Team Project
Assigned: September 19, 2017
Progress Report Due: October 20th, 2017
Final Project Due: November 17th, 2017
Accepted late with no penalty until Friday, November 17th at End of Day
Accepted late with -20 point penalty on Monday, November 27th at End of Day

Objective:

Each cohort is to create a multi-state, digitally encoded processing unit of some form. Each system should have a minimum of four system states: Select, Calibrate, Start, and Stop. Each of these states should have at least four instructions. Combinational and Sequential Logic should both be covered in the project.

Format

The format should be double-spaced, in a 10-point font, in either Times New Roman or Ariel. The paper should have one-inch margins, and be full-justified.

Grading:

Progress Report	Draft Writeup & Member Tasks	10
Progress Report	Draft Software Research	5
Final Project	Updated Writeup & Member Tasks	5
Final Project	Updated Software Research	5
Final Project	System Design	25
Final Project	Verilog Program	25
Final Project	Verilog Output	25
Final Project	Bonus (Optional)	25
	Total:	125

Project Topics

How complex?

Create a Verilog system that creates a reasonable number of inputs and outputs.

How do you know? They would be from the *definitions*.

Think about the problem:
Four states (2 bits)
And usually four operations (2 bits)
And usually two or more multi-bit inputs. (2x8 Bits)
Outputs depend on the problem chosen, but usually include the current state, the current action, and a data result. (2 bits, 2 bits, 8 bits)
For inputs, that's 2^{20}, which is 1048576 table entries, That's 2^{12} possible outputs, or 4096 table entries.
That is too much.

Instead, include an example of each operation, of each state (and use case), and reasonable inputs. The resulting example should be about one page of size 10, Courier New Font. So, aim for around a minimum of 52 lines of output, and keep the maximum around 512 lines of output.

Example topics

The Classic: Create an ALU with full datapath

Add, multiply, divide, subtract two numbers, and remember the result to add the next number. An ALU can be turned on, can be turned off, can be reset, can be actively processing. Numbers should be a minimum of 8 bit integer. This is the classic example that is in the textbook. What is desired is a Verilog system that can operate as a calculator with a set of logic gates attached.

- Add two numbers
- Subtract two numbers
- Multiply two numbers
- Do an AND on two numbers
- Do an OR on two numbers
- Do a NOT on one number
- Do an XOR on two numbers
- The current value in the ALU must remain until cleared. A clear command must be included.

Build a Lightsaber Control Unit

How would a lightsaber work? It must have a certain number of states and values that control the lightsaber. While the emitter and Kyber Crystal is beyond the scope of this course, the logic that can control such a weapon is not.

- Be able to turn the lightsaber on or off.
- Set the color with three integer numbers 0-255 Red, 0-255 Green, 0-255 Blue for a full spectrum.
- Set the length of the blade with a floating point numbers. Remember to define the distance unit.
- Set the configuration of the blades with an integer register: single, double, or hilted.
- Set the power of the blade from training, to dueling, or to cutting through bulkheads.
- Have some form of integer counter and display the current power remaining in the lightsaber. This should decrease over time, and the rate of decrease should change with the power setting.

5.3 Your own device.

Should you desire to create your own device, it must be as complicated as all the other examples. It must have at least four states, and a defined set of features, inputs, and outputs.

The Dally Book has PONG as an example in Chapter 21. You cannot have Pong for a project.

The Progress Report (15 Points)

Title Page

Turn in title page with the name of the cohort, its members, and the name of the project.

Draft Description and Member Tasks (10 points)

Every circuit has a reason for its usage. Some are very serious, such as biomedical tools and industrial development. Some are very silly, used for advertisements and games. Ever open up a piece of junk mail to find a small circuit board and battery? Write a one -page description of the problem or use of the circuit. The writeup must include a description of why and how the circuit will be used and needed

Turn in a Member Tasks page, listing out what each member of the cohort is doing for the project. This ideally should be in the form of a table listing out each member, or subgroup of members, and what they are doing.

Draft Software Discovery (5 Points)

Our textbook refers to circuit development software and drawing software. These include Synopsis, Logisim, Visio, and even PowerPoint or MS Paint. Search the internet or campus resources to shop around for a freeware tool. Open source or closed source is fine. Ideally, such software would have an easy-to-create circuit drawing tool, and a simplification tool as well. Write a one page report on why this piece of software was chosen, including ease of use, availability, and cite it. Include a second page with screen captures of the tool being used to create the project control circuit.

The Final Project

Part 1: Updated Writeup and Member Tasks (5 points)

Update the write up and member tasks from the progress report. Anything removed, mark with a ~~strike-through~~. Anything added should be marked in **bold blue**. Anything that remains the same should be in black.

Part 2: Updated Software Research (5 points)

Update the software research from the progress report. Anything removed, mark with a ~~strike-through~~. Anything added should be marked in **bold blue**. Anything that remains the same should be in black. This includes any pictures or screen captures.

Part 3: System design: Diagrams, tables, write-up (25 points)

All the various components covered in class may be used for the creation of the system, everything from digital encoding, combinational components, and up through finite state automata using sequential circuits. Such a design should include Descriptions and Diagrams.

For Descriptions:

- A parts list.
- A list and *definition* of all the inputs.
- A list and *definition* of all the outputs.
- A list and *definition* of all the features and operations performed
- A list and *definition* of all the modes and states.
- A description of the register, registers, or other components that control the *current state*.

For Diagrams:

- The main circuit diagram for the project.
- A state-machine as needed.

Part 4: Verilog Simulator with Output (50 points)

After completing the design, the Verilog can be used to create the various components to run the system. The Verilog modules need to match the parts list in your specification. With the Verilog, the system can be tried for every possible combination, even setting error states. Given that Verilog is a language, and can create enormous truth tables very quickly, the system is expected to be of a suitable complexity.

The Verilog code should have:

- A Module to match each component in the design
- Modules that combine components into the circuits
- A testbench that shows the key features, not all features
- A display to the screen that shows
- Each Input with a label
- Each operation with a label
- State information with a label
- Each output with a label.

Something like:

```
Num 1          Num 2         Operation    Current Output          Next State
10011001 (153) 10101010(341) 00 (Add)     Running 111101110 (494) Running
10011001 (153) 00000000(  0) 11 (Divide)  Running XXXXXXXXX (Nan) ERROR
…
```

Bonus: Hardware or GUI

If your cohort is willing to physically create the circuit, or to create a fully interactive GUI and is willing to show it off to the class…..…Dr. Becker will award and extra 25 points bonus on the project. No, you may not have supplies from the CS 4141 Laboratory for the project.