**Lab 10: Control Unit Decomposition**

**Primary Objectives**

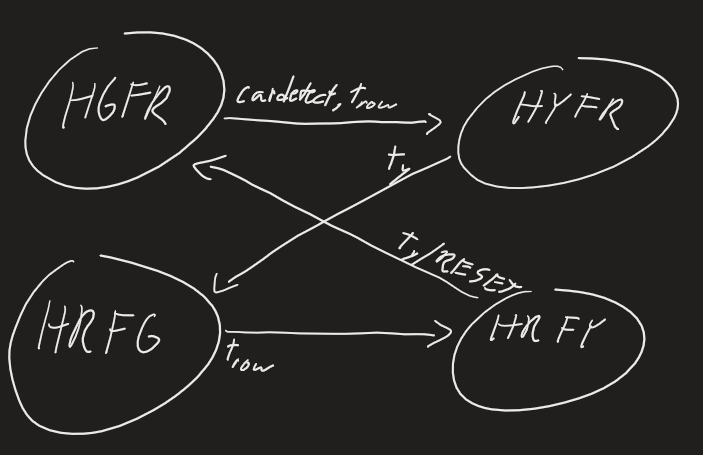
1. Design a control unit for an RTL system

2. Implement control unit using Logisim

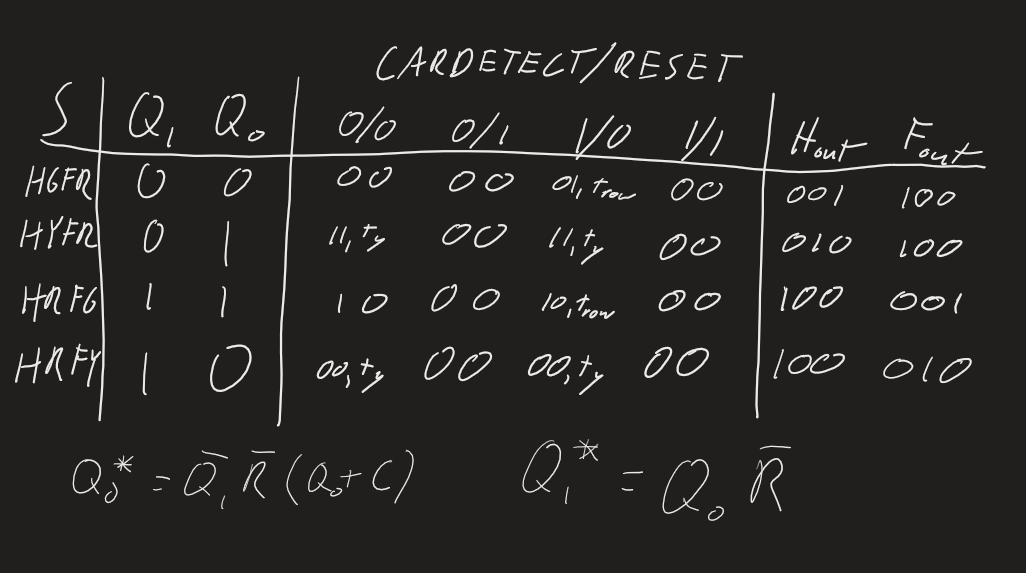
3. Test to verify the functionality of control unit

*Objective 1 Design*

The purpose of this system is to manage an intersection between a farm road and a highway, where only one road gets right-of-way at a time. The highway prioritizes right-of-way, and the farm road only gets right-of-way for a limited time when a car is detected at the traffic light. The system has two primary inputs, CARDETECT and RESET, and six primary outputs, HLG, HLY, HLR, FLG, FLY, and FLR which represent the colors of the traffic light for each road. (See Lab 9 for datapath details.) For the control unit, I began by setting up two D flip-flops whose outputs would act as the control signals for the data path, meaning they would control the mux selection for which lights would be active at a given time. I also drew up a state diagram and table to provide Boolean expressions for the next-state functionality using the flip-flops and primary inputs.



State Diagram



State Table

I also needed a way to send and receive signals to implement the counters to track *tright-of-way* and *tyellow*. This portion of the design was mostly done ad-hoc. I used simple Boolean logic gates to determine what combination of primary inputs and state variables should activate the counters. I then decided to connect the output of the counters to comparators to determine if they had reached their max value. I then went through several iterations of Boolean logic gate designs before I created a simple combination of gates that would disable the flip-flops and, by extension, state changes when the counters have not reached their max values.

*Objective 2 Implementation*

Figure 1 shows how I implemented the system using Logisim.

A diagram of a circuit

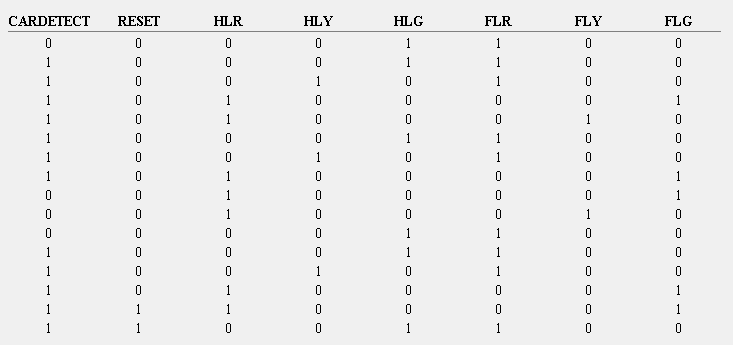
Description automatically generated

Figure 1 Implementation

We can see that every see to the left of the counters and mux is considered part of the control unit. The control signals are output directly from the flip-flops to the splitter before the mux. The status signals include the output from the counters communicating to the control unit how long it has until it can switch states.

*Objective 3 Testing*

To test this circuit, I tested various scenarios involving turning CARDETECT and RESET on at random times in the simulation to verify proper functionality. When CARDETECT and RESET are both 0, the highway light stays green and farm road light stays red no matter what. When CARDETECT is 1, the simulation waits 16 ticks to give up right-of-way before going yellow for 3 ticks and switching the farm road to green. If CARDETECT remains 1, the highway and farm road will keep switching right-of-way every 16 ticks (plus 3 to turn yellow). When RESET is 1, the system automatically switches to the initial safe state where the highway is green and farm road is red, ignoring the right-of-way time and time to turn yellow. This happens regardless of the system’s current state and CARDETECT’s value. A small sample log is shown below to demonstrate the interaction between the primary inputs and the primary outputs. (The clock is not shown to keep the log concise.)



Log 1

As can be seen, the system’s functionality matches exactly how it is described above.

**Conclusion**

The lab went smoothly for the most part. I initially struggled to get the counters’ output to interact properly with the flip-flops. It took a couple logic gate revisions before I managed to allow the counters to only change the system’s current state when they reached their max value. That process was very beneficial is teaching me the mechanics of how a comparator works.