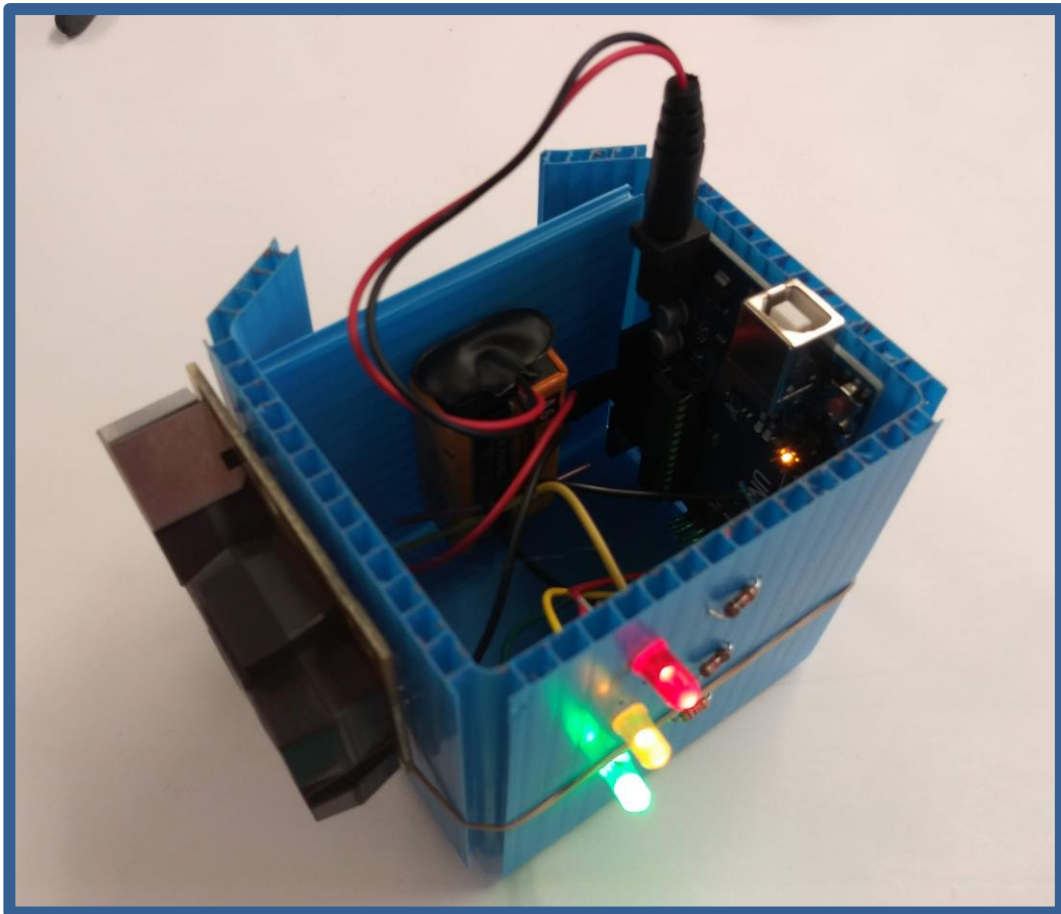


# Arduino Particle Meter

A Do-It-Yourself Guide to Promote STEM Skills and Awareness



## Summary

Introduce students to real world data collection by measuring particulates in the air with a handheld device that they make themselves. Code for this Arduino-based device is provided, and no soldering is required.

**Authors:** Kathleen Mead Vandiver, [kathymv@mit.edu](mailto:kathymv@mit.edu)  
Diane Brancizio, [dianebr@mit.edu](mailto:dianebr@mit.edu)

**Date:** March 11, 2019

**Revision Number:** 1

Edgerton Center STEM Projects are licensed under a  
Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.  
<http://creativecommons.org/licenses/by-nc-sa/4.0/>

## Why We Use Creative Commons Licensing:

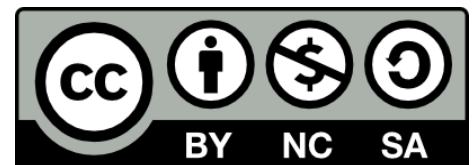
Our mission at the Edgerton Center is to uphold the legacy of Harold “Doc” Edgerton by providing the necessary resources, encouragement, and support for students to gain an education in engineering both in the conventional classroom and particularly outside of it. We promote an overarching ethos of building, learning, and sharing.

As a non-profit entity committed to “building, learning, and sharing” resources for students, we want to make our materials available for free whenever possible, and encourage others to share. Publishing our materials under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](http://creativecommons.org/licenses/by-nc-sa/4.0/) allows us to do this by requiring users and contributors to redistribute at no charge, and by helping others to trace our materials back to their original source.

More information: <http://creativecommons.org/licenses/by-nc-sa/4.0/>

## You are free to use, modify, and share our materials under these terms:

**BY:** Give proper **Attribution** - this requires that when you use or share our materials, you provide the following: the name of the creator (MIT Edgerton Center) and other parties that collaborated or provided funding for the work; a copyright notice; this license notice; a disclaimer notice; a link to the material; and indications of changes that were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.



**NC:** You may only use the material for **NonCommercial** purposes.

**SA:** You may remix, transform, or build upon the material - and if you do, you must **ShareAlike** - distribute your contributions under the same license as the original. When sharing or adapting, there are to be **no additional restrictions** — you may not apply legal terms or technological measures that legally restrict others from doing anything the original license permits.

### In your advertising materials:

**You may** say that your program uses curriculum developed by the MIT Edgerton Center, and where appropriate, provide a link to our home page: <http://edgerton.mit.edu/>.

**You may not** describe your program as a partnership or collaboration with MIT or the MIT Edgerton Center.

Help others to find us by citing the MIT Edgerton Center website, where a variety of curriculum materials are available for download: <http://edgerton.mit.edu/k-12>

# Construct a Particle Meter

## 1. Assembly Instructions with Housing Template

Adapted from the EPA website

[https://www3.epa.gov/airnow/teachers/gh\\_pmsensorkit\\_handoutandinstructions.pdf](https://www3.epa.gov/airnow/teachers/gh_pmsensorkit_handoutandinstructions.pdf)

**Target audience:** middle/high school teachers and students

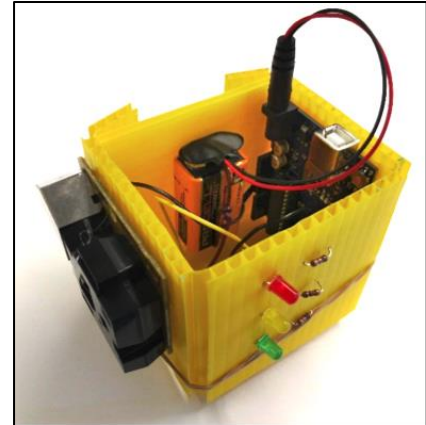
**Estimated Time:** 60 - 90 minutes

### Objective:

Learn the basics about particulate matter (PM) air pollution and how electronic particle meters work

### Materials Provided:

1. Assembly Instructions with Housing Template
2. Technology Notes
3. Arduino Code (separate file) PM\_SensorCore\_EC.ino
4. Lesson Plan Information



Fully Assembled Particle Meter

### Materials Needed:

The list of components is below with a total cost per kit at approximately \$25-\$40.

Electronics components per Particle Meter model:

Items to buy individually (suggested source)	Quantity	Cost
Dust sensor – Shinyei PPD42NS <a href="https://www.seeedstudio.com/s/PPD42NS.html">https://www.seeedstudio.com/s/PPD42NS.html</a>	1	\$12.00
Arduino Uno R3 (or equivalent)	1	\$8 - \$25
9 V battery (local hardware store)	1	\$2.00
9V battery clip with barrel jack	1	\$2
<b>Items to buy in bulk or packs</b>	<b>Quantity</b>	
Hook up wire, solid 22 awg: red, yellow, green, black,+1 other color	~8" of each	
Jumper wires, M-F 8" long: red, yellow, black	1 of each	
LEDs: 1 green, 1 yellow, 1 red	3 LEDs	
1K Ohm resistors	3	

### Housing components per model:

Corrugated plastic sheets, 7" x 11" per kit

Hot glue sticks, low temperature

2 Rubber bands + 1 piece of 8" long solid wire to create a twist tie for the battery.

### Tools (typically shared):

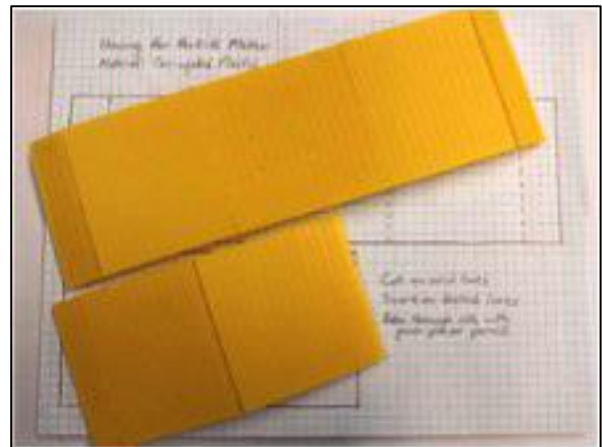
- Needle-nose pliers, 4"
- Wire strippers, 20 – 30 awg
- Wire Cutters, small
- Hot Glue guns, low temperature + glue sticks
- Utility knife
- Cutting surface
- Pushpin or pencil
- Rulers

### Part 1. Program all Arduinos

Download and install the Arduino platform on your computer and upload the custom program (PM\_SensorCode\_EC.ino) to all Arduinos. See details provided in the handout on page 12.

### Part 2. Prepare the housing materials

- Cut housing from template. (See separate page in handout.)
- Lay template over corrugated plastic sheet.
- Tape and cut out OR...
- Transfer outline with push pins and then cut. (To transfer with a push pin you can poke holes into the housing through the template. These holes then mark the lines for cutting and mark the spots for poking holes.)
- Poke holes though the housing with push pin or pencil.

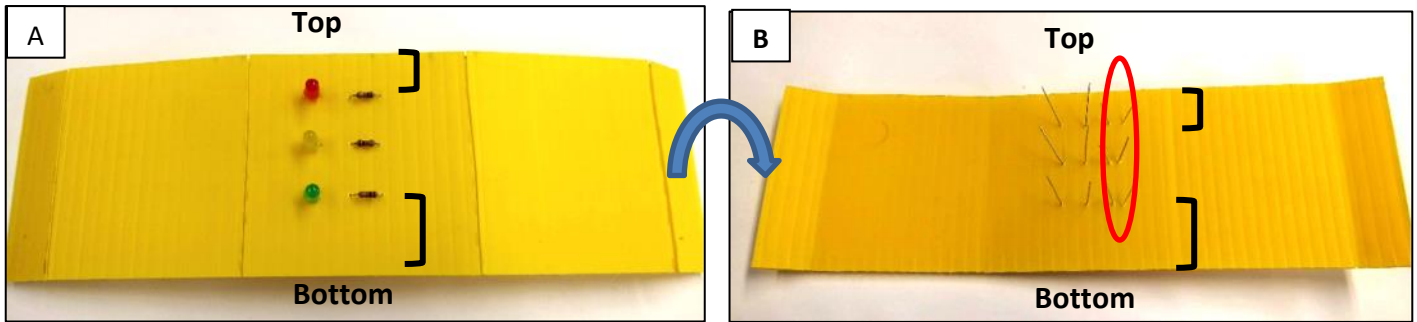


### Part 3. Assemble Electronics on housing

#### Each model requires:

- 1 Arduino
- 1 PM sensor
- 3 short wires-4" – preferably 1 red, 1 yellow, and 1 green
- 1 long wire-8" (preferably black)
- 3 LEDs (red, yellow, green)
- 3 resistors (1K Ohm)
- Jumper wires, M-F 8" long: red, black, yellow

**Step 1.** Insert 3 LEDs in order as shown. (Insert the longer LED wire to the left)  
 Insert 3 resistors into the holes on the housing as shown in Photo A.  
 Notice the bottom of the housing has a bigger margin. Compare the size of the brackets ].  
 Turn the housing over (Turn over LEFT TO RIGHT.) See Photo B



**Step 2.** Strip the wires as shown in Photo C below. Twist resistor wires and LED wires together with pliers.

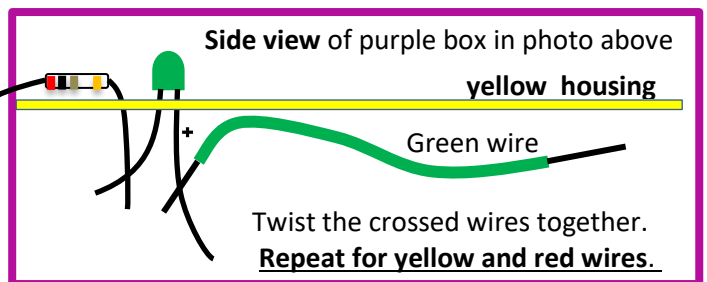
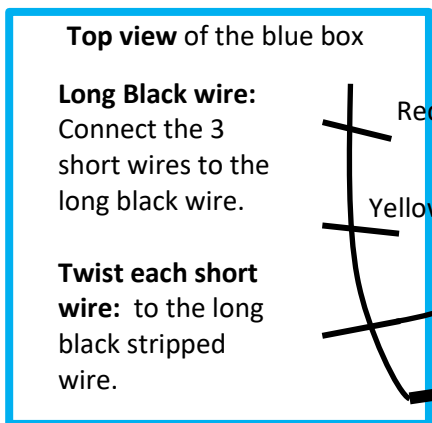
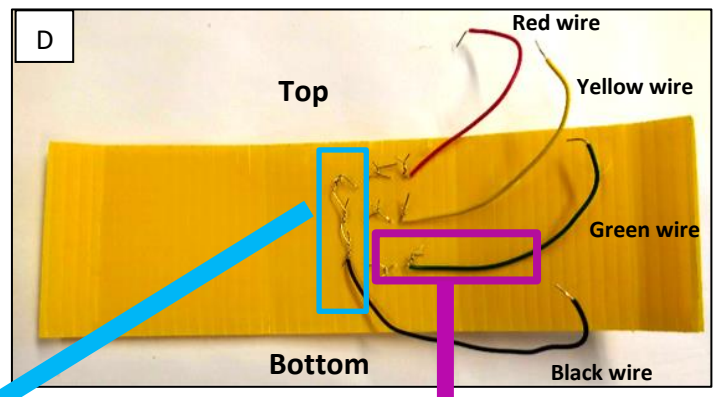
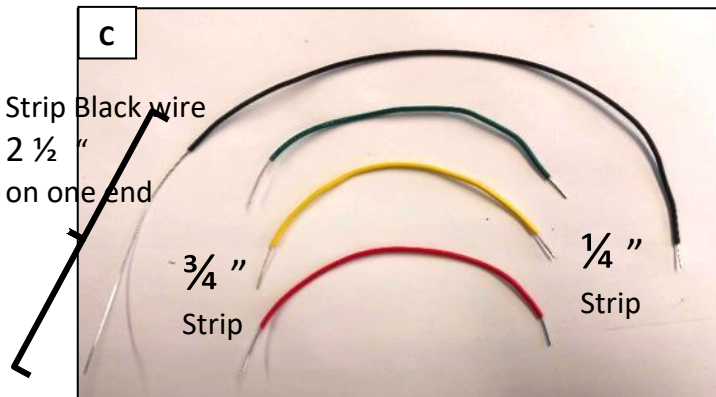
**CONNECTIONS:**

Resistors can be inserted either way into the circuit. (See **Technology Notes** for more info.)

However, LEDs have (+) and (-) wires. The (+) is the longer wire.

Make sure the longer wire (+) of each LED is on the right as shown in orange circle in Photo B above.

Use the needle-nose pliers to twist the wires together tightly as shown in the diagrams below.

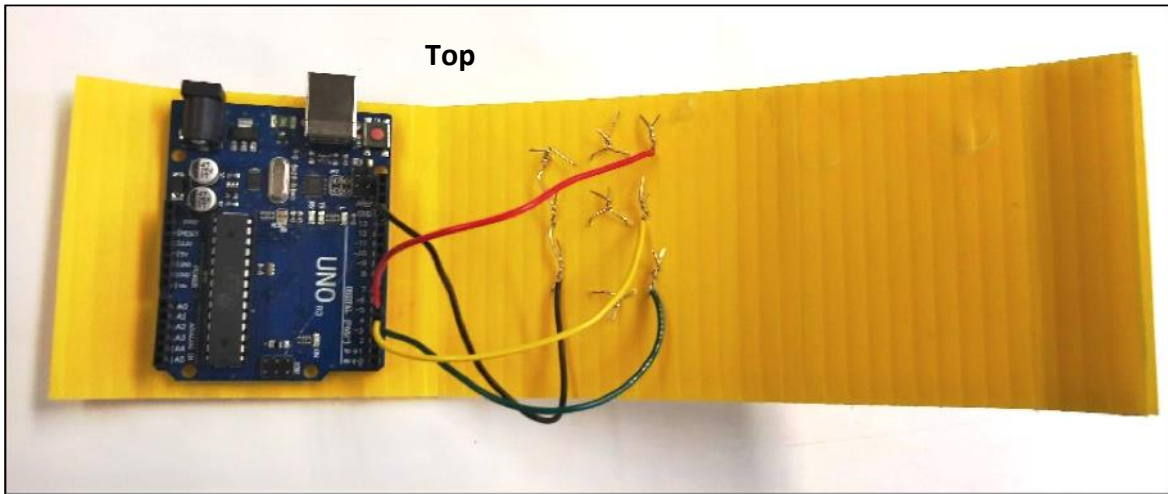


**Step 3.** Assemble the Arduino to the housing with hot glue. Place Arduino as shown.  
Add glue spots to 2 diagonal corners on the back. Use a dime-sized spot of glue on each.

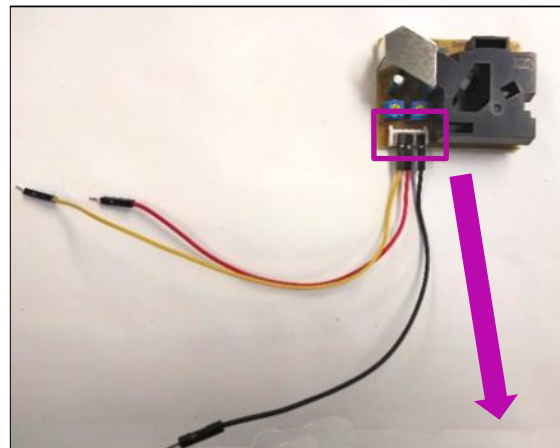
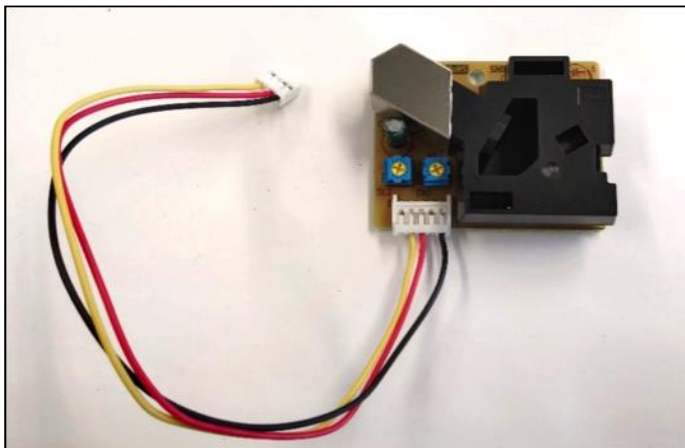
Connect LEDs and resistors to the Arduino by pushing the ends of the wire into numbered pins.

**CONNECTIONS to make:**

- Green wire: Arduino pin 3, Green LED long wire end
- Yellow wire: Arduino pin 4, Yellow LED long wire end
- Red wire: Arduino pin 5, Red LED long wire end
- Black wire: Arduino GRD, Black wire short-end is stripped.



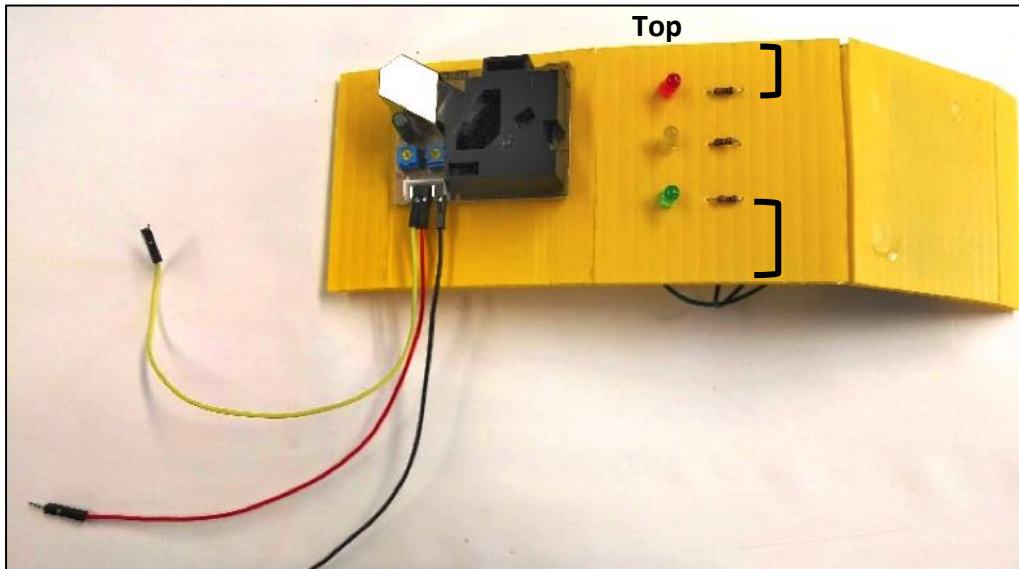
**Step 4.** Pull off the white connector from PM Sensor. (It will not be used.) Replace the white connector with the 3 jumper wires in the colors as shown. The jumpers will slide onto the pins.



**Step 5.** Assemble the PM Sensor to the housing with hot glue.

Locate the PM Sensor towards the TOP of the housing, as shown.

Add glue spots to 2 diagonal corners on the back. Use a dime-sized spot of glue for each.



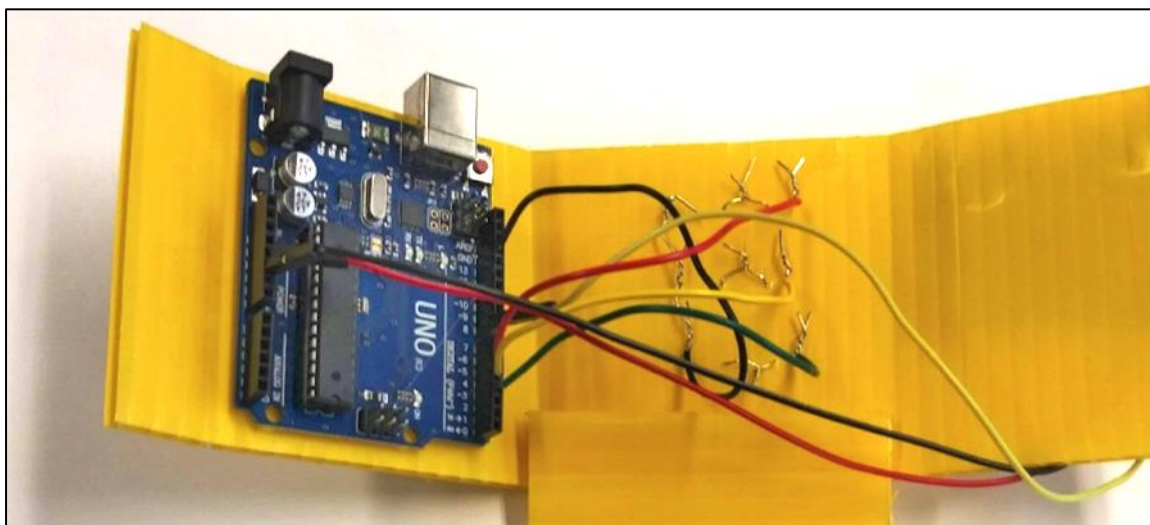
**Step 6.** Connect the PM Sensor wires to the Arduino as shown. (See Arduino photo Page 7 for pins.)

CONNECTIONS:

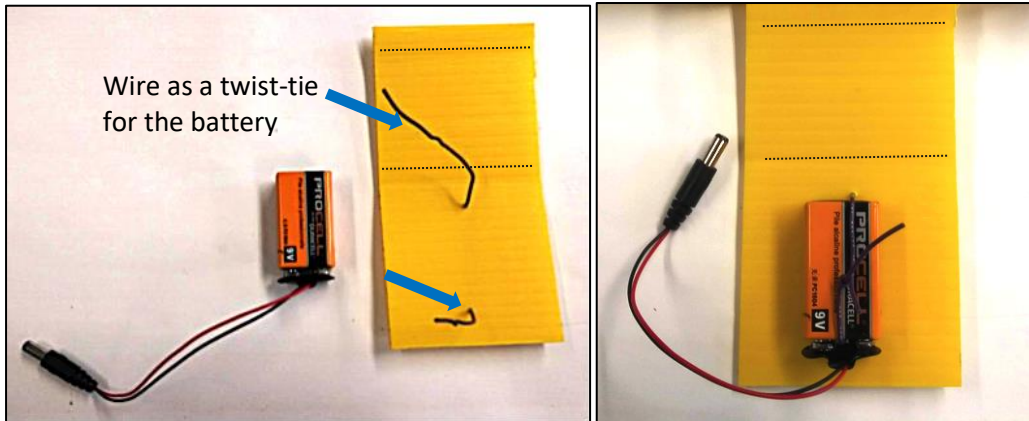
Jumper Yellow wire to Arduino pin 8

Jumper Red wire to Arduino 5V pin

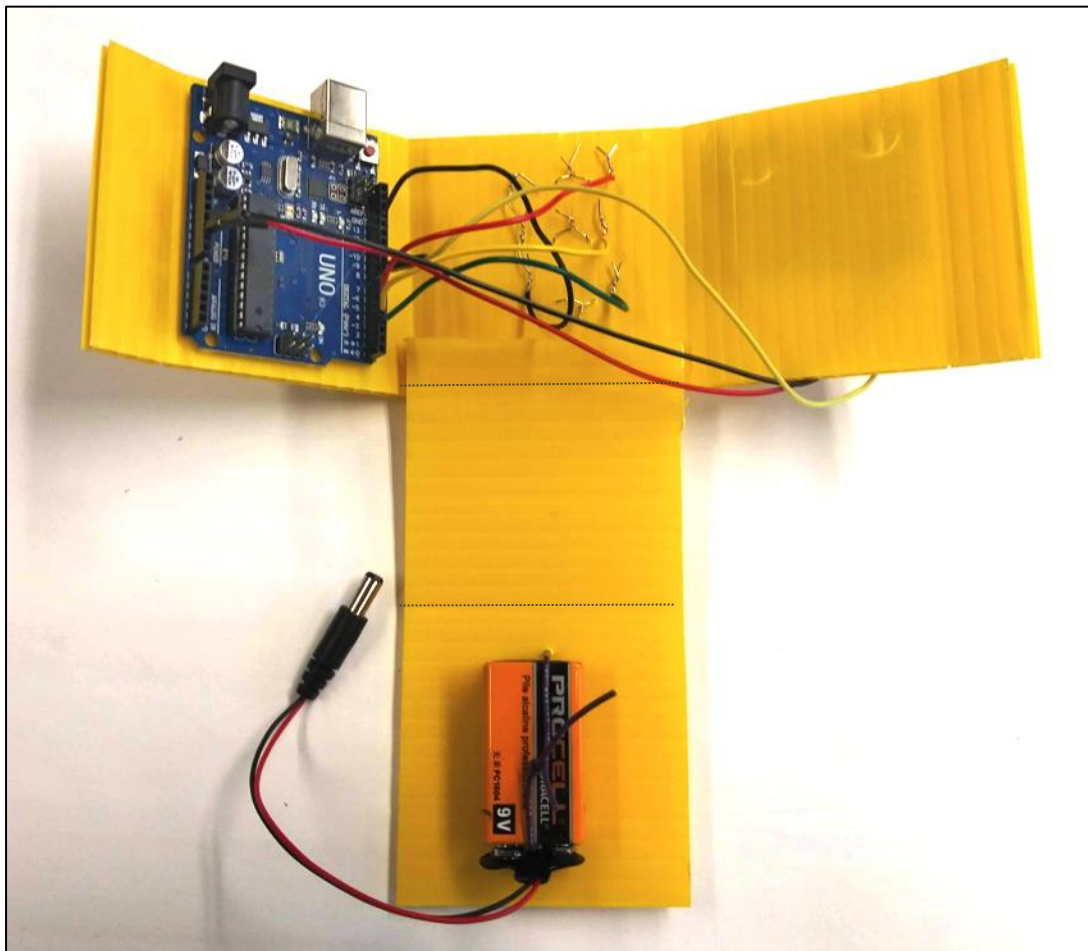
Jumper Black wire to Arduino GND pin



**Step 7.** Assemble the 9-volt battery inside the smaller housing piece. Folds are towards you. Attach the snap connector to the battery. Use a piece of wire as a twist-tie as shown.

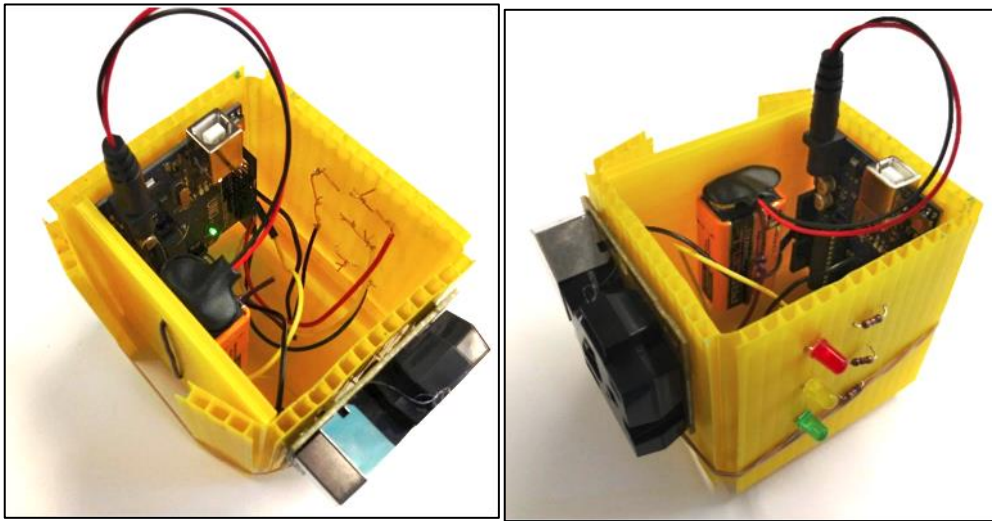


**Step 8.** Assemble the smaller housing piece onto the larger one with hot glue. Be sure to center it on the center panel. Leave room for bending the sides around it as shown. Add glue across the width of the panel. Hold the housing pieces together until glue cools.





**Step 9.** Fold up the housing and fasten with rubber bands. Here are two views:



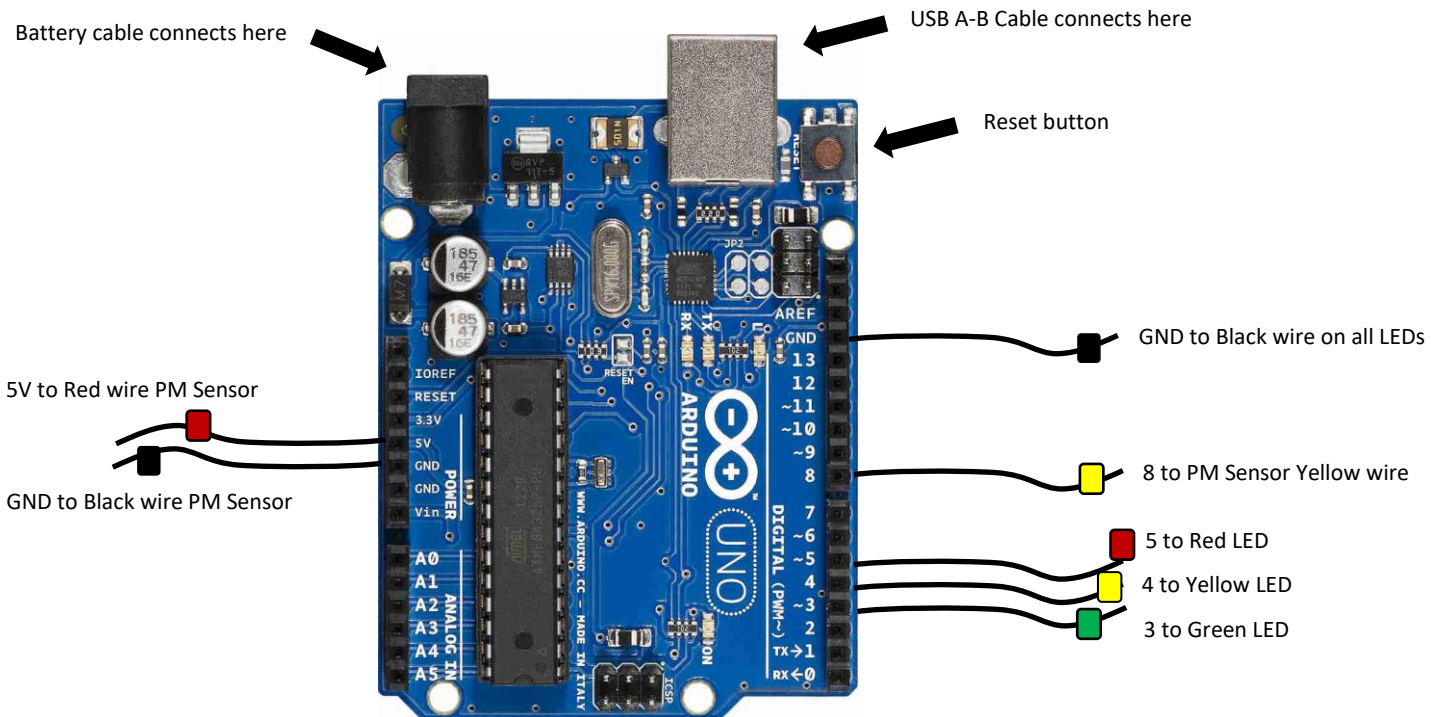
☞ Grasp your PM meter from this side with one hand.

You will be able to explore the air in front of you and watch the LEDs light up.

**Step 10.** Open the Housing and review the wiring again (below).

Power the Arduino by plugging the battery connector into the board as shown above.

All 3 LEDs should blink together two times. Then the green LED will turn on.



## Part 4. Test and Use the Particle Meter

--Disconnect battery cable--

1. Connect your Particle Meter to the Arduino with the USB A-B cable.
2. Open the Arduino IDE and upload the program: PM\_SensorCode\_EC.ino
3. Check that the 3 LEDs blink on and off twice.

This will let you know that the wiring is correct and that all components are working. To check this at any time, hit the “Reset” button on the Arduino to start the program again.

4. Test and calibrate the sensor
  - Let the sensor warm up for 2 minutes.  
Generate particles by shaking pieces of paper, rubbing fingers, etc.
  - While connected to the computer, open the Serial Monitor (Tools → Serial Monitor) and watch the display of data. Make sure the baud rate is set to 9600.
  - Revise the thresholds in the code as needed. The LED thresholds in the code are based on the ratio of time that particles are ‘seen’ and are 3 and 6 (“if (ratio<3)”, “if(ratio<6)”). Try lowering the numbers to make the sensor more sensitive. (More LEDs light up.) Try raising these numbers to make the sensor less sensitive. (Fewer LEDs light up.)
  - Due to age or variation between PM sensors, the code may need to be adjusted.
5. Use the particle meter without the computer.
  - Disconnect the USB and connect the 9v Battery.
  - Try a variety of particle situations!
6. Unplug the 9v battery when finished to avoid running the battery down.

# Construct a Particle Meter

## 2. Technology Notes

Adapted from the EPA website

[https://www3.epa.gov/airnow/teachers/gh\\_pmsensorkit\\_handoutandinstructions.pdf](https://www3.epa.gov/airnow/teachers/gh_pmsensorkit_handoutandinstructions.pdf)

### How the PM sensor works:

(Images: [www.shinyei.co.jp](http://www.shinyei.co.jp))

The images below show the PM sensor. Since warm air rises, a heater causes air to flow through the sensor. This is why the sensor must be mounted vertically. An LED shines light on the particles and the amount of light that is deflected by the particles is measured by a light receptor. This type of particle detection generally sees particles greater than 1 micrometer, which makes the sensor more sensitive to larger particles like dust and less sensitive to very small particles formed in smoke.

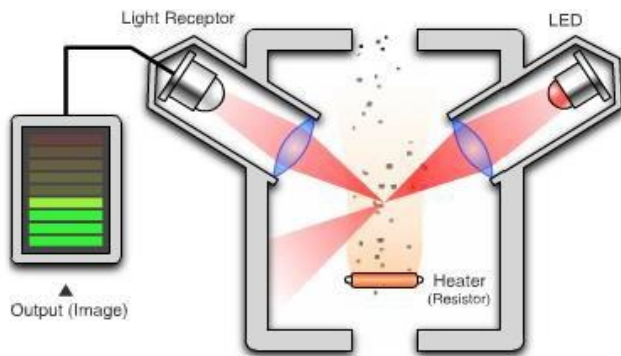


Figure 4. Schematic showing how the particle sensor operates.



Figure 5. Shinyei particle sensor used in the kit.

### Basic Electronics:

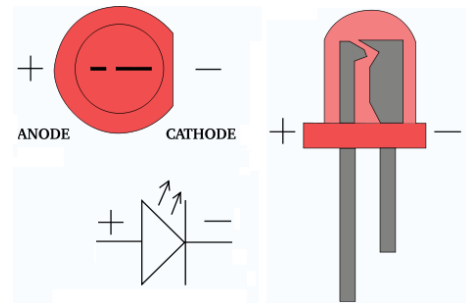
#### Circuit:

“An electric circuit is like a pathway made of wires that electrons can flow through. A battery or other power source gives the force (voltage) that makes the electrons move. When the electrons get to a device like a light bulb, your computer, or a refrigerator, they give it the power to make it work. The word ‘circuit’ sounds like ‘circle,’ and a circuit needs to be circular to work. The wires have to go from the power source to the device and back again, so that the electrons can go out and come back.”

<http://kids.discovery.com/tell-me/curiosity-corner/science/how-do-electric-circuits-work>

## LED:

This is an LED. "Inside each LED is a small bit of chemical that when electrons are passed through, it emits radiation (i.e., light). By changing this chemical compound, you can in effect change the wavelength emitted - **infrared**, green/blue/red, near-ultraviolet, etc.



Be aware that an LED is a diode, meaning that it is polarized. By convention, current can only go from the anode (positive end) to the cathode (ground, or negative end). Note that the two wires on the LED are different lengths. The longer wire is always + and the shorter is -."

(Images: [http://www.societyofrobots.com/electronics\\_led\\_tutorial.shtml](http://www.societyofrobots.com/electronics_led_tutorial.shtml))

## Resistor:

This is a resistor. If there is too much current the LED will glow too brightly and will be damaged. The battery will also run out very quickly. The resistor resists current of electrons resulting in a lower current. Think of water flowing when you think of current. Inserting a resistor is like inserting a smaller pipe in the middle of a river. The stripes on the resistor are a code for the resistance and tolerance values of the resistor. The color code can be interpreted using the chart shown here or by using one of the many apps available online and for smartphones.



**Resistor Color Code**

Color	1 <sup>st</sup> Band	2 <sup>nd</sup> Band	3 <sup>rd</sup> Band	Multiplier	Tolerance
Black	0	0	0	x 1 Ω	
Brown	1	1	1	x 10 Ω	+/- 1%
Red	2	2	2	x 100 Ω	+/- 2%
Orange	3	3	3	x 1K Ω	
Yellow	4	4	4	x 10K Ω	
Green	5	5	5	x 100K Ω	+/- 5%
Blue	6	6	6	x 1M Ω	+/- 25%
Violet	7	7	7	x 10M Ω	+/- .1%
Grey	8	8	8		+/- .05%
White	9	9	9		
Gold				x .1 Ω	+/- 5%
Silver				x .01 Ω	+/- 10%

## Electrical connections:

Components in an electrical circuit must have a good electrical connection. There are many ways to do this including soldering and mechanical screws, and detachable connectors. In this project the wires will be twisted together. When done well, this will work as well as soldering. For long-lasting connections, soldering is preferable and can be done on the twisted connections.

## How the

## Arduino works:

An Arduino is a microcontroller - a device that runs a programmed set of instructions over and over. It is designed to be easy to use and to interface with a variety of electronic sensors and actuators. Users download the Arduino IDE (Interactive Development Environment) on a computer, and create code to upload to the microcontroller. The platform is open-source, which means the code and specifications are open and free to the public, and people are allowed to build and even sell their own versions code created on a computer. In this project it is used here to read data from the particle sensor, interpret it, and display it on the LEDs.



*Useful links if students want to learn more about Arduino:*

*Arduino tutorial videos: search "Jeremy Blum Arduino tutorials" on YouTube*

*Additional assistance with Arduino coding and wiring: <http://forum.arduino.cc/>*

*Tutorials for Arduino: <http://learn.adafruit.com/>*

**The one rule while we are doing this activity is: never adjust the wires on the Arduino or plug anything into the Arduino while it is plugged in to the battery. Plug in the battery when all wires are situated and if you need to change something unplug the battery first.** Arduinos are fairly rugged but if it becomes hot unplug it as this is a sign of a short check for errors in your wiring and plug it back in once the problem is corrected (credit: Arduino.cc).

How the Arduino Code works:

Code has been written and installed on your Arduino to run the PM sensor you will build. Basically the Arduino is checking the particle sensor continuously to see whether it is detecting particles or not. The amount of PM pollution is determined based on the percent of time the sensor is detecting particles. The Arduino then lights up the LEDs based on how many particles are seen. 1 LED for a low number of particles, 2 LEDs for a moderate amount of particles, and 3 LEDs for a high amount of particles. *See code in attachment*

Arduino code has 4 basic parts which can be seen in the provided code.

Header – variable definitions

Setup() – code that runs once and initializes monitor and hardware

Loop() – code that repeats and runs continually while the Arduino is powered


Functions() - code that can be called in setup() or Loop() to do specific tasks

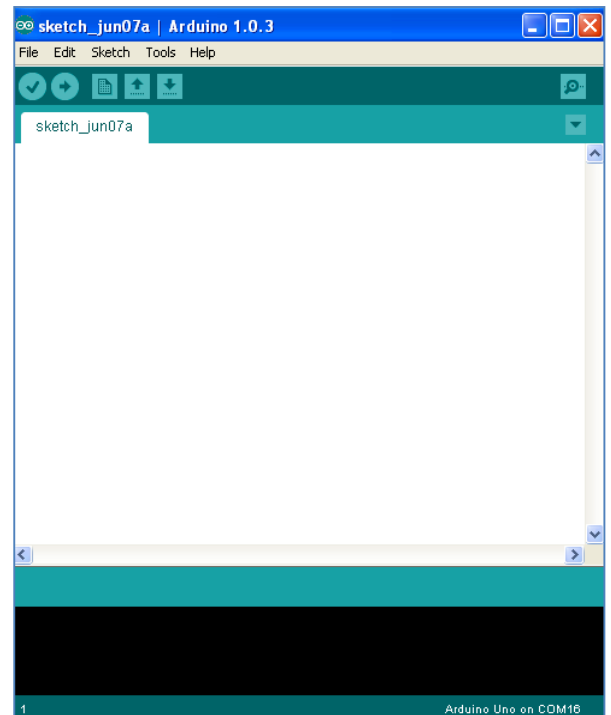
## Arduino Preparation

### Installing Arduino software:

1. Go to <http://arduino.cc/>
2. Click on the download tab
3. Under the Download heading select the program you need based on the type of computer you will be using (Windows, Mac, or Linux).
4. Run the installation software.

Once the Arduino program has been installed on your computer:

1. Save the provided Arduino code (PM\_SensorCode\_EC.ino) saved on your computer in a folder you can easily navigate to.
2. Open the Arduino program either through your desktop icon or through the programs folder on your computer. You should have a sketch open that looks similar to the image on the right.
3. Open the Build your own PM sensor program by clicking on the “File” tab in the navigation menu, then “Open”, and then selecting the program file (PM\_SensorCode\_EC.ino)
4. Plug an Arduino into your computer using the USB cable.
5. Click the  arrow button to upload the code.
6. In the event of an error pop-up, two things to check:
  - a. If the correct “COM” port is not selected you will get an error. You will need to change the COM port either through the error message that pops up or click on the “Tools” tab and then click on “Serial Port”.
  - b. Make sure that you have the correct board selected under the “Tools” tab and board “Arduino Uno”.



7. Once you click the arrow button to upload the code and it uploads the code successfully, check that the code is running by clicking on the “Tools” tab, then selecting “Serial Monitor”. After a few seconds you should see "Starting PM Code" and a particle ratio should appear every 5 seconds.
8. Once you have verified that the code has been uploaded unplug the Arduino.
9. Repeat steps 5-8 for all Arduinos

In addition, it is recommended that you assemble one full particle sensor kit after uploading the code to make sure it is all functioning properly. To verify the wiring and the components, the EC code is set up to blink all 3 LEDs on and off twice. This will happen whenever the Arduino is powered and whenever the Reset button is pushed.

# Construct a Particle Meter

## 3. Arduino Code

Adapted from the EPA website

[https://www3.epa.gov/airnow/teachers/gh\\_pmsensorkit\\_handoutandinstructions.pdf](https://www3.epa.gov/airnow/teachers/gh_pmsensorkit_handoutandinstructions.pdf)

PM\_SensorCode\_EC.ino

```
/* PM sensor code
based on code from http://wiki.seeedstudio.com/Grove-Dust_Sensor/
and EPA "Build Your Own Particle Sensor" outreach activity
shared by MIT Edgerton Center, rev 7/25/2018 Kathleen Vandiver, Edgerton Center K12 Advisor, kathymv@mit.edu
*/

// Pin definitions
int PM = 8;
int LED1=3;
int LED2=4;
int LED3=5;

//variables
unsigned long duration; //measured in microseconds
int sampletime_ms = 30; // 30 millisecond between reads of sensor
float ratio = 0; // ratio = duration/(sampletime*1000)
float concentration = 0; // from datasheet

int moderatelevel = 3; // when ratio > 3%, show moderate level
int highlevel = 6; // when ratio > 6%, show high level

void setup()
{
  Serial.begin(9600); //initialize Serial Monitor
  Serial.println("Starting PM code");
  pinMode(LED1,OUTPUT);//Define the LEDs as outputs
  pinMode(LED2,OUTPUT);
  pinMode(LED3,OUTPUT);
  pinMode(PM,INPUT);
  checkLEDs();
}

void loop()
{
  duration = pulseIn(PM, LOW);
  ratio = duration/(sampletime_ms*1000); // Integer percentage 0=>100
  concentration = 1.1*pow(ratio,3)-3.8*pow(ratio,2)+520*ratio+0.62; // using spec sheet curve
  Serial.print("pulse: ");
  Serial.print(duration);
  Serial.print(", ratio: ");
  Serial.print(ratio);
  Serial.print(", concentration: ");
```



```

    Serial.println(concentration);
    delay(sampletime_ms);

    if (ratio < moderatelevel) //if "low concentration" PM light up 1 LED
    {
        //- numbers selected based on experimentation not actual concentration values
        Serial.println("Low concentration");
        digitalWrite(LED1, HIGH);
        digitalWrite(LED2, LOW);
        digitalWrite(LED3, LOW);
    }
    else if (ratio < highlevel) //if "moderate concentration" PM light up 2 LEDs
    {
        Serial.println("Moderate concentration");
        digitalWrite(LED1, HIGH);
        digitalWrite(LED2, HIGH);
        digitalWrite(LED3, LOW);
    }
    else //"high concentration" light up all 3 LEDs
    {
        Serial.println("High concentration");
        digitalWrite(LED1, HIGH);
        digitalWrite(LED2, HIGH);
        digitalWrite(LED3, HIGH);
    }
    Serial.println(); // newline
}

/***** functions *****/
void checkLEDs()
{
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, HIGH);
    delay(1000);
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    delay(1000);
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, HIGH);
    delay(1000);
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    delay(1000);
}

```

# Housing for Particle Meter

Material: Corrugated Plastic or Cardboard, ~.12" thick

