# EEE 120

## Lab 4 Answer Sheet

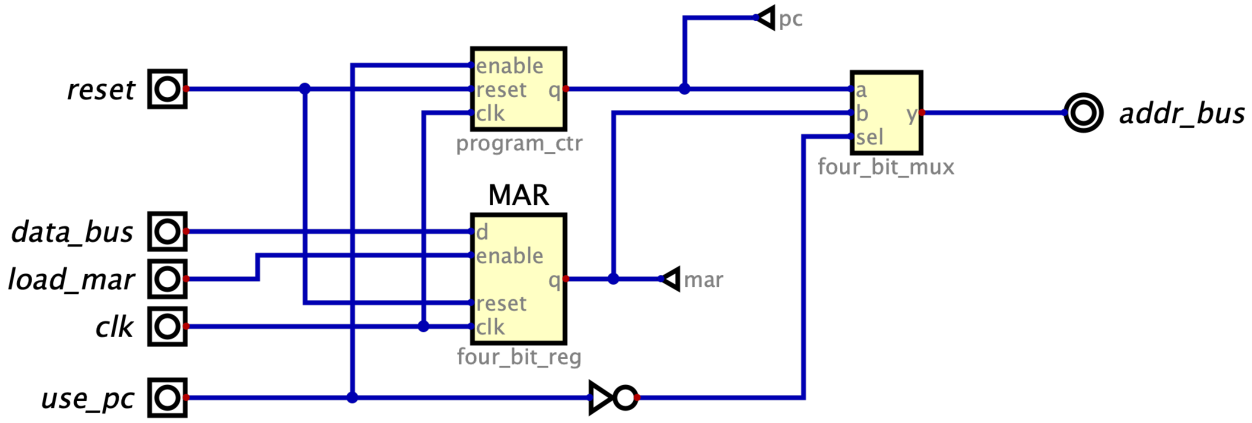
## The Complete Microprocessor

Name: Fauzan Amaan Mohammed.   Instructor/Time: Josh Hihath (Mon & Wed 3:00 to 4:15)

Date: 11th April 2025

## Task 4-1: Build and Test the Memory-Address-Generation Circuit

Include a picture of your Digital circuit here:



Please comment on the single biggest issue you were facing when designing the circuit.

No issues

Did the circuit behave as expected? If no, what was wrong?

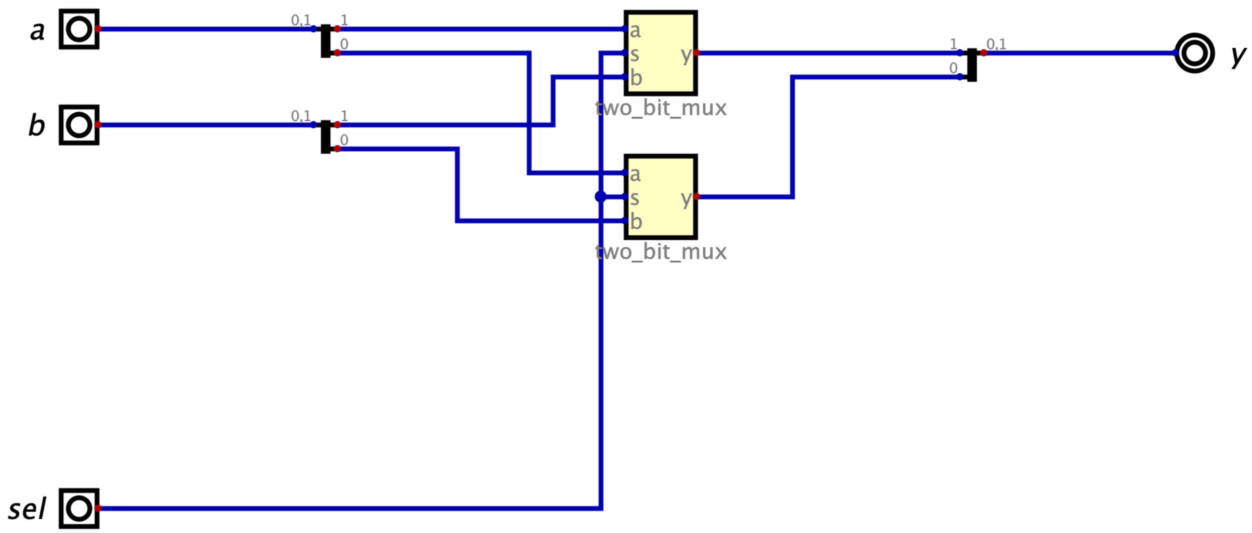
The circuit behaved as expected. The addr_bus incremented as it should as the clock changes.

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues

## Task 4-2: Build and Test the Controller Circuit

Include a picture of your two_bit_bus_mux circuit here:



Please comment on the single biggest issue you were facing when designing the circuit.

No issues

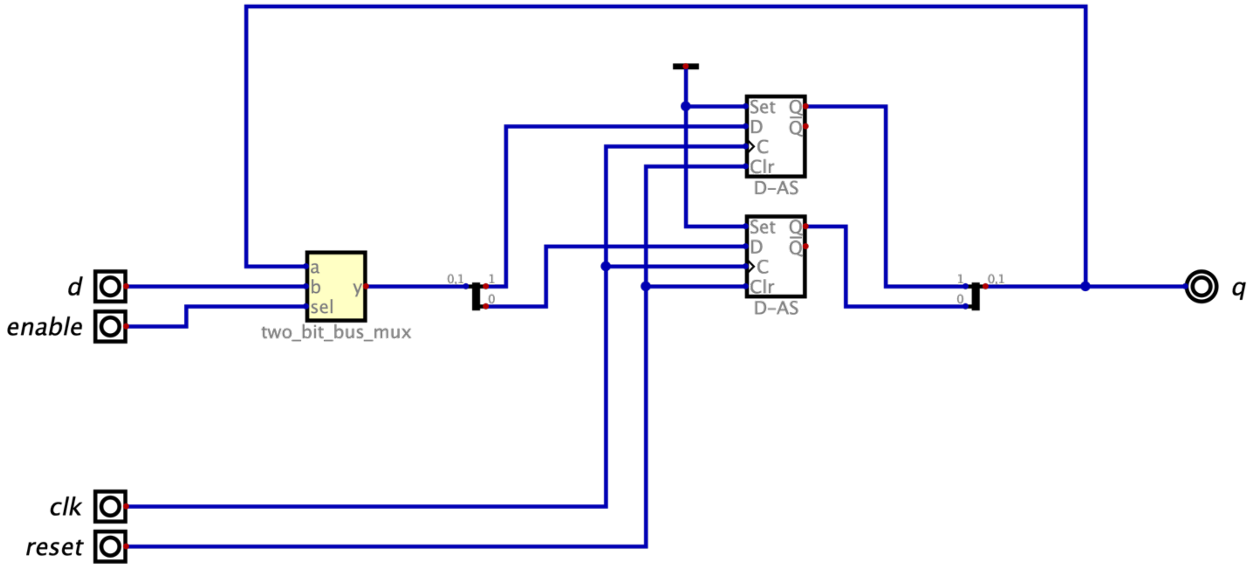
Did the circuit behave as expected? If no, what was wrong?

The circuit behaved as expected.

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues

Include a picture of your two_bit_reg circuit here:



Please comment on the single biggest issue you were facing when designing the circuit.

No issues

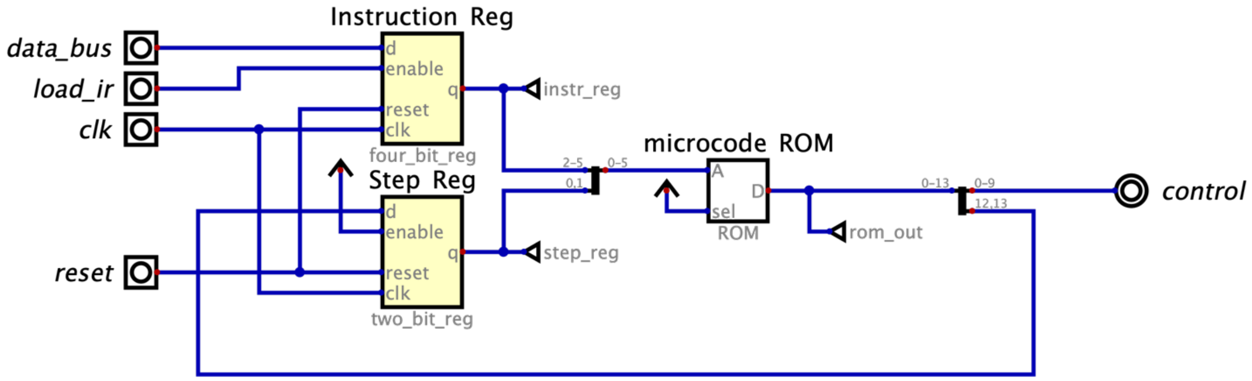
Did the circuit behave as expected? If no, what was wrong?

The circuit behaved as expected

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues

Include a picture of your controller circuit here:



Please comment on the single biggest issue you were facing when designing the circuit.

No issues

Did the circuit behave as expected? If no, what was wrong?

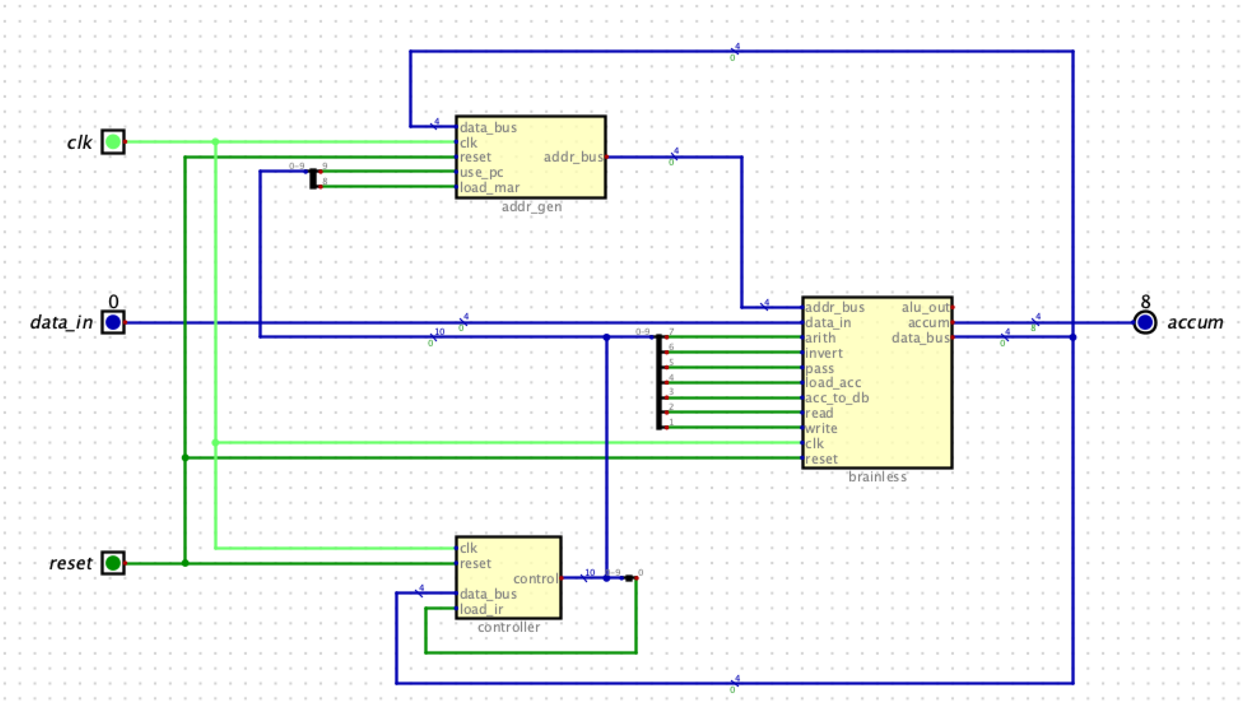
The circuit behaved as expected.

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues

## Task 4-3: Build the Complete Microprocessor Circuit

Include a picture of your Digital circuit here (make sure to show final values as shown in figure 17):



Please comment on the single biggest issue you were facing when designing the circuit.

No issues

Did the circuit behave as expected? If no, what was wrong?

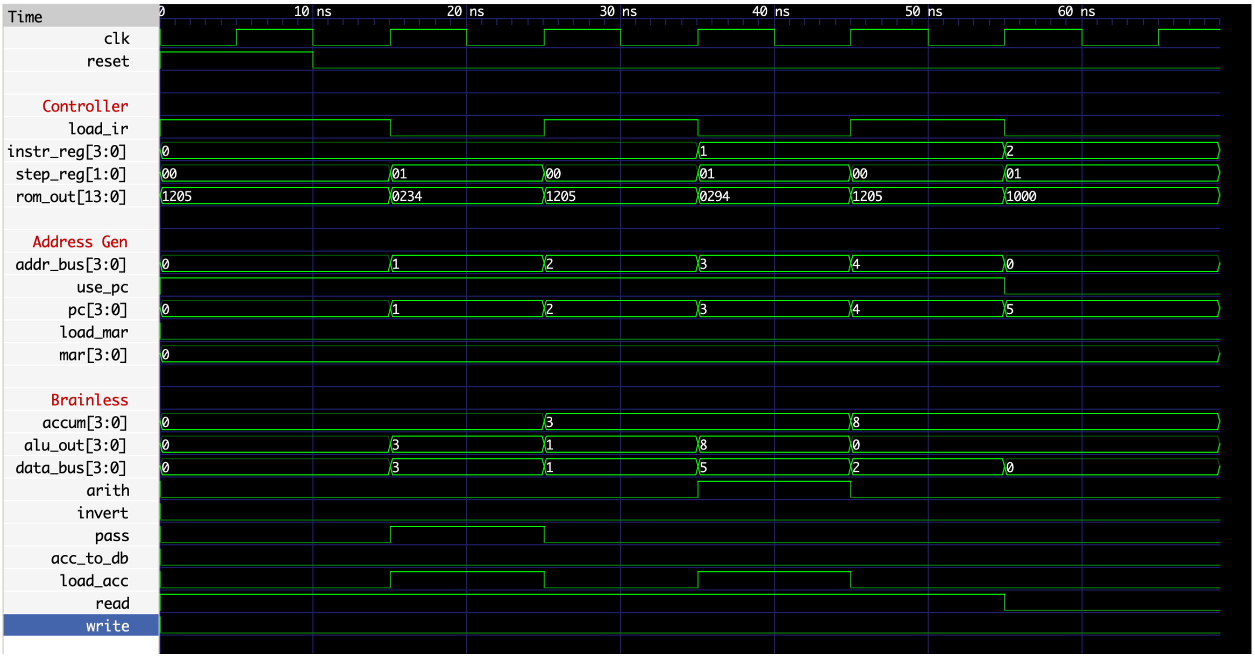
The circuit behaved as expected. The addr bus changed after every clock cycle and the accumulator has added the values that has been read from the ram.

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues but initially forgot to preload the ram_vals.hex file for this circuit.

## Task 4-4: Simulate the Design in Verilog

Include a picture of your waveforms here:



Please comment on the single biggest issue you were facing when simulating the processor.

No issues

Did the circuit behave as expected? If no, what was wrong?

The circuit behaved as expected and followed the exact procedure in the waveform.

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues

## Task 4-5: Add the AND, ZERO, SUB, and STORE ACC Instructions

Paste the contents of your final rom_vals here:

```
1205  # addr 00: LOAD step0 (fetch)
0234  # addr 01: LOAD step1
0000
0000
1205  # addr 04: ADD step0 (fetch)
0294  # addr 05: ADD step1
0000
0000
1205  # addr 08: STOP step0 (fetch)
1000  # addr 09: STOP step1 (loop)
0000
0000
1205  # addr 0C: AND step0 (fetch)
0214  # addr 0D: AND step1 => ACC = ACC & RAM
0000
0000
1205  # addr 10: ZERO step0 (fetch)
00D8  # addr 11: ZERO step1 => ACC = 0
0000
0000
```

```
1205  # addr 14: SUB step0 (fetch)

02D4  # addr 15: SUB step1 => ACC = ACC - RAM

0000

0000

1205  # addr 18: STORE step0 (fetch)

2304  # addr 19: STORE step1 => load MAR with operand

000A  # addr 1A: STORE step2 => write ACC to MAR

0000

3FFF  # final line to fill out Digital's ROM
```

Test your instructions by writing and executing programs. Paste the contents of your ram_vals.txt file for each program. Note which instruction or instructions each program tests.

**AND Instruction**

```
0     //  addr 0: opcode 0 (LOAD)

C     //  addr 1: operand => 0xC

3     //  addr 2: opcode 3 (AND)

5     //  addr 3: operand => 0x5

2     //  addr 4: opcode 2 (STOP)

0     //  addr 5-F: unused
```

**ZERO Instruction**

```
0     // addr 0: opcode 0 (LOAD)

6     // addr 1: operand => 0x6

4     // addr 2: opcode 4 (ZERO)

2     // addr 3: opcode 2 (STOP)

0     // addr 4-F: unused
```

**SUB Instruction**

0 // addr 0: opcode 0 (LOAD)

9 // addr 1: operand => 0x9

5 // addr 2: opcode 5 (SUB)

4 // addr 3: operand => 0x4

2 // addr 4: opcode 2 (STOP)

0 // addr 5-F: unused

**STORE Instruction**

0 // addr 0: opcode 0 (LOAD)

A // addr 1: operand => 0xA

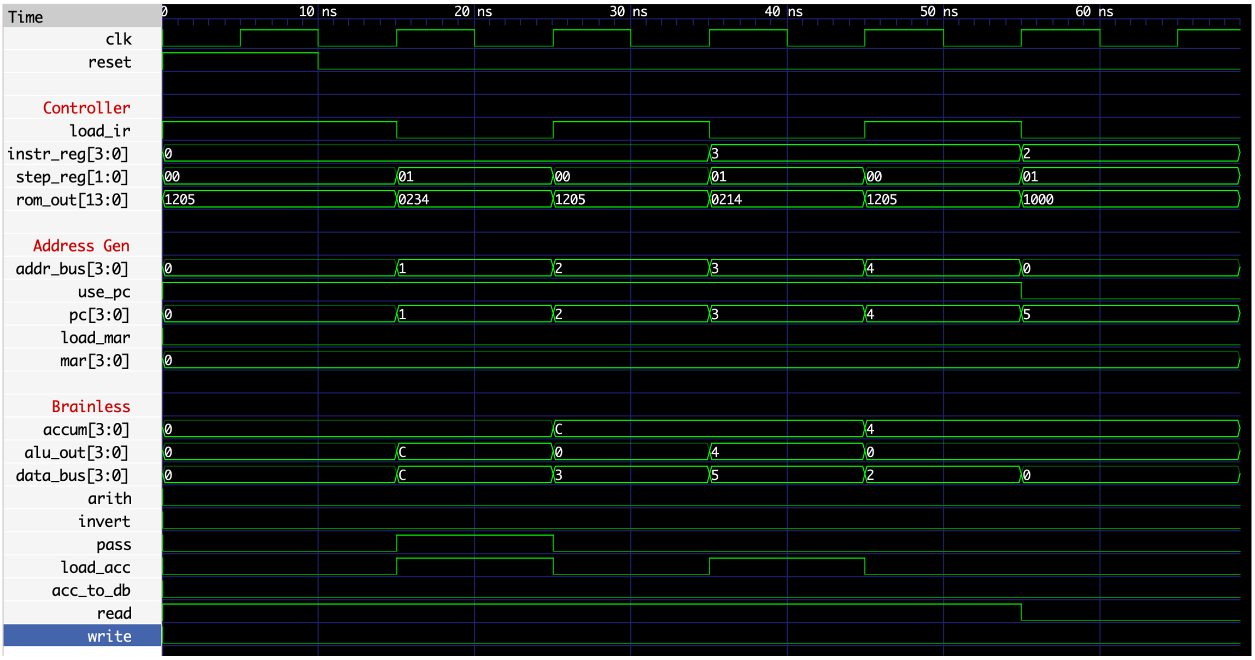
6 // addr 2: opcode 6 (STORE)

7 // addr 3: operand => address 0x7

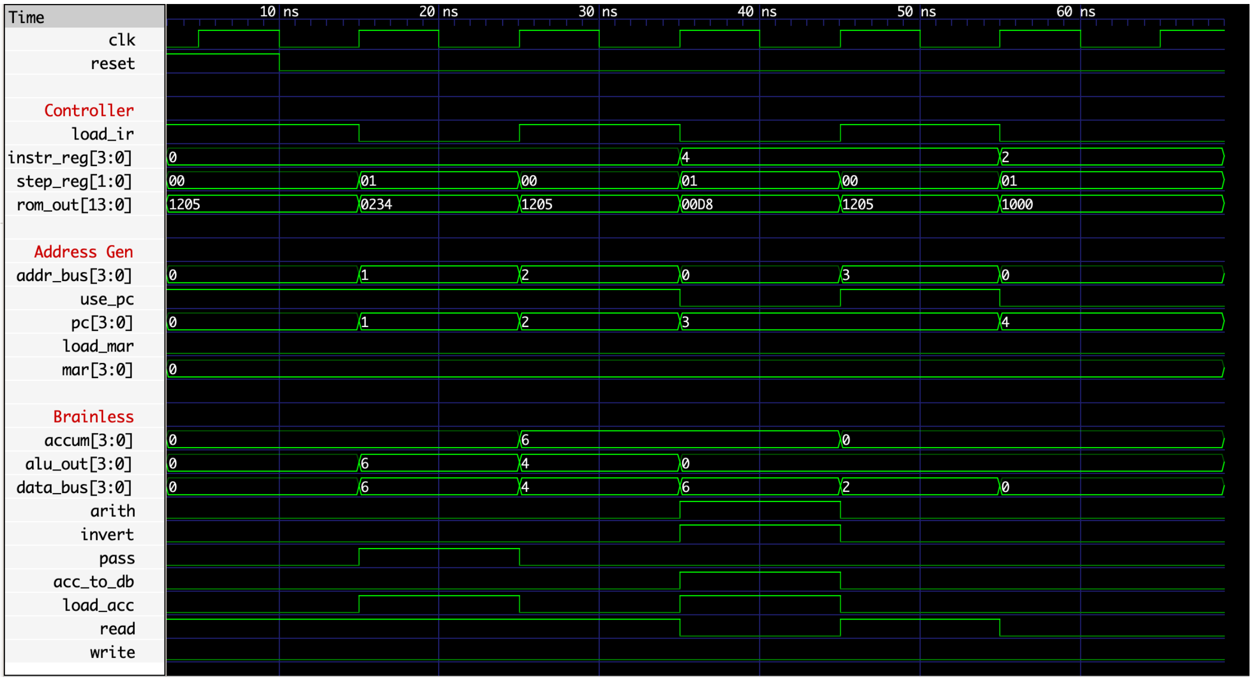
2 // addr 4: opcode 2 (STOP)

0 // addr 5-F: unused

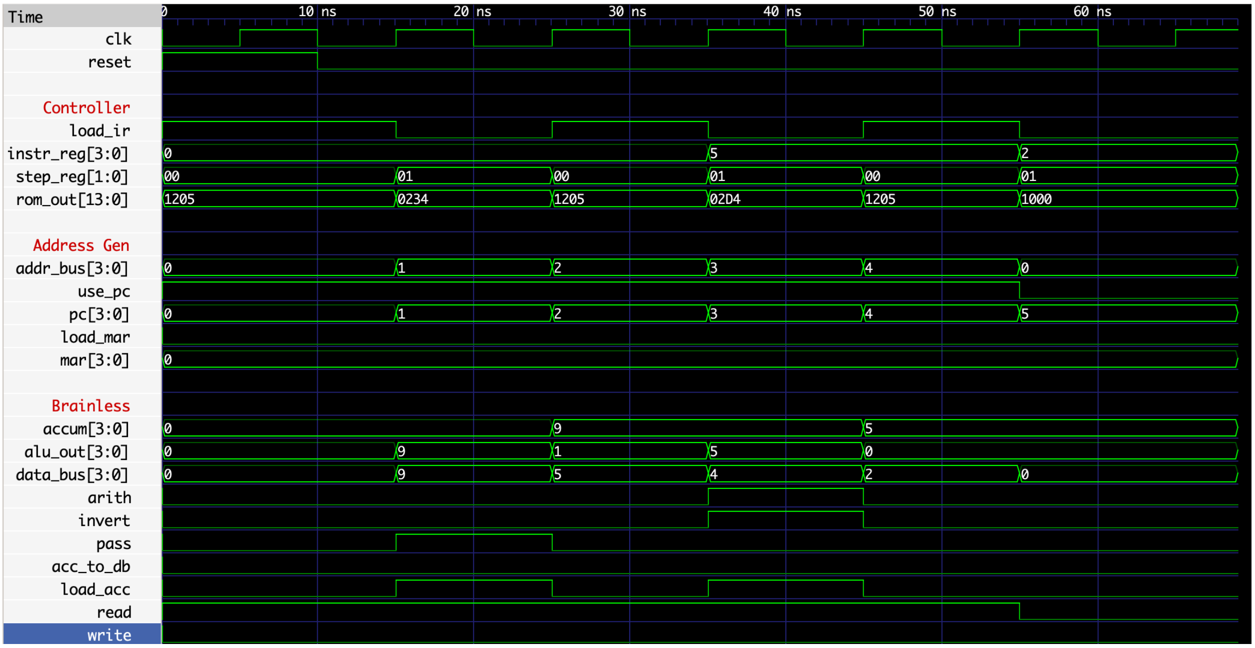
Include a picture of your AND waveforms here:



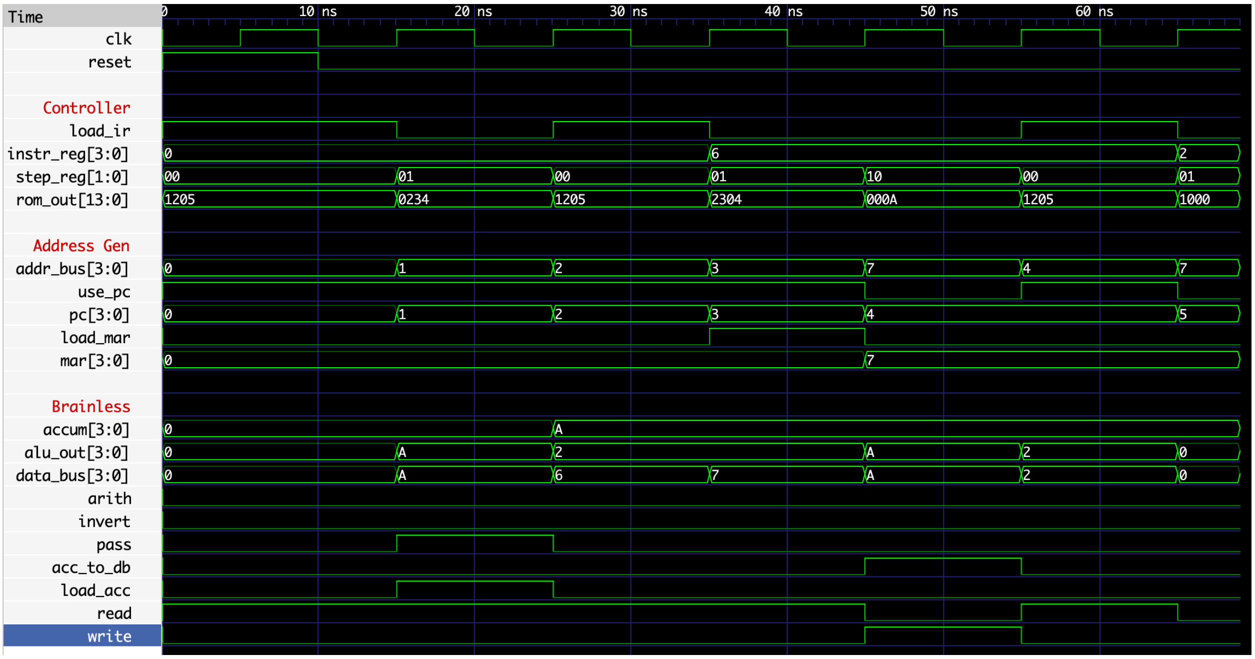
Include a picture of your ZERO waveforms here:



Include a picture of your SUB waveforms here:



Include a picture of your STORE ACC waveforms here:



Did the circuit behave as expected? If no, what was wrong?

The circuit behaved as expected

Please comment on the single biggest issue you were facing when simulating the circuit.

No issues

## Task 4-6: Invent Your Own Instruction (Extra Credit)

Place the contents of the rom_vals for the extra credit instruction here:

1205  # addr 0 : Step 0 (fetch opcode)

0234  # addr 1 : Step 1 (ACC <- [RAM])

0000  # addr 2 : unused

0000  # addr 3 : unused

1205  # addr 4 : Step 0 (fetch opcode)

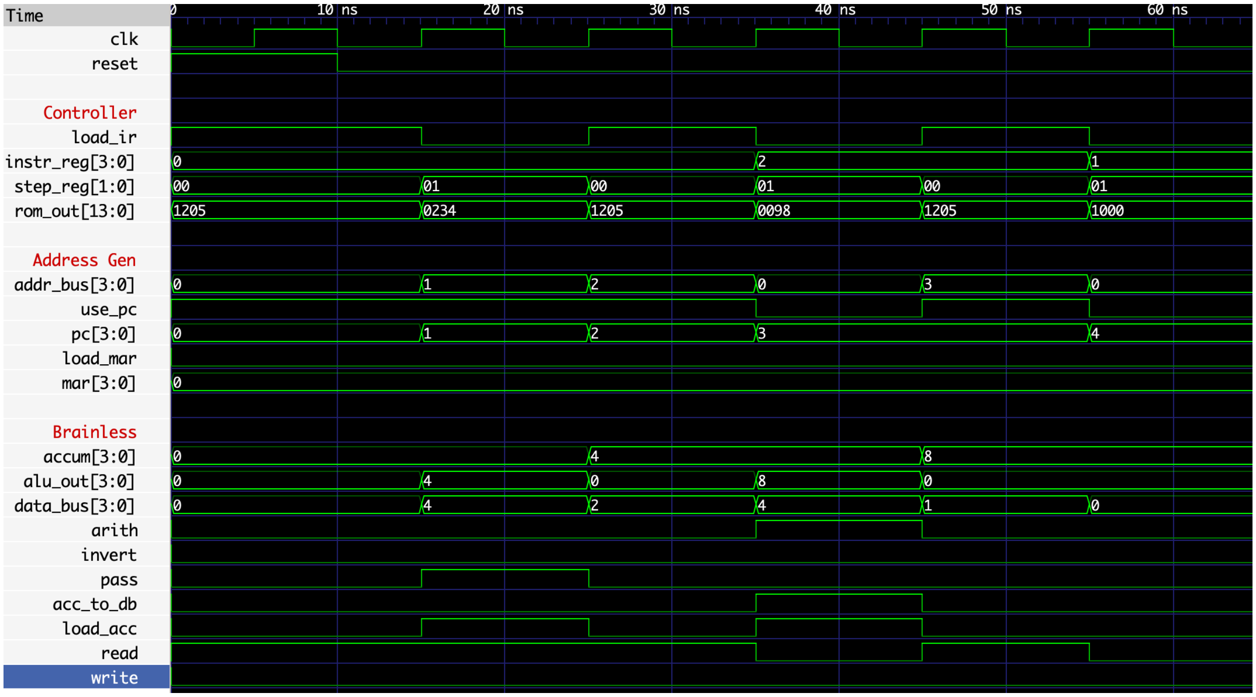
1000  # addr 5 : Step 1 (halt in place)

```
0000  # addr 6 : unused
0000  # addr 7 : unused
1205  # addr 8 : Step 0 (fetch opcode)
0098  # addr 9 : Step 1 (ACC <- ACC + ACC)
0000  # addr A : unused
0000  # addr B : unused
3FFF  # final line to ensure full ROM expansion
```

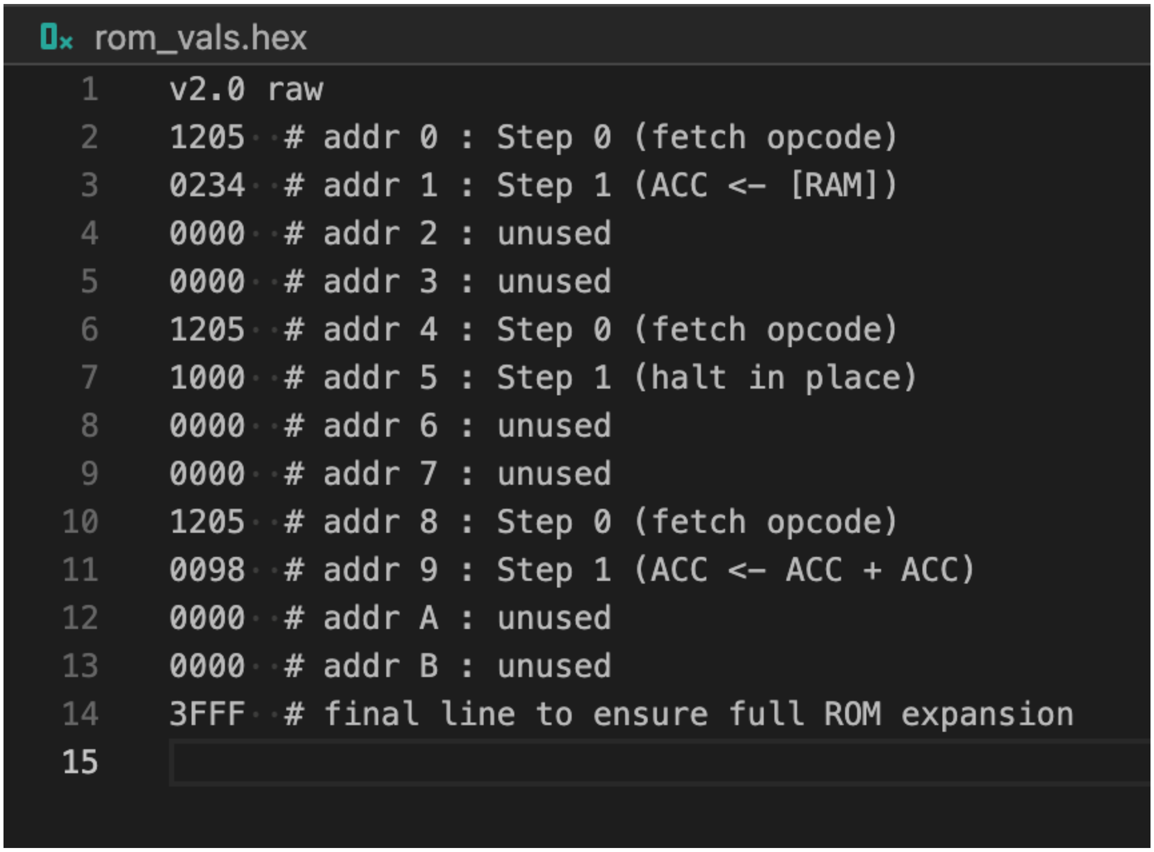
Include your Verilog ram_vals.text program used to test the extra credit instruction here:

```
0   // addr 0: opcode 0 (LOAD)
4   // addr 1: operand => load literal 4
2   // addr 2: opcode 7 (DOUBLE => ACC = 2×ACC)
1   // addr 3: opcode 2 (STOP => halt)
```

Include a picture of your waveforms here:



Include a picture of your ROM contents here:



```
v2.0 raw
1205  # addr 0 : Step 0 (fetch opcode)
0234  # addr 1 : Step 1 (ACC <- [RAM])
0000  # addr 2 : unused
0000  # addr 3 : unused
1205  # addr 4 : Step 0 (fetch opcode)
1000  # addr 5 : Step 1 (halt in place)
0000  # addr 6 : unused
0000  # addr 7 : unused
1205  # addr 8 : Step 0 (fetch opcode)
0098  # addr 9 : Step 1 (ACC <- ACC + ACC)
0000  # addr A : unused
0000  # addr B : unused
3FFF  # final line to ensure full ROM expansion
```

## Task 4-7: Create a video and submit your report

Record a short video showing your schematics in Digital and your waveforms in GTKWave. Be sure to show yourself in the video and show your screen. Explain how your circuit works – you need to convince the grader you did the lab and understand it! **Copy and paste the link to your video below. Make sure the link is working and pointing to the correct video. Remember to include the password if required. Do NOT upload your video to Canvas. It is recommended that you use Zoom to record to the cloud, pasting the link and password below.** If your circuit is not working as expected, explain in the video how it is not working and why you think it is not working.

**Video Link:** https://asu.zoom.us/rec/share/Y7Dai1BzXMdGVJTGfS4VwLdeyMJ99xQLE1w-S8GK0Q9vjD3a4FcYdG-8aB2UhRJS.yxxoudb5Aa_6ldNY?startTime=1744428310000

**Passcode: $PF@7L@8**

**At the beginning of your recording, say your name and the lab name. Be brief in your recording. Submit the completed template to Canvas.**

**Make sure all your files are in the Lab2 directory. Create a zip file of the Lab2 directory. Remember to turn in the zip file and your completed template on Canvas!**

**Do not include the video in the zip file! This makes the file very large and you run the risk of the zip file not uploading or taking so long to upload that your submission will be late. Remember that the submission is dated at the time the upload completes, not when it starts!**

# LAB 4: LAB REPORT GRADE SHEET

**Name:** ______________________________________

**NOTE: You submit the zip file in order to show your work.**
**If the zip file is not submitted you will receive a 0 for this lab!**

**Instructor Assessment**

| **Grading Criteria** | **Max Points** | **Points Lost** |
|---|---|---|
| **Description of Assigned Tasks, Work Performed & Outcomes Met** | | |
| Task 4-1: Build and Test the Memory-Address-Generation Circuit | 10 | |
| Task 4-2: Build and Test the Controller Circuit | 10 | |
| Task 4-3: Build the Complete Microprocessor Circuit | 10 | |
| Task 4-4: Write and Execute a Simple Program for Your Microprocessor in Simulation | 10 | |
| Task 4-5: Add the ‘AND’, ‘Zero’, ‘Subtract’, and ‘Store ACC’ Instructions | 30 | |
| Task 4-6: Invent Your Own Instruction (Extra Credit) | 10 | |
| Task 4-7: Record your video | 10 | |
| **Lab Score (80 points total)** | **Points Lost** | |
| | **Late Lab** | |
| | **Lab Score** | |