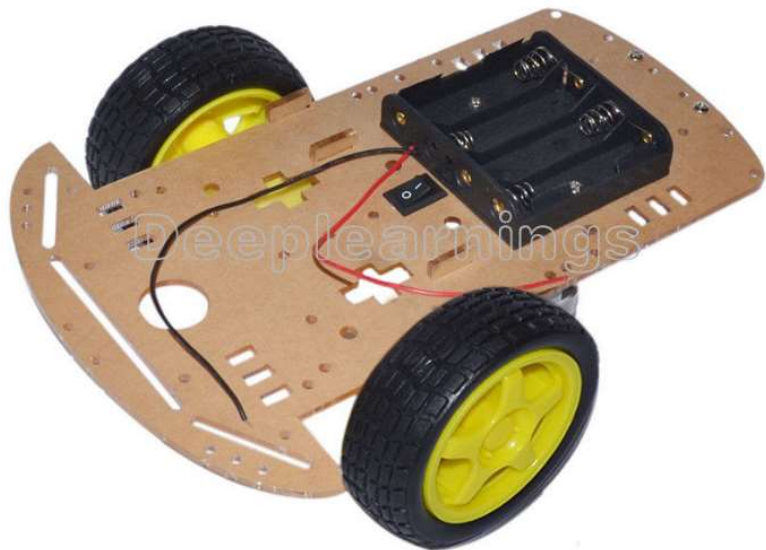


circleBot Assembly

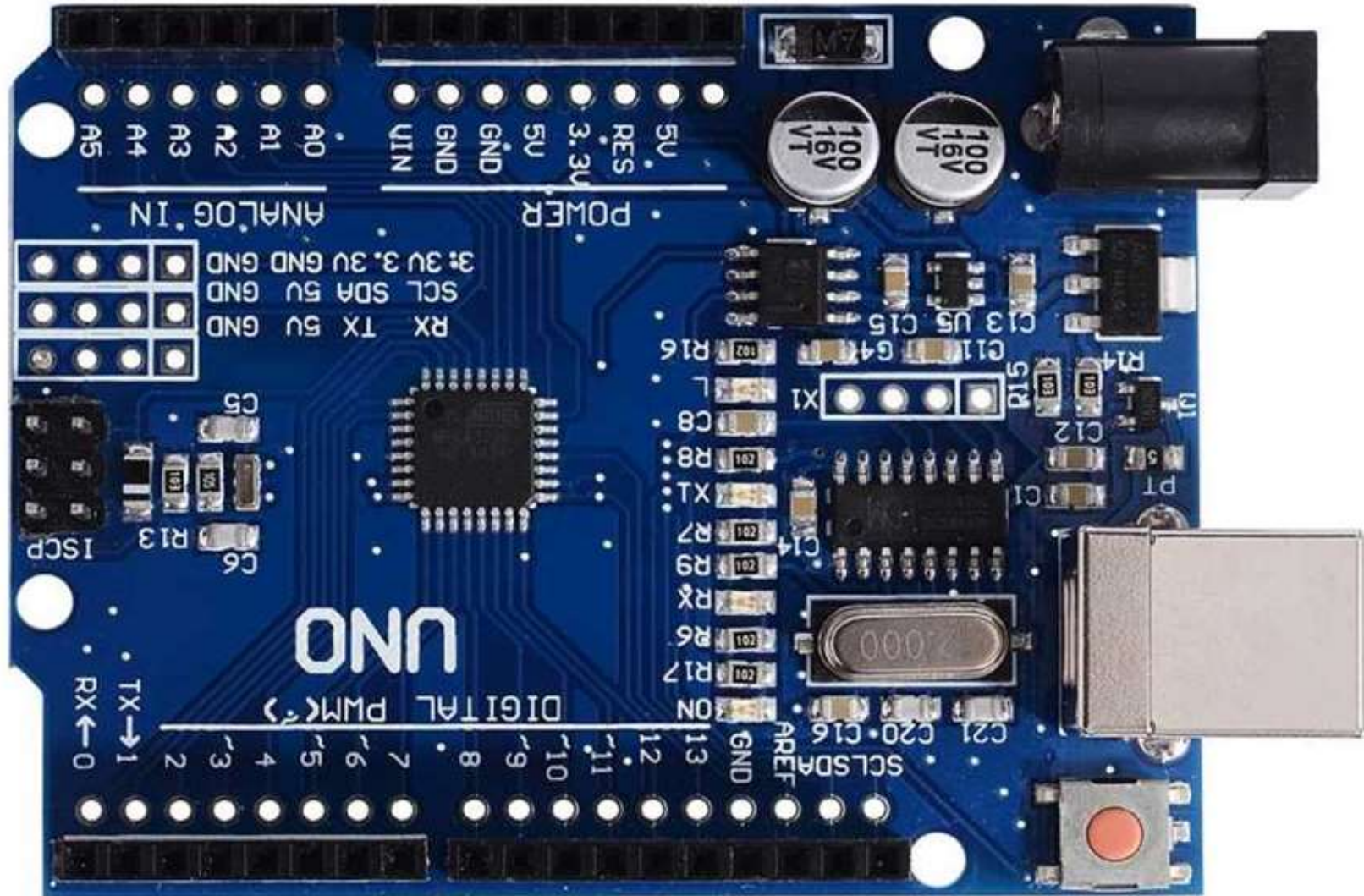
circleBot Parts – A Robot Chassis kit

Any two-wheel drive robot kit will work fine.

(ex. DF Robot Turtle)



circleBot Parts – Arduino UNO

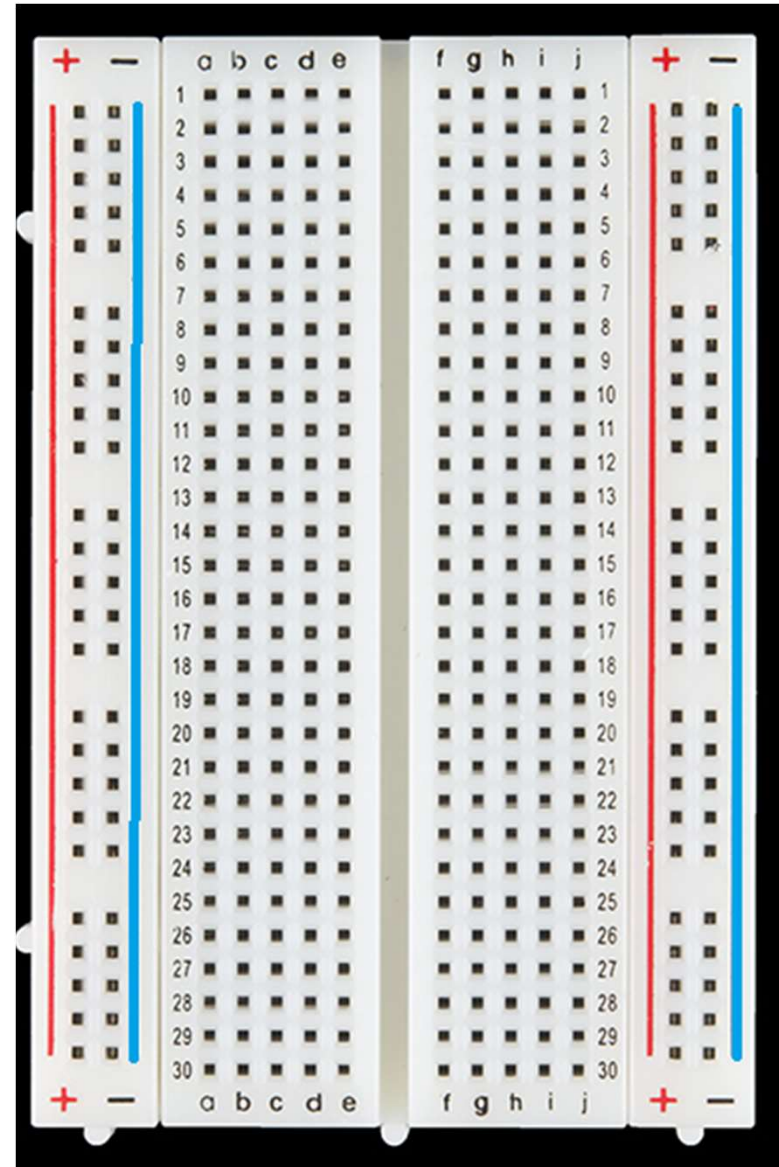


circleBot Parts – Bread board

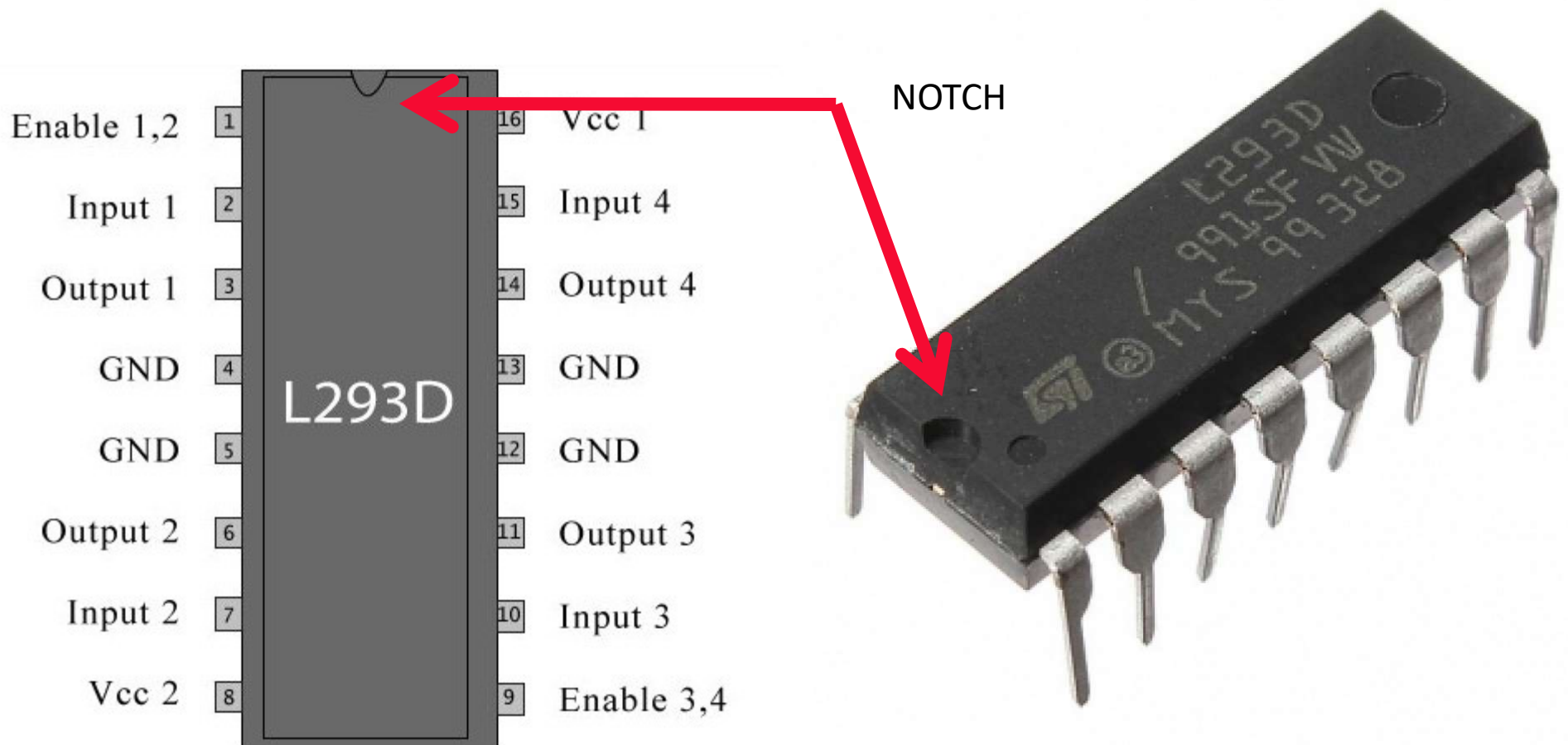
We mount our parts to this board

Red is POSITIVE 5 volts

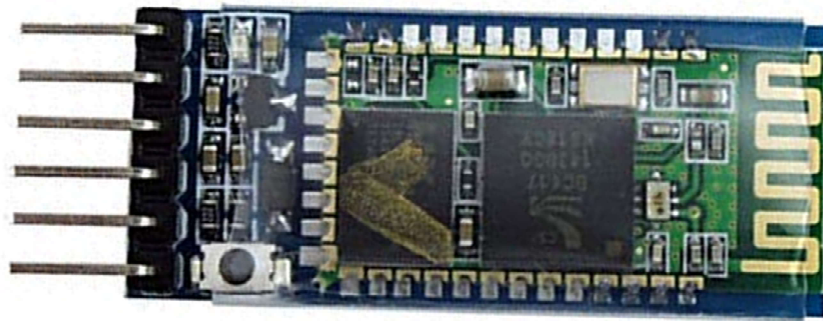
Blue is NEGATIVE GROUND (GND)



circleBot Parts – Motor Chip



circleBot Parts – BT-05 BluetoothLE module

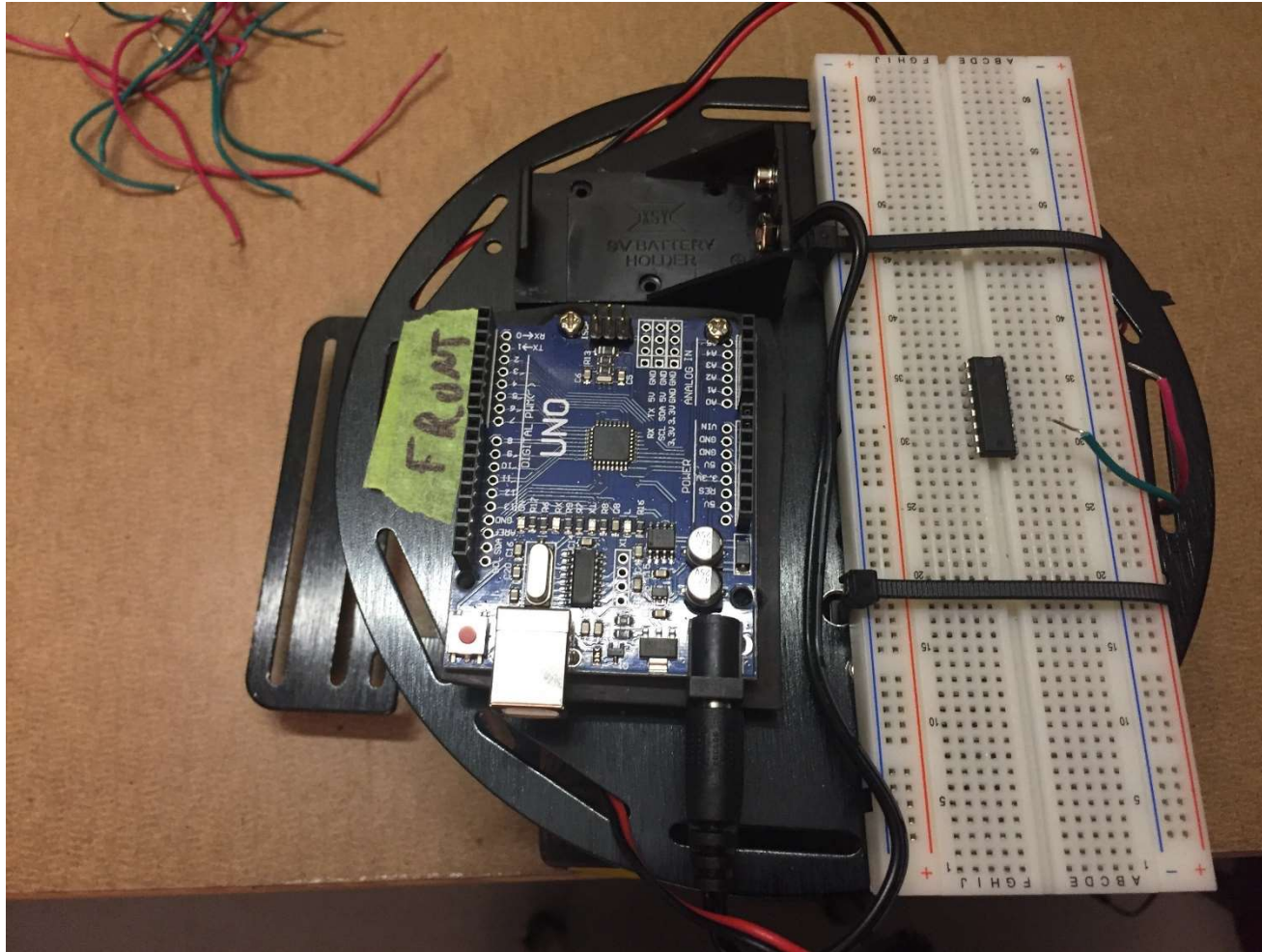


circleBot Parts – Battery Connector



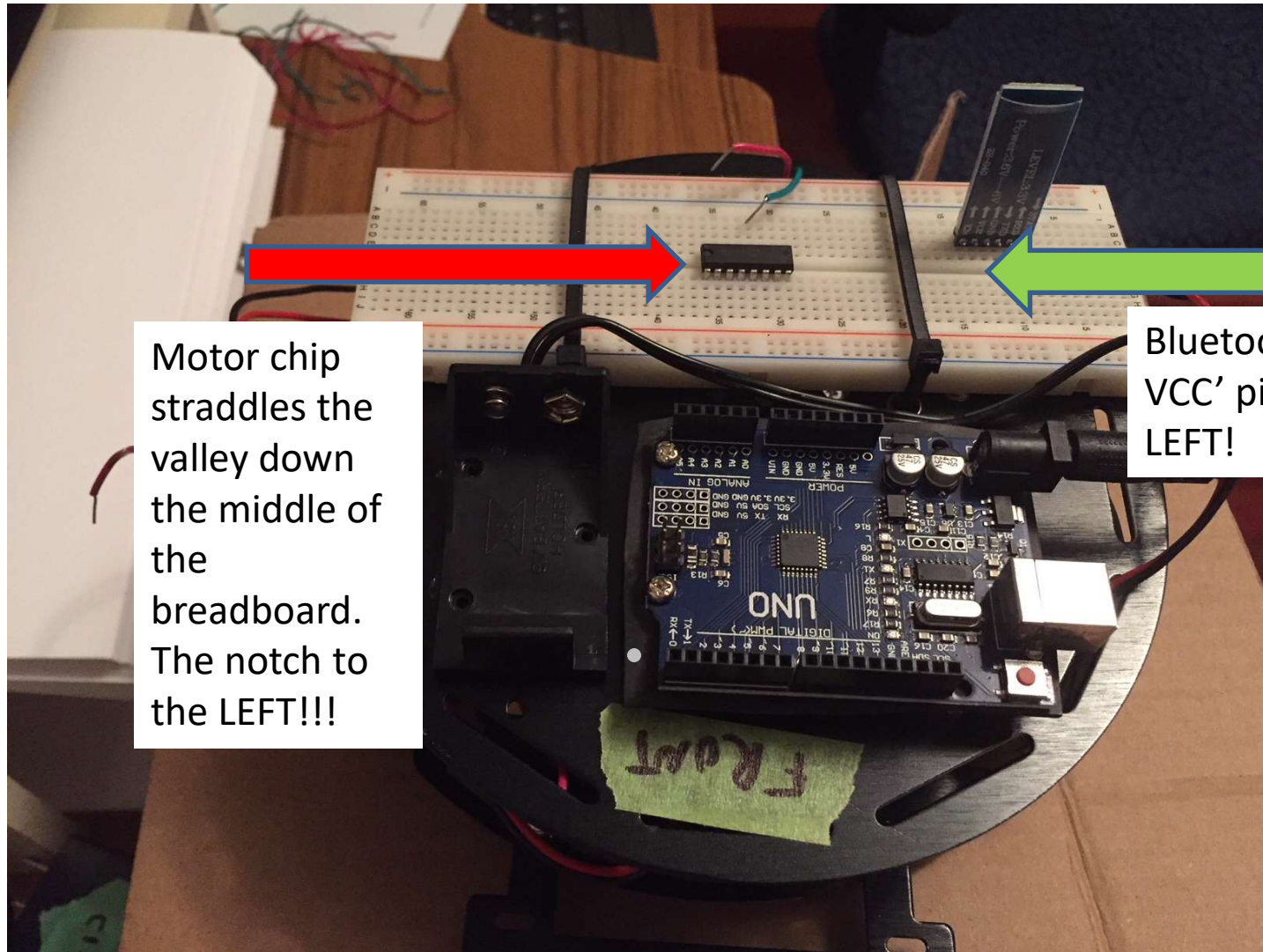
This one powers the ARDUINO board.

Assemble the kit



Assemble the kit as directed by the instructions in the box. Attach the Arduino Uno to the top with 2 or 3 screws and standoff posts. Attach the breadboard with some zip ties. Attach the 9 volt battery holder to the frame with some tape.

Mount the Components



Short RED wire from RED rail to Motor chip pin 16

Short GREEN wire from BLUE rail to motor chip pins 12

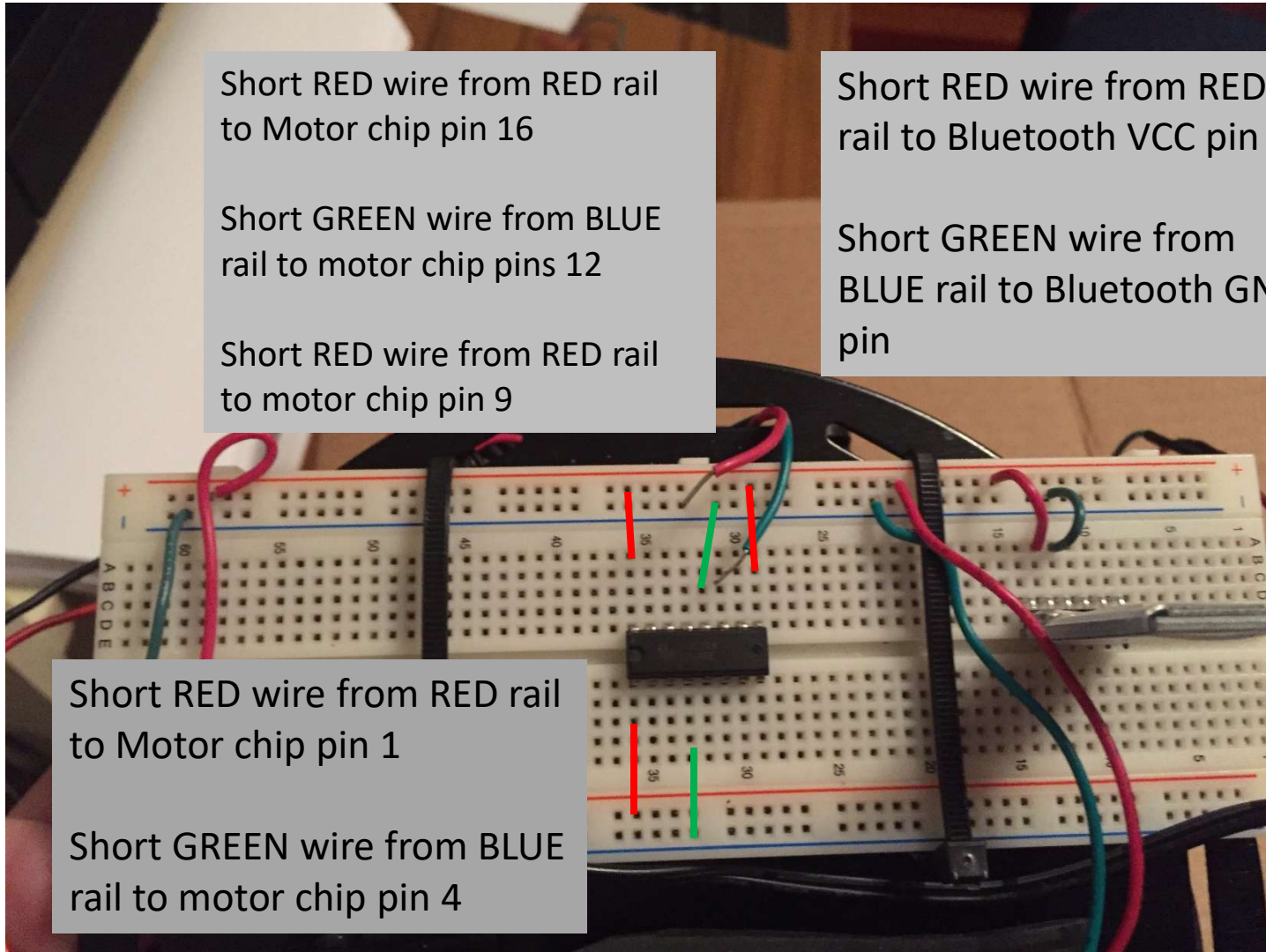
Short RED wire from RED rail to motor chip pin 9

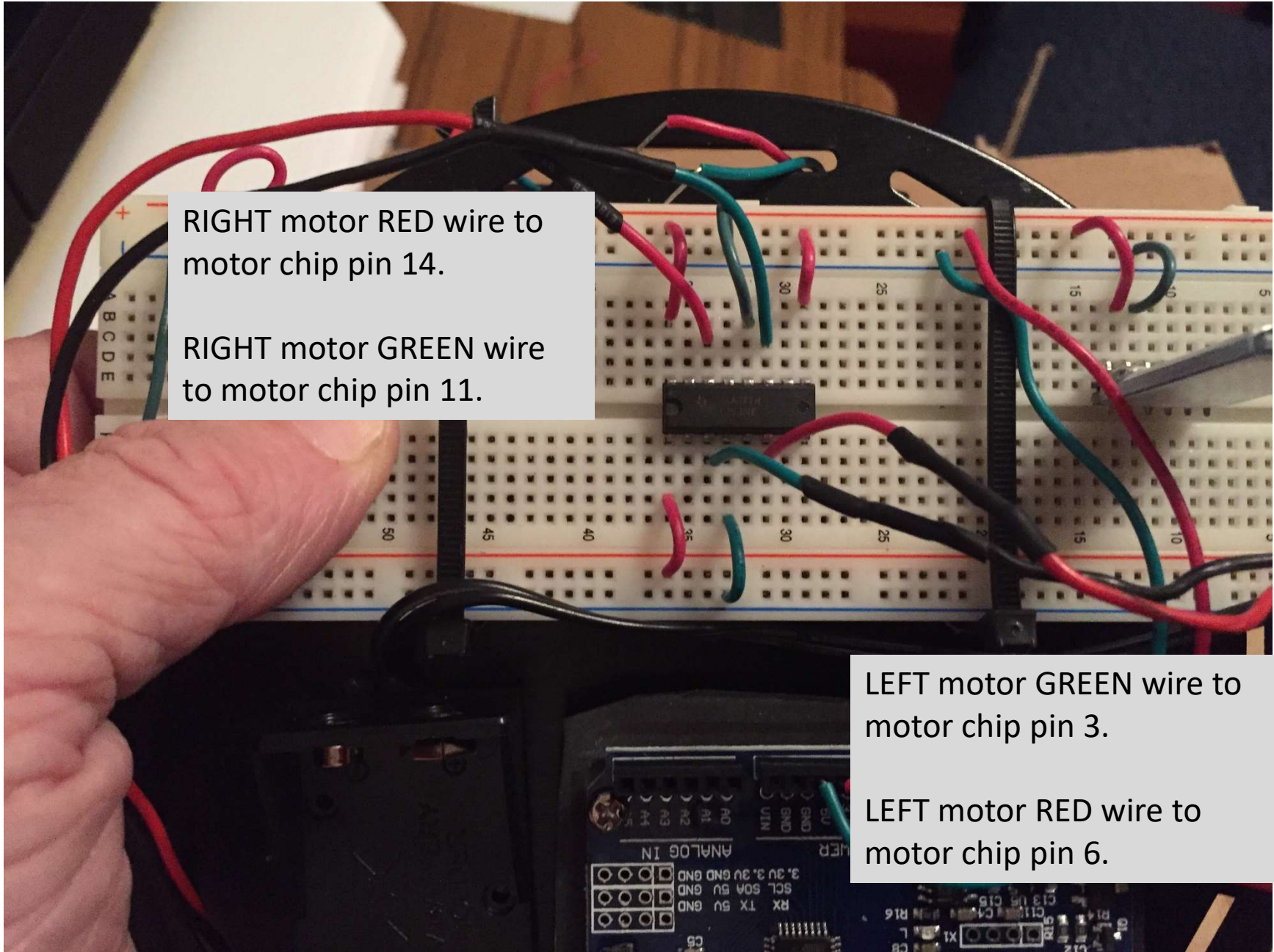
Short RED wire from RED rail to Bluetooth VCC pin

Short GREEN wire from BLUE rail to Bluetooth GND pin

Short RED wire from RED rail to Motor chip pin 1

Short GREEN wire from BLUE rail to motor chip pin 4



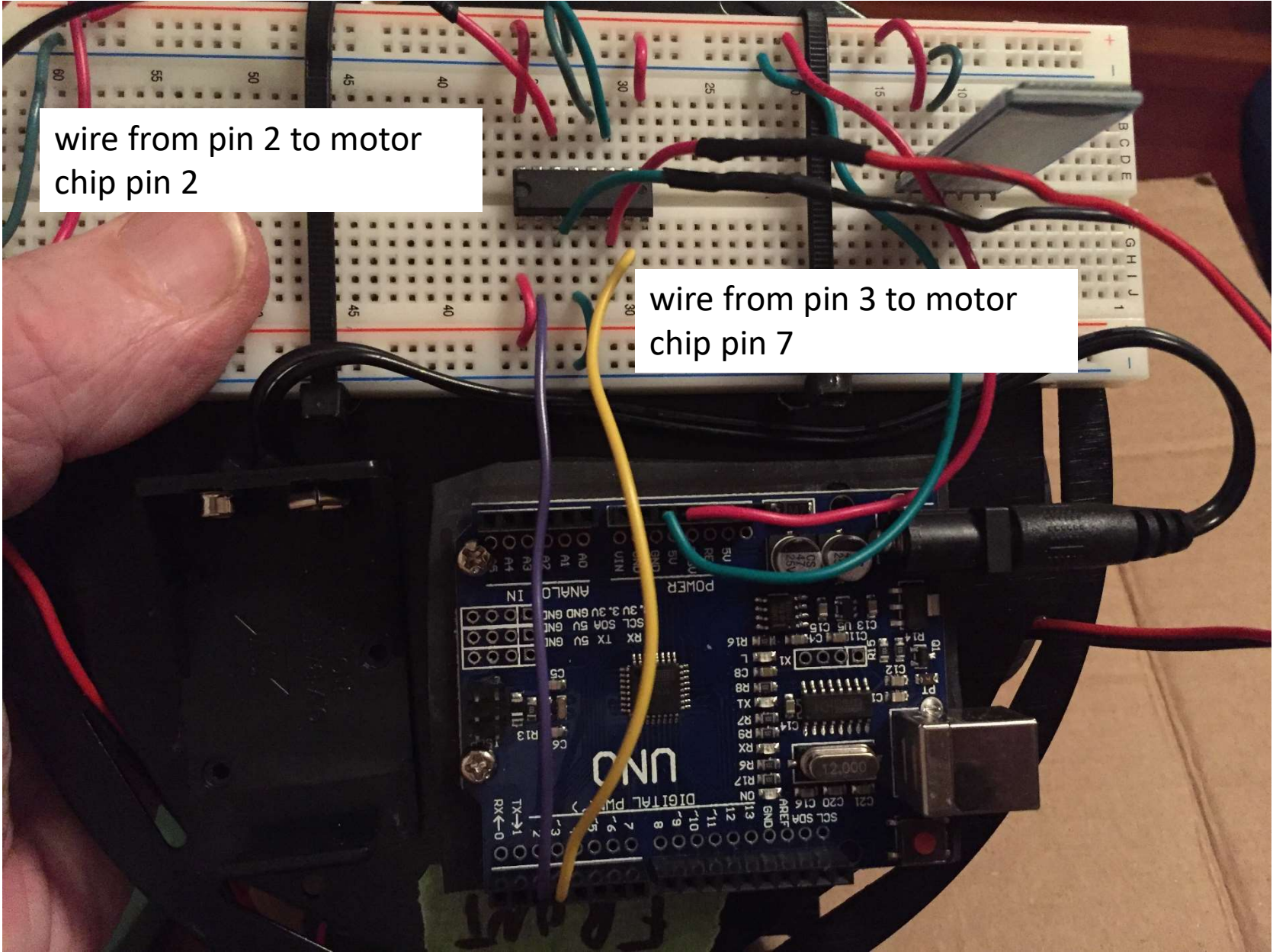


RIGHT motor RED wire to motor chip pin 14.

RIGHT motor GREEN wire to motor chip pin 11.

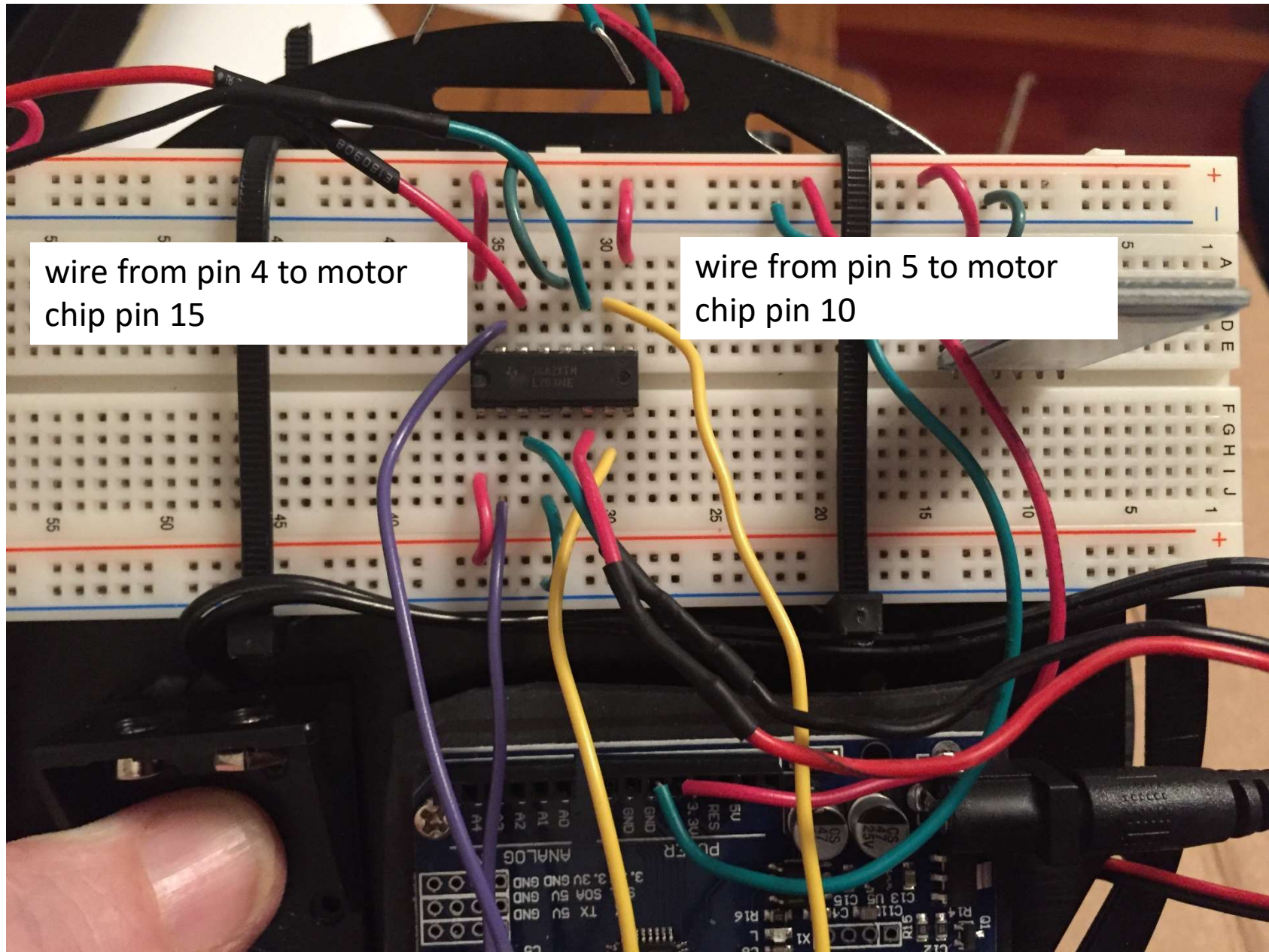
LEFT motor GREEN wire to motor chip pin 3.

LEFT motor RED wire to motor chip pin 6.



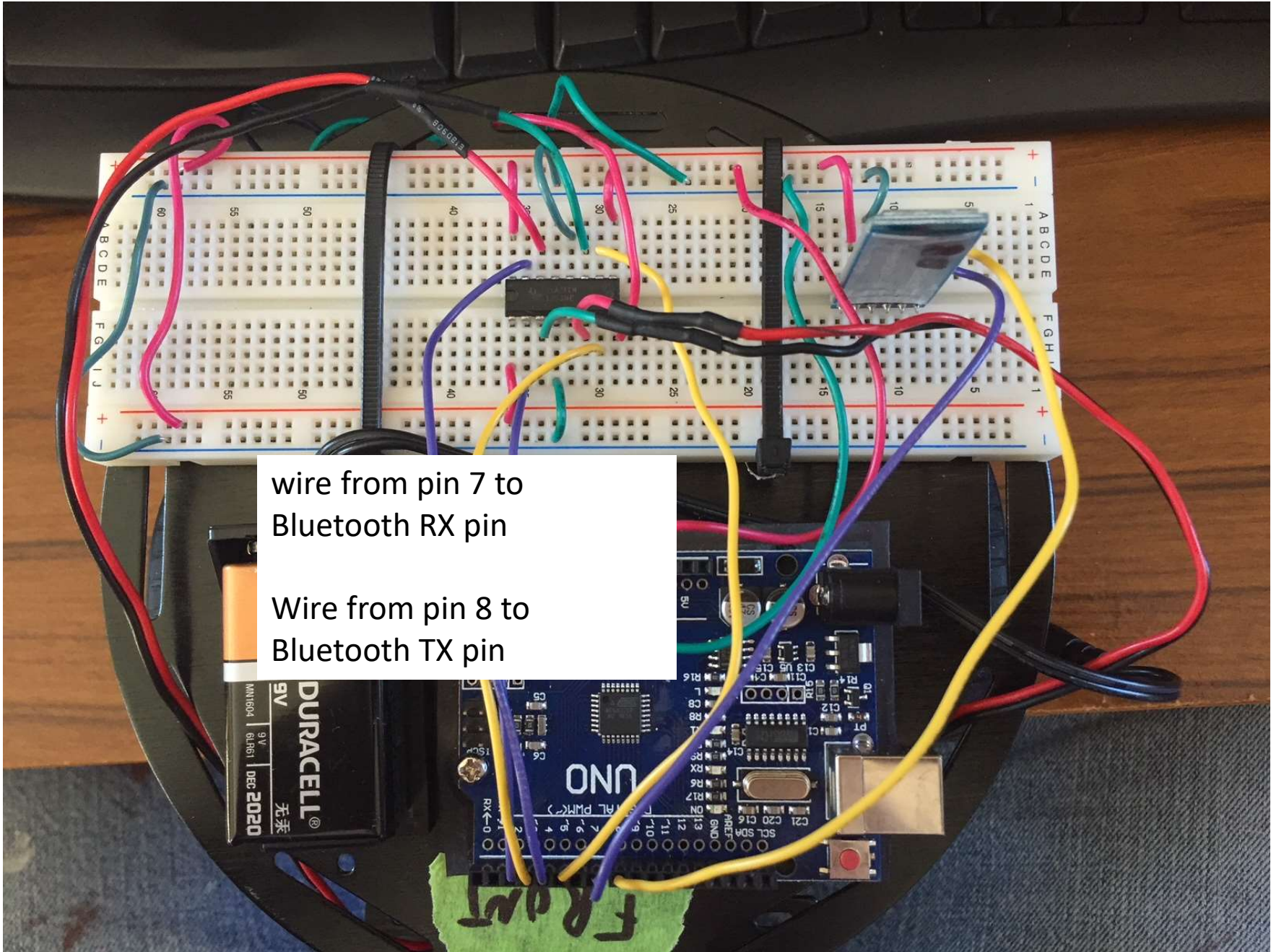
wire from pin 2 to motor
chip pin 2

wire from pin 3 to motor
chip pin 7



wire from pin 4 to motor chip pin 15

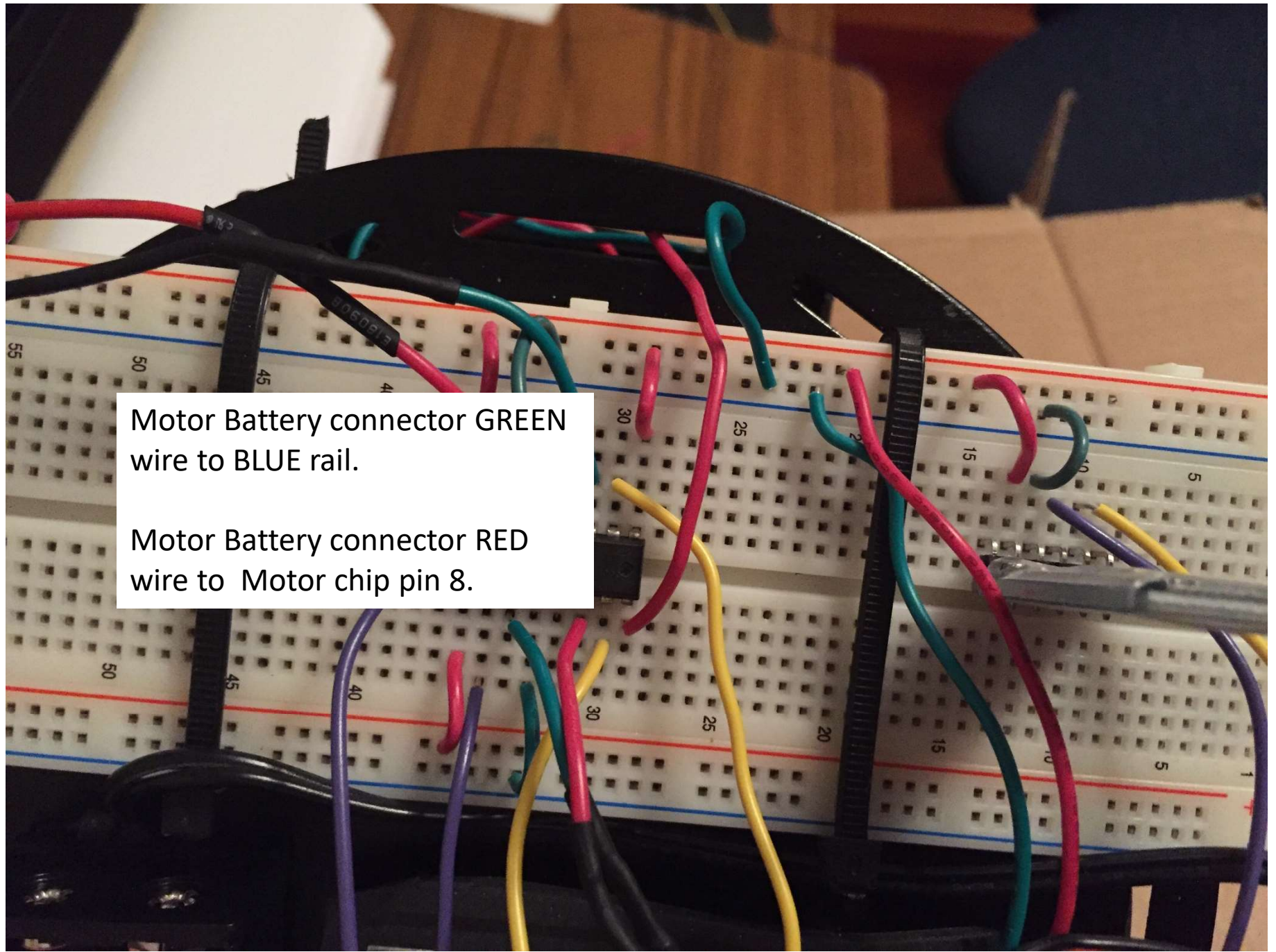
wire from pin 5 to motor chip pin 10



wire from pin 7 to
Bluetooth RX pin

Wire from pin 8 to
Bluetooth TX pin

FRONT



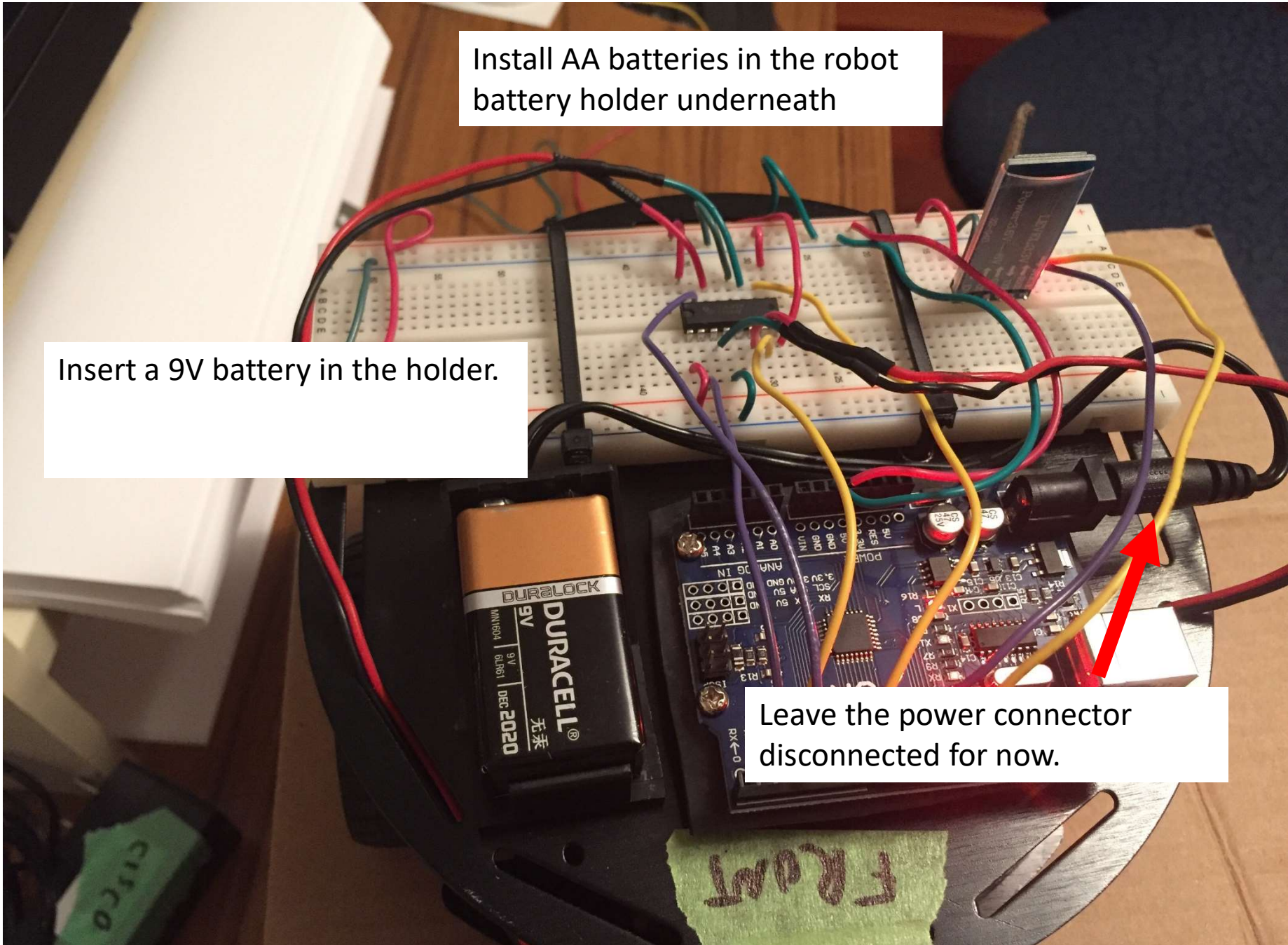
Motor Battery connector GREEN wire to BLUE rail.

Motor Battery connector RED wire to Motor chip pin 8.

Install AA batteries in the robot battery holder underneath

Insert a 9V battery in the holder.

Leave the power connector disconnected for now.



circleBot Coding

- Log into a workstation and go to the Windows Apps menu.
- Select Arduino.
- Delete all the code and save the sketch as 'circleBot' in your H drive.
- Enter the code from the following pages.

Take your time. Typos won't work.

SAVE YOUR WORK AS YOU GO!!!



circleBotBlueTooth

```
// circleBot BlueTooth for HC-05(5 pin - BLE) OR HC-06(4 pin NON-BLE) modules
// 2018 - Gord Payne
// BLE (Bluetooth Low Energy) is required for iOS devices
// or newer Android devices with built-in Bluetooth Low Energy
// HC-06 is for older Android devices only

// Recommended App Button Mappings
// Android App - Arduino Bluetooth Controller (ABC) by Ioannis Tzanellis
//           Works with HC-06. MAY NOT WORK with BLE modules
//           For BLE modules, may need a different APP
//
// iOS App - BLE Joystick - Works with BLE modules only
//
//
// App Button Mappings - these will work with either App
//
//           character
//           sent by App
// UP ARROW - forward           a
// LEFT ARROW - forward left    d
// RIGHT ARROW - forward right  b
// DOWN ARROW - reverse         c
// TRIANGLE BUTTON             e (rotate left BLE JoyStick) not used on Android app
// CROSS BUTTON - stop          g
// SQUARE BUTTON - reverse left h
// CIRCLE BUTTON - reverse right f
// SELECT BUTTON - rotate left  i not BLE JoyStick app
// START BUTTON - rotate right  j not BLE JoyStick app
```

```

#include <SoftwareSerial.h>
SoftwareSerial mySerial(8, 7); // TX, RX
// wired connections

// button mapping values
const char fwd = 'a';
const char forL = 'd';
const char forR = 'b';
const char rev = 'c';
const char rotL = 'e'; // rotate left on iOS app only
const char stp = 'g';
const char revL = 'h';
const char revR = 'f';
const char rotL = 'i'; // rotate left on android app
const char rotR = 'j'; // rotate right on android app

int theChar = ' '; // character received from Android device
int lastChar = '/';
int driveChar = '5';
#define aIa 3 // Motor A Input A --> MOTOR B + LEFT from top
#define aIb 2 // Motor B Input B --> MOTOR B -
#define bIa 5 // Motor A Input A --> MOTOR A + RIGHT from top
#define bIb 4 // Motor A Input B --> MOTOR A -
// functional connections
#define motorBpwm bIa // Motor B PWM Speed
#define motorBdir bIb // Motor B Direction
#define motorApwm aIa // Motor A PWM Speed
#define motorAdir aIb // Motor A Direction

```

```
void setup() {
  Serial.begin(9600); // for diagnostics if needed
  mySerial.begin(9600); // the Bluetooth channel
  pinMode(bIa, OUTPUT);
  pinMode(bIb, OUTPUT);
  pinMode(aIa, OUTPUT);
  pinMode(aIb, OUTPUT);
  pinMode( motorBdir, OUTPUT );
  digitalWrite( motorBdir, LOW );
  analogWrite( motorBpwm, 0 );
  pinMode( motorAdir, OUTPUT );
  digitalWrite( motorAdir, LOW );
  analogWrite( motorApwm, 0);
  delay(150);
}
```

```

void loop()
{
  if (mySerial.available() > 0) { // if a character has been sent from the App
    theChar = mySerial.read(); // read the character
    if (theChar != lastChar) { //if it's a different character, update lastChar. Otherwise, ignore it.
      lastChar = theChar;

      // display the sent character from the App in the Serial Monitor
      // these characters are for Android devices/Apps. For iOS, consult the information inside the App

      Serial.println(theChar);

      // now do robot action based on the command sent from the App
      switch (theChar) {
        case revL: // square button
          reverseLeft();
          break;
        case rev:
          reverse();
          break;
        case revR: // circle button
          reverseRight();
          break;
        case rotI: // triangle button on BLE Joystick app only
          rotateLeft();
          break;
        case rotL:
          rotateLeft(); // 'select' button on Android app only
          break;
        case rotR:
          rotateRight(); // 'start' button on Android app only
          break;
      }
    }
  }
}

```



```
    case stp:// 'x' button
        allStop();
        break;
    case forL:
        forwardLeft();
        break;
    case fwd:
        forward();
        break;
    case forR:
        forwardRight();
        break;
    default:
        break;
}
}
}
```

```
void allStop() { // stop the robot
  Serial.println("stop");
  digitalWrite(motorBdir, LOW);
  analogWrite( motorBpwm, 0);
  digitalWrite( motorAdir, LOW);
  analogWrite( motorApwm, 0);
}

void forward() { // drive forward
  Serial.println("forward");
  analogWrite(motorBpwm, 220); // don't use numbers bigger than 240
  analogWrite(motorApwm, 220);
  digitalWrite(motorBdir, LOW);
  digitalWrite( motorAdir, LOW);
}

void reverse() { // drive backward
  Serial.println("reverse");
  analogWrite(motorBpwm, 30); // notice in reverse, the speeds are OPPOSITE
  analogWrite(motorApwm, 30); // don't use speeds smaller bigger than 100
  digitalWrite(motorBdir, HIGH);
  digitalWrite( motorAdir, HIGH);
}
```

```
void forwardRight() {  
    // Serial.println("forward right");  
    analogWrite(motorBpwm, 100);  
    analogWrite(motorApwm, 200);  
    digitalWrite(motorBdir, LOW);  
    digitalWrite(motorAdir, LOW);  
}
```

```
void forwardLeft() {  
    // Serial.println("forward left");  
    analogWrite(motorBpwm, 200);  
    analogWrite(motorApwm, 100);  
    digitalWrite(motorBdir, LOW);  
    digitalWrite(motorAdir, LOW);  
}
```

```
void reverseLeft() {  
    // Serial.println("reverse left");  
    analogWrite(motorBpwm, 20);  
    analogWrite(motorApwm, 100);  
    digitalWrite(motorBdir, HIGH);  
    digitalWrite(motorAdir, HIGH);  
}
```

```
void reverseRight() {  
    // Serial.println("reverse right");  
    analogWrite(motorBpwm, 100);  
    analogWrite(motorApwm, 20);  
    digitalWrite(motorBdir, HIGH);  
    digitalWrite(motorAdir, HIGH);  
}
```

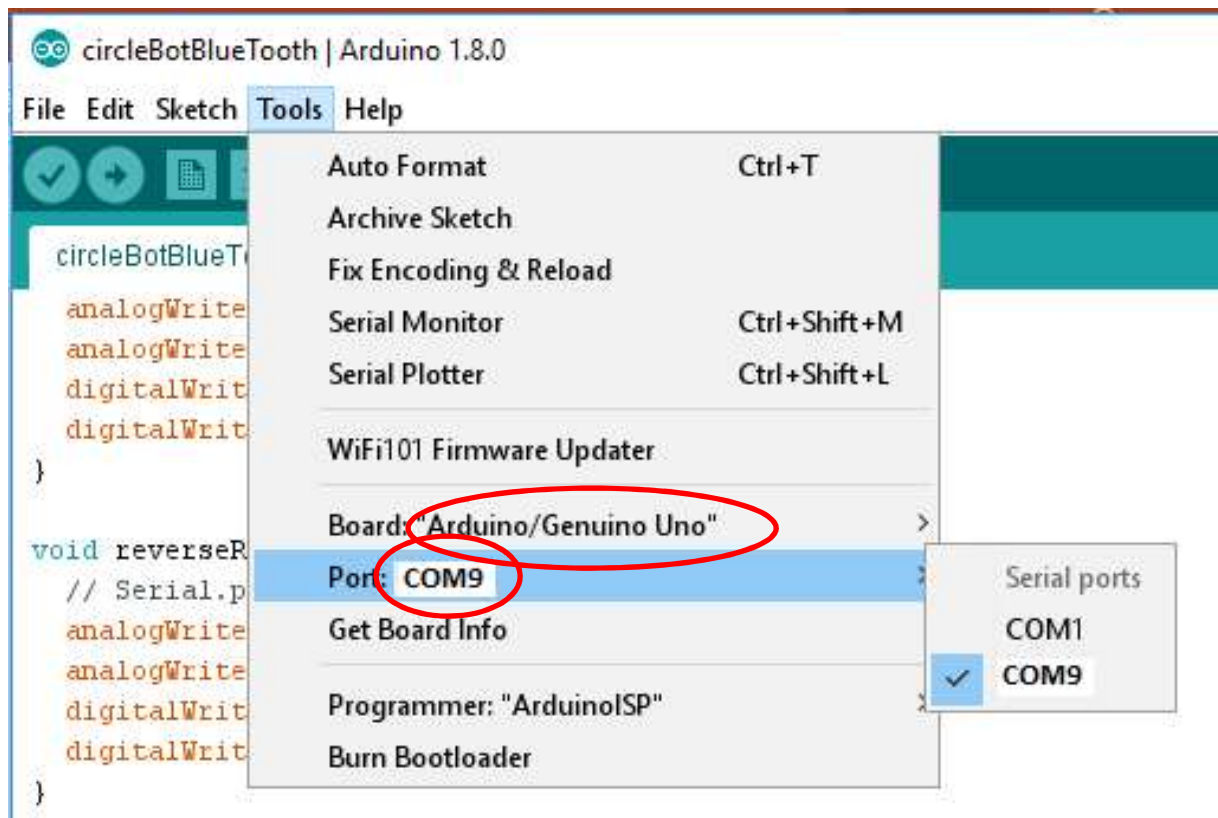
```
void rotateRight() {  
    // Serial.println("rotate right");  
    analogWrite(motorBpwm, 150);  
    analogWrite(motorApwm, 150);  
    digitalWrite(motorBdir, LOW);  
    digitalWrite(motorAdir, HIGH);  
}
```

```
void rotateLeft() {  
    // Serial.println("rotate left");  
    analogWrite(motorBpwm, 150);  
    analogWrite(motorApwm, 150);  
    digitalWrite(motorBdir, HIGH);  
    digitalWrite(motorAdir, LOW);  
}
```

SAVE YOUR WORK!!!

Uploading your code

Go to the Tools Menu. Select the **Board** and the **Port** as below:



NOTE: for school computers, NEVER choose COM1 or COM3.

Click the Right Arrow button at the top left of the Arduino screen.



Arduino will attempt to compile and upload your code to the Arduino board on the robot. You'll see a green progress bar in the lower right of the screen.

If you entered the code correctly, you'll see 'Done uploading.' in the bottom left of the Arduino screen.

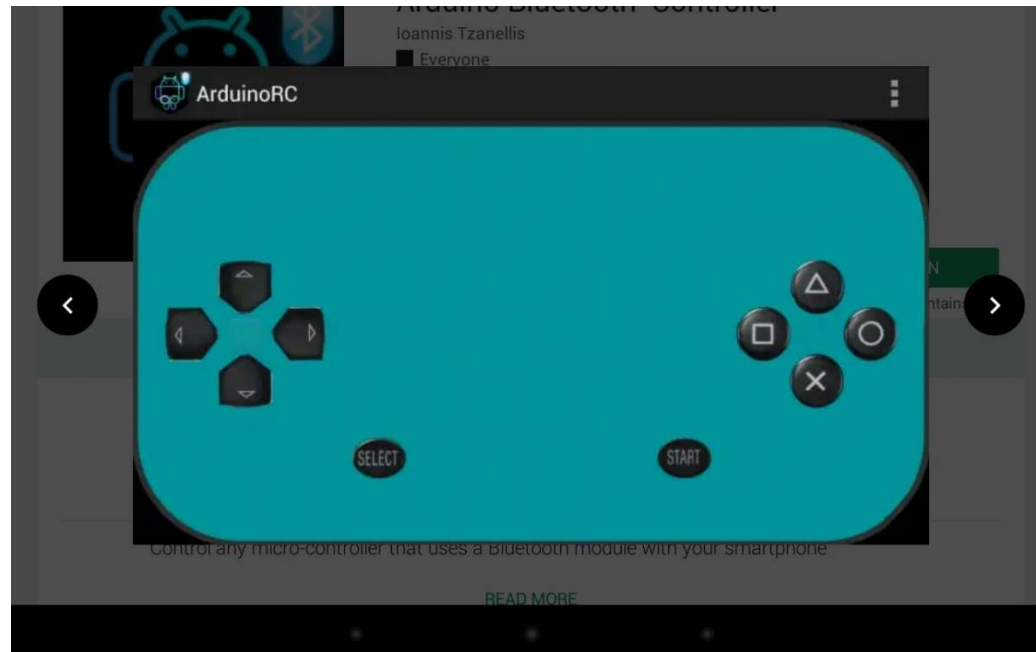
If not, you have some debugging to do! 😊

Setting up the App

Android App

A recommended app is

‘Android Bluetooth Controller’

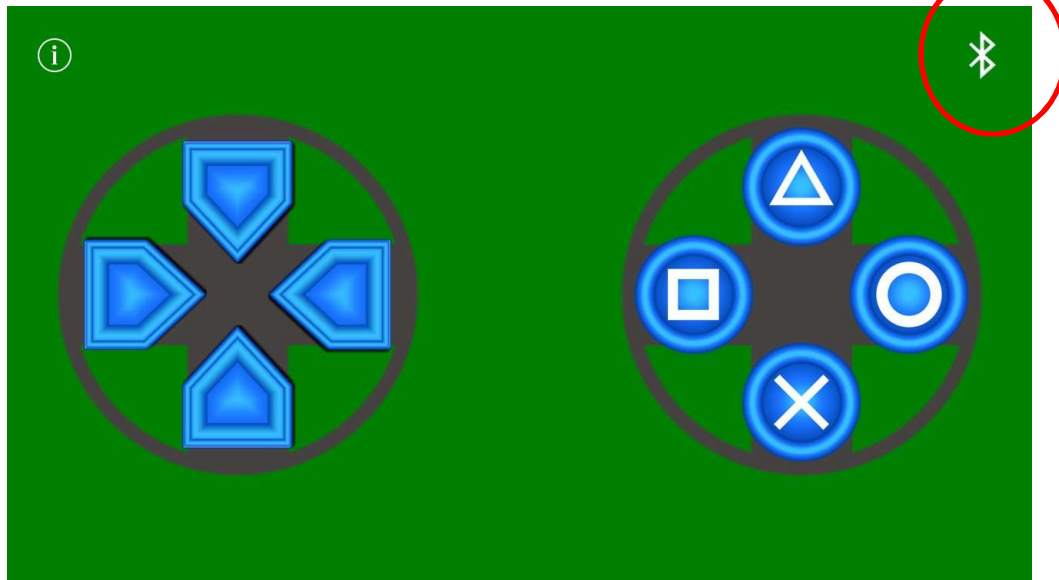


Pair the app to the Bluetooth module and then select ‘Controller Mode’.
In the top right corner you’ll be able to choose the ‘Set Commands’ option and assign the different characters to be sent when each button is pressed. (see the Arduino code for the characters for each robot function)

iOS App

A recommended app is

'BLE Joystick'



Just click the 'Bluetooth' icon in the top right and select your Bluetooth module. Then start driving your robot!

Some Ideas for Adding to your Robot

- Add an SH-04 distance device and have the robot stop when it encounters an obstacle
- Add a second Bluetooth module(just need another SoftwareSerial object line on new pins) and you can have a second phone control more devices (servos, LEDs etc)
- Attach a phone to the bot and use Facetime(or other) to do FPV driving (like a Mars Rover!)
- Endless possibilities!!!

**NOW GO OUT AND MAKE
SOMETHING WONDERFUL!!!**

