# Final Design

The final design is composed of the 3 main modules as in the proposal:

* Obstacle Detection/Anticipation
* Output
* Automation

To implement and integrate the 3 modules, we design 2 major parts. This section will describe these two parts, and the considerations that went into design of each.

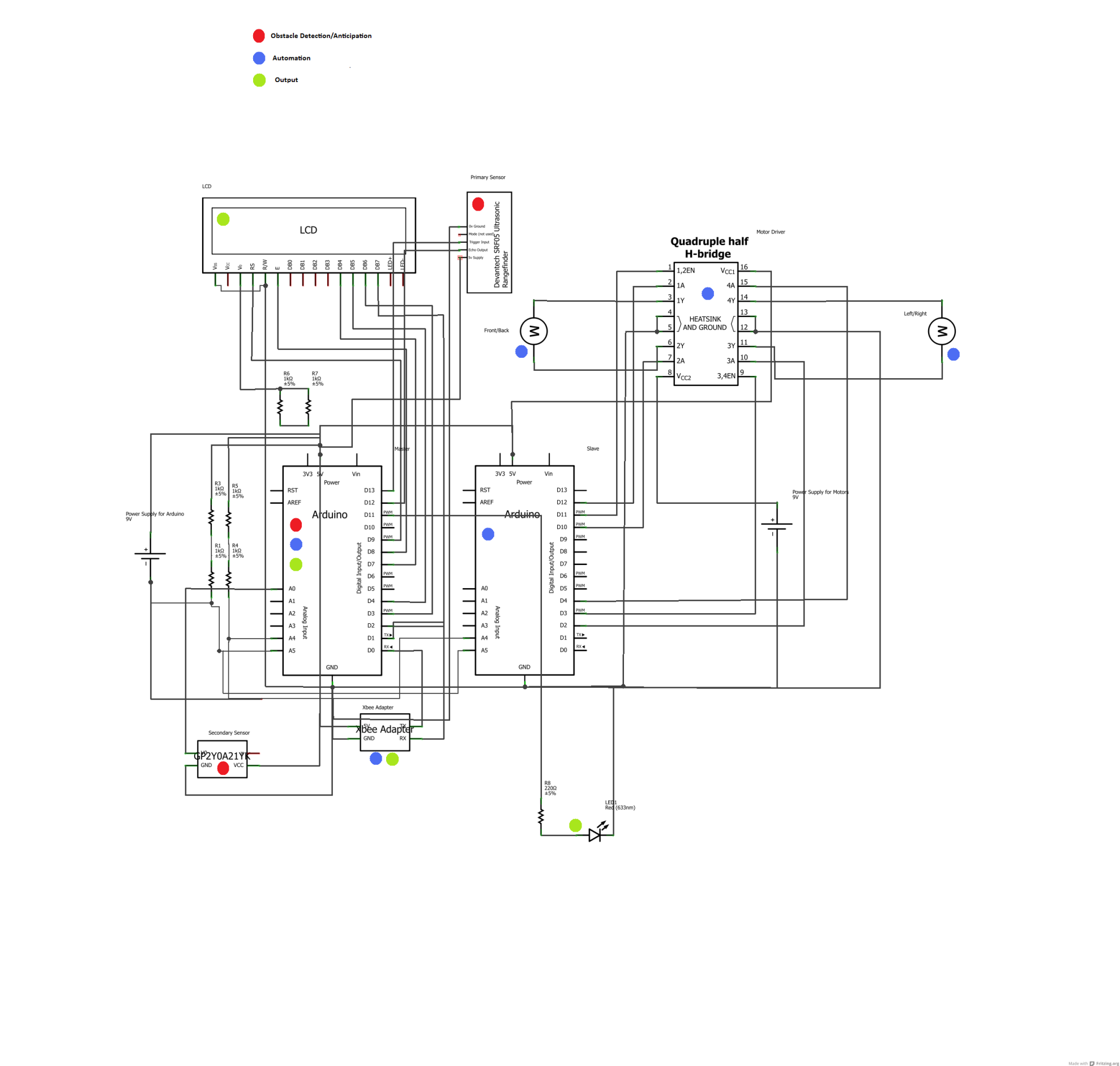
1. Onboard Implementation
   1. Hardware Schematic
   2. Description of Components
   3. Additional Sensor and Considerations
   4. Output Module and Considerations
   5. Automation Module and Considerations
   6. Onboard Software
2. Remote GUI for control and output

## Onboard Implementation

This subsection describes the hardware and software within the onboard design.

### Hardware Schematics

The following is the schematic for the hardware of the design:



As seen, the main components of the hardware have been labeled according to the module they belong to.

### Description of Components

The following is a table to further outline the function of each of the major hardware components:

|  |  |  |
| --- | --- | --- |
| Component | Function | Module |
| Arduino (Master) | * Drive the sensor * Perform distance, speed, and time calculations * Receive commands from the user * Transmit command to Arduino (slave), receive speed levels from Arduino (slave) * Drive output (LCD, LED, External) | Obstacle Detection/Anticipation  Output  Automation |
| Arduino (Slave) | * Operate the motor driver according Arduino’s (Master) commands * Track speed levels | Automation |
| Motor Driver (SN754410) | * Regulate the speed and direction of the two motors on the user vehicle (steering, front/back) according to Arduino’s (Slave) commands | Automation |
| LED Light | * Indicate danger levels with varying brightness | Output |
| LCD Screen | * Provide summarized information to the user | Output |
| Ultrasonic (Primary) Sensor (SRF05) | * Gather “time of flight” data to be used for distance values for obstacle anticipation | Obstacle Detection/Anticipation |
| Infrared Proximity (Secondary) Sensor (GP2Y0A21YK) | * Detect obstacles in the blind spot when it is out of the primary sensor’s range | Obstacle Detection/Anticipation |
| Xbee Wireless Module (Series 1) | * Gather wireless commands from user, and transmit sensor data wirelessly to be used in the GUI | Automation  Output |

### Additional Sensor and Considerations

One major revision in the final design from the proposed design is the addition of an extra sensor. The secondary sensor only covers the blind spot region of the car. This region is not fully covered by the primary sensor due to its placement. The primary sensor is placed to extend the range of the anticipation feature, and as a result and additional sensor is needed to fully cover the user’s vehicles blind spot.

The situation can be seen in the following picture of the physical implementation of the circuit on the user vehicle:

[picture]

### Output Module and Considerations

The LCD screen, and the LED are very important components of the design as it acts as the onboard part of the output module. It provides a summary of all the relevant information analyzed by the microcontroller, according to the various danger levels. The danger levels are determined by the time left calculations, and the presence of an obstacle within the blind spot. The following table outlines the behavior of the LCD screen, and the LED:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Situation | Sensor Involved | Time Left [s] | Danger Level | LCD Screen Output | LED Output | Steering Locked |
| No Danger | Primary | t > 15 | 0 | Distance to obstacle, “No Danger” | Off | N |
| Obstacle Approaching | Primary | 11<t<15 | 1 | Distance to obstacle, relative speed, “Obstacle Approaching” | Brightness from 0% to 78%, depending on increasing distance | N |
| Obstacle Approaching | Primary | 6<t<11 | 2 | ‘’ | ‘’ | N |
| Obstacle Approaching | Primary | 3<t<6 | 3 | ‘’ | ‘’ | N |
| Obstacle Very Near | Primary | t<3 | 4 | Distance to obstacle, relative speed, “Caution” | Fully On | N |
| Obstacle at Blind Spot | Secondary | t<=0 | 5 | “Danger, Check Blind Spot” | Fully On | Y |

The LED behavior has been revised from the proposal since blinking the LED requires delay to be added to the software. This delay reduces the frequency of operation of the sensor, and degrades the overall performance of the system. As a result, the current implementation of the output module was designed to ensure intuitive feedback to the user, without sacrificing performance.

### Automation Module and Considerations

The automation module is activated if a certain level of danger has been detected. As seen on Table Y, danger levels 4, and 5 activate the steering lock. The reason for this implementation is to maximize user control of the vehicle, but prevent collisions. There are dangers associated with this solution. Due to this feature, more serious frontal collisions may occur. These situations will have to addressed in future iterations of the design.

### Onboard Software

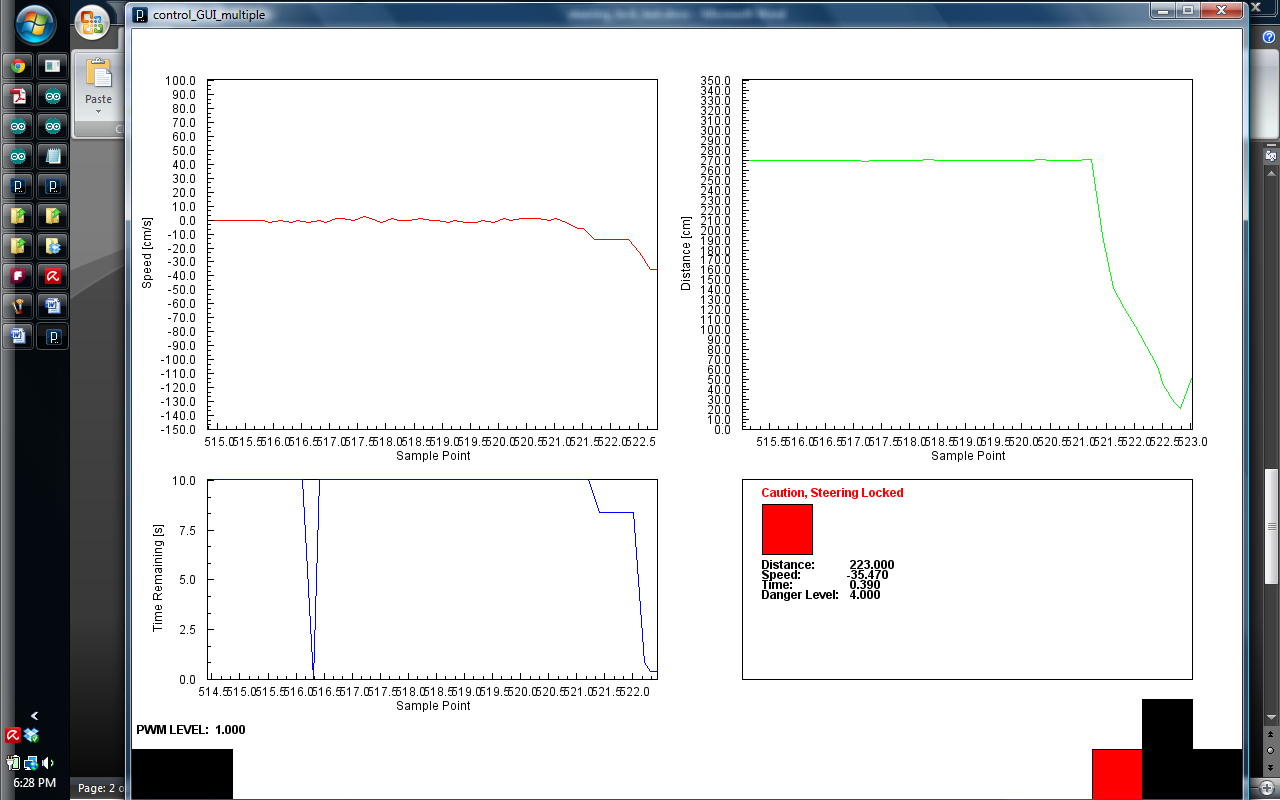
To drive all the hardware components in harmony in order for them to perform their required functions, software has been programmed into the two microcontrollers. Some main features of the software can be found in the”Function” column of the two Arduino components in Table X. The software, with comments explaining the functions of the separate pieces of the code has been included in Appendix X.

## Remote GUI for Control and Output

It was noted that the LCD screen wasn’t a sound enough output needed for the debug, testing, and demonstration of the project. Therefore, a remote GUI was designed. The main functions of the GUI are:

* Remote control the user car for testing and demonstration
* Remotely output the microcontroller’s (Arduino Master) output in a graphical form for debug, testing, and project demonstration

The following is a screenshot of the GUI:



The GUI consists of the following components:

* Real-time plots
  + Relative Speed
  + Distance
  + Time Remaining
* Summary Box mirroring a combination of the physical LED, and LCD screen; outlining information regarding danger levels, and instantaneous readings from Arduino Master
* Indication of the current speed level (motor PWM) of the users vehicle
* Virtual buttons indicating the following controls:
  + Accelerate
  + Reverse
  + Steer Left
  + Steer Right
  + Increase Speed Level
  + Decrease Speed Level

The GUI has been written in Processing, and the commented code can be found in Appendix X.

Appendix X

1. Arduino (Master) Code

[paste code here]

1. Arduino (Slave) Code

[paste code here]

1. GUI Code

[paste code here]