, which is MOV R0,R0.

Configuration: Load any file and check for first printed instruction.

Expected Results: The first instruction should be MOV, with SRC of R0, and DST of R0.

Actual results: The instructions print as expected.

```
0ffe: MOV    WB: 1 SRC: R0 DST: R0
1000: MOVL   BYTE: 0040 DST: R0
```

2. Prints first real instruction at the correct memory start address

Purpose: Checks for successful recognition of program memory address and prints first instruction after “MOV R0, R0” at that address.

Configuration: Load any file and check for second printed instruction.

Expected Results: The first instruction should be MOV, with SRC of R0, and DST of R0.

Actual results: The instructions print as expected.

```
Source filename: ECED3403_lab1_1.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0
1000: MOVL   BYTE: 0040 DST: R0
```

3. Recognizes end of program and prints instruction hexadecimal 0000

Purpose: Checks for successful recognition of end of program by printing 0000 after completion of reading the program.

Configuration: Load any file and check that the instruction hexadecimal 0000 is printed after the last command.

```
45      ;decrement the current counter r1
46      1016  4289      sub      $1,R1
47      ;
48      ;jump to loop
49      1018  3FF9      bra      LOOP
50      DONE
51      101A  3FFF      bra      DONE      ; loop
52      end MAIN
Successful completion of assembly - 2P
```

Expected Results: The 0000 instruction bytes will be printed at memory address 101C, after instruction 3FFF.

Actual results: The instructions print as expected,

```
101a: 3fff
101c: 0000
```

4. Recognizes all commands from MOVL to MOVH

Purpose: Checks for if the program recognizes MOVL to MOVH instructions' mnemonics and operands.

Configuration: Load a .xme file with MOVL to MOVH instructions.

```

                                org #1000
MAIN
                                MOVL #1234,R1
                                MOVLZ #1234,R1
                                MOVLS #1234,R1
                                MOVH #1234,R1
DONE
                                bra      DONE          ; loop
end MAIN
```

Expected Results: The memory will print all expected instruction mnemonics and operands.

Actual results: The instructions print as expected.

```
Source filename: MOVLtoMOVH.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV   WB: 1 SRC: R0 DST: R0
1000: MOVL  BYTE: 0034 DST: R1
1002: MOVLZ BYTE: 0034 DST: R1
1004: MOVLS BYTE: 0034 DST: R1
1006: MOVH  BYTE: 0012 DST: R1
1008: 3fff
100a: 0000
```

5. Recognizes all commands from MOV to SWAP

Purpose: Checks for if the loader recognizes MOV to SWAP's instruction mnemonics and operands.

Configuration: Load a .xme file with MOV to SWAP instructions.

```
MAIN
    MOV R1,R2
    SWAP R1,R2
DONE
    bra     DONE
end MAIN
```

Expected Results: The memory will print all expected instruction mnemonics and operands.

Actual results: The instructions print as expected.

```
Source filename: MOVtoSWAP.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0
1000: MOV    WB: 0 SRC: R1 DST: R2
1002: SWAP   SRC: R1 DST: R2
1004: 3fff
1006: 0000
```

6. Recognizes all commands from SRA to RRC

Purpose: Checks for if the loader recognizes SRA to RRC's instruction mnemonics and operands.

Configuration: Load a .xme file with SRA to RRC instructions.

```
MAIN
    SRA R1
    RRC R2
DONE
    bra     DONE
end MAIN
```

Expected Results: The memory will print all expected instruction mnemonics and operands.

Actual results: The instructions print as expected.

```
Source filename: SRAtorRC.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: SRA     WB: 0 DST: R1

1002: RRC     WB: 0 DST: R2

1004: 3fff

1006: 0000
```

7. Recognizes all commands from SWPB to SXT

Purpose: Checks for if the loader recognizes SWPB to SXT's instruction mnemonics and operands.

Configuration: Load a .xme file with SWPB to SXT instructions.

```

MAIN                                org 1000
                                SWPB R1
                                SXT R2

DONE                                bra     DONE

end MAIN
```

Expected Results: The memory will print all expected instruction mnemonics and operands.

Actual results: The instructions print as expected.

```
Source filename: SWPBtoSXT.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: SWPB    DST: R1

1002: SXT     DST: R2

1004: 3fff

1006: 0000
```

8. Recognizes all commands from ADD to BIS

Purpose: Checks for if the loader recognizes ADD to BIS's instruction mnemonics and operands.

Configuration: Load a .xme file with ADD to BIS instructions.

```

                                org #1000
MAIN
                                ADD R1,R2
                                ADDC R1,R2
                                SUB R1,R2
                                SUBC R1,R2
                                DADD R1,R2
                                CMP R1,R2
                                XOR R1,R2
                                AND R1,R2
                                OR R1,R2
                                BIT R1,R2
                                BIS R1,R2

DONE
                                bra     DONE
end MAIN
```

Expected Results: The memory will print all expected instruction mnemonics and operands.

Actual results: The instructions print as expected.

```
ADDTOBIS.XME
Source filename: ADDtoBIS.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0
1000: ADD     RC: 0 WB: 0 SRC: R1 DST: R2
1002: ADDC    RC: 0 WB: 0 SRC: R1 DST: R2
1004: SUB     RC: 0 WB: 0 SRC: R1 DST: R2
1006: SUBC    RC: 0 WB: 0 SRC: R1 DST: R2
1008: DADD    RC: 0 WB: 0 SRC: R1 DST: R2
100a: CMP     RC: 0 WB: 0 SRC: R1 DST: R2
100c: XOR     RC: 0 WB: 0 SRC: R1 DST: R2
100e: AND     RC: 0 WB: 0 SRC: R1 DST: R2
1010: OR      RC: 0 WB: 0 SRC: R1 DST: R2
1012: BIT     RC: 0 WB: 0 SRC: R1 DST: R2
1014: BIS     RC: 0 WB: 0 SRC: R1 DST: R2
1016: 3fff
1018: 0000
```

9. Recognizes invalid commands

Purpose: Checks for if the loader recognizes LD to ST as invalid commands

Configuration: Load a .xme file with LD to ST instructions.

```

,
MAIN          org #1000
              LD R1,R0
              ST R1,R0
DONE
              bra      DONE
end MAIN
```

Expected Results: The memory will print the instruction opcode instead of the instruction mnemonic and operands.

Actual results: The instruction opcodes print as expected.

```
Source filename: LDtoST.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: 5808

1002: 5c08

1004: 3fff

1006: 0000
```

10. Can recognize all 4 registers

Purpose: Checks for if the loader can recognize R0, R1, R2, and R3.

Configuration: Load a .xme file that uses all four registers.

```

MAIN
              MOV R0,R1
              MOV R2,R3
DONE
              bra      DONE
end MAIN|
```

Expected Results: The loader will recognize all four registers.

Actual results: The loader behaved as expected.

```
Source filename: registers.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: MOV    WB: 0 SRC: R0 DST: R1

1002: MOV    WB: 0 SRC: R2 DST: R3

1004: 3fff

1006: 0000
```

11. Can recognize all constants in constant array

Purpose: Checks for if the loader can recognize valid constant values of 0, 1, 2, 4, 8, 16, 32, -1.

Configuration: Load a .xme file that uses all valid constant values.

```
MAIN
    MOV R0,R1
    MOV R2,R3
DONE
    bra     DONE
end MAIN
```

Expected Results: The loader will print a success statement after xme file is loaded.

Actual results: The loader behaves as expected.

```
Source filename: constant.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: ADD    RC: 1 WB: 0 CON: 0 DST: R1

1002: ADD    RC: 1 WB: 0 CON: 1 DST: R1

1004: ADD    RC: 1 WB: 0 CON: 2 DST: R1

1006: ADD    RC: 1 WB: 0 CON: 4 DST: R1

1008: ADD    RC: 1 WB: 0 CON: 8 DST: R1

100a: ADD    RC: 1 WB: 0 CON: 16 DST: R1

100c: ADD    RC: 1 WB: 0 CON: 32 DST: R1

100e: ADD    RC: 1 WB: 0 CON: -1 DST: R1

1010: 3fff

1012: 0000
```

12. Can print more than one file's opcode and operands

Purpose: Checks for the loader successfully printing more than one file's instruction mnemonic and operands, one after the other.

Configuration: Two files are loaded in.

Expected Results: The loader will be able to successfully print both file's instruction mnemonic and operands, one after the other.

Actual results: The loader behaves as expected.

```
Source filename: MOVtoSWAP.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: MOV    WB: 0 SRC: R1 DST: R2

1002: SWAP   SRC: R1 DST: R2

1004: 3fff

1006: 0000

Option: 1
Enter .XME file to load
registers.xme
Source filename: registers.asm
File read - no errors detected. Starting address: 1000

0ffe: MOV    WB: 1 SRC: R0 DST: R0

1000: MOV    WB: 0 SRC: R0 DST: R1

1002: MOV    WB: 0 SRC: R2 DST: R3

1004: 3fff

1006: 0000

Option: |
```