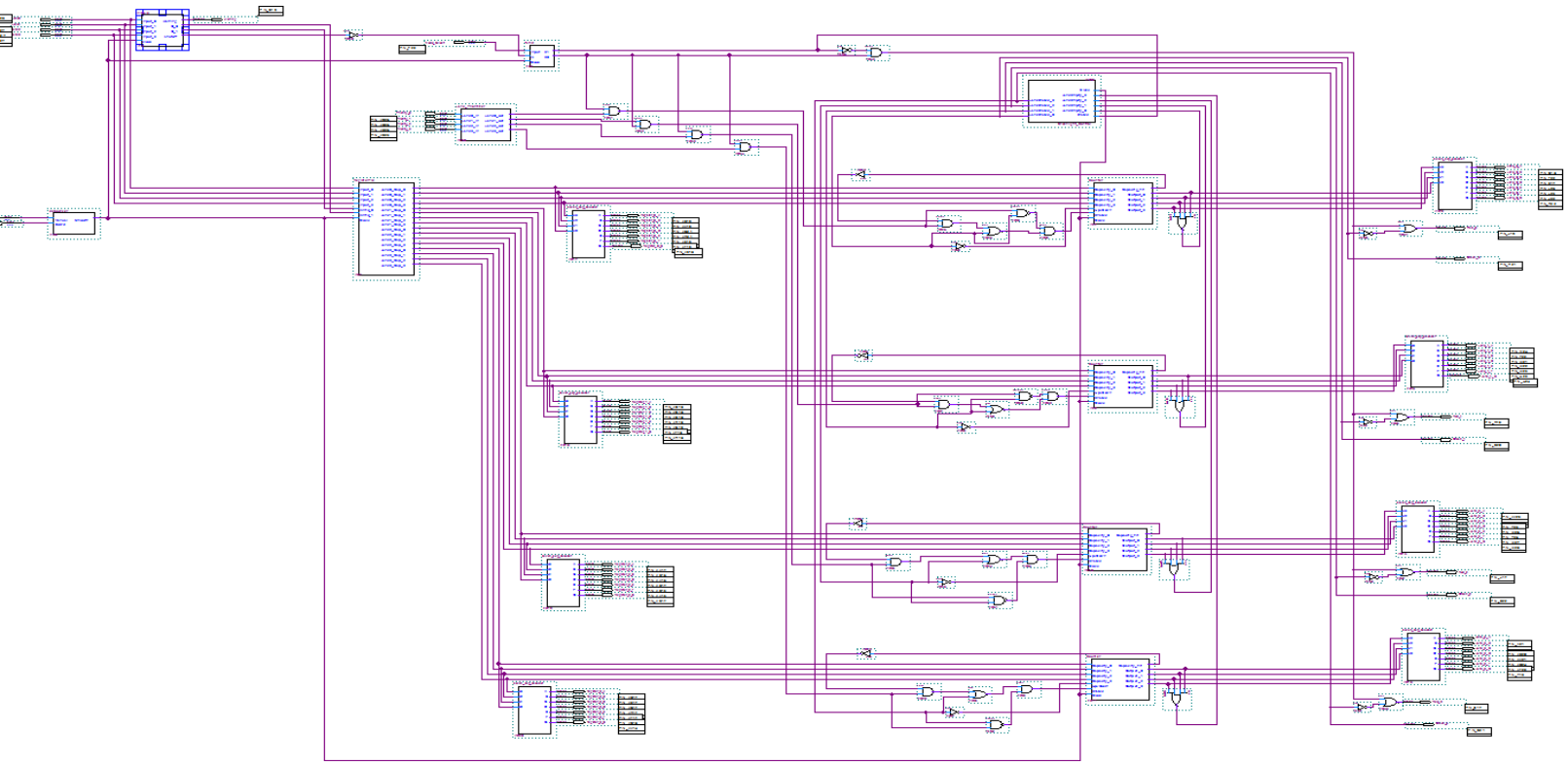
Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

Final Project Report

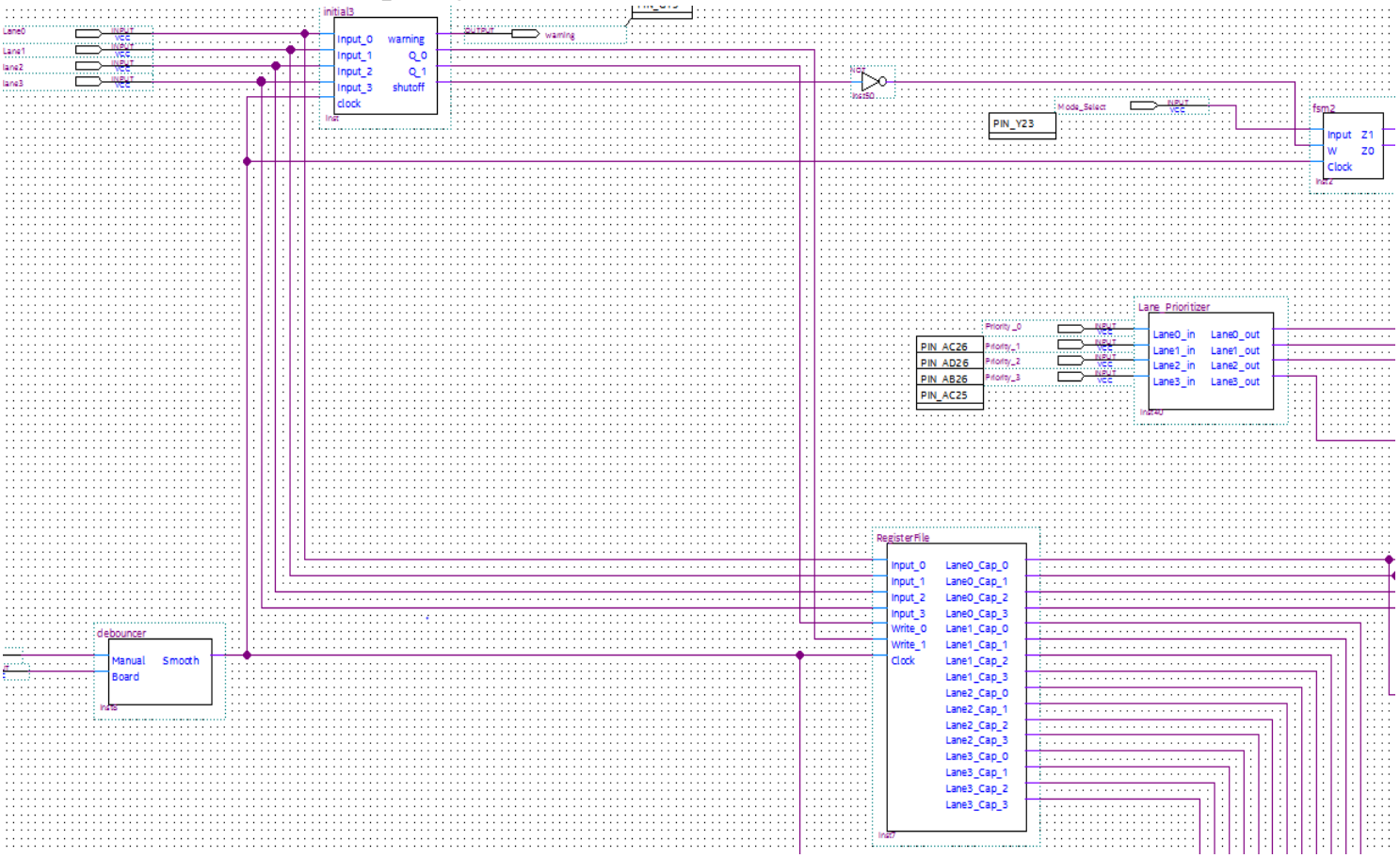
I completed project option number 3 (Traffic Light Fairness System). This report details all the workings and derivations required to build the circuit. The whole circuit, excluding the seven segment display, were built using block diagrams. I will detail all the inner workings of each module except the basic building block circuits, like decoders and multiplexers, that we covered in class.



This is the overall top-level diagram. Since it is difficult to view from this size I will split the projects into its major components: Initial lane capacity control, lane addition and subtraction, lane safeguards, and display. I will briefly go over how each system works before I go into depth about each of the blocks function.

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

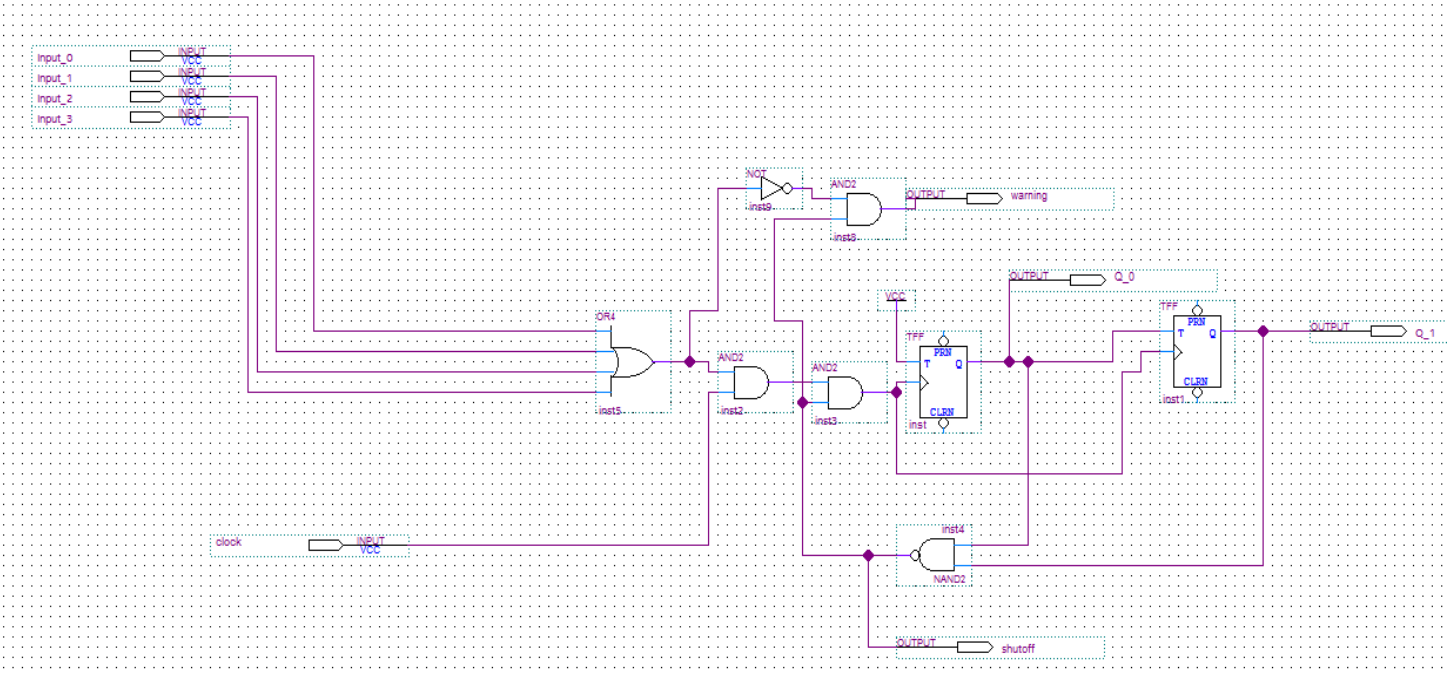
Initial Control Capacity:



This section The inputs are Lane_0, Lane_1, Lane_2, Lane_3, and Clock. The main blocks for this system are the register file, initial 3 and fsm2. For this initial loading phase, Lane_0, Lane_1, Lane_2, Lane_3 will be loaded into the registers if they have a nonzero value. After they are all loaded in a signal is sent to the finite state machine.

Initial3:

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

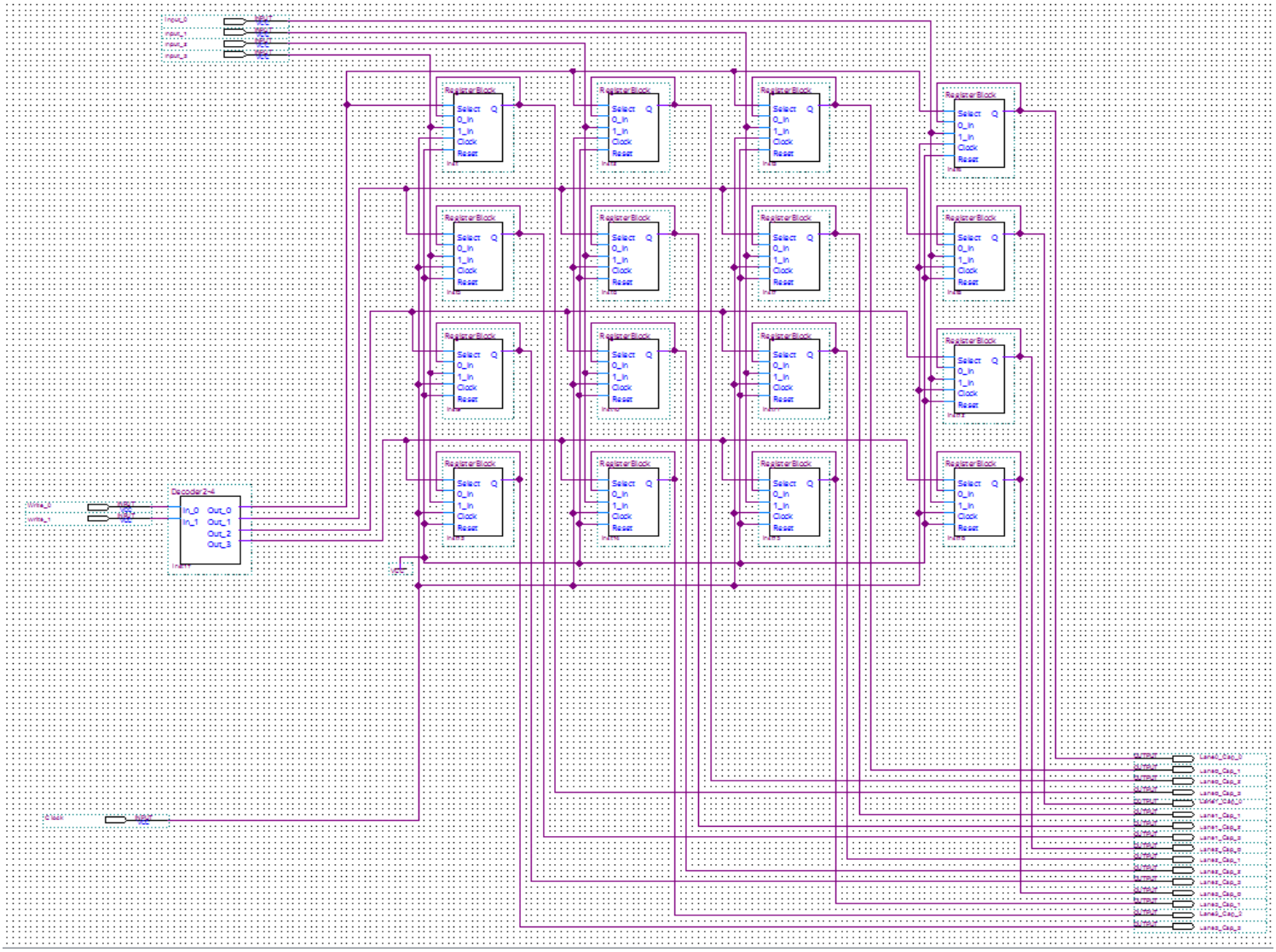


This Circuit gave me the most grief the out of the entire project. This is my third iteration of making it. I had troubles not because the circuit wouldn't work, but because how Quartus handled it. I would have it working before I made it into a block file. But once it was made into a block file it would behave differently in my final project.

The way the circuit operates is that it will count up from 00 to 11 if the 4-bit binary input is nonzero. The counter value goes to the register file and chooses what register is being written too. After the value 11 is reached the circuit is disabled and sends a zero to the finite state machine. After this circuit reaches 11 and is disabled, you will have four nonzero values stored in your register file.

Register file:

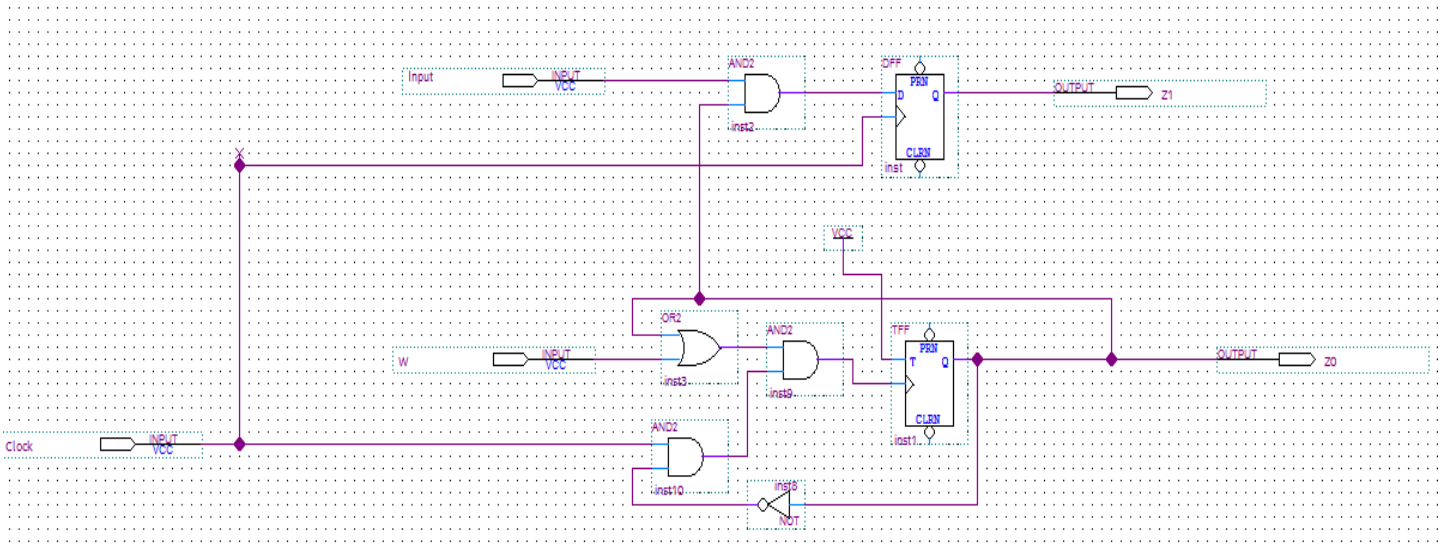
Brandon Johnson
Section M
CPRE 281
Student ID: 727079388



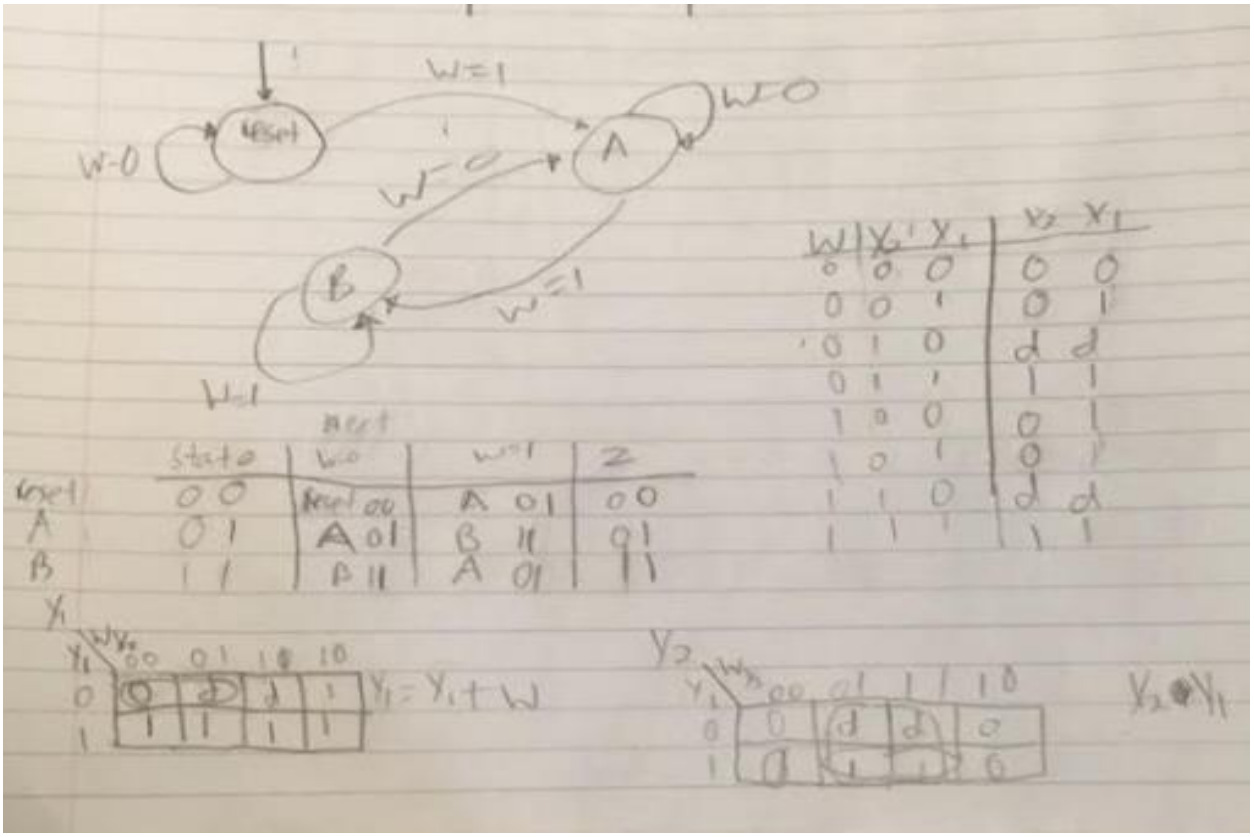
This circuit was copied straight from the notes. The modifications that I made was to remove the two read line and allow all 4 4-bit registers to be viewed simultaneously. This makes it easier to feed the capacity values straight to the counters. You can also display each register value simultaneously with this modification.

Finite State Machine:

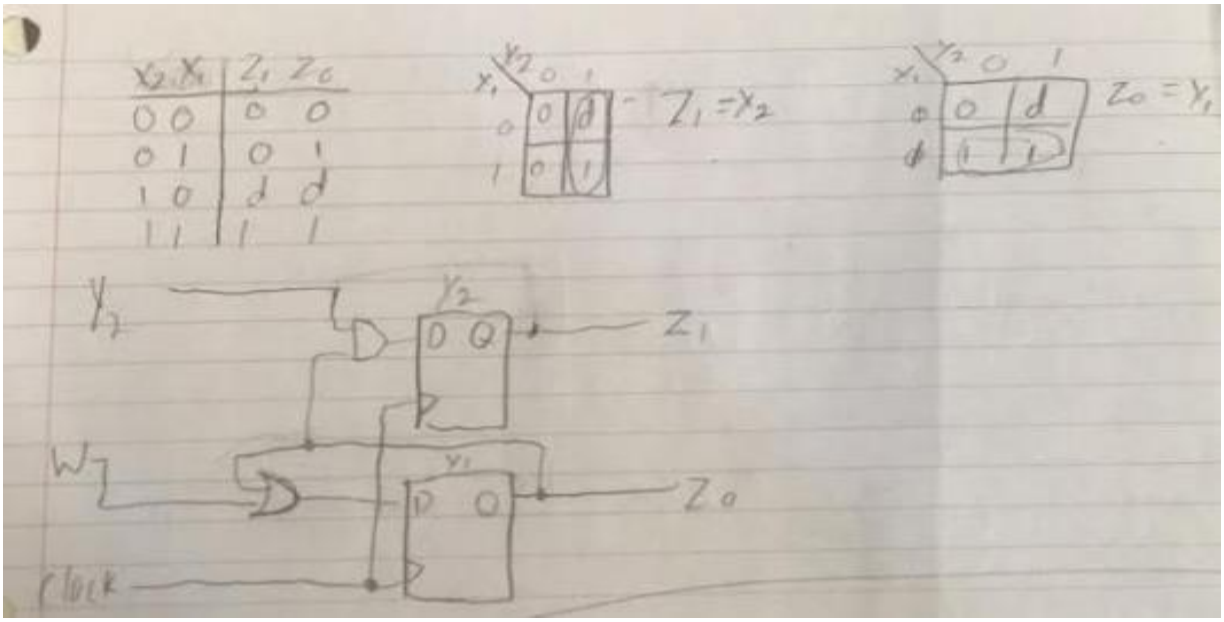
Brandon Johnson
Section M
CPRE 281
Student ID: 727079388



The finite state machine above does not have the same circuit diagram as the one I derived, but it has the same function. The reason it is built different is that my dff in the diagram would initialize to Q=1 instead of Q=0.

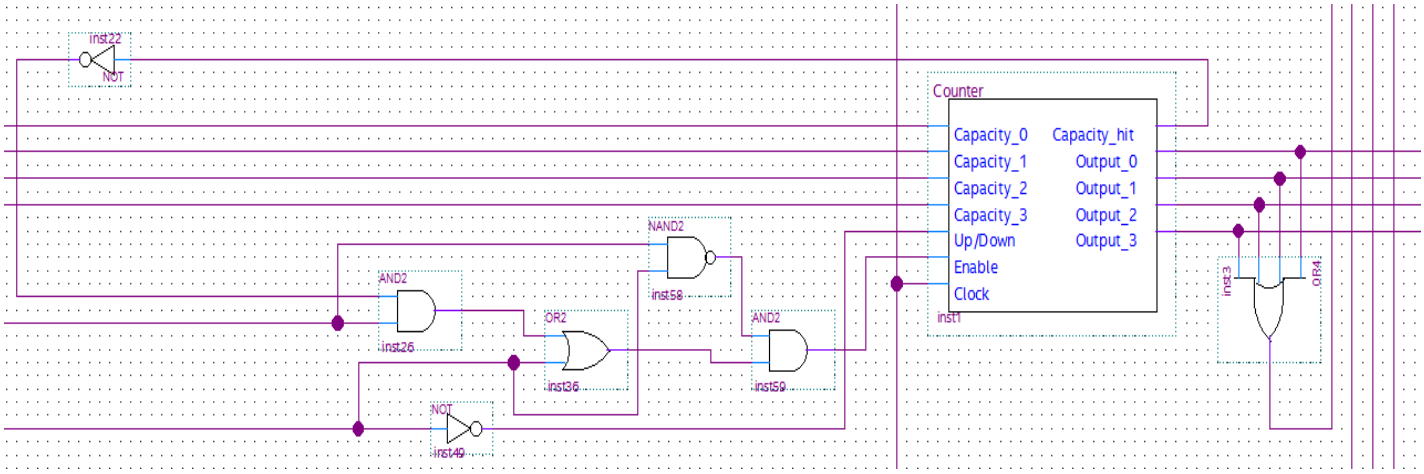


Brandon Johnson
Section M
CPRE 281
Student ID: 727079388



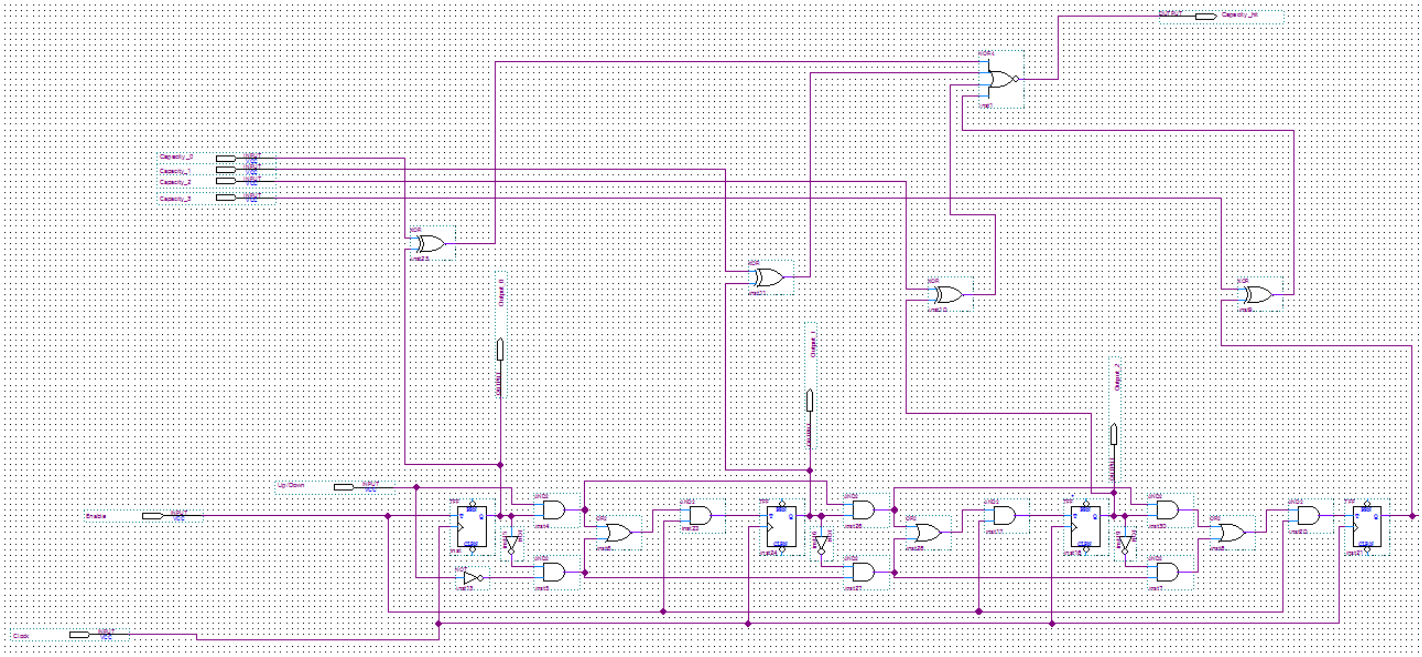
The state machine functions like this: Z1 and Z0 stay zero until W=1. When W=1, the Output Z0 will always stay 1 and Z1 will be whatever the second input is. This translates to that my circuit will stay in the loading phase until a 1 is received. Once a 1 is received you will never go back to the loading phase and you will always be in either state A or B. the W=1 is received by the shutoff output from initial3. Shutoff is active once you have 11 in both Q0 and Q1. The Z0 = 1 goes to enable input of each counter.

Counters and Lane Safegaurds:



Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

The counter above is identical to the other 3 counters that this project requires. The inputs are the 4-bit capacity values that were loaded into the register files from the previous state, Up/Down that causes the circuit to count up if its value = 1 and count down if its value = 0. Enable when given a 0 causes the counter to hold its current value, and clock. The Outputs are the 4-bit counter values and capacity_hit. Capacity hit returns 1 if the output is the same as the input. There is also an or gate to detect if the counter is empty. If a counter is empty a zero is sent to the Greenlight control unit which tells it to not subtract from this laneThe circuit diagram of the counter is below.

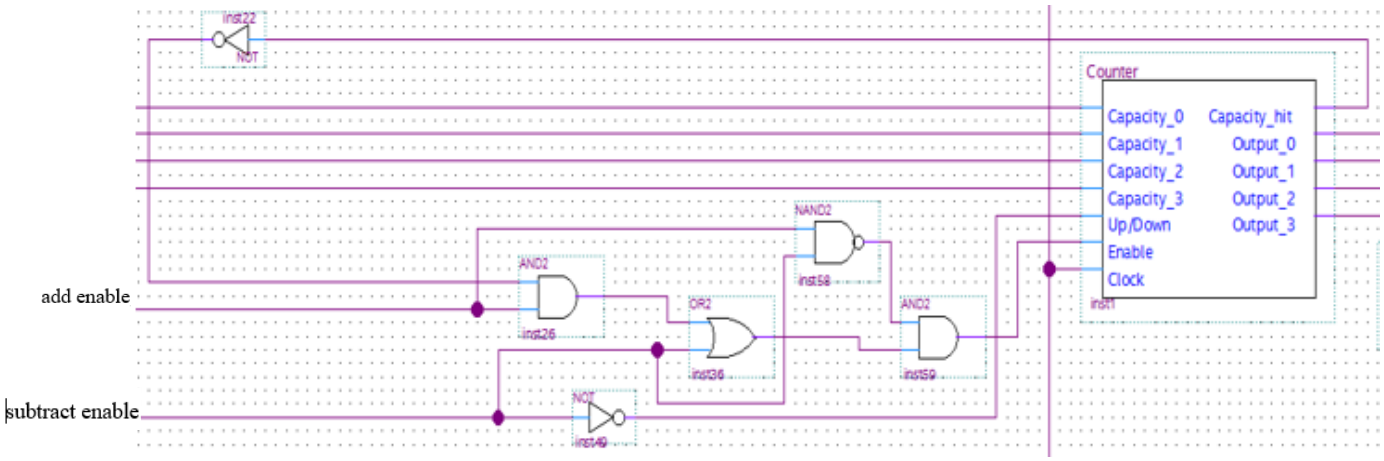


The Enable was implemented through the extra and gate between each of the t-flip flops. The way the circuit determines if the capacity is reached is by looking at the xor of the current counter values and the capacity values, if they are the same the output of the xor is 0, if they aren't the output is 1. The nor gate returns 1 if all of the xor values return 0.

Next we have the safeguards there are three inputs. Capacity_hit, add enable, and

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

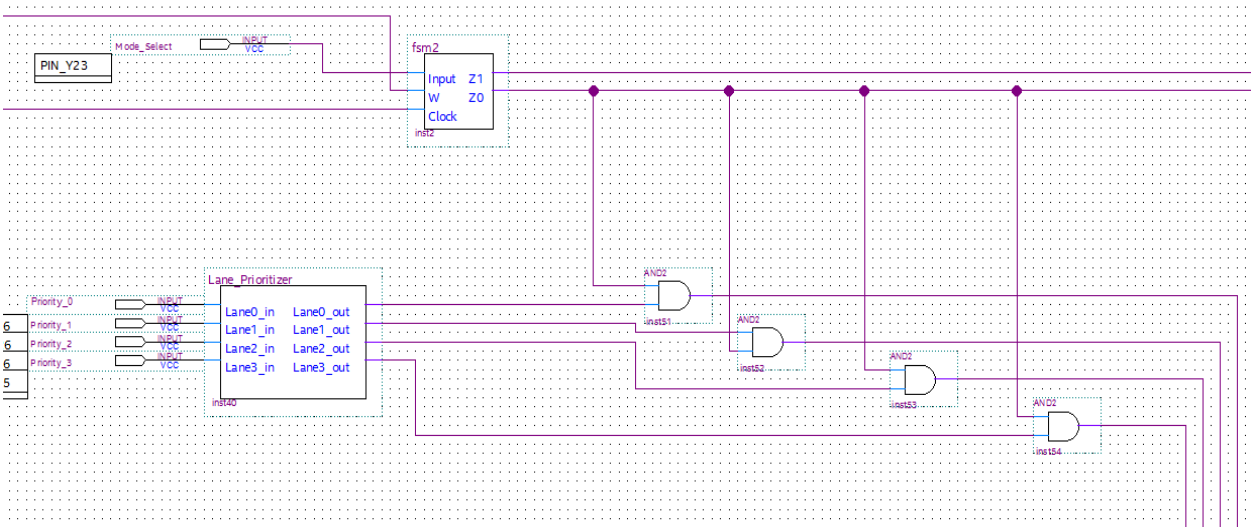
subtract enable.



This circuit was derived logically. When capacity hit is 1 it forces enable to be zero. This locks in the value in the counter until you subtract a car from the lane. When subtract enable is 1 the or gate is active causing enable to be 1 and Up/Down to be 0. When add enable is 1 enable is 1 until until capacity hit = 1. Finally, when subtract enable and add enable are on, enable is off, this allows a zero change when trying to add and subtract.

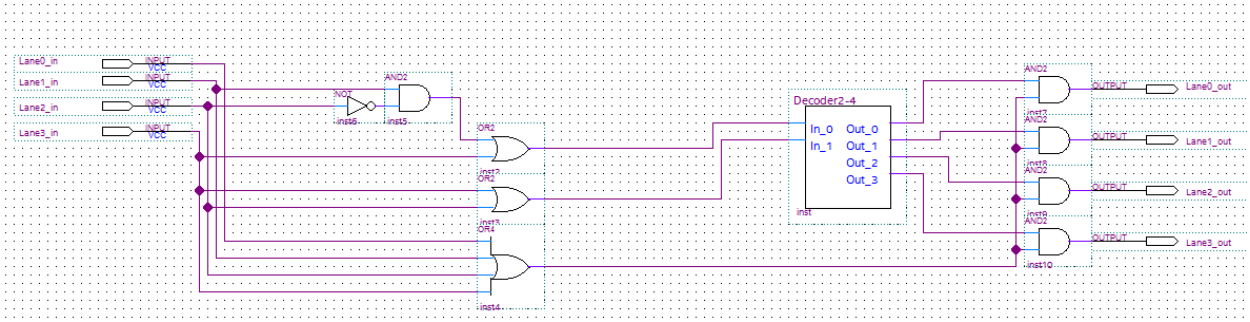
Addition/Subtraction:

This circuit enables 1 lane for addition based on the 4 toggle switch inputs.

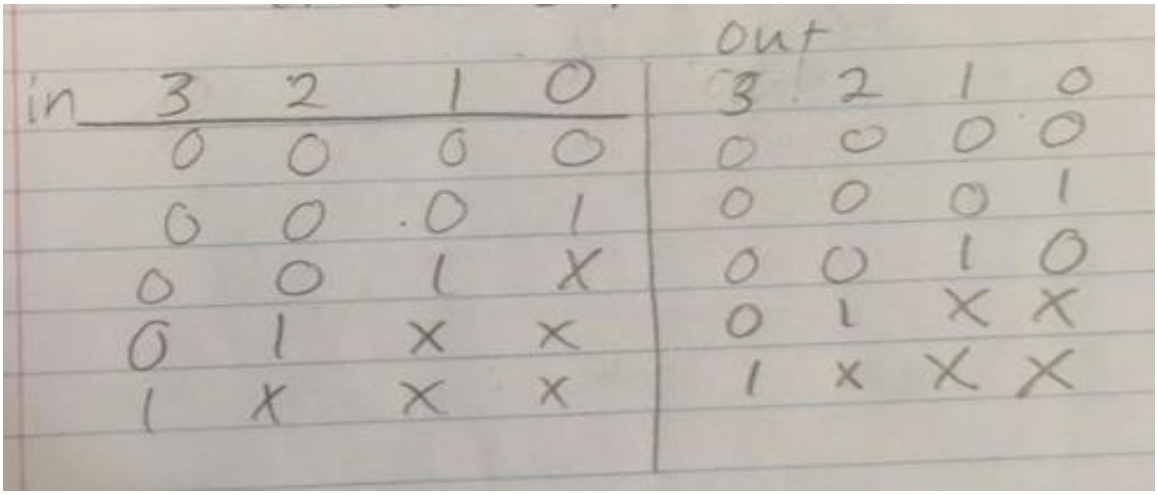


Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

This circuit can only be enabled when you are you done with the loading phase which makes Z0 = 1. It is noted that the output of the lane prioritizer is one hot encoded and when a lanes value is 1, it means you will try to add a car to a lane on the next clock cycle. The inside of the lane Prioritizer is shown below.



This circuit is a 4-2 priority encoder attached to a 2-4 decoder. The output is one hot encoded and the truth table is shown below.

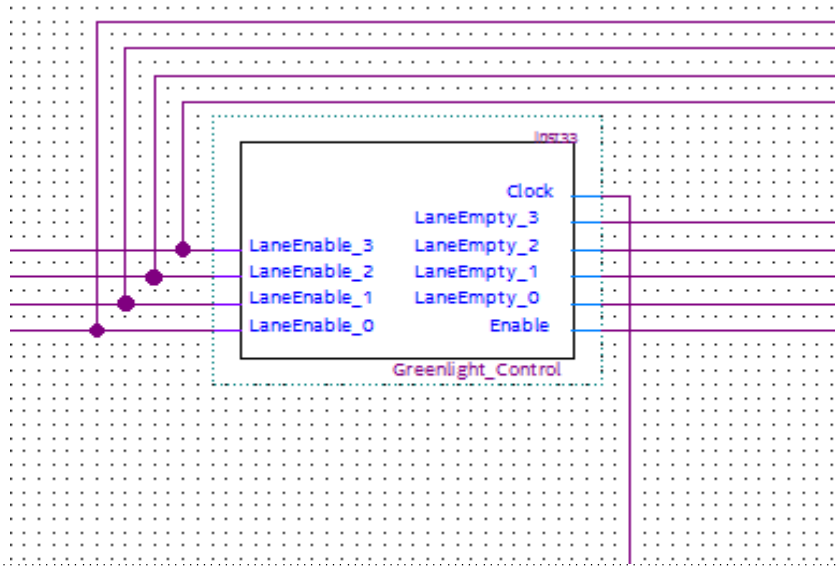


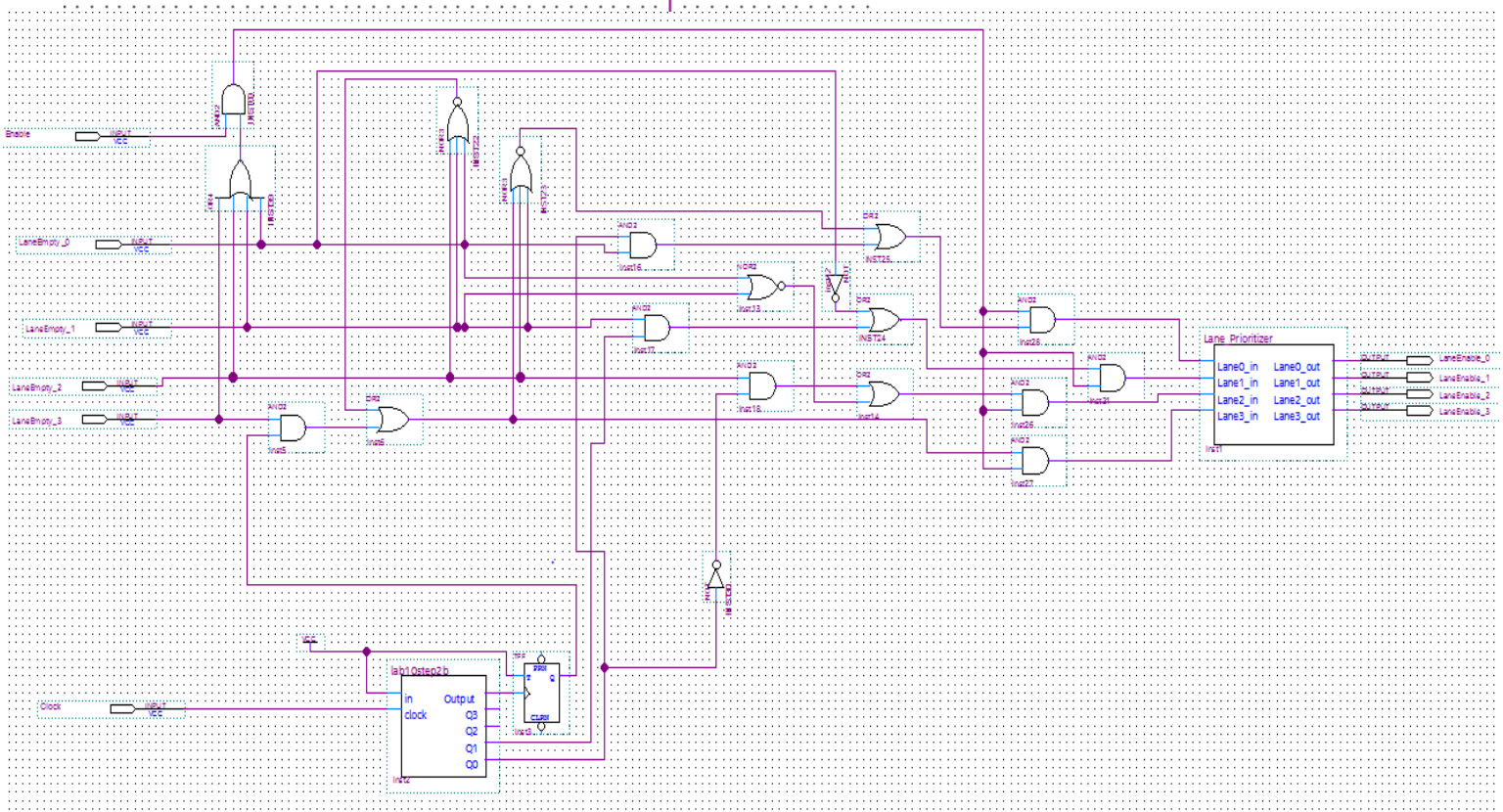
This shows that lane_3 has the most priority while lane_1 has the least priority. Also if no toggle switches are enabled, then there won't be any additions on the next clock cycle.

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

Subtraction Control:

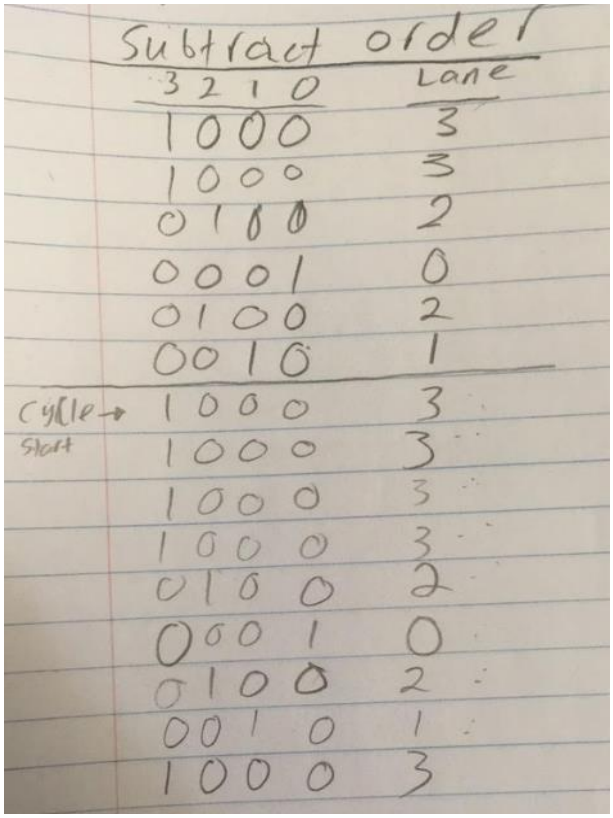
The inputs, LaneEmpty_1 through 3, to this circuit are 0 if the corresponding lane is empty. It is a check to see if a lane has is empty and should not be subtracted from. The outputs are one-hot encoded to insure that you only subtract from one lane at a time. If enable is zero all the outputs are zero. Enable receives its value from the finite state machine and is only on if you aren't in the loading phase and set the toggle switch to 1. The outputs of the





Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

The main attraction of this circuit is the Lane Prioritizer that we just talked about. This means we will have a one hot encoded output and that lanes with higher priority will turn on if they have a more significant bit. The inputs are all connected to an and gate which turns them off if they are empty If all lanes are empty then There will not be an Output. Next, we have a modulo 5 counter, called lab10stepb, which toggles an and gate for lane 3 every 4 clock cycles. This Means lane 3 will be chosen every 4 clock cycles if it has a car in it. Next when lane 3 is toggled off, lanes 2, 1, 0 are chosen from the counter as well. Lane 3 is Enabled when ~Q0 = 1. Lane 2 is Enabled when Q1=1 and ~Q0 = 0. Finally lane 1 is Enabled when Q0 = 1, Q1 = 0 and ~Q0 = 0. There are also measures that make a lane turn on and stay on if it is the only lane with cars in it. This is done by a or gate placed in front of all the other gates and is one if all other lanes = 0. If all lanes have cars in them a subtraction cycle is listed below.

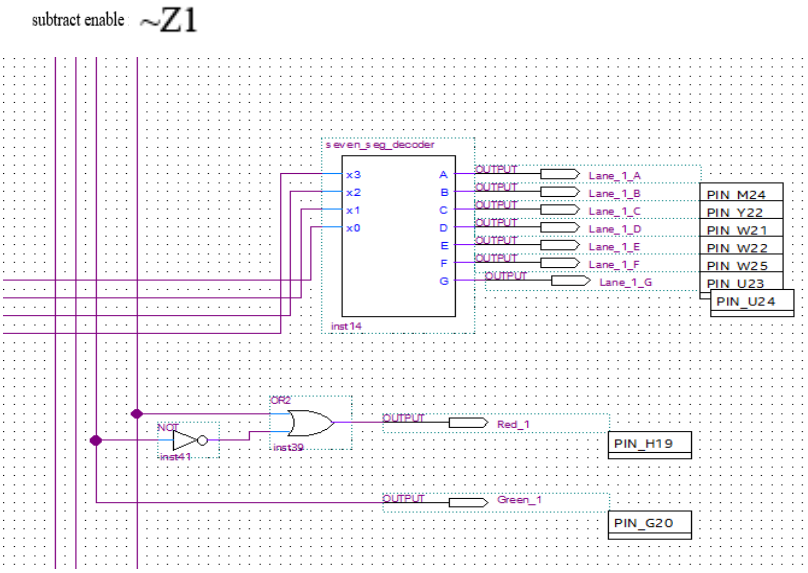


Subtract order	
3 2 1 0	Lane
1 0 0 0	3
1 0 0 0	3
0 1 0 0	2
0 0 0 1	0
0 1 0 0	2
0 0 1 0	1
1 0 0 0	3
1 0 0 0	3
1 0 0 0	3
1 0 0 0	3
0 1 0 0	2
0 0 0 1	0
0 1 0 0	2
0 0 1 0	1
1 0 0 0	3

cycle → start

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

Display:



I have 8 seven segment decoders for this project. 4 of them show the capacity values from the registers and 4 of them show the counter values. I also have to control the display the control for the red and green lights. The two inputs the control my lights are ~Z1 from the finite state machine and subtract enable from the Green light control module. If Z1=0 then all the Lights turn on and become red. If Z1=1 All the lights will stay on except for the one lane that is being subtracted from. In that case it will have a green Light.

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

List of Components and run through:

All the components with the submodules the are made up of.

Initial Control:

- 2-bit up counter with enable

Register File:

- 4 4-bit parallel access registers
- 2-4 decoder

Finite State machine:

- D and T-Flip flops

4 Up/Down Counters

- T-Flip flops

Lane Prioritizer (add enable)

- 4-2 priority encoder with enable
- 2-4 decoder

Greenlight Control (subtract enable)

- Lane Prioritizer.
- Modulo 5 Counter

4 Lane SafeGaurds

Brandon Johnson
Section M
CPRE 281
Student ID: 727079388

8 seven segment decoders

A sample run through of my circuit. Circuit initializes, and you can begin loading the capacity for each lane. If the value is nonzero you move to the next lane. Adding and subtracting does not do anything to the counter values in this stage. Once all 4 capacities are set, you will either be in the adding or subtracting phase depending on what your toggle switch is. You will want to be in the adding phase because your counters will have 0 cars in it. You can add cars too the lane of your choice. You can keep adding cars to a new lane if it is enabled by the toggle switches or it is not reached capacity. Once at capacity you can no longer add cars to that lane. When in subtract mode the greenlight counter will active a greenlight on a lane if it has a nonzero value. A lane will not be active for more than 5 clock cycles, but the other lanes have no cars it, the lane can be greenlighted until it reaches a nonzero value. If all lanes are empty, there won't be any greenlights. Finally, you can add cars to each lane while in greenlight mode, but if you try to add a car to a lane that is greenlighted, the number of cars in that lane will remain the same.

A note about organization:

For my project I did not use any busses, I find busses to be annoying, so I didn't use them. I did organize my wires in a bus structure, the register file outputs are a good representation of how this was done. I made the wires all equally spaced, so it isn't as messy and you can follow the wires more easily. I also organized all the control components at the top of the circuit and all of the counters stacked on top of each other, so they form a column.