Nick Barnes CS 7680 Homework 03 October 15th 2024

Task C [40 points]: Answer the follow questions

1) How do you know that your Raspberry Pi GPIO pins are in safe states at the end of your code? Why is this important?

To ensure a safe state, all used GPIO pins are set to PI_INPUT before terminating the program/when the program loop is exited, ensuring that they are not trying to send any signal/current when the program is terminated. By setting the pins to PI_INPUT, we ensure that they are in a “listening” state and are not active. If we were to terminate the program while pins were in PI_OUTPUT/active, they could behave in unexpected ways, which we want to avoid.

2) Describe without writing code how you might adjust the traffic light configuration of Tasks A and B by introducing a road pressure plate (a device that signals to the system that a vehicle is waiting at the traffic signal for a change in direction). What might be needed to adjust the hardware circuit? What might be needed in the software to support this input signal? How might the traffic signal operational loop be modified?

If there was a pressure plate involved, we would need to include in our code the functionality to constantly be checking for input from a sensor, so that whenever that sensor is triggered, we could exit the loop/jump to a specific point. We would need to add this sensor as an input to one of our pins, and then while the software loops through traffic light colors at regular intervals, be constantly checking for any input from that pin, and give that lane a green light when the pressure plate is triggered. In practice, since there would be multiple lanes/directions with pressure plates, each would be hooked up to an input, and could operate with a queue similar to how the elevator model used a queue for requests, so that when one pressure plate is triggered, they get a green light for a certain period of time, and then the next plate that has been triggered/added to the queue could get the green, etc. Lanes which have not had a pressure plate triggered could either be skipped, or could be included in the rotation but with a shorter green light duration, since we don’t expect that there are cars waiting there.

3) Consider a problem solution where illuminating a light/LED would be useful. Design a different circuit that could be used and describe (not code) the software rules and flow that would be necessary to deliver that solution.

This may not be a particularly exciting example, but what first comes to mind is a recording device of some sort, which may have LEDs to indicate that the device is on, as well as to indicate when the device is recording.

The **power** LED could be hooked up to the main power source for the device, such that when the device is turned on and power starts flowing, it goes through an appropriate resistor to power the LED as long as there is power. When the device is shut off and power stops flowing, that LED would cease to illuminate.

The **recording** LED would require more software logic, and the circuit would be similar to the one we built for this homework, where we want to control that LED relatively independently of the other hardware. We could set up some logic so that when we hit the record button (assuming there are no errors and we can record), we set that LED to PI_HIGH, and leave it in that state until we stop recording (either by hitting the record button again, or powering off the device I suppose), at which point we would set it to PI_LOW. Now that I'm writing it out, it is pretty much exactly what we did in the in-class activity a week or two ago, where we configured a button to turn an LED on when it was pressed, and stay on until the button is hit again.