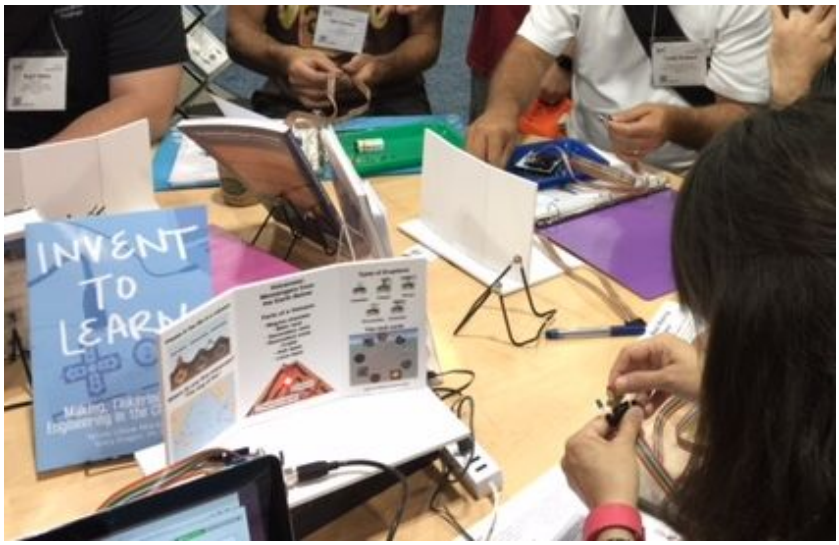
HyperDuino for Chrome Tutorial - Part 1

(https://goo.gl/8m0FEu)

Connect narrative videos and other web-based media to any physical student-made poster, diorama or model project.



Licensed under Creative Commons Share Alike
You may copy, remix, and distribute with attribution

Important: Before beginning this tutorial, you'll need to install the HyperDuino for Chrome app, and know how to open apps that you have installed on your Chromebook or other device.

Contents

Part 1 - Interactivity without a Computer

1. Creating basic interactivity: LEDs & touch sensing
2. Things to observe & learn
3. Planning the Interactivity: LEDs & touch sensing
4. Planning your project: media content & timecode list
5. Constructing the project board
6. Inserting the LEDs & connecting the sensors
7. Testing the LEDs & sensors

- What can you make with the HyperDuino?

Part 2 - Sensors Play Videos and Videos Turn on/off LEDs

8. The HyperDuino for Chrome app, setting the serial connection
9. Playlists
10. Creating a New Playlist
11. Saving Playlists
12. Adding your videos
13. Controlling LEDs
14. Sharing Playlists
15. Playlist options

- Also Included in the HyperDuino & Accessories Kit

- Bonus Experiment: The RGB LED

Introduction

The HyperDuino for Chrome system is an entirely new thing in the world of educational technology. In fact, entirely new in the world! Never before has it been possible to interconnect videos and student-made projects using a Chromebook (or any laptop running the Chrome browser).

With the HyperDuino system, you can take traditional curriculum projects like poster-board projects and physical models, and link them to videos and other digital media.

And, unlike many “starter kits”, the HyperDuino system provides a continuous path to ever more advanced activities, from learning about coding with block programming languages, to robotics, all the way to Arduinos in space with the ArduSat project.

The HyperDuino helps answer a common makerspace question, “Where do we start, and where is the curriculum?”

With the HyperDuino, schools use the curriculum-based classroom projects that they’re already doing as the foundation of the classroom maker area or library makerspace.

Imagine a principal or superintendent coming by to visit a new makerspace, looking at a random assortment of projects, and even 3D printing and rolling “robots”, and asking, "Very nice, but how does this raise our test scores? Where does this connect to social studies and language arts?" By using existing curriculum-based projects as the foundation of the makerspace, it's obvious and satisfying to administrators (and parents) who look in on a makerspace, and then the robotics, electronics, and coding are the icing on the standard curriculum cake.

The library turned makerspace is also now in a very familiar modality. Pre-Internet and student-owned mobile devices, teachers would schedule times to bring their students to the library to do research, and then the parents would make models at home for the students. Now teachers can let students do their research on their own or school devices anywhere/anytime, and then use the time in the library makerspace so that students can create those projects themselves. A multiple gain all around. More authentic hands-on work by students, and enhanced learning by their use of digital media to explain their projects.

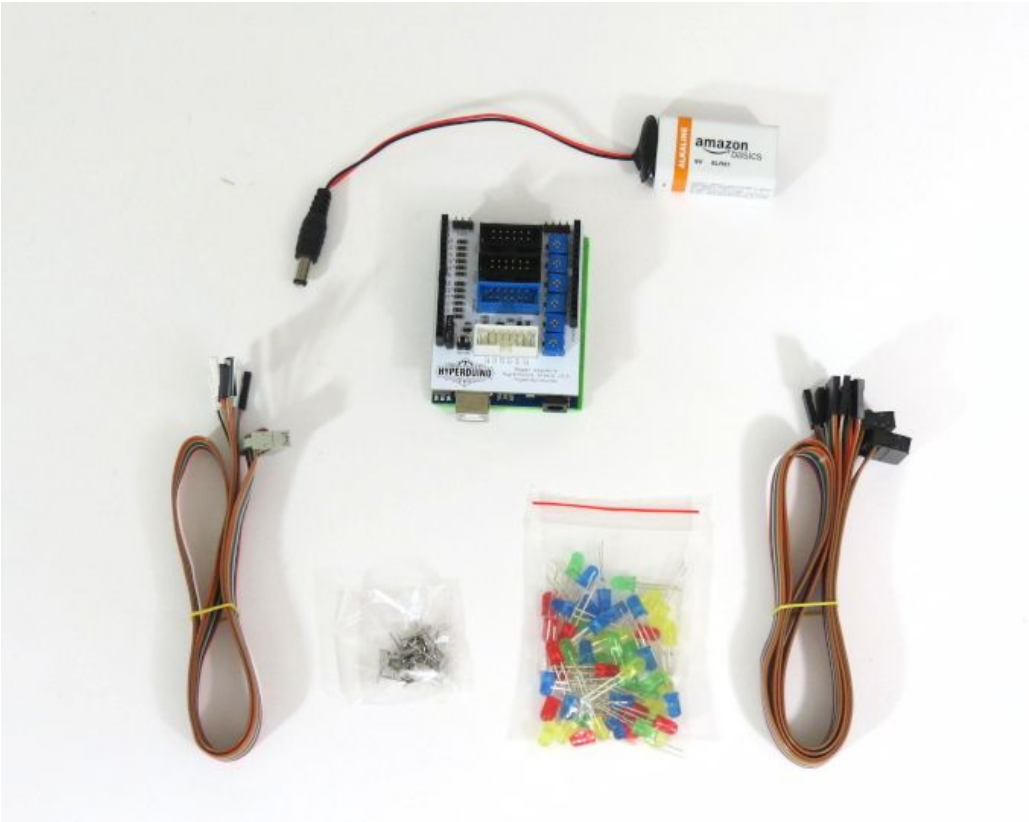
1 Creating basic interactivity: LEDs & touch sensing

We know you're eager to get started and see something happen, so let's do that now!

Start out by locating the materials that you will need for the first part of this tutorial.

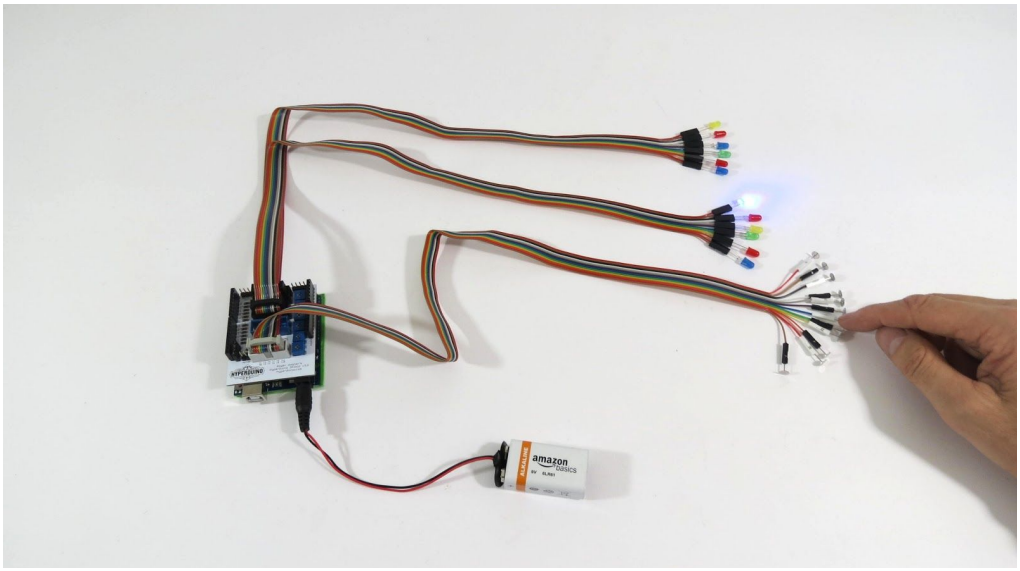
In the HyperDuino kit, locate and set out the following:

1. HyperDuino
2. Two rainbow ribbon cable with a black block connectors
3. One rainbow ribbon cable with a grey block connector
4. Plastic bag of colored LEDs
5. Plastic bag of "touchpoints" (they look like thumb-tacks)
6. 9 volt battery
7. Battery clip and wire

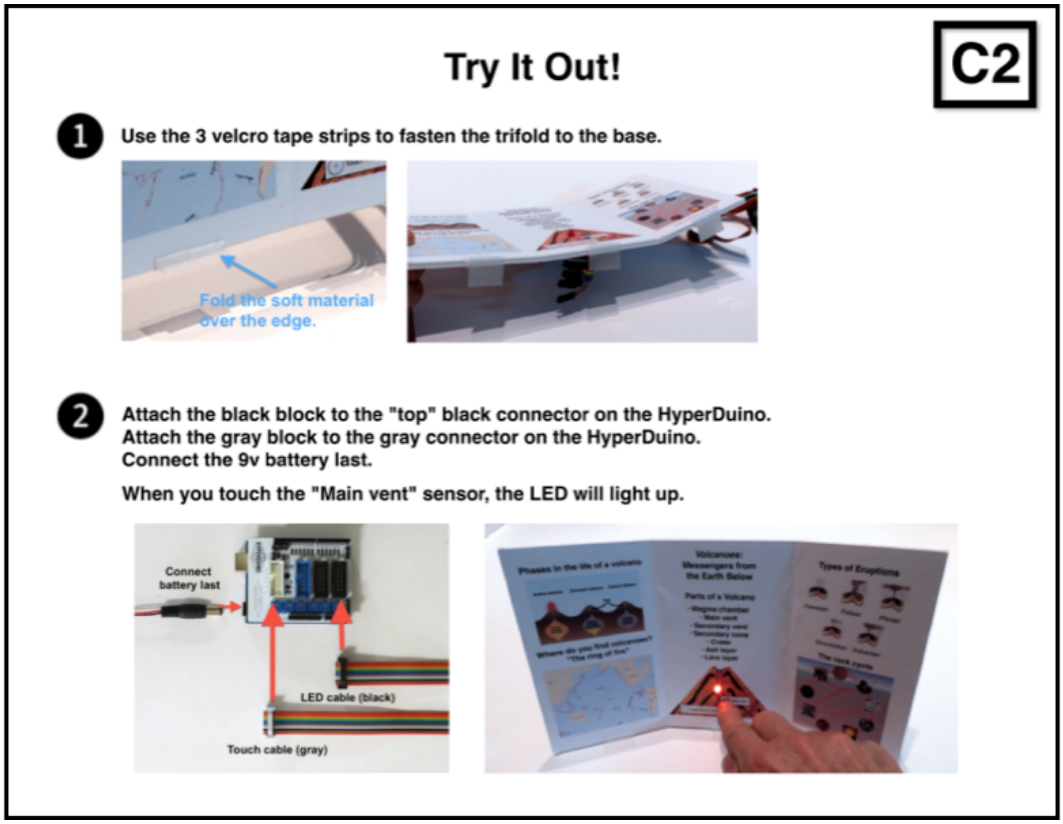


(http://hyperduino.com/resources/What's%20needed.jpg)

Put together the parts to look like this:

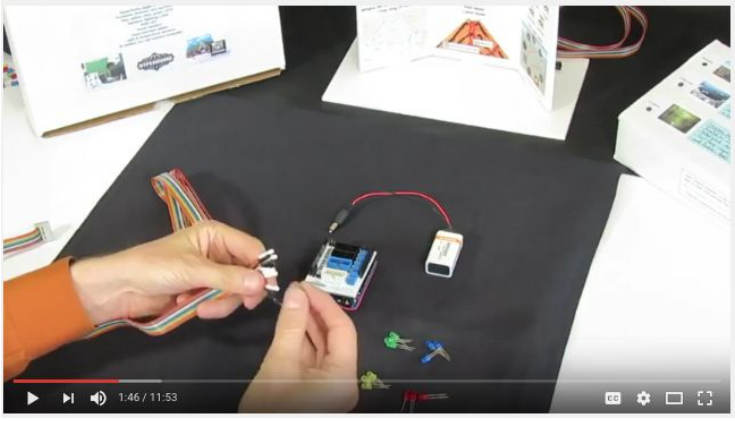


There is also an abbreviated graphic version of the instructions here:



If you use the abbreviated instructions, when you're finished, skip ahead to the next section, "Things to observe & learn". (Later, it's still a good idea to come back and read the more complete tutorial here.)

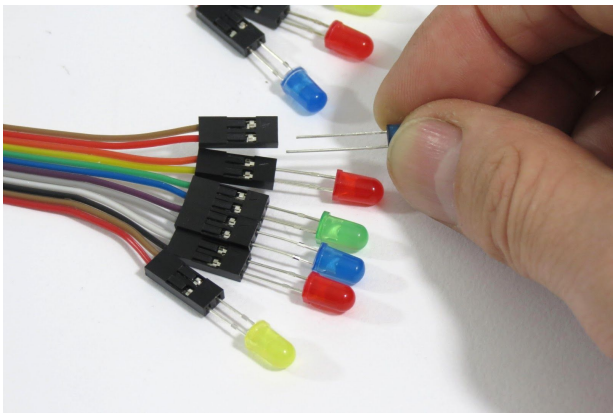
If you have access to YouTube, and would prefer a video to written instructions for the first part of this activity, click on the link here.



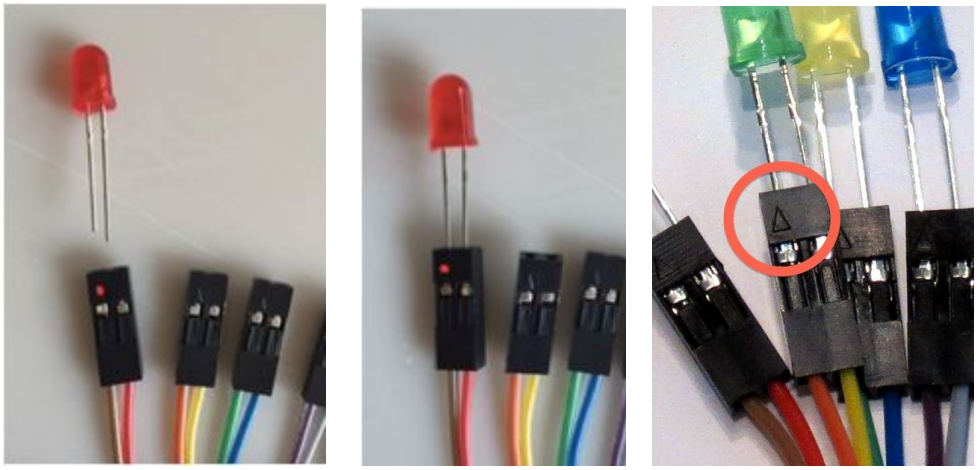
When you've finished watching the video, and if you were able to duplicate what is shown in the video, you can now skip ahead to the next section, "Things to observe & learn".

Insert the LEDS

Insert LEDs ("Light Emitting Diodes") into each of the socket-pairs of the LED (black block) rainbow cable (12 in all). The order of the LED colors is up to you!



When you put the legs of the LEDs into the sockets, **put the shorter leg into the the left-most** (left-most when you can see the metal part of the connectors) **socket that has a tiny raised triangle on one of the sockets**. The “short leg socket” will also be either the **brown, orange, green, purple, or white wire** of the pair of wires going to your LED.

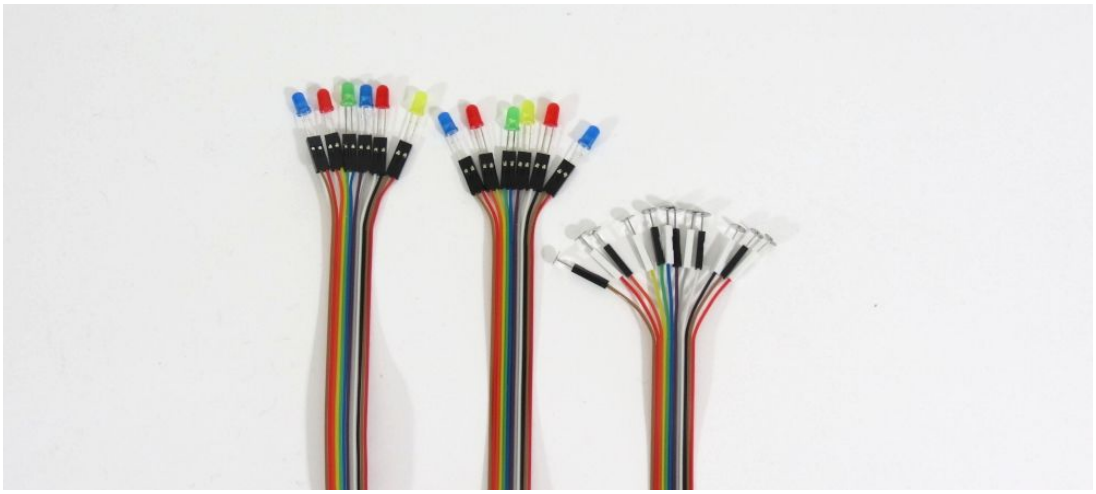


(http://hyperduino.com/resources/LEDs%20closeup%20triangle.jpg)

Note: if you can’t easily see the triangle on the left side of each socket pair, just insert the LED anyway. Later if it doesn’t light up, you can just flip it around to make it work.

Insert the Touchpoints

Insert touchpoints into each of the individual (alternating black and white) sockets on the side of the touch rainbow cable (white block).



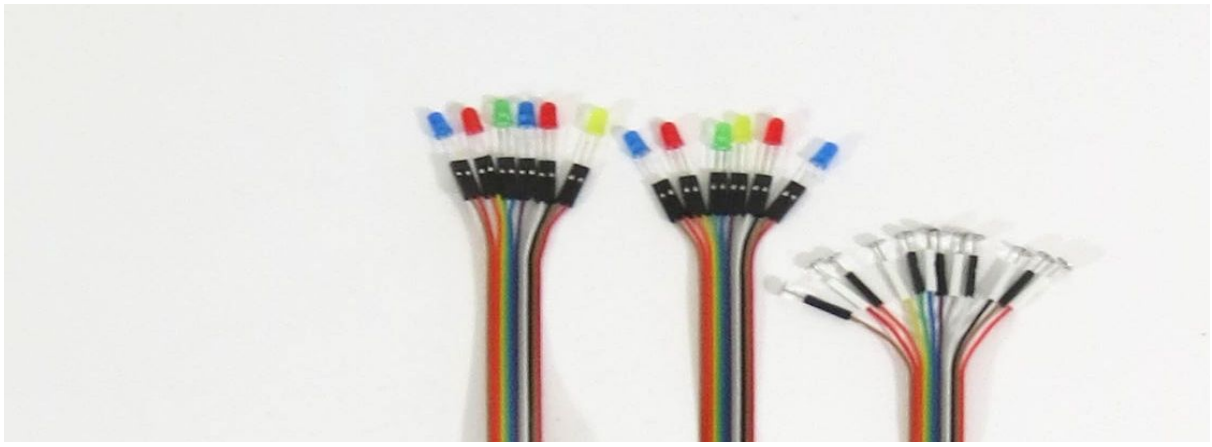
For each LED and touchpoint, try to push the LED legs down into the socket, so as to have a good electrical connection.

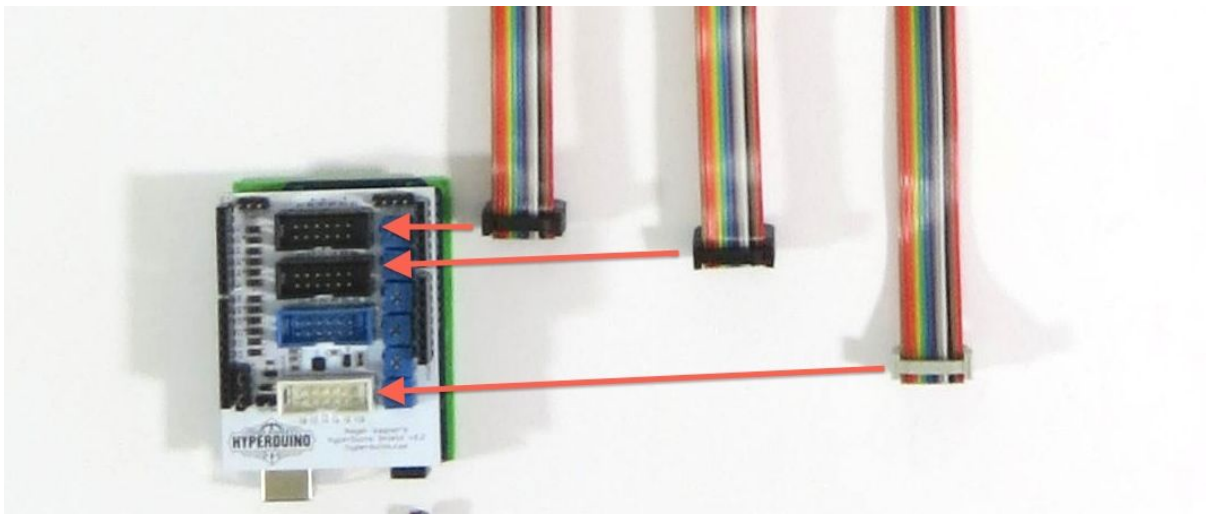
Note however that they don't go all the way into the socket (see the above photos). There should be about a half-inch of the metal legs still showing for each LED and touchpoint. This is to allow for them going through foamcore, cardboard, paper, and other materials of a project.

Connect to the HyperDuino

Connect the cables with the black blocks to the black boxes on the HyperDuino. The black boxes are for controlling LEDs.

Connect the cables with the grey blocks to the white box on the HyperDuino. The white box is for the touchpoints.





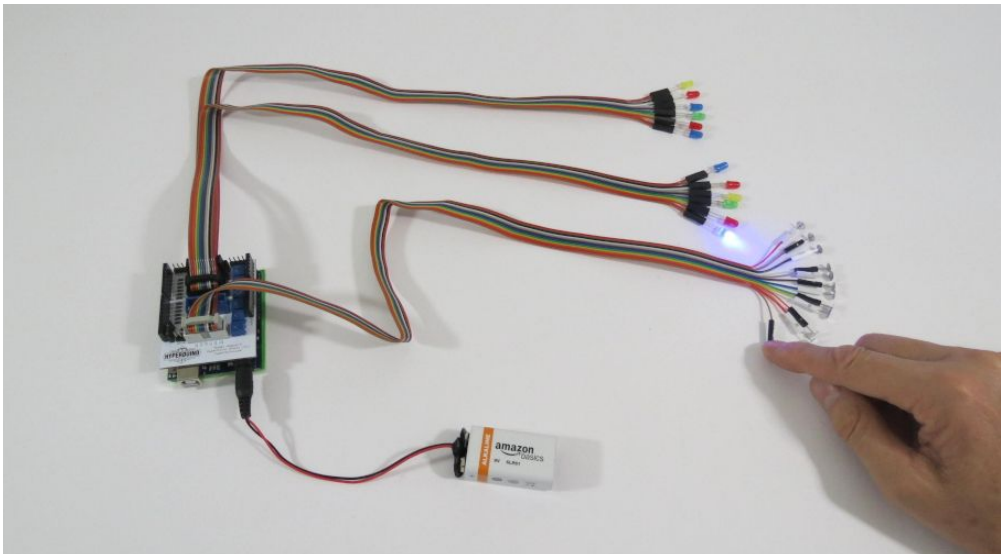
Connect the Battery

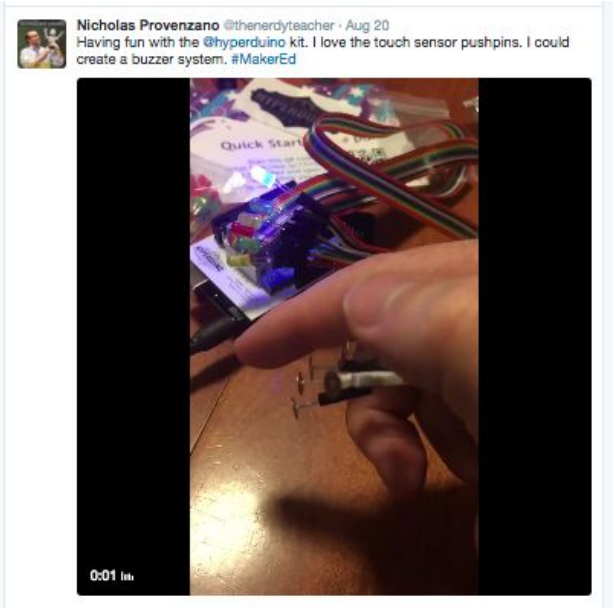
Connect the battery last. This is because when the HyperDuino starts up, it does an automatic process called calibration that measures the electrical behavior of the touchpoints that are plugged in at the moment that they are not being touched. That way it can tell the difference in their behavior when they are being touched.

If in your enthusiasm, you already connected the battery earlier, it's not a problem. Just disconnect and reconnect the battery plug from the HyperDuino now. This calibrates the HyperDuino for the touchpoints that you've attached.

What should be happening…

Now, if you touch each of the touchpoints, the LEDs should each come on as they're shown in this photograph, and in Nicholas Provenzano's (@thenerdyteacher) Twitter video.





https://goo.gl/FUisH1

If the touch sensors seem to not be responding, or the LEDs are coming on even when you're not touching a sensor, try unplugging and re-plugging the battery while you are **not touching the touch cables**.

If an LED doesn't light up, try reversing the legs in the sockets. Perhaps you have the short leg in a socket that doesn't have a triangle. If that doesn't work, try a different LED. Sometimes there are ones that don't work.

When you hold the touch cables, try to hold them lightly by the edges. Better still, let them lay on the table, and only touch the touchpoints to test. Touch the flat part of the touchpoint for a good contact, not just the edge.

If the LEDs seem to flicker on their own, try a different battery, in case the one that you are using is low. When the battery voltage gets low, the sensor behavior becomes somewhat random.

Note: We recommend using Amazon 9v batteries for replacements. Not only are they the least expensive, they are the easiest to disconnect. Other 9v batteries brands can be difficult to separate from the clip without damaging it.

2 Things to Observe & Learn

- **Each touchpoint controls one light**. Try touching multiple touchpoints, or running your finger across them like a musical instrument.

- LEDs are "Light Emitting Diodes". Diodes are electrical components that allow electrical current to flow through them in one direction. The triangle socket is connected to the negative side of the battery. This is also called the "Ground" side of a circuit. **For the LED to work (light up) the short leg must be connected to Ground (triangle) socket**.

- The touchpoints only have one wire, but the LEDs have two. How can they both work? **The answer is that the LEDs require that a continuous current flows through them to light up**, in part because they are using energy to create the light, and that energy needs to continuously flow.

  The touchpoints use a different electrical behavior called "capacitance". With capacitance, the HyperDuino (using the battery) momentarily charges up the touchpoint rather like the way a comb gets charges with static electricity when you run it through your hair on a dry day. A larger comb would hold more of a charge, and a smaller comb less.

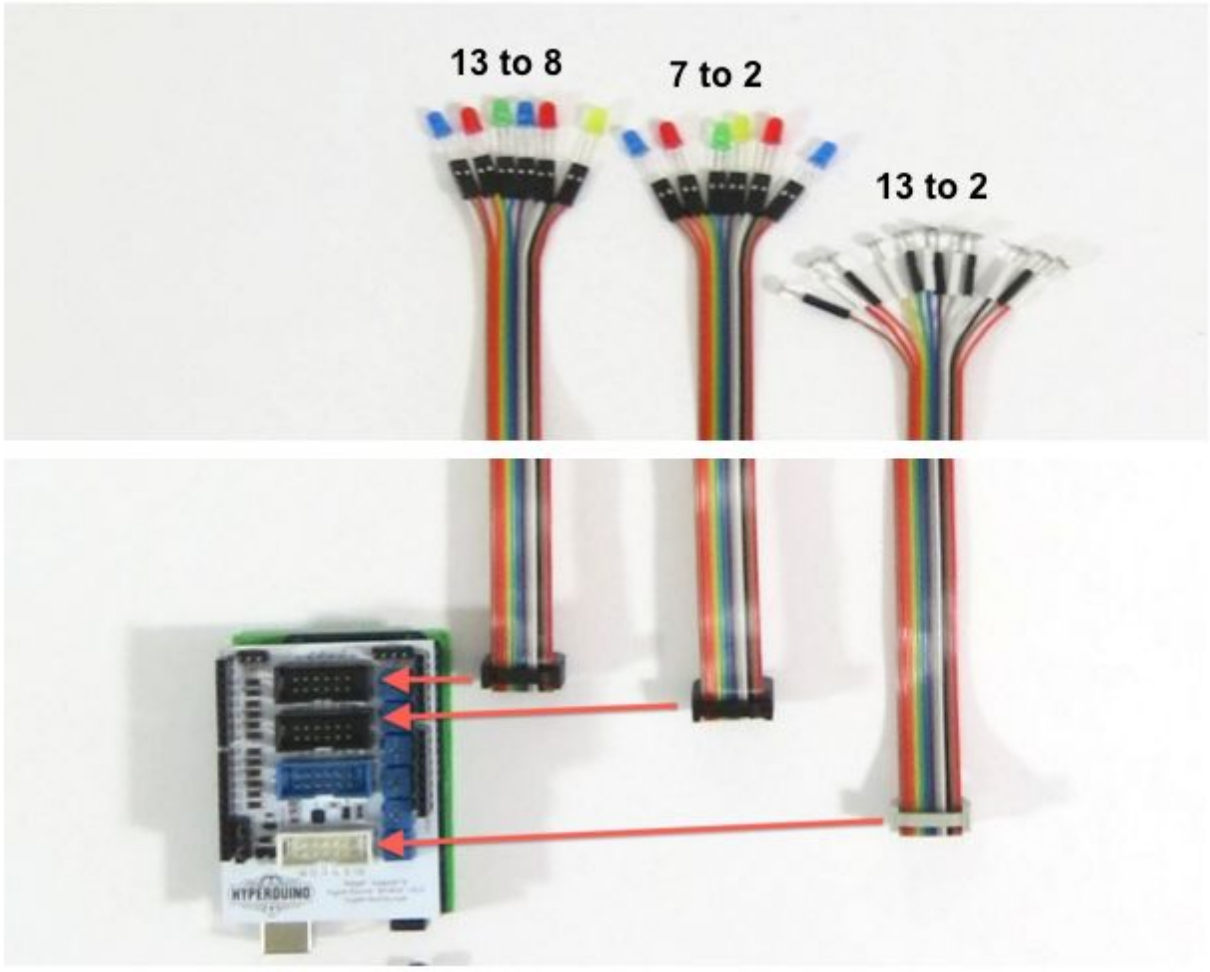
  When you touch the touchpoint, your body absorbs some of the charge, and the electronics on the HyperDuino can detect that. Only one wire is needed because the current doesn't flow continuously. It's sort of like the difference between running a circulating fountain pump, or just filling a glass of water and seeing how long it takes. One needs a "loop", the other just a hose and a timer.

- The touchpoints and LEDs are numbered, and the wires and socket colors can help you to know which is which.

  Each wire or pair of wires connects to pins in the boxes on the HyperDuino. Those connections are numbered **2 through 7 for the top box of the LED connections, 8 through 13 for the second box**.

- For the touch cable, the wires starting with the **brown wire on the left outside edge** of the cable controls the LEDs in numbered positions **13 through 8,** which are attached to the lower black connector of the HyperDuino. The other side of the touch rainbow cable, with **red on the outside edge**, controls LEDs in numbered positions **2 to 7**, which are connected to the upper black connector.

- If you're wondering why touch sensor and LED numbering starts with "2", [click here](#).



Note: If you would like to play back YouTube videos without ads, you might consider https://safeshare.tv/ or http://safeyoutube.net/ You would use "Webpage" as the option in the playlist. You won't be able to respond to a particular time range, but this might be something to consider for your projects.

3 Planning Your Project: LEDs & Sensors

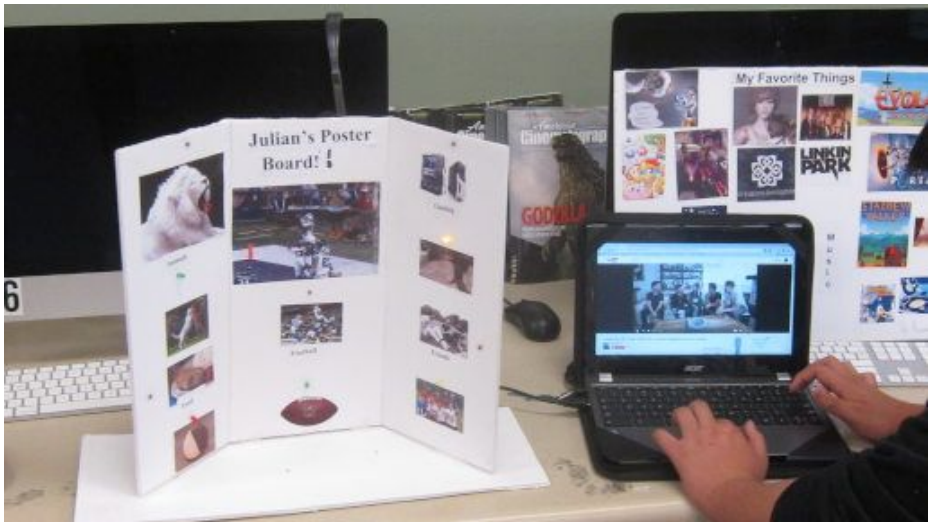
The HyperDuino kit includes a foamcore board printed to represent a tri-fold poster presentation on volcanoes.

We will use it to go through the basic planning and construction steps of a typical curriculum-based interactive maker project.

In the sample volcano tri-fold, there is a pre-made hole for an LED, and two places indicated with circles for the touchpoints.

One touch sensor will play a short video about the parts of a volcano, and the other sensor will play just the part of the video where the student says “main vent”, which is the part of the volcano diagram where the LED is located.

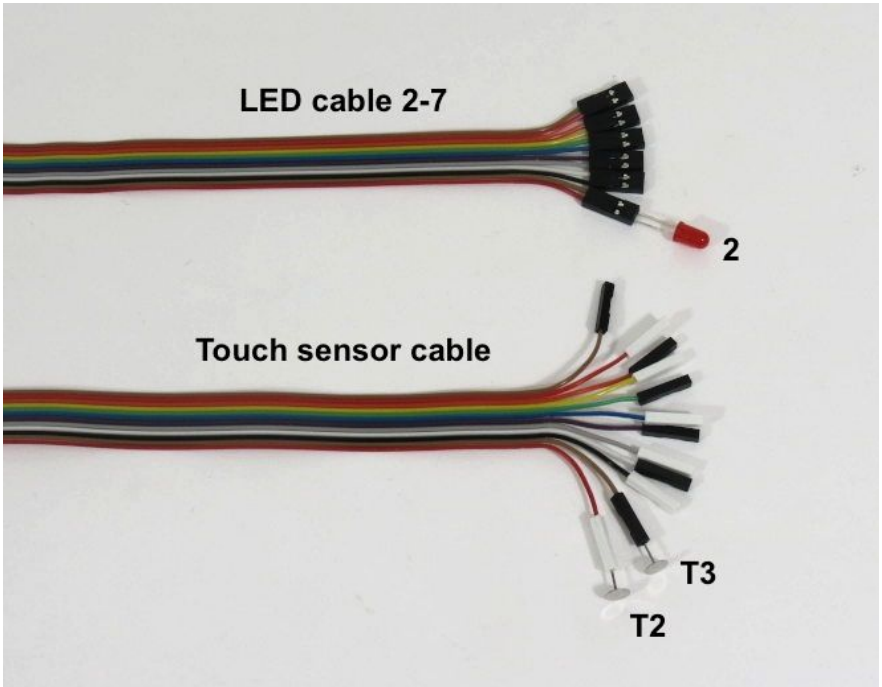
In a more developed project, there would be sensors on all of the panels to start videos that explained the subject, and LEDs that would illuminate to draw attention to the relevant parts of the diagrams.



Earlier, you attached 12 LEDs and 12 sensors, and observed the 1-to-1 relationship between each sensor, and its LED with the same number.

In the example tri-fold project, you'll just use 2 of the sensors (numbers 2 and 3), and one LED (#2).

The cables before you wire them to your project will look like this:



4 Planning Your Project: Media Content & Timecode List

In a typical project, before actually connecting the LEDs and sensors, and even before using the HyperDuino for Chrome app, you would create a Media & Timecode list, where you would organize the URLs for the videos and other digital media that you would use in your project, along with assigning the numbers for the touchpoints and LEDs.

In planning, keep in mind that each sensor can only play one video, but a playing video can trigger one or more LEDs.

For this example project, we will just turn on one LED, the one for the “main vent”, at the moment when the student actually says the words, “main vent”.

The Media & Timecode Planning list looks like this:

- **Sensor 3, “Start the video”:**
  - Complete volcano video
  - Start: 00:00 End: 00:17
  - https://www.youtube.com/watch?v=VHrjZ7YmvlA

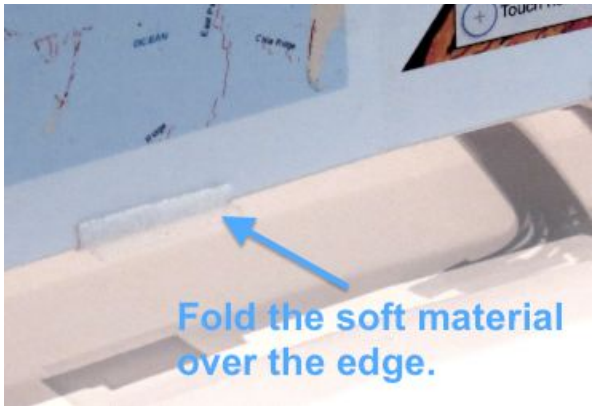
- **LED 2, “Main vent”:**
  - “Main vent” segment of volcano video
  - Start: 00:10 End: 00:12
  - https://www.youtube.com/watch?v=VHrjZ7YmvlA

- **Sensor 2, “Main vent”:**
  - “Main vent” segment of volcano video
  - Start: 00:10 End: 00:12.2 (just slightly longer than the LED is “on”)
  - https://www.youtube.com/watch?v=VHrjZ7YmvlA

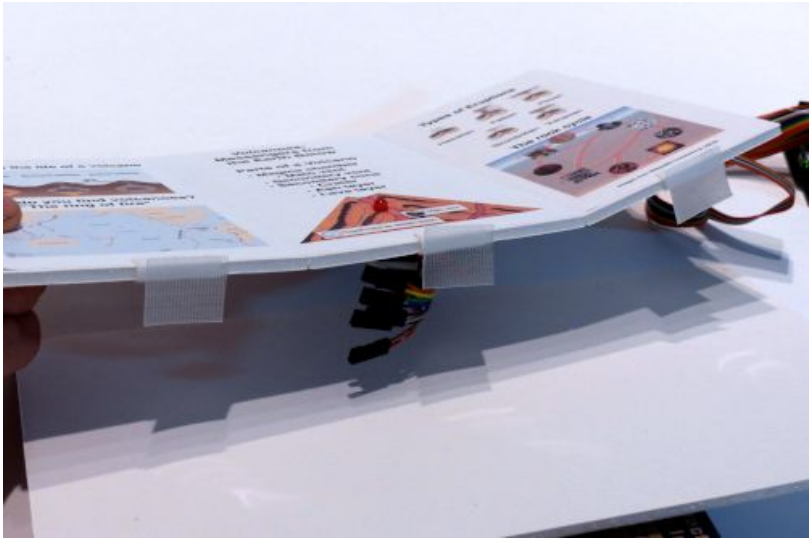
5 Constructing the project board

Prepare the “tri-fold” shape for the base, by pressing firmly on the front of the board to “snap” open cuts that have already been made in the back of the board, but gently enough also that you don’t tear the panels apart.

Use the velcro strips included in the HyperDuino kit to attach the tri-fold to the base. There are two types of material for the velcro, one soft (fuzzy) and the other more firm (the “hooks”). Use the soft material to wrap over the edge of the bottom of the tri-fold.



Gently place the strips of the matching velcro on the soft strips attached to the tri-fold, and remove the backing just before you carefully align and place the tri-fold on the base.

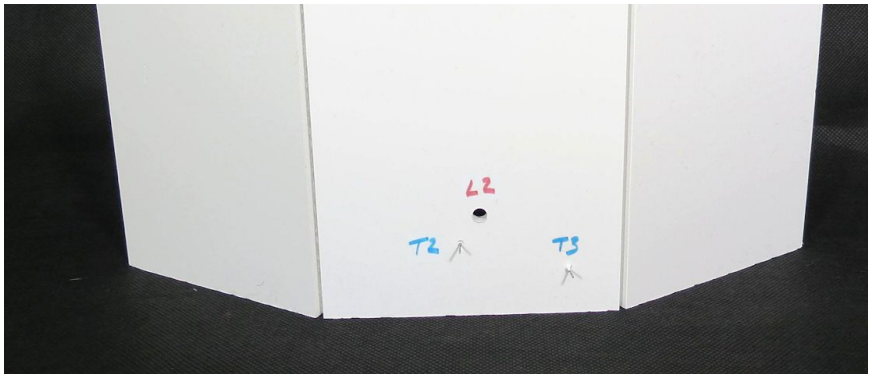


Label the Back of the Board

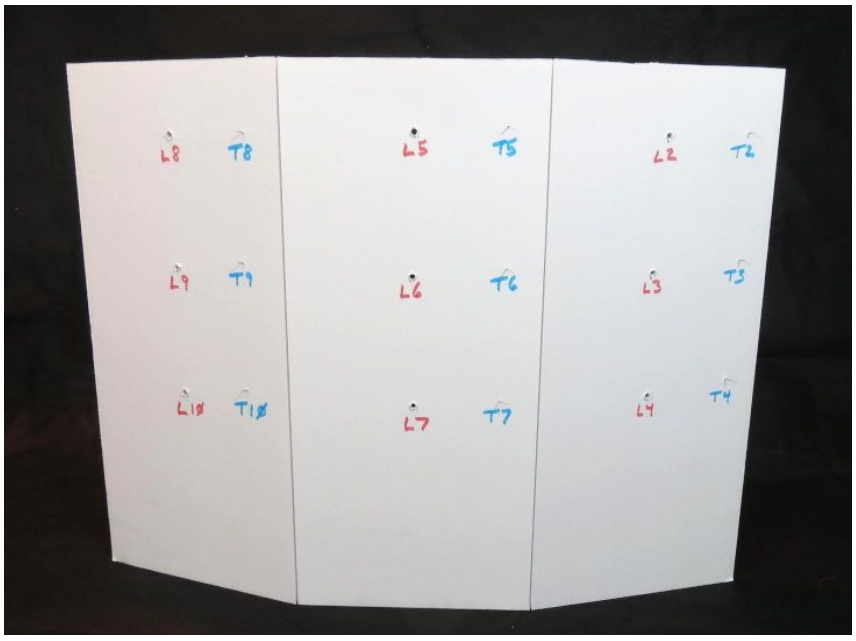
Looking at the back of the board, use a felt tipped pen or similar to write the correct number for each touchpoint and LED on the back of the board near the LED or touchpoint post.

You will be using touchpoints #2 and #3, and LED #2. To make the labels more understandable, you can use “T” for touchpoints, and “L” for LEDs.

Write “T2” and “T3” for the touchpoints, and “L2” for the LED.



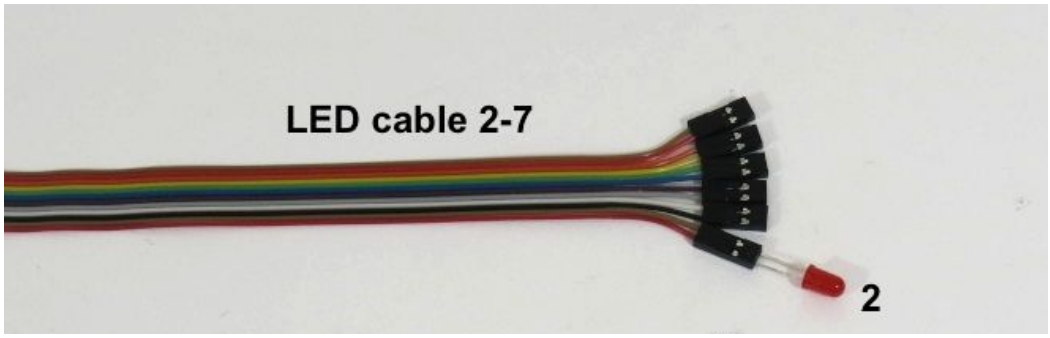
For a more developed tri-fold board, the labeling might look like this:



6 Inserting the LEDs & Connecting the Sensors

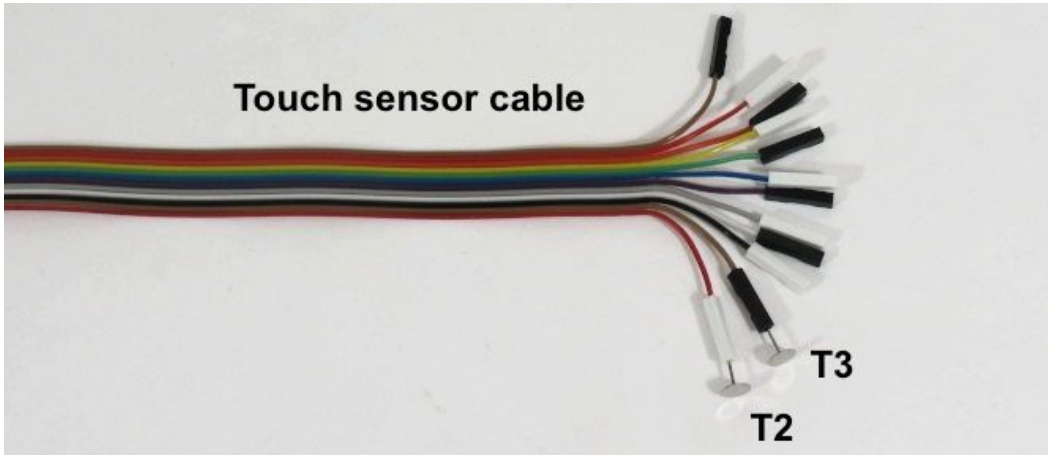
LEDs

Remove 5 of the LEDs that you inserted in Part 1 of this tutorial, leaving just one at the far right, when you hold the cable so that you can see the metal part of the socket connectors. This is the LED in numbered position 2 of the HyperDuino. You can also identify it by noticing that the red wire is on the outside edge of the rainbow cable.



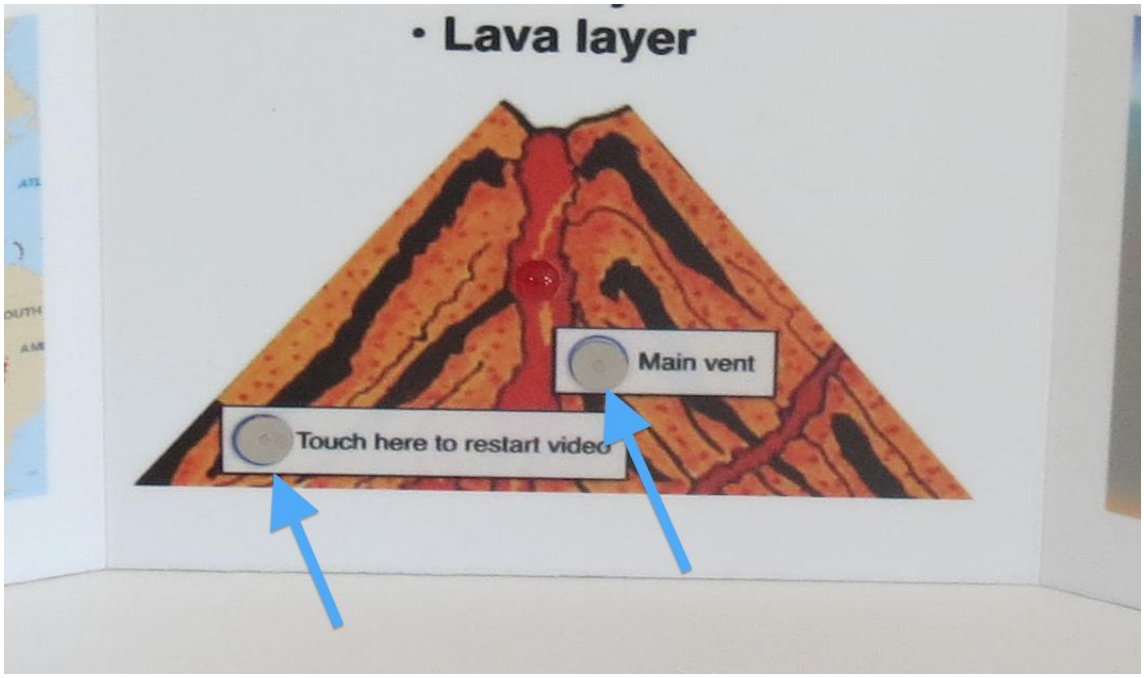
Touchpoints

Remove all but one of the touchpoints in the sockets of the rainbow cable that you had inserted earlier, leaving just two touchpoints in the brown and red wires next to the black wire..



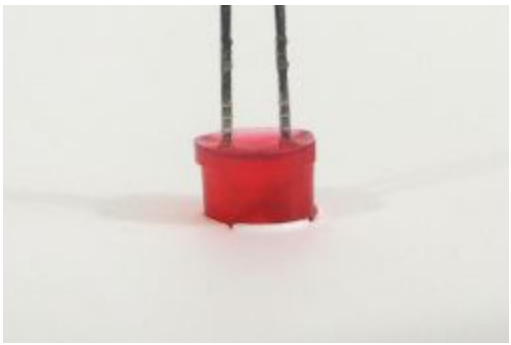
Keeping track of the wires that are connected to the touchpoints, remove them from the sockets, and push the touchpoints into the white foamcore from the front, centered in

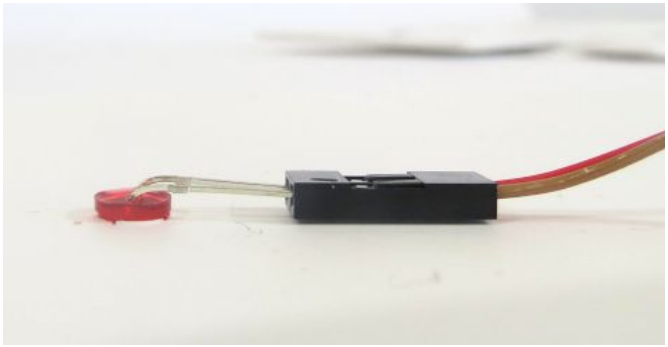
the circles labeled “Main vent” with the red wire (touchpoint T2) and “Touch here” with the brown wire (touchpoint T3), and then reconnect them on the back of the board.



For the LED, push it through the board from the back. A little extra pressure may be necessary, but you should get a nice, snug fit of the LED. It’s not necessary to disconnect the wires of the LED while pushing it into position. In the future, you can use the blue plastic tool for making holes for LEDs.

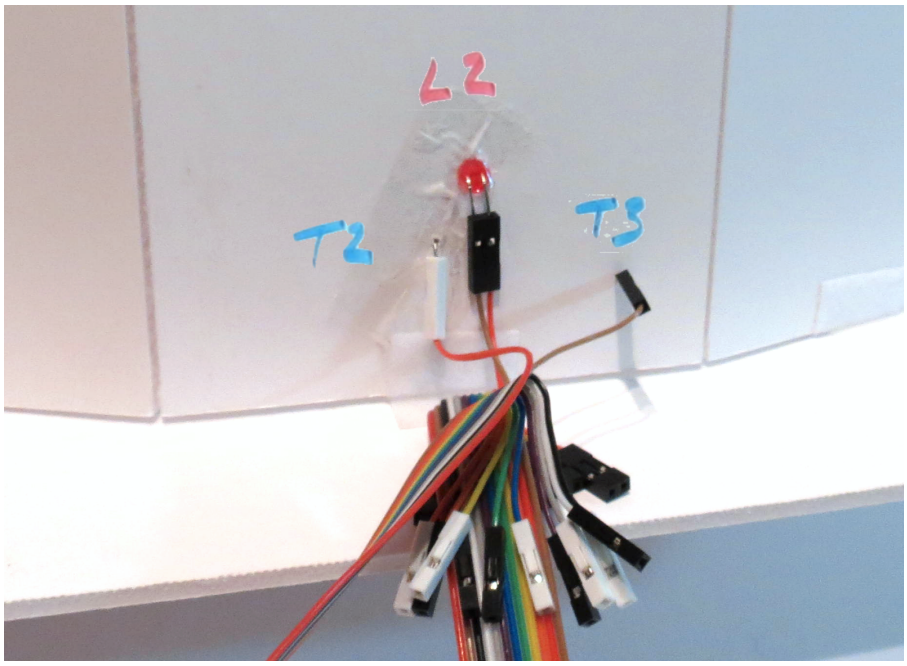
There is a lip on the LED that will keep it from pushing all the way through, if you don’t push too hard. For your own projects, you can use a pointed object to make a hole for the LED. You’ll get the best results making the hole from the front, and inserting the LED (but not pushing it all the way through) from the back.



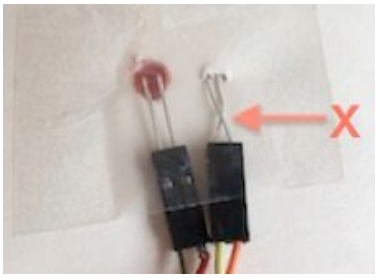


A bit of transparent tape behind the LED will help secure it to the board.

The socket connectors to the touch points can either be left perpendicular to the board, or bent over at 90 degrees and taped down.

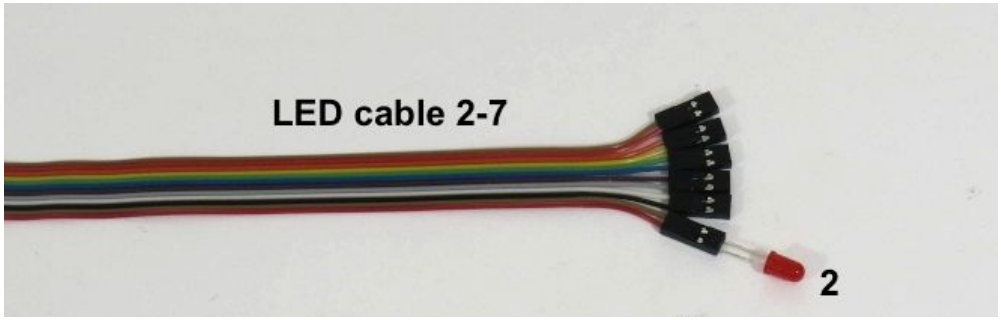


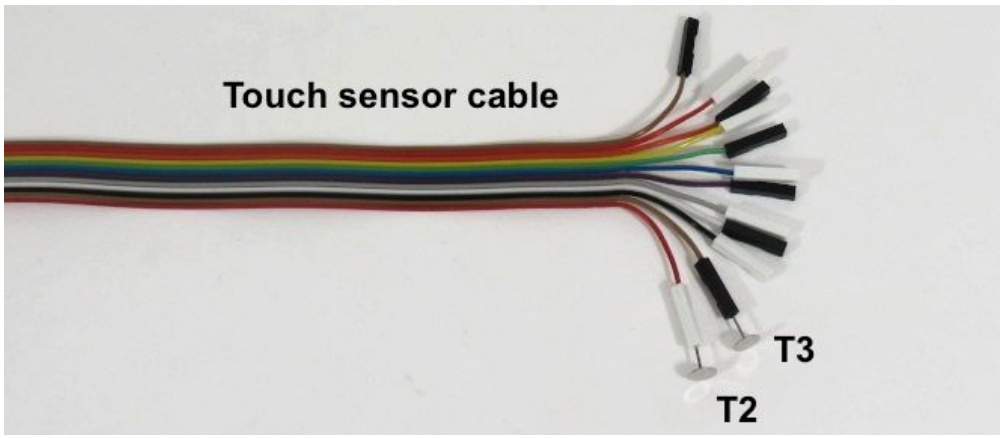
Note: Be careful not to cross the wires of the LED. It doesn't hurt anything if they're crossed, but it will prevent it from lighting up properly and in other situations, crossed wires can damage electronic equipment. (The HyperDuino is designed to be rather forgiving of things like crossed wires).



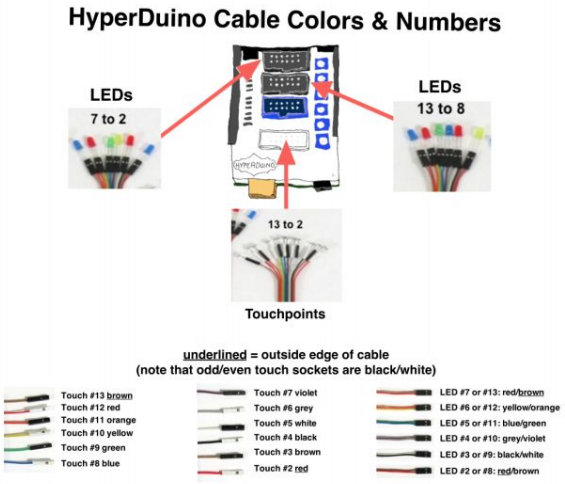
For future reference, you should note that the different color wires of the rainbow ribbon cable can help you identify which number LED or touchpoint is associated with that wire.

The lowest number for any cable starts on the side of the cable that has the red wire on the edge.





The easiest way is to identify a socket number is to just count starting at “2” from the outside edge. However, you might also like to [print out this page] to use as a guide.

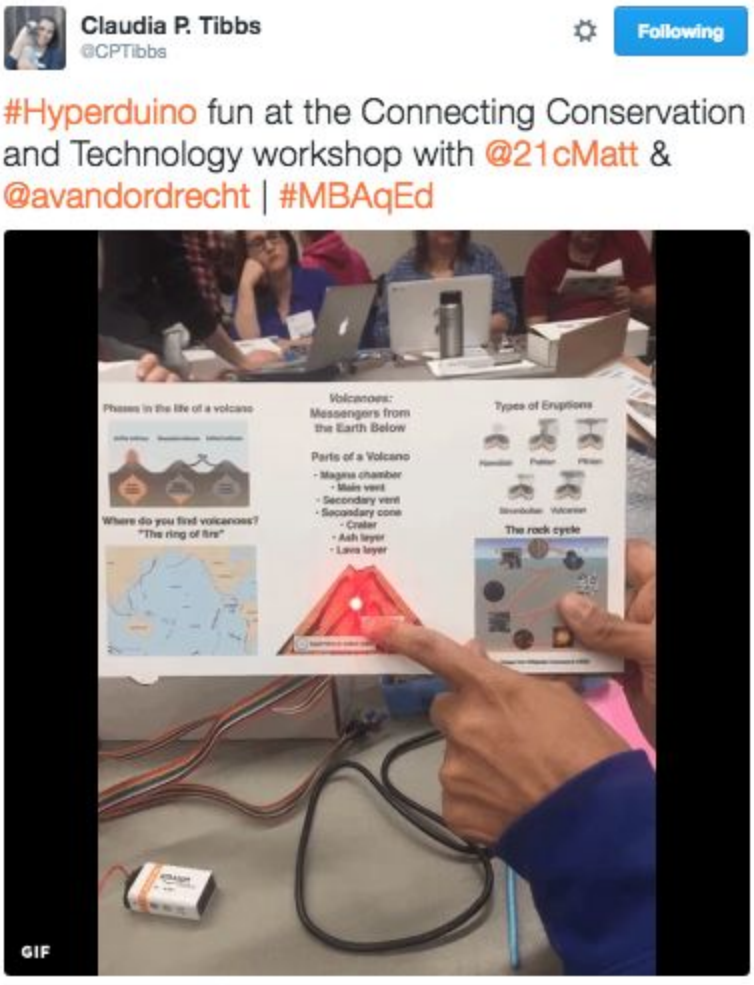


https://goo.gl/M0K9UH

7 Testing the Sensors and LEDs

When you've inserted your LED and touchpoints, re-connect the battery to the HyperDuino. If you are using a HyperDuino produced after October 1, 2016, or if you have updated the HyperDuino firmware since then, LED #2 should blink a few times. At that point, touching the "Main vent" touchpoint should light up the LED, since without a computer, touch sensor T2 always turns on LED 2.

If the LED doesn't light up, see if it was perhaps flipped around with the short leg not in the triangle socket when it was re-attached. That is also to say, if an LED doesn't light up, just flip it around in the sockets to see if that makes it work. If that doesn't work, try a different LED.



https://twitter.com/CPTibbs/status/797564004117848064

What Can You Make With This?

You have now set up the tri-fold model with basic interactivity. There are many, many projects that can be created just using the simple interactivity that you have now established.

You can now easily understand how to place the lights in not just a physical model, but things like a map, diagram, or even a work of art.

The touch sensors can also be positioned in a “menu” for the display, perhaps simply made from 3 x 5 cards, with the touch sensors next to titles and paragraphs of explanatory information.

If a person looking at the display touches the sensor, they can read more detail about a part of the project, and also see a light come on to direct attention exactly that place in the model.

Next up, is connecting a video to the project, and controlling the LED by a video as it plays. To learn how to connect digital media such as videos, webpages and more to your interactive models, click here to go to Part 2 of this tutorial.

Another Example Project: National Parks

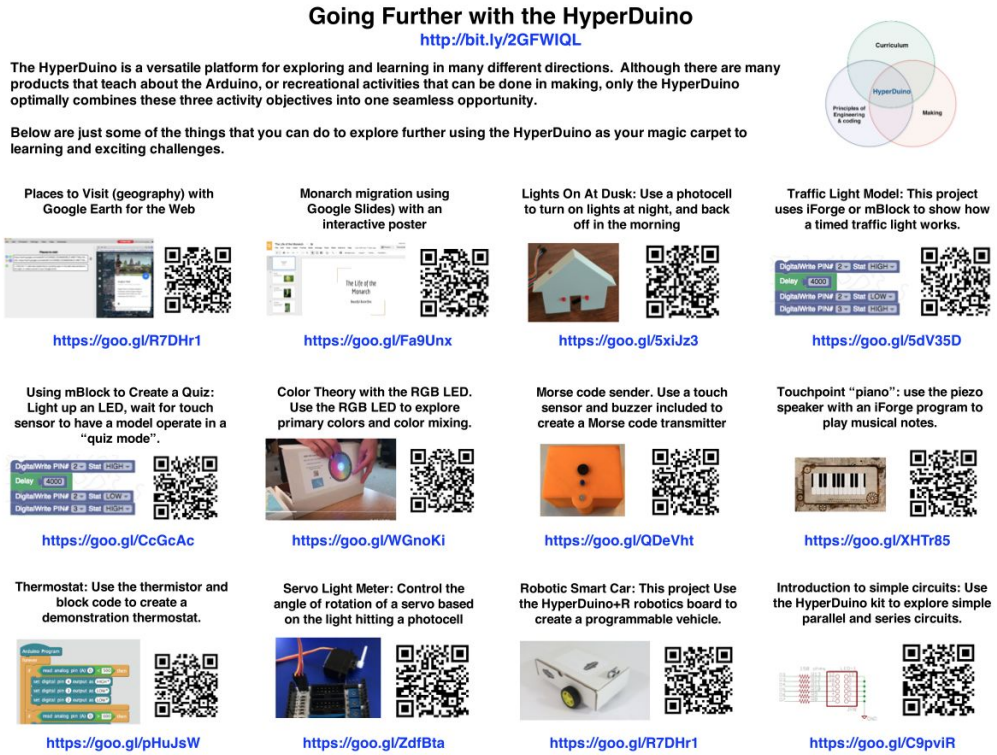
You can use the cardboard box that the HyperDuino or the update came in (or any other box) to make a simple interactive project highlighting some of your favorite U.S. national parks.

[Click here for the instructions](#) on how to make your own National Parks Project.

https://twitter.com/MrsTrakulboon/status/746462955563212801

https://www.youtube.com/watch?v=Mbv_3yyMGHk

And that's just a start! Here is another Google doc with a lot of ideas for going further with the HyperDuino:



Going Further with the HyperDuino

http://bit.ly/2GFWIQL

The HyperDuino is a versatile platform for exploring and learning in many different directions. Although there are many products that teach about the Arduino, or recreational activities that can be done in making, only the HyperDuino optimally combines these three activity objectives into one seamless opportunity.

Below are just some of the things that you can do to explore further using the HyperDuino as your magic carpet to learning and exciting challenges.

Places to Visit (geography) with Google Earth for the Web
https://goo.gl/R7DHr1

Monarch migration using Google Slides) with an interactive poster
https://goo.gl/Fa9Unx

Lights On At Dusk: Use a photocell to turn on lights at night, and back off in the morning
https://goo.gl/5xiJz3

Traffic Light Model: This project uses iForge or mBlock to show how a timed traffic light works.
https://goo.gl/5dV35D

Using mBlock to Create a Quiz: Light up an LED, wait for touch sensor to have a model operate in a "quiz mode".
https://goo.gl/CcGcAc

Color Theory with the RGB LED. Use the RGB LED to explore primary colors and color mixing.
https://goo.gl/WGnoKi

Morse code sender. Use a touch sensor and buzzer included to create a Morse code transmitter
https://goo.gl/QDeVht

Touchpoint "piano": use the piezo speaker with an iForge program to play musical notes.
https://goo.gl/XHTr85

Thermostat: Use the thermistor and block code to create a demonstration thermostat.
https://goo.gl/pHuJsW

Servo Light Meter: Control the angle of rotation of a servo based on the light hitting a photocell
https://goo.gl/ZdfBta

Robotic Smart Car: This project Use the HyperDuino+R robotics board to create a programmable vehicle.
https://goo.gl/R7DHr1

Introduction to simple circuits: Use the HyperDuino kit to explore simple parallel and series circuits.
https://goo.gl/C9pviR

Click here to continue to Part 2 of the HyperDuino Tutorial