CU and Performance
Due: October 31, 2019 at 23:55 on MyCourses

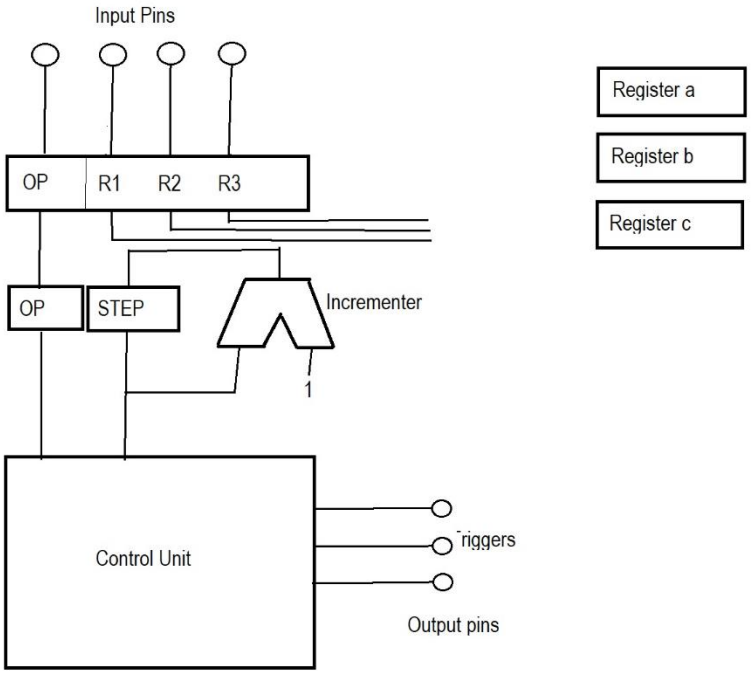
Tutorials E and F will be helpful for this assignment. Tutorial E covers control units. Tutorial F covers faults and performance. The assignment assumes you are familiar with LOGISIM.

QUESTION ONE: The CU in LOGISIM

Using LOGISIM built the CU chip set: The IR register, the OPCODE:STEP register, the STEP incrementor, the CU, and the trigger wires exiting the CU. In addition, create three registers: register a is 8 bits long, register b is 4 bits long, and register c is 4 bits long. These three registers will interact with the control unit.

You are permitted to use the following LOGISIM components: the **full**-adder, the register, the black box, the clock, the flip-flop, AND-gate, OR-gate, NOT-gate, XOR-gate, wires, input pins, and output pins.

Design your solution to follow this template:



Note: the CU box does not need to look like the above diagram but try to keep the general arrangement of the parts.

Imagine you have the following assembler instructions:

```
MOVE Rx, Ry        // copy the contents of Ry into Rx
MERGE Rx, Ry, Rz   // Rx.low = Rz and Rx.high = Ry, where: low=right-most nibble
```

The instruction MOVE uses OP:R1:R2 but not R3 from the IR. The instruction MERGE uses OP:R1:R2:R3 from the IR. MOVE can move any register to any other register. Since Ra is twice as big, only the lower 4 bits are copied or overwritten. MERGE can merge any register with any other register. It makes most sense that Rx would always be Ra, but your instruction must handle the case when Rx is not the Ra register. In this case, MERGE reduces to the MOVE command.

For example:

```
MOVE Ra, Rb        // copies the contents of Rb into the right-most 4-bits of Ra
MOVE Rb, Rc        // copies the contents if Rc into Rb
MOVE Rb, Ra        // copies the bits from Ra.low (right-hand 4-bits) into Rb
MERGE Ra, Rb, Rc   // Ra.low receives the bits from Rc, Ra.high from Rb
MERGE Rb, Ra, Rb   // Behaves like MOVE Rb, Ra.low
MERGE Rb, Rc, Rb   // Behaves like MOVE Rb, Rc
```

You will run the circuit by inputting the instruction by hand into the IR input pins and by setting the contents of the registers Ra, Rb, and Rc. Then, you will activate the clock to run the instruction. Your control unit will run its microcode to execute the instruction. You must figure out a way to stop the CU once the instruction has fully executed. The user will then be able to see the values in the Ra, Rb and Rc registers to validate the circuit.

Build the above circuit in LOGISIM.

QUESTION TWO: Performance Calculations

Answer the following questions:

A) Assume a classical CPU takes 4 clock ticks to execute an instruction that does not need to access RAM (called CPU instructions) and 20 clock ticks to execute instructions that do need to access RAM (called RAM instructions), how many clock ticks would it take to execute a program with 25 instructions, of which, 10 are CPU instructions and the remaining are RAM instructions?

B) Assume a pipeline CPU having 5-stages, where the first four stages execute CPU microcode and the fifth stage (the last stage) is an optional stage that accesses the RAM. Each CPU microcode stage takes 1 clock tick to execute and the RAM stage takes 20 clock ticks to execute. This means that CPU instructions complete their execution at the fourth stage, while RAM instructions need the additional fifth stage. When the fifth stage is invoked the pipeline is paused until the fifth stage completes. Assume you have the following algorithm: 8 CPU instructions, then 1 RAM instruction, then 2 CPU instructions, then 1 RAM instruction. How many clock ticks will the program need to execute completely?

C) Assume a pipeline CPU having 4-stages, where all the stages execute CPU microcode. There are no RAM instructions. Each stage takes 1 clock tick to complete. Assume you have a program with 10 instructions, however instruction 3 and 4 result in a fault that is detected when instruction 4 enters stage 1. There is an additional fault between instructions 7 and 8, which is detected when instruction 8 enters stage 3. Assume the CPU uses dump and null

instructions to clear the pipeline and reset itself. How many clock ticks will the program need to execute completely?

WHAT TO HAND IN

Hand in the following to MyCourses:

- The LOGISIM circuit file(s) that implement question 1. You can zip them together.
- A PDF document that answers the three problems from question 2.

HOW IT WILL BE GRADED

Question 1 — Total points = 20

IR	: 1 points
OP:STEP+Add	: 3 points
CU	: 3 points
Register logic	: 2 points
Registers	: 1 point
Stop execute	: 1 point
Specified format	: 1 points

Each question is graded proportionally compared with the official solution.

Question 2

Q2 a)	: 2 points
Q2 b)	: 3 points
Q2 c)	: 3 points

-5% per day late.
After 2 late days, the assignment is not accepted.