CMSC389E Project 4: Clock & Program Counter

Assigned **Monday November 1**

Due **Monday November 8**

1 Project At a Glance

Here's the deal- we've got some components left to create, but they're frankly just one big new circuit, mishmashed in with a bunch of concepts that you've already heard of. We've got decoders and adders galore, mixed with a healthy amount of logic gates in order to create something truly beautiful- a fully functionaly computer. Building all of that out, however, is very tedious. As such, we'll be assigning you a portion of the final project that will give you all the tools you need to (painstakingly) build out the entire computer by yourself, but will save you the pain for the sake of your packed student schedule.

TL;DR we've created this last project so that you may take some time to conceptually understand what's needed to make our computer tick, but the actual effort you'll put in will be a lot less than what's needed to actually build the entire computer.

2 Drake, where's the ROM?

In an effort to make the end of the semester a little easier for you all, we decided to create the ROM portion of this project for you. As you may recall from our previous lectures, we're going big-picture now and focusing on the larger components of our computer as a whole. That is, now that we've built the ALU, we need to think about ROM, RAM, Clock and PC. We've taken the liberty of building out the ROM for you in the starter world for this project, but in case you'd like to toy with it a little more, we've included an extra credit opportunity for it as well. More on that later- for now, let's just say that we've taken care of the ROM portion for you. You'll see it as soon as you load into the Project 6 starter world.

So, now that our ROM is done, let's recap. We've finished out the ROM, and we've also finished out the ALU.

This ROM is excellent- it does all that we need to do, as per the last few lectures in class. That is, it represents 389E Assembly using a decoder system in the following way:

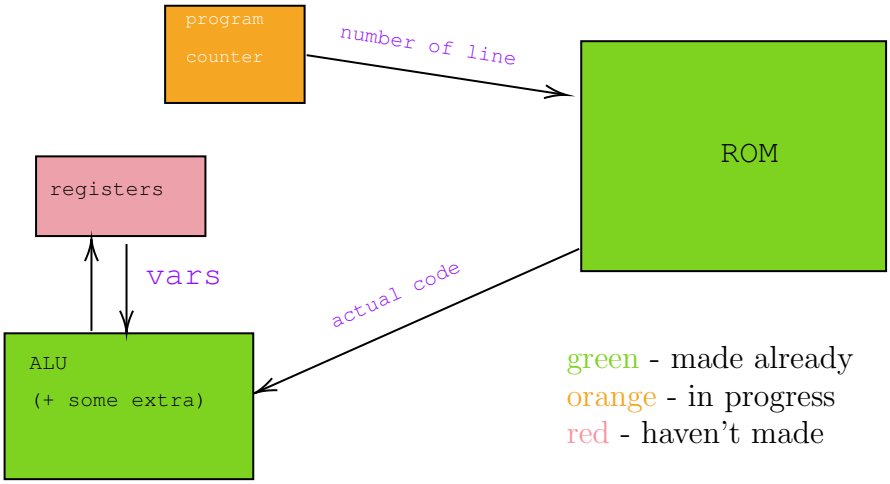
- The circuit receives a binary input dictating which 'line' of code to output.

- The circuit uses a decoder to activate the appropriate 'line' of code.
- The 'line' of code, which is just series of 14 blocks with/without torches is activated, projecting its output onto the output bus.
- The 'line' of code travels down the output bus and is ultimately interpreted by other circuitry.

The ALU, you already know about- so we won't talk about it here. Let's move on now to the stuff that's left to build- i.e. what remains for us to construct?

3 So, What's Left To Build?

The answer is pretty simple. Let's take a look at this diagram to see where our extensions to our ever-growing machine fit into our master plan to make a computer. Take a look at the diagram we've got below.



You'll pretty much just be building the program counter and clock, which will incrementally tell the ROM which line that it should send to the ALU. We haven't wired it up like that yet for the purpose of the project's simplicity, but let's suppose this has been done.

In simple terms, you will need to do a few things. First, create a full adder. This will be the basis of our creation. We will need a bit more than that- this adder needs to retain the state that it's in, and it needs to be able to increment by one. We can address the incrementing by one by simply wiring up the adder to add one every time that it's pulsed. We can address the retaining state portion by integrating the flip-flop design that we learned in class.

This project elegantly combines a 3 bit memory register (representing a program counter register), and a modified full adder, and a reliable three-bit output. It also integrates the clock that we addressed briefly in the last two classes.

4 Crafting is Believing

You have access to a video that goes step by step, building out this project block by block. We encourage you to watch it if you are lost- copying block by block is completely valid here. However, please ensure that you build this yourself, and recall our academic dishonesty policy.

Incrementing Full adder with Clock

Make sure that your design replicates the functionality demonstrated in the video, and you should be good to go. As this project can be done by pretty much following the video, we did not go in-depth with the circuit diagram here. However, if you're interested in information regarding the memory construct or the clock, please feel free to email us.

You will also be needing the **Project 4 starter world**, which you may find on the course website as a link.

5 25 % Extra Credit!

A special task for those of you with a discerning eye- as you may imagine, our Program Counter needs not only to increment by one, it also needs to do two more things. I recommend that you implement these capabilities one after the other, each of them are worth 12.5%

You will essentially be building out the **JUMP** functionality. That is, instead of incrementing by one every cycle, you'll be allowing the circuit to decrement if a control wire is on, and you'll be allowing the circuit to add/subtract by any quantity from 1 to 9 (you may disregard overflow).

- Subtract by one as well - implement the subtractor logic that you did in the previous project for this adder.
- Add/Subtract by any quantity (up to three bits). You will simply have to change what input is being fed into the adder here instead of 1. You may get creative with this solution, but the best way to go is to allow what input number to provide using levers.

6 Testing & Submission

Submission is a little different this time! There will be no test, as we discussed. Instead, you must send us a world that contains a self incrementing adder, as demonstrated in the video that we have provided you. It should be built reasonably close to spawn, and should function as expected. If it does not, please include some notes in your ELMS submission about what works and what doesn't. Failure to submit a working project AND then having a lack of explanation on ELMS will net you an automatic zero on this project.

After this, you can submit the project as you usually would- remember to attach the:

- `.zip` file containing your world
- Optional- a screen recoridng of your PC incrementing properly

Congrats- you've put together a fantastic model program counter and clock!