

Group Assignment 2 -- EECS 270, Fall '09

Due Monday, Oct 26th @2:00pm in box.

Name: _____ unique name: _____

Name: _____ unique name: _____

Name: _____ unique name: _____

You are to turn in this sheet as a cover page for your assignment. This is a group assignment, all of the work should be that of only your group members. Assignments that are unstapled, lack a cover sheet, or are difficult to read will lose at least 50% of the possible points and we may not grade them at all.

It is expected that each group member contributed to the assignment; non-contributing members should not have their name on this document. This assignment is worth about 1% of your grade in the class and is graded out of 30 points.

Traffic light controller

1. Design Specification

In this homework, there are 5 inputs (the traffic sensors), and 5 outputs (the traffic lights associated with each sensor) and a reset. The sensor names are:

- **WS** -- West-bound straight
- **WL** -- West-bound left turn
- **NR** -- North-bound right turn
- **NL** -- North-bound left turn
- **E** -- East-bound (both right and straight)

The sensors are all either on (indicating a car is present in that lane) or off (indicating no car is present). The traffic lights associated with each direction have the same name, but end in a "TL". So **WSTL** is the West-bound straight traffic light. Each light can be Green, Yellow or Red.

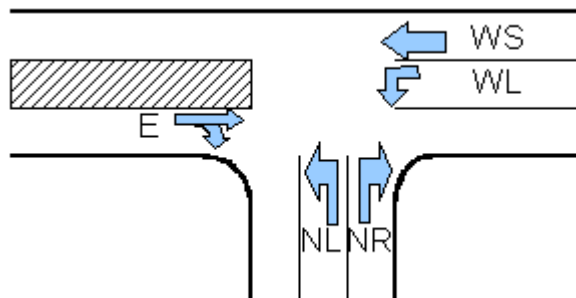


Figure 1: The intersection

The system clock is assumed to have a period of 1 second.

Your traffic light controller is to follow the following rules. These rules are NOT a complete specification. That is, there is some ambiguity in what to do in certain situations. You need to pick some reasonable behavior for those situations.

1. Two lanes of traffic are considered to "conflict" if having both of them going at the same time could cause a collision (that is if their paths cross). For example, having the both north-bound left lane and the west-bound left lane having a Green or Yellow light at the same time could cause a collision, so those lanes are considered to "conflict".
2. No two lanes in conflict should have "Green" or "Yellow" at the same time.
3. When a light is to change from Green to Red, it should be Yellow for exactly one second in between the Green and Red.
4. There should be no "starvation". That is, if a given sensor is on, its lane should get a Green light at some point, no matter the state of the other sensors.
5. When sensors for two conflicting lanes are on, no one direction should be Green for more than 10 seconds.
6. When only one sensor is on, that light must become and stay Green until some sensor input changes. It should become green as quickly as possible while still following the other rules.
7. If either North-bound lane sensor is on, the duration of its Green must be at least 7 seconds, no matter the state of the other sensors. (If they go off, the Green can be taken away immediately).
8. No Green should be less than 3 seconds in duration.
9. When no sensor is on, the default configuration should be having both North-bound lights Green. All other lights should be Red.
10. The traffic lights should behave in a way that would best move traffic through the intersection given the above rules. You are to assume the sensors are perfect and will always detect a car in that lane.
11. In general it is best to avoid having Red and Yellow lights. So it should never be the case that a given light could be Green but isn't.

The counter

All timing-related issues greater than 1 second are to be handled by using a counter. You may use $>$, $<$ and $==$ to compare to the current value of the counter. The only control you have over the counter is a reset line. The counter will be zero if reset was "1" last cycle, otherwise it will be one higher than last cycle.

2. Design Notes and Hints

- The best starting point is to figure out how many "basic" states you will have. That is, how many different states will you have that have green lights? Draw those states, and then think about what you want to do when transitioning from one state to the other.
- Think about how to use the counter. If you find yourself doing complex comparisons, you are probably doing something wrong.
- The instructor's solution used 4 bits to keep track of the state. You shouldn't need any more than that.
- We've provided a state diagram that provides an idea of how the controller might be drawn. You may find it useful to use that word document as a starting point though hand-drawn solutions are acceptable if neat.

Draw a state diagram which implements your controller. The logic on the transitions may be fairly complex. Use the naming conventions used above. You should expect to have things like "Counter<3" and the like as conditions on your state machine. Also think about when you want to reset the counter (this should be an output of your state machine).