n3rfgun

Created by Scott McCulley

Last updated on 2017-04-26 03:09:43 AM UTC

Guide Contents

Overview

It's my son's fault, really. He came to me one day with a YouTube video of someone who modified a Nerf gun to make it fire faster. "Dad, can we make this?"

And, here we are!

Of course, I had to take it to the next level. Not only did we increase the firing speed, we added a bunch of NeoPixels to make it look cool at night, and gave the firing motor several speeds. But, like the "turbo" button on the early PC's, who would ever use it in non-turbo mode?

This particular Nerf gun, the Elite Rapidstrike CD-18 Blaster was purchased online, because it is hard to source at our local Target store. I'm not sure if they are still in production, so fire up your favorite search engine and see what you can find!

Also, stock up on the extended magazine and a couple of hundred darts. You are going to go through them rather quickly!

Parts

Not only are you going to need a Nerf gun to disassemble, but you are going to need a bunch of electronic parts, since we are taking the "brains" out of the blaster, and replacing them with a Trinket.

1 x Trinket
Adafruit Trinket 5 Volt
Buy Now
1 x Programming Switch
Mini Panel Mount Switch
Buy Now
2 x Resistor
2.2k Resistor
Buy Now
3 x Resistor
10k Resistor
Buy Now
1 x Darlington Transistor
TIP120 Power Darlington Transistor
Buy Now
2 x Diode
1N4001 Diode
Buy Now
6 x NeoPixels
NeoPixel Mini
Buy Now
1 x Breadboard
Half-size Breadboard
Buy Now

You will also need some tape, glue, diagonal cutters, soldering equipment, patience, and maybe some padding for the guy who is about to be your target!

I included the Adafruit ID numbers for the parts required. The quantity needed for the project may not exactly match the quantity of parts from Adafruit. For example, you only

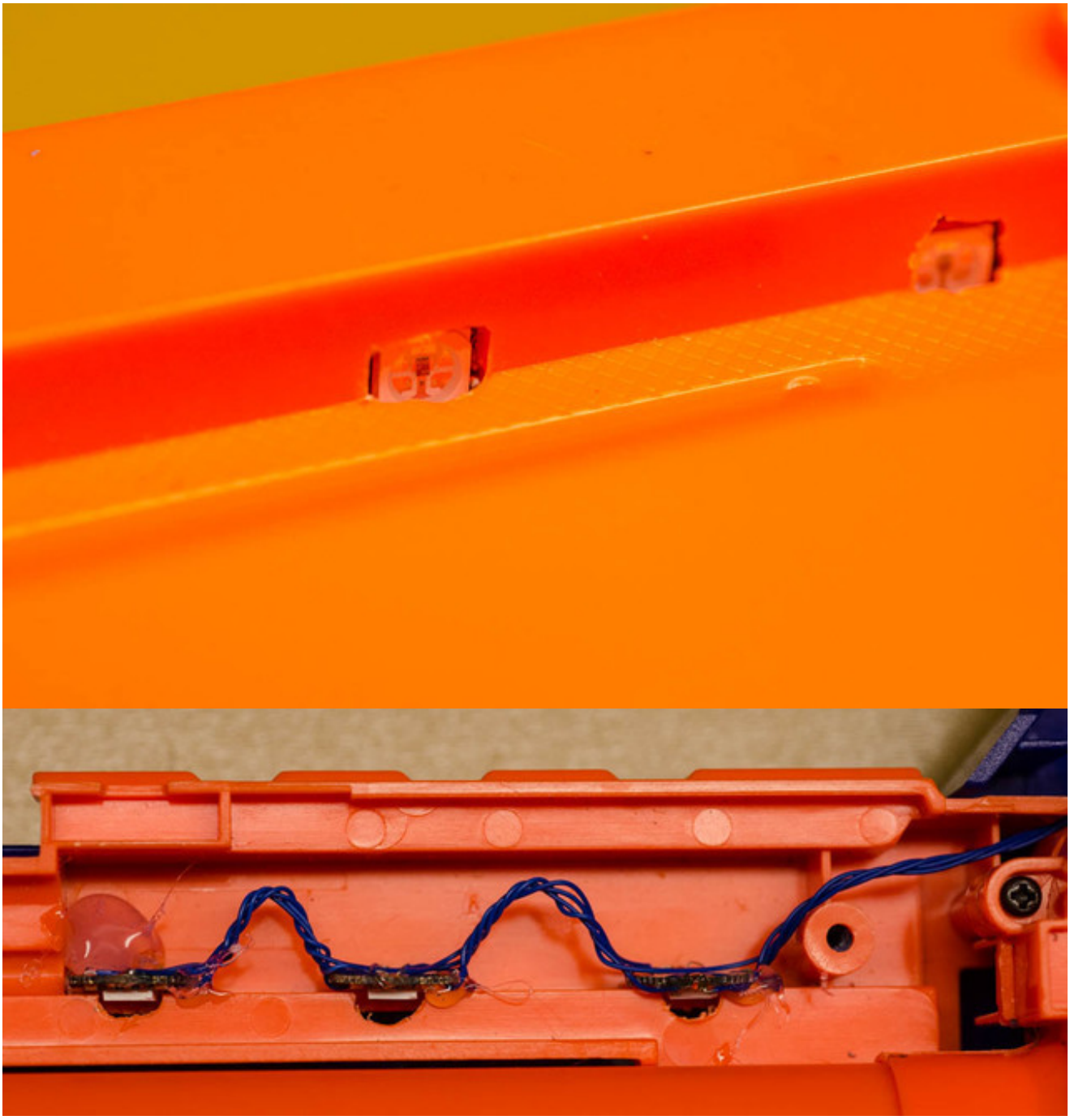
need 2 TIP120, but Adafruit sells them in a 3 pack.

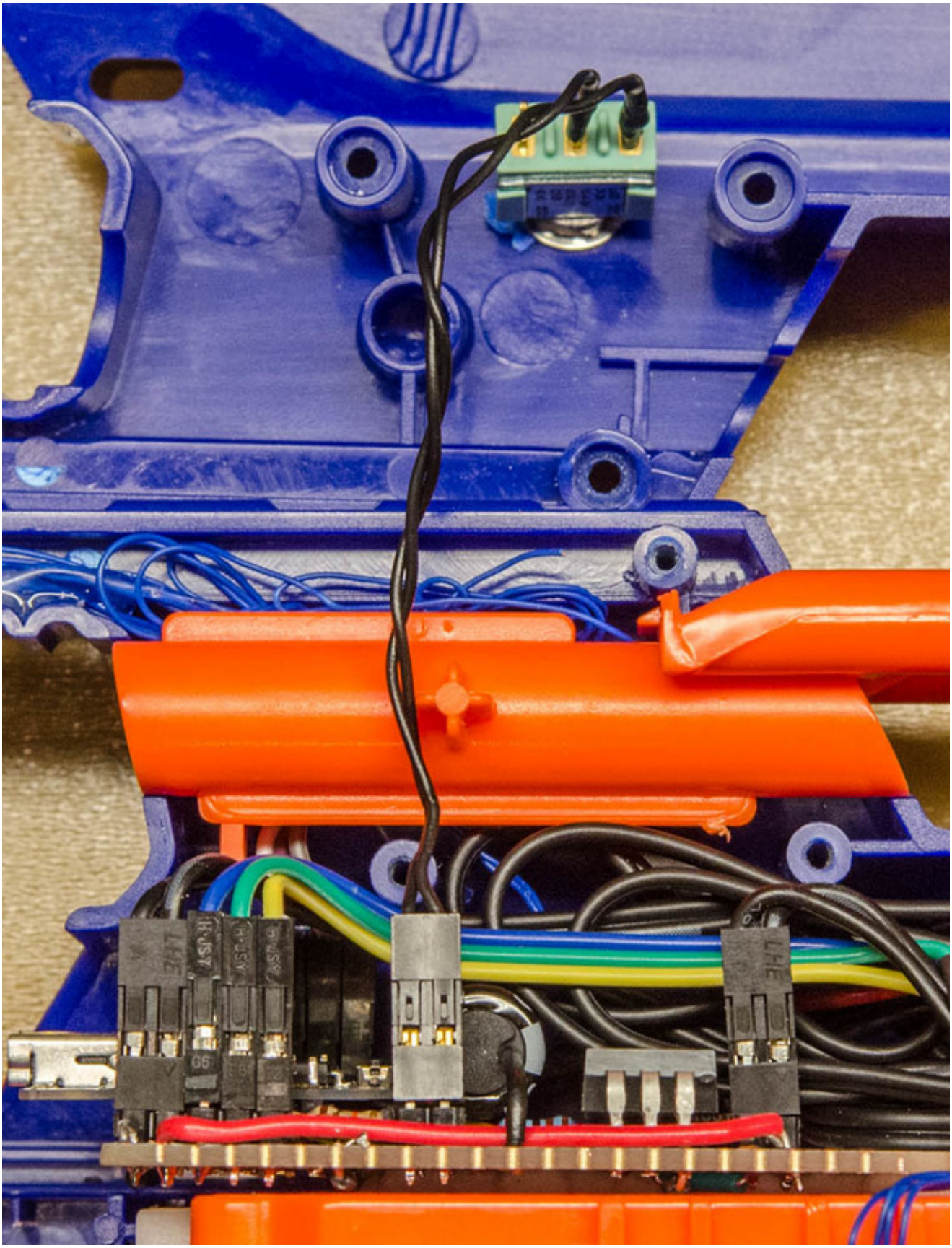
Blaster Hacking

To begin, you need to disassemble the Nerf Blaster, and take out the guts we don't need. Be careful to save the important parts, like the switches and motors. There are several teardown examples on YouTube with different approaches. I suggest reviewing a few of them to get a good idea what is necessary for this project.

1. Disassemble everything, and save all the parts, including screws, springs, switches, motors, and anything else that looks important.
2. Drill 3 holes in each side of the front of the blaster, just above the barrel, but not *through* the barrel. (see image below)
3. Drill a hole for the programming switch. I put it in the side of the case, but you can mount it wherever you want. Just don't interfere with the mechanics of the firing mechanism. (see image below)
4. There may be some other mods to make to the firing mechanism based on what the YouTube videos say. I actually let my son handle that part, so I didn't get any pictures of it. But I think he had to remove a safety interlock somewhere. Basically, if you don't see it in my remaining pictures, it probably shouldn't be there!

That should be all of the physical hacking necessary. The rest is all circuits, wiring, and coding.

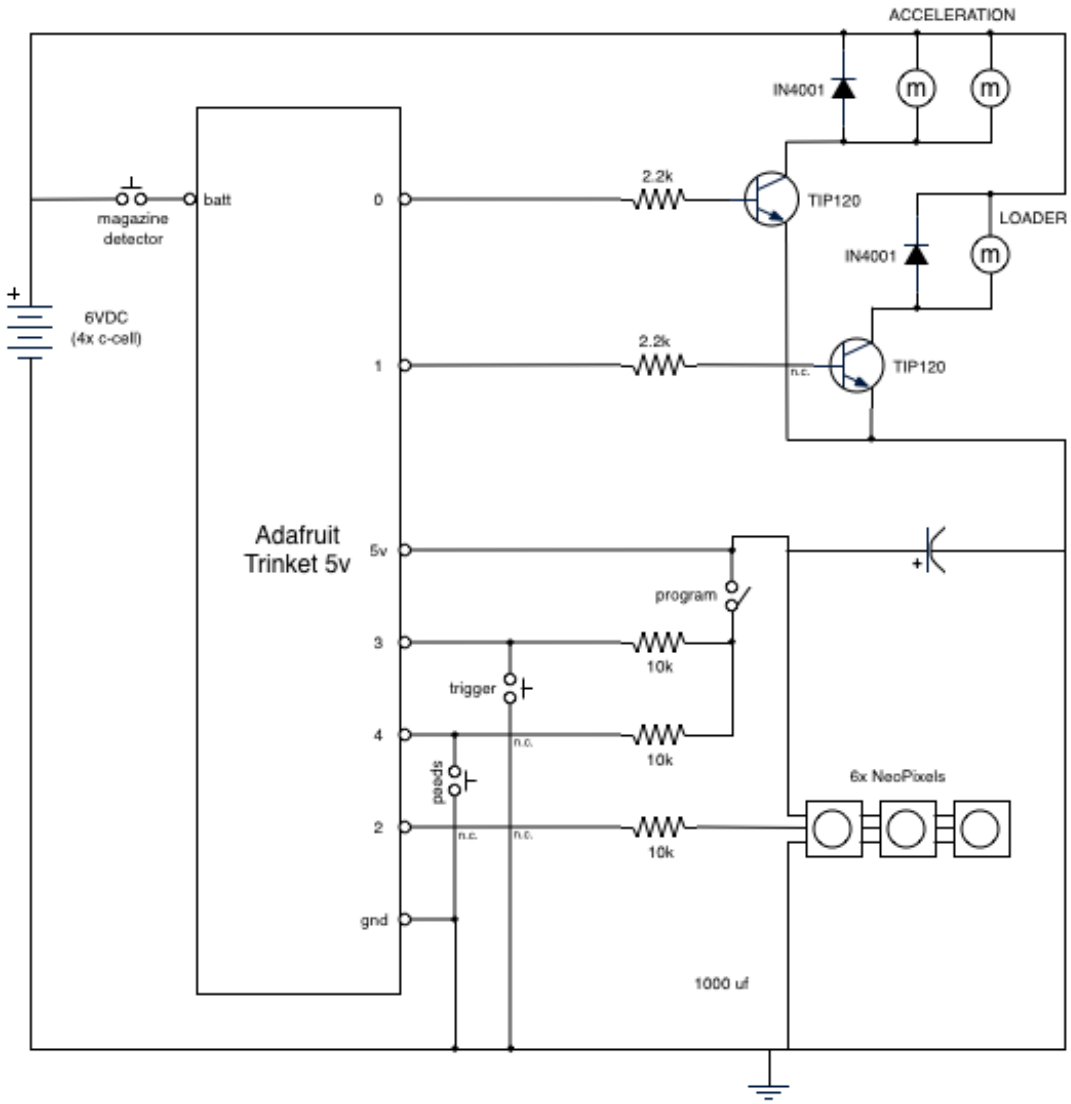




Circuit

This is what the circuit looks like. It's a very simple design, requiring few parts.

At the center of this project, is an Adafruit Trinket 5v. You could substitue for a Trinket Pro or even a Feather. Anything larger probably won't fit in the case. But, if you are creative, go for it!



Pins 0 and 1 are used to turn on/off the loading mechanism and the acceleration engine. Since the motors draw a considerable amount of power, we have to use a power transister to switch the load so it pulls directly from the battery. If you connect the motor directly to the Arduino pins, you will get a visit from the blue smoke monster.

In many cases, people will use an NPN transister for switching loads. In fact, I started out

that way, and found out quickly that the specs on the 2N2222 are about 700mA, and the motors want more than that. So, I swapped out the 2N2222 for a TIP120 that can supply between 2 and 5 Amps. Just what the motors need for speedy operation.

The 1N4001 diode is installed to drain off any power that might be generated when the motors slow down or stop. Since there is a huge rush of electons to start the motor, they need somewhere to go when the motor starts acting like a generator and is creating electricity. If too much were to build up, we could fry the TIP120 or the Arduino interface. Always use the diode with inductive loads (motors, solenoids, relays -- anything with magnets).

Couldn't you just use a relay to switch the load?

Yeah, sure, but this project called for multiple motor speeds. A relay would just turn the motor on full blast and then off. We are using PWM, or Pulse Width Modulation to control the speed of the motor. Since the TIP120 can turn on and off very quickly, it works perfectly to turn it on and off very quickly and control how fast the motor turns.

Moving down to pins 2,4, and 5 is where is gets a little tricky, and why I had to add the programming switch.

The Trinket shares pins 4 and 5 with the USB port. If you attempt to program it and the 10k resisters are pulling the data lines high, your computer can't program it. The programming switch simply removes the 5v power to those resistors, and the USB port is able to function normally.

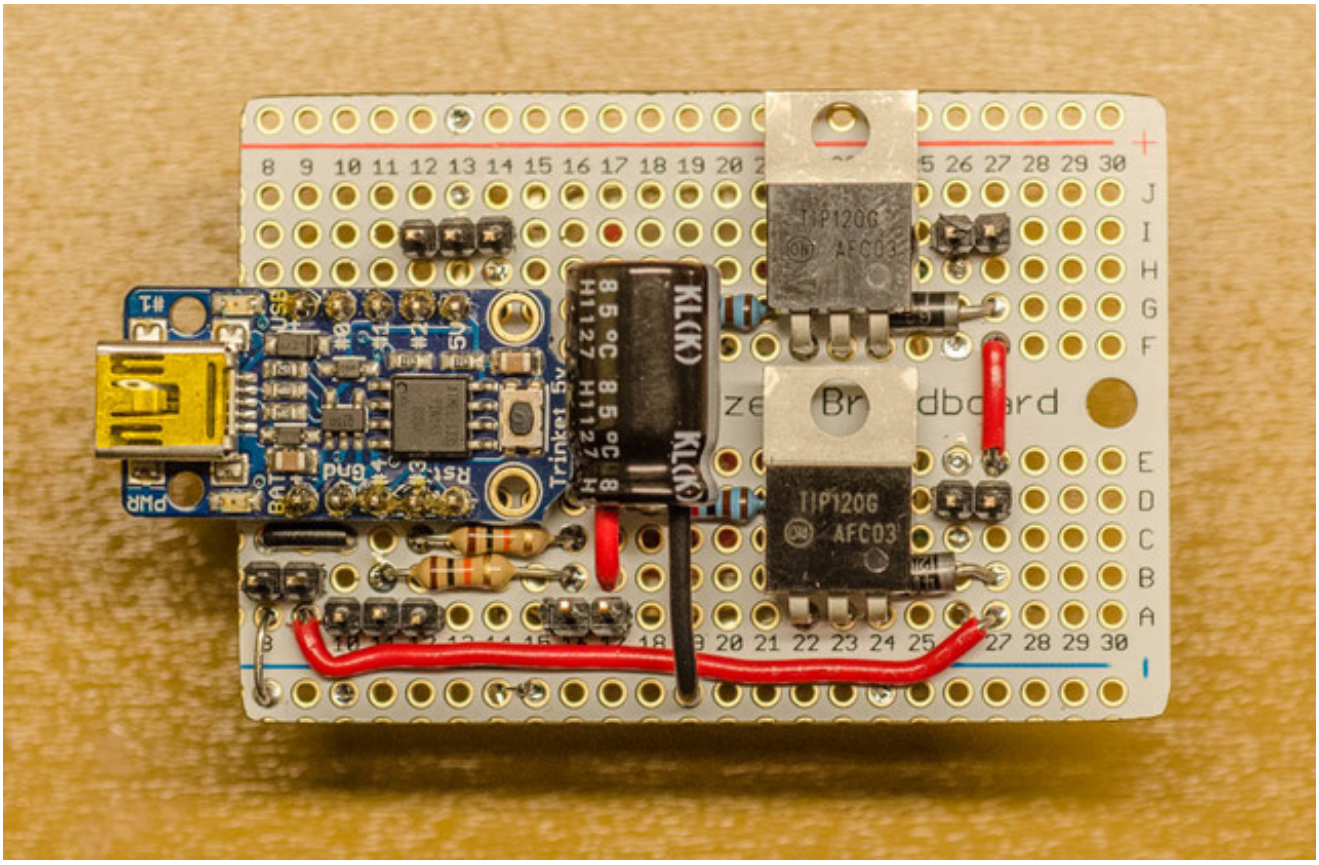
Pins 3 and 4 are used for the trigger and speed switches on the blaster assembly. They are pulled high by those 10K resisters via the programming switch. (programming is when the switch is open).

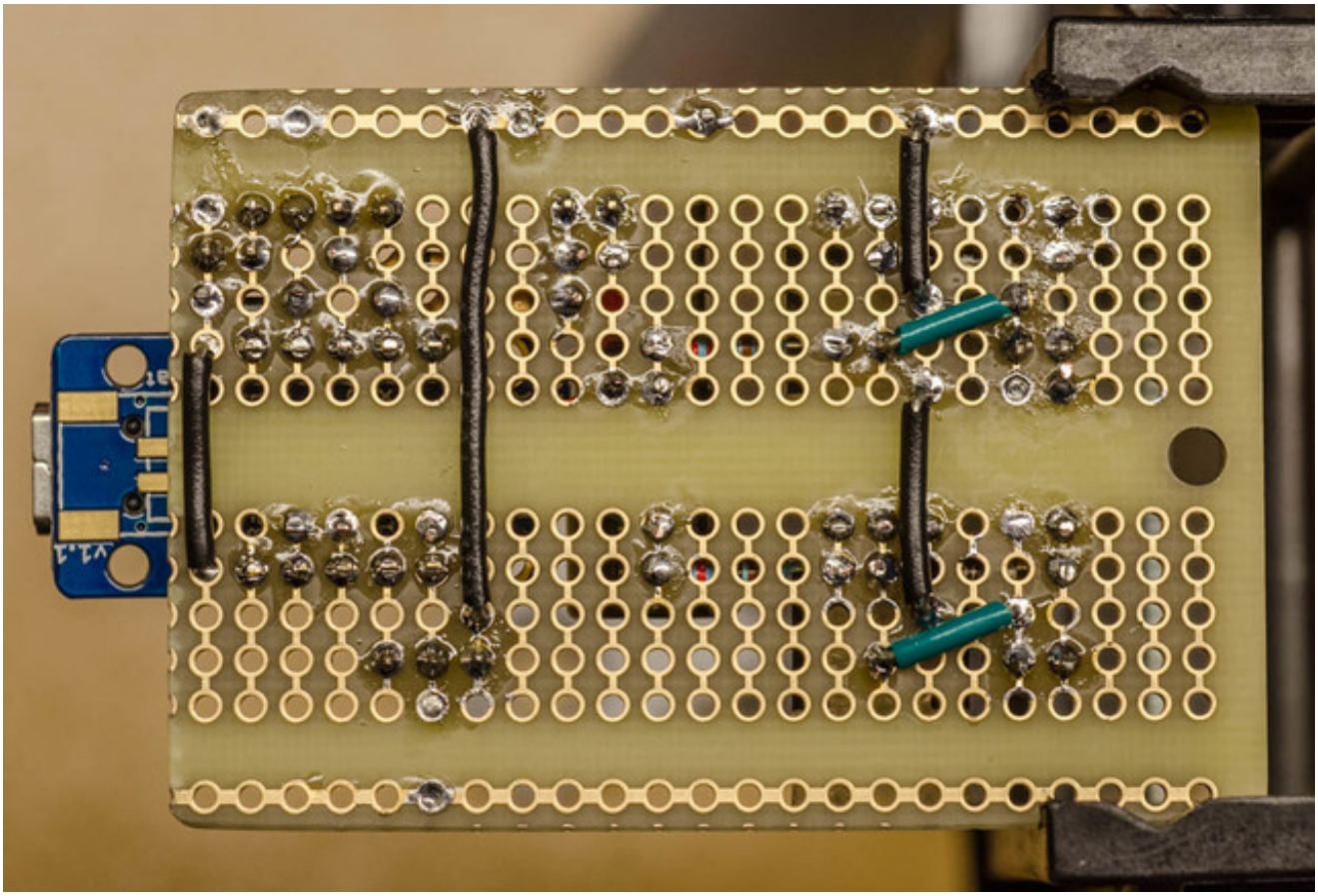
Pin 2 is the NeoPixel data pin. It connects through a 10k (some people use a 4.7k and that should be fine) to the DataIn port on the first NeoPixel. Look at the NeoPixel Uber Guide for more tips on using NeoPixels. https://learn.adafruit.com/adafruit-neopixel-uberguide/overview

Power and Ground also need to be connected to the NeoPixels with a 1000uf Electrolytic capacitor to smooth out the power. This is especially important becase of the motor noise in the circuit.

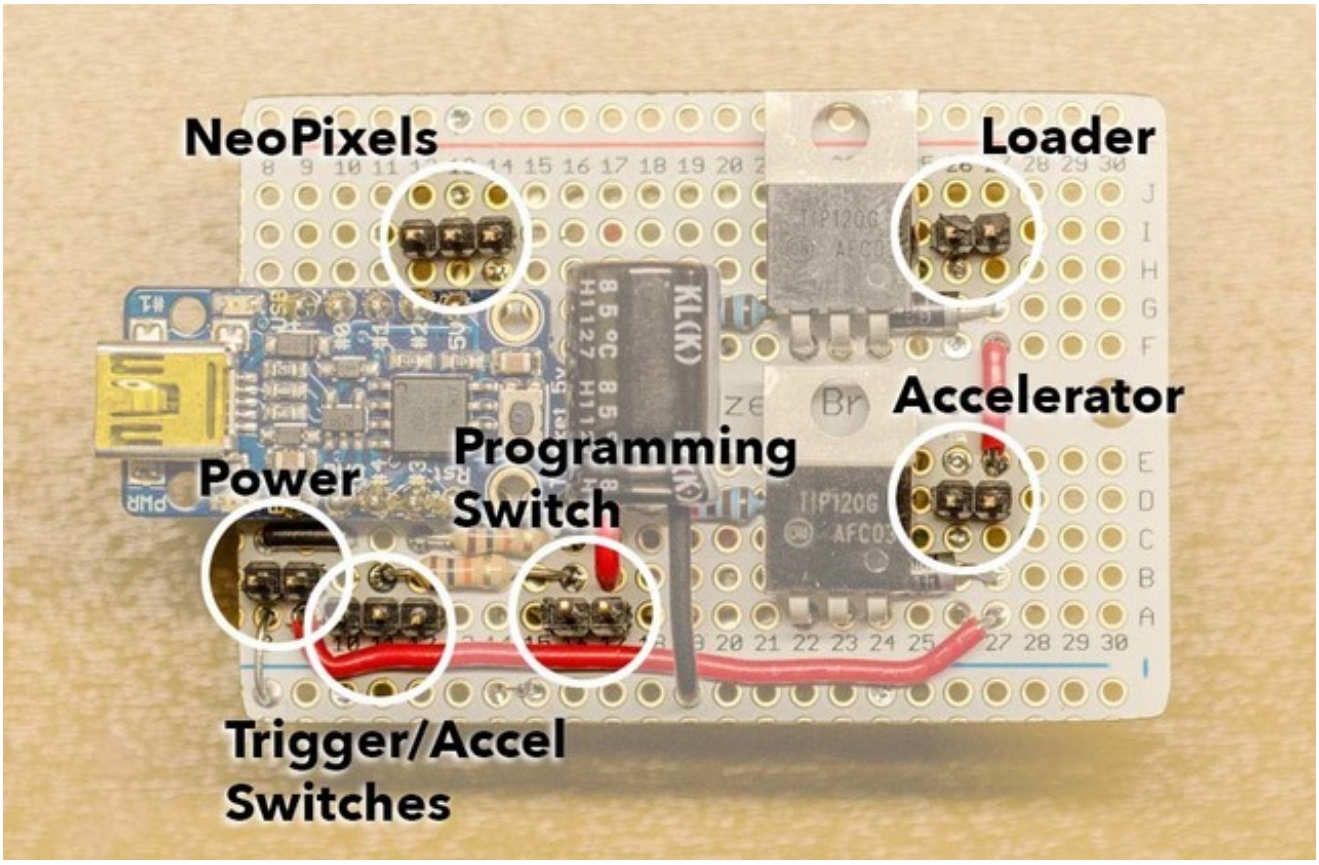
Main Circuit Wiring

This is the finished perfboard. I put it all together on a breadboard first, just to make sure everything worked, then soldered it exactly as it was layed out on the breadboard. Soldering is highly recommended becuase the blaster will get tossed around, and the components will get unplugged if they are in a breadboard.

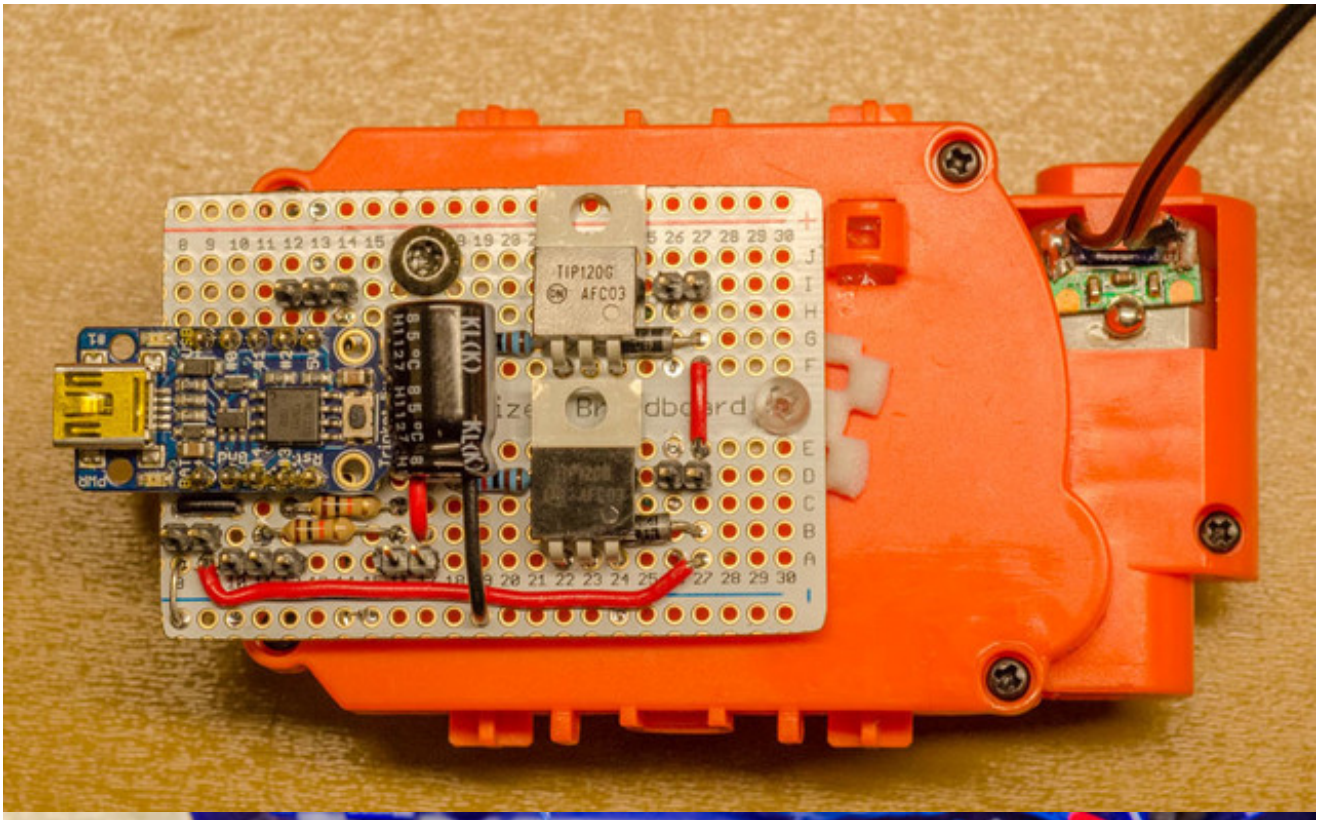


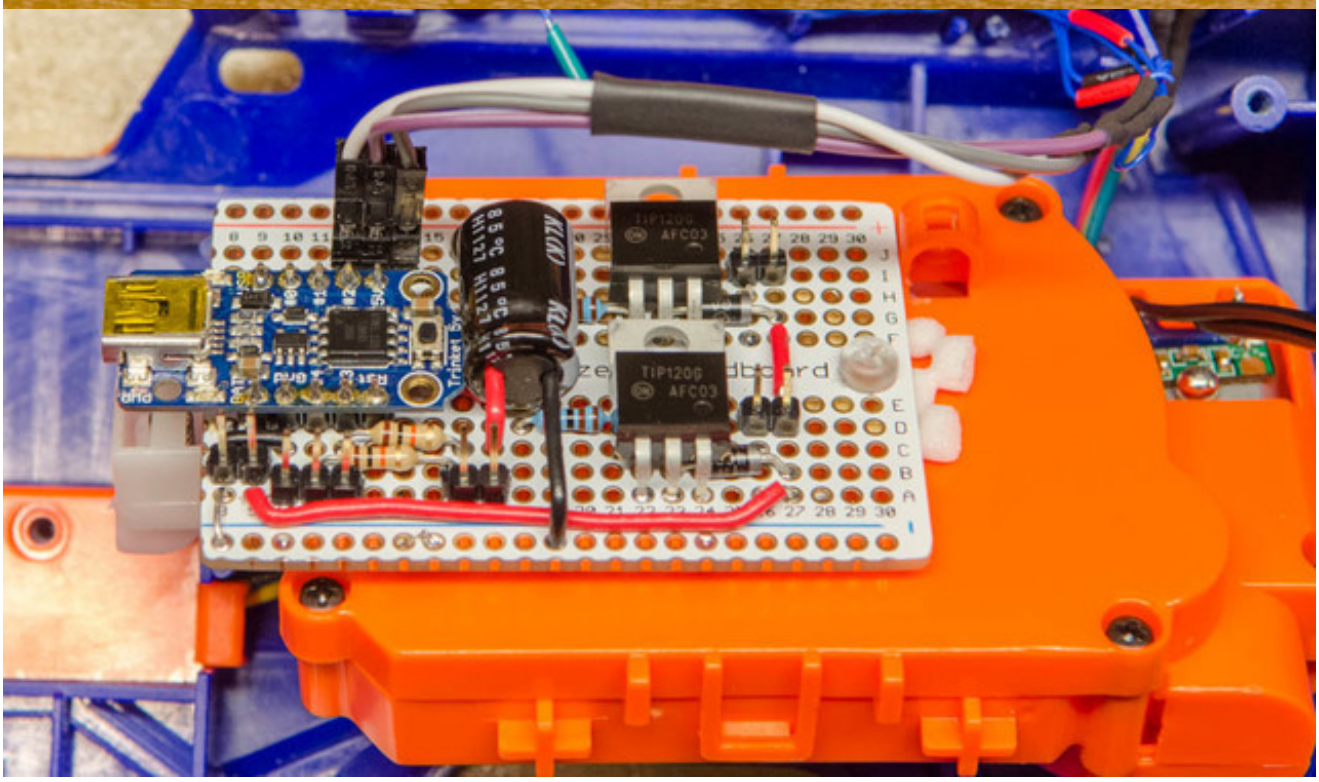


If you look closely, you will notice that the outside rows of the perboard are missing. I had to trim them down so it fit nicely on top of the accelerator housing. I also used pins for all of the connectors to the motors and switches. You could also hard-wire them instead of using header pins.



Here it is, attached to the top of the accelerator housing. I used a two screws to attach it to the case, with some foam for shock absorption. You could easily glue the board to the top as well.


TIP120G
AFC03

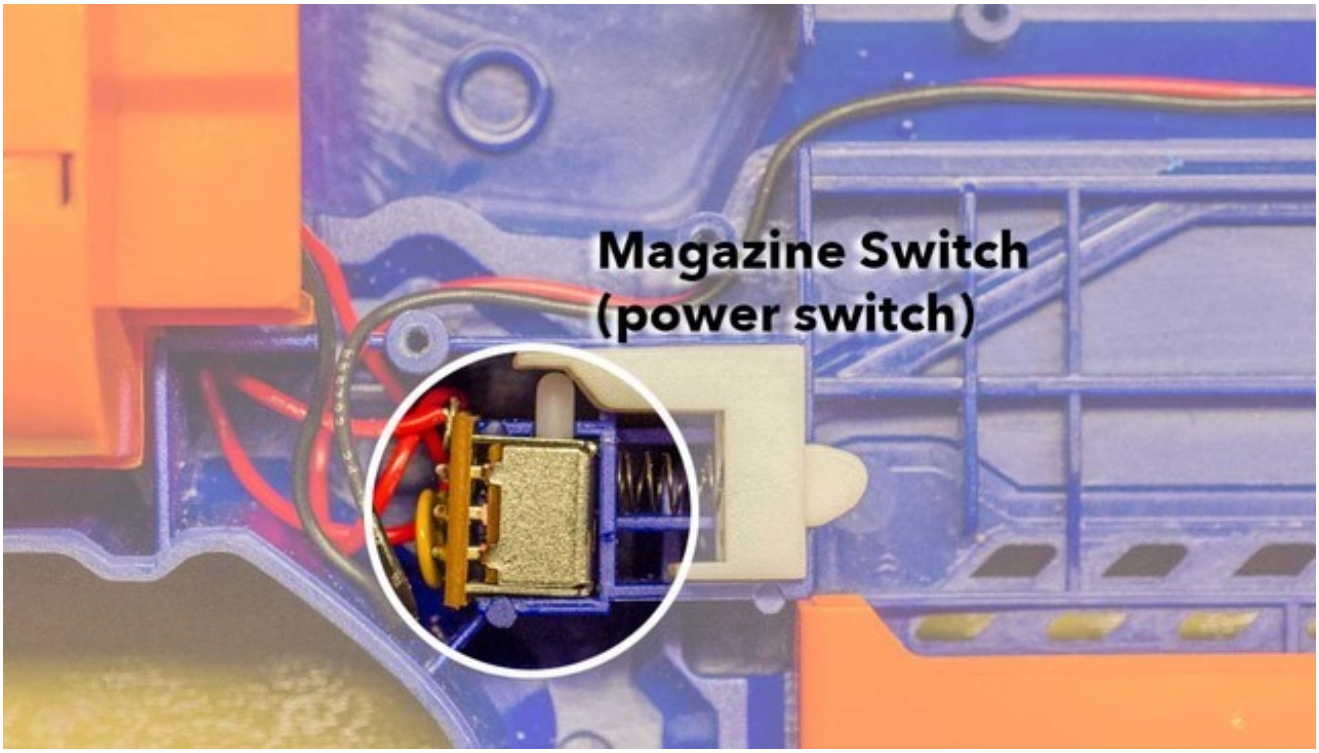

TIP120G
AFC03
TIP120G
AFC03

Blaster Wiring

Now that you have the main circuit assembled, you need to next work on connecting the NeoPixels, motors, and switches inside the blaster case.

This is the placement of the NeoPixels. I soldered them together using 30 gaugewire wrapping wire (http://adafru.it/1446), since it was very tiny and easy to snake through the case. The NeoPixels are glued in place with hot glue.

The magazine switch is actually powering the entire circuit. When there is no magazine inserted in the blaster, there is no power to the circuit. Basically, it connects the battery case in front (4x C-size) with the BATT connector on the Trinket.



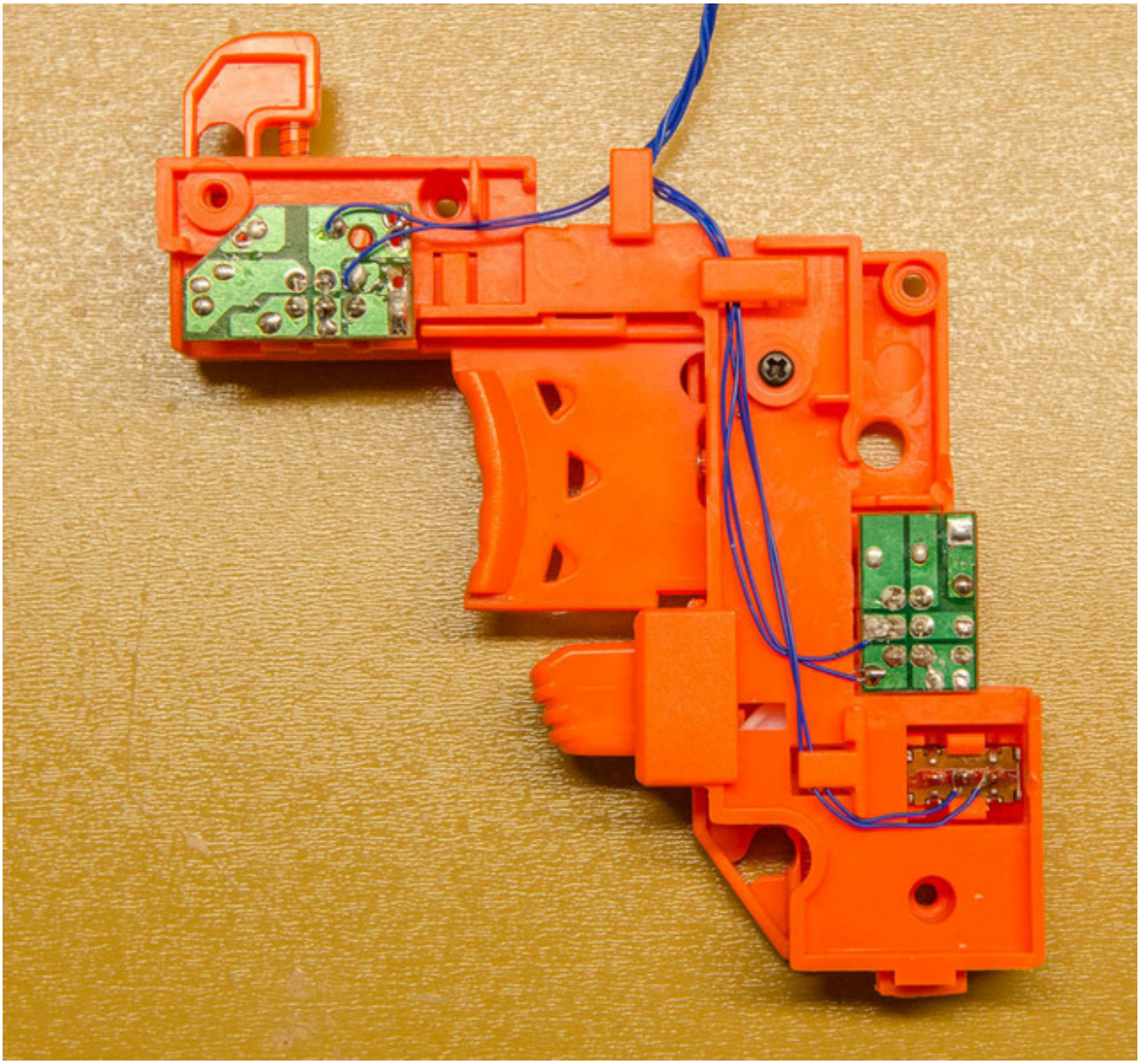
The trigger assembly is what we use to adjust the accelerator speed and to activate the loader, that pushes the dart into the firing engine. The switch on top detects when there is a dart available.

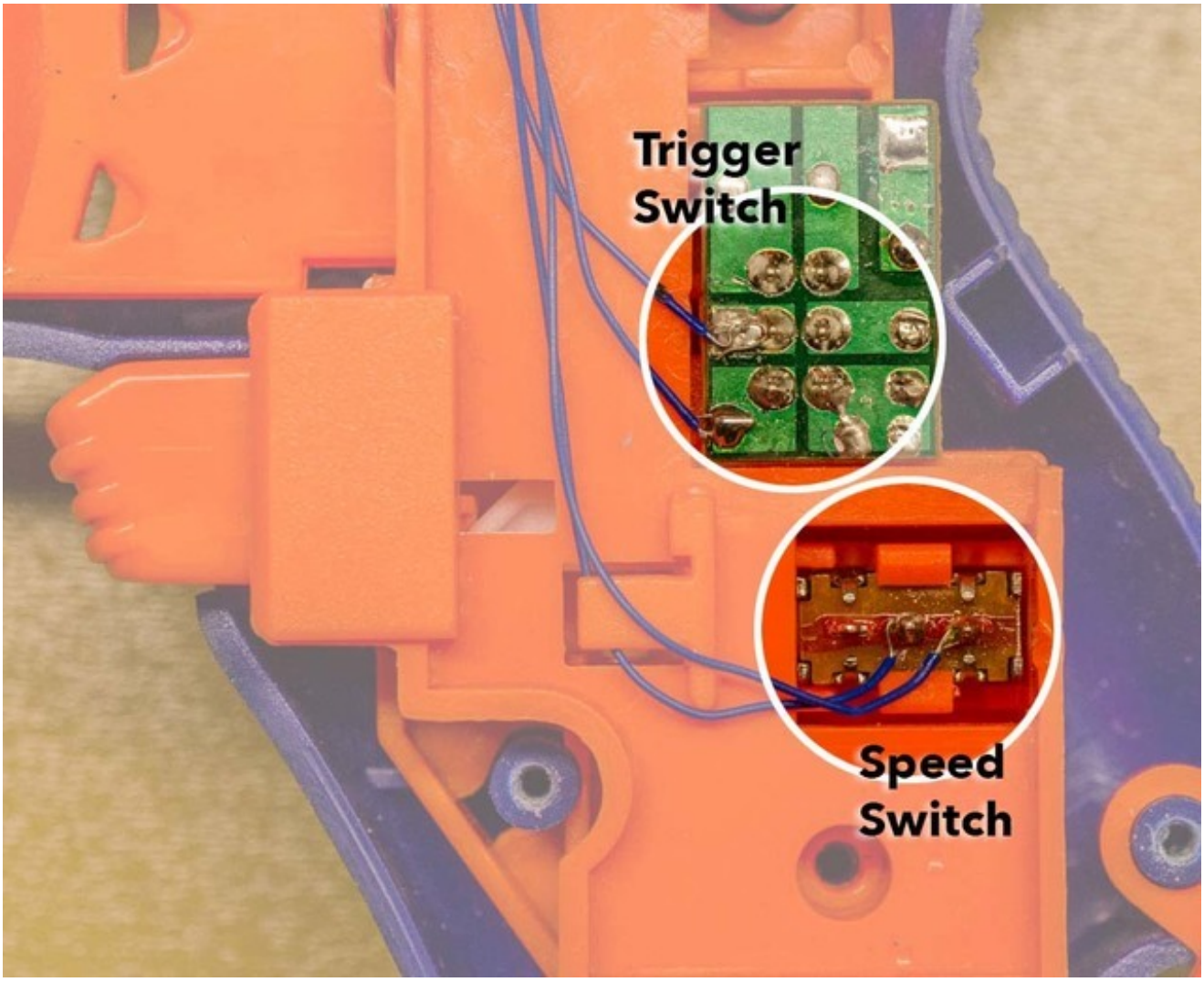
The trigger switch is the larger of the two triggers, and is used to activate the loader, that pushes the dart into the firing engine (accelerator). Wire it in series with the dart sensor on top, so that both switches must be activated in order for the dart to load.

GND --> TRIGGER --> DART SENSOR --> PIN 3

The speed switch is the smaller, lower of the two triggers. It is used to adjust the speed of the accelerator motors. Wire it directly to pin 4.

GND --> SPEED --> PIN 4



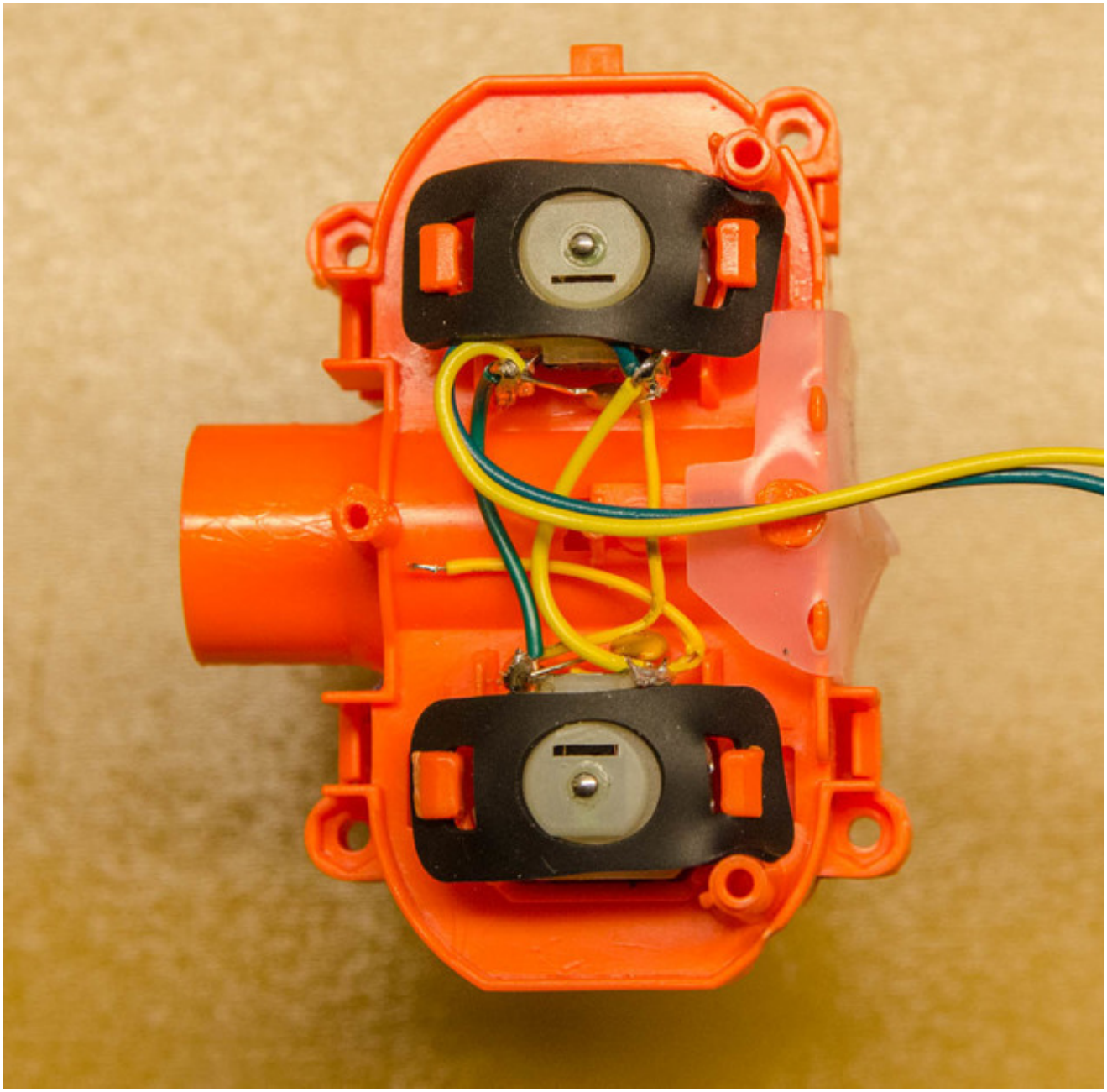


Wiring up the motors requires some more effort. Depending on the age of your Nerf blaster, the motor assemblies may be different. I have two different versions of this blaster that are 3 years apart in age. They changed the assembly a little between versions.

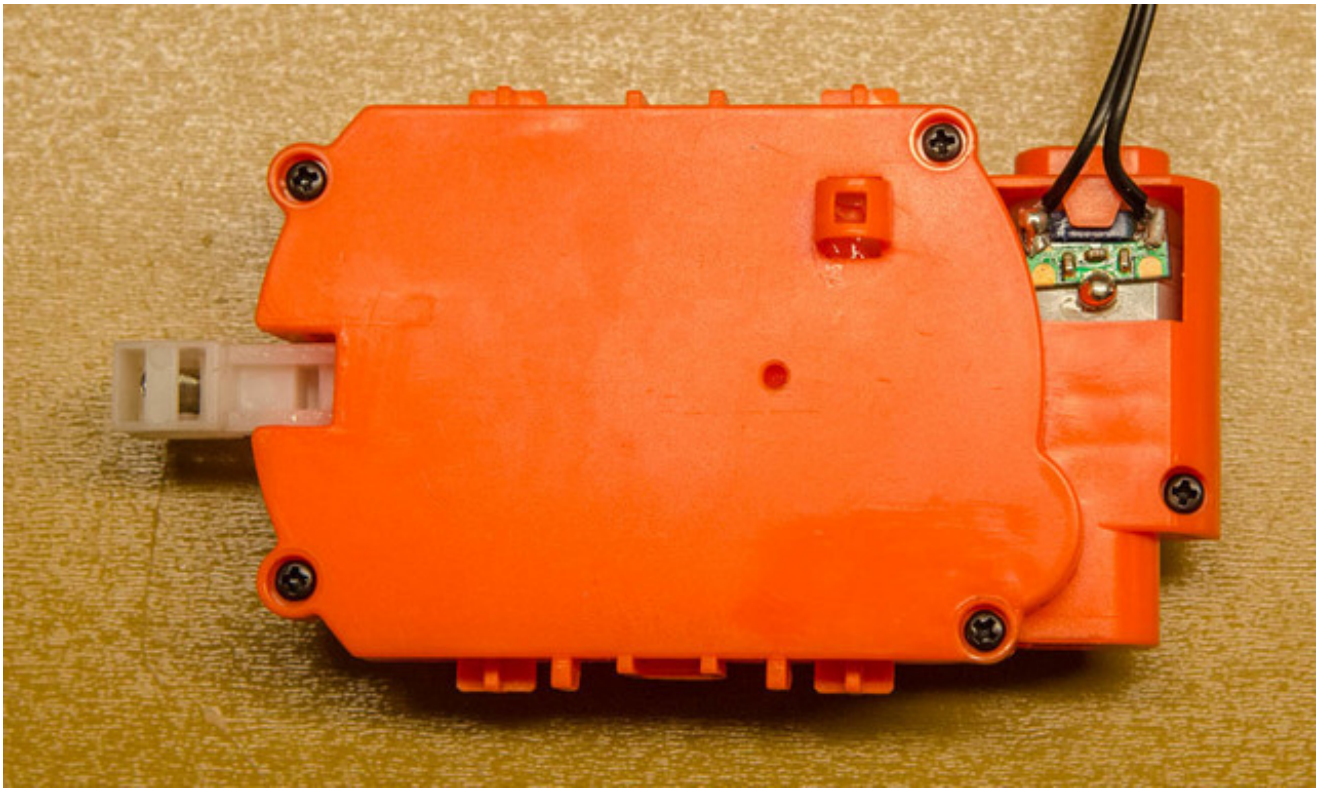
Basically, remove all of the circuit boards that may be attached to the accelerator. We are going to wire directly to the motors. See the image below, where we put both motors in parallel. Note that we have the polarity reversed between the two motors. In other words, we want one of them going clockwise and the other going counter-clockwise in order to fire the dart. If they go the same direction, it won't work.

Also, you might have to experiment to see which direction is forward and which is reverse. You don't want to fire the dart back into your face. You have been warned.

As long as the motors wired opposite each other, it's pretty easy to flip the polarity of the accelerator on the main circuit board.

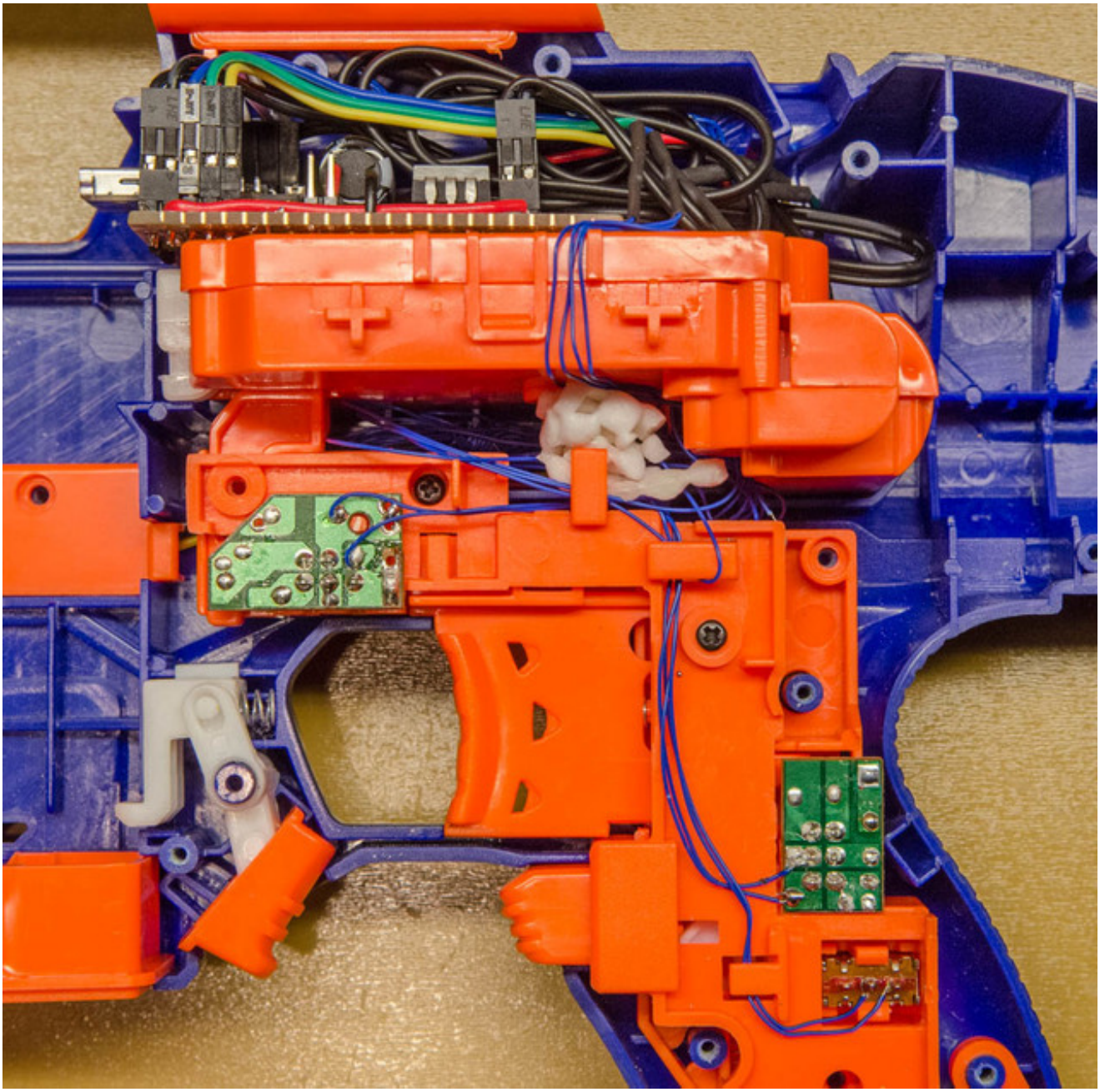


Next is the dart loader assembly. It is very straightforward, and simply has two wires that need to be connected to the LOADER transistor. One of the wires is probably labeled ground, so that goes to ground. If not, just experiment to get the correct polarity of the motor so that it pushes the dart forward when the trigger is pressed.



Put everything in the case, so it looks like this. Make sure the wires are routed so that they don't interfere with the mechanics of the unit. Use tape or little globs of hot glue to keep wires in place.

Also watch for pinch points where the case is screwed together.



Coding

What's a bunch of wires without a bunch of code?

You can download the code by pressing the download button below.

```
/*
 *  N3RFGUN
 *
 *  When magazine is inserted into the gun, the power switch is
 *  activated, and the circuit is powered by 4x c-batteries (6v).
 *  The gun should be silent, but the NeoPixels will start with the
 *  rainbow pattern, showing that the gun is active.
 *  When the speed switch is pressed, the accelerator motors will start
 *  to spin at 50%. Another click goes to 75%, and a third brings it
 *  to full power. One last click drops back to 0%.
 *  The main trigger will not fire unless the speed switch is pressed
 *  and held in place.  When you click to your desired speed, simply
 *  hold the button, then fire at will!
 *  Pressing the trigger button will activate the loader, which places
 *  a dart in the accelerator, and launches it out of the gun.
 *  This is a fully automatic gun, so it will fire until the darts are
 *  gone, or you let up on the trigger.
 *  Pressing the trigger also activates a lightning animation sequence
 *  on the NeoPixels. This simulates the muzzle flash, and looks cool
 *  in the dark or at night!
 *
 *  https://github.com/sKr0d/n3rfgun/wiki
 *  Scott and Alex McCulley, September, 2014
 */

int numpix = 6; // number of pixels in the chain

#include <Adafruit_NeoPixel.h>
#ifdef __AVR_ATtiny85__ // Trinket, Gemma, etc.
  #include <avr/power.h>
#endif
#define PIN 2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(numpix, PIN, NEO_GRB + NEO_KHZ800);

int TRG = 4;   // trigger pin
int SPD = 3;   // speed pin
int ACT = 0;   // action pin
int LOD = 1;   // loader pin
int NEO = 2;   // neopixel pin

int ACTS = 0;   // action speeed
```

```
int LODS = 240; // loader speed
int REL = 0;    // release speed button

int i = 0; // for loop
int k = 0; // for loop

void setup() {
  pinMode(TRG, INPUT);
  pinMode(SPD, INPUT);
  pinMode(ACT, OUTPUT);
  pinMode(LOD, OUTPUT);
  pinMode(NEO, OUTPUT);

  pixels.begin();
  pixels.setBrightness(85);
  pixels.show();
}

void loop() {
  analogWrite(ACT, ACTS); // set the acceleration motor speed
  rainbow(20);   // start the rainbow animation
}

void spd() {
 if (digitalRead(SPD) == LOW && REL == 1) {
   switch (ACTS) {
     case 0:
       ACTS = 128; // 50%
       break;
     case 128:
       ACTS = 192; // 75%
       break;
     case 192:
       ACTS = 255; // 100%
       break;
     default:
       ACTS = 0;   // 0%
     break;
   }
   analogWrite(ACT, ACTS);
   delay(40);
   REL = 0;
 } else {
   if (digitalRead(SPD) == HIGH) {
     REL = 1;
   }
 }
}

void trig() { // check for press of trigger button
   if (digitalRead(TRG) == LOW) {
```

```
    analogWrite(LOD, LODS);
    fire();
   } else {
    analogWrite(LOD, 0);
   }
}

void rainbow(uint8_t wait) { // make pretty rainbow colors
  uint16_t i, j;

  for(j=0; j<256; j=j+4) {
    spd(); // check for press of speed button
    trig(); // check for press of trigger button
    for(i=0; i<pixels.numPixels(); i++) {
      pixels.setPixelColor(i, Wheel((i+j) & 255));
    }
    pixels.show();
    delay(wait);
  }
}

void fire() {
  for(k=0; k<6; k++) {
    fastWipe(pixels.Color(255,255,255), 50);
    delay(20);
    fastWipe(pixels.Color(0,0,0), 50);
    delay(10);
  }
}

void fastWipe(uint32_t c, uint8_t wait) {
  for(uint16_t i=0; i<pixels.numPixels(); i++) {
      pixels.setPixelColor(i, c);
      pixels.show();
      delay(1);
  }
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  if(WheelPos < 85) {
   return pixels.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
  } else if(WheelPos < 170) {
   WheelPos -= 85;
   return pixels.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else {
   WheelPos -= 170;
   return pixels.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
}
```

n3rfgun code
http://adafru.it/uwd

PINS

In the top of the code, you will see where the pins for all of the functionality is defined. If you use something other that the Trinket, you might need adjust as necessary to fit your platform.

```
  int TRG = 4; // trigger pin
  int SPD = 3; // speed pin
  int ACT = 0; // action pin
  int LOD = 1; // loader pin
  int NEO = 2; // neopixel pin
```

TIMING

The speed of the loader can be adjusted by changing the value of LODS. This is the PWM value between 0 and 255.

The speed of the accelerator is controled in the spd function. You can adjust the steps by changing the ACTS variable. Each time the button is pressed, it increments to the next level. So, if you change the ACTS variable, you also need to change the corresponding case variable.

```
void spd() {
 if (digitalRead(SPD) == LOW && REL == 1) {
   switch (ACTS) {
 case 0:
   ACTS = 128; // 50%
 break;
 case 128:
   ACTS = 192; // 75%
 break;
 case 192:
   ACTS = 255; // 100%
 break;
 default:
   ACTS = 0; // 0%
 break;
}
 analogWrite(ACT, ACTS);
 delay(40);
 REL = 0;
} else {
 if (digitalRead(SPD) == HIGH) {
 REL = 1;
}
}
```

}

UPLOADING

Follow the directions for uploading code to a Trinket. It can sometimes be confusing if you over think it, since the Trinket does not show up as a USB device. Just trust that it is there!

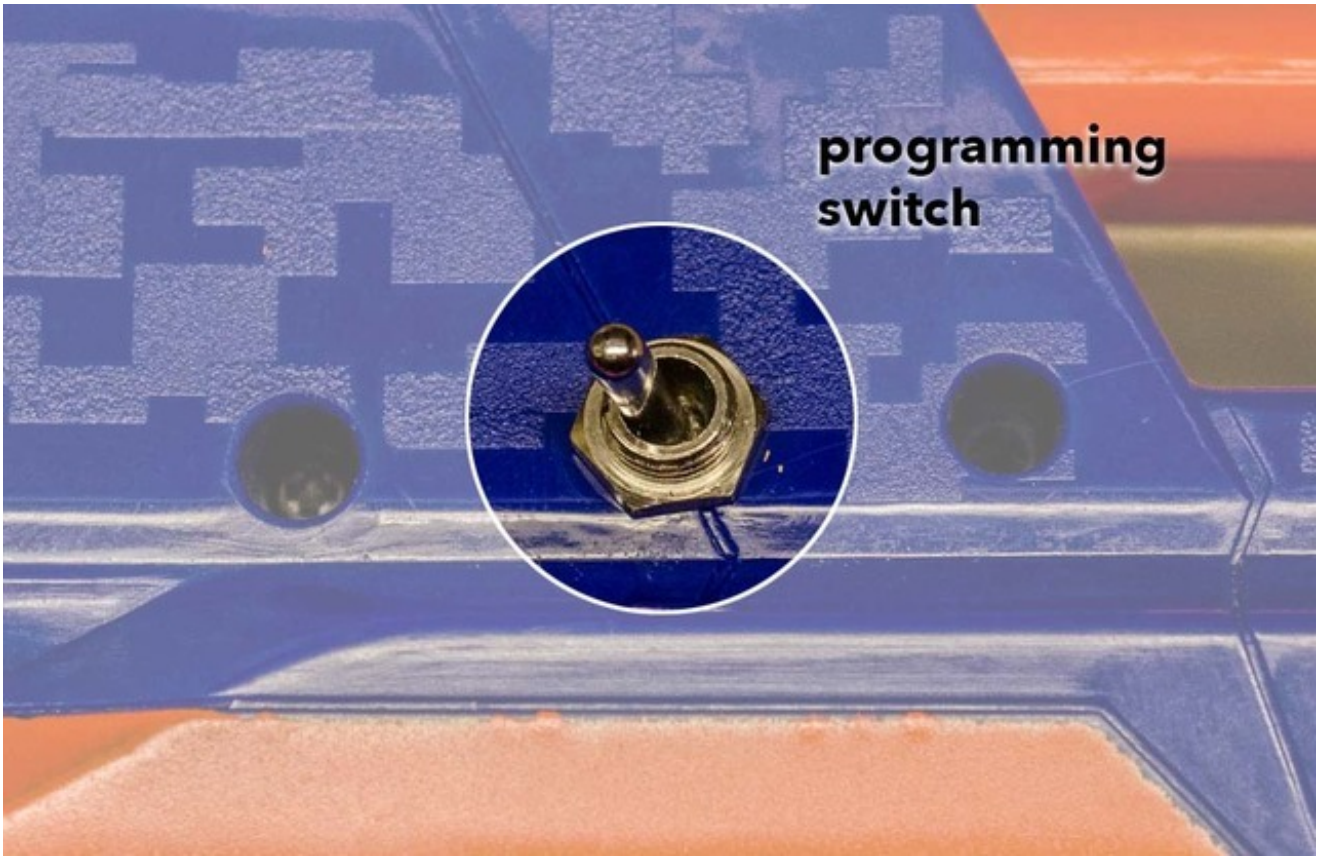
https://learn.adafruit.com/introducing-trinket/introduction (http://adafru.it/dpf)

Make sure the programming switch is enabled, so that there is no connection between 5v and the trigger resistors. This is important, because the Trinket will not be visible to the computer if the switch is supplying power to the resistors, pulling the pins high.

Upload the code.

As soon as it is done, you should see the rainbow pattern on the NeoPixels!! If not, there might be something wired incorrectly. You tested it out on a breadboard, right?

Disconnect the Trinket from your computer, then flip the programming switch so that it is supplying 5v to the trigger resistors. Without it, the triggers may or may not function, since they are not pulled high. If the guntends to missfire, check this.

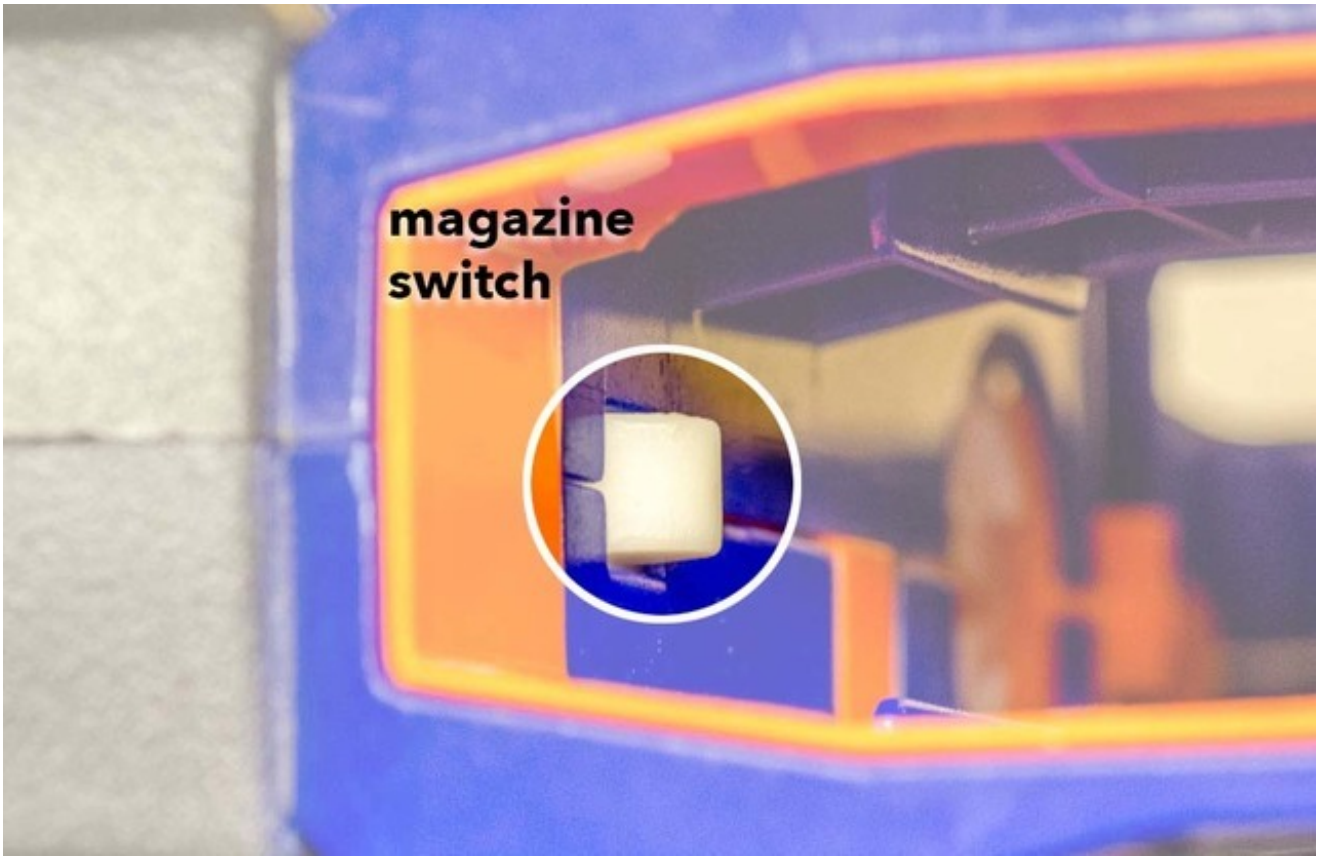


Blaster Operation

When the circuit is done, and the code has been uploaded, the blaster is ready for operation. This is how it works.

POWER

In order for the blaster to function, a magazine must be loaded into the blaster. This will activate the power switch so that the Trinket can power up. You will know it is on when the NeoPixels start flashing the rainbow pattern.



SPEED

When power is first applied to the blaster, the speed is set to zero. So, there will be no whirring of the accelerator motors. Press (and hold) the lower trigger to engage the accelerator at 50%. Press (and hold) again to increase to 75%. Press (and hold) again to increase to full power.

When you release the lower trigger, the accelerator will stop. This saves batteries, and

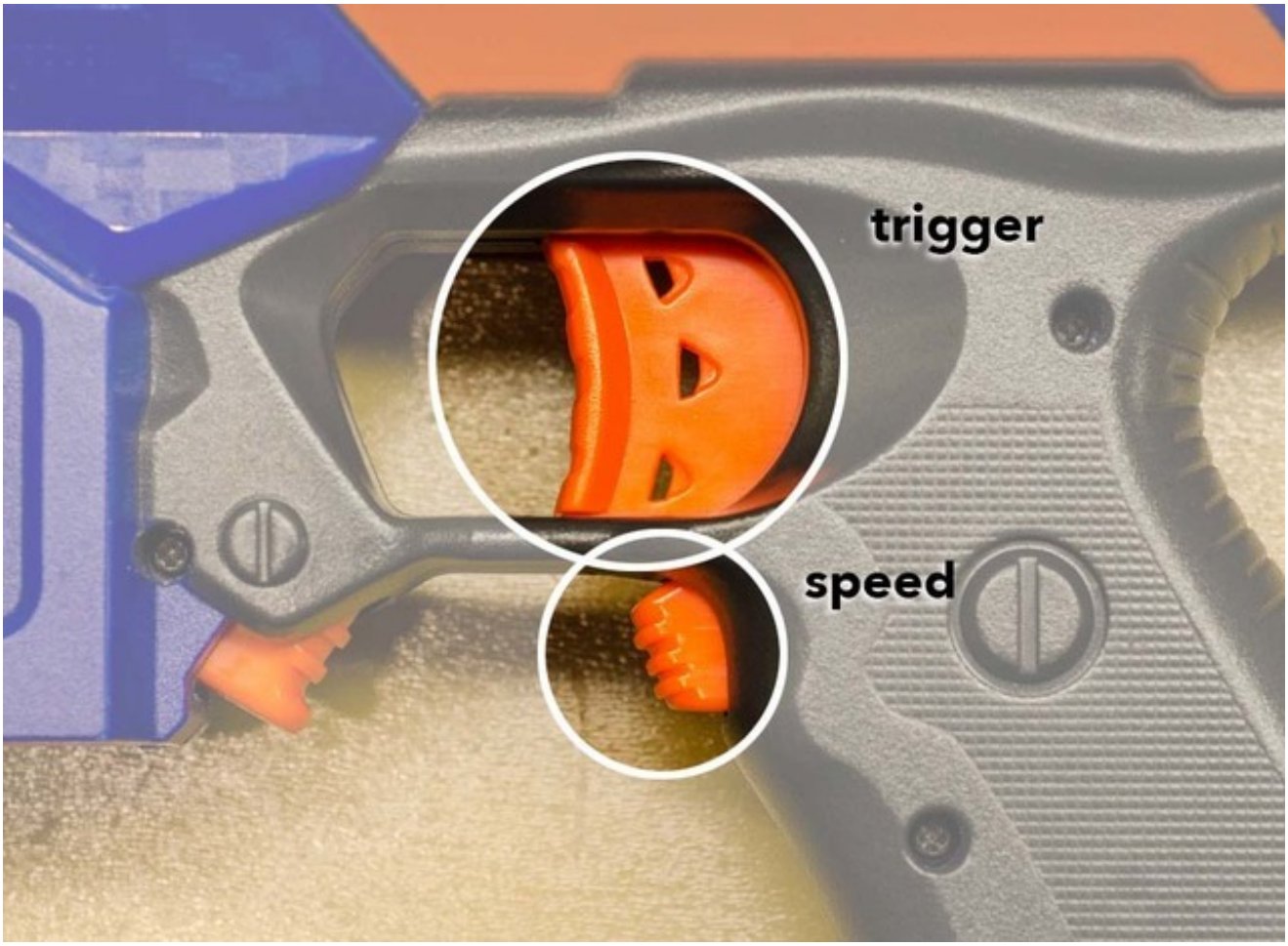
makes the gun silent for when you are trying to sneak up on someone. (well, someone who can't see the blinky lights anyway.)

TRIGGER

There is a dart sensor at the top of the trigger assembly that will detect the presence of a dart in the magazine. If there is no dart, the loader will not advance it to the firing engine.

Once a dart is detected at the top of the magazine, the trigger switch will activate the loader mechanism, which pushes the dart forward to the accelerator (firing engine.) The faster the speed of the accelerator, the more velocity applied to the dart, and the farther it flies (or the more it hurts.)

When you are out of darts, the trigger simply makes the NeoPixels flash.



Have a great time building this Nerf blaster! Use this guide as a starting point, then customize it and make it your own!!