TEAM PROGRAMMING CHALLENGE

This challenge is designed to demonstrate how widely-available open-source hardware and software can be used to quickly prototype new products. You will be using the components in front of you to build a prototype of WiFi-controllable "smart" thermostat.

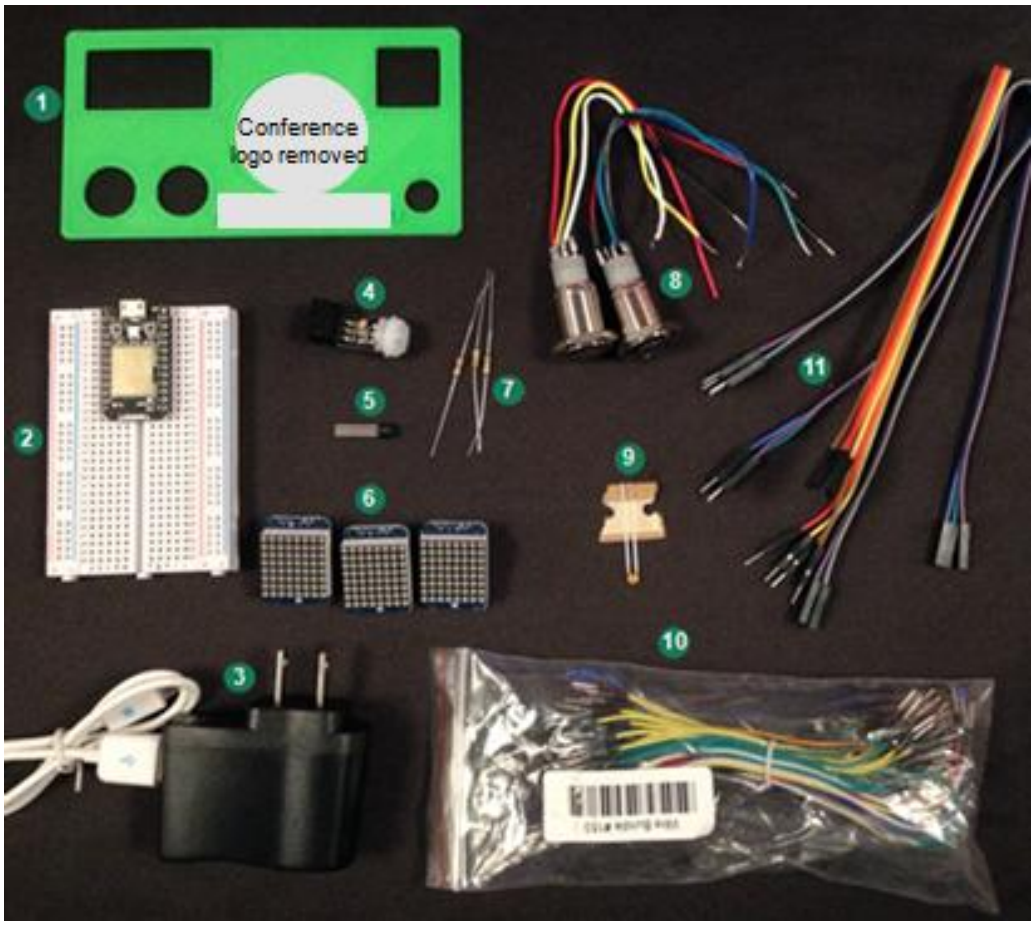
The following instructions incrementally build the thermostat prototype, but there are opportunities to parallel process between the hardware assembly and software steps (each step is labeled either Hardware or Software). We suggest organizing your team into a hardware and software team.

In addition to completing the assembly and programming tasks, each team will also be required to develop a sales pitch, which will be judged qualitatively (see final steps in instructions for details).

BILL OF MATERIALS

Each kit should have the following components:

1. (1) 3D Printed mouting board
2. (1) SparkCore chip & breadboard
3. (1) Power supply & USB Charger
4. (1) Motion sensor
5. (1) Temperature sensor
6. (3) LED matrices
7. (3) Resistors
8. (2) Push buttons
9. (1) Capacitor
10. Connecting wires
11. (4) Jumper cables



ACTIVITY INSTRUCTIONS

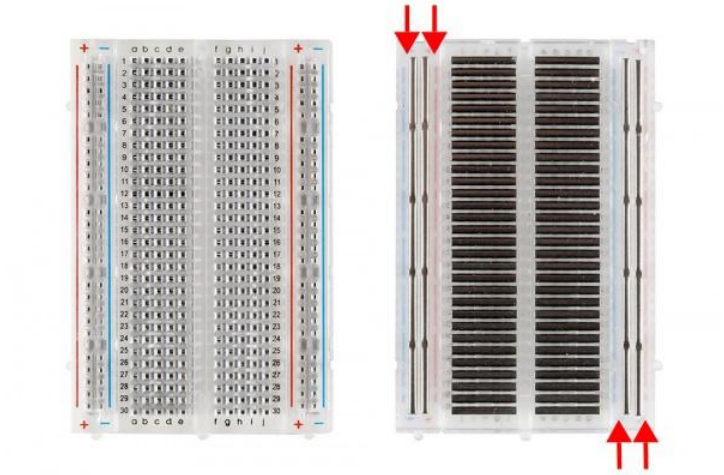
1. HARDWARE: Learning to use a breadboard

An electronics breadboard is a component to prototype circuits without the need for soldering.

Under the plastic covering, a breadboard contains horizontal and vertical rows of metal strips

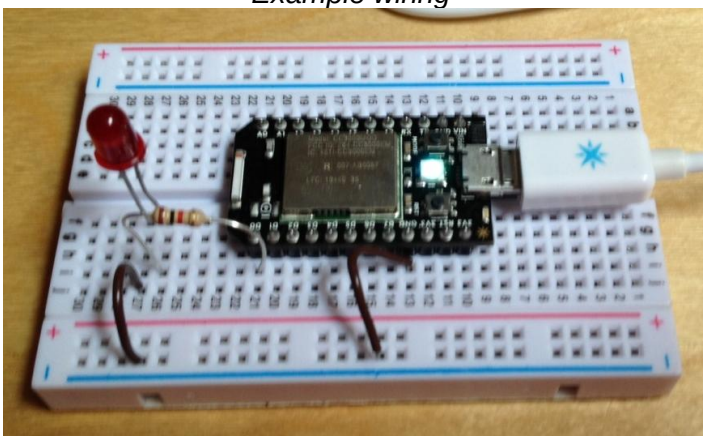
The horizontal rows allow you to electrically connect a given component to any other component placed on that same row (up to 5 components), by inserting pins into the plastic pin holes. It is important to note that the 5-pin rows on each side are not connected.

The vertical strips on the left and right are called power rails and are connected across all rows. This gives the builder easy access to a power source. The power strips on both sides are also not connected.



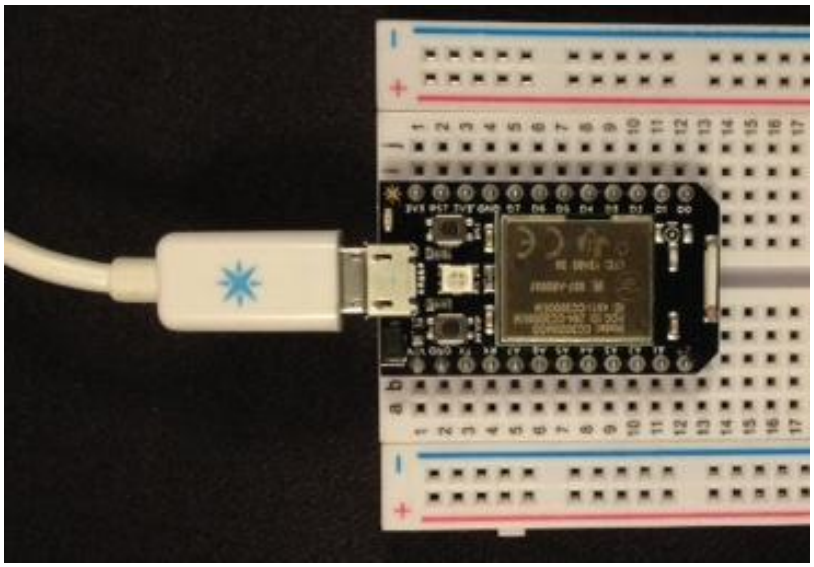
"Connecting wires" are easily inserted into the plastic peg holes, creating circuits without soldering.

Example wiring



2. HARDWARE: Connect your SparkCore to power

2.1. Disconnect SparkCore from breadboard (gently wiggle it out) and reconnect as shown below, such that the Spark core's A0 pin aligns with the breadboard's row 12.



2.2. Connect SparkCore to power supply (connect the USB to a computer or plug included in the BOM)
- When powered on you should see a blue "breathing" light

2.3. Do not push the two small black buttons on the SparkCore at any point (this resets the base software)

3. SOFTWARE: Upload initial software package

The Spark OS has a distributed operating system, where servers in the Cloud do the heavy lifting. That way the product can handle complex, intelligent behaviors when it's online, even with only kilobytes of memory.

4.1. Go to https://www.spark.io/build and login using the following credentials and the number that was written on your Spark's box

4.2. Name your Current App "ThermostatApp"; press enter

4.3. Now add a custom library of code to simplify our program.
- Click the Libraries icon (banner-icon 4th up from the bottom left)
- Type Therm
- Click Therm_LED_Library
- Scroll down and click "Include in App" button

Now we want to add code to our device. In the spirit of open source software, we've written most of the code for you.

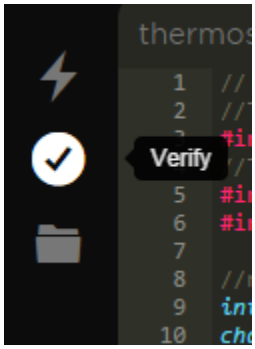
4.4. Go to Step1.cpp and copy the entire block of code

4.5. Return to the Spark build website and paste code into your "Current App" replacing all existing code
- The top of the window where you are pasting code should say "thermostatapp.ino"

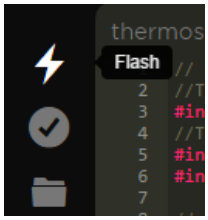
4.6. In line 9 of the code, change your CoreID to match the number written on the Spark's box (e.g. 7)

4.7. In line 10 of the code, change your TeamName to something creative and unique! (This will appear on the leader board projected in the room)
- E.g. char *teamName = "**Witty Team Name**"; //max length 20 characters!!!

4.8. Click the verify logo (second down from top) to confirm your code compiles correctly; you should see "Code Verified! Great work" message on bottom right



4.9. Click the flash button (see image below) to send your new code to your Spark (this will take a moment)
- The light on the spark chip will go through a series colors and blinking (it will first flash purple), before returning to blue "breathing"



Color code:
- Flashing green: Connecting to Wi-Fi network
- Flashing cyan: Connecting to Spark Cloud
- Breathing cyan: Connected to Spark Cloud
- Flashing purple: Updating firmware

!!! CHECK-POINT #1 !!!

Congrats, you've earned your first points!

In the next 60 seconds, the dashboard projected in the room should show that your team has completed this step.

---

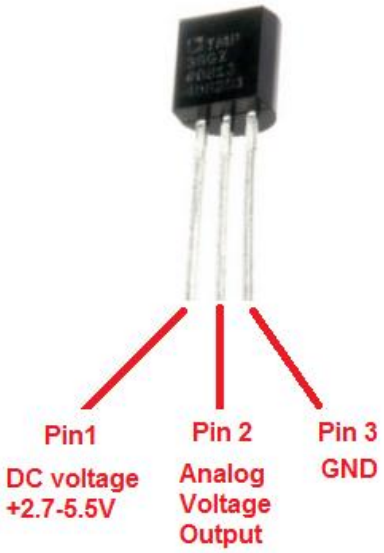
5. HARDWARE: Connecting temperature chip to your breadboard
You're now ready to connect your first component to your breadboard!

5.1. Connect the breadboard's (-) power rail to SparkCore's GND connection point using a connecting wire

5.2. Connect the breadboard's (+) power rail to SparkCore's 3V3* connection point using a connecting wire (make sure it is 3V3*, not 3V3), (hint the wire will arc over the chip)

5.3. Insert temperature sensor's three pins into breadboard's B28, B29, B30

5.4. Connect temperature chip's Pin1 and Pin3 to appropriate power rails using connecting wires (see graphic below)



5.5. Insert capacitor (see BOM) connecting breadboard's ground power rail on A-E side of board, to breadboard's A5

5.6 . Using a connecting wire, connect Pin2 of temperature chip to spark core's A7 (your connecting wire should be placed into breadboard's B5)

You now need to sync with the software team to update the device's software and read the temperature! They will need to know what Spark pin you have connected the temperature sensor to.

!!! CHECK-POINT #2 !!!

In the next 60 seconds, the dashboard projected in the room should show that your team has completed this step.

---

6. SOFTWARE: Reading temperature from device

You're now ready to wirelessly flash new software to your device and read the temperature!

6.1. Go to Step2.cpp and copy the block of code

6.2. Return to the Spark build website and paste code into your "Current App" replacing all existing code
- Recall, the top of the window where you are pasting code should say "thermostatapp.ino"

6.3. In the code, change the following (prior changes are lost when new code updated, sorry!)
- Line9: Change your CoreID to match the number written on your Spark's box (e.g. 7)
- Line10: Update your TeamName again

6.4. In line 16 of the code, you need to replace the "xx" with Spark's connection point that the thermostat's output wire is connected to. Ask your Hardware team for this information based on what they completed in step 5. Do not use quotes (e.g. #define THERM_PIN **A3**)

6.5. Click the verify button (second down from top) to confirm your code compiles correctly; you should see "Code Verified! Great work" message on bottom right.

6.6. Click the flash button (see image below) to send your new code to your Spark (this will take a moment)
- The light on the spark chip will go through a series colors and blinking (it will first flash purple), before returning to blue "breathing"

7. SOFTWARE: Mobile Access

Because your device is cloud connected, you can access it from any mobile device. To demonstrate this we've set up a simple mobile website available <insert website location>

7.1. Open the website
7.2. Login using your credentials from your BOM
7.3. Confirm the temperature on your phone your previous/current reading

Congrats! You've built a WiFi-accessible thermometer! But we can do more!

!!! CHECK-POINT #3 !!!

In the next 60 seconds, the dashboard projected in the room should show that your team has completed this step.

---

8. HARDWARE: Connecting LED Matrices

Now we'd like to be able to display the temperature directly to the device.

8.1. Connect 4-stranded jumper cables to your three LED Matrices (order does not matter, but push hard to ensure connection is made)

8.2. Connect the LED's +/- and negative connection points to your power strips in rows ~21-23 using the jumper cables (Do this for all three LED matrices)

- Look at the top of the LED matrix to match the given jumper cable strand to its connection point.
- You may want to tear back the wires/apart some to give you more room to manipulate

8.3 For each LED Matrix, connect its "C" connection point to row 21 on the A-E side of the breadboard; For "D" connection point, connect to row 22.

8.4. Create a connection between the LED Matrix pin "C" and Spark's D1 (should be breadboard's row 11) using a connecting wire via rows 21.

8.5. Create a connection between the LED Matrix pin "D" and Spark's D0 (should be breadboard's row 12) using a connecting wire via rows 22.

LED trouble shooting: If an LED is not working, disconnect the Spark's power source and reconnect.

You now need to sync with your software team. They will need to know what Spark pins you have connected the LEDs to.

9. SOFTWARE: Display temperature on LED Matrix

We need to update the code to utilize the hardware team's new LED matrices.

9.1. Go to Step3.cpp and copy the block of code

9.2. Return to the Spark build website and paste code into your "Current App" replacing all existing code

9.3. In the code, change the following (prior changes are lost when new code updated, sorry!)

- Line 9: Update your CoreID
- Line 10: Update your TeamName

9.4. Now update Lines 17 and 18 for I2C_D and I2C_C. To update these ask your hardware team which of Spark's PINs they connected to the LED matrices' C and D connection points.

- For example, if the hardware team advised it was connected to A0, the code be: #define I2C_D **A0**

9.5. Click the verify logo (second down from top) to confirm your code compiles correctly

9.6. Click the flash button to send your new code to your Spark (this will take 15-30 seconds and the devices will flash purple)

!!! CHECK-POINT #4 !!!

In the next 60 seconds, the dashboard projected in the room should show that your team has completed this step.

---

10. HARDWARE: Wire buttons to set temperature

Next we want the ability to set the temperature vs. read it. We'll add two buttons to do this.

10.1. Connect the breadboard's other power strips (the F-J side) to Spark's GND and 3V3* connection points using connecting wires.

10.2 Unscrew nut from bottom of push button; pass wires through front of 3D printed mounting board; re-thread nut (place blue-LED to left; Red to right)

10.3. Connect the black wires from both push buttons to the ground power rail (suggest rows 29/30 on F-J side)

10.4. Connect the red power wire directly to breadboard holes…(**DO NOT CONNECT TO POWER RAIL**)

- For blue button, connect to breadboard i7 (which should connect to Spark's D5 connection point)
- For red button, connect to breadboard i6 (which should connect to Spark's D6 connection point)

10.5. Find the wire on each button that is closest to the black ground wire (color should be white or blue), and plug it into the + power rail (suggest ~rows 15/16 on F-J side)

10.6. Now we need to connect a resistor to the board to limit the flow of electricity through the circuit (this is known as a pull down resistor, but we don't need to get into the details).

- Find the (2) identical resistors in your BOM with the black stripe
- Plug one end of the resistor into the ground power rail (F-J side) and the other end into row 24 (rows F-J); with the second resistor, do the same between the ground power rail and row 27.

10.7. Complete the circuit using the button, resistor, and spark

- For the blue push button, take the remaining unattached wire and connect to breadboard's row 27 (F-J side)
- For blue push button, use a connecting wire to complete the circuit to Spark's D2 (should be at breadboard's row 10) connection point from breadboard's row 27

- For the red push button, take the remaining unattached wire and connect to breadboard's row 24 (F-J side)
- For red push button, use a connecting wire to complete the circuit to Spark's D3 (should be at breadboard's row 9) connection point from breadboard's row 24

11. SOFTWARE: Set temperature

We need to update the code to utilize the hardware team's new push buttons.

11.1. Go to Step4.cpp and copy the block of code

11.2. Return to the Spark build website and paste code into your "Current App" replacing all existing code

11.3. In the code, change the following (prior changes are lost when new code updated, sorry!)
- Line 9: Update your CoreID
- Line 10: Update your TeamName

11.4. Click the verify logo (second down from top) to confirm your code compiles correctly

11.5. Click the flash button to send your new code to your Spark (this will take a moment)

!!! CHECK-POINT #5 !!!

In the next 60 seconds, the dashboard projected in the room should show that your team has completed this step.

---

12. HARDWARE: Connect motion sensor device

Lastly, we want to add a motion sensor so we can introduce logic based on the activity around the device.

12.1. Connect motion sensor to 3–stranded jumper cable (you may want to tear back the wires to give you more room to manipulate them)

12.2. Thread the jumper cables through the front of the 3D printed mounting board.

12.3. Connect the motion sensor's "VDD" point to power (+) on the breadboard's F-J side (suggest row ~5)

12.4. Connect the motion sensor's "GND" point to ground (-)on the breadboard's F-J side (suggest row ~5)

12.5. Find the remaining resistor (Red-Red-Brown-Gold) and connect breadboard's G15 and G17

12.6 Connect the remaining jumper cable strand (motion sensor's "Out") to F15

12.7 Use a connecting wire to connect breadboard's F17 to Spark's D4 (should be breadboard's row 8)

Congrats, your hardware is now fully wired! You need to speak with your software team to tell them where you've wired the motion sensor.

13. SOFTWARE: Motion sensor

We need to update the code to utilize the hardware team's motion sensor.

13.1. Go to Step5.cpp and copy the block of code

13.2. Return to the Spark build website and paste code into your "Current App" replacing all existing code

13.3. In the code, change the following (prior changes are lost when new code updated, sorry!)
- Line 9: Update your CoreID
- Line 10: Update your TeamName

13.4. Now update Line 25 with the Spark pin that the hardware team connected the motion sensor to.
- For example, if the hardware team advised it was connected to A0, the code be: #define Motion **A0**

13.4. Click the verify logo (second down from top) to confirm your code compiles correctly

13.5. Click the flash button to send your new code to your Spark (this will take a moment)

!!! CHECK-POINT #5 !!!

In the next 60 seconds, the dashboard projected in the room should show that your team has completed this step.

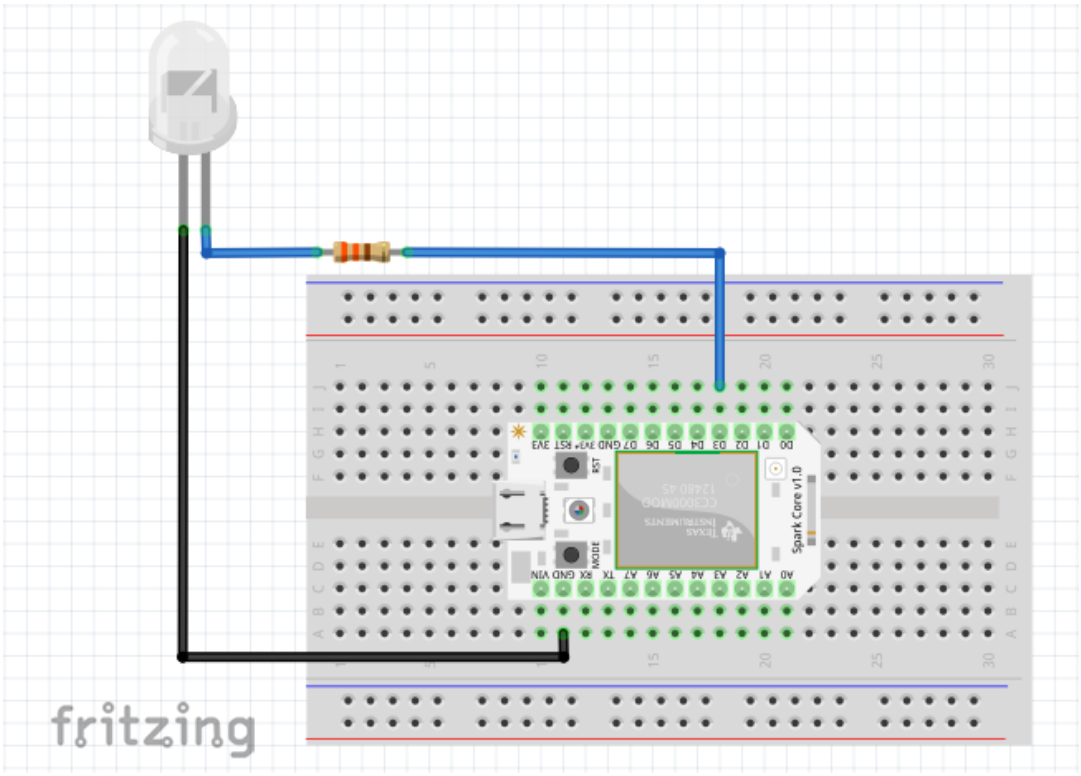
---

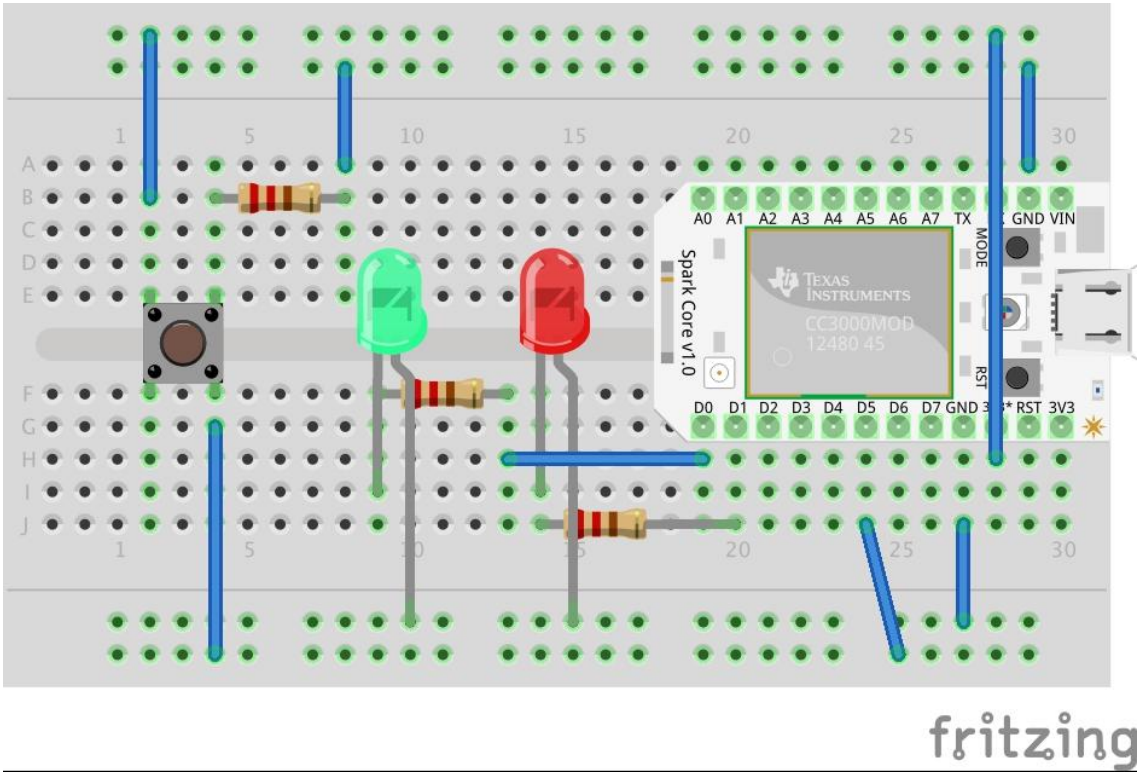
14. Final assembly

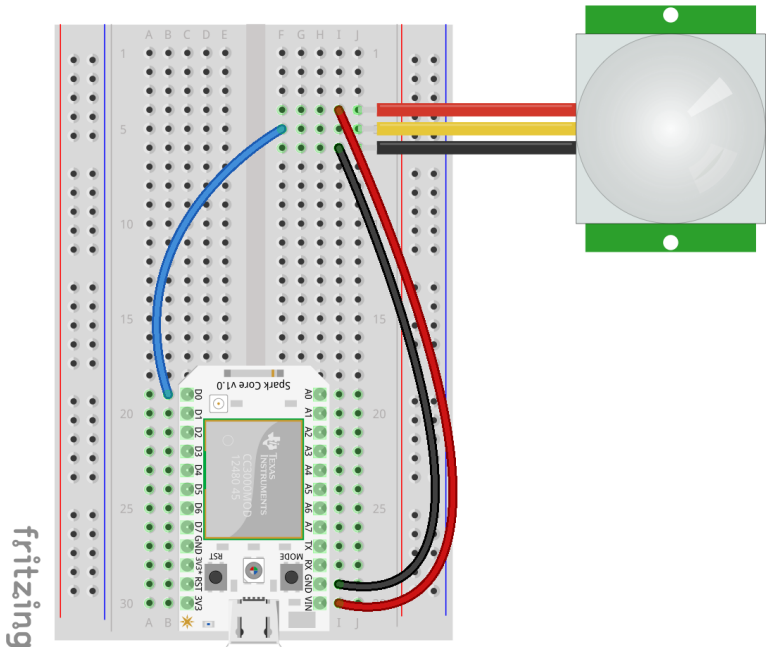
14.1. Use super glue (shared across groups) to fix your LEDs and motion sensor to the mounting board
14.2. Gently bend wires to stand display upright. Take a picture and smile.

The electronics of your prototype could be encased into a 3D printed housing (see sample).

Appendix: Example circuits (not part of this design)






fritzing
Spark Core v1.0
D0 D1 D2 D3 D4 D5 D6 D7 GND 3V3 RST 3V3
A0 A1 A2 A3 A4 A5 A6 A7 TX RX GND VIN
RST
MODE