RISC-V RV32M Processor

Anika Mahesh, Henry Tejada Deras

Code: [GitHub - H-TejadaDeras/32-bit-RISC-V-Processor: 32-bit RISC-V Processor Integer Microprocessor with a Harvard Architecture for Computer Architecture Course (ENGR3410-01.25FA) in the 2025 Fall Semester.]

Note: In our project proposal, we listed two different ideas of which we were going to do at least one. We decided on doing the idea consisting of implementing the RV32M instruction set.

RV32M Extension Implementation Design Overview

We implemented 8 new instructions into the RISC-V processor we implemented in mini project 4: div, divu, rem, remu, mul, mulh, mulhsu, mulhu. To accommodate for more complex operations like multiplication and division, we added functionality to change the number of execution clock cycles that each operation takes to run. Multiplication and division instructions take longer to run compared to all the other instructions. All of the new operations are registered ALU operations so they use the same opcode. However, the M instruction set instructions have different funct7 values than base ALU operations. Because of this, an if statement in the ALU that checks the funct7 value is used. The logic for implementing and decoding register ALU operations is unchanged as it works for all the M extension operations. The different instructions can be distinguished by the funct3 values and this can be used to tell them apart. The different instructions we implemented and tests for these instructions are shown below.

`div`
Unsigned division between two values stored in register.
`divu`
Unsigned division between two values stored in register.
`rem`
signed remainder between two values stored in register.
`remu`
Unsigned remainder between two values stored in register.
`mul`
The lower 32 bits of a signed multiplication between two values stored in the register.
`mulh`
The upper 32 bits of a signed multiplication between two values stored in register.
`mulhsu`
The upper 32 bits of a multiplication between two values stored in register where the first value is signed and the 2nd is unsigned.
`mulhu`
The upper 32 bits of a signed multiplication between two values stored in register.

Instruction Functionality Validation Tests

Division Operations (division_test.s + test_mem_div_.hex)

Testing `div`, `divu`, `rem`, and `remu` instructions. Note: This test makes use of test_mem_div_.hex

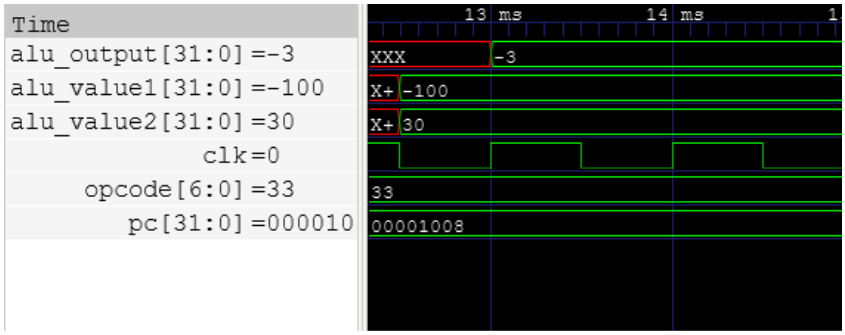
division_test.s

```
lw x1, 4(x1)
lw x2, 8(x1)
div x3, x2, x1
divu x4, x2, x1
rem x5, x2, x1
remu x7, x2, x1
div x7, x5, x3
div x3, x2, x1
```

test_mem_div_.hex

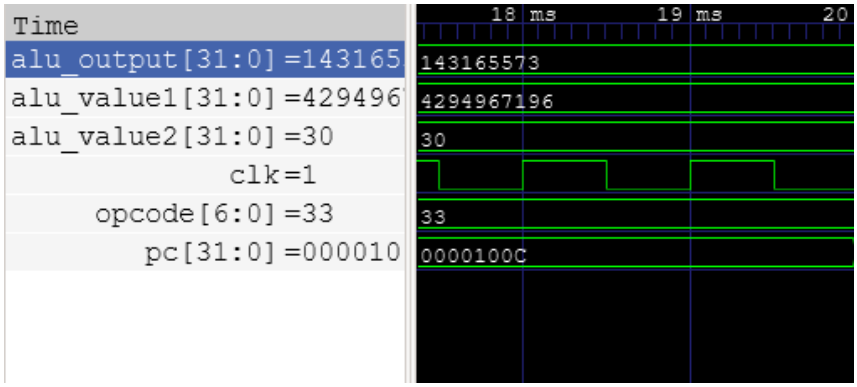
```
ffffff9c
0000001e
ffffff9c
ffffff9c
…
ffffff9c
00000000
…
```

div x3, x2, x1 x2 = ffffff9c x1 = 0000001e (loaded by lw)



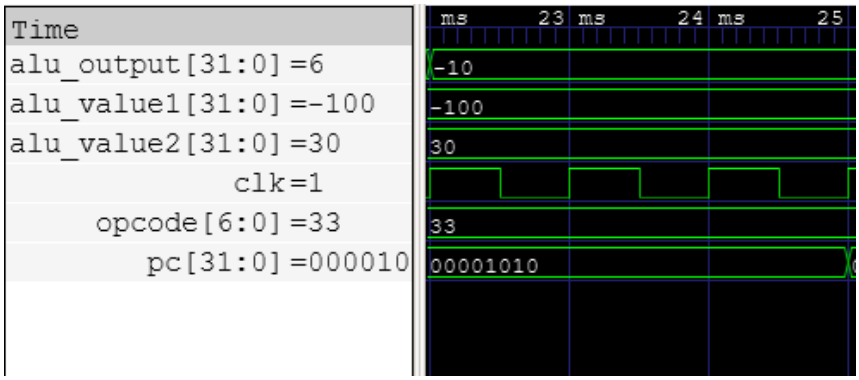
-100/3 is -3 so true

divu x4, x2, x1 x2 = ffffff9c x1 = 0000001e (loaded by lw)



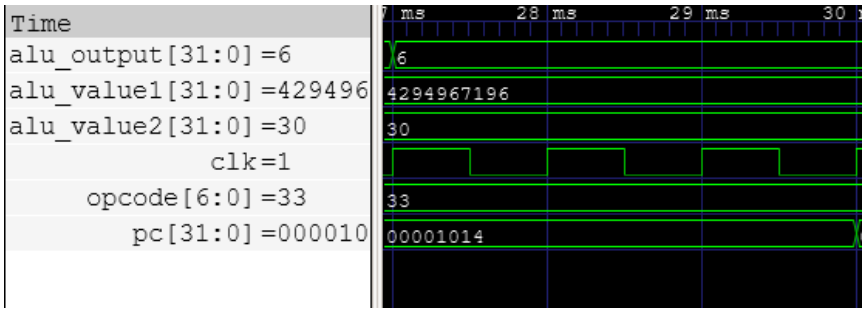
4,294,967,196 / 30 = 143,165,573 so true

rem x5, x2, x1 x2 = ffffff9c x1 = 0000001e (loaded by lw)



-100 % 30 = -10 so true

remu x7, x2, x1 x2 = ffffff9c x1 = 0000001e (loaded by lw)



4294967196 % 30 = 6 so true

Multiplication Instructions (mult_instruct_test.s + tester_mult_dmem.hex)

Testing mul, mulh, mulhsu, and mulhu instructions. Note: This test makes use of tester_mult_dmem.hex.

tester_mult_dmem.hex

```
00000008
0000000A
01010101
00011010
00000001
00000000
09876354
98756462
18647312
FFFF0123
…
```

mult_instruct_test.s

```
add x2, x0, x0 # Set stack pointer
lw x5 0(x3) # 00000008 -> x5
lw x6 4(x3) # 0000000A -> x6
mul x7, x5, x6 # 8 [00000008] * 10 [0000000A] = 80
[00000050]
lw x6 36(x3) # FFFF0123 -> x6
mul x7, x5, x6 # 8 [00000008] * -65245 [FFFF0123] =
-521960 [FFF80918]
lw x5, 28(x3) # 98756462 -> x5
lw x6, 24(x3) # 09876354 -> x6
mul x7, x5, x6 # -1737137054 [98756462] *
159867732 [09876354] = -277712160996141528
[FC255E42CE04D628]; saves lower 32 bits
[CE04D628]
mulh x7, x5, x6 # -1737137054 [98756462] *
159867732 [09876354] = -277712160996141528
[FC255E42CE04D628]; saves upper 32 bits
[FC255E42]
mulhsu x7, x5, x6 # -1737137054 [98756462] *
159867732 [09876354] = -277712160996141528
[FC255E42CE04D628]; saves upper 32 bits
[FC255E42]
mulhu x7, x5, x6 # 2557830242 [98756462] *
159867732 [09876354] = 408914519629551144
[05ACC196CE04D640]; saves upper 32 bits
[05ACC196]
mulhu x7, x6, x5 # 2557830242 [09876354] *
159867732 [98756462] = 408914519629551144
[05ACC196CE04D640]; saves upper 32 bits
[05ACC196]
```

Processor Status (when running the above program)		
Program Counter: 00001000 Current Instruction: 00000133 [add x2, x0, x0] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000	Program Counter: 00001004 Current Instruction: 0001a283 [lw x5 0(x3)] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000	Program Counter: 00001008 Current Instruction: 0041a303 [lw x6 4(x3)] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000

Register x5: 00000000 Register x6: 00000000 Register x7: 00000000 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000	Register x5: 00000008 Register x6: 00000000 Register x7: 00000000 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000	Register x5: 00000008 Register x6: 0000000a Register x7: 00000000 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000
Program Counter: 0000100c Current Instruction: 026283b3 [mul x7, x5, x6] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 00000008 Register x6: 0000000a Register x7: 00000050 Register x8: 00000000 Register x9: 00000000	Program Counter: 00001010 Current Instruction: 0241a303 [lw x6 36(x3)] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 00000008 Register x6: ffff0123 Register x7: 00000050 Register x8: 00000000 Register x9: 00000000	Program Counter: 00001014 Current Instruction: 026283b3 [mul x7, x5, x6] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 00000008 Register x6: ffff0123 Register x7: fff80918 Register x8: 00000000 Register x9: 00000000

Register x10: 00000000 Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mul properly saves 00000050, the correct answer, to register x7.	Register x10: 00000000 Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000	Register x10: 00000000 Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mul properly saves FFF80918, the correct answer, to register x7.
Program Counter: 00001018 Current Instruction: 01c1a283 [lw x5, 28(x3)] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: ffff0123 Register x7: fff80918 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000	Program Counter: 0000101c Current Instruction: 0181a303 [lw x6, 24(x3)] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: 09876354 Register x7: fff80918 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000	Program Counter: 00001020 Current Instruction: 026283b3 [mul x7, x5, x6] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: 09876354 Register x7: ce04d628 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000

Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000	Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000	Register x11: 00000000 Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mul properly saves CE04D628, the correct answer, to register x7.
Program Counter: 00001024 Current Instruction: 026293b3 [mulh x7, x5, x6] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: 09876354 Register x7: fc255e42 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000	Program Counter: 00001028 Current Instruction: 0262a3b3 [mulhsu x7, x5, x6] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: 09876354 Register x7: fc255e42 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000	Program Counter: 0000102c Current Instruction: 0262b3b3 [mulhu x7, x5, x6] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: 09876354 Register x7: 05acc196 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000

Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mulhsu properly saves FC255E42, the correct answer, to register x7.	Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mulhsu properly saves FC255E42, the correct answer, to register x7.	Register x12: 00000000 Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mulhu properly saves 05ACC196, the correct answer, to register x7.
Program Counter: 00001030 Current Instruction: 025333b3 [mulhu x7, x6, x5] Register x0: 00000000 Register x1: 00000000 Register x2: 00000000 Register x3: 00000000 Register x4: 00000000 Register x5: 98756462 Register x6: 09876354 Register x7: 05acc196 Register x8: 00000000 Register x9: 00000000 Register x10: 00000000 Register x11: 00000000 Register x12: 00000000	The rest of the processor cycles were omitted due to there being no more instructions.	

Register x13: 00000000 Register x14: 00000000 Register x15: 00000000 Register x16: 00000000 Register x17: 00000000 Register x18: 00000000 Register x19: 00000000 Register x20: 00000000 Register x21: 00000000 Register x22: 00000000 Register x23: 00000000 Register x24: 00000000 Register x25: 00000000 Register x26: 00000000 Register x27: 00000000 Register x28: 00000000 Register x29: 00000000 Register x30: 00000000 Register x31: 00000000 Note: mulhu properly saves 05ACC196, the correct answer, to register x7.		