HOW TO:

Creating a Digital Light Wand (Programmable RGB LED Light Strip) for use with Light Painting Photography

(Using an Arduino MEGA board to control an addressable

RGB LED Light strip that uses the HL 1606 RGB LED Controller chip)

BY: Michael Ross

(aka TxPilot on Flickr)

http://www.flickr.com/txross

http://www.mrossphoto.com

http://www.facebook.com/michaelrross

TABLE OF CONTENTS

Forward

First of all, I want to make sure and give credit where credit is due. If it had not been for the work John Cohn and Xander Hudson in cracking the code of the HL1606 chip and creating an Arduino Library for it, I would not have gotten as far as I did within such a short period of time in creating this digital light painting tool. As far as I know, this tool was the first of its kind in regards to the size and number of LED Lights that were being controlled for specific use in Light Painting or long exposure photography.

I also want to give a huge shout out to a few friends and fellow light painting photographers out there for helping me make sure this documentation was as detailed and comprehensive as it needed to be to basically stand on its own. **blacklodge!!** (http://www.flickr.com/photos/thebadseed/) and **garnatha** (http://www.flickr.com/photos/garnathan/) and **Dennis Calvert** (http://www.flickr.com/photos/denniscalvert) on Flickr were all great in being test subjects and giving me feedback to get this project completed. THANKS A MILLION GUYS!!!! ☺

This programmable RGB LED Light strip is the product of a lot of trial and error and working towards what I considered to be "The Ultimate Light Painting Tool!" (at least from a technical standpoint). Being able to program a strip of RGB LEDs to turn on and off in a specific sequence at specific color values was a vision of mine every since I started Light Painting a few years ago.   At that time, there had been a few others that had created something similar on a smaller scale using single color LED's or a few RGB LED's, but all my research in trying to find something that could be used for this on a larger more flexible scale led me to many dead ends. I knew that RGB LEDs could have been wired individually but with 4 wires per LED it would have taken so many wires and several Microcontroller boards to do what I wanted to do. Controlling a large number of LEDs with this method was just not feasible without a way to "Address" specific LEDs that were chained together.

Then one day in about November of 2009, I ran across a short video of an RGB LED strip that had the capability I was looking for! John Cohn had created a headband with multi-color "Chasing Lights" effect using different colors in the cycle! I knew my search was over! I knew at this point that this strip could do what I wanted it to do! It was just a matter of figuring out how to hook it up to an Arduino Microcontroller and figuring out the proper commands to program it. The strip I saw originally was programmed using the PIC microcontroller chip. A little more research revealed the Arduino Library created by Xander Hudson and I had what I needed then to make this work! By January of 2010, I had completed testing and put the light strip into use! The plus to all of this is that a 9 volt battery can be used to power the Arduino board and the power to the LED strip can come straight from the Arduino as well keeping the entire package VERY portable! And only 6 wires were required to drive the RGB LED Strip! Two wires are used to supply power (positive and negative) to the strip, and the other four to send digital commands to the light strip to set the color of each LED and turn them on and off.

How To: Creating a Digital Light Wand for Light Painting Photography

The specific micro-controller board I ended up using is the Arduino Mega. This is mainly due to the amount of memory that the board has for holding program code and this one had the most programming memory at the time. The amount of memory on this board allows for several switchable programs using a shorter strip and a large very inefficient program for a larger 48 light strip. I have one set up for 16 LEDs and use a rotary switch to allow for 6 different program patterns, and also a longer strip with 48 LEDs. For the longer strip, there is just enough memory to work out about 230 switching sequences for all 48 LEDs without condensing the code using looping statements. This based on the specific program and some shortcuts can be used if are specific patterns that can be reprogrammed into smaller code units by using looping or if statements. Anyway… one of my goals with this tutorial is to try to make it as NON-Technical as possible so that a novice with electronics (but someone with at least some DIY skills) could make one of the strips and get it working for them.   I have already gone too far by talking about micro-controllers, memory, and code loops, so it is important to remember that you do not need to know anything about all of that to make this work.

I did have to jump a few hurdles in dealing with the programming code to get the strip to do what I wanted it to do but without getting into the details of how I figured it all out, here is the complete tutorial. This is all still a very crude method of going about all of this and I am sure that others with more programming expertise and time will be able to come up with more efficient ways of doing things. I really do hope that happens and that anyone doing that will take that and share it with the rest of the world just as John Cohn, Xander Hudson, and myself have done with everything that has gone into this up to now. The coding that is created with what I have done that is sent to the Arduino board is not as efficient as it could be by any means. Ultimately it would be nice to create a custom interface to "Draw" the image that will then generate and download the code directly to the Arduino. So I hope someone does just that and lets everyone else benefit!

I would imagine that this document will be fairly dynamic and changing as things change and improve with the light strip itself. Please let me know if you find any errors in this document or if you have a suggestion to make it better.   First of all, I know that I cut some corners and that I did some things that others would consider the wrong way of doing it. I published this tutorial in a way to keep it easy for the non-geek to follow! If you are geek, by all mean, apply your geekiness to your own version! ;-) I know there are plenty of ways to go about all of this, so if you have another way, then just do it.

ENJOY!!

This document and all images are protected by Copyright, please do not repost this document or any image **anywhere** without express permission from myself to do so.   Always link back to the original location of this document as opposed to creating a copy to be posted. Thank you for your cooperation. Michael Ross

The original location for this document can be found at http://www.mrossphoto.com/DigitalLightWand

(Additional Note: As I was nearing completion of this tutorial, another guy located in Sydney Australia announced that he had created a tool very similar to this. He is using a little different approach to it but the final outcome is the same. We are talking about combining our efforts and putting together the best of both tools. More to come on this)

WARNING! - The light strip and Arduino board contain sensitive electronic devices that can be damaged by Electrostatic Discharge (static electricity). If you are in an area where static electricity is a problem, it would probably be best to get a grounding wrist strap to ground yourself as you work with these devices.   In my almost two years of working with these two items, I have not taken any real precautions with this whatsoever and I have not grounded myself in any way when working with them with no damage to them at all. That doesn't mean it will not happen so use your own judgment depending on the area you are in and how much static electricity you have to deal with in your location.

# HARDWARE – BUILDING THE DIGITAL LIGHT WAND

FIRST STEPS

PARTS LIST

Here is a list of things you will need:

1) A portion of the HL1606 controlled RGB LED Light Strip
2) An Arduino Mega Microcontroller board or better (clones will work as well but the amount of memory is what is important here. At the time of this writing the Arduino Mega (ATmega256) is out and it has 256 KB of memory)
3) Wire (To connect the Arduino Mega Board to the LED Light Strip) (20 gauge solid or stranded will work fine)
4) A USB Cable (To connect your computer to the Arduino Mega Board)
5) A Soldering Iron (To solder the wire to the RGB LED Light Strip)
6) Solder (To solder the wire to the RGB LED Light Strip)
7) A 9 Volt DC Battery
8) A 9 volt battery connector
9) An "N" size DC power connector plug the 9 volt power into the Arduino Board
10) Optional: A push button switch (either momentary or Push on/Push off) to make it easier to apply power to the Arduino board from the 9 volt battery
11) A strip of wood trim to mount the RGB LED Light Strip to.
12) Optional: A Project Box to Mount the Arduino Board. Depending on the project box you purchase, you may have to do some cutting and drilling to make it work for you so that you can get to the power connector and also the USB connector on the Arduino board. I am fairly certain that there are specific project boxes that are made for the Arduino boards and it might be easier to purchase one of those if you would like to skip all of the customization with small power tools. The project box I used is made by SERPAC and it is the model No. 251.
13) Optional: A six position rotary Switch (To allow running of six different programs)
14) Optional: A Knob for the rotary Switch to make it easy to select different positions.
15) Optional: 5 - 1Kohm resisters for the six-position rotary switch to create a resistance bridge that the Arduino board can read to detect the rotary switch position.
16) Optional: Pin connectors to plug into the Arduino board pin outs (also called headers)
17) Optional: Connector to place in the 6 wire connection between the Arduino board and the Light Strip (Mainly if you plan on using the Arduino as a portable one to switch between a short and long light strip)
18) You will need to download and install the latest Arduino Software (detailed in the software section of this tutorial)
19) You will also need to install the Light Strip Library code so that the Arduino Software will recognize the commands for the Light Strip. (detailed in the software section of this tutorial)

How To: Creating a Digital Light Wand for Light Painting Photography

Note: All of this, of course, is just the route I ended up taking with it all. I wanted to keep it all as simple as possible but, as with just about all things, there are many different ways to accomplish the same task. So if you have the technical background to improve any of this and do things better or more efficient, by all means do so! :-) But PLEASE SHARE!!

On the following pages are some photos of each of the components listed in the "things you will need" list so that you can see what they look like in case you are not familiar with one or more of them.
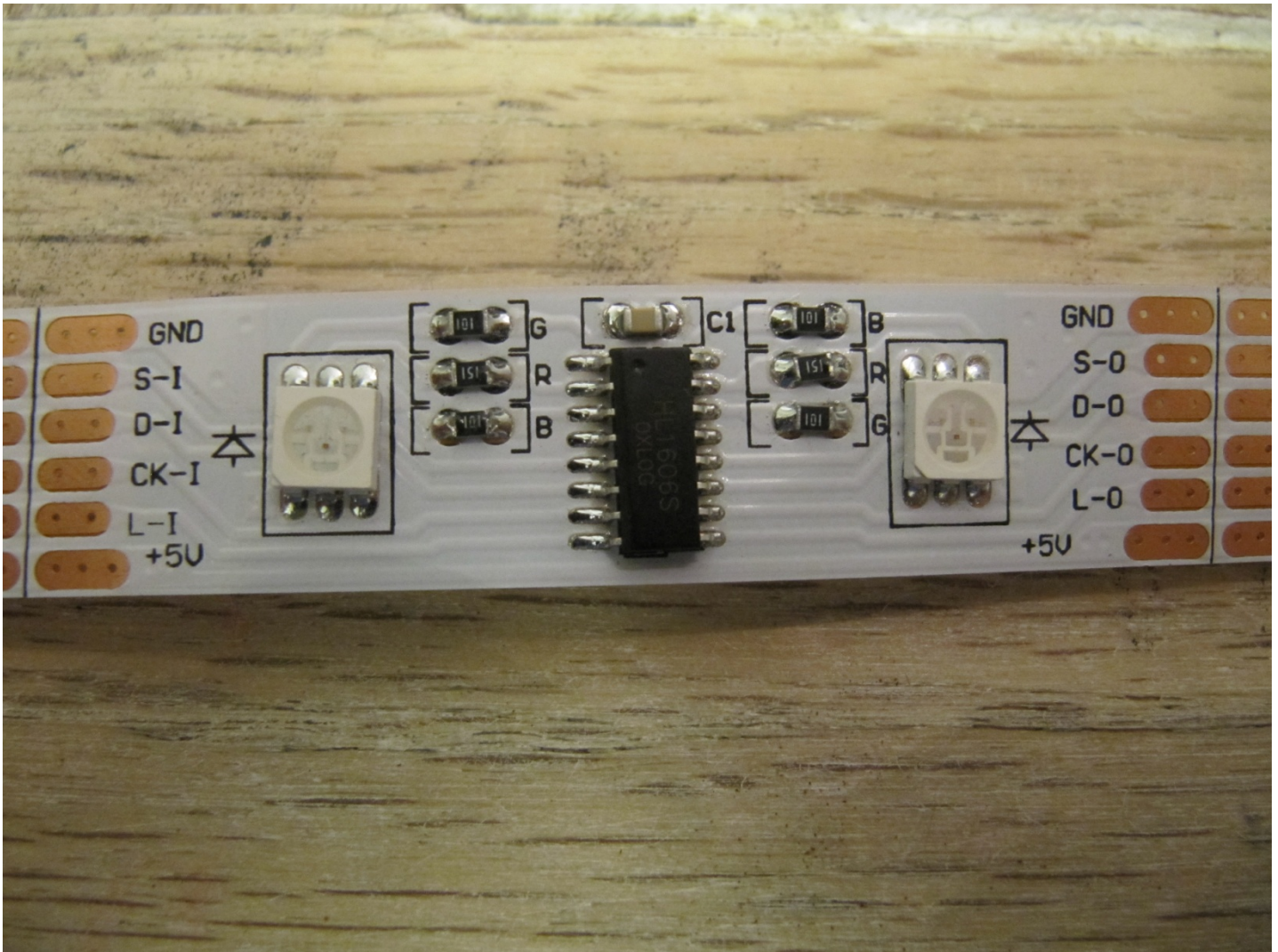
THE RGB LED LIGHT STRIP – (with HL1606S LED controller chip). The RGB LEDs are the white squares to either side of the HL1606 chip in the middle. They are super bright! The small black chips on the strip are resistors that are used in the circuit for each color of the RGB LEDs (G, R, B = Green, Red, Blue respectively). You may notice that the strip is very flexible and will turn fairly tight corners if needed.   The Line between the copper pads at each end is where you can cut the strip to length using household scissors. Also note that the identifiers of each end of each section of the strip are marked. The end with the "I" designators are the "Input" end and the "O" is for "Output". This is important to remember when soldering the wires to the strip to connect to the Arduino board. You want them soldered to the INPUT end!
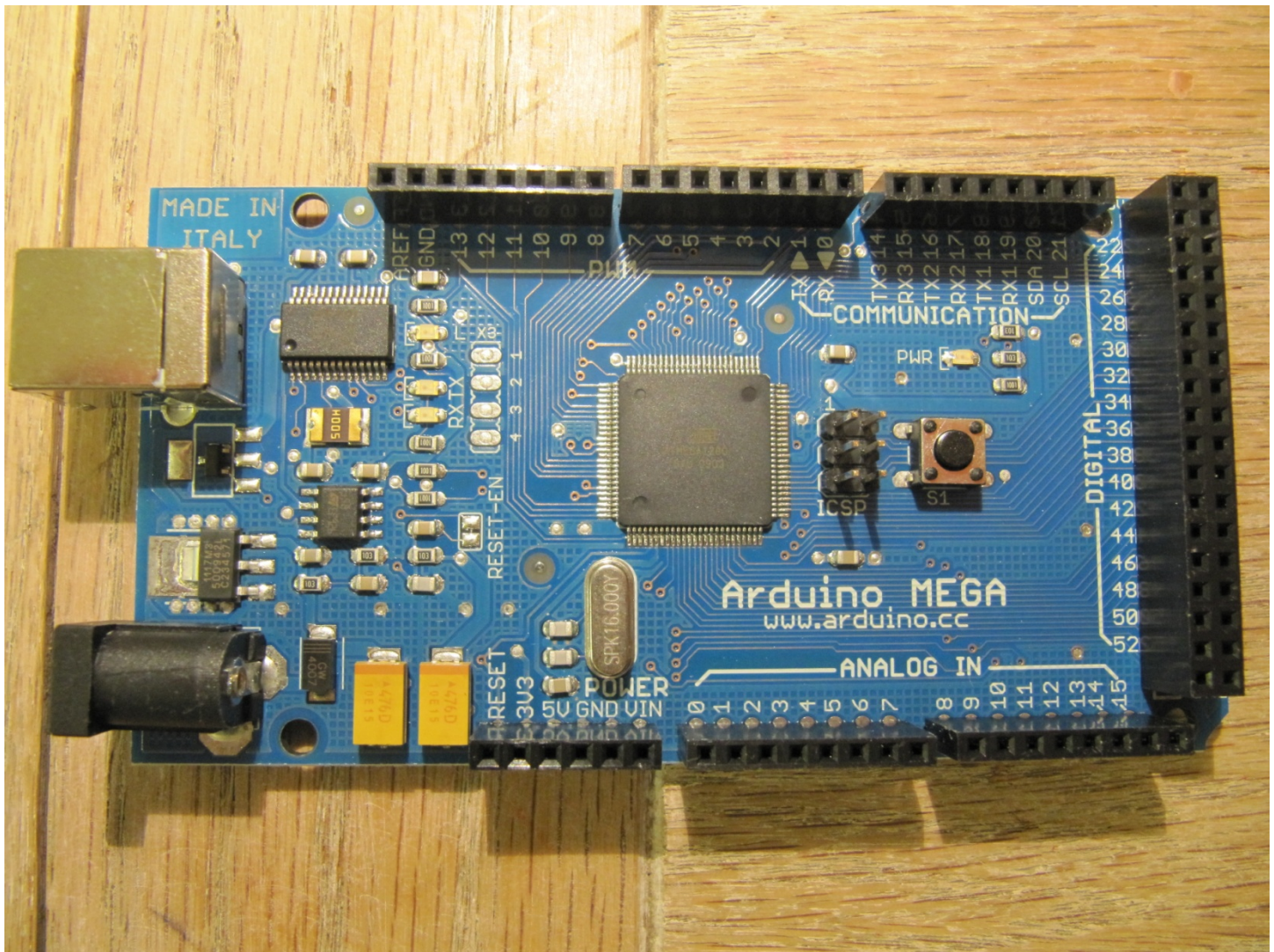
Here is another photo of the HL1606 Light strip. This is 5 meters of the strip the way it is packaged in a waterproof packaging. The photo above shows the strip being driven by the RGB LED Controller that can be purchase with the strip. This also shows how flexible the strip is as well with the tight turns in the middle of the coiled up roll.

The Arduino MEGA microcontroller board. There are several versions of this it seems out there and all of them will work. The important things on this to note are the USB connection (large chrome connector on the top left), the type N power connector (large black connector bottom left), and the specific connections along the edge connectors of the Arduino. You can see here that there is a POWER section, an ANALOG IN section, a DIGITAL section, a COMMUNICATION section, and a PWM section. I will specify which pin connections to use in the detailed part of this document.

20 Gauge Wire to connect the Light Strip to the Arduino board. I used solid conductor wire because there would need to be no flex in the wire connections between the two, but you can use stranded wire if you want or if you plan on changing up the design a bit using connectors maybe to use one Arduino board for two different light strips.

A USB cable for connecting your computer to the Arduino board to download the program scripts to it.
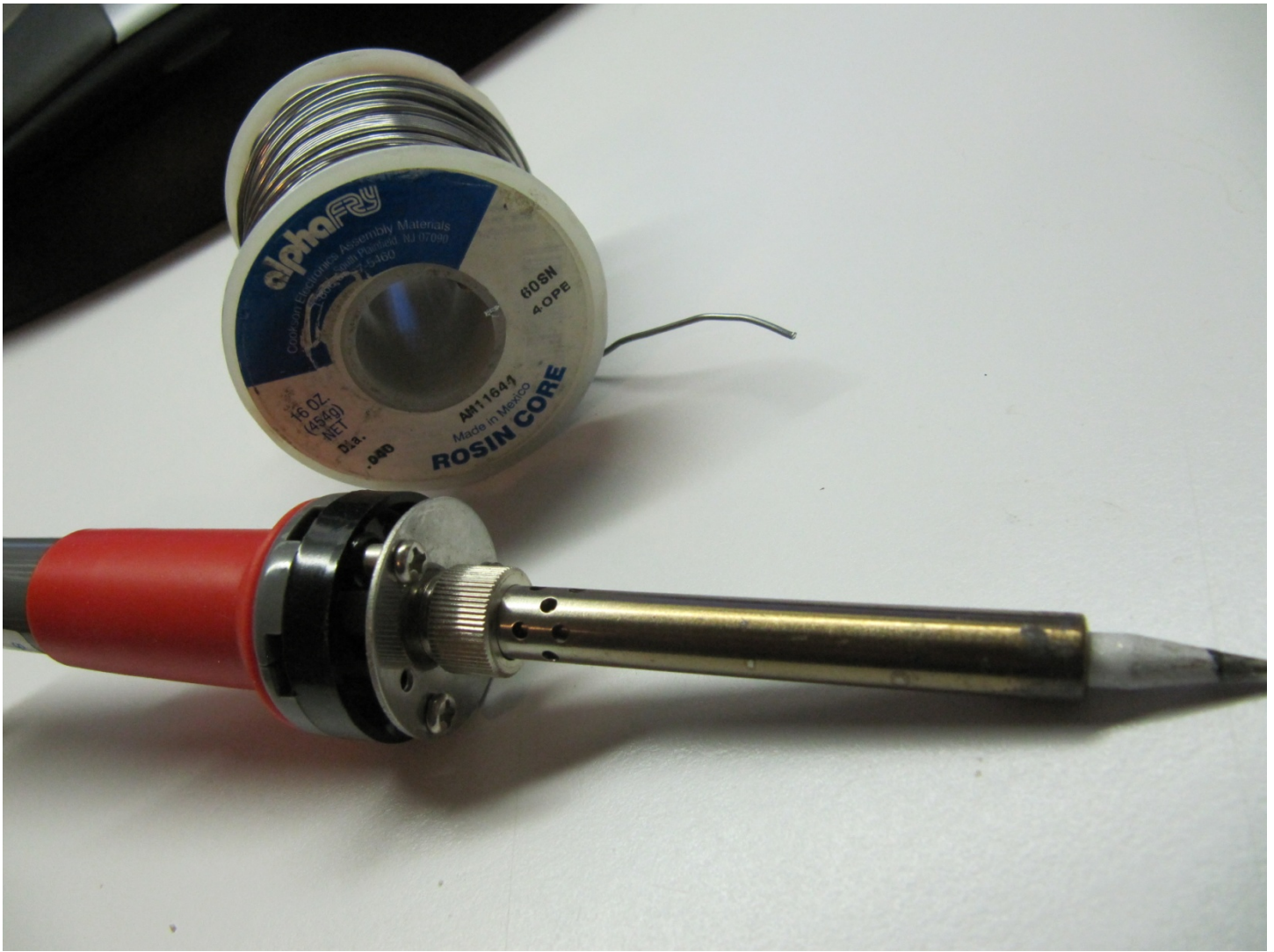
A soldering iron and solder.   If you are not familiar with what these are or how to use them, this project may not be for you. But if you know what they are and are not afraid to learn, look up "how to solder"on Google and do a little practicing to get the hang of it. The Light Strip will melt if you use too much heat on it so proper soldering techniques are a must!
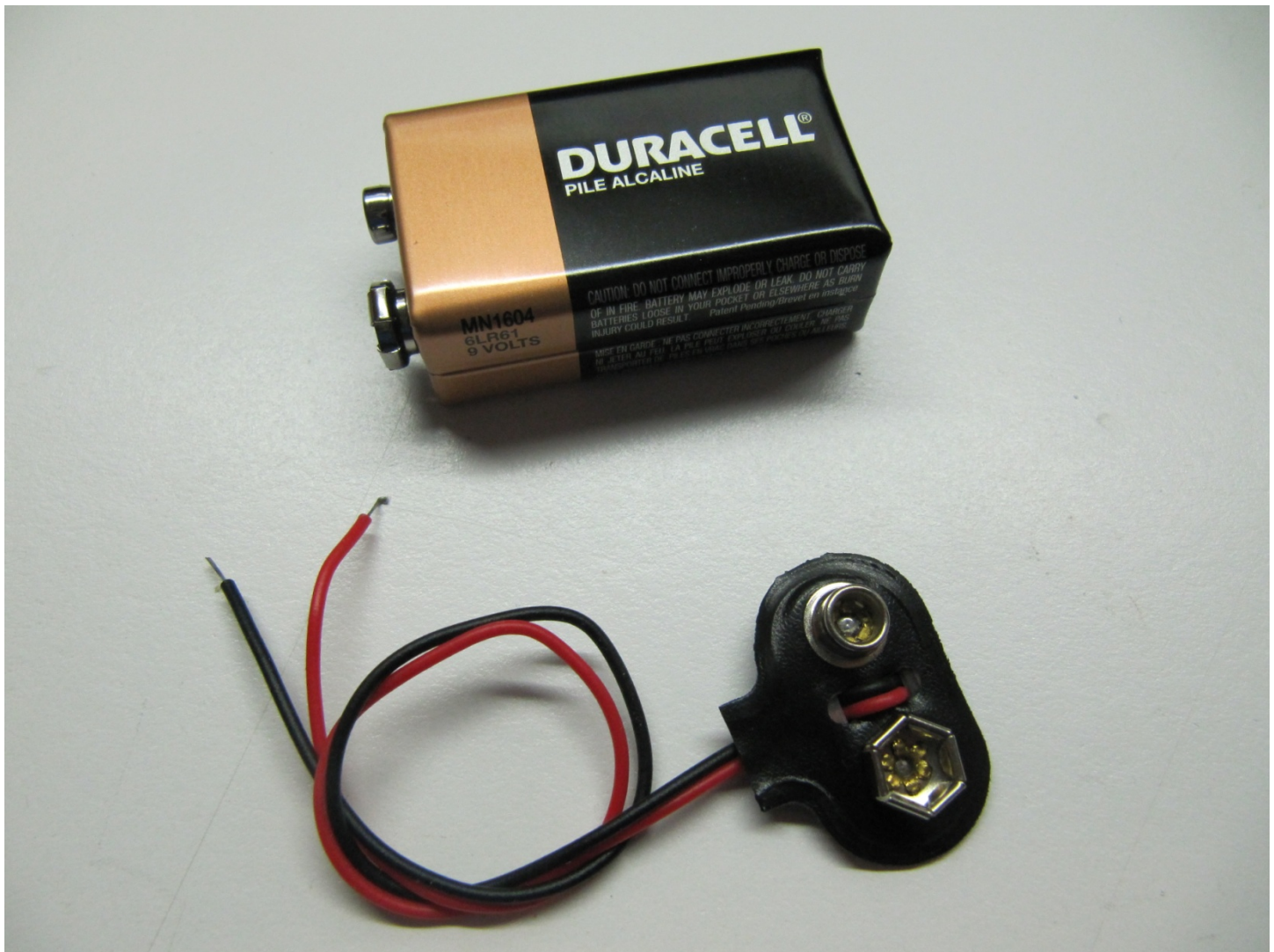
9 Volt DC Battery and 9v Battery connector. It is important to note that the smaller (male) connector on the battery is the Positive (+) side and the larger (female) connector is the Negative (-) side.   On the battery connector itself it is just the opposite. The small (male) side is Negative (-) and the large (female) side is Positive.   Also, as a general rule, Positive (+) connections are normally marked with a plus (+) sign and also are designated with the color RED where the Negative (-) connections are normally marked with a minus(-) sign and also are designated by using BLACK.

An "N" size DC coaxial power connector. It is important to note that the center tab (smaller tab) on the connector should be used for the Positive (+) connection and the larger outer tab used for the Negative (-) connection when soldering the power wires to this connector.

Push Button Switch – This can be either a momentary switch (which means it will only stay on while you continue to press the button) or a Push On/Push Off type switch. You will only need one of these as opposed to two of them that are shown in this picture. These just came two to a package. These also come in many shapes and sizes but I would

How To: Creating a Digital Light Wand for Light Painting Photography

recommend getting a larger one that will be easy to find and push since you will be dealing with it in the dark and holding it down for several seconds at a time.

A plastic project box to mount the Arduino board into along with the 9 volt battery, battery connector, the push button switch, and the rotary 6 position switch if you are going to use those. Note: There are project boxes available specifically for the Arduino boards and you can purchase one of those if you prefer. I am sure there will be openings for the USB connector and power built right into those so you won't have to do any customizations.

Six Position Rotary Switch – This particular switch is a 2 pole switch which means it is two switches in one or has two connections for each position of the switch. A single pole rotary switch will work just fine since we are only going to use one side of it. Also you will notice that the shaft is very long on this switch. A lot of them come this way so that you can cut them to length before placing a knob on the shaft. Make sure you cut it the proper length for the knob you will be using.

Control Knob to place on the 6 position rotary switch

Five 1K ohm resisters (small 1/8th watt resisters are fine) You will note that there is a color coding on the resisters and you will want to make sure the color coding is Brown, Black, Red, Gold as shown. This color coding is the designation for

1K Ohm along with the tolerance of the resister. These are used with the rotary switch and give the Arduino board something to make a "Decision" on to select different programs or patterns to run with the light strip.
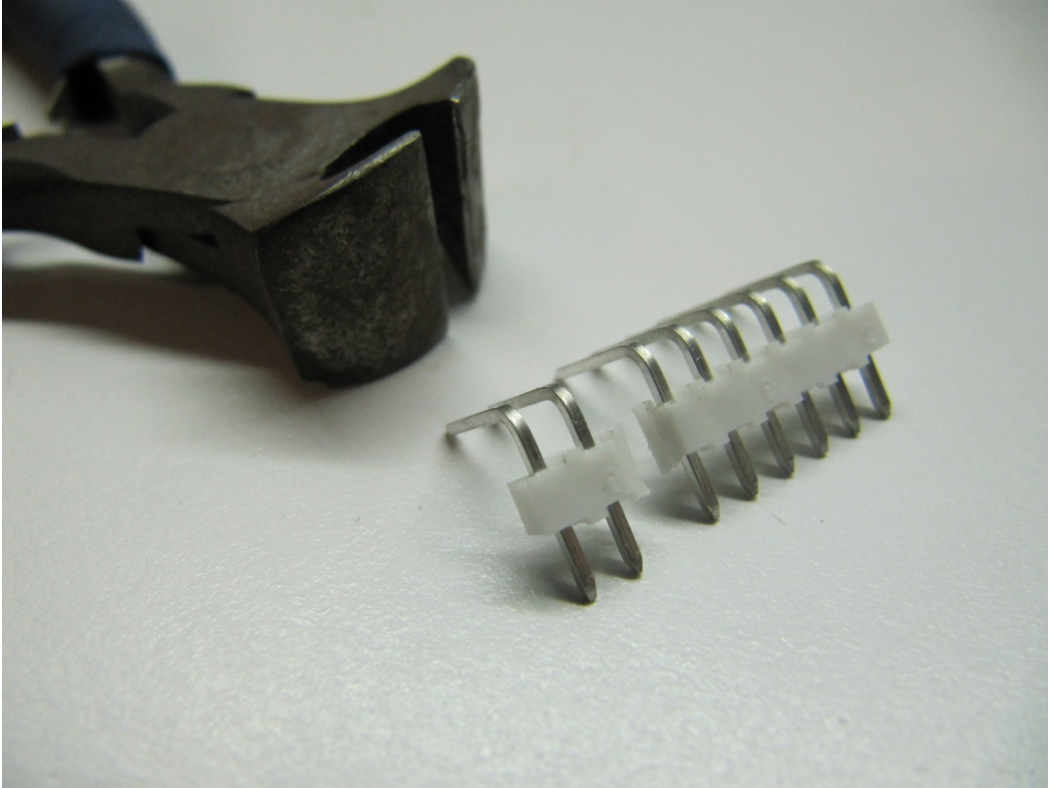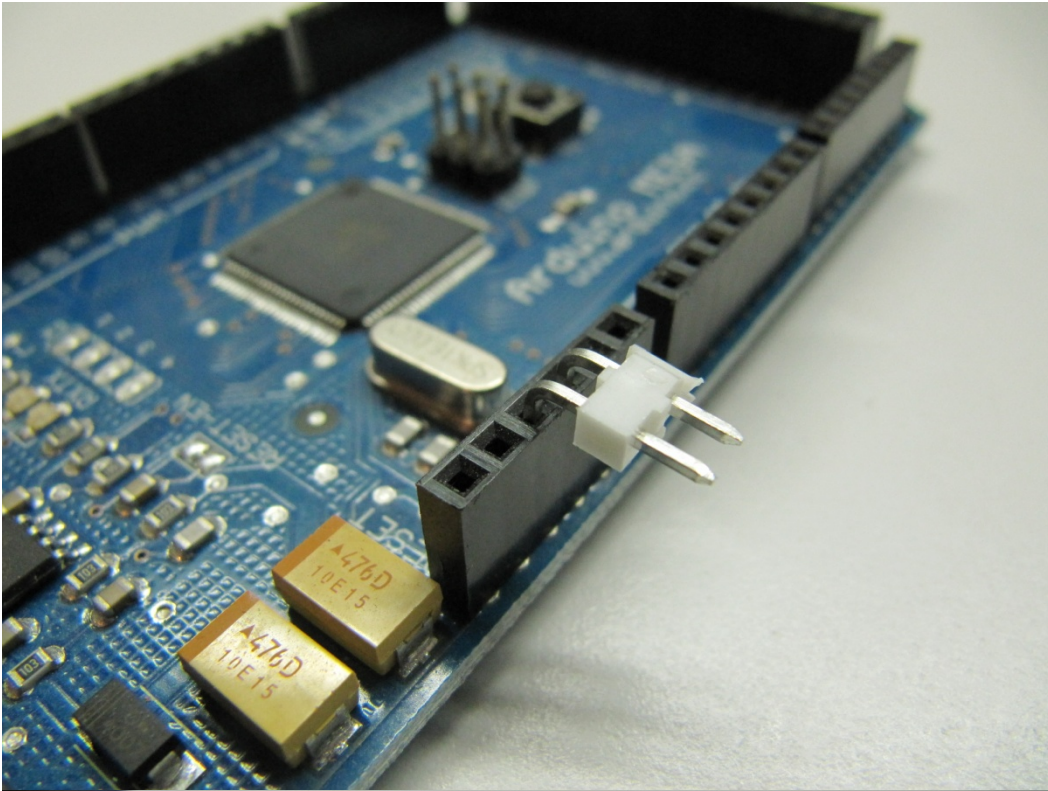
How To: Creating a Digital Light Wand for Light Painting Photography

Pin connectors to plug into the Arduino Board. These can be purchase with straight pins or 90 degree pins as the ones shown. They can be cut so that you only have to use the number of pins you want to use. A pair of wire cutters or dikes can be used to cut them or a sharp knife can be used as well.

Wire Ties (Zip Ties) – These are ALWAYS handy to have around for all sorts of things! These are the smaller ones and they are easier to use with a project this size since they are a bit more flexible. I used these to secure the Arduino board to the wood trim strip. They are easy to cut away if you need to remove them and put them back on when you need as well.

Finding and Purchasing the Addressable Light Strip

I purchased the strips that I have from a place called CoolLight in China but I am not sure they are in business any longer. There are places around that do carry it in stock and I am fairly sure it is still in production so it is just a matter of a quick search on Google. Use HL1606 as the keyword search and that should lead you to a place where you can purchase the strip from. But be aware that you may have to jump through a few hoops to purchase it.   The only way I was able to to buy it was in 5 meter rolls and I had to use bank wire transfers to pay for it. It was a bit expensive with all of that. But there are places where you can purchase shorter lengths of it.

At the time of this writing the following company offers shorter lengths of the strip online

Adafruit Industries - http://adafruit.com/products/306 (They also sell the Arduino board but the price is higher than other places I have seen online)

Purchasing the Arduino Mega Controller board

The Arduino Mega is used since it has the most program memory of all of the Arduino boards. This is important particularly when using a strip that has more LEDs on it. I have created both a programmable 16 LED version of the strip and also a 48 LED version of the strip and those lengths are what is covered in this tutorial. For the 16 LED version, I have added a 6 position switch to allow me to switch between 6 different patterns on the fly. This along with the programming memory space needed for a more complex 48 LED program, the memory capacity of the Arduino Mega is a must. I am sure that other Arduinos made in the future will have more memory and that will make it even better. There also may be ways to increase the memory capacity of the Arduino but I have not looked in to that path.

How To: Creating a Digital Light Wand for Light Painting Photography

GETTING STARTED

Okay, now that we have what we need to do the project, we can get down to the business of assembling it all. One of the first things you have to decide is how you will be mounting the light strip. The strip is flexible and can be mounted to just about anything and can be straight or curved or even shaped if you want to create a more specialized effect. One thing to remember if you are going to mount the strip on a curved or shaped surface, the LEDs on the strip are mostly directional and do not have a wide angle in which the light is emitted from them. So keep that in might just as you would for all Light Painting tools.

I mounted both the shorter and longer strips that I have to a length of wood trim that is about the same length and width of the strip itself. Wood trim is light and sturdy enough to keep the tool from moving around when using it for light painting. Be careful about using anything metal since the back of the strip has connections that can be shorted out and will probably give you headaches down the road if not from the start! If you use metal or something like an aluminum strip to mount the light strip to, make sure you insulate the metal to keep the strip connections from making direct contact.

Here is a photo of the long and short strip side by side

Notice that the long light strip has the Arduino board mounted to the back side of the wood strip. It is hard to see this in the photo but there is a 9 volt battery taped to the back side of the long strip also and the power plug at the end of the wire shown is just plugged into the Arduino to start whatever program is loaded running.

23

Creating the 48 LED Light Strip
Step 1 – Mounting the Light Strip

Cut a piece of wood trim to length for the light strip to be mounted to. I used lattice trim for my project which is a piece of wood trim that measures 1 1/8 inch by 1/4 inch by 59 1/2 inches (28mm by 6mm by about 15cm)

The 48 LED length of the strip is 58 3/4 inches long or about 14.92cm. You can cut it a little longer if you wish but do not make it any shorter than this. You can also use any other material to mount the strip to but be aware that if you use metal or some other type of conductive surface, you will need to insulate it from the light strip in some way to keep the connections from shorting out.

You can mount the LED strip to the wood or other surface using double sided tape, clear cellophane tape, RTV silicone sealant, wire ties (zip ties), or any other method you wish. I used clear cellophane tape as shown below. (I know… certainly not the optimal mounting method but it has worked well for a year and half of use in the field. And this tool has been abused a lot! This includes bouncing around loosely in the back of a trailer for over 3,500 miles!!
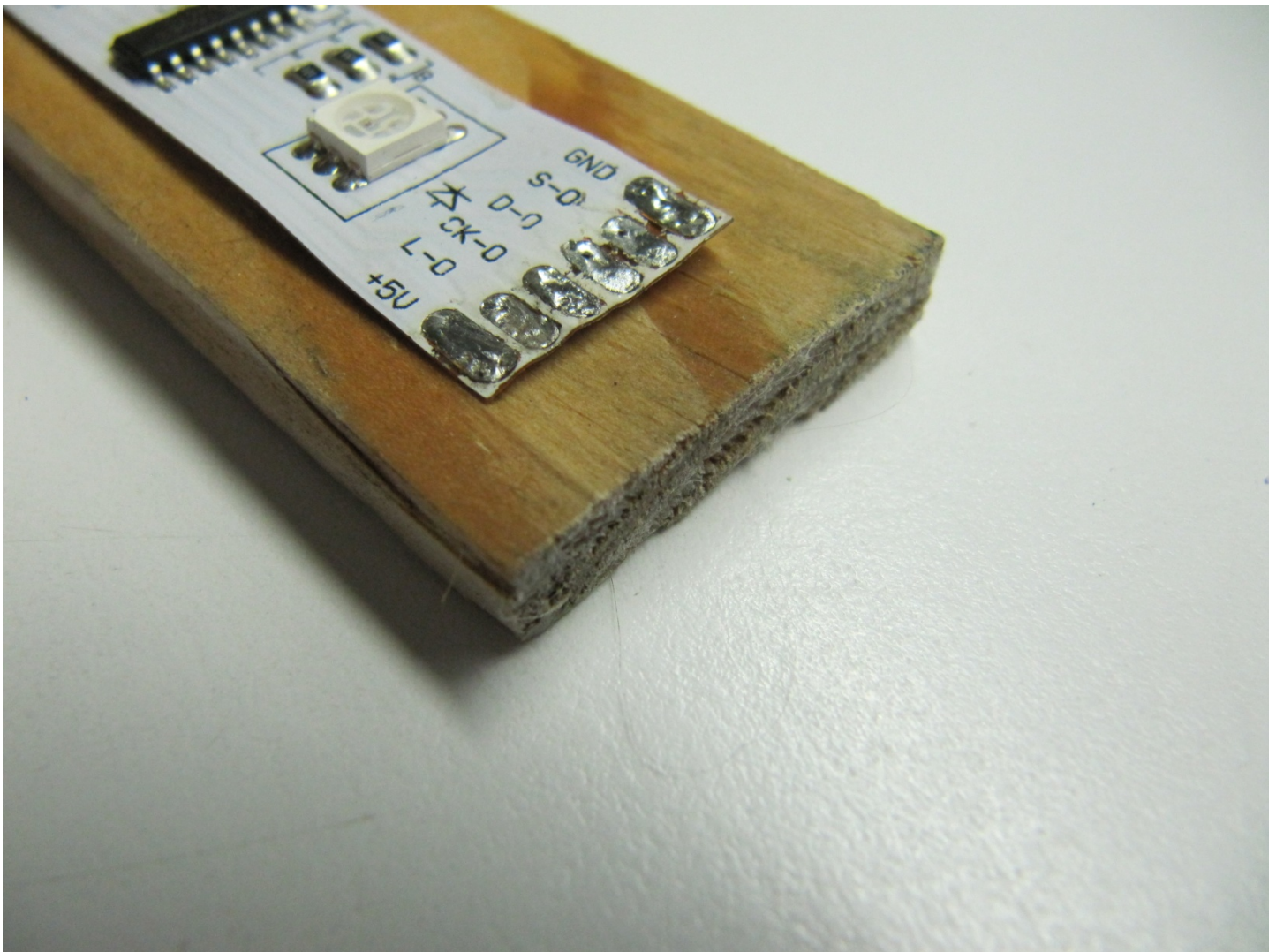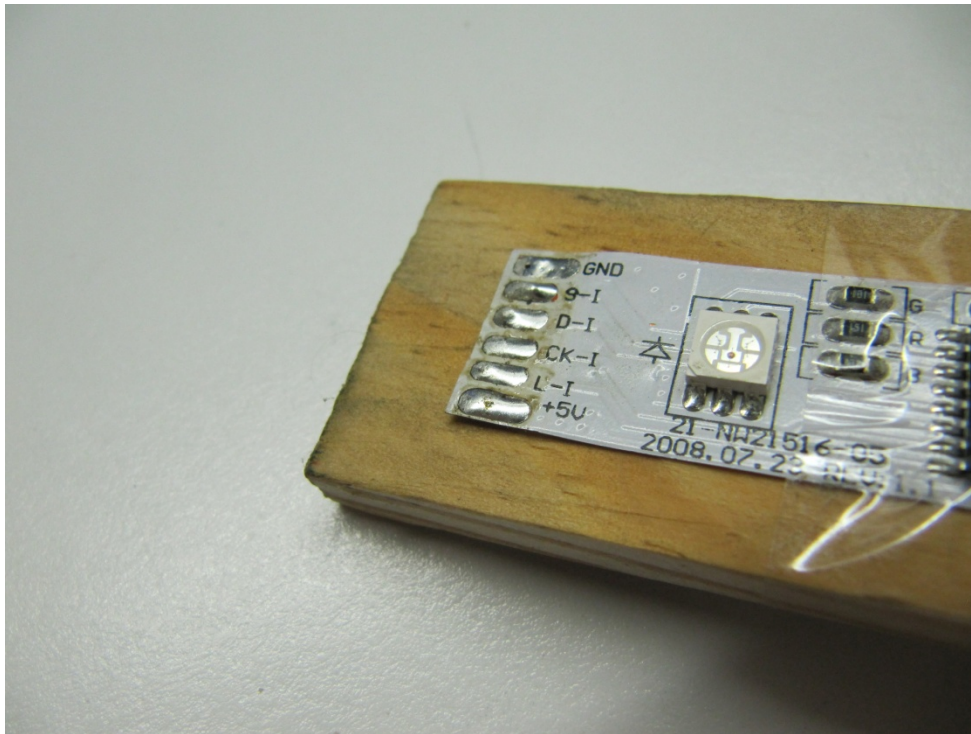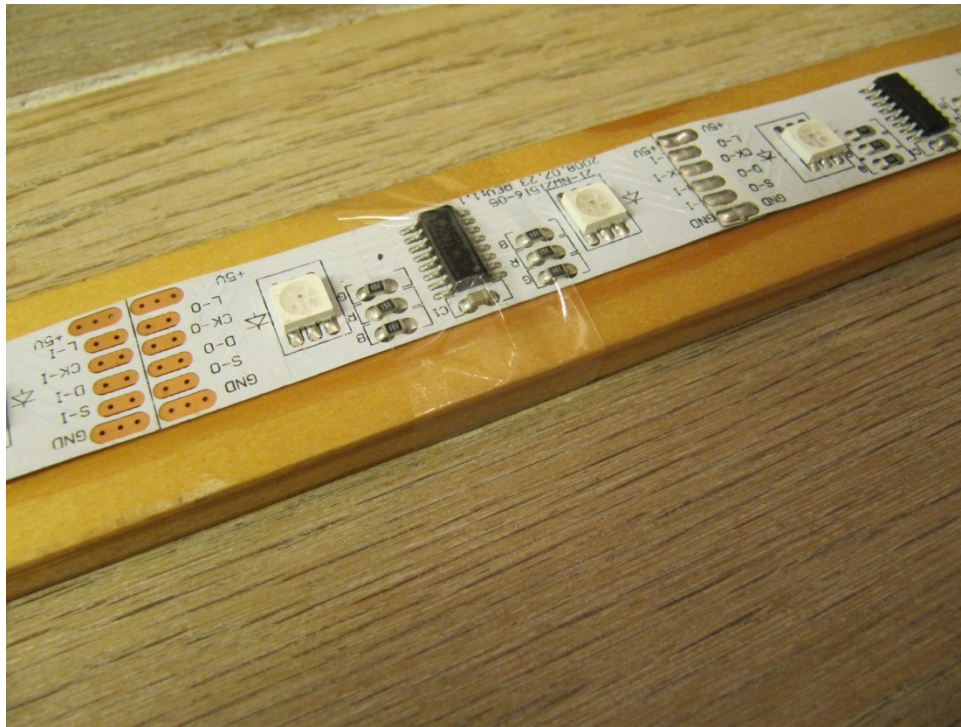
Here you can see the cellophane tape that I used to secure the light strip to the wood trim. I used tape all the way around the wood on every other HL1606 chip. I did not use tape over the LED's since they tend to get a little warm when turned on for a bit but I have had no problems with heat on the chips.

Step 2 – Mounting the Arduino board to the assembly

THIS IS VERY IMPORTANT!!!! Take a look at the ends of the LED strip that you mounted to the wood and look for the markings that identify one end as the "INPUT" end and the other as the "OUTPUT" end. The "INPUT" end is indicated by the "I" after the S, D, CK, and L markings such as S-I, D-I, CK-I and L-I. The "OUTPUT" end is indicated by the "O" after the S, D, CK, and L markings such as S-O, D-O, CK-O, and L-O. The Arduino board will need to be connected to the "INPUT" end of the strip so make sure you are mounting the Arduino board to the same end the "INPUT" is on! See the photo below and make sure you have this correct. If you mount it to the wrong end the strip will not work! You can see this in the photos on the previous page.
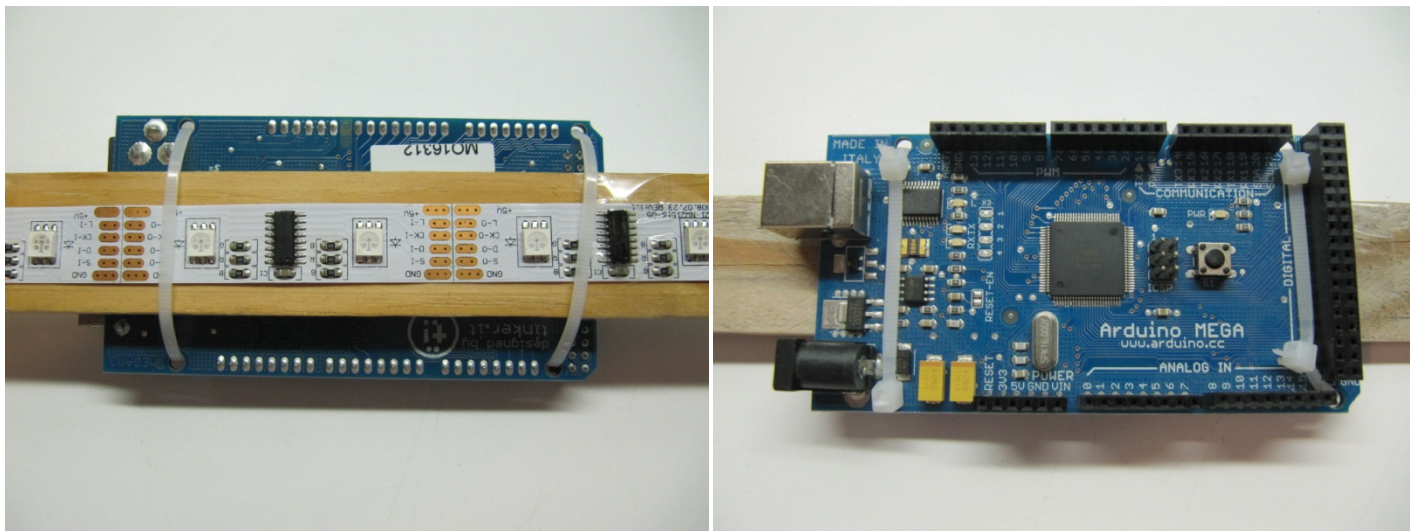
I recommend using small wire ties (or zip ties) to secure the Arduino board to the wood/LED assembly. It is quick, easy, and works just fine for what we need. See the photos above as an example. Note that I routed the wire ties through the holes in the Arduino board to make sure it is secure. I used black electrical tape for this at the beginning but it did not last long that way. If you decide to do that, make sure you do not cover any of the LEDs on the light strip.

How To: Creating a Digital Light Wand for Light Painting Photography

**Step 3 – Prepping the wires and Soldering the wires to the LED Strip**

Once you have the LED strip mounted to the wood or whatever it is you are using to mount it on, it is time to solder the wires to the strip. You will need 6 wires for this. I recommend 20 or 22 gauge wire (solid or stranded). You can cut the wires to a length of about 6 inches to give you some extra to work with. We will be cutting them to size when we connect them to the Arduino board. I used a different color for each of the wires to make it easier to see what goes where with the connections but the color is really not all that important.
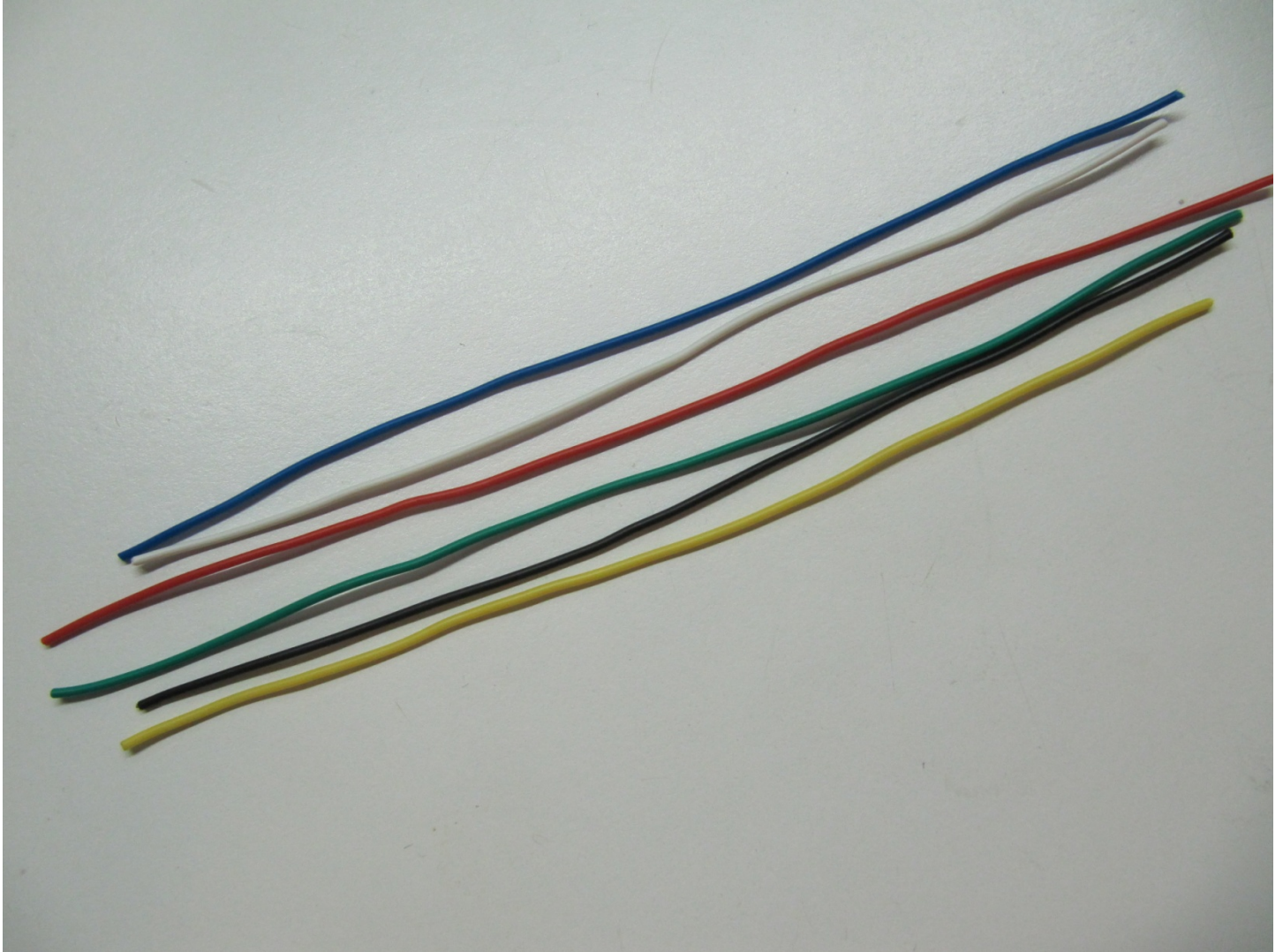
Six 22 Gauge wires cut to a length of 6 inches each.

Strip the insulation on one end of each of the six wires. We will need one end bare at this point to solder the wires to the RGB LED Light Strip. You can strip the insulation off about 1/4th of an inch.
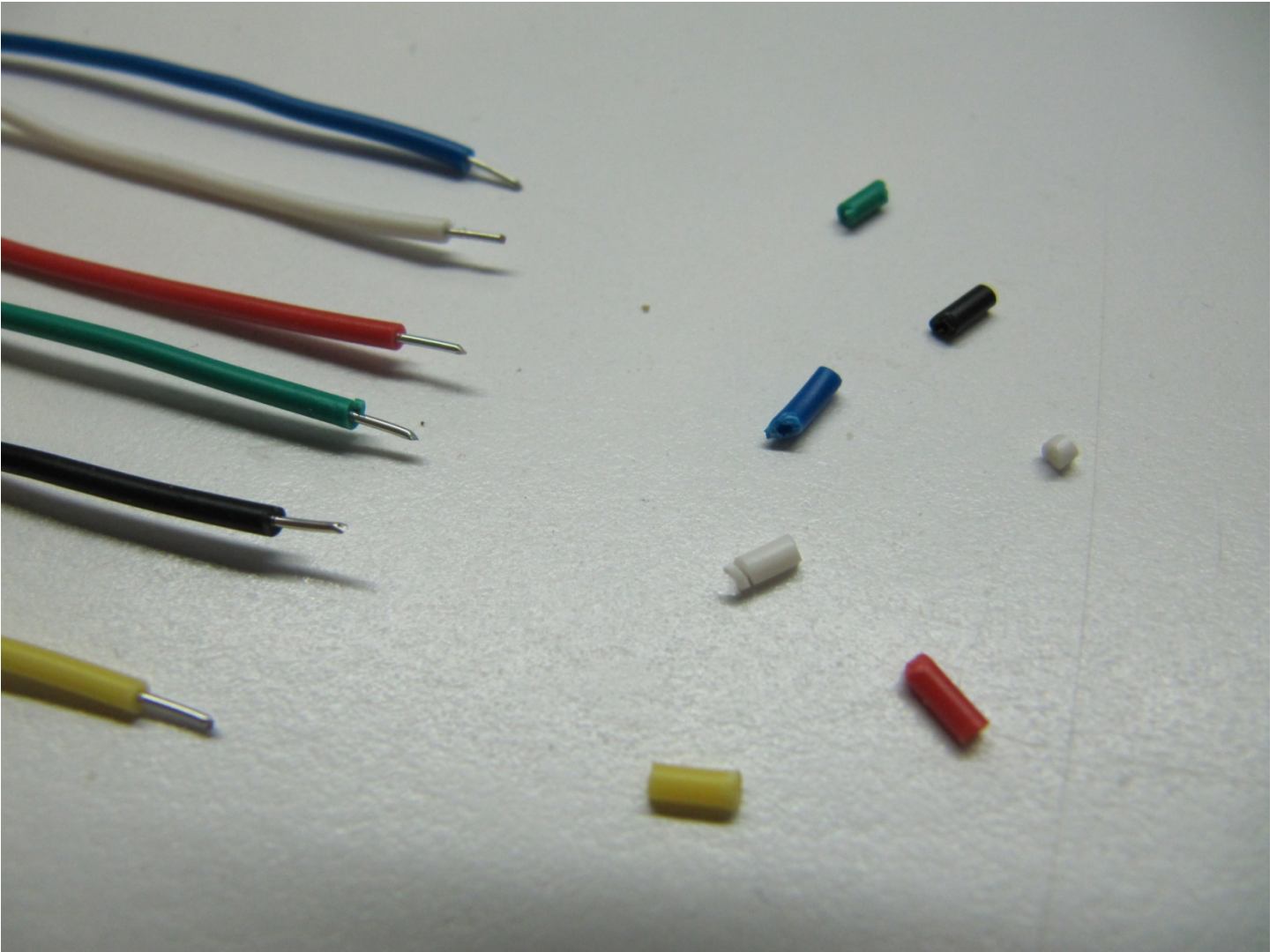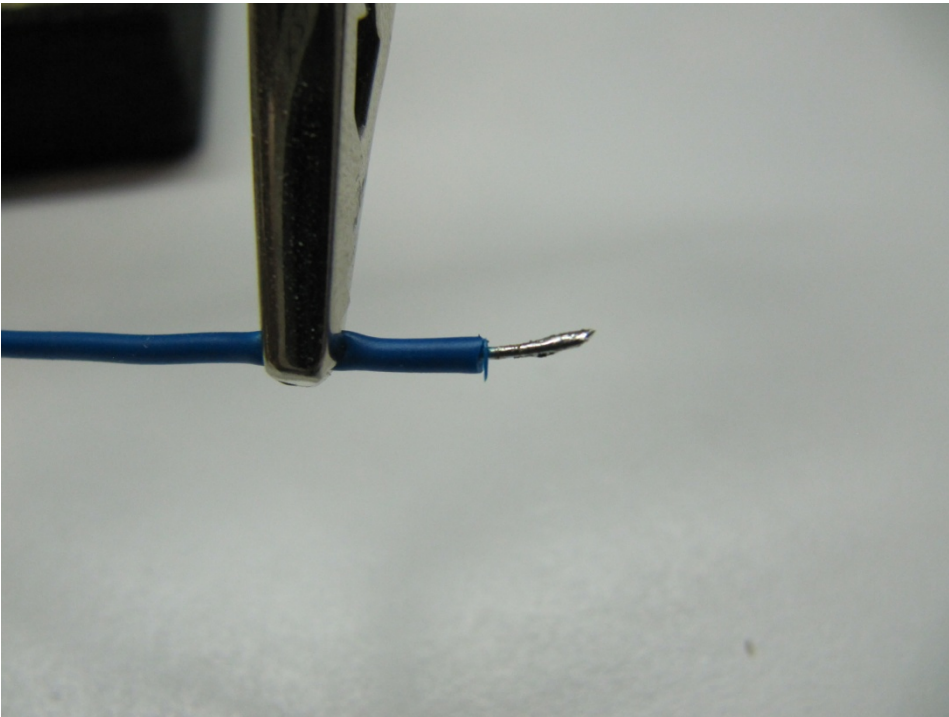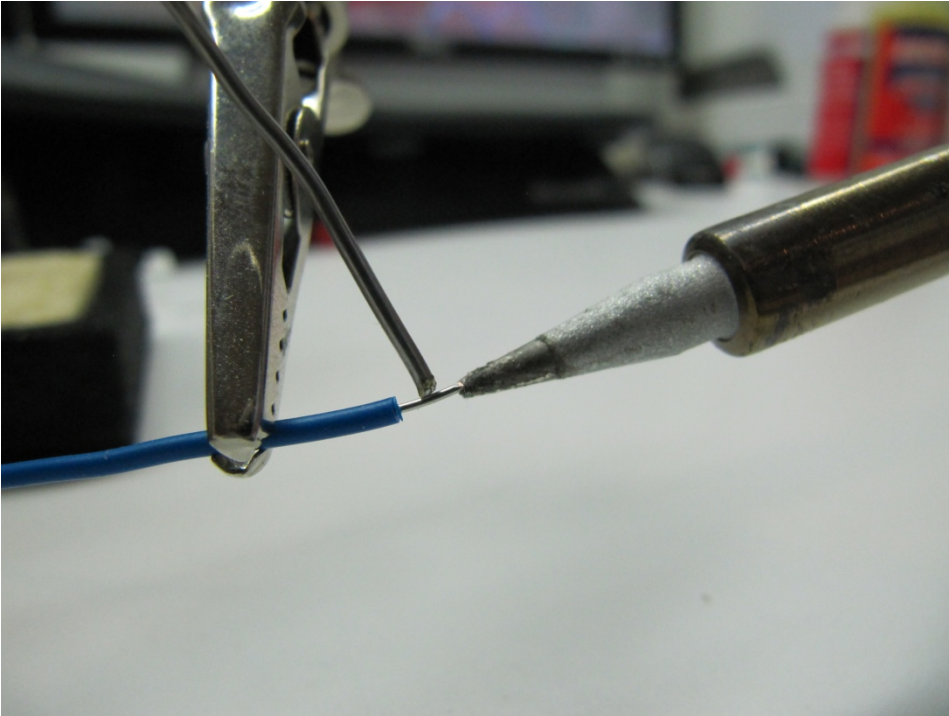

Each of the six wires with the insulation cut off about 1/4th of an inch from the end.

It is common practice when soldering wire to "Tin" or put a thin layer of solder on wires to make them easier to solder to a connection and get a good solid solder joint. Below are two photos that show the process and what the wire looks like after it is "Tinned".

How To: Creating a Digital Light Wand for Light Painting Photography

Now solder each of the wires to the LED Light Strip. Double check one more time that you are doing this on the INPUT end of the strip.
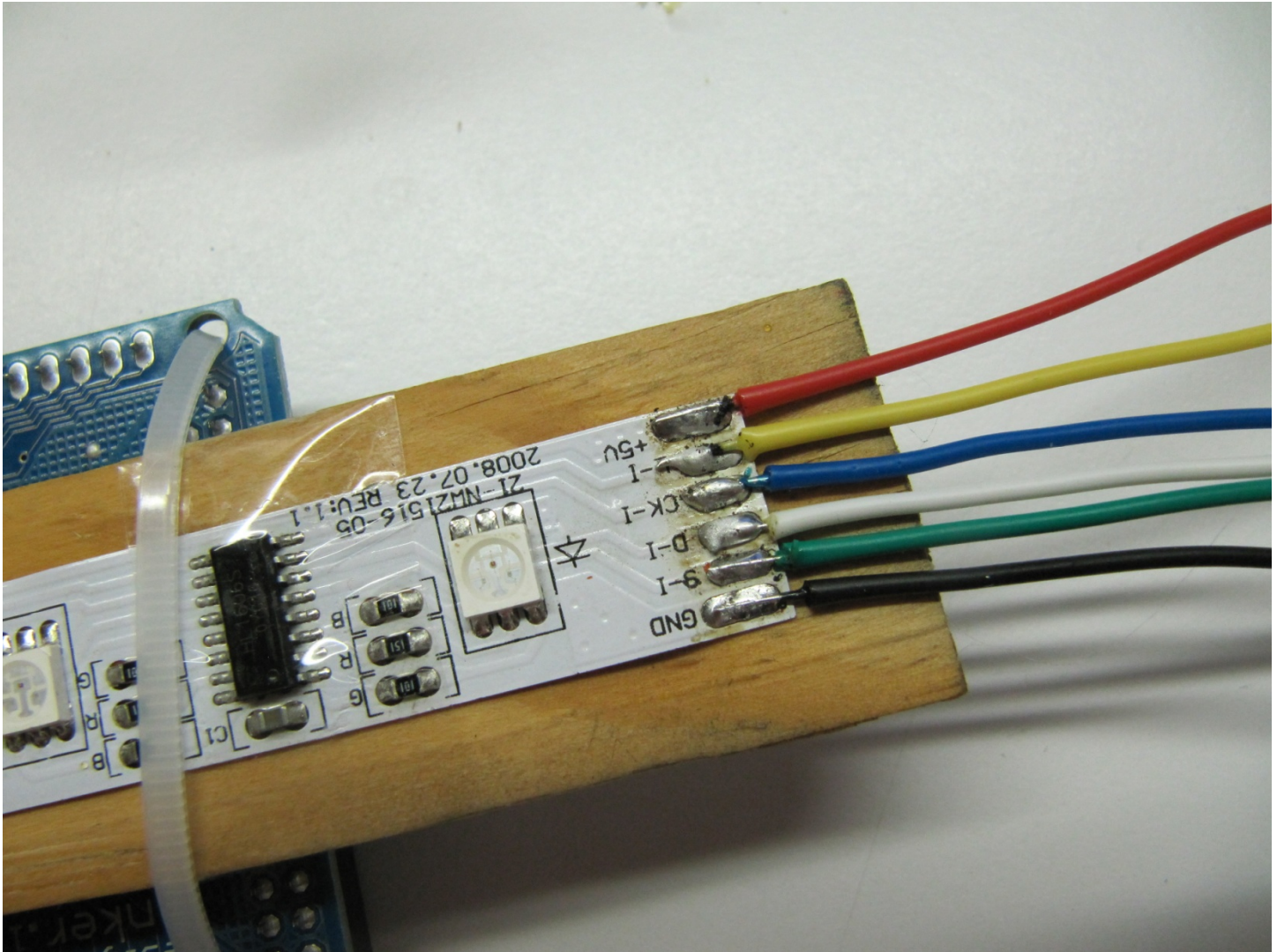
Here are the wires soldered to the connection pads on the light strip. Make sure that you do not use so much solder that the connections touch one another causing a short. Note that I used the red wire for the positive 5 volt connection and the black wire for Ground. This is sort of a universal rule of thumb to follow for power connections.

How To: Creating a Digital Light Wand for Light Painting Photography

Step 4 – Routing the wires and connecting the other end to the Arduino board

Route the wires toward the connections on the Arduino board where they will ultimately be connected. Use the following as a guide with regards to which connection goes to which pin on the Arduino.

CONNECTION GUIDE

On the RGB LED Light Strip there are 6 connections. They are labeled as follows:

GND (This is the Ground or Negative(-) power connection)
S-I (Sync / Speed Clock Input)
D-I (Data Input)
CK-I (Clock Input)
L-I (Latch Signal Input)
+5V (This is the Positive (+) 5 Volts DC connection)

Note: Please be aware that there is also an "Output" end of the strip or sections of strip as well and these are designated by "S-O", "D-O", "CK-O", and "L-O". Make sure you are connecting to the "Input" side of the strip.

Each of these connections will connect directly to a pin on the Arduino Mega Board.

The Pin assignments are as follows:

GND -> Connect to GND Pin on the Arduino Mega
S -> Connect to PWM Pin 9 on the Arduino Mega
D -> Connect to PWM Pin 10 on the Arduino Mega
CK -> Connect to PWM Pin 4 on the Arduino Mega
L -> Connect to PWM Pin 3 on the Arduino Mega
+5V -> Connect to 5V Pin on the Arduino Mega

The specific pins that I used on the Arduino were just to keep things separated a bit for the wiring, particularly the pins used in the PWM area. Any of the PWN pins will work but you will have to make sure the programming code that is downloaded to the Arduino reflect the changes made and reference the correct pin. What all of these do is not really important unless you want to dig more into digital electronics a lot more. Here are the basics of how it all works. The GND and +5V connections supply power to the strip and the S, D, CK, and L connections are used to send commands to the HL1606 chip on the strip. (These are controlled in a chain on the strip so when you "push" a data command to the strip, you are pushing it to the first LED on the strip, the second command pushes another command to the first LED and in turn pushes the command that was already there to the second LED and so on for the full length of the strip. So if you have 16 LEDs in the strip, you will need to push 16 commands to it to tell each LED what to do for each cycle.

All of the code for doing this is automatically generated using the Spreadsheet that I created that allows you to "Draw" a picture directly in the spreadsheet. This will generate the commands that need to be sent to the Arduino Mega via the Arduino interface and USB cable.
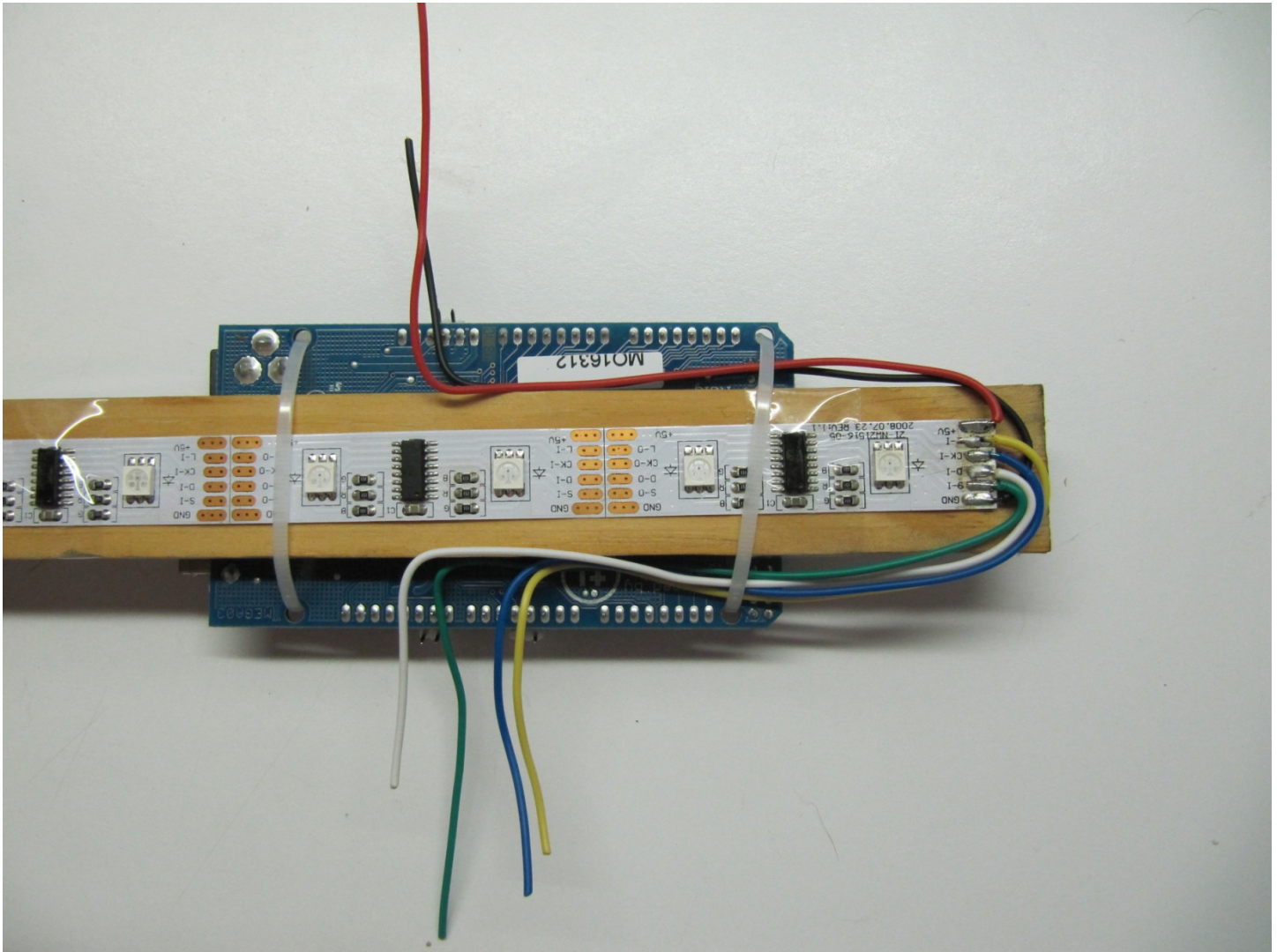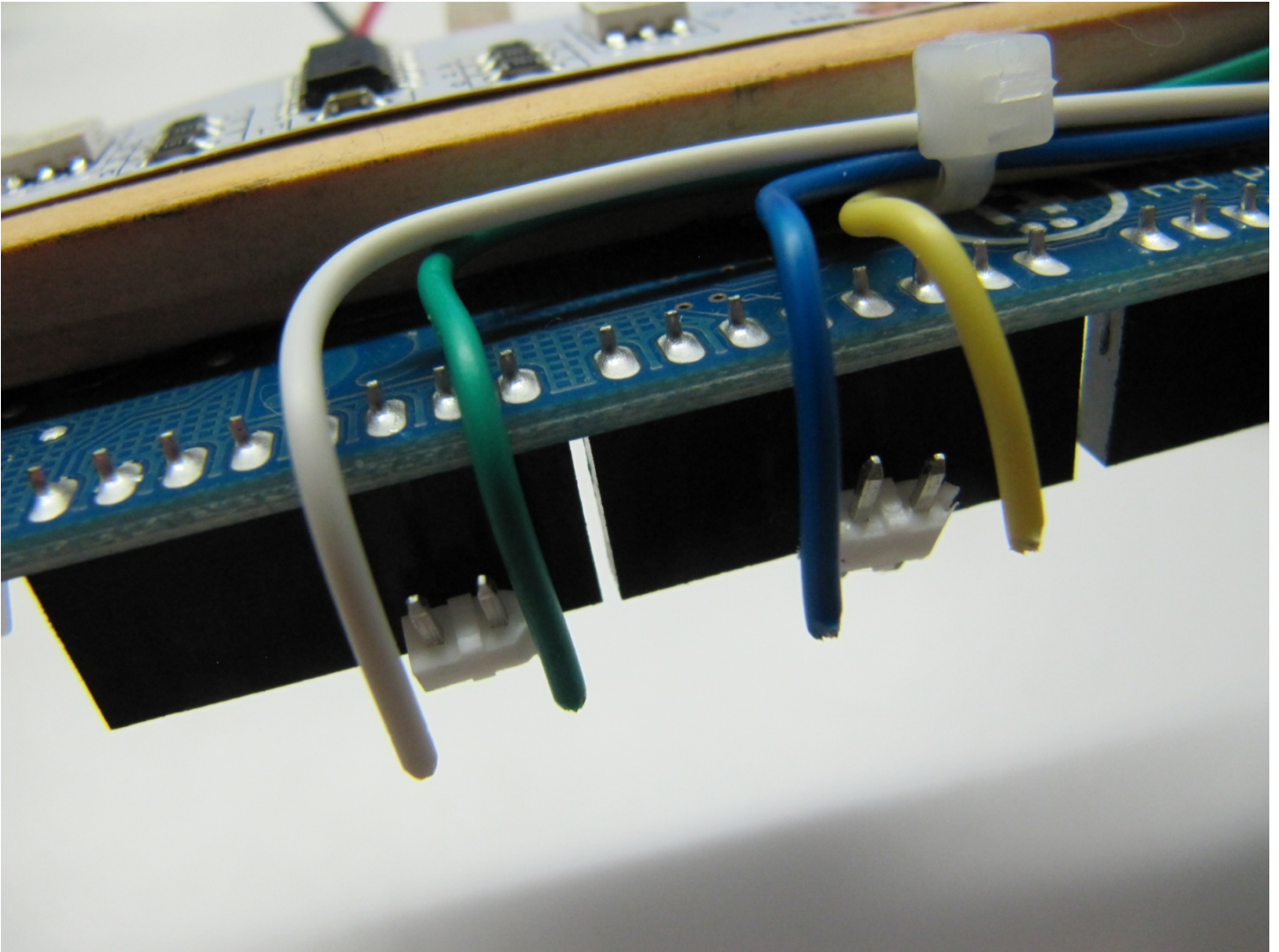
Here are the wires routed to the general area they will be connected to.

How To: Creating a Digital Light Wand for Light Painting Photography

Next, cut the wires to length making sure that you have a little extra room to work with.   Note that I used more wire ties to help keep the bundles of wire together on each side of the board.

**How To: Creating a Digital Light Wand for Light Painting Photography**

Now you will need to strip the insulation from this end of the wires so that we can solder them to the connectors that I already have plugged into the Arduino board. I bent the connectors down slightly towards the wires to make the connection easier to solder. Then solder the wires to the connectors as shown.
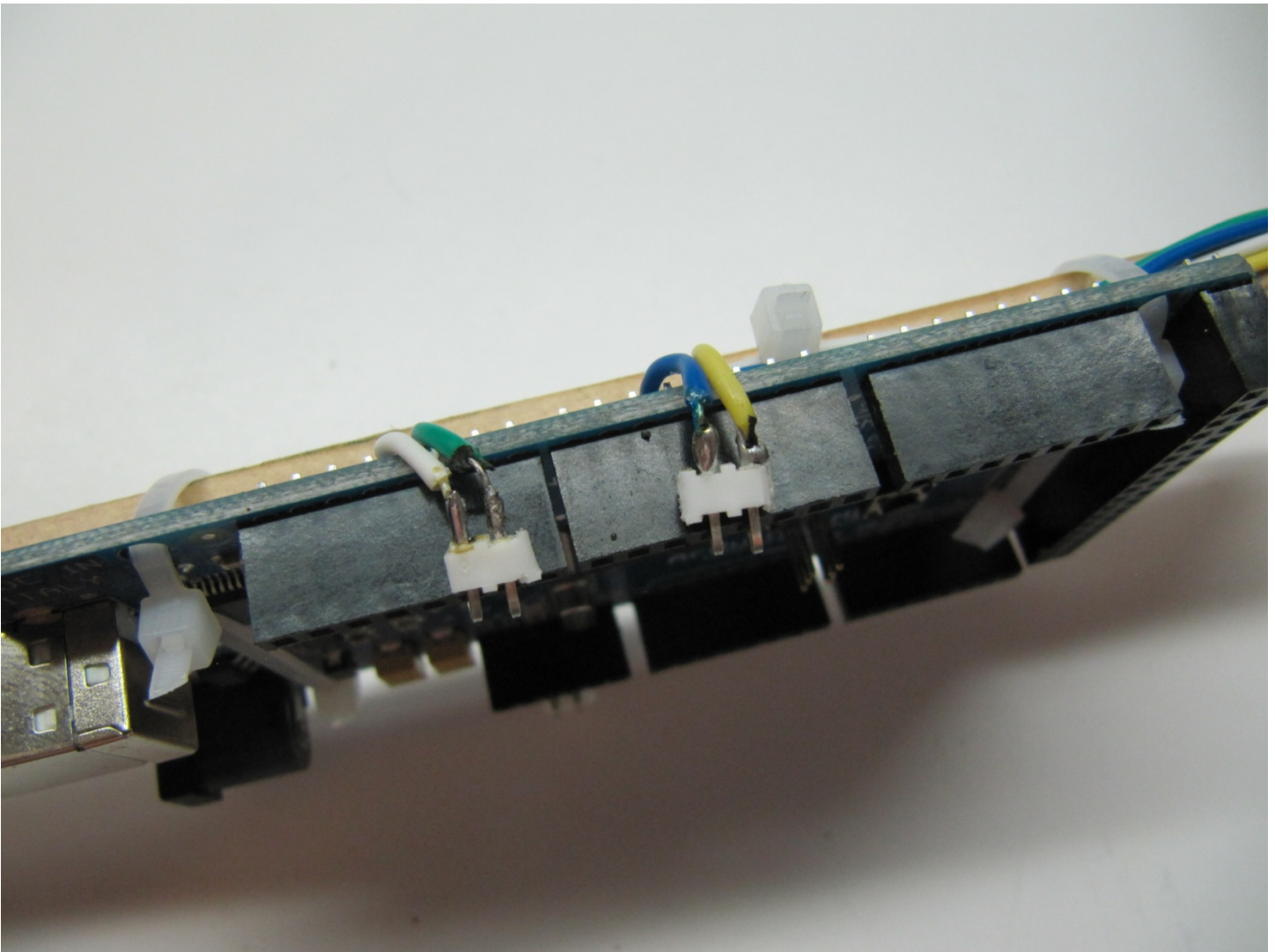
Solder the wires to the connector for the power connection side as well.
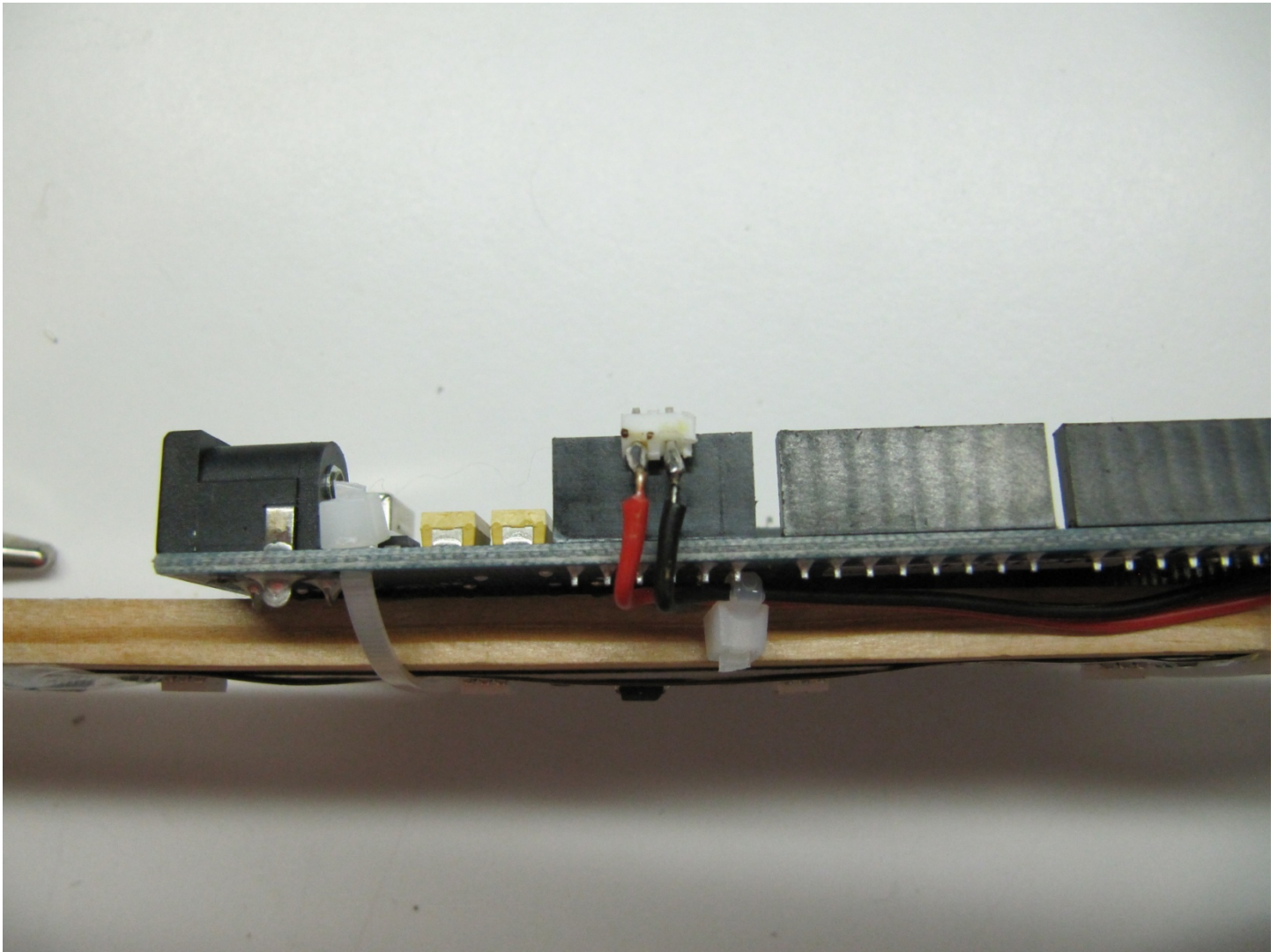
One thing that I would like to note here… On the first version of this that I created, I did not use any connectors on the Arduino board. I simply used 20 gauge wire and plugged the stripped end of the wires directly into the pin connectors on the Arduino board. This worked just fine and did not give me any problems except on a few occasions where a wire pulled out while bouncing around in the back of my truck!   It doesn't HAVE to be fancy… it just needs to work! ;-) The main reason I am doing the tutorial with connectors is that it is much more solid connection and should not give you the same problems I had, even if they were few and far between.

How To: Creating a Digital Light Wand for Light Painting Photography

Step 5 – Creating the power connection wire for the 9 volt battery

For this you will need the 9 volt battery connector, a length of black wire, a length of red wire, and the "N" size power connector that plugs into the Arduino board.

Solder the red wire to the CENTER connection on the "N" connector and black wire to the OUTER connection on the "N" connector as shown below. Double check to make sure you have these correct and that the connections do not touch each other when you slip the connection shield over the connection to screw it on. You can use a piece of plastic or black electrical tape between the two connections to help ensure that the two connections will not short out or touch each other.

Then connect the black and red wires to the 9 volt battery connection and wrap them in black electrical tape to make sure they are insulated from each other and do not touch. You will end up with an assembly that looks as follows:
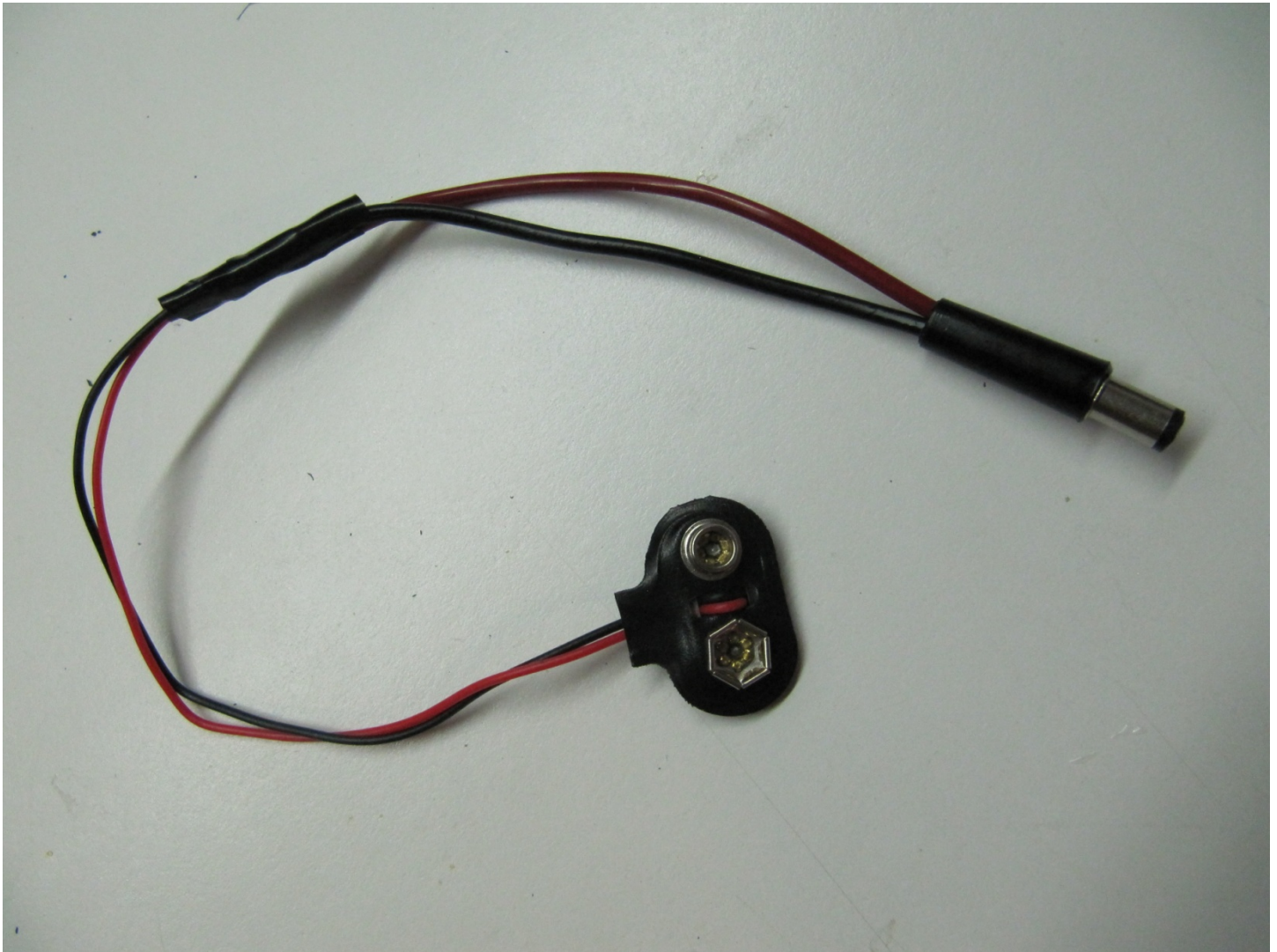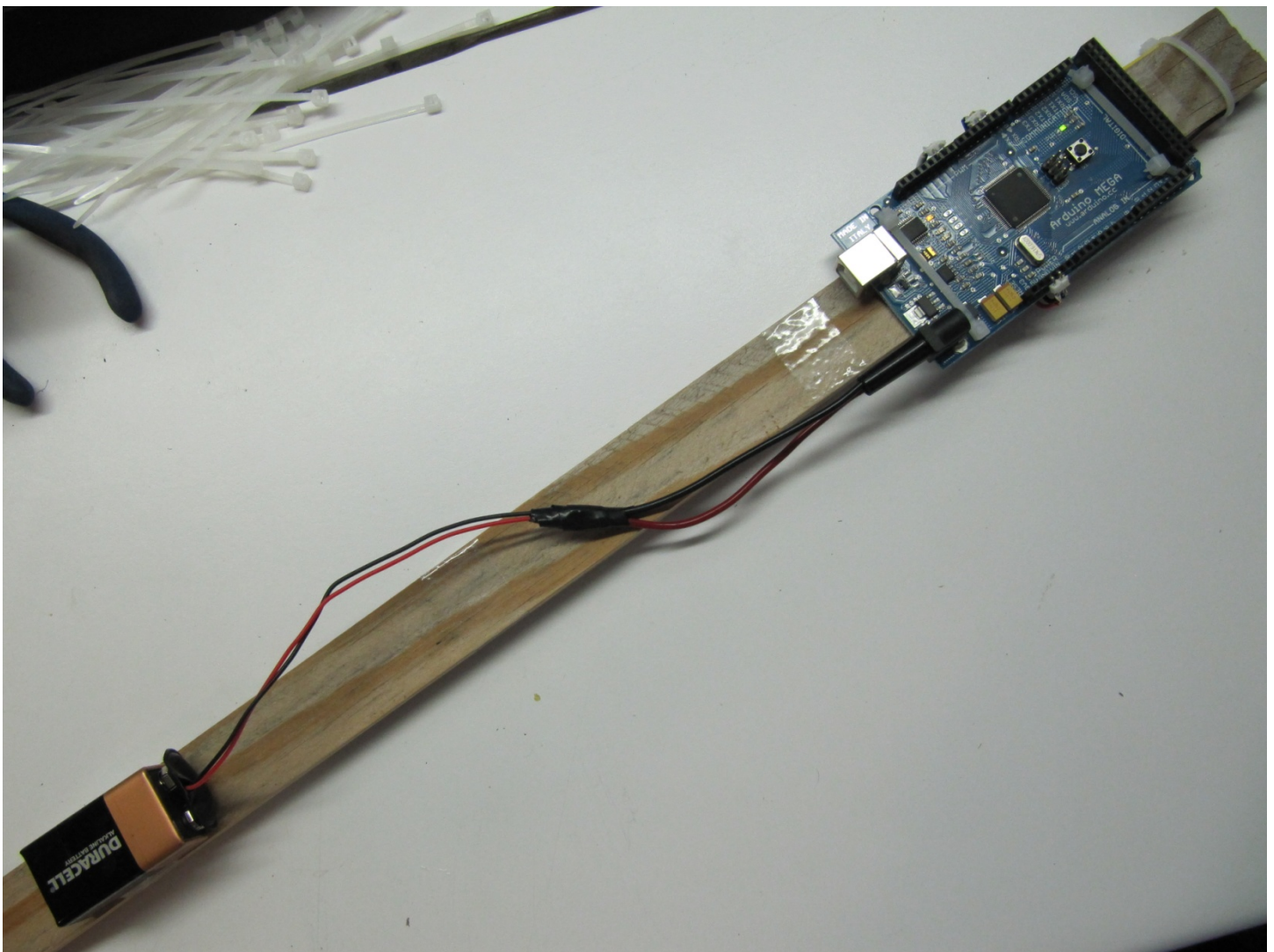
How To: Creating a Digital Light Wand for Light Painting Photography

Step 6 – Mounting the battery to the light strip assembly and plugging the power in to test it.

Plug the 9 volt battery into the connector assembly that you made in the last step. Note, if for any reason there is a short in the "N" connector between the black and red wires, when you make this connection the wires may start to heat up quickly. If this happens disconnect the assembly from the battery and correct the problem before continuing any further!

If the wires do not heat up, then the connection is fine and you can continue. Mount the battery to the light strip using cellophane tape or any other method you want to use. I have used the same 9 volt battery for over a year so it is not something you will be changing a lot unless you run the light strip a lot. Make sure you leave yourself some slack in the connection assembly so that you can plug in and unplug the "N" connector on the Arduino board easily. The Arduino board will run its program as soon as you plug the power in and will stop when it is unplugged. You can use a switch for this if you prefer but this method has worked fine for me on the long strip.

Tape the battery to the strip as shown below
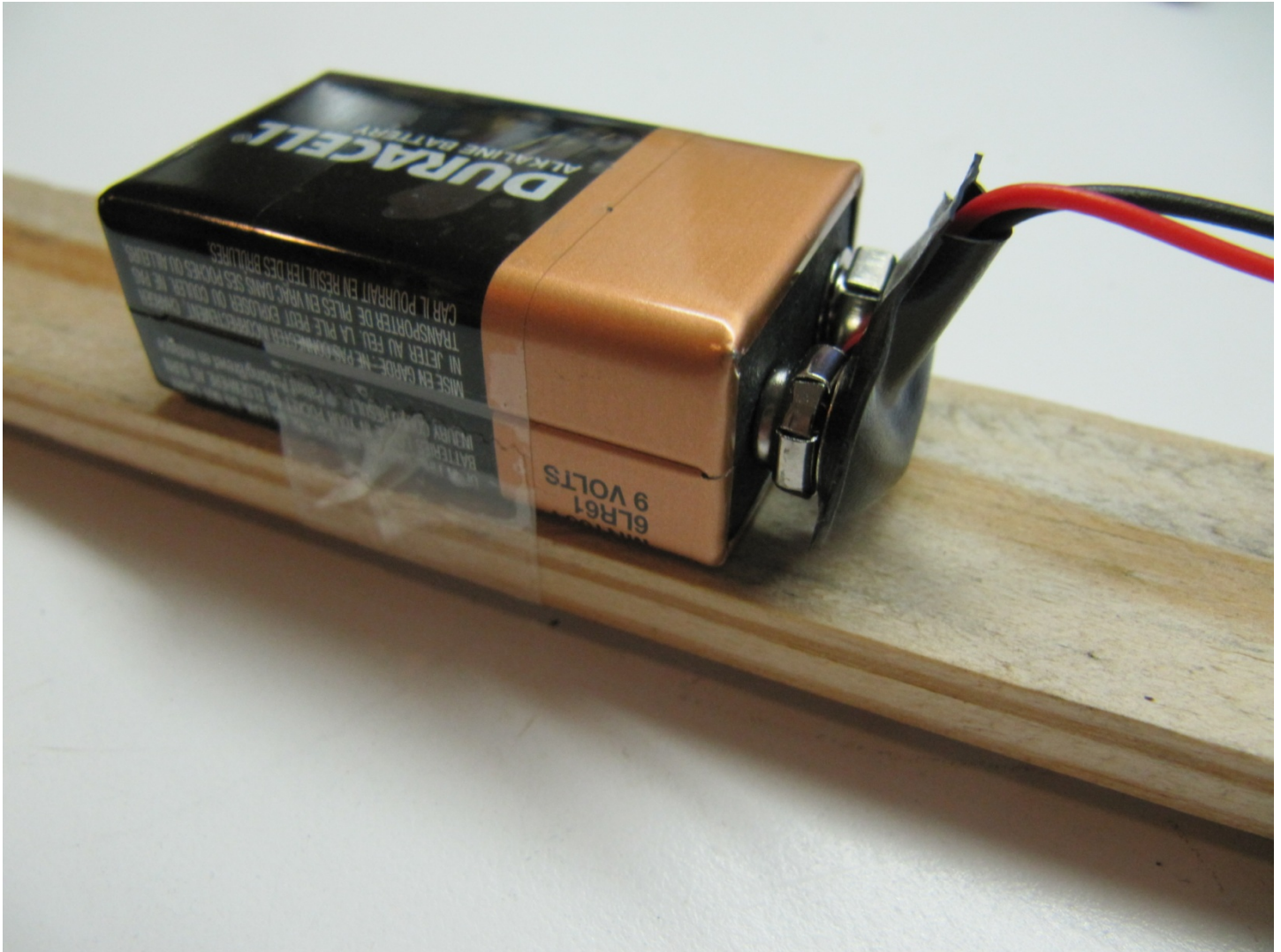
How To: Creating a Digital Light Wand for Light Painting Photography

When you plug the power connector into the Arduino board you should see the small green power LED light up as shown in the upper right of the photo below.   As long as you see this, the assembly of the strip is now complete and you are ready to download a program to the strip and start using it!!! ☺
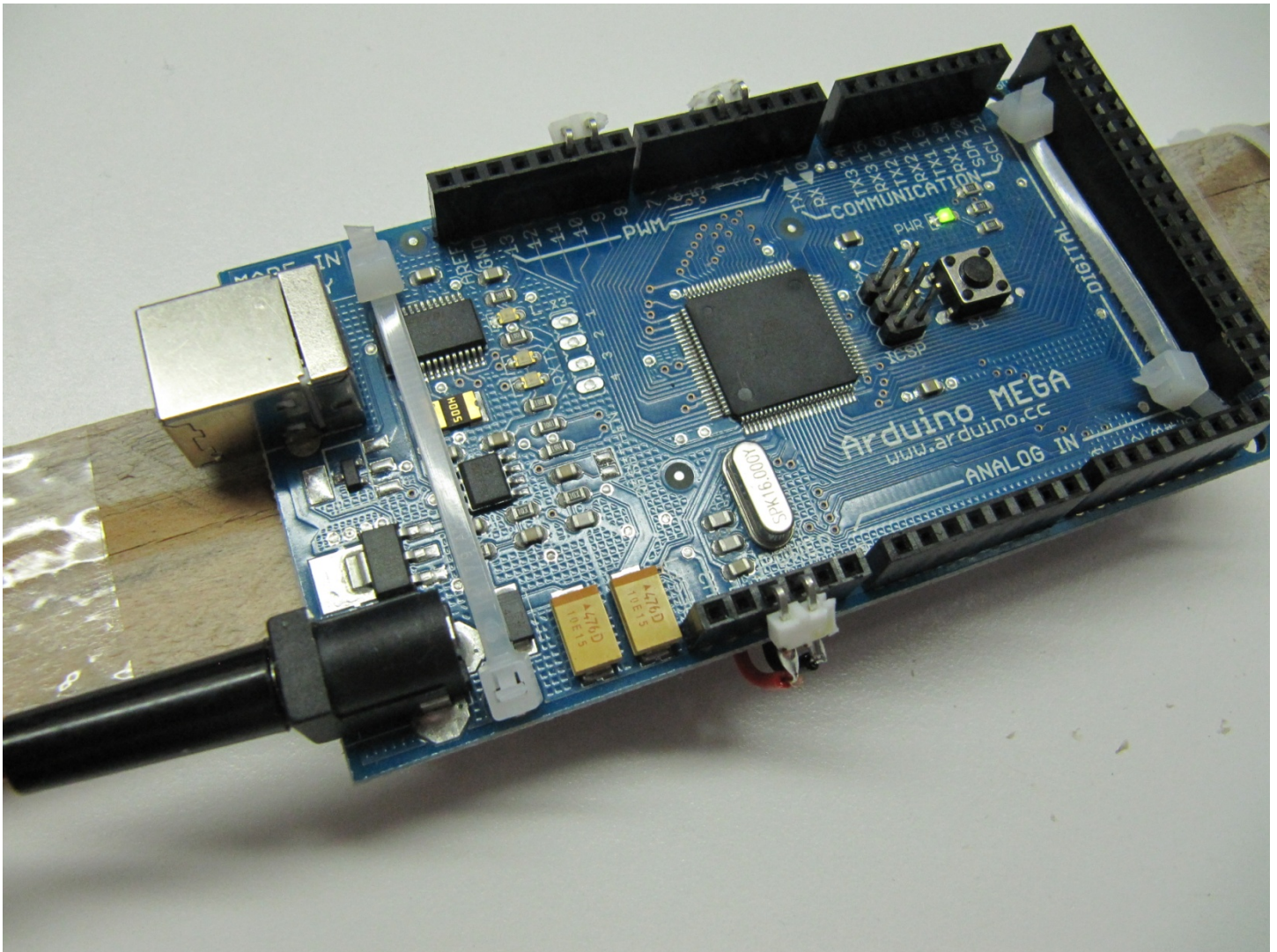
Creating the 16 LED – 6-Program capable Light Strip
**Step 1 – Getting the project box ready**

Cut a piece of wood trim to length for the light strip to be mounted to. I used some trim that was just a little bit wider than the HL1606 Light Strip for my project.

The 16 LED length of the strip is 19 3/4 inches long. You can cut the wood a little longer if you wish but do not make it any shorter than this. You can also use any other material to mount the strip to but be aware that if you use metal or some other type of conductive surface, you will need to insulate it from the light strip in some way to keep the connections from shorting out.
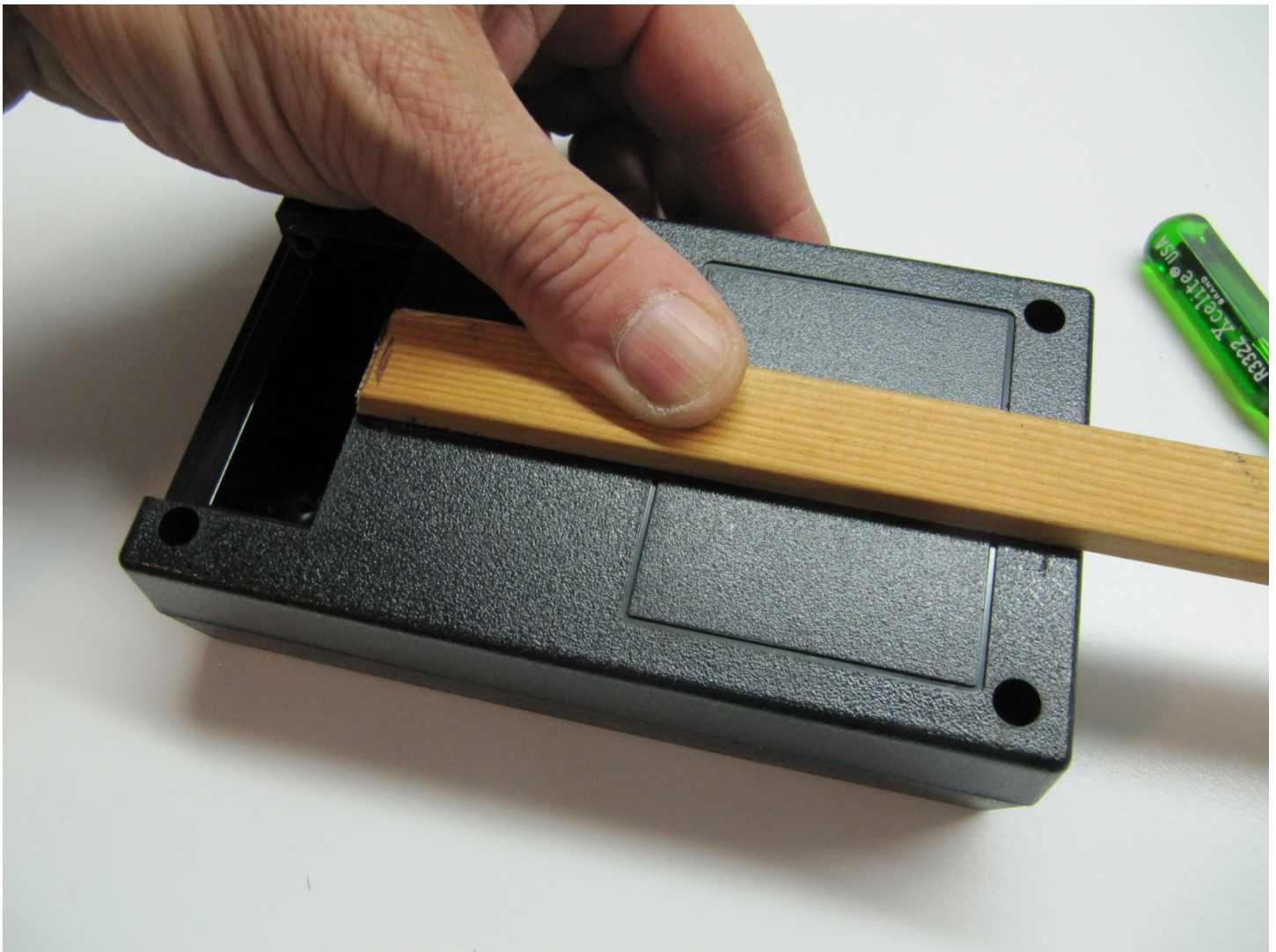
First you will need to mount the wood trim (or whatever you are using to mount the light strip to) to the project box. I am using Wire Ties (Zip Ties) to secure the wood strip to the box so I will need to drill four holes into the project box to route the Wire Ties through. To mark the location of where I will need the holes I centered the wood trim on the project box as shown above and used a small flat bladed screwdriver to score some lines on the box.
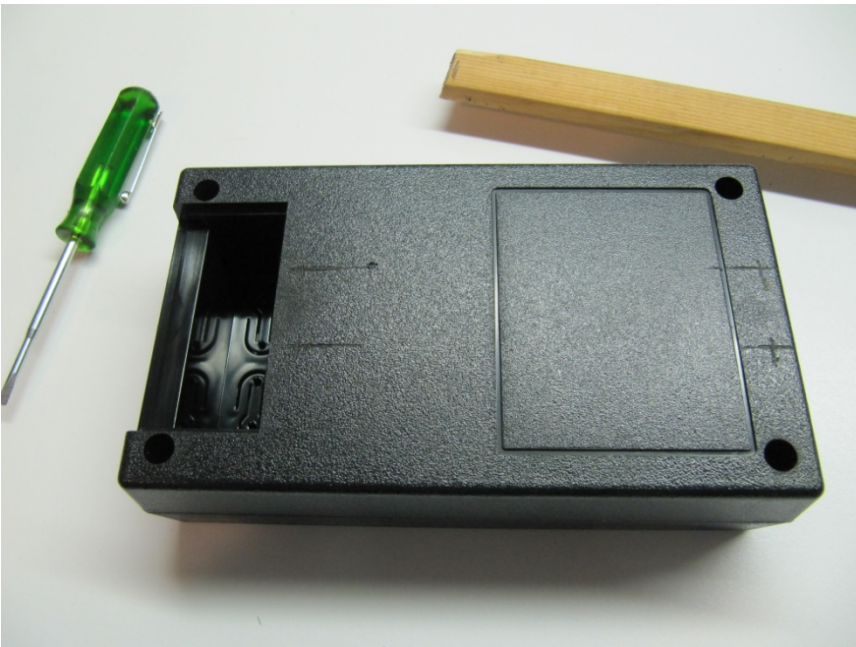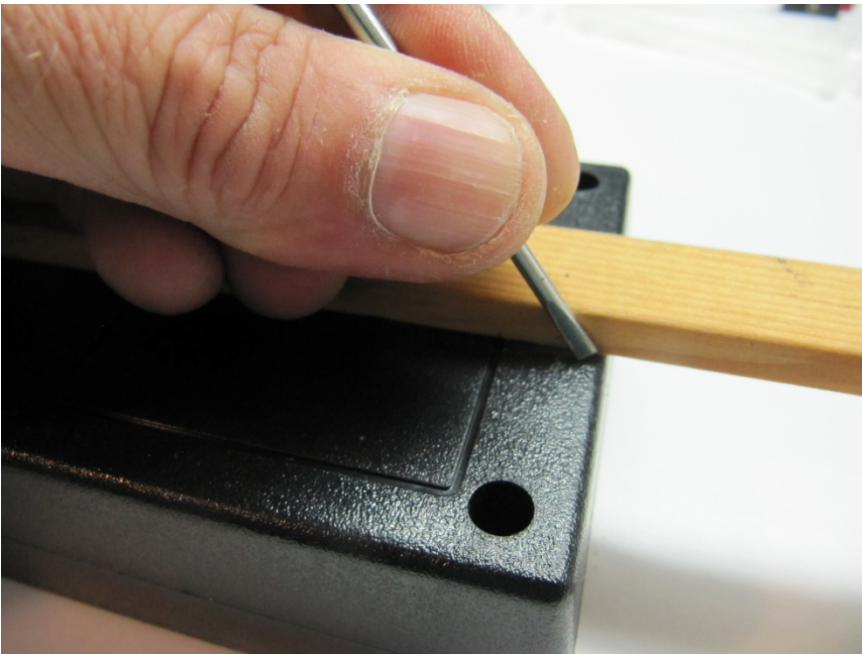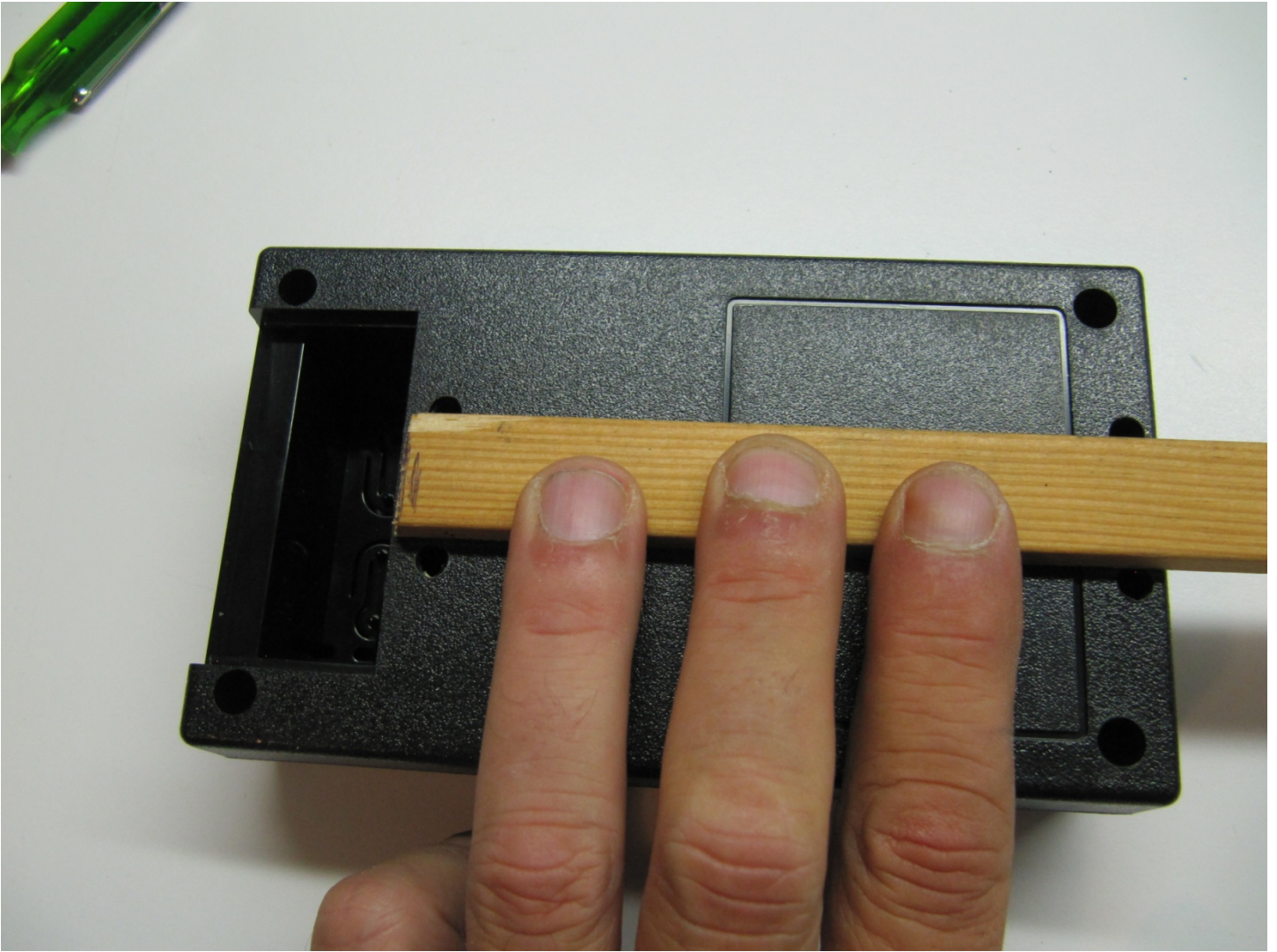
In the photos above, you can see the marks that were scored into the plastic box to mark the location of where the 4 holes will be drilled.
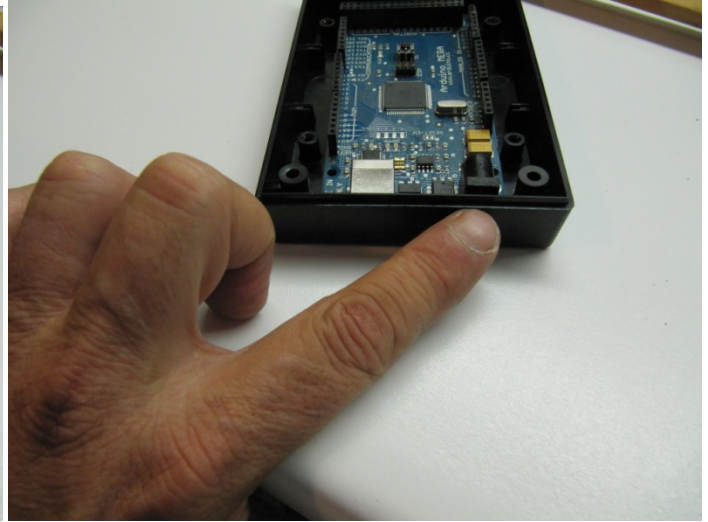
Here you can see that the holes have been drilled into the project box and that they are just outside of the lines that were scored into the box.

How To: Creating a Digital Light Wand for Light Painting Photography

Next we will need to cut the project box so that we will have access to the USB port and the DC power plug on the Arduino board when it is mounted inside the box.

Place the Arduino board inside the project box and score the box where it will need to be cut for access to these ports and cut the holes using a dremel tool or any other method that you may prefer to use for this.
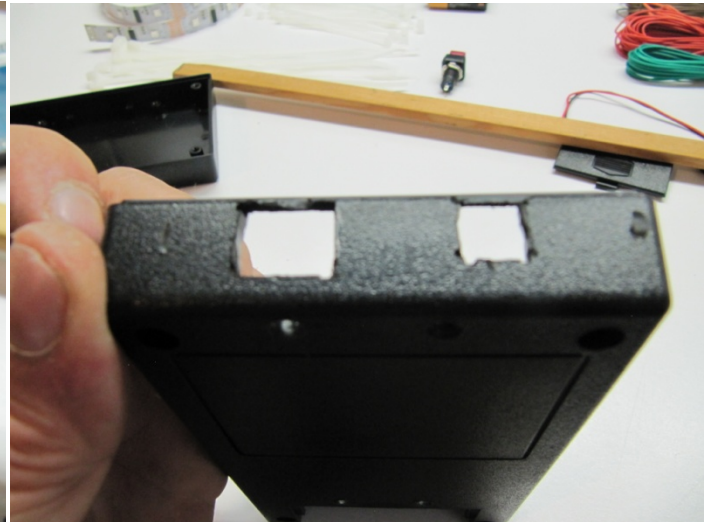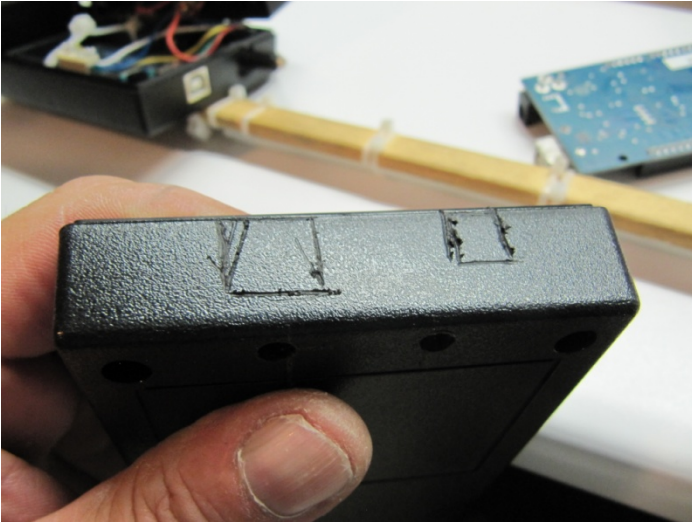
How To: Creating a Digital Light Wand for Light Painting Photography

After cutting the holes, place the Arduino board inside the box to test the fit and make sure that access to these two plugs is not blocked.
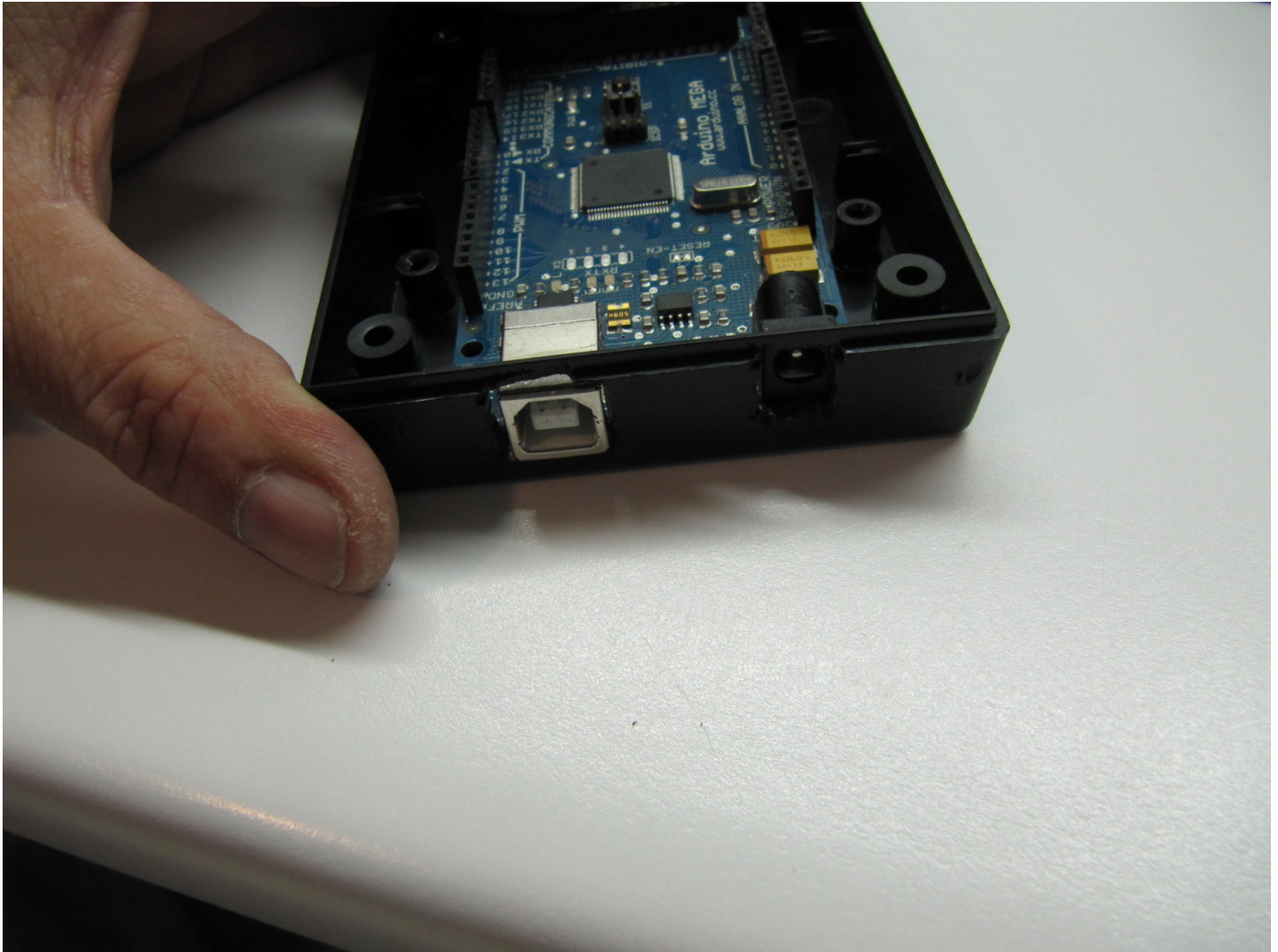
At this point you can secure the wood trim to the project box using wire ties routed through the holes drilled earlier.
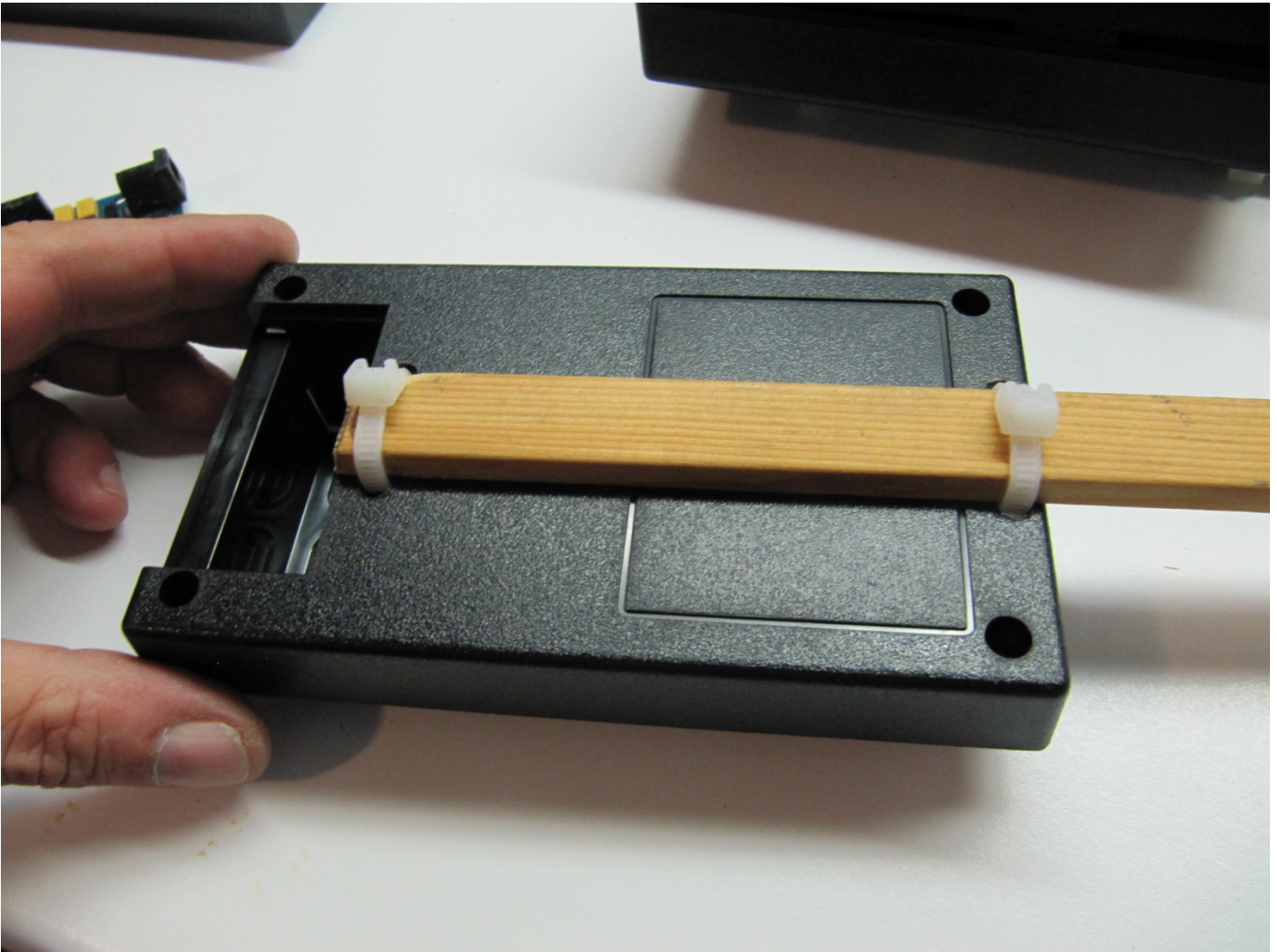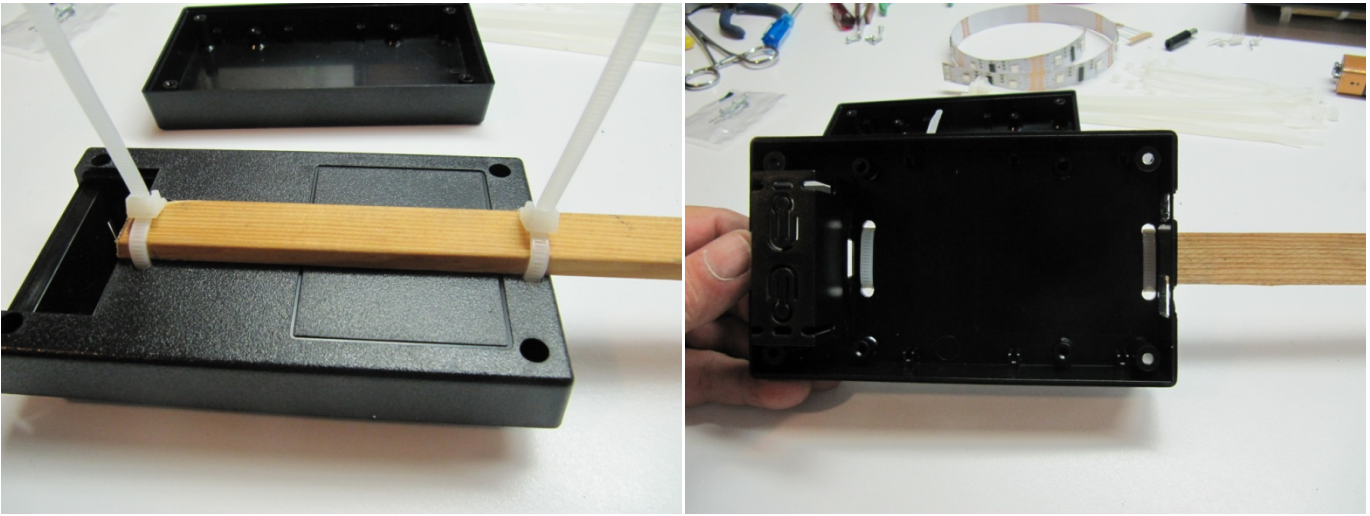
How To: Creating a Digital Light Wand for Light Painting Photography

Now we are ready to mark the box for two more drilled holes to secure the Arduino board inside the box using wire ties. Place the Arduino board in the project box and make sure the USB and power plugs are lined up with the holes that were cut in the last step. Then score the box again on each side of the Arduino board on the end opposite where the USB and power plug are. Then remove the Arduino board and drill the holes.
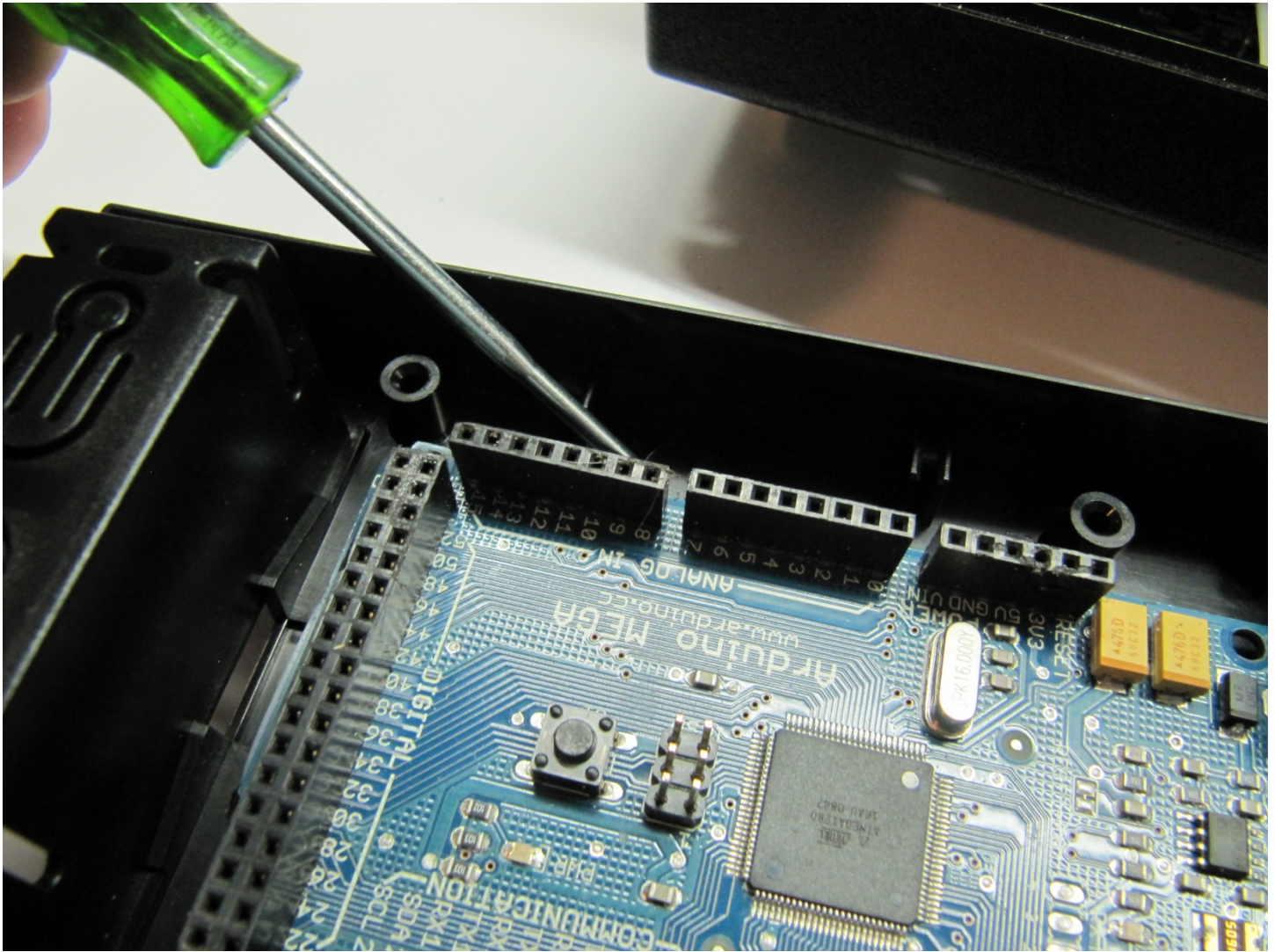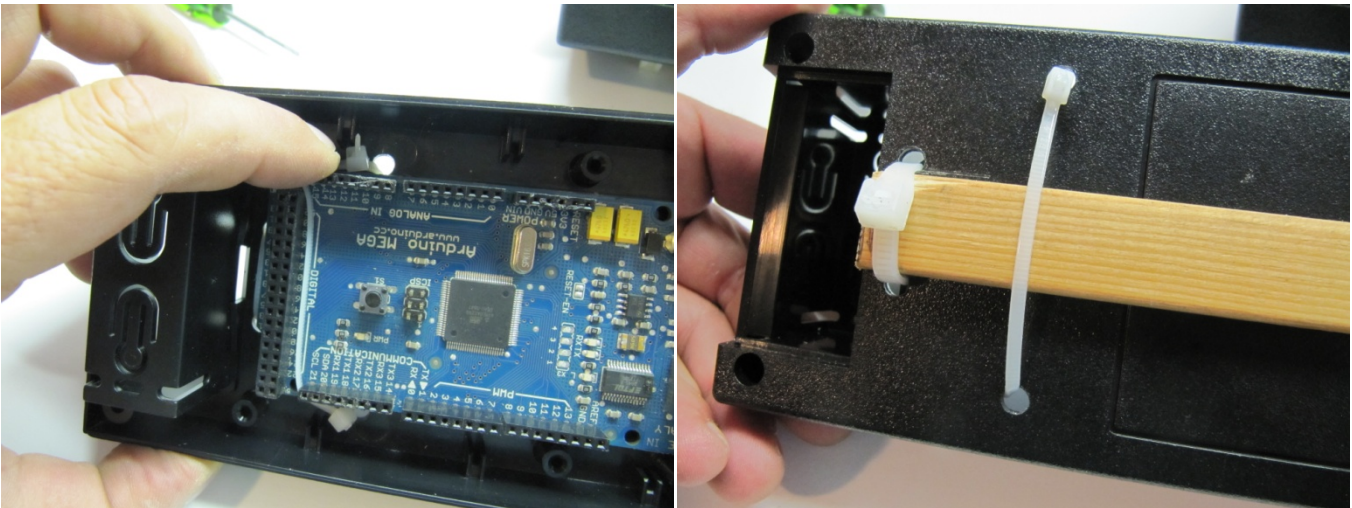
Using small wire ties, mount the Arduino board inside the box and make sure it is secure in its position.

Now we prepare the Rotary Switch by soldering the 1K ohm resistors and wires in place. It helps to have extra hands at this point or at least something that can hold the switch while you are working on it.

Thread the resistors into the holes on the switch as shown. The way this will ultimately work is that as the switch is rotated to the next position, another resistor is added to the chain and acts as a stepped potentiometer and basically a voltage divider circuit. The Arduino board will measure this value using the pin we attach it to in the Analog In section.
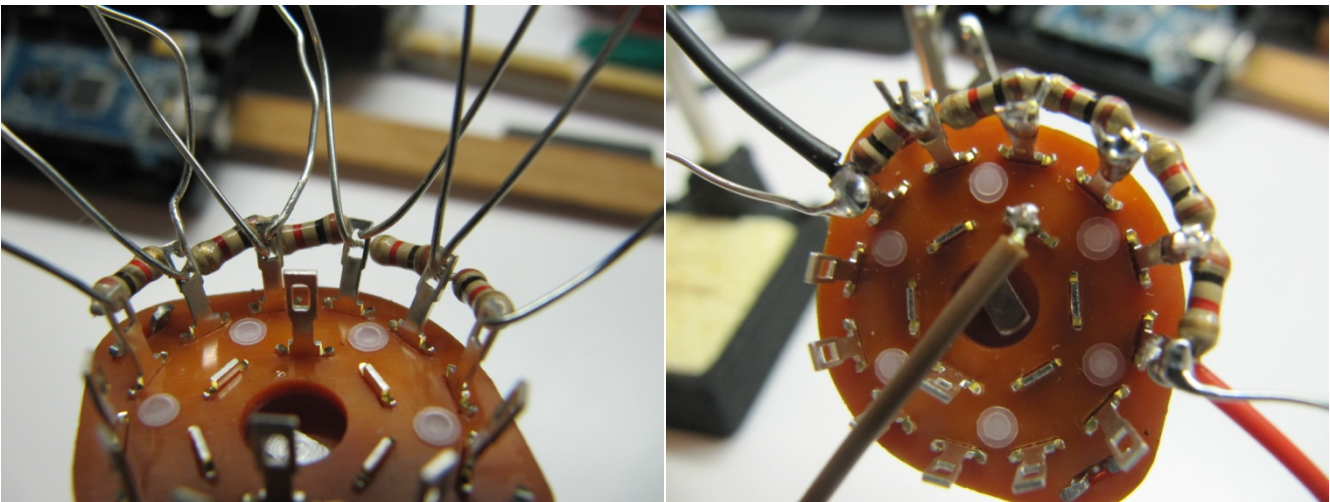
Solder the resistors to the switch and also attach three lengths of wire to the switch as well. I used about an 8 inch length of Red, Black, and Brown 20 gauge wire. The red and black will connect to the power and Ground on the Arduino board and the Brown will connect to the Analog In section of the Arduino board. These connections will be shown a little later in this document.

If necessary, cut the shaft of the rotary switch to the length needed depending on the knob you will be using with it as shown.
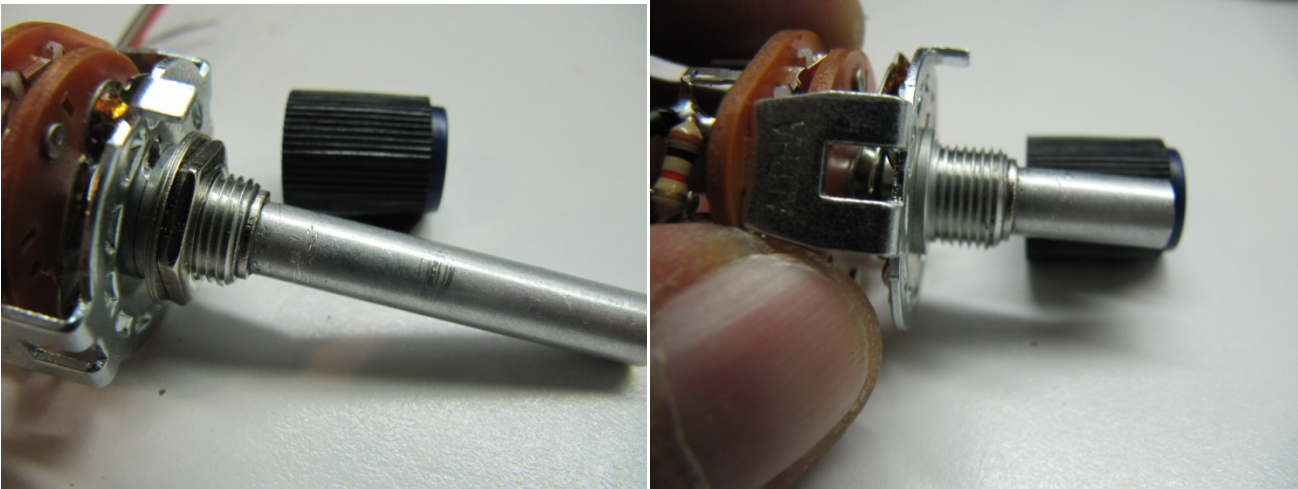
Now we will drill a hole to mount the switch in the top of the project box. Note the tab that is sticking out in the same direction as the shaft in the photos above. When mounting the switch, you will need to cut a small grove that will allow this tab to slip into it. This tab is to help keep the switch from rotating when changing switch positions.   Also, when determining a location to mount the switch, make sure there is adequate clearance inside the box for the switch itself.
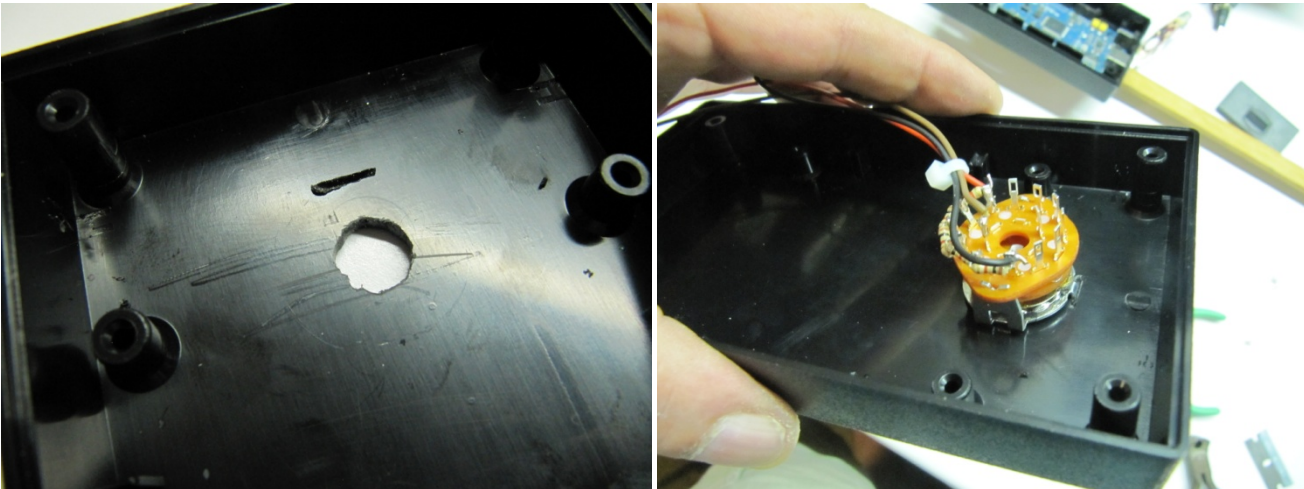
How To: Creating a Digital Light Wand for Light Painting Photography

Now we will drill a hole and mount the momentary push button switch that will supply power to the Arduino board on demand. My preference with this is that the switch is fairly large surface switch with a light spring that can be depressed easily and held down without having to push too hard. I also like to mount these in a place where my thumb will rest on the switch when holding the device in my right hand. Your preferences may vary from this so you may want to change the position or type of switch you are using.
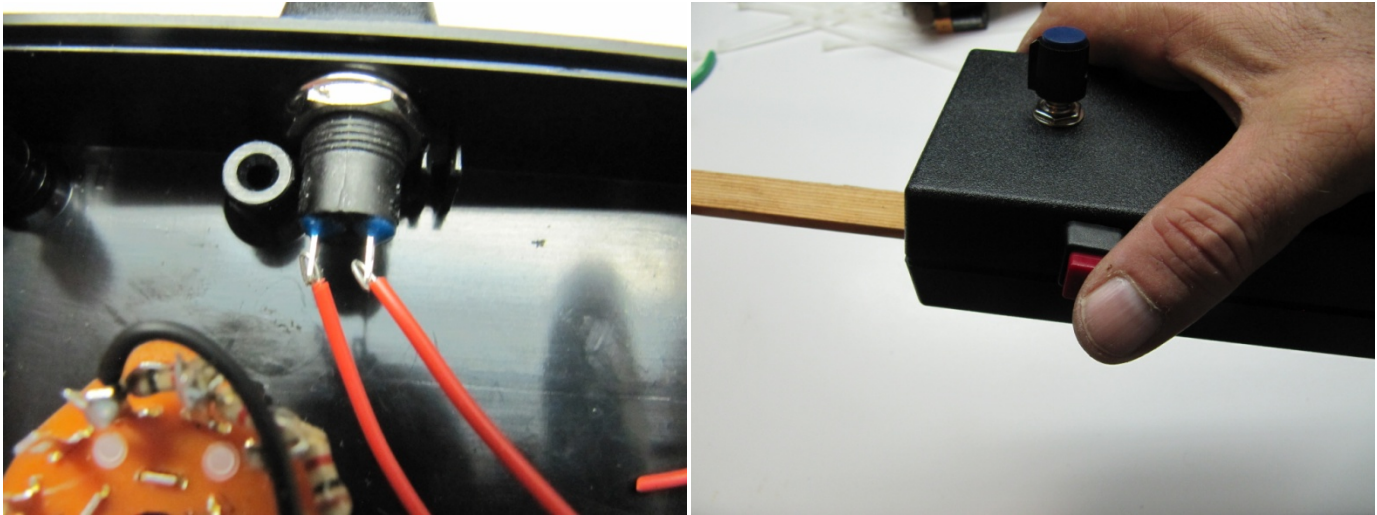
Place the 9 volt battery connector in the project box in the proper place and route the black and red wires into the box. Use a small wire tie as a "strain relief" to keep the wires from being pulled out easily.

Now we will create the power cable assembly. Cut a length of speaker wire or some other dual conductor stranded wire and strip the ends of the wire about 1/8th of an inch as shown.   Unscrew the cover from the N size Coax or Barrel connector and slip it over the wire as shown.
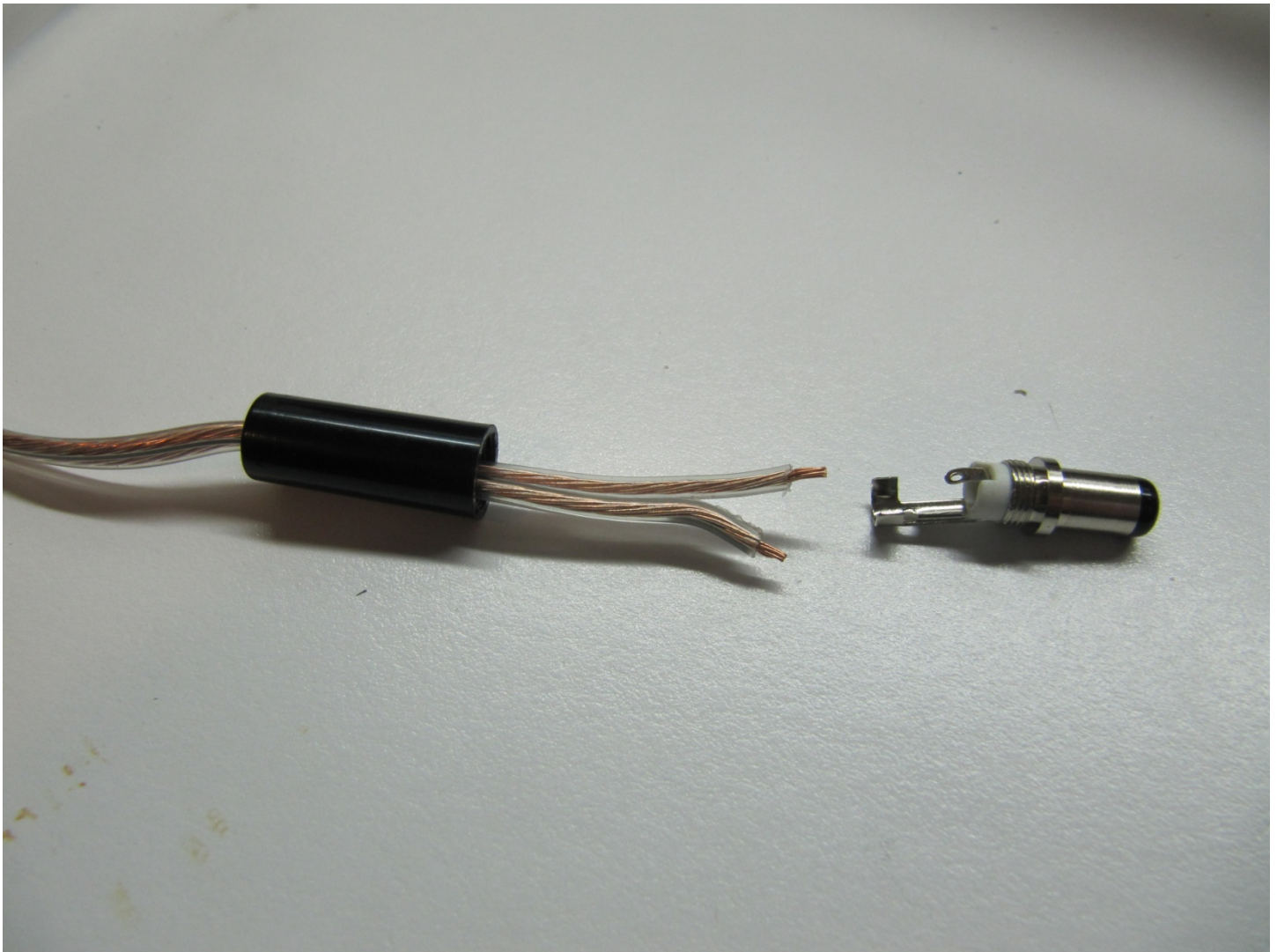
Solder the wires to the connector and make sure you can tell which wire is connected to the center of the plug (short tab) and the outer shell of the plug (long tab). In the photo below you can see that one wire has a black stripe on it. This wire is connected to the out shell of the plug which will need to be the Negative or (-) side of the power. The Center of the plug needs to be connected to the Positive or (+) side of the power which will be supplied through the momentary switch. (See Wiring Diagram for full details)

How To: Creating a Digital Light Wand for Light Painting Photography

The following photo shows how I routed the power wire back into the project box. I cut a slot in the box to allow the wire to route inside the box when the top is fastened to it. I also tied a knot into the wire to act as a strain relief.
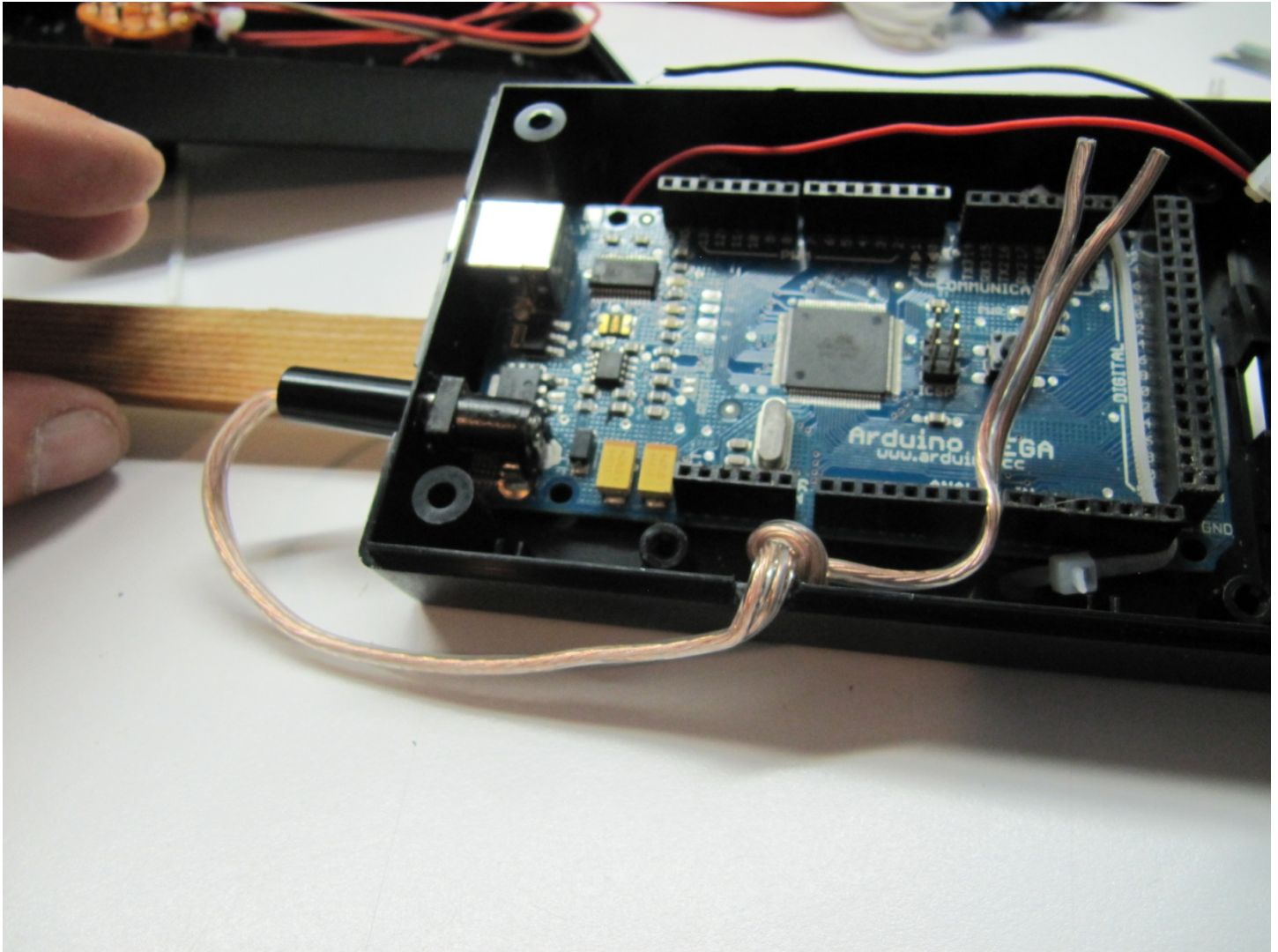
How To: Creating a Digital Light Wand for Light Painting Photography

Next, strip the ends off the power wires and twist the wires together as shown. Solder each connection and then wrap the bare connections with Electrical or Insulation tape to make sure they do not short to anything else or each other.
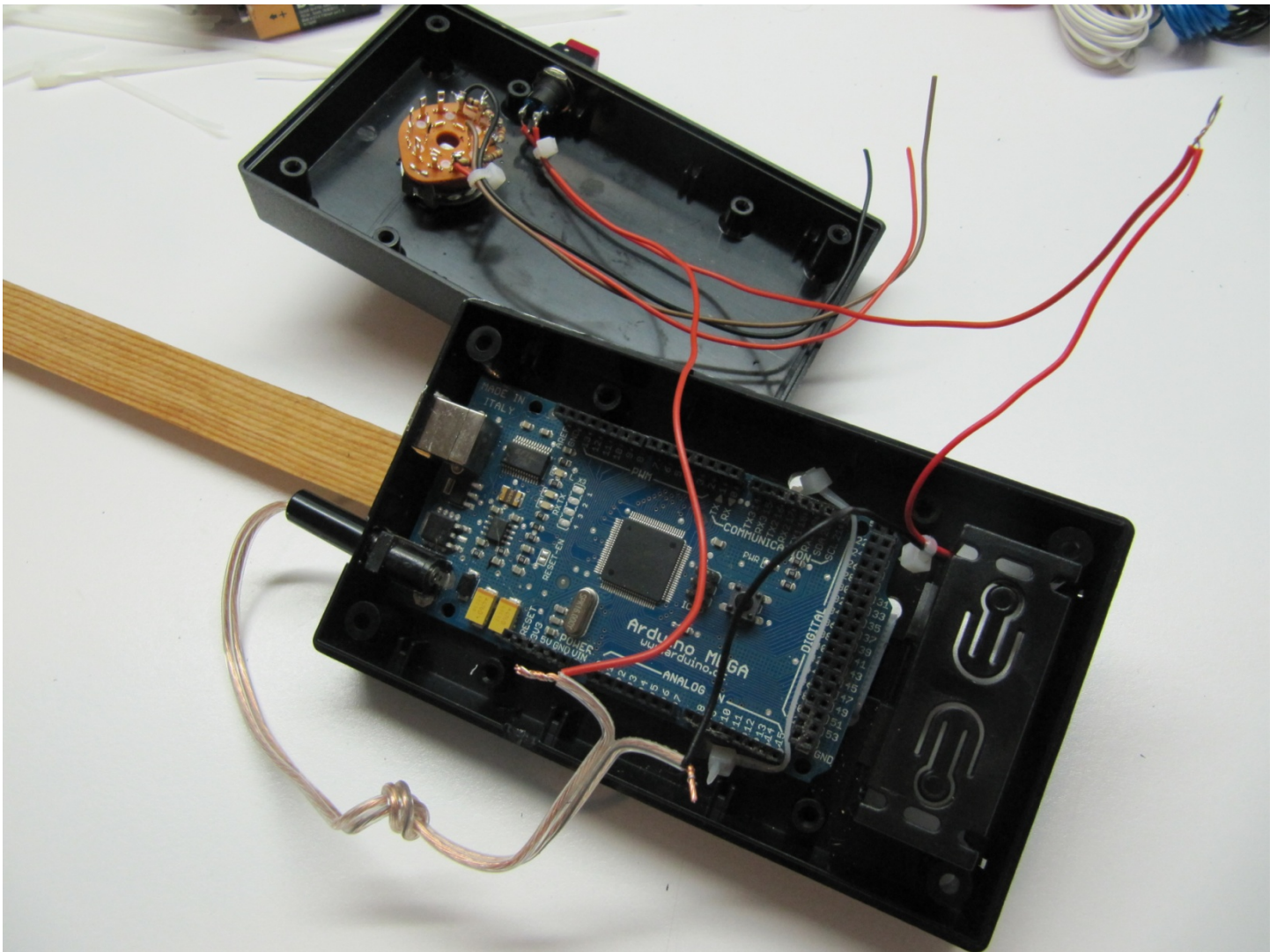
Here is a shot of the wires after soldering and taping them up. In this photo I have also connected the 9 volt battery to the connector and plugged the power connector into the Arduino board. Press the momentary switch to test the circuit. You can see the Green power Light on the Arduino is glowing showing that it is getting power.
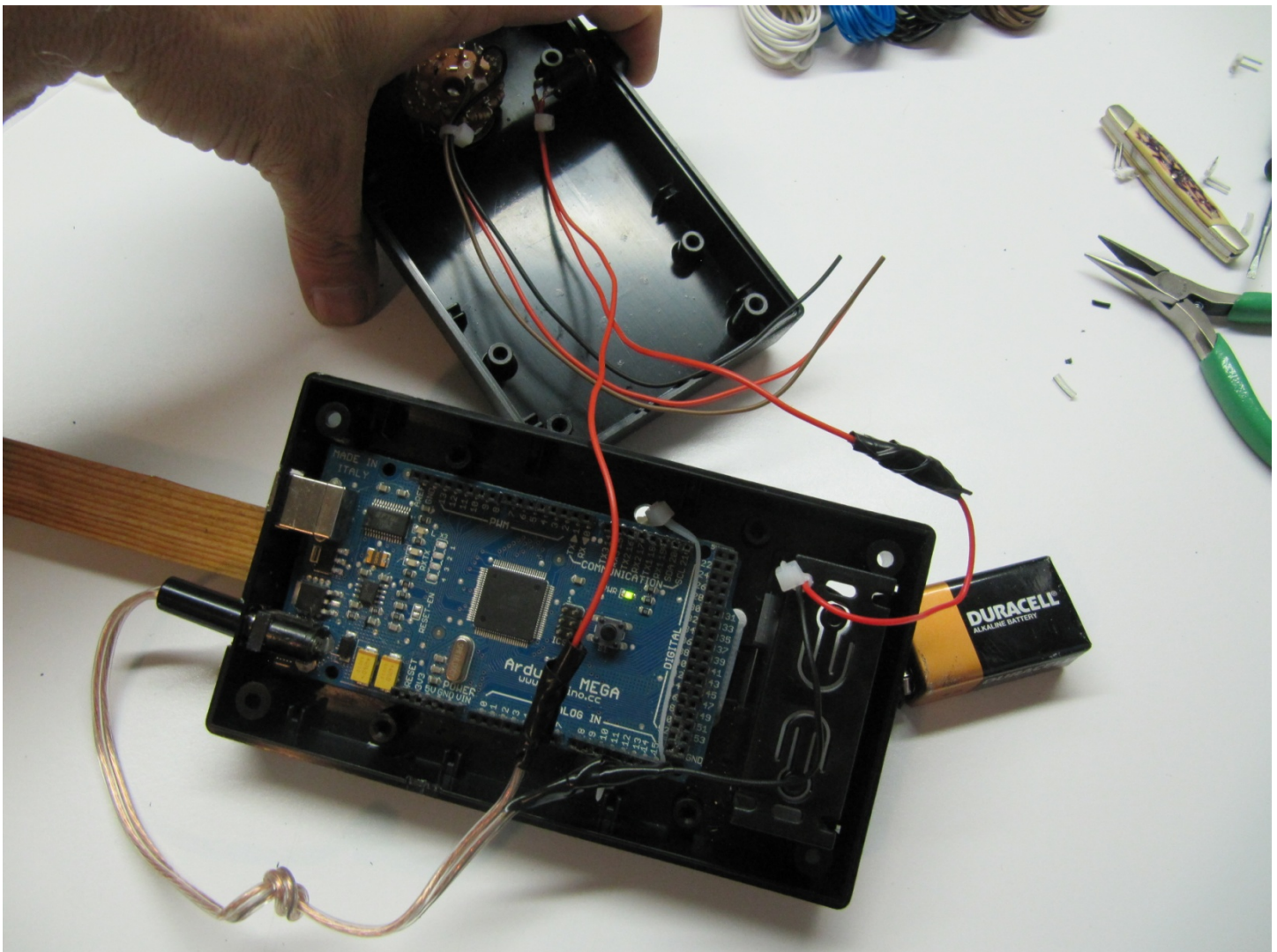
How To: Creating a Digital Light Wand for Light Painting Photography

Okay.. now we will work on mounting the HL1606 LED strip to the assembly and wiring the strip to the Arduino board. I have stated this in other areas of this documentation and it is worth stressing again. THIS IS VERY IMPORTANT!!! Make SURE that you know which end of the Light strip you are working with and will be connecting to the Arduino board. There is an INPUT and an OUTPUT end of the strip. The INPUT end is designated with the S-I, D-I, CK-I, and L-I labels where the OUT PUT end is designated with S-O, D-O, CK-O, and L-O labels. Make sure the INPUT end is mounted to the wood trim closest to the Arduino board / Project Box!
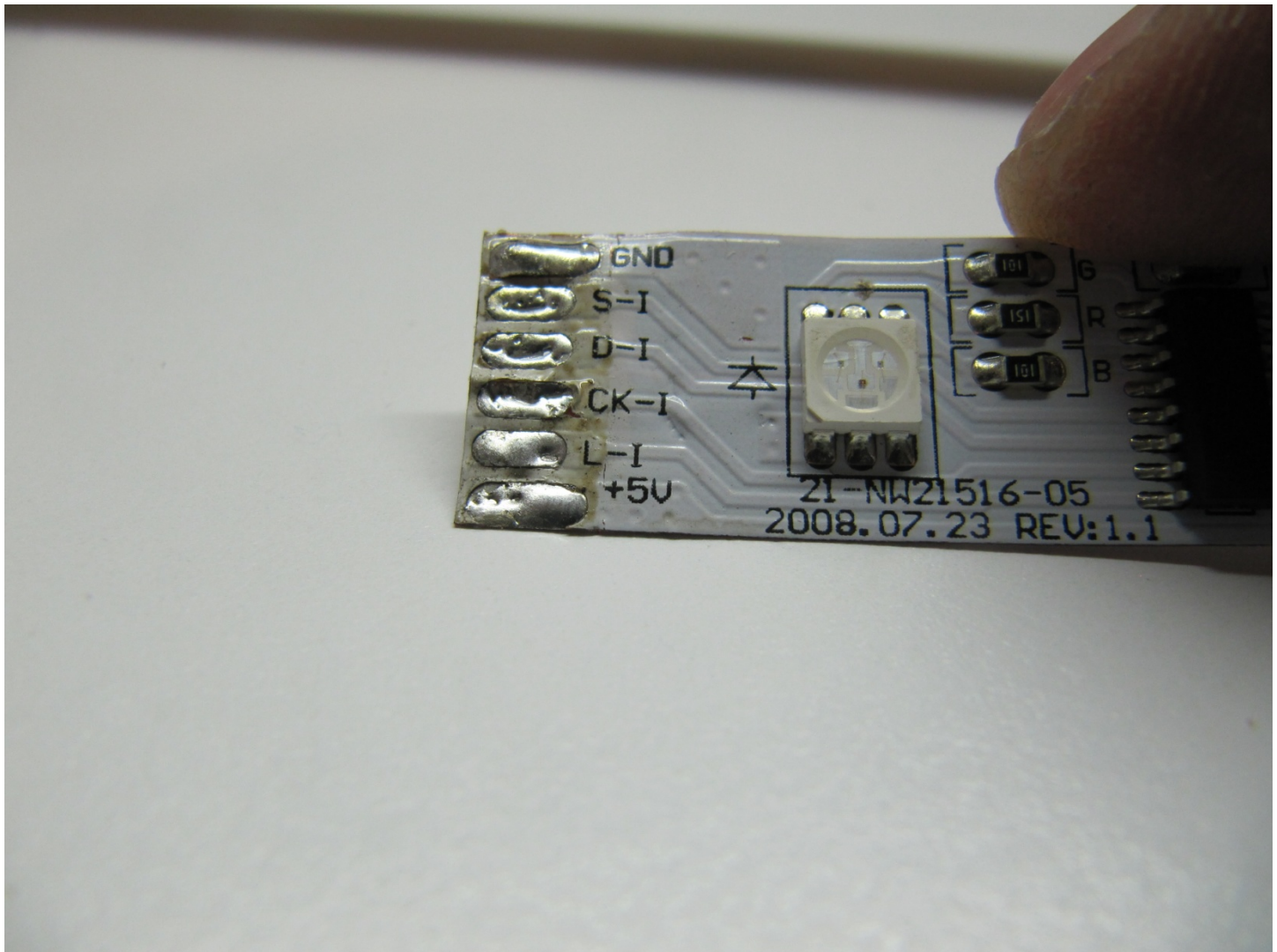
How To: Creating a Digital Light Wand for Light Painting Photography

Mount the Light strip to the wood trim using small wire ties as shown. You can use other methods to do this as well. Just make sure the light strip is secure and will not slide around.
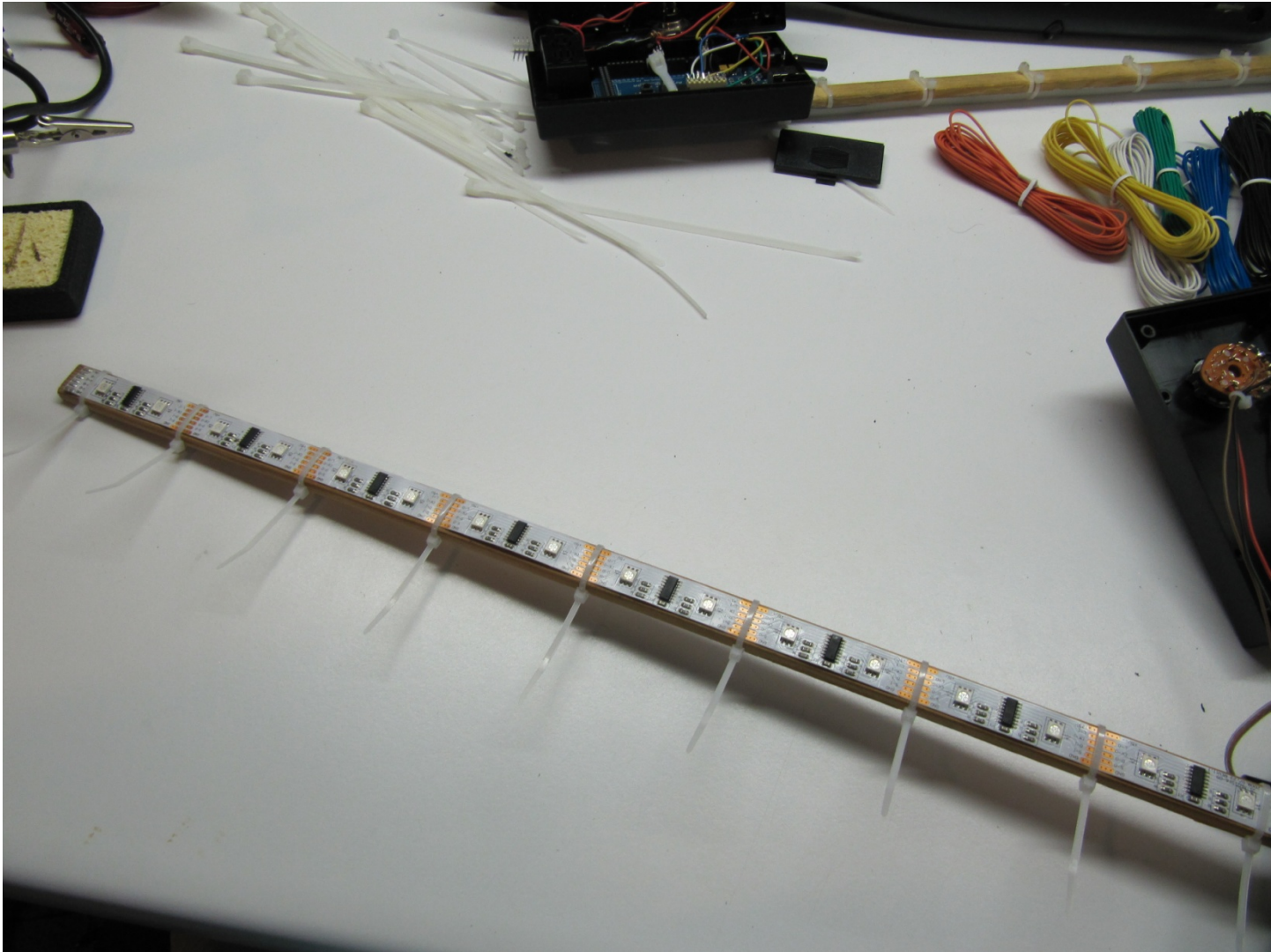
How To: Creating a Digital Light Wand for Light Painting Photography

Now cut 6 wires that will be used to solder to the Light Strip on the INPUT end and will then be routed into the Project box to mount to the Arduino board. I used different color wires to make it easier to keep track of which wire goes where. I also cut them about 10 inches in length to give me extra to work with in case I needed it. Strip the insulation back about 1/4th of an inch on one end of each of the six wires.
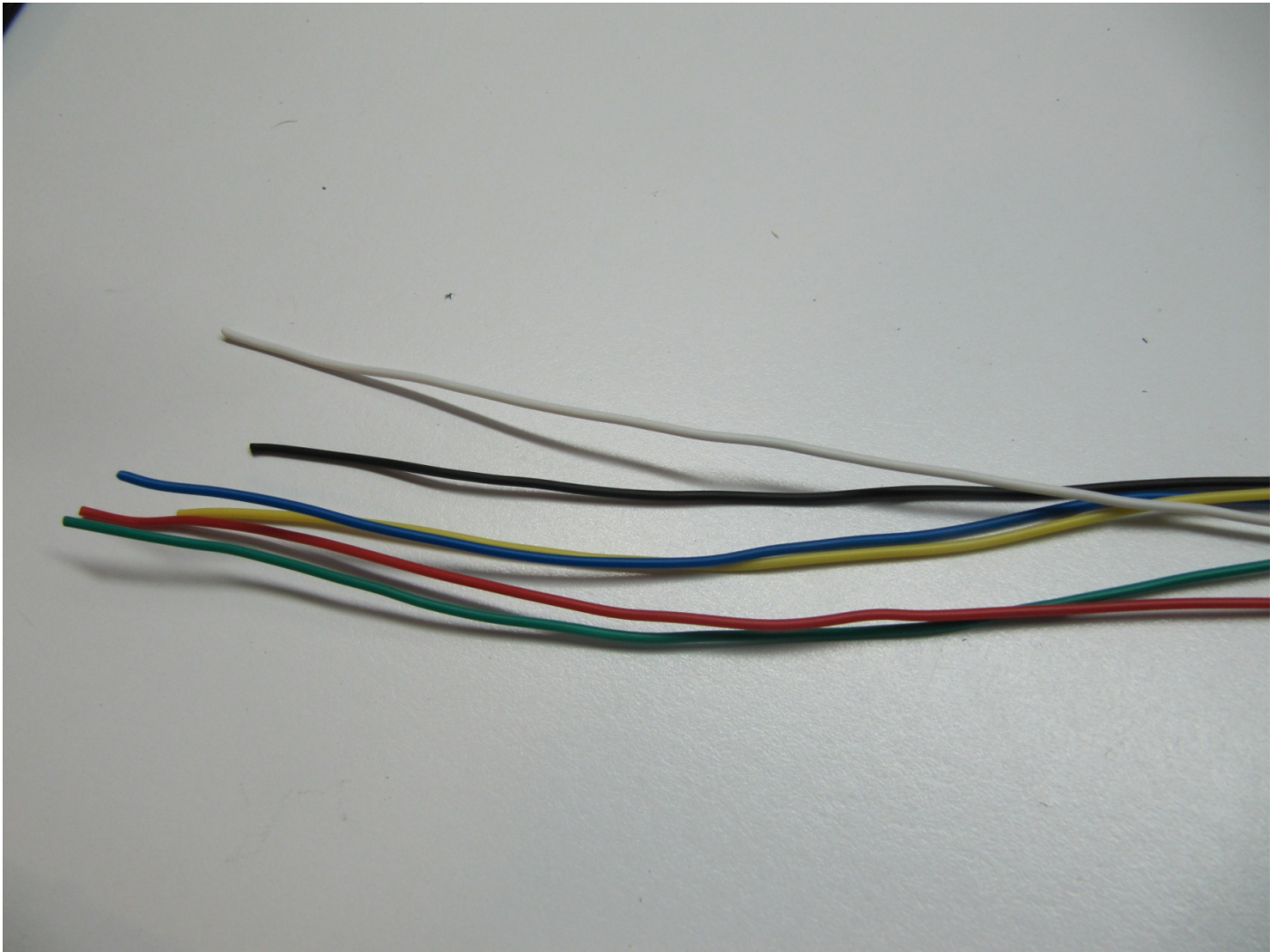
How To: Creating a Digital Light Wand for Light Painting Photography

Now solder each of the wires to the solder pads located on the INPUT end of the light strip. Be careful to work quickly and also not to use too much solder here. The light strip will melt if you apply too much heat to it for too long. Also, if you try to use too much solder, you may end up bridging the connection between the solder pad you are working on and one next to it. If this happens you will need to remove the solder and make sure it is not shorted and try it again. This is where practice in proper soldering techniques will help a LOT!
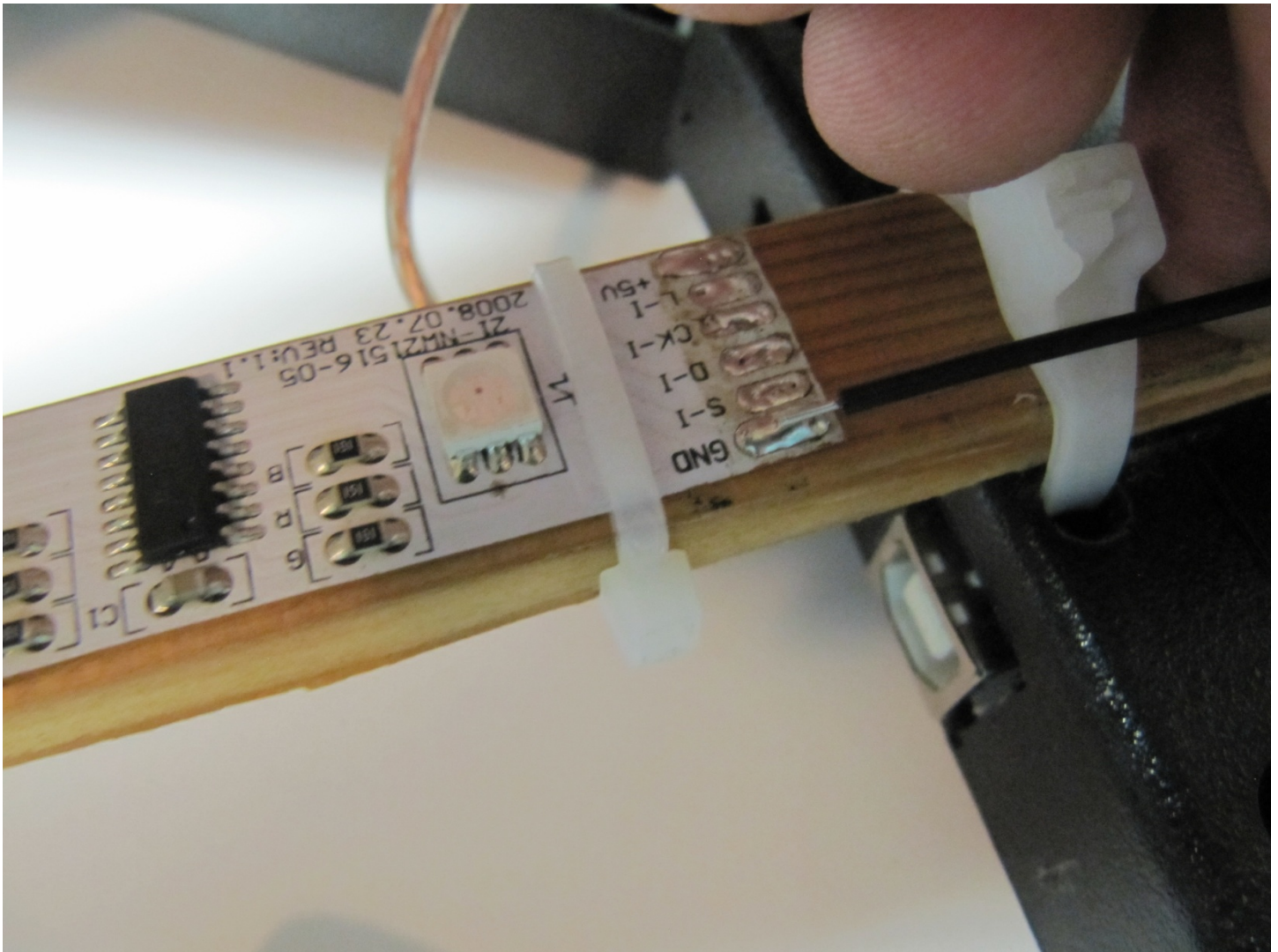
How To: Creating a Digital Light Wand for Light Painting Photography

Continue until you have all six wires soldered to the Light Strip. Note: It is a universal standard that RED wiring goes with a Positive or (+) connection with power and Black wire is commonly used as a Ground or Negative (-) wire. I followed that standard with the connections I used, but it is up to you if you want to do that. You can have all the wires the same color if you want.
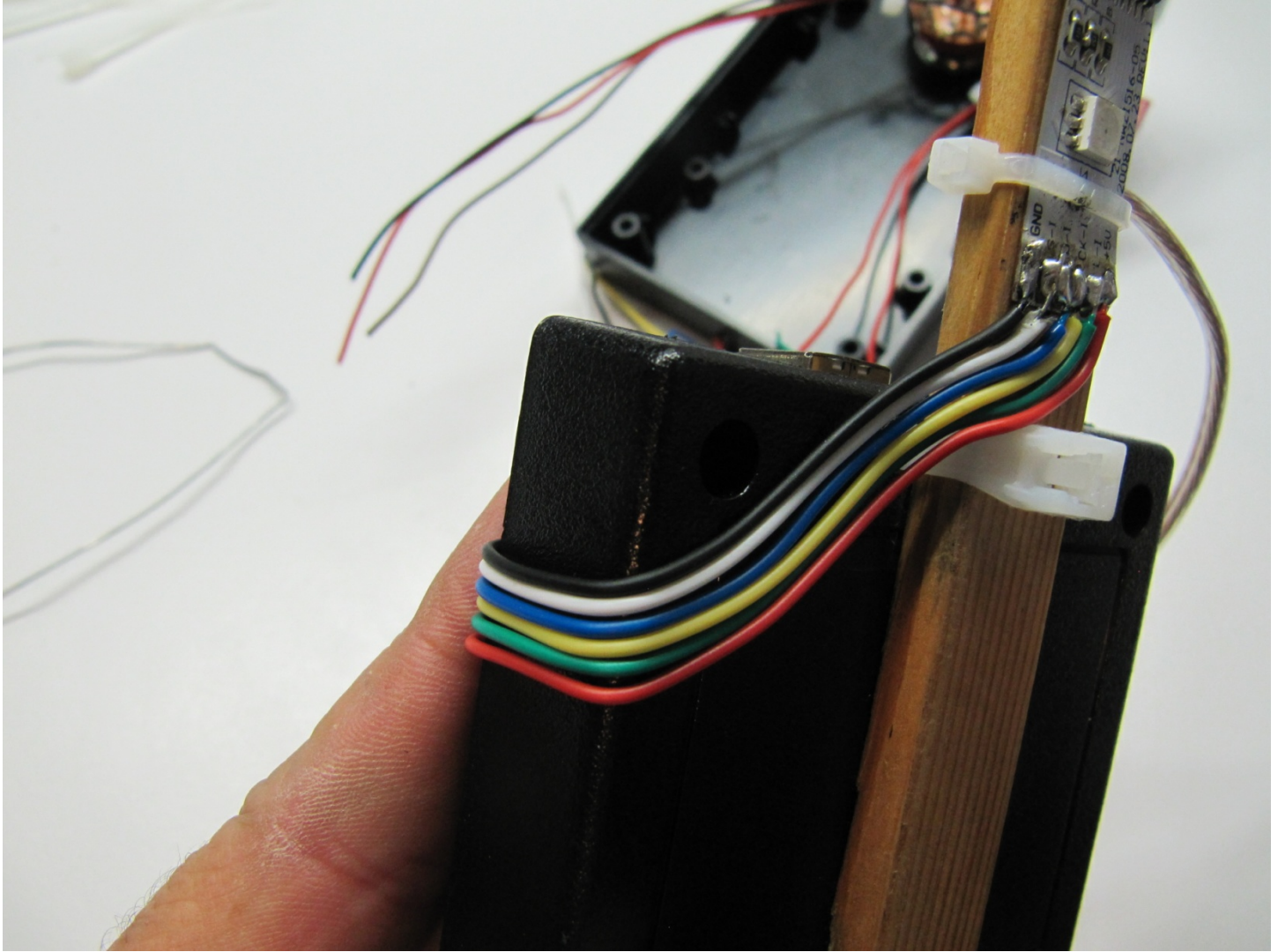
Note how I wrapped the wires around and into the project box. I cut another slot in the project box so that the wires will not be in the way of mounting the top to the box.
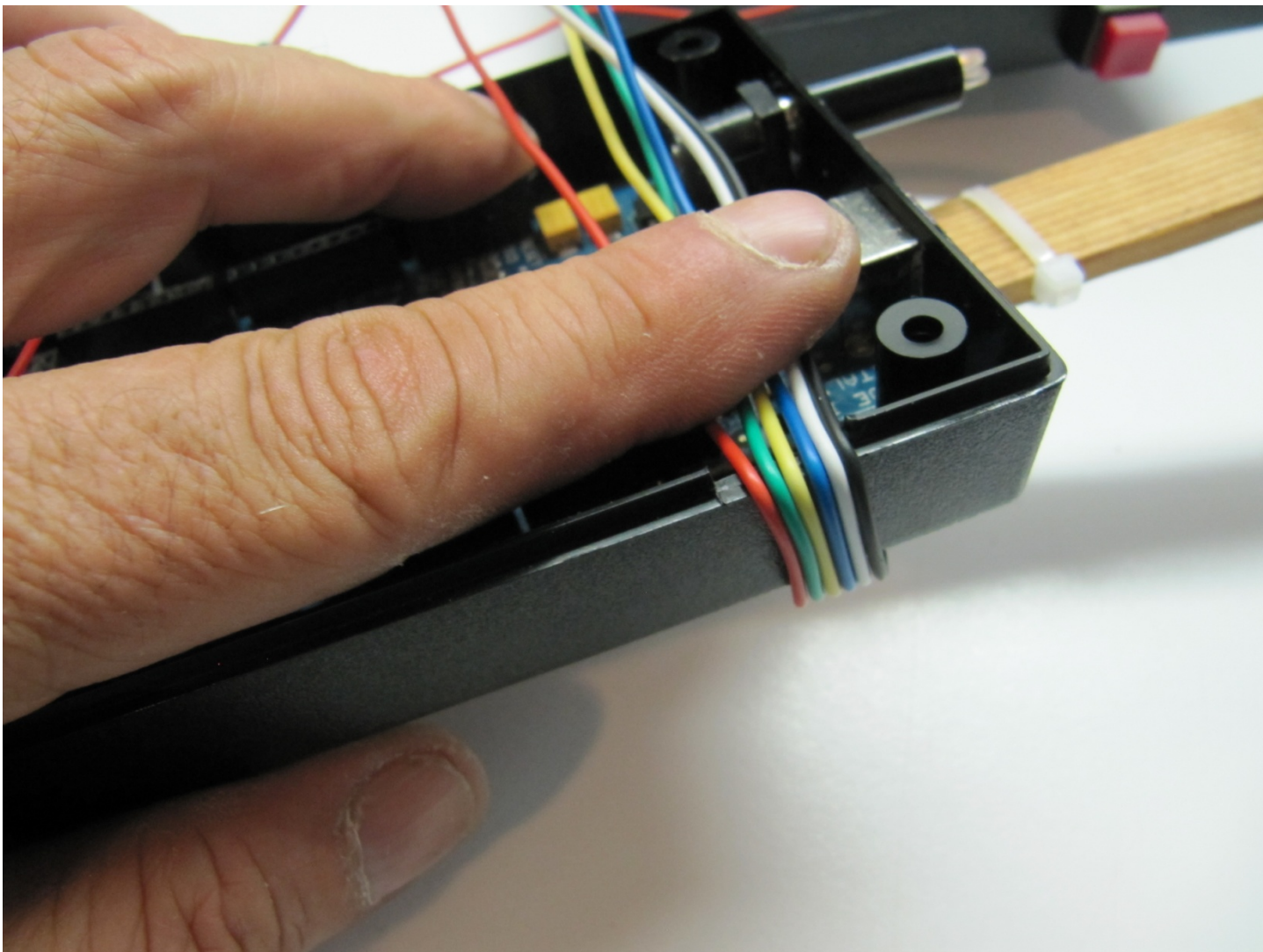
Now route the wires inside the project box to the approximate location they will be when plugged into the Arduino board. Refer to the wiring diagram or the photos as a guide.
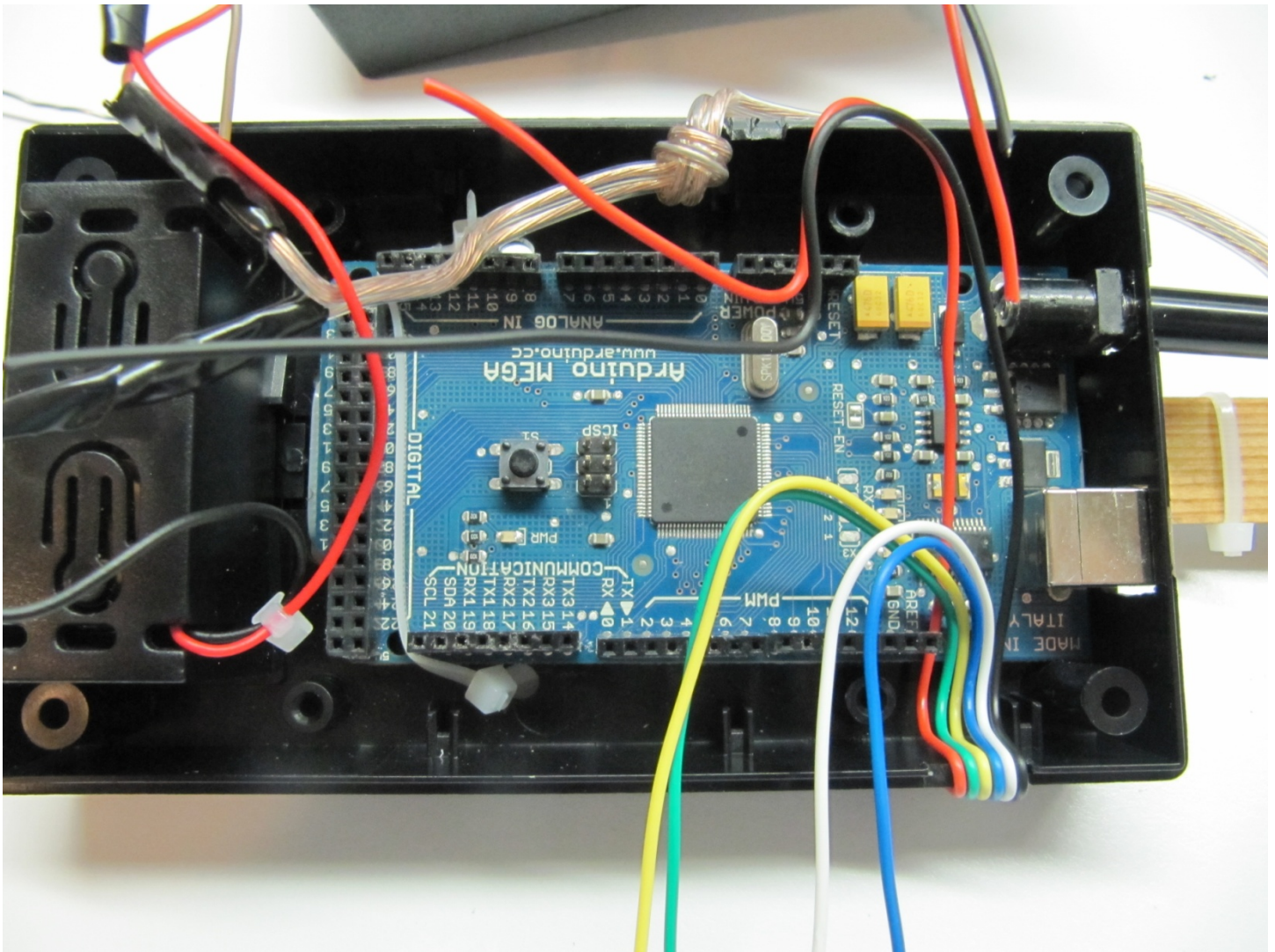
How To: Creating a Digital Light Wand for Light Painting Photography

Next, take each of the connections as they will plug into the Arduino board and solder them to a pin header as shown then plug them into the Arduino board in the proper locations. Refer to the following photo. Note that there are two red wires that need to be soldered and connected to the +5v connection on the Arduino and there are also two black or Ground wires that need to be connected to the GRN connection. These are the Positive and Negative connections from both the Light Strip AND the Rotary switch.
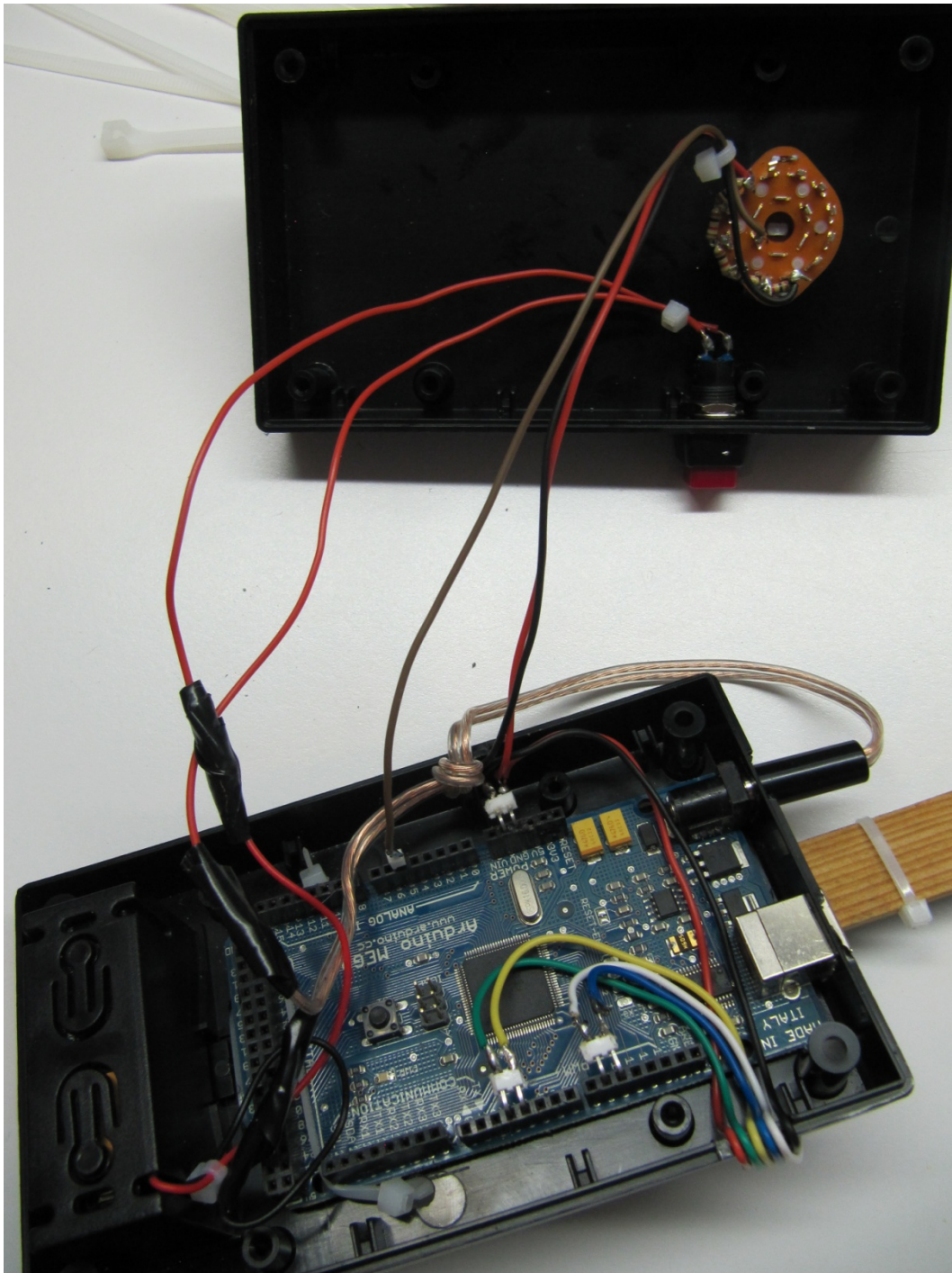
At this point everything should be ready to go!!! Place the top onto the project box and secure it with the screws provided. Then you will be ready to move to the Software section to get it going! ☺

As an option, you can mount some type of diffuser over the light strip to help spread the light a little bit. I used a length of white plastic pipe that I cut down the middle using a scroll saw.
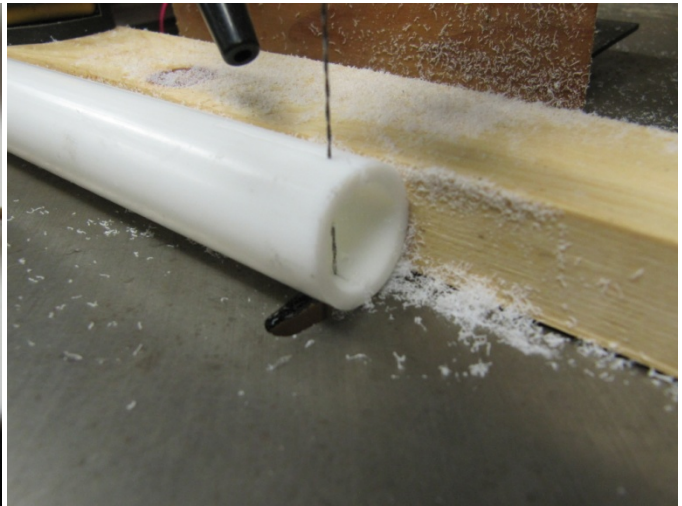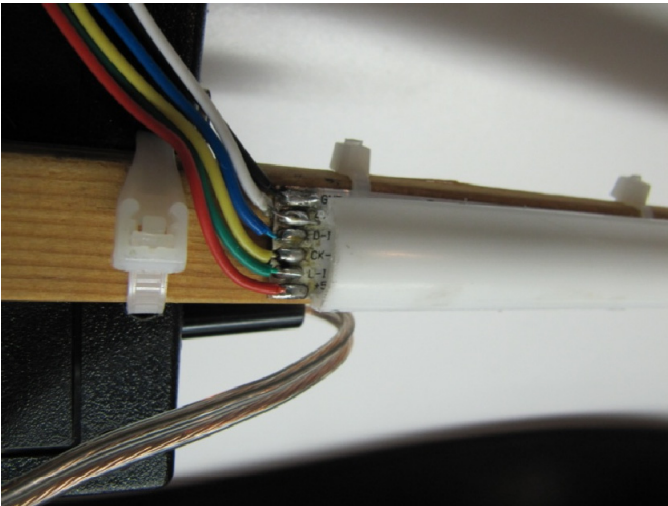
WIRING DIAGRAM

HL1606 RGB LED Addressable Light Strip

Light Strip		Arduino
GND	→	GND
S-I	→	PWM 9
D-I	→	PWM 10
CK-I	→	PWM 4
L-I	→	PWM 3
+5V	→	+5V

GND
S-I
D-I
CK-I
L-I
+5V

HL1606

USB

Arduino Mega

Power

+5V GND

Analog In
5

9 volt
Batt.

Momentary
Push Button
Switch

Rotary Switch
w/ 1 Kohm
Resistors
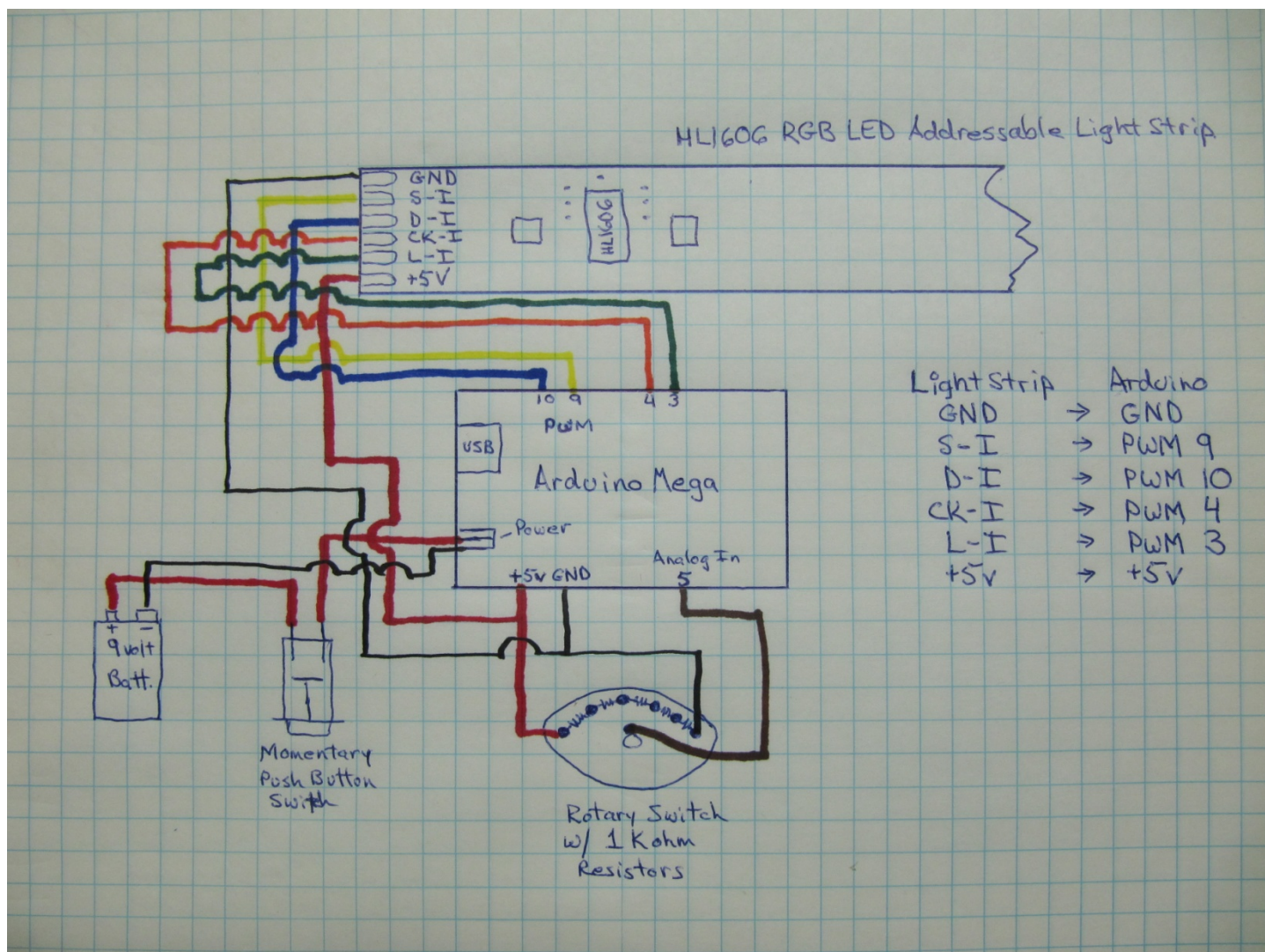
I apologize for the all the wire "hops" ☺

SOFTWARE – Making the Digital Light Wand DO ITS THING!!!

Downloading and installing the Arduino Software

1) Go to the following website:
   http://www.arduino.cc/
2) Click on the download link to take you to the Arduino Download page. Note: You will need to download the version of software that you need for your own operating system. I am detailing these instructions for downloading and installing the Windows version of the software so if you have a different operating system, you will need to modify these instructions accordingly.
3) Click on the link for the latest version of the software (Version 0022 as of the writing of this document) for the appropriate Operating System. Windows in this case.
4) Save the compressed file (zip) to a location on your computer where you will know where to find it.
5) Once the download of the zip file is complete, you will need to extract the zip file to a permanent location on your hard drive. I personally extracted the software to a folder called Arduino-0022 on my C: drive. (C:\Arduino-0022) But you can install it in any location that you want to.

Once the software is extracted, you should have the following files and folders in the location where you extracted the zip file:
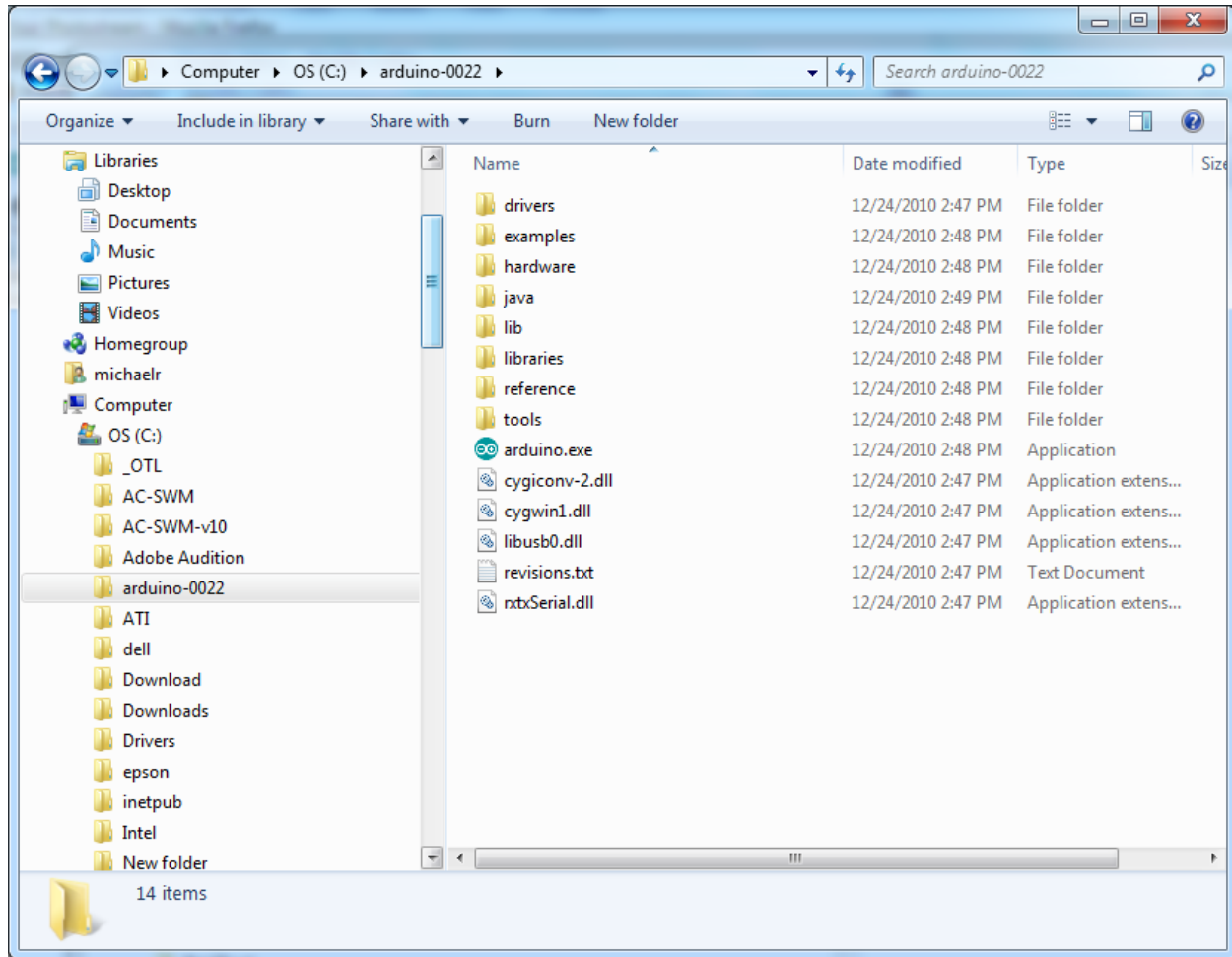
6) If you do not have this file and folder structure, then something has not gone right and you probably need to try to download and extract the Arduino software one more time.
7) If everything looks as it should, you can move to the next step to install the library files for the Light Strip

Downloading and installing the library for the RGB Light Strip

1) Go to the following website:
    http://code.google.com/p/ledstrip/source/browse/#svn%2Ftrunk
2) You will be downloading the three files listed on this page. The filenames are:
    LEDStrip.cpp
    LEDStrip.h
    keywords.txt

3) Click on each of the filenames to display the content of the file in your browser. On the right hand side of the page, there is a link to "View raw file" It is easier to click on this link to get rid of the line formatting on the original display so you can copy just the text of the file easier.

4) Highlight and copy the entire contents of each of the files to the clipboard in windows by highlighting the text then pressing Ctrl-C to copy. Do this for each file one at a time.

5) Open a new Notepad file and paste the contents of the file you copied into the blank Notepad file and save the filename with the same name for each of the three files.

6) Once you have the files saved, you will need to move them to the "Libraries" subfolder under where you extracted the Arduino Software in a new folder called "RGBStrip"

7) So… in Windows Explorer, go to the location where you installed the Arduino software. Under this folder there will be a folder called "Libraries". Double click on the Libraries folder to display the contents of that folder.

8) There will be several folders displayed in the Libraries folder. Create a new folder called RGBStrip along with all the other folders that are already there.

9) Then double click on the new RGBStrip folder that you created.

10) Now copy the three library files you downloaded above to this new folder.

11) The new library will now be available to the Arduino software so that you can use the library commands designed for the Light Strip.

12) To verify that the library is available, go back to the location on your hard drive where you extracted the Arduino software and double click on the Arduino.exe file to run the Arduino software.

13) On the main menu, click on SKETCH, then hover your mouse over the IMPORT LIBRARY menu item. You should see the RGBStrip library listed among the other libraries available. As long as it is showing here, then everything is in its proper place! :-)

NOTE: Depending on your computer, the operating system, and what drivers you may or may not have installed, you may have to download and install additional drivers for the USB interface or the Arduino board.   Please refer to the Arduino website if you experience any problems with this. I will be more than happy to help in any way I can if you do experience problems but it is best to start with the Arduino website.

How To: Creating a Digital Light Wand for Light Painting Photography

Getting the Arduino Board ready to download programs to it

When you plug the Arduino board into your computing using the USB cable, your computer should recognize the board and install the drivers for it. When this takes place, a Serial or COM port will be assigned to the board. This port number is important to know in case the Arduino software recognizes more than one COM port. If you are not sure which COM port is assigned, you can always select COM ports in the Arduino software until you find the right port.

To start getting things ready you will need to create a blank Arduino project to work with in the future for each of the programs you will download to the board.   Complete the following steps to create the blank Arduino project for use with the programmable light strip.

1) Open the Arduino software by double clicking on the Arduino.exe program in the folder where you installed the Arduino program files.

How To: Creating a Digital Light Wand for Light Painting Photography

The screen will look as follows:
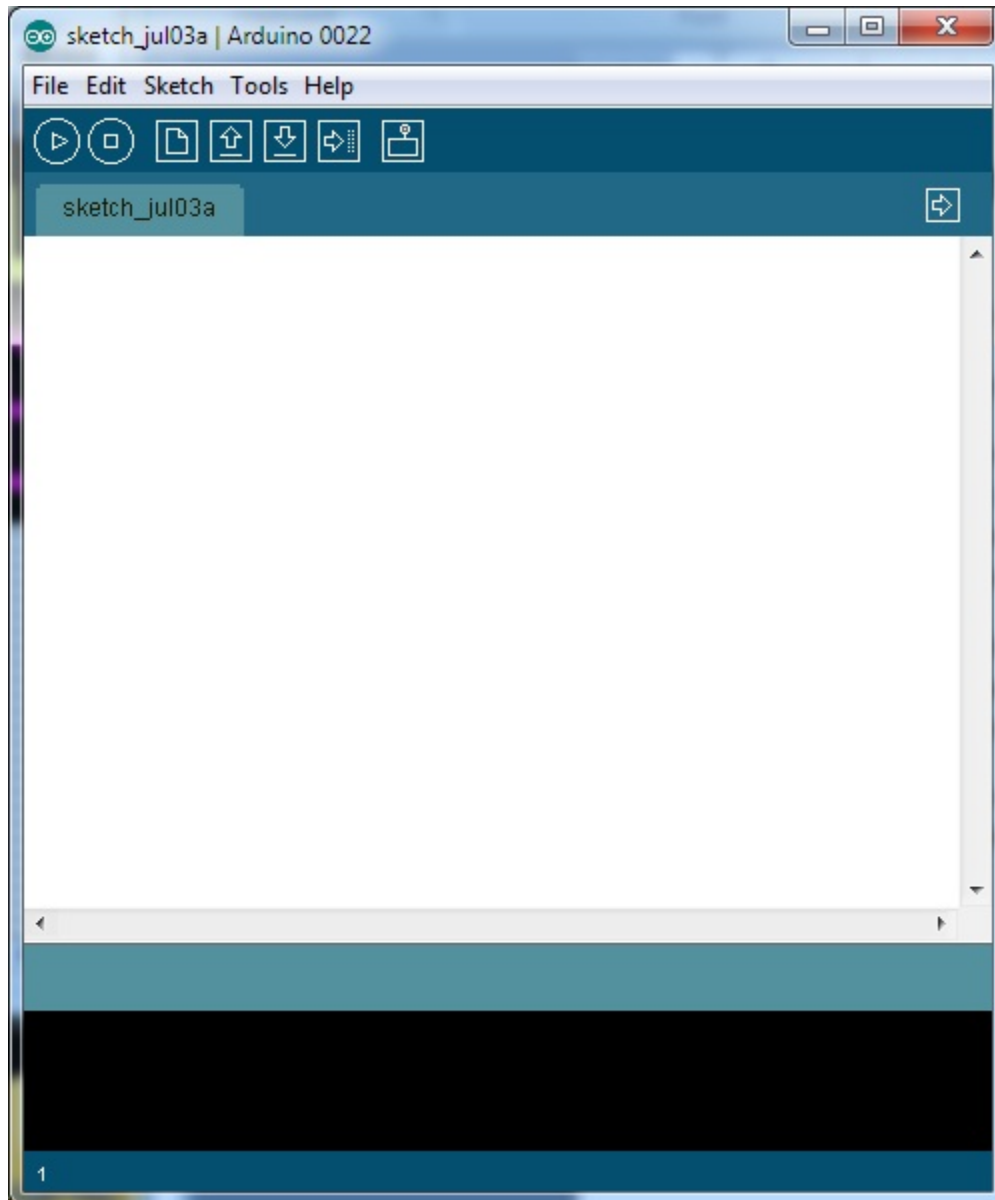
2) Now copy and paste the contents the code below (in between but not including the dashed lines) into the Arduino project window. (Note, the following code is for the 48 LED Light Strip. You can modify this code to work with 16 LED Light Strip by changing the "Void ClearStrip" subroutine area to only include 16 `mystrip.pushCmd(BL);` commands instead of 48 of them.

```
--------------------------------------------------------------------------------------
/* HL1606 LED Strip control of individual LEDs
 *
 * B10000001 = Bright Blue
 * B10000100 = Bright Red
 * B10010000 = Bright Green
 * B10010001 = Bright Cyan
```

```
 * B10000101 = Bright Magenta
 * B10010100 = Bright Yellow
 * B10010101 = Bright White

 * B10000010 = Dim Blue
 * B10001000 = Dim Red
 * B10100000 = Dim Green
 * B10100010 = Dim Cyan
 * B10001010 = Dim Magenta
 * B10101000 = Dim Yellow
 * B10101010 = Dim White

 * B10000000 = Black / Off
*/

#include <LEDStrip.h>

#define SIPIN 9
#define DIPIN 10
#define CLKPIN 4
#define LATCHPIN 3

LEDStrip mystrip(DIPIN,SIPIN,LATCHPIN,CLKPIN);

#define SPULSES 5000
byte BB = B10000001; // Bright Blue
byte BR = B10000100; // Bright Red
byte BG = B10010000; // Bright Green
byte BC = B10010001; // Bright Cyan
byte BM = B10000101; // Bright Magenta
byte BY = B10010100; // Bright Yellow
byte BW = B10010101; // Bright White
byte DB = B10000010; // Dim Blue
byte DR = B10001000; // Dim Red
byte DG = B10100000; // Dim Green
byte DC = B10100010; // Dim Cyan
byte DM = B10001010; // Dim Magenta
byte DY = B10101000; // Dim Yellow
byte DW = B10101010; // Dim White
byte FDBB = B11000011; // Bright Blue
byte FDBR = B11001100; // Bright Red
byte FDBG = B11110000; // Bright Green
byte FDBC = B11110011; // Bright Cyan
byte FDBM = B11001111; // Bright Magenta
byte FDBY = B11111100; // Bright Yellow
byte FDBW = B11111111; // Bright White
byte FUDB = B11000010; // Dim Blue
byte FUDR = B11001000; // Dim Red
byte FUDG = B11100000; // Dim Green
byte FUDC = B11100010; // Dim Cyan
```

```
byte FUDM = B11001010; // Dim Magenta
byte FUDY = B11101000; // Dim Yellow
byte FUDW = B11101010; // Dim White
byte BL = B10000000; // Black / Off

void setup()
{
  ClearStrip(100);
}

void loop()
{

  ClearStrip(5000);
  PatternA(150);

}

void ClearStrip(int duration)
{
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
```

```
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.pushCmd(BL);
      mystrip.latch();
      delay(duration);
}


void PatternA(int duration)
{

/*
 * Paste the copied code from the spreadsheet
 * before the Close Curly Brace symbol below
*/



}


------------------------------------------------------------------------------------------
```

3) After you have copied the code above into the Arduino window, click on FILE, then click on SAVE AS and name the project whatever name you would like to save it as. You will use this BLANK file to paste the code generated from the Excel Spreadsheet into later.
4) Now, we need to verify that you have the correct Arduino board selected and also that the correct COM port is selected to make sure the program will download to the Arduino board.
5) To check these two, click on TOOLS, then hover your mouse over the BOARD menu selection and a list of boards will be displayed as shown below:
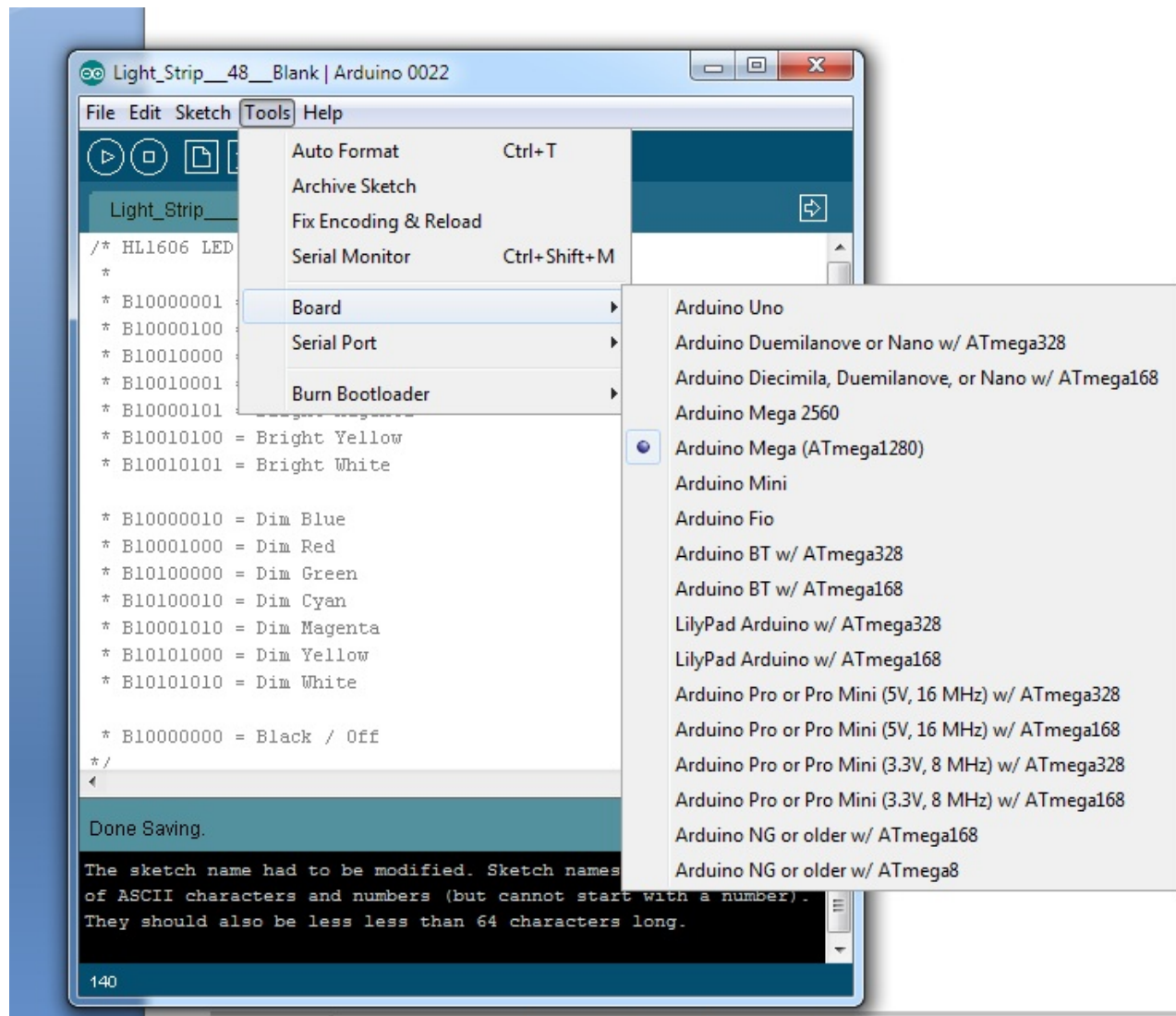
6) Make sure the proper Arduino board is selected for the specific Arduino board you have as shown.

7) Then Click on TOOLS and hover the mouse over the Serial Port menu selection. A list of COM ports will be displayed as shown below:
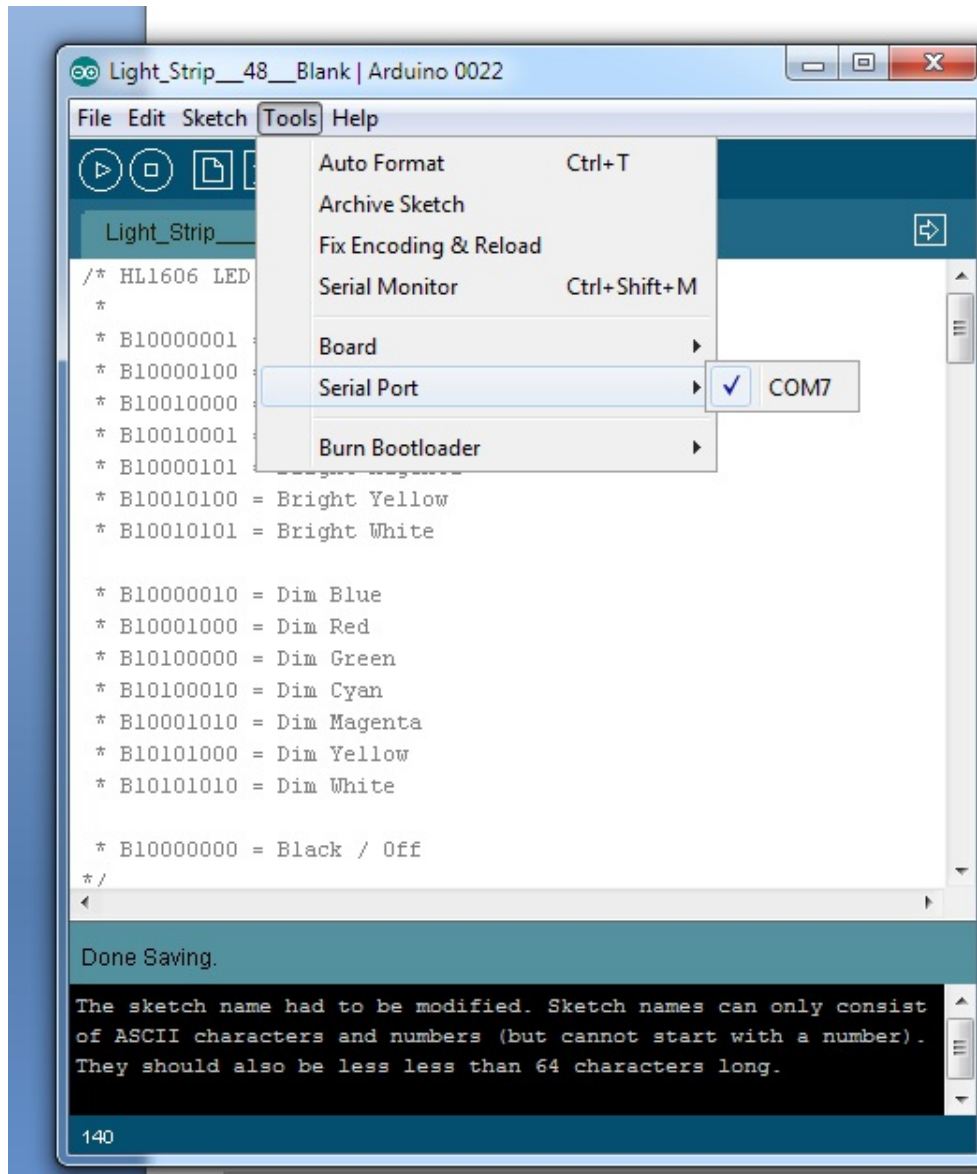
8) Select the COM port in use by the Arduino as shown. If more than one COM port is shown and you are not sure which one is correct, you can select them one at a time and try to download a program to them to see if the Arduino lights will respond to the download. Once you have found the correct COM port, the last one selected will always be the default.

9) The previous steps only need to be followed one time. Once you have created the file you will always have it to open and use in the future. This completes the initial setup of the Arduino board and blank project for the light strip.

How To: Creating a Digital Light Wand for Light Painting Photography

Using the Excel Spreadsheet to "Draw" a picture or pattern

The spreadsheet was made to basically be able to create an image that you want the strip to "Paint" and then translate that to the Arduino code that can then be copied and pasted into the Arduino Interface software and downloaded to the strip. Here is a basically a step by step guide of doing just that.

First of all, make sure you have the blank Arduino project open and then open the spreadsheet. Across the first row at the top of the spreadsheet I have placed a "Pallet" of all of the colors available on the Light strip. These are there for convenience so you can "Copy and Paste" these colors into the cells in the Black area to create an image.

The black area of the spreadsheet is where you will create the image that you want the strip to create when moved in front of a camera. Each row in the black area corresponds to a specific RGB LED on the light Strip. Each column represents a slice of "Time". The time depends on the delay that is put into the code before it is downloaded to the Arduino board. I use a delay of 150 milliseconds and this allows me to walk at a fairly slow pace to keep things steady. You can adjust this faster or slower to suit your own needs. This is covered in more detail in the Arduino code area of this document.

How To: Creating a Digital Light Wand for Light Painting Photography
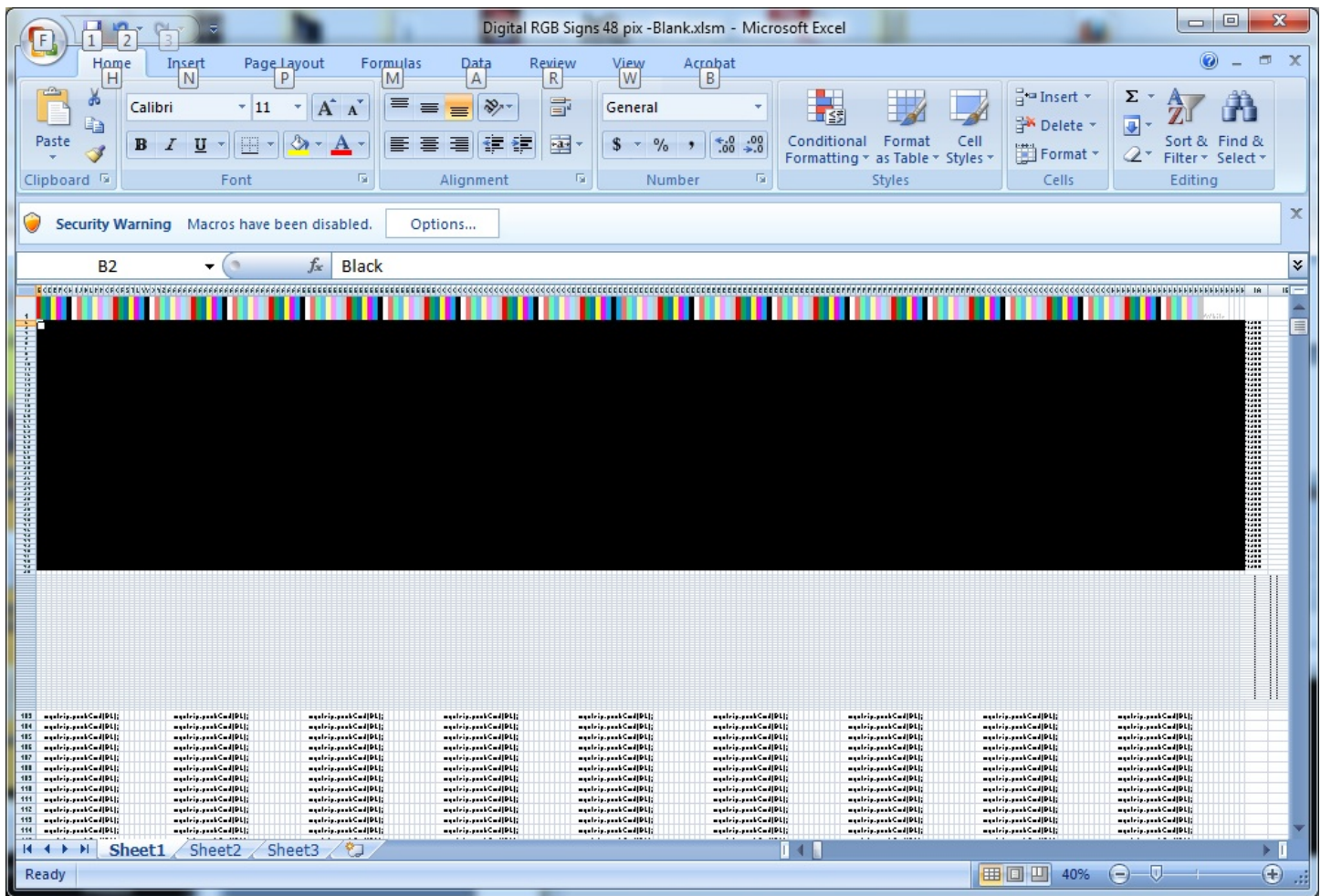
Above is a screen shot of what the Excel Spreadsheet looks like. This particular spreadsheet is for the 48 LED version of the light strip. The top row is the "Color Pallet" area, the next 48 rows (in black) are where you will "Draw" your picture by copying cells from the color pallet area to the black area. The next section contains some formulas for converting the drawing to Arduino code for the Light Strip, and the last section is the actual Arduino code that you will copy and past to the Arduino program for downloading to the Arduino board.

Okay, to create an image, Copy colors from the "Pallet" into the black area using the "Control-C" (Copy) and "Control-V" (Paste) to create the picture that you want the strip to draw. Each line in the Black area represents a single LED on the Light Strip. There is a different spreadsheet for the 16 Light strip and one for the 48 light strip. Each column on the spreadsheet in the black area represents as segment in time. The time is adjusted by changing the delay time in the Arduino program itself. This will depend on your preference for how fast you want to move the strip to create the image when the shutter is open on your camera.

Below the Black area where you create the image, the code is generated that will need to be copied to the Arduino software in order to download the code to the Arduino board. DO NOT make any changes to this area of the spreadsheet unless you know what you are doing with it and you have familiarized yourself with how the spreadsheet works in generating the code and make changes to all of the formulas accordingly. Otherwise the spreadsheet will not

work properly and you will end up with a bunch of code that will either have errors in it or will not run the light strip the way you expect.

Once you have the image created, use the Copy and Paste to copy each column of code that is needed for the image to the Arduino software.

What I usually do is go to the last column in the black area where I did my drawing where the image that I created ends. (The rest of the image area will still be black to the right of this column) and see which column I am on. That way I know which columns of code that need to copy. Each column of code corresponds to each set of columns for the Alphabet. So the first column encompasses column B through Z (A is used for the line numbers), then the second column encompasses BA through BZ, the third column CA through CZ and so on. You only need to copy the columns that include code for your image. (Since the rest will be black (or LED turned off in this case)) ☺
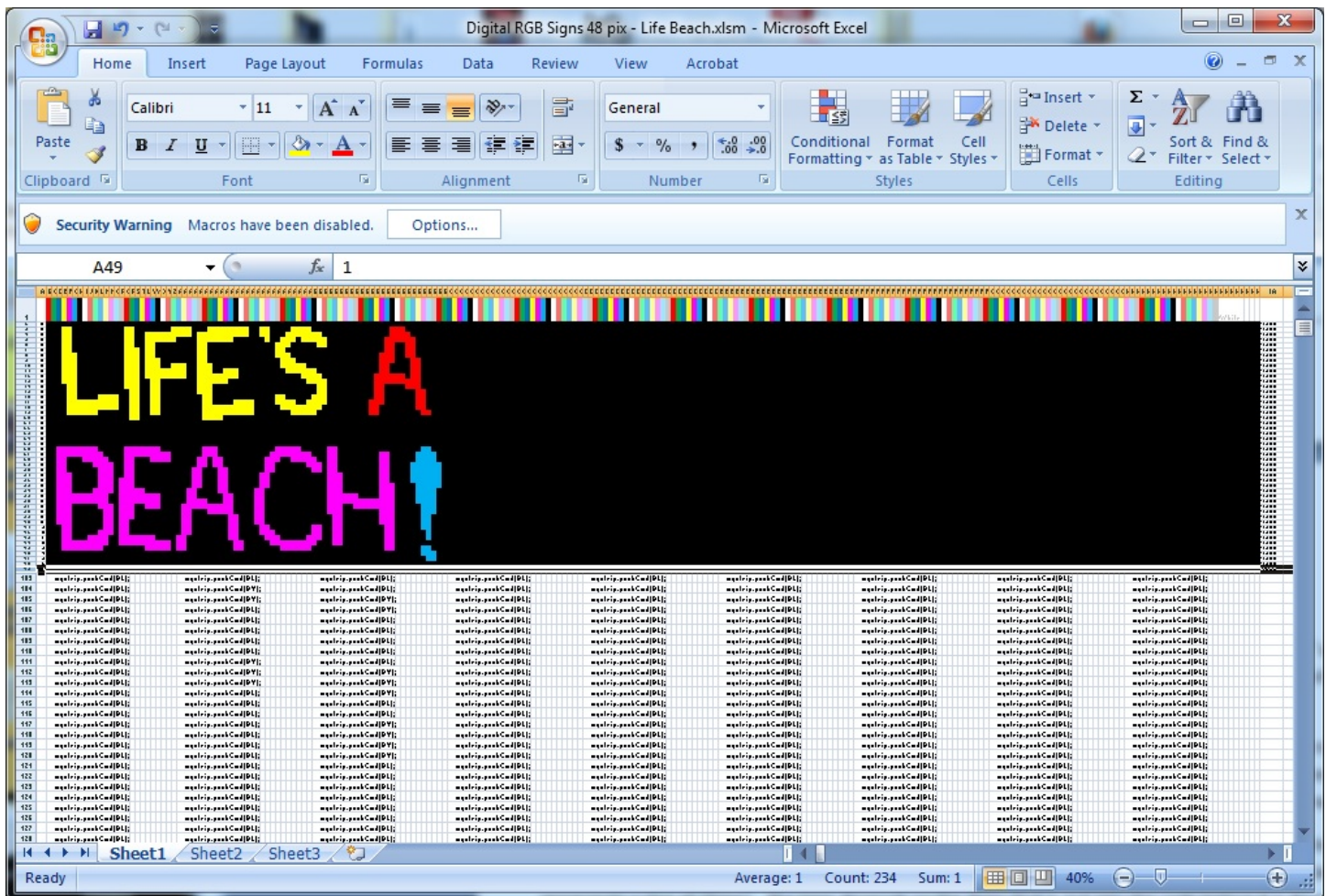
Above is an example of an image that has been "Drawn" in the black area of the spreadsheet. With this particular image, the last column used is the "BY" column. So we will only need to copy the first three columns of code generated at the bottom of the spreadsheet to the Arduino program to make this image work.

How To: Creating a Digital Light Wand for Light Painting Photography

Opening and saving an Arduino Project

To start creating a new program, you will need to open the Blank Arduino project and then use the spreadsheet to create and copy the program file into the Arduino project.

Open the Blank Arduino Project and you will see the following code contained in the program file.

```
/* HL1606 LED Strip control of individual LEDs
 *
 * B10000001 = Bright Blue
 * B10000100 = Bright Red
 * B10010000 = Bright Green
 * B10010001 = Bright Cyan
 * B10000101 = Bright Magenta
 * B10010100 = Bright Yellow
 * B10010101 = Bright White

 * B10000010 = Dim Blue
 * B10001000 = Dim Red
 * B10100000 = Dim Green
 * B10100010 = Dim Cyan
 * B10001010 = Dim Magenta
 * B10101000 = Dim Yellow
 * B10101010 = Dim White

 * B10000000 = Black / Off
*/

#include <LEDStrip.h>

#define SIPIN 9
#define DIPIN 10
#define CLKPIN 4
#define LATCHPIN 3

LEDStrip mystrip(DIPIN,SIPIN,LATCHPIN,CLKPIN);

#define SPULSES 5000
byte BB = B10000001; // Bright Blue
byte BR = B10000100; // Bright Red
byte BG = B10010000; // Bright Green
byte BC = B10010001; // Bright Cyan
byte BM = B10000101; // Bright Magenta
byte BY = B10010100; // Bright Yellow
byte BW = B10010101; // Bright White
byte DB = B10000010; // Dim Blue
```

```
byte DR = B10001000; // Dim Red
byte DG = B10100000; // Dim Green
byte DC = B10100010; // Dim Cyan
byte DM = B10001010; // Dim Magenta
byte DY = B10101000; // Dim Yellow
byte DW = B10101010; // Dim White
byte FDBB = B11000011; // Bright Blue
byte FDBR = B11001100; // Bright Red
byte FDBG = B11110000; // Bright Green
byte FDBC = B11110011; // Bright Cyan
byte FDBM = B11001111; // Bright Magenta
byte FDBY = B11111100; // Bright Yellow
byte FDBW = B11111111; // Bright White
byte FUDB = B11000010; // Dim Blue
byte FUDR = B11001000; // Dim Red
byte FUDG = B11100000; // Dim Green
byte FUDC = B11100010; // Dim Cyan
byte FUDM = B11001010; // Dim Magenta
byte FUDY = B11101000; // Dim Yellow
byte FUDW = B11101010; // Dim White
byte BL = B10000000; // Black / Off

void setup()
{
  ClearStrip(100);
}

void loop()
{

  ClearStrip(5000);
  PatternA(150);

}

void ClearStrip(int duration)
{
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
    mystrip.pushCmd(BL);
```

```
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.pushCmd(BL);
        mystrip.latch();
        delay(duration);
}


void PatternA(int duration)
{

/*
 * Paste the copied code from the spreadsheet
 * before the Close Curly Brace symbol below
*/


}
```

The way the Arduino works is basically under an endless loop. In the "Void Loop" area above, there is code to clear the strip with a delay of 5 seconds (5000 milliseconds) and then run a subroutine called PatternA. It is in this "PatternA" area that we will place the code generated within the spreadsheet once we have created the image or pattern that we want to generate.   The rest of the code should remain unmodified with exception of maybe the delay or duration number that is in the Void Loop area in the parenthesis in the   `PatternA(150);` Line. This number is the duration in milliseconds that you want each segment of the light pattern to remain lit for before moving to the next segment in the pattern (By "Segment" I am referring to each column in the spreadsheet image).

How To: Creating a Digital Light Wand for Light Painting Photography

SUMMARY OF IMAGE CREATION

To summarize the process of creating an image and getting it loaded into the Arduino board to use, here are the general steps:

1. Open the Blank Excel Spreadsheet for the light strip
2. Copy and paste colors from the color pallet to the black area of the spreadsheet to draw the image
3. Open the blank Arduino project
4. Copy the code generated from the spreadsheet to the Arduino project in the proper location
5. Verify the code in the Arduino Project
6. Connect the USB cable to your computer and the Arduino board
7. Download the code to the Arduino board