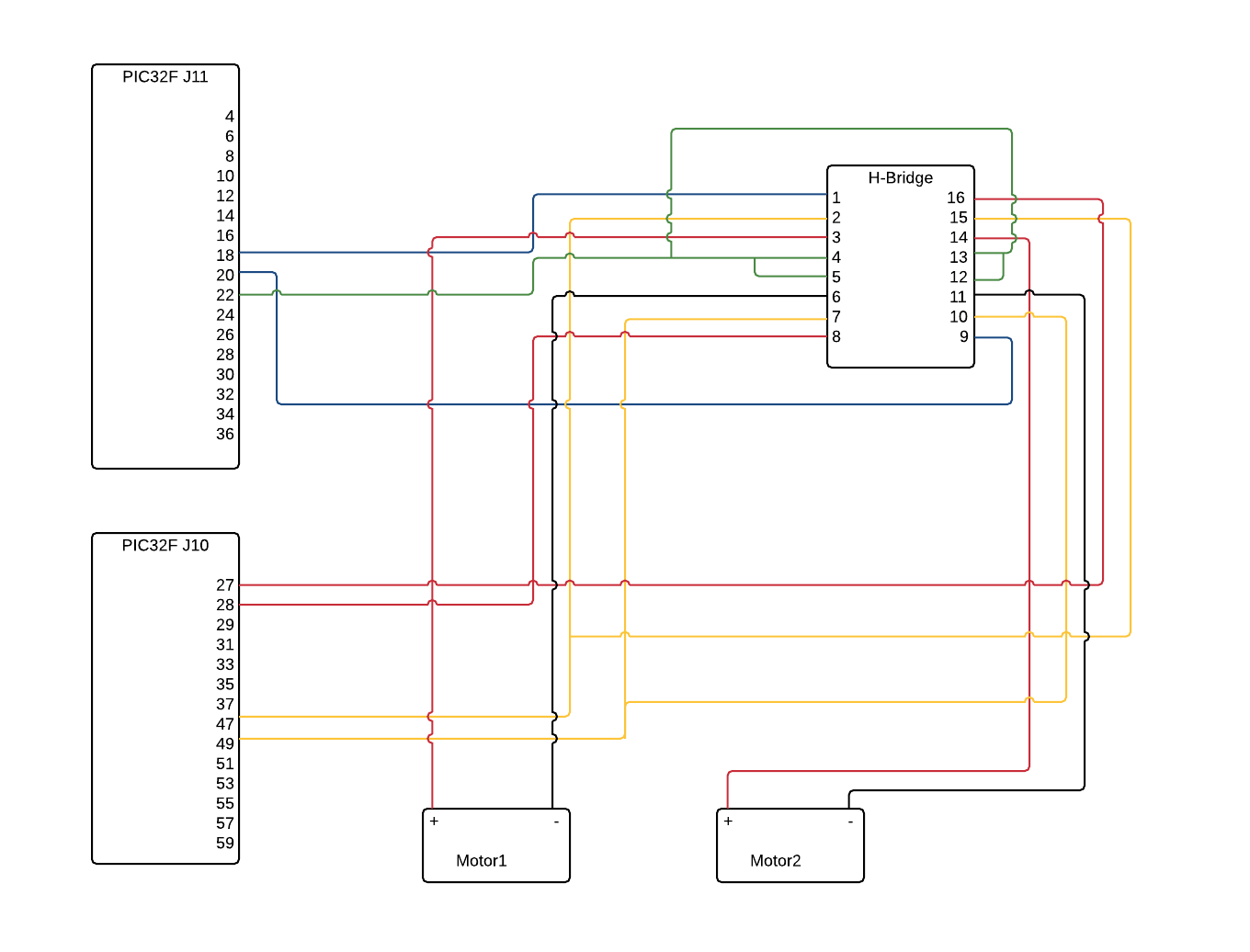
Preliminary Design Review

Design Summary:

Our design will consist of three infrared transmitters and receivers that will be utilized to track a black tape path on the ground. The transmitters and receivers will be mounted on a board suspended a few centimeters above the ground. One pair of transmitters and receivers will be positioned at the front of the car and centered on the black tape path. The other two sensors will be about 4 centimeters behind the lead pair of sensors and will be positioned in a way that they straddle the black tape line. Both the transmitters and receivers will be positioned downward at a slight angle in order to maximize the range of light detected by the receivers. The software will utilize the readings from the three receivers in order to adjust the speed of the wheels to correctly follow the path.

The extra sensor that we are integrating with our microcontroller will be a Bluetooth receiver. This receiver will take prompts for a phone/computer and communicate them to the microcontroller via SPI. Using those prompts the microcontroller will control the speed of the motors in order to navigate the course. We will still be able to use the IR sensors if we choose to but controlling the robot via Bluetooth should result in a more accurate trip through the course than the sensors could offer.

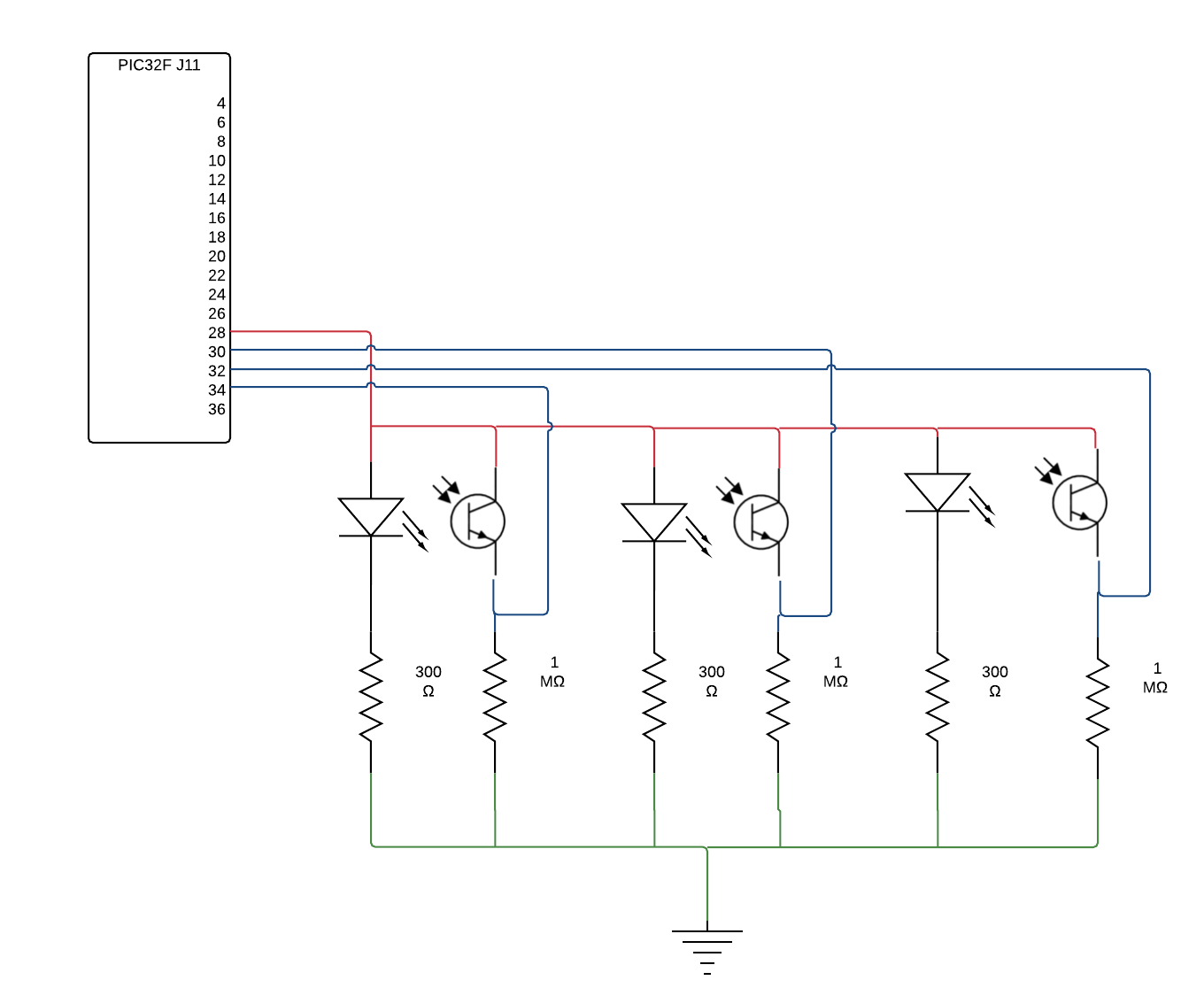
Hardware Design: Motor Connections

All pins on the H-Bridge will be connected to the PIC32 by wire-wrapping

The motors will connect to the H-Bridge by jumper wires

All Grounds and sources will be connected by wire-wrapping

IR Sensors/LED Connections

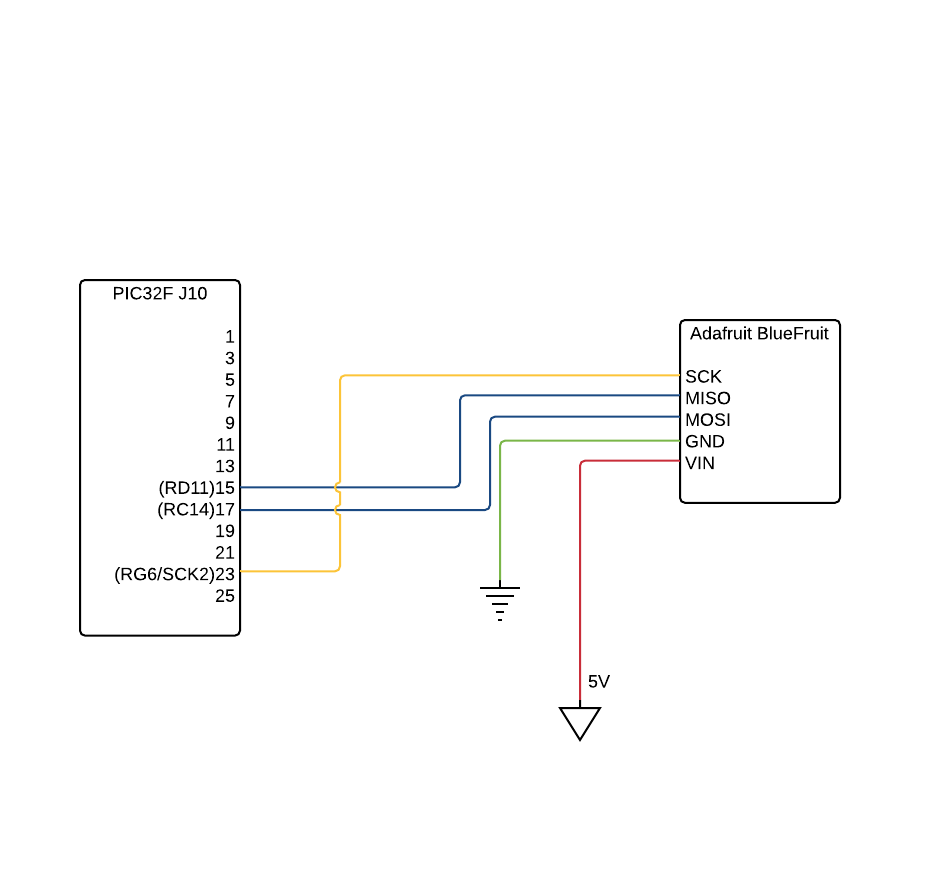


Connections between the LED, Phototransistors, Resistors, and the PIC32F will be connected by wire-wrapping.

Grounds will be connected by wire-wrapping.

All connections that are for the motors apply in this diagram as well.

Adafruit Bluefruit LE SPI Friend Connection



All connections will be wire-wrapped between the PIC32F and the Bluetooth module.



Software Design:

The software will utilize the readings of the infrared receivers in order to adjust the speed of the wheels on the robot to follow the black tape line. Because the state machine is more or less circular in functionality I will describe the functionality of the code as opposed to graphically showing it.

1. Line-up: The lineUp() will be called to ensure that the robot is starting in the correct position. It will ensure that the middle sensor is over the black tape while the two side sensors are not over the tape. The robot will stay in this initial state until one of the buttons on the pic is pressed.
2. Drive: At this state the robot will be moving through the course. There will be a number of functions called (readFromADC, calculateODC, adjustLED) that will be called while the robot is in motion. These functions will be sampling the infrared receivers and using that data to adjust the ODCs for the two motors in order to turn the wheels accordingly. The LEDs will be on or off depending on which sensor is over the black tape. (On indicates over the tape and off means not over the tape) This functionality is more important for debugging purposes.
3. Turn Around: When we get to the end of the course we will encounter three “T”-like patterns. At that point the robot needs to stop sensing where the line is and turn around. It will do this by completely stopping one wheel and powering the other. Once it detects that it has completely turned around the state will switch back to the drive state and it will begin to do the course backwards.
4. When the Bluetooth has been integrated the drive state will become a case statement. We will pass it a character and depending on the character the wheels will operate in a certain way. This will allow for far more accuracy when navigating the course because we can tell the robot exactly what to do.

Functions:

Void lineup()

-Calls readFromADC() function

-Checks if left and right both see white and front sees black

If above condition is true turn on RD0, RD1, RD2 LEDs

-If all three LEDs are on waits for switch input from RD6 switch

Void readFromADC()

-Pulls the readings form the infrared receivers out of the ADC1BUF0 buffer

-Using a pointer places the correct buffer data into a global variable corresponding to the position of the sensor (front, left, and right).

Void calculateODC()

-Uses the values found in variables front, left and right to determine speed of the wheels

-If left and right see white while front sees black keep going straight

-If left and front see black while right sees white turn left

-If right and front see black while left sees white turn left

-If left right and front see black go to turn around mode

-If left right and front see white continue slowly left or right sees black

Void adjustLED()

-If left sees black turn on RD0 LED/ If left sees white turn off RD0 LED

-If front sees black turn on RD1 LED/ If front sees white turn off RD1 LED

-If right sees black turn on RD2 LED/ If right sees white turn off RD2 LED

Final Project Device Proposal

# Summarized Proposal

For part 2 of this project we will be integrating a Bluetooth receiver to our Pic microcontroller. We will be using SPI to communicate between the receiver and the microcontroller. Using our phone or computer to communicate with the Bluetooth receiver we will be able to control the speed of the two motors in order to navigate the course. No additional parts will be needed since we already have access to enough headers to properly connect the receiver to the microcontroller.

# Part Details

The device we will be using is the “Adafruit Bluefruit LE SPI Friend - Bluetooth Low Energy (BLE)”. Its part number is 2633 and it is sold by Adafruit for $17.50 (USD). This Bluetooth device uses SPI to communicate to the microcontroller and operates at 3.3V so it is directly compatible with the microcontroller. Find it online at https://www.adafruit.com/products/2633

