**Group Member Names:** Josh Hanenburg, Nick Hongo, Vinh Bui, Weixun Wang

**Course and Quarter:** ENGR 114, Fall 2018

**Date:** December 11th, 2018

**Revision Number:** 1

## **Parachute Release Mechanism**

#### **Problem Statement:**

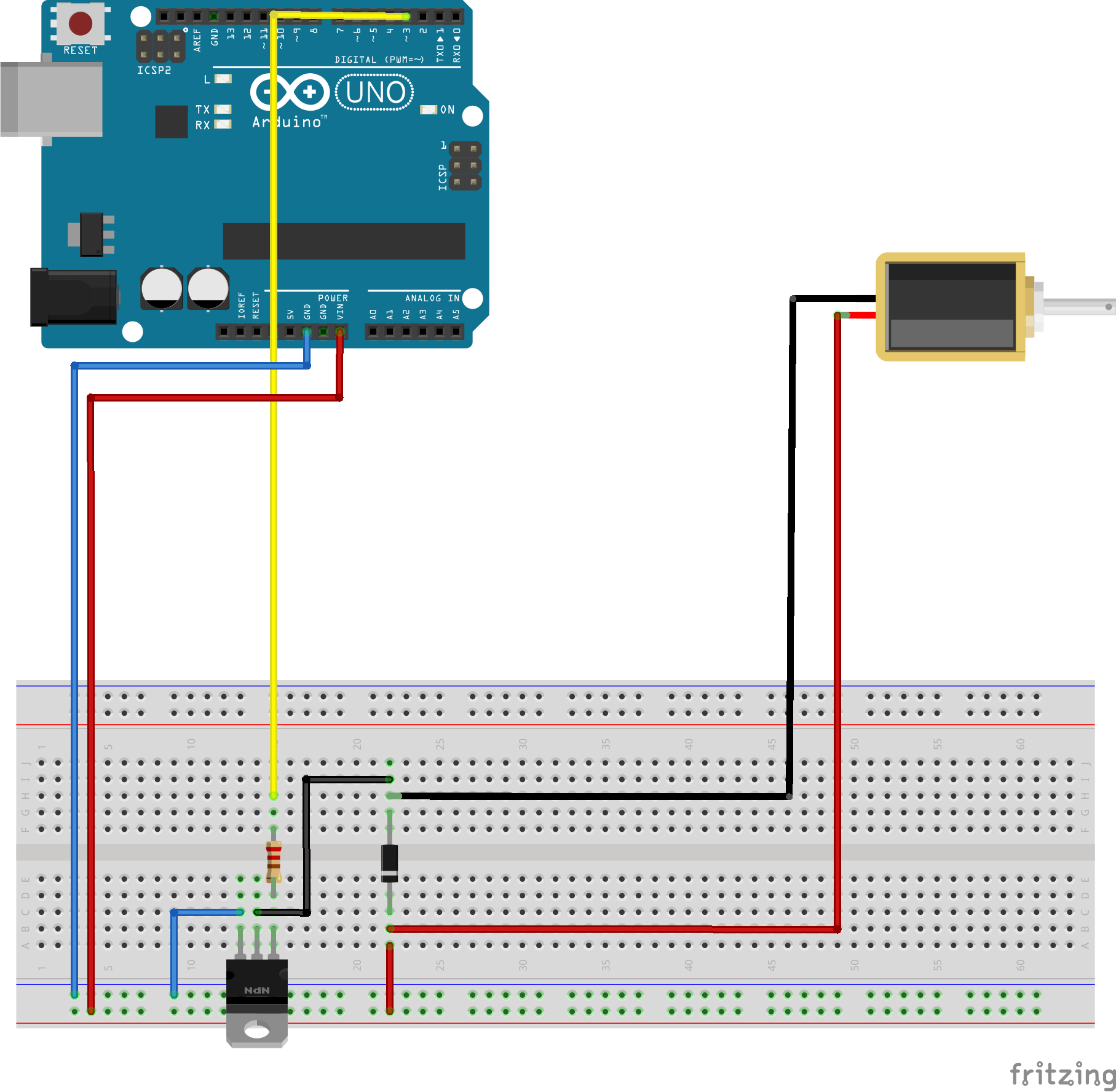
Our team was tasked with designing and building a dropping/release mechanism for the ENGR 101 parachute project. Our team integrated MATLAB and Arduino software to control an Arduino RedBoard, and solenoid to release a parachute by activation through MATLAB’s command window prompts.

**Hardware Setup:**

Bill of Materials:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Part Name | Purpose | Item Name | URL | Price |
| Arduino  (Received from ENGR 114 Class) | It can connect with a computer or laptop and run with Arduino and MATLAB code to operator sensors and/or lights | SparkFun RedBoard - Programmed with Arduino | https://www.sparkfun.com/products/13975 | $ 19.95 |
| Breadboard  (Received from ENGR 114 Class) | A solderless board that allows easy connection between hardware components for prototyping | Breadboard - Self-Adhesive (White) | https://www.sparkfun.com/products/12615 | $ 4.95 |
| USB cable  (Received from ENGR 114 Class) | Transfers data from a computer or laptop to the red board and also can supply 1.5 volts of power to the red board | SparkFun USB Mini-B Cable - 6 Foot | https://www.sparkfun.com/products/11301 | $ 3.95 |
| Job box  (Received from ENGR 114 Class) | Storage box for hardware for easy transport and protection from weather | Job Box - Enclosure | https://www.sparkfun.com/products/11366 | $ 8.95 |
| Solenoid  (Received from ENGR 114 Class) | Open/close to release a parachute | Large push-pull solenoid | https://www.adafruit.com/product/413 | $ 14.95 |
| Jumper wires  (Received from ENGR 114 Class) | The core help connect the power from red board to breadboard and breadboard to the solenoid | Jumper Wires Standard 7" M/M - 30 AWG (30 Pack) | https://www.sparkfun.com/products/11026 | $ 1.95 |
| Diode  (Received from ENGR 114 Class) | Eliminates transient voltages caused when a magnetic coil (such as those found in a motor, relay, or solenoid) suddenly loses power. Without this diode in place, the transient voltage spikes can damage other elements of the circuit. | Schottky Diode | https://www.sparkfun.com/products/10926 | $ 0.15 |
| Power transistor  (Received from ENGR 114 Class) | For controlling the solenoid and the power from the low to high power | FQP30N06L 30N06L 30N06 | https://www.aliexpress.com/item/Free-shipping-FQP30N06L-30N06L-in-stock/32234508850.html?aff\_platform=promotion&cpt=1544242252747&sk=ZRFybm6&aff\_trace\_key=cdd48b5365ad424f889402622d51e227-1544242252747-04070-ZRFybm6&terminal\_id=0e86cdc9f29642a9b35965df5e412761 | $ 17.98 for 100 pieces |
| 330 Ohm resistor  (Received from ENGR 114 Class) | Commonly used in breadboards and perf boards, these 330 Ohm resistors make excellent for solenoid current limiters and are great for general use | Resistor 330 Ohm 1/6 Watt PTH - 20 pack | https://www.sparkfun.com/products/11507 | $ 0.95 |

**Hardware Schematic:**



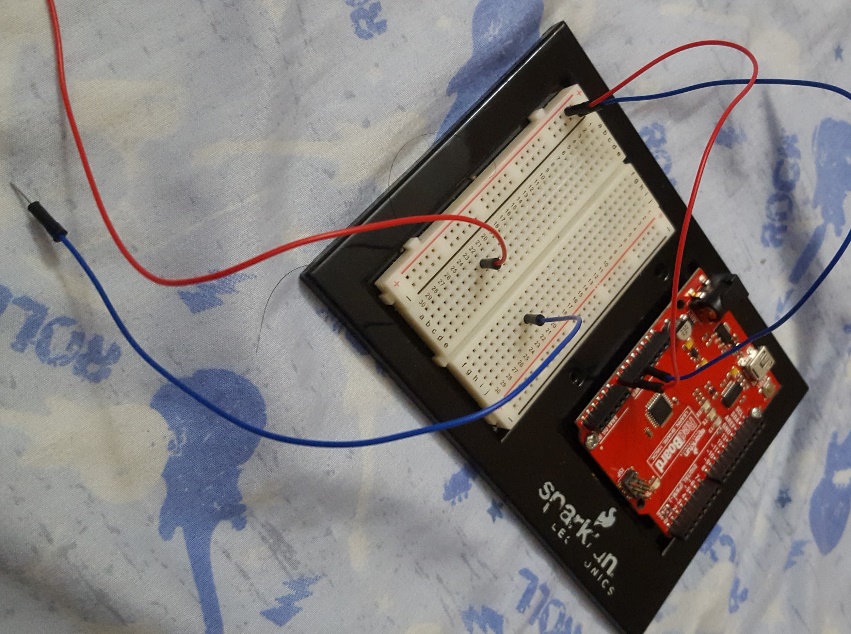
**Hookup Guide:**

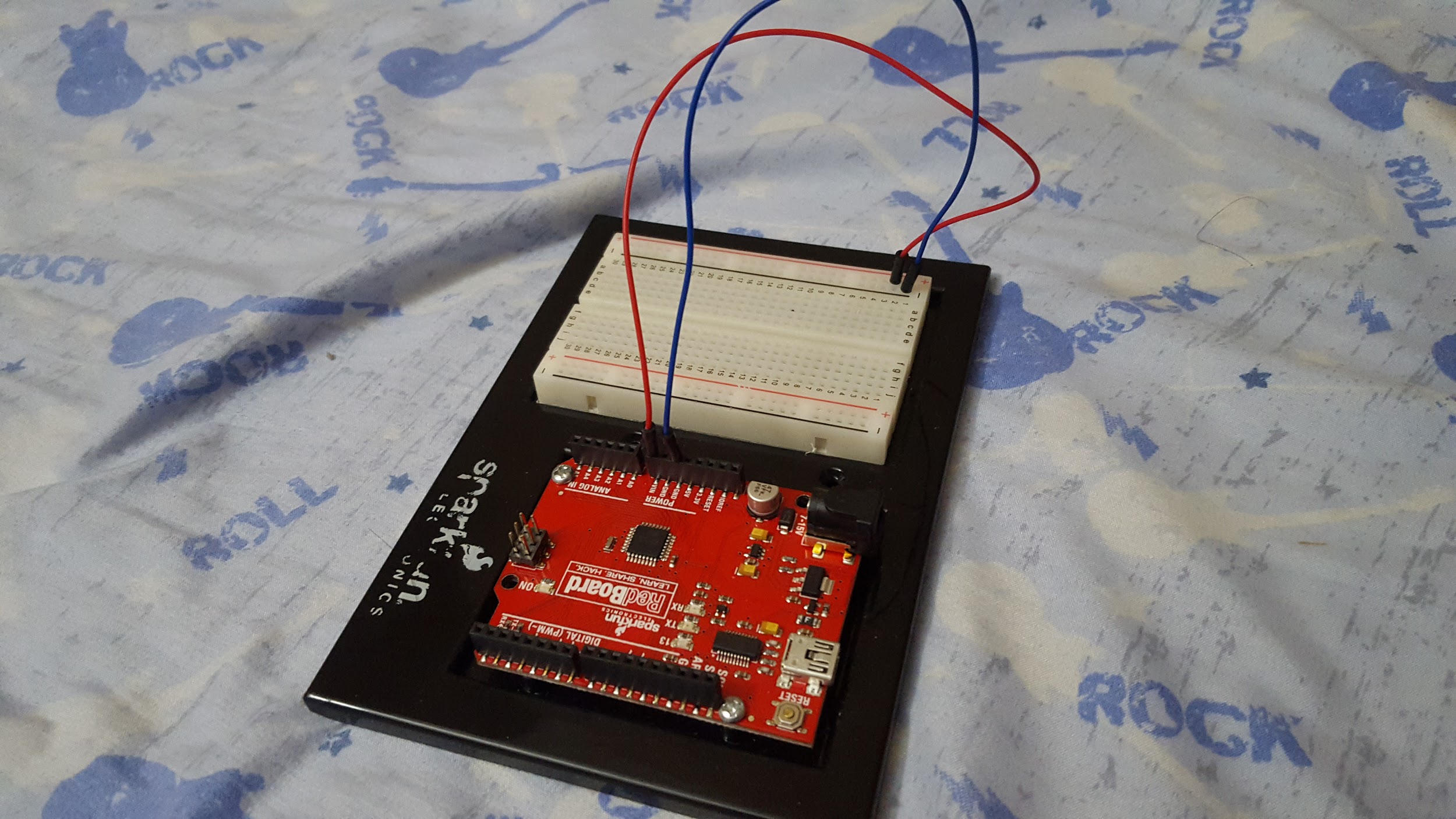
**Step 1- Powering the breadboard**

Start by connecting one of the jumper wires to the “VIN” pin located next to the GND (ground) on the RedBoard and running it over to the positive side of breadboard. Then run a wire from the GND pin on RedBoard over to negative rail on the breadboard.

**Step 2 - Wire Solenoid to Breadboard**

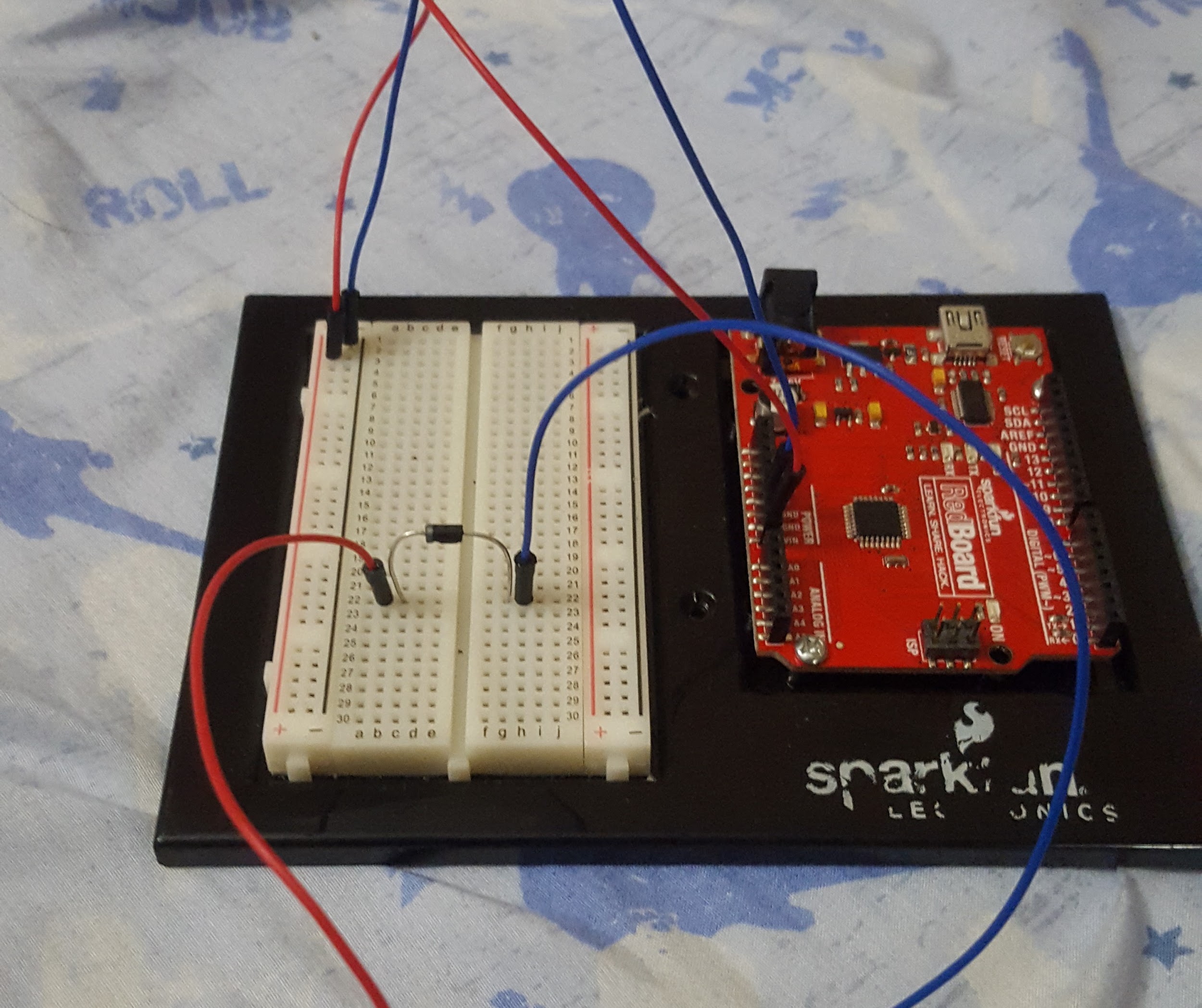
Connect the solenoid jumper wires to the breadboard leaving some space between for the diode to be added. (Positive red PIN 22B, negative black PIN 22H)





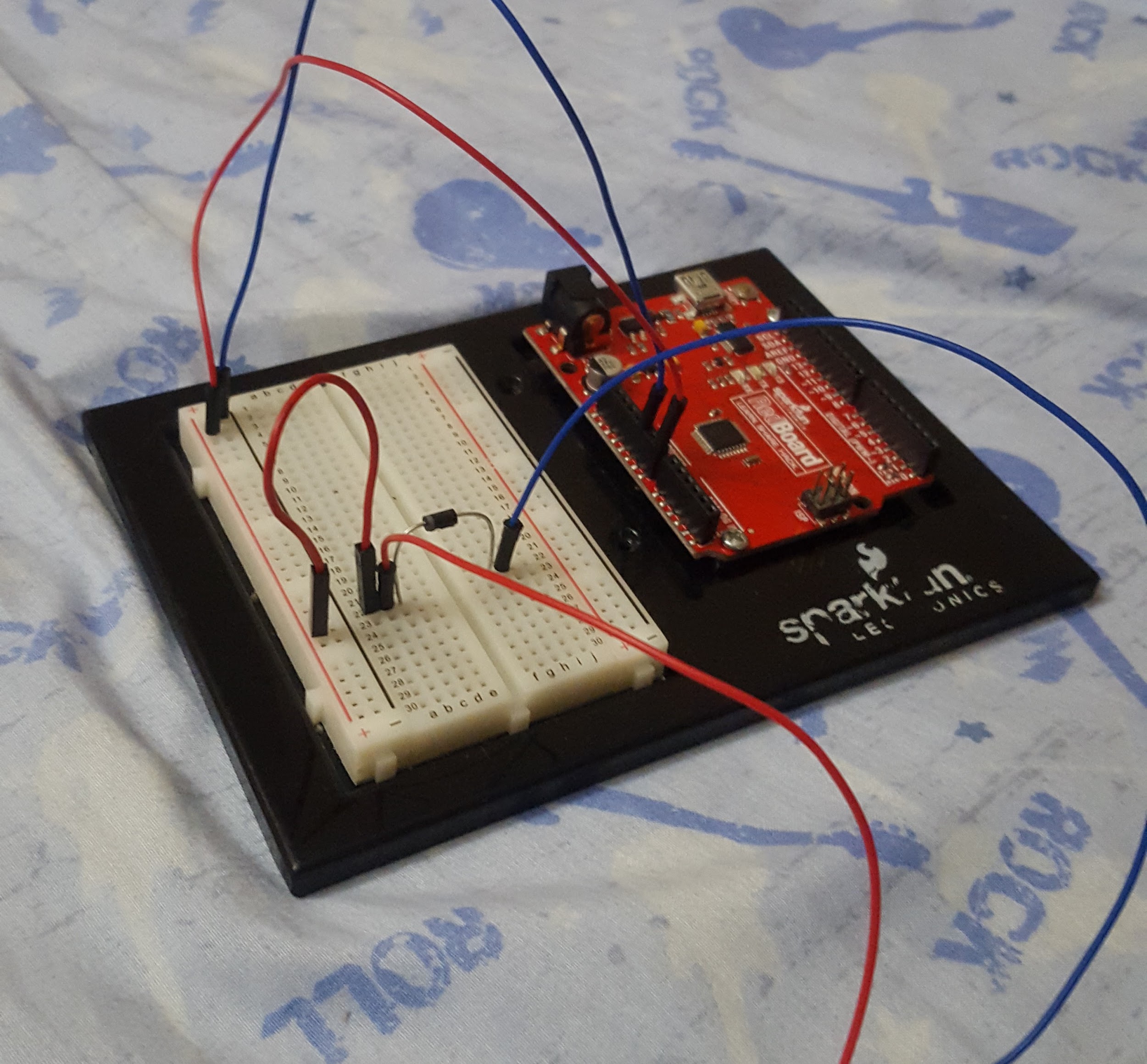
**Step 3 - Place the diode on the breadboard**

The diode is placed from the negative side of the coil to the positive side. Be careful when you connect the diode, it allows current to flow in one direction. Correct installation is crucial, otherwise there will be a short between the power and ground. Make sure the side with the silver stripe is connected to the positive side of power and the other side connected to the negative. (Silver stripe of diode to 22C remaining side to 22G)



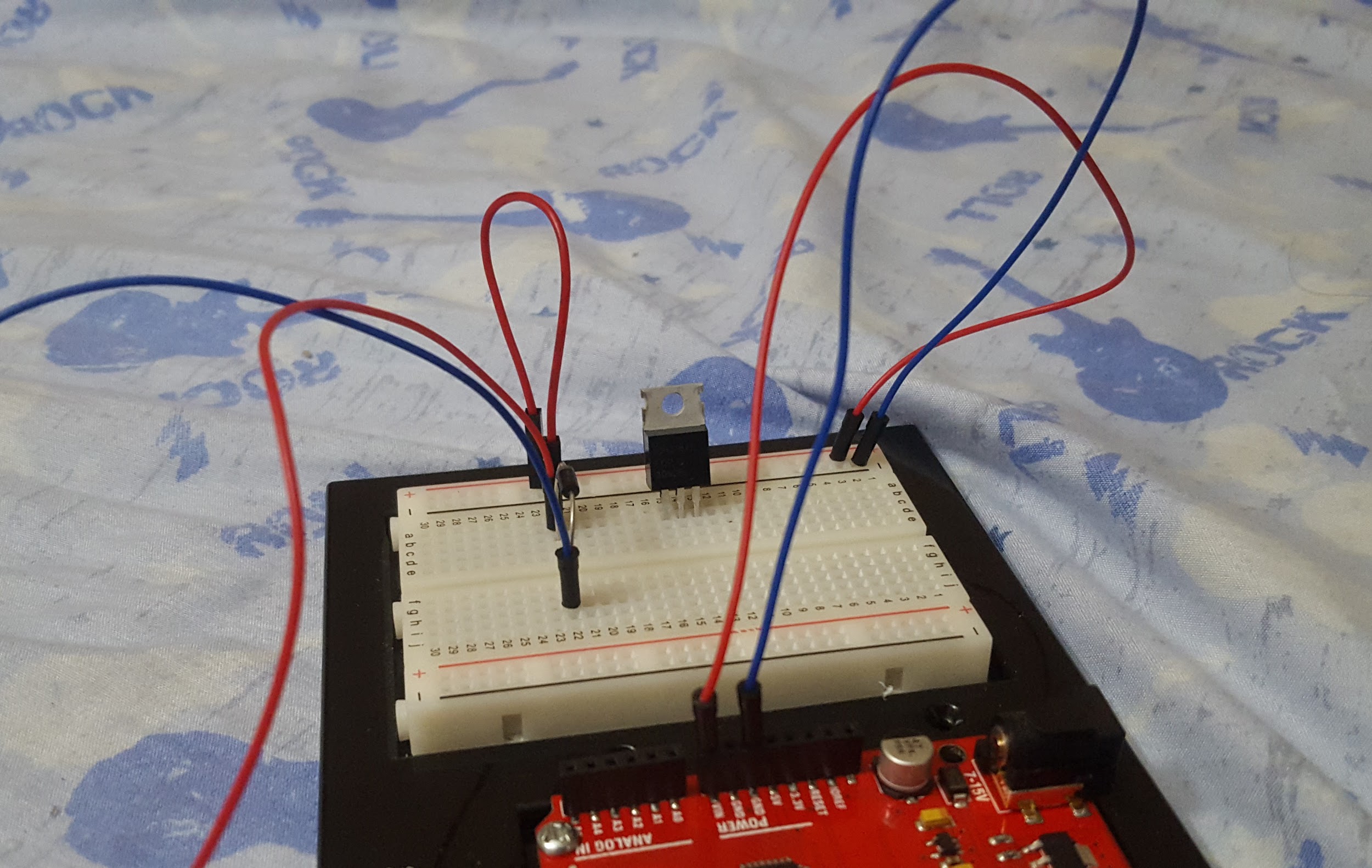
**Step 4 - Power for the solenoid**

On this step we run a jumper wire from the positive power side of the board to the positive solenoid wire. This will power the solenoid and we will be used as the low side to turn the solenoid on or off. (Positive rail to PIN 22A)



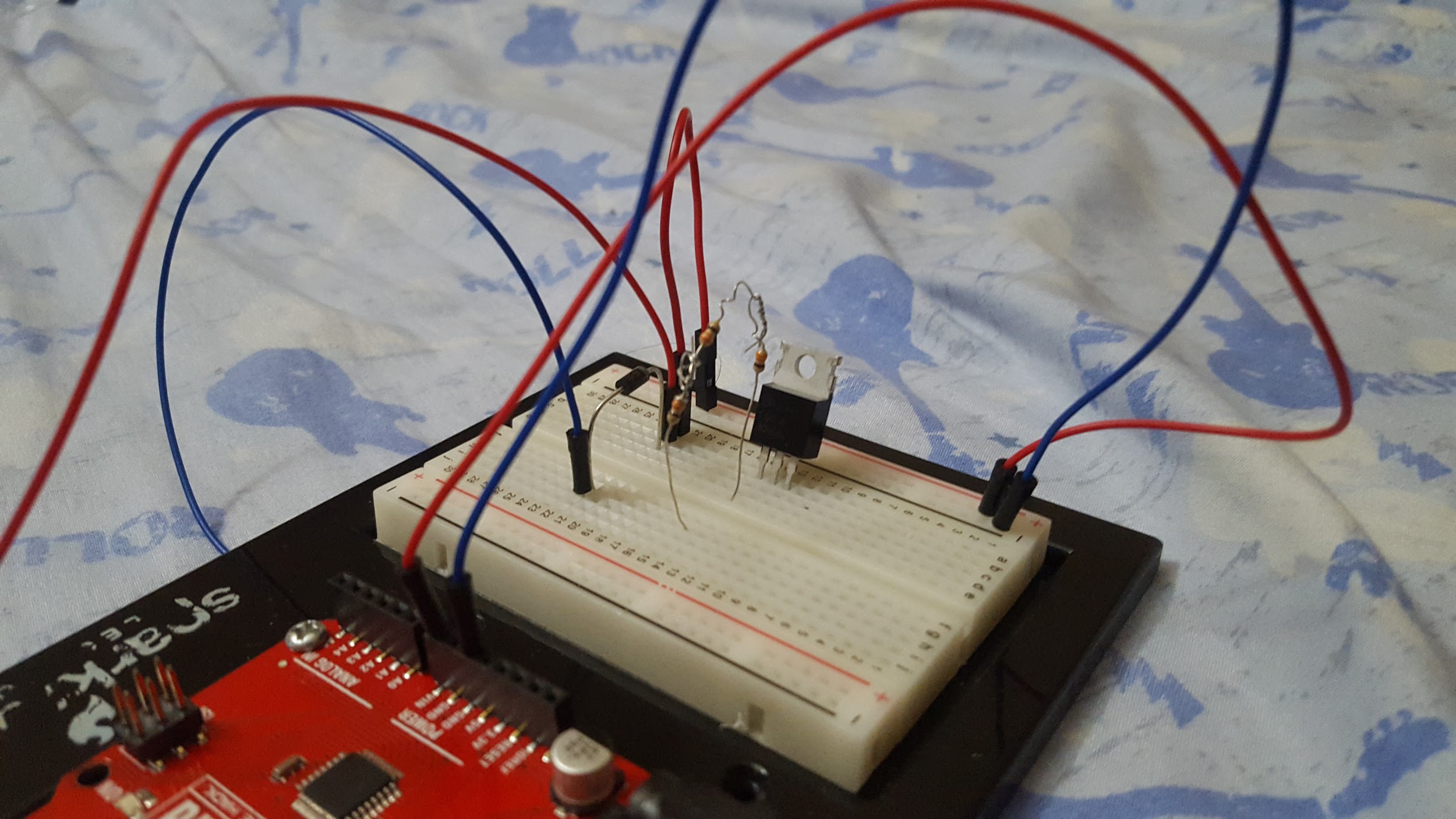
**Step 5 - Placing the transistor on the breadboard**

The current draw of the solenoid is higher than a standard transistor can handle, therefore we will be using a transistor, P/N FQP30N06L. (PIN’s B13-B15, metal tab facing outside of board towards A PIN’s)



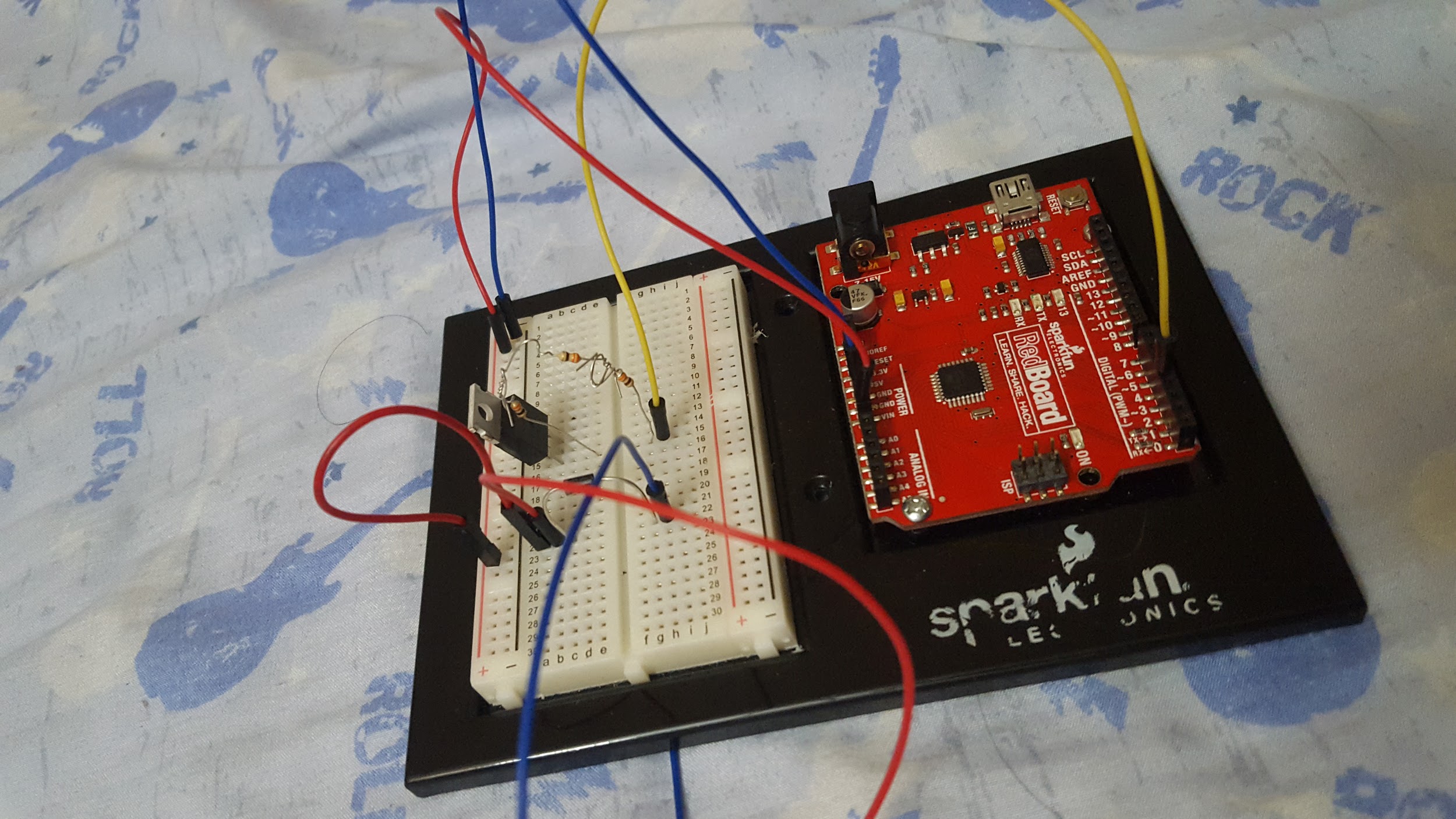
**Step 6 - Resistor connection**

Connect the resistors (3 - 330 Ohm resistors in a series) from the base of the resistor to the other side of the board. (PIN 15D to 15F)



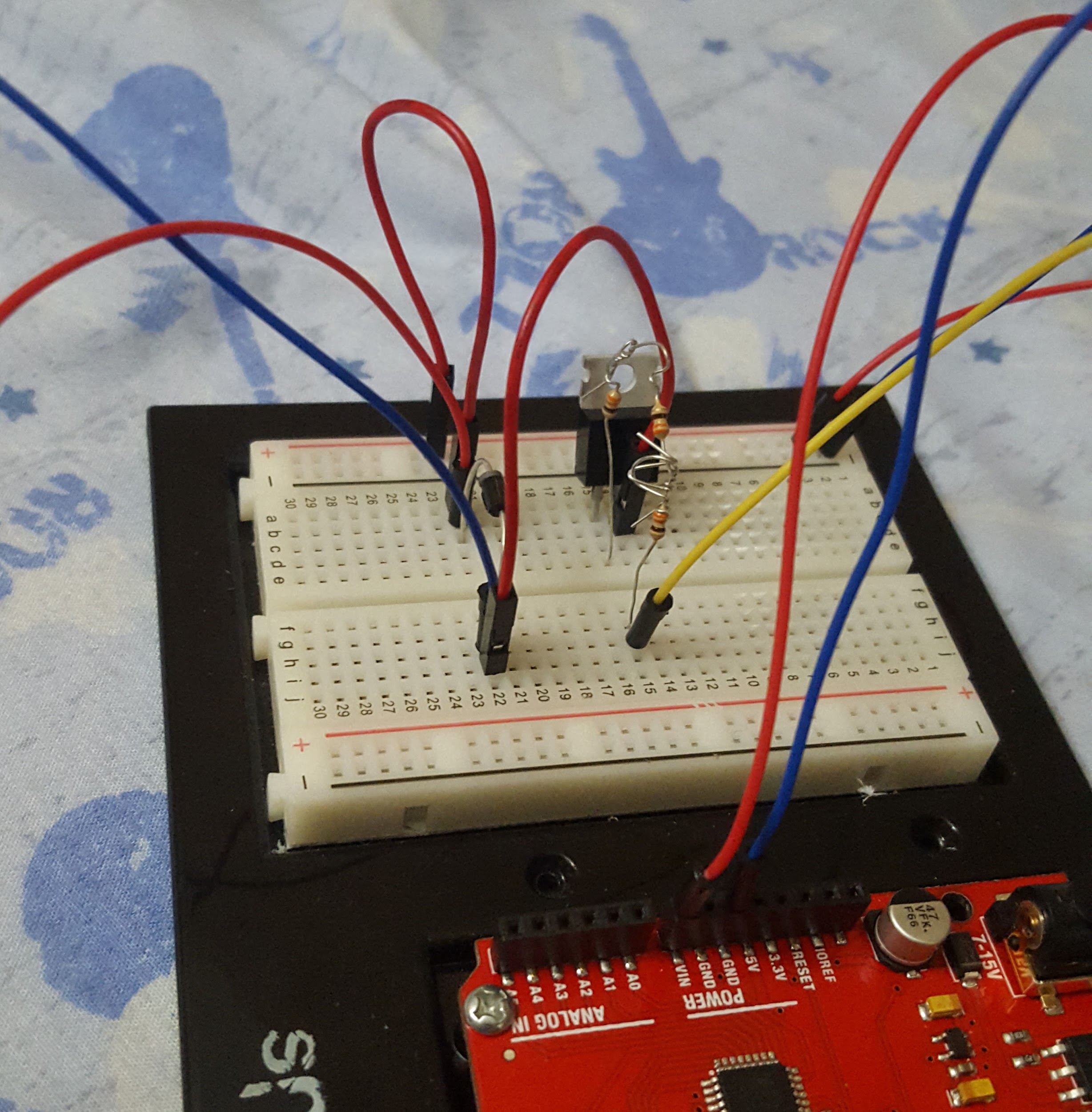
**Step 7- Connecting resistor to the breadboard**

Now that the current limiting resistors are in place, we can connect the it to pin 4 on digital line of RedBoard. (Jumper wire from PIN 15H to PIN 4 digital line of RedBoard)



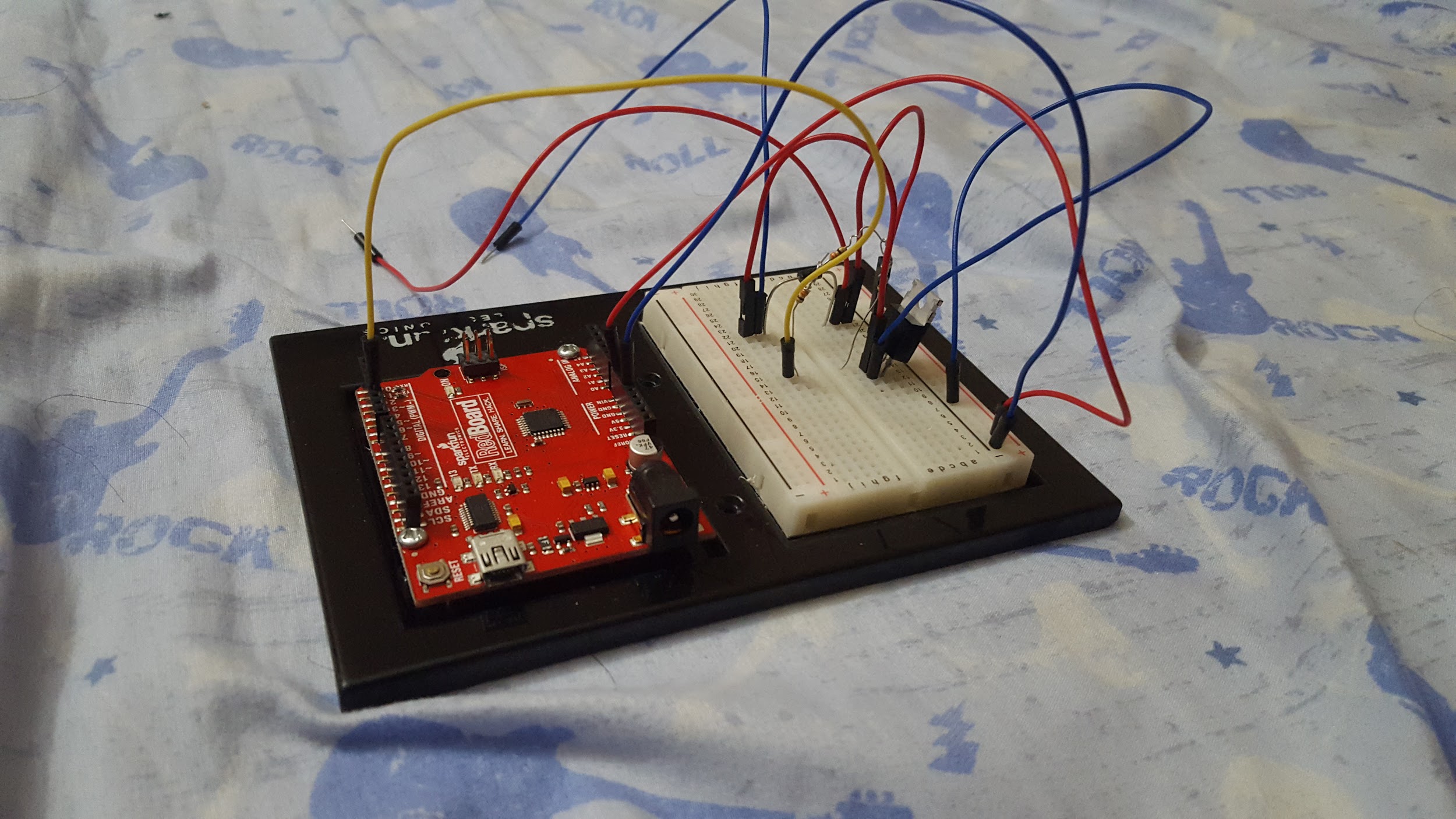
**Step 8 - Connect negative wire to solenoid**

Next, we connect the negative wire of the solenoid to middle of the transistor with a jumper wire. (PIN 14C to 22I)



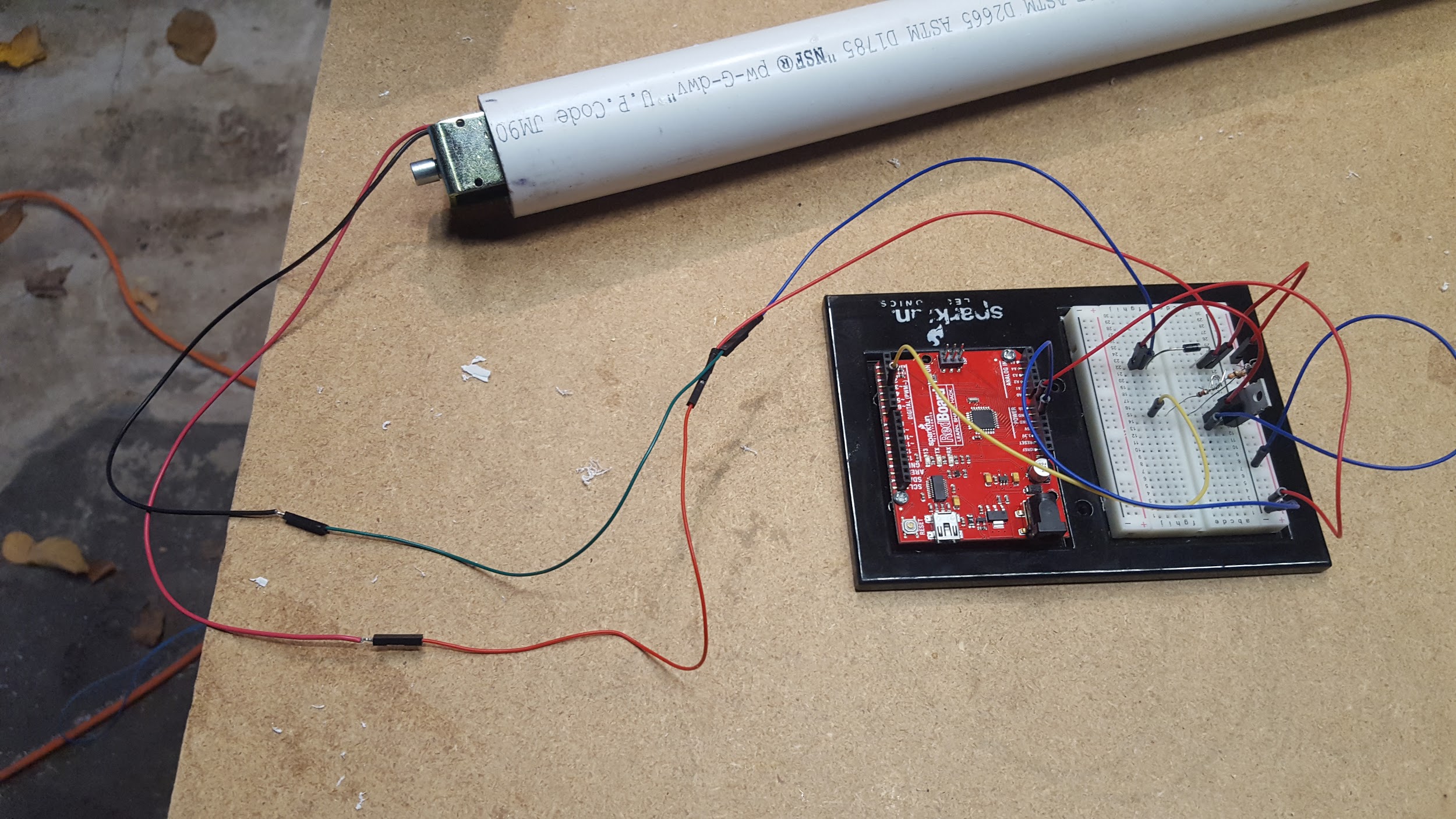
**Step 9 - Connecting the transistor to the ground**

This step we connect the negative power on breadboard to PIN 13B of the transistor. (PIN 13B to negative rail)



**Step 10 - Connecting solenoid and external power**

Now we can connect solenoid to the solenoid wires that come from the board, verifying strong connections are made to the jumper wires. (Positive red from solenoid to jumper wire PIN 22B, negative black from solenoid to 22H)



**Step 11 – Installing hardware into job box**

Place the whole board inside of the job box, holes were made for solenoid wires, USB cable, and external power. At this time, hardware setup is complete.



#### **Code:**

#### **MATLAB Code:**

**%% Final Project - Parachute Release Mechanism**

**% Authors: Josh Hanenburg, Nick Hongo, Vinh Bui, & Weixun Wang**

**% Date: 12/11/2018**

**% Course: ENGR 114, Fall 2018**

**% Description: This MATLAB code interfaces with Arduino software to operate a solenoid to open a release mechanism to drop a parachute. MATLAB software will ask a user to enter 'D' to drop the parachute, 'C' to close the solenoid or 'q' to quit the program.**

**%% Clear command window, workspace variables, figure windows and connected devices**

**clc; % Clears command window  
clear; % Clears all workspace variables  
close all; % Closes all open figure windows  
delete(instrfindall); % Finds all visible and hidden serial port objects and closes them**

**Port = 'COM6'; % Port the Arduino is connected to  
delete(instrfindall); % Deletes any connected ports  
a = serial(Port); % Creates the serial port  
fopen(a); % Opens the serial port**

**% Pause () for 1 second to make sure a connection is made  
pause(1);**

**out = instrfind('Port', Port); % See if the port specified is open  
disp('Serial Port is Open') % Display text to user that the port is open**

**%% Ask for user input, D to open solenoid, L to close solenoid, q for quit**

**while true % Create while statement for true  
 % Ask user for input as a string  
 user\_input = input('Type "D" to open the release mechanism , "C" to close the release mechanism or "q" to quit : ','s');  
 if strcmp(user\_input,'D') % If statement with strcmp() logical statement  
 send\_str = 'D'; % Creates send\_str variable as a string 'D'**

**% fprintf() to send signal to Arduino software and display string text  
 fprintf(a, '%s', send\_str);**

**% Elseif statement with strcmp() logical statement**

**elseif strcmp(user\_input,'C')**

**send\_str = 'C'; % Creates send\_str variable as a string 'C'**

**% fprintf() to send signal to Arduino software and display string text  
 fprintf(a, '%s', send\_str);**

**% Elseif statement with strcmp() logical statement**

**elseif strcmp(user\_input,'q')**

**break % Terminates if statement  
 end % Ends if statement  
end % Ends while statement**

**%% Close the serial port**

**fclose(a); % Closes the serial port  
delete(a); % Deletes serial port from memory  
clear a; % Removes serial port from workspace  
disp('Serial Port is closed') % Displays text to command window**

#### Arduino Code:

**const int ledPin = 13; // the pin that the LED is attached to**

**int incomingByte; // a variable to read incoming serial data into**

**void setup() {**

**// initialize serial communication:**

**Serial.begin(9600);**

**// initialize the LED pin as an output:**

**pinMode(ledPin, OUTPUT);**

**// initialize the solenoid as an output through port 4**

**pinMode(4, OUTPUT);**

**}**

**void loop() {**

**// see if there's incoming serial data:**

**if (Serial.available() > 0) {**

**// read the oldest byte in the serial buffer:**

**incomingByte = Serial.read();**

**// if it's a D (ASCII 72), open the solenoid:**

**if (incomingByte == 'D') {**

**// turns the LED on red board on**

**digitalWrite(ledPin, HIGH);**

**// sets the digital pin 4 on**

**digitalWrite(4, HIGH);**

**}**

**// if it's a C (ASCII 76) close the solenoid:**

**if (incomingByte == 'C') {**

**// turns the LED on red board off**

**digitalWrite(ledPin, LOW);**

**// sets the digital pin 4 off**

**digitalWrite(4, LOW);**

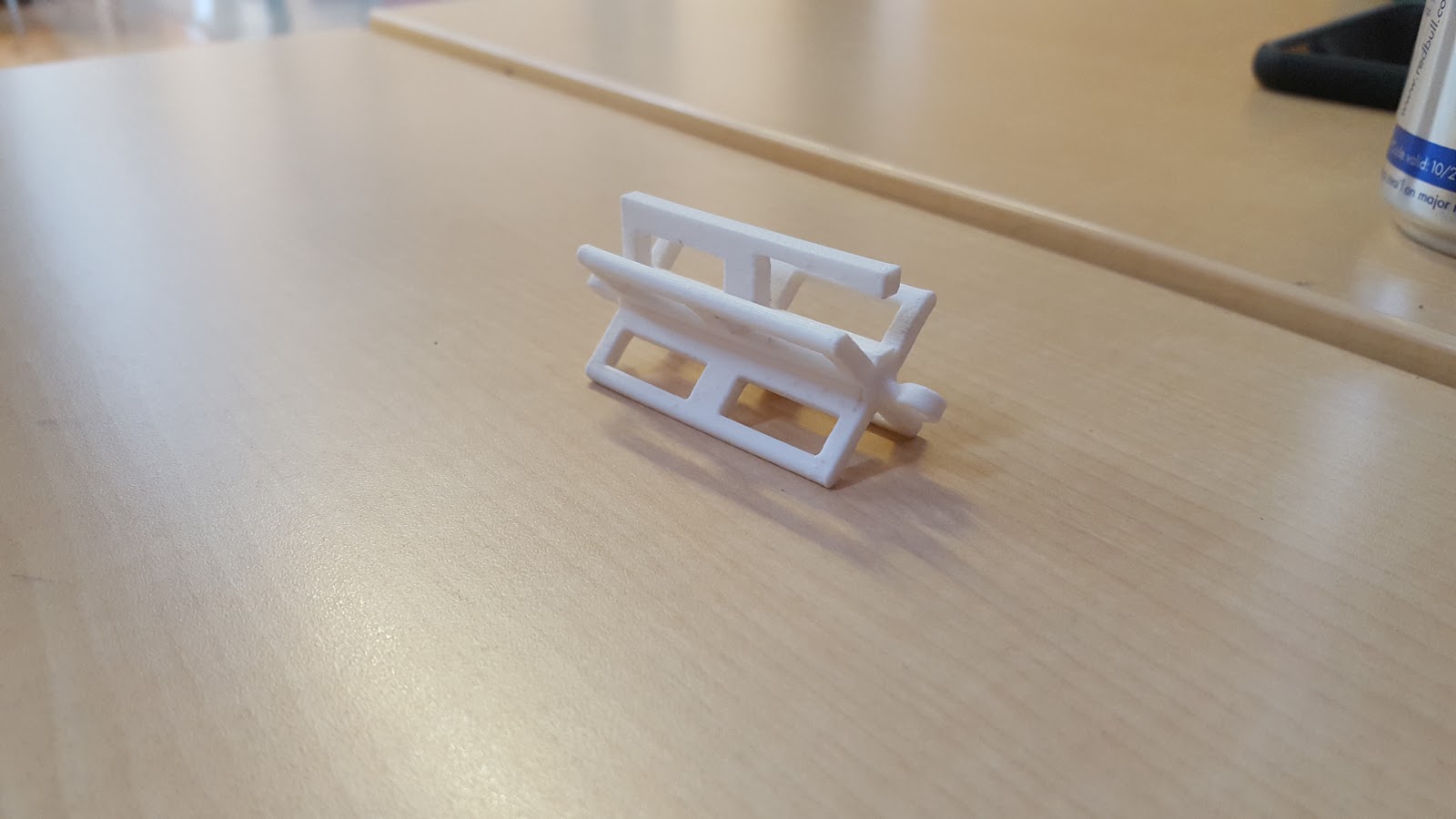
**}**

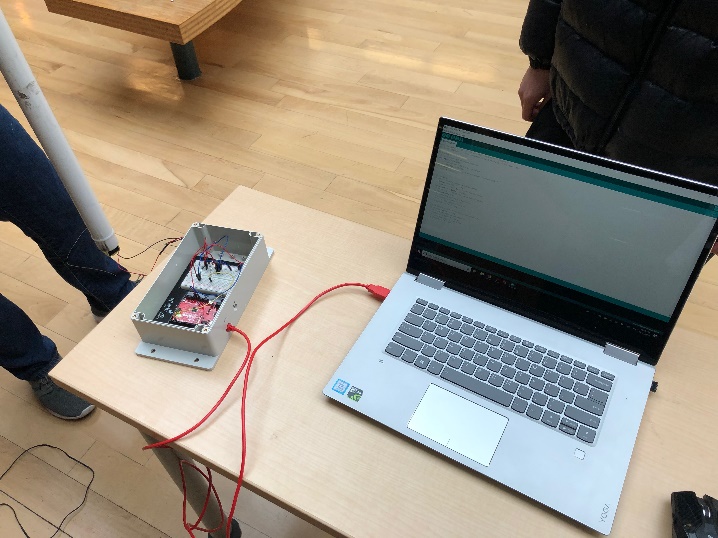
**}**

**}**

#### **Results:**

****

****

****

#### **Future Work:**

Unfortunately, we did not integrate the red button to open/close the solenoid as there was a time delay with the solenoid when the button was pressed. In the future this could be something that could be integrated and perhaps the delay could be removed with MATLAB and/or Arduino coding.

#### **License**

#### GNU General Public License v3.0