Funded by
the European Union

PROJECT “CONNECTING STEM WITH ART”

2021-1-LT01-KA210-SCH-000031296

STEAM WORK BOOK

IOT DEVICES, SINGLE BOARD COMPUTERS AND PRINCIPLE OF COMPUTER MEASUREMENT AND MEASRUMENT OF NON – ELECTRICAL AND ELECTRICAL QUANTITIES

The Internet of things (IoT) describes the network of physical objects - a.k.a. "things" - that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet.

Things have evolved due to the convergence of multiple technologies, real-time analytics, machine learning, commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the "smart home", including devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smartphones and smart speakers. The IoT can also be used in healthcare systems.

A growing portion of IoT devices are created for consumer use, including connected vehicles, home automation, wearable technology, connected health, and appliances with remote monitoring capabilities. IoT devices are a part of the larger concept of home automation, which can include lighting, heating and air conditioning, media and security systems and camera systems. Long-term benefits could include energy savings by automatically ensuring lights and electronics are turned off or by making the residents in the home aware of usage.

The basic hardware components for SBC (Single Board Computers) have a GPIO connectors (inputs and outputs for general purposes) - pins that allow connection various input and output measuring and control devices such as sensors and actuators.

Sensors are devices for measuring non-electrical physical quantities or signals. Actuators are output devices such as motors, heaters, light bulbs, alarms, sirens and similarly. Actuators, unlike sensors, often require more energy for activation and operation, which is why they are connected via special devices and power sources.

SBC is a complete computer built on one board, with microprocessors, memory, input / output (I / O) and other functions required by functional computer. Sensors and actuators enable interaction with the environment in order to perform the necessary functionalities that include measurement and control.

General purpose computers - PCs, laptops, tablets, phones do not have these connectors. By connecting the appropriate expansion cards to computers in general for this purpose, it is possible to connect sensors and actuators, similar to what is possible directly to SBC, but that significantly increases the price of such systems and the elegance of use due to significant dimensions and consumption. This is not a good choice for continuous operation because it is done about general purpose computers.

SBCs are computers of small size, price, consumption and modest resources that are just right intended to be dedicated to one particular continuous function. We used the programmable Arduino nano SBC card shown in figure 1 which is programmed by a programming language based on the Wiring language (syntax and library), which is basically similar to the C ++ programming language with some simplifications and changes.

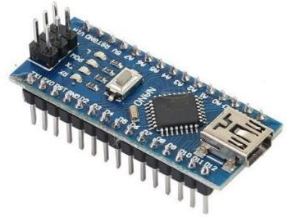
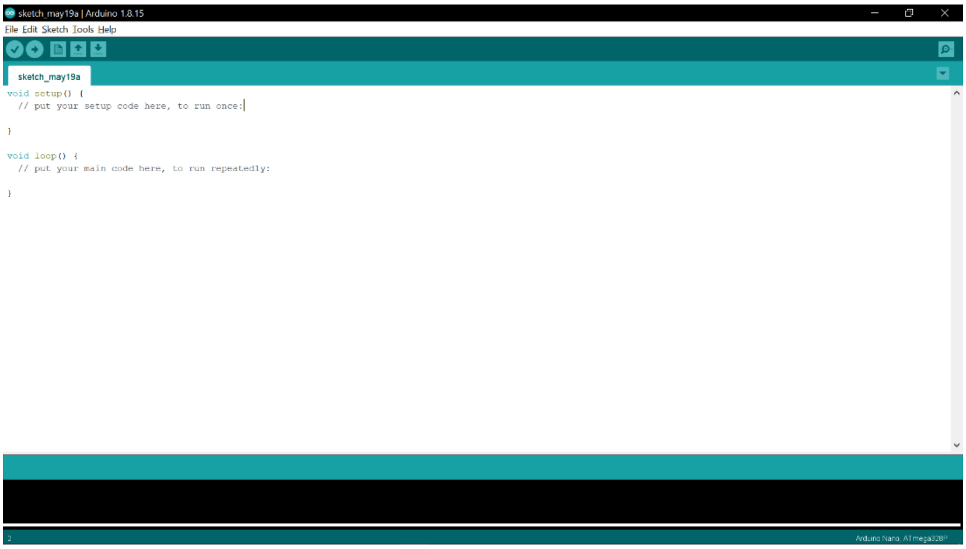


Figure 1. Arduino nano programmable card.

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online.



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Arduino Nano, ATmega328P

You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments.

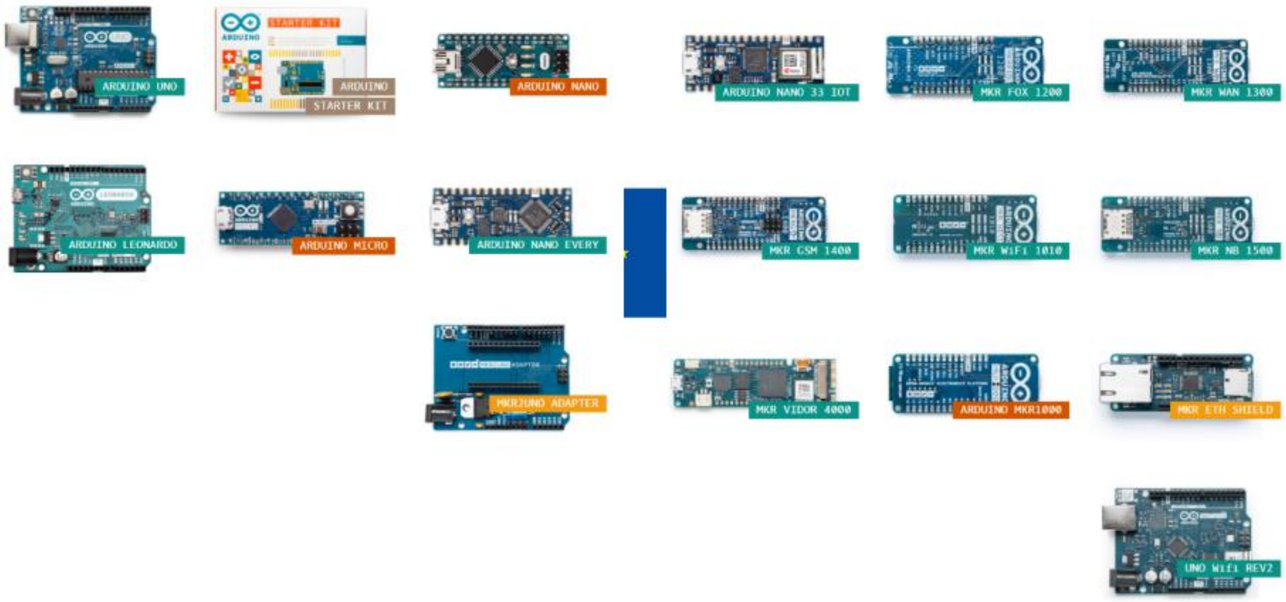
All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users.

It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics.

Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example.

Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.



There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package.

Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

- Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms.

- Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

- Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

- Open source and extensible software - The Arduino software is published as opensource tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.

- Open source and extensible hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

Physical quantities that characterize an environment or environment are mostly analogous and non-electrical in nature. In order to measure such quantities with the help of a computer, it is necessary to first translate them into an electrical signal.

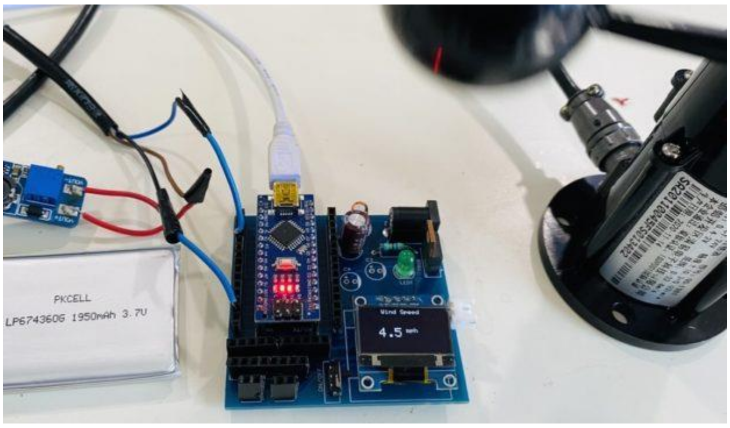
The conversion of non-electrical quantities into an electrical signal is done by means of devices called sensors and in this paper they are of utmost importance since we are working with non-electric quantities. The principle of operation of the sensor is based on various physical phenomena that enable conversion given non-electric quantities in electric voltage.

Every physical phenomenon is mathematically described using equations which create a relationship between the measured output voltage and the value of the non-electric physical quantity. Based on the measured value of the electrical signal (voltage), a value can be calculated measured non-electric quantities with some measurement error.

DEVICE FOR WIND SPEED MEASURING

In this project, we will learn how to how to measure Wind Speed using Adafruit Anemometer Sensor & Arduino. An anemometer is a device used for measuring wind speed and direction. It is also a common Weather Station instrument.

In all studies that involve wind speed measurement such as meteorology, wind turbines & agriculture, we need accurate wind speed information for decision making. There are several types of anemometers with medium and high costs such as cups, hot wire & pitot tubes. The anemometer used here is the cup-type anemometer.

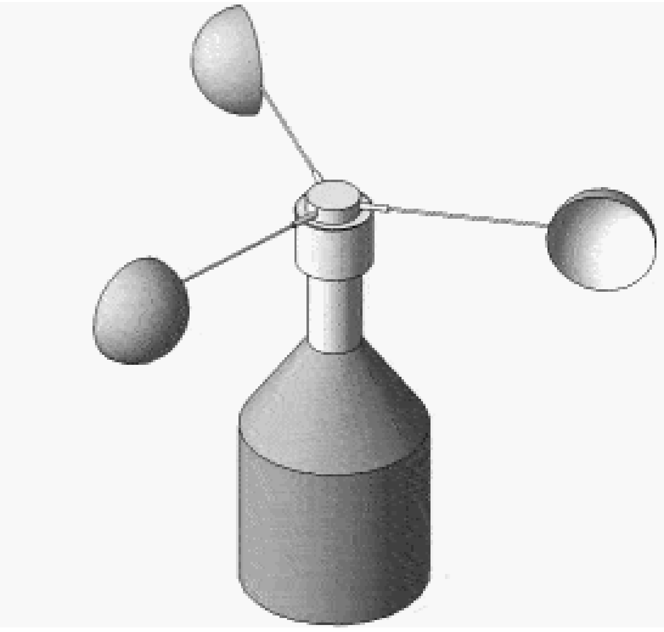


The anemometer sensor we are using here is the Adafruit anemometer. The Adafruit anemometer is capable of measuring wind speeds up to **70 m/s** or **156 mph** which should be adequate for our location. We can interface the **Adafruit Anemometer Sensor with Arduino** and OLED Display.

The sensor will measure the wind speed in m/s and displays the value on **OLED Screen**. You can convert the wind speed from **m/s** to **miles per hour** or **kilometer per hour**. Following are the components that we need for making **Wind Speed Measurement Project** or **Arduino Anemometer Project.**

	Components Name	Quantity
1.	Arduino Nano Board	1
2.	Adafruit Anemometer Sensor	1
3.	0.96" I2C OLED Display	1
4.	MT3608 Boost Converter Module	1
5.	3.7V Li-Ion Battery	1

An anemometer is a device used for measuring wind speed and direction. It is widely used for measuring the speed of **airflow** in the atmosphere, in wind tunnels, and in other gas-flow applications. The term is derived from the Greek word **anemos**, which means wind, and is used to describe any **wind speed instrument** used in **meteorology**.



The most widely used Anemometer for **wind-speed measurements** is the revolving-cup electric anemometer. The revolving cups drive an **electric generator**. The output of the generator operates an electric meter that is calibrated in wind speed. **Three-cup anemometers** are currently used as the industry standard for wind resource assessment studies & practice. You may use Ultrasonic Anemometer if you don’t want movable parts.

The cup-type anemometer consists of 3 or 4 **hemispherical cups** mounted on horizontal arms, which were mounted on a **vertical shaft**. The airflow past the cups in any horizontal direction turned the shaft at a rate that was roughly proportional to the **wind speed**. Therefore, counting the turns of the shaft over a set time interval produced a value proportional to the **average wind speed** for a wide range of speeds.

Funded by the European Union

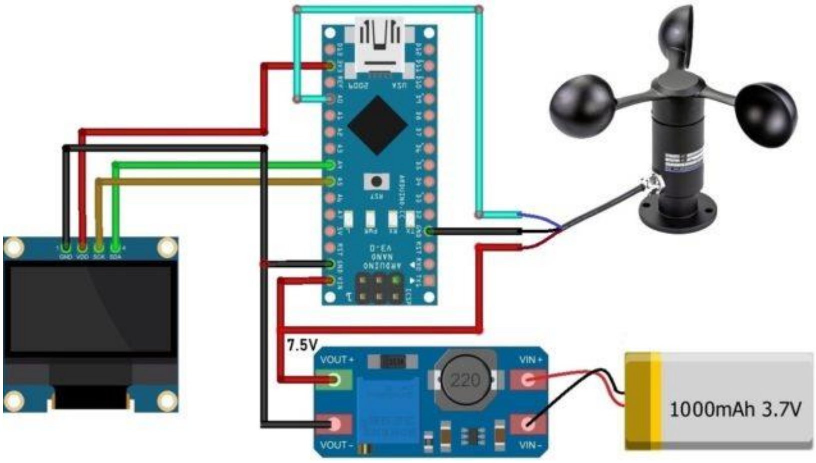
Adafruit Anemometer

The **Adafruit Anemometer Sensor** is a Three-cup type anemometer that is capable of measuring wind speed up to **70m/s or 156mph**. It is composed of a shell, the wind cup, and the circuit module.

The sensor is supplied with a length of **3 core cables** and three connections. A **black wire** to power and signal ground, a **brown wire** for power which can be anything from **7-24v DC**, and a third **blue wire** that provides measurements via **analog voltage**. The output analog voltage will range from **0.4V (0 m/s wind)** up to **2.0V (for 32.4m/s wind)**. The internal system can work normally in **– 20 ℃ ~ + 50 ℃**, humidity **35% ~ 85%**. The plug of the cable is a **military plug**. It has a good anticorrosive and prevents erosion performance that it can ensure the instrument used for a long time.

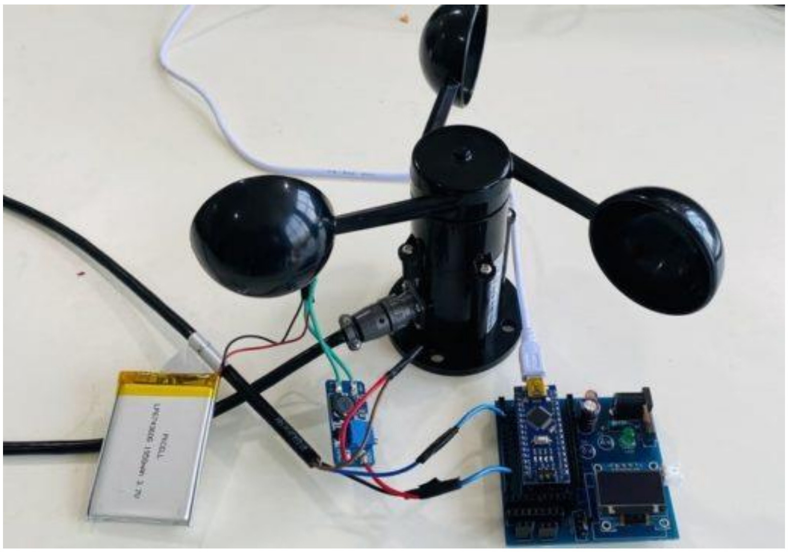
Measure Wind Speed using Anemometer and Arduino

Now let us interface the Adafruit Anemometer Sensor with Arduino & measure the wind speed. Check the schematic or circuit below.



The Adafruit Anemometer works between **7-24V DC**. So the voltage from the Arduino is not enough to power on the sensor. Hence I have to use MT3608 **DC-to-DC Boost Converter Module** to boost the **3.7V from Lithium-Ion Battery** to 7.5V. To adjust the output voltage, connect the battery first to the input of the MT3608 Module and then rotate the potentiometer until the output voltage shows 7.5V on multimeter.

The Sensor VCC wire and the Arduino Vin pin are supplied with **7.5V** from the output of the Boost Converter Module. The blue wire that is the analog output pin of the anemometer sensor is connected to the **A0** of Arduino. Similarly the **I2C Pin** of OLED Display, i.e **SDA & SCL** is connected to **A4 & A5** of Arduino. The OLED Display is supplied with **3.3V** from Arduino 3.3V Pin.



Program for Arduino Anemometer

Here is a simple **Arduino Anemometer code** written for Arduino IDE. The Arduino anemometer project code can be compiled and uploaded to the Arduino Board. You can either change the wind speed to m/s or mph.

The code requires **SSD1306 & GFX OLED library** for compilation. First download the following libraries and add it to the Arduino IDE.

Download SSD1306 Library: https://github.com/adafruit/Adafruit_SSD1306

Download Adafruit GFX Library: https://github.com/adafruit/Adafruit-GFX-Library

You can now copy the code and upload it to the Arduino Board.

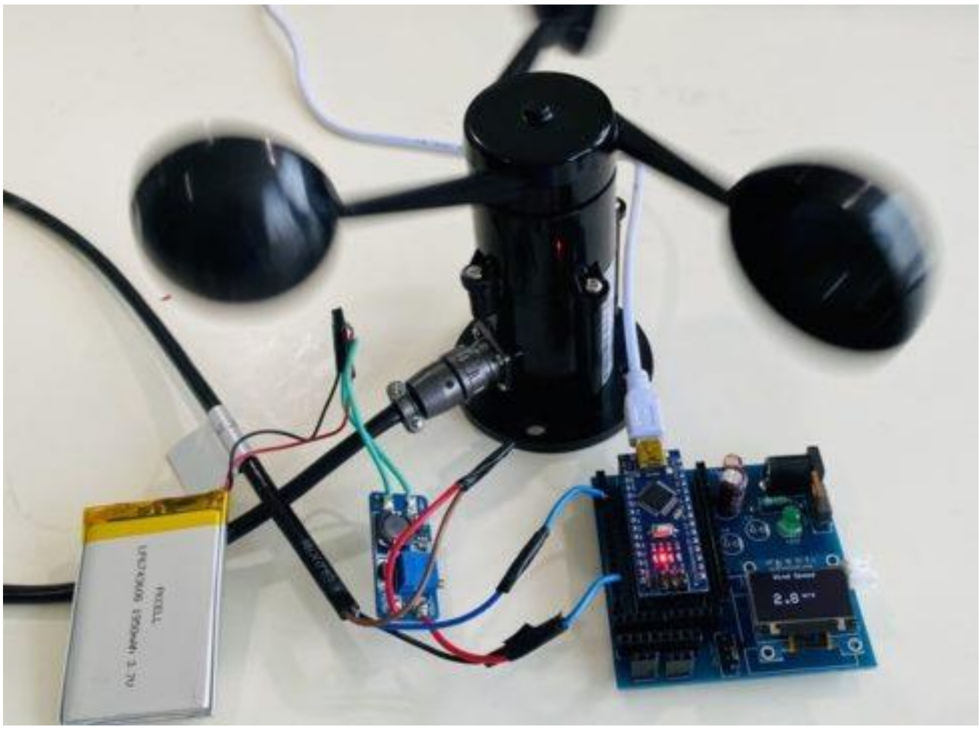
```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128  // OLED display width, in pixels
#define SCREEN_HEIGHT 64  // OLED display height, in pixels
#define OLED_RESET     4  // Reset pin # (or -1 if sharing reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

```

```

void setup()
{
  Serial.begin(9600);
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
  {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;); // Don't proceed, loop forever
  }
  display.display();
  delay(100);
  display.clearDisplay();

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(0, 20);
  display.print("Anemometer");
  display.display();
  delay(3000);
}

void loop()
{
  float sensorValue = analogRead(A0);
  Serial.print("Analog Value =");
  Serial.println(sensorValue);

  float voltage = (sensorValue / 1023) * 5;
  Serial.print("Voltage =");
  Serial.print(voltage);
  Serial.println(" V");

  float wind_speed = mapfloat(voltage, 0.4, 2, 0, 32.4);
  float speed_mph = ((wind_speed *3600)/1609.344);
  Serial.print("Wind Speed =");
  Serial.print(wind_speed);
  Serial.println("m/s");
  Serial.print(speed_mph);
  Serial.println("mph");

  display.clearDisplay();

  display.setTextSize(1);
  display.setCursor(30, 0);
  display.println("Wind Speed");

//  display.setTextSize(2);
//  display.setCursor(25, 30);
//  display.print(wind_speed, 1);
//  display.setTextSize(1);
//  display.print(" m/s");

  display.setTextSize(2);
  display.setCursor(25, 30);
  display.print(speed_mph, 1);
  display.setTextSize(1);
  display.print(" mph");

  display.display();

  Serial.println(" ");
  delay(300);
}

float mapfloat(float x, float in_min, float in_max, float out_min, float out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

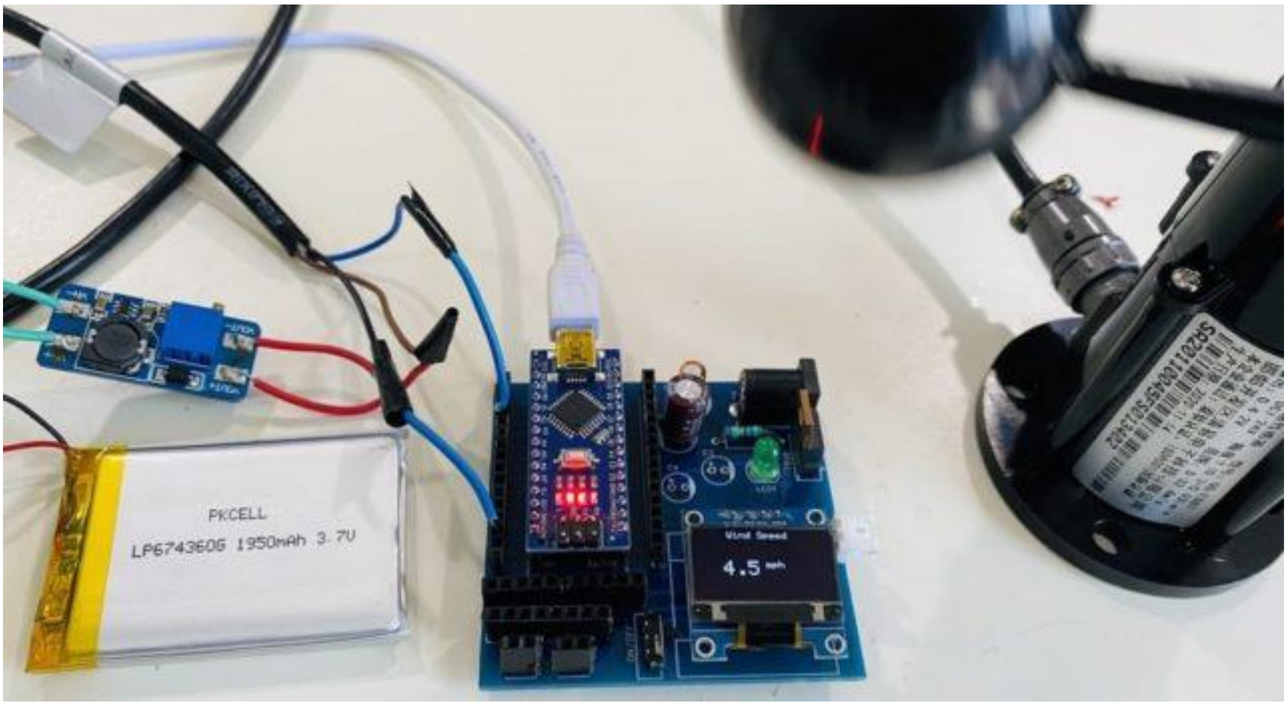
Wind Speed & Testing the Device

Once you upload the Arduino Anemometer code to the Arduino Board, the OLED Display will start displaying the wind speed. When the anemometer cup is stable the OLED will show speed around 0.1m/s to 0.3m/s. The speed will vary or increase when the sensor is taken in a windy region. I used my home roof to check the wind speed.

Initially we tried measuring the speed in m/s and got the result as shown in the image below. The speed variation is increases or decreases depending upon the blade rotation.



Then we used the **industrial standard** measurement and converted the m/s to miles per hour (mph). You can also use the mathematical calculation to convert the speed to a kilometer per hour (kph).

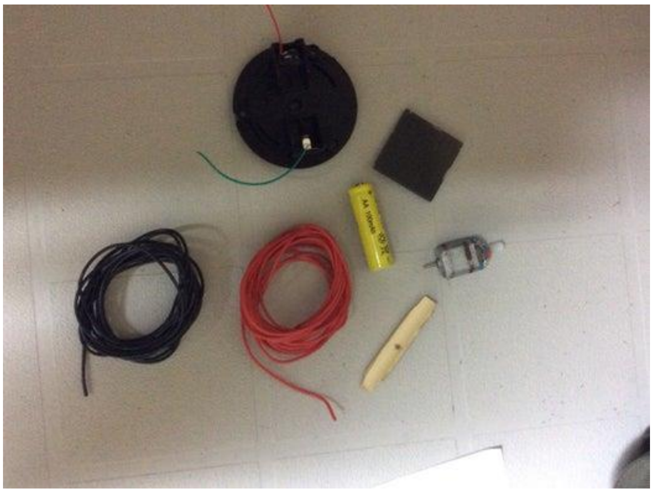


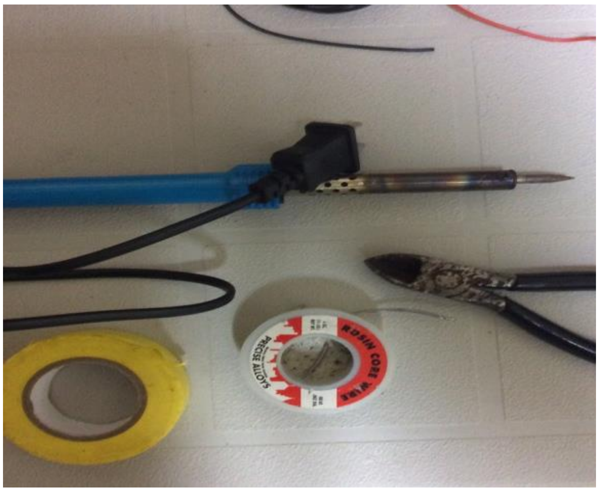
SOLAR POWERED DC MOTOR

DC motors can be used for a variety of projects and tasks, from homemade projects such as fans or improvised drills to more practical machines, like circular saws. The uses of DC motors are endless and can make your life easier. In this Instructable we will show how to make a solar powered DC motor that can be used a fan.

This small fan project has multiple uses and could be for keeping cool from the summers hot weather or possibly as a fan when soldering to remove fumes. The benifit is that it has a built in battery that will allow it keep working, even when the sun is not shining.

Step 1: Materials and Tools



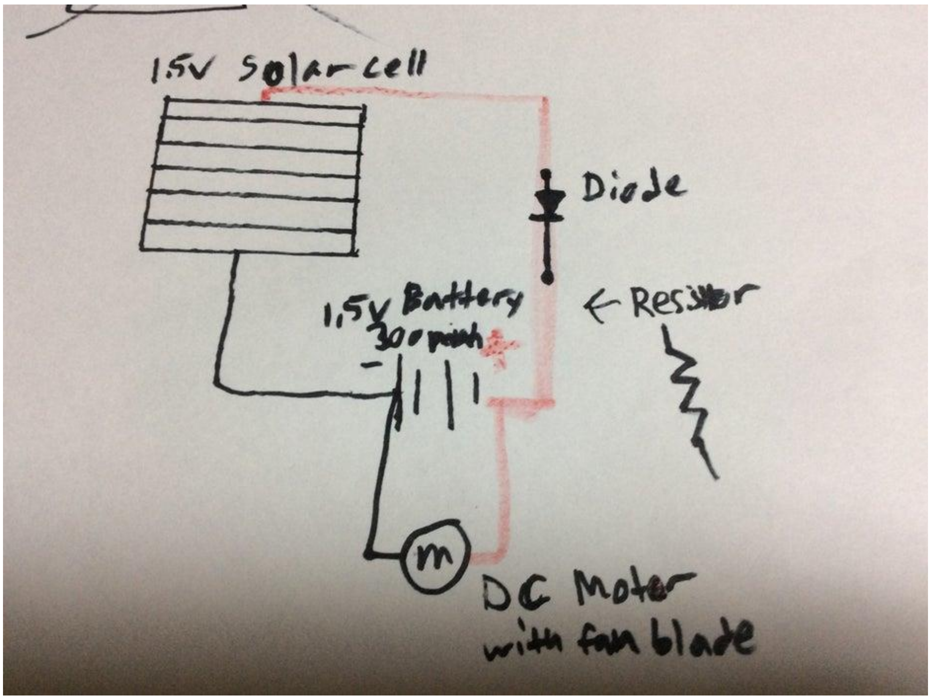


Materials:
. Mini DC motor
. Positive wire
. Negative wire
. Battery holders
. Solar cell 1.5V 100mah (Dollar store garden light)
. Popsicle stick (Fan blades)
. Diode
. Battery (I used a NiCd 1.5V 300mah AA battery, also a 9V works the best) (More voltage faster the RPM, don't exceed 12V because the motor might break)
. Ceramic capacitor (To minimizes RF noise)

Optional:
. Propellor
. Resistor (If necessary)

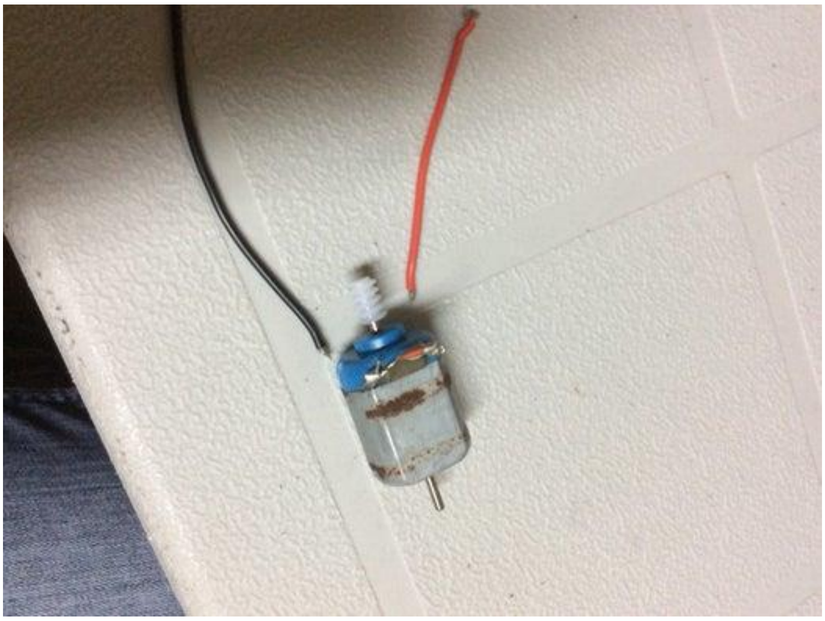
Tools:
. Soldering iron
. Solder rosin 60/40 core
. Electrical tape
. Wire cutters
. Drill (Not shown in diagram)
. Voltmeter (Not shown in diagram)

Step 2: Circuit Diagram



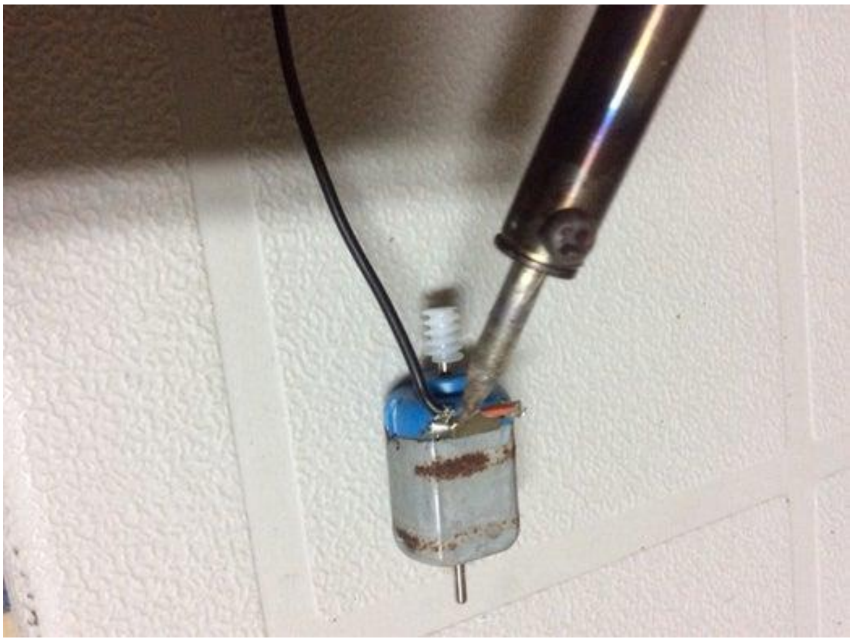
The circuit itself is pretty basic and the diagram shown will show you the layout of the components in the circuit. If you are using different batteries or components to increase the voltage then swap the out with the ones I have used. The resistor amount will be determined by your set up, depending on what type of battery you use or the solar cell you choose.

Step 3: Set Up Before Soldering



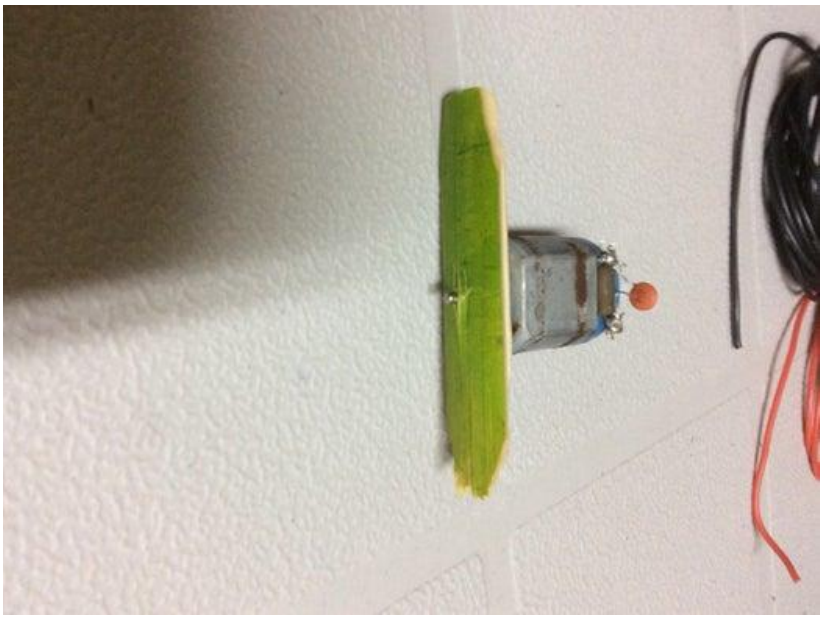
Make sure you have looked over the schematic diagram indicated in step 2. Set up everything as shown in the diagram, then look it over before you solder everything You can let your soldering iron warm up will doing this process.

Step 4: Solder



Once everything is lined up prepare to solder everything together as was shown in the diagram.

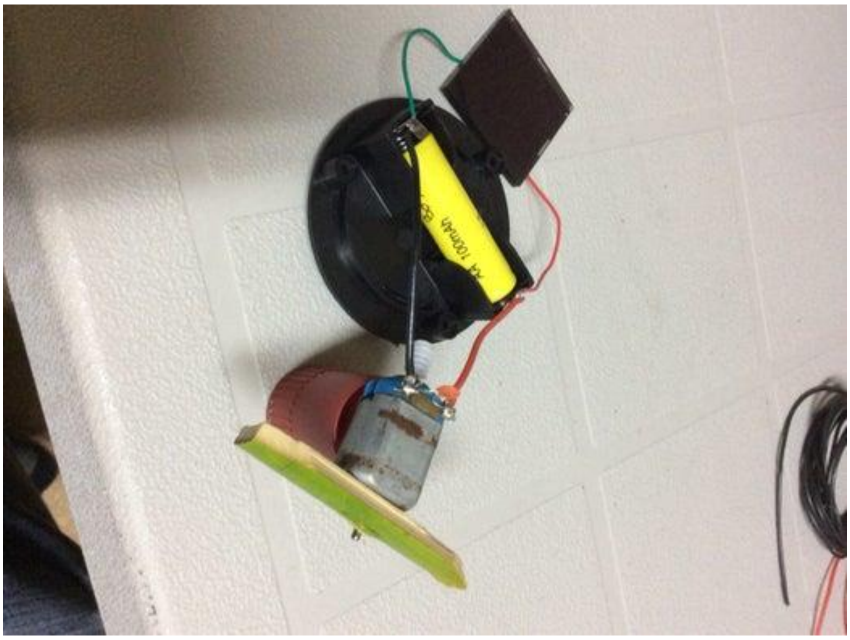
Step 5: The Fan Blade



We used a popsicle stick as my fan blade and for maximum breeze shape the blades so they can create airflow. Google images can show you how to create effective blades just type in fan blades. For more realistic and effective blades We would recommend using a plastic mini propellor, these can be found at RC stores or even from old broken RC toys.

Step 6: Fan Stand (Optional)

If you want need a way of mounting your fan, you could make a tripod device, I used a cap to stand mine up.

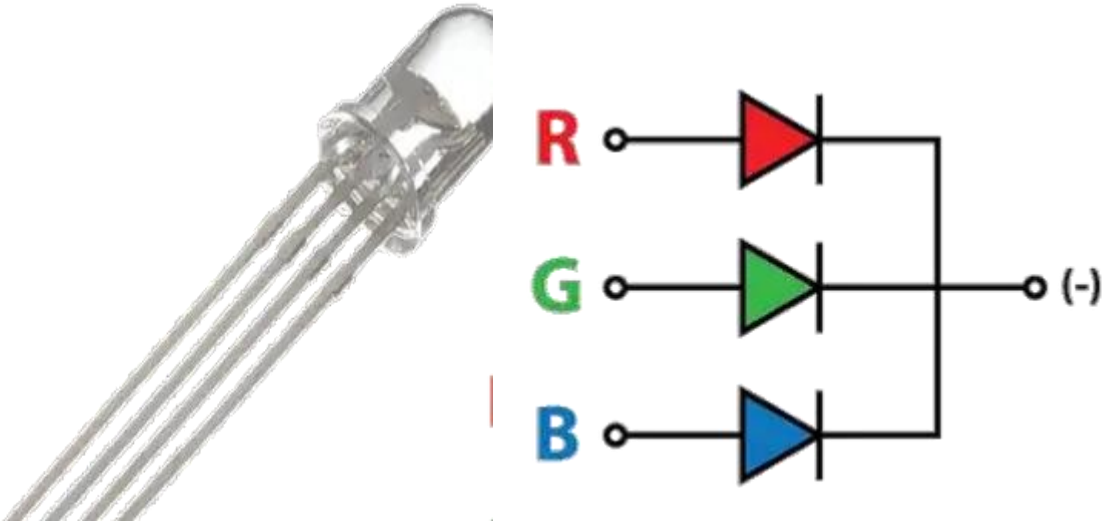


You can make a helicopter form cardboard, and put this device into helicopter for better visual experience.

Funded by the European Union

USE RGB LED WITH ARDUINO

The RGB LED can emit different colors by mixing the 3 basic colors red, green and blue. So it actually consists of 3 separate LEDs red, green and blue packed in a single case. That’s why it has 4 leads, one lead for each of the 3 colors and one common cathode or anode depending of the RGB LED type. In this project we will be using a common cathode one.



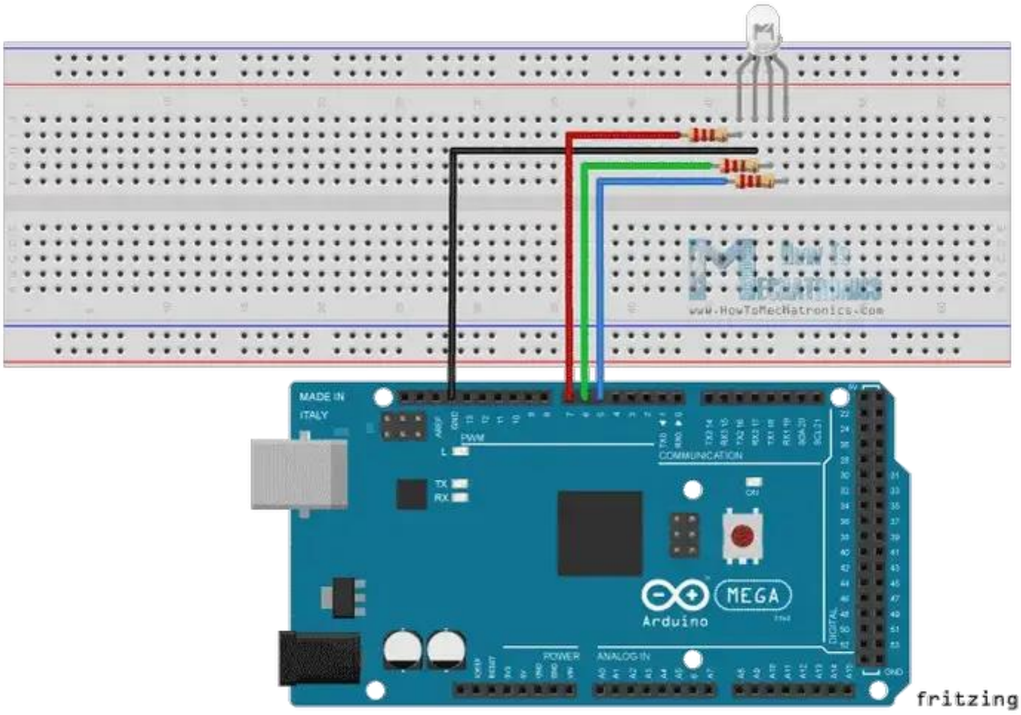
Components needed for this project:

- RGB LED
- 3x 220 Ohms Resistors
- Arduino Board
- Breadboard and Jump Wires

Arduino and RGB LED Circuit Schematics

The cathode will be connected to the ground and the 3 anodes will be connected through 220 Ohms resistors to 3 digital pins on the Arduino Board that can provide PWM signal. We will use PWM for simulating analog output which will provide different voltage levels to the LEDs so we can get the desired colors.

We will use PWM for simulating analog output which will provide different voltage levels to the LEDs so we can get the desired colors.



SOURCE CODE

Now let’s see the Arduino sketch. we will use the pins number 7, 6 and 5 and we will name them redPin, greenPin and bluePin. In the setup section we need to define them as outputs. At the bottom of the sketch we have this custom made function named setColor() which takes 3 different arguments redValue, greenValue and blueValue. These arguments represents the brightness of the LEDs or the duty cycle of the PWM signal which is created using the analogWrite() function. These values can vary from 0 to 255 which represents 100 % duty cycle of the PWM signal or maximum LED brightness.

```
int redPin= 7;int greenPin = 6;int bluePin = 5;

void setup() {

 pinMode(redPin, OUTPUT);

 pinMode(greenPin, OUTPUT);

 pinMode(bluePin, OUTPUT);

}
```

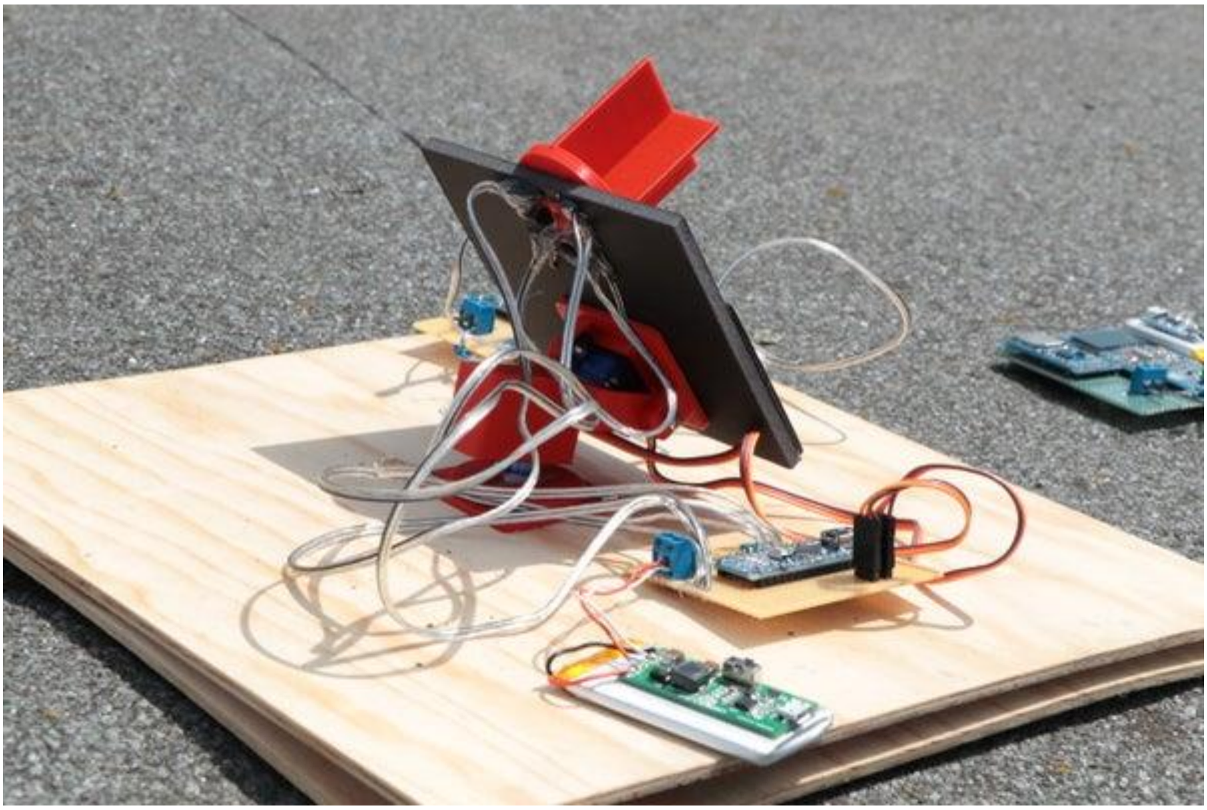
```
void loop() {
 setColor(255, 0, 0); // Red Color
 delay(1000);
 setColor(0, 255, 0); // Green Color
 delay(1000);
 setColor(0, 0, 255); // Blue Color
 delay(1000);
 setColor(255, 255, 255); // White Color
 delay(1000);
 setColor(170, 0, 255); // Purple Color
 delay(1000);
}
void setColor(int redValue, int greenValue, int blueValue) {
 analogWrite(redPin, redValue);
 analogWrite(greenPin, greenValue);
 analogWrite(bluePin, blueValue);
}
```

So now in the loop function we will make our program which will change the color of the LED each second. In order to get red light on the LED we will call the setColor() function and set value of 255 for the redValue argument and 0 for the two others. Respectively we can get the two other basic colors, green and blue. For getting other colors we need to mix the arguments values. So for example if set all 3 LEDS to maximum brightness we will get White color and we will get a purple color if we set the following values to the arguments: 170 redValue, 0 greenValue and 255 blueValue. Here's the demonstration of the sketch.

Funded by
the European Union

SOLAR TRACKER PROJECT

In this project we will create a solar tracker which like the name implies can follow the movement of the sun throughout the day.



Here you can find a parts list:

1x Arduino Nano

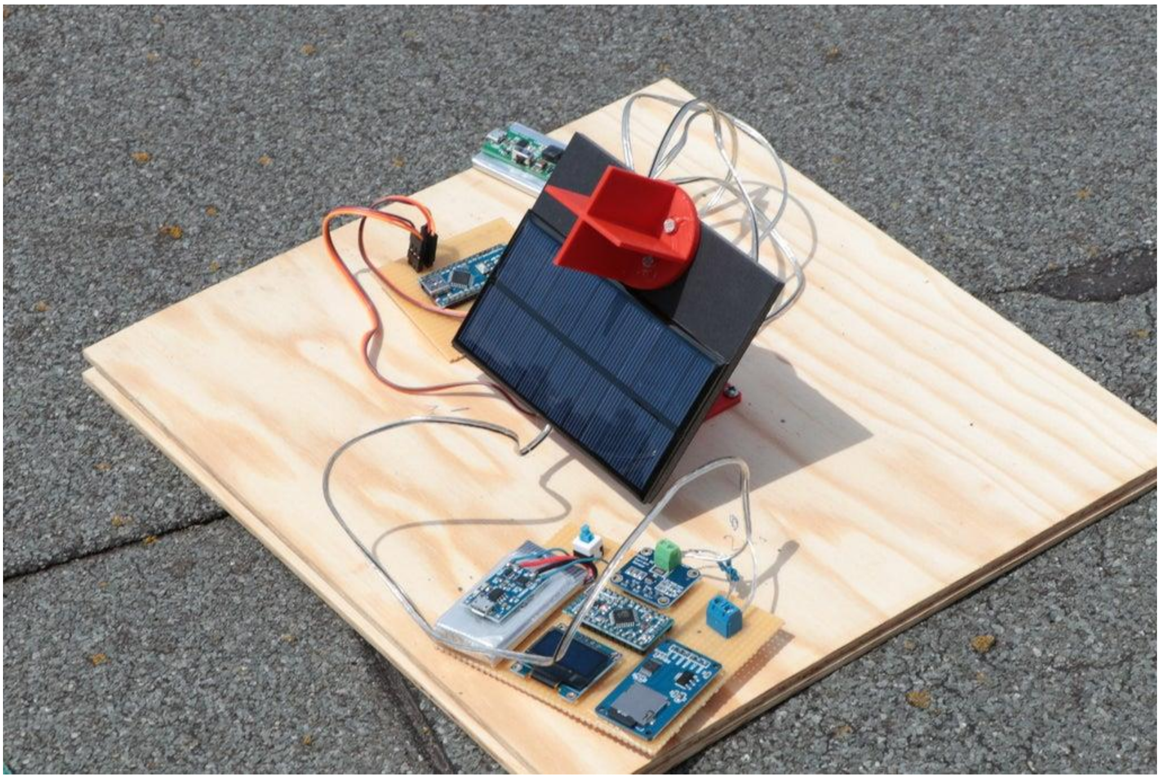
2x SG90 Servo

4x Photoresistor

4x 1kohm Resistor

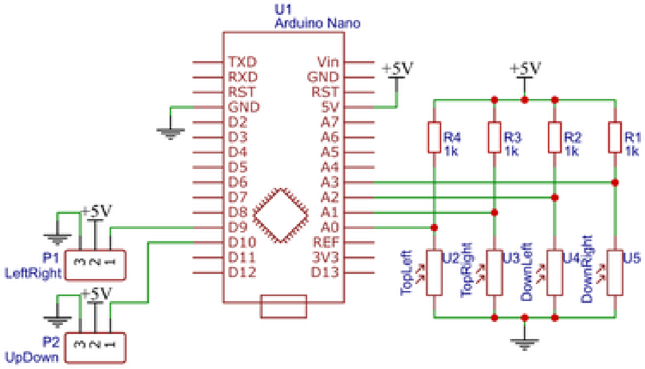
Funded by the European Union

Step 1: 3D Print the Solar Tracker Parts



Here you can find the utilized model on thingiverse: https://www.thingiverse.com/thing:708819

Step 2: Connect the components according to the diagram in the picture



Step 2: Copy source code to Arduino IDE and upload it to Arduino board

```
#include <Servo.h>

Servo myservo1, myservo2;
int LDR1 = A0, LDR2 = A1, LDR3 = A2, LDR4 = A3;
int rRDL1 = 0, rRDL2 = 0, rRDL3 = 0, rRDL4 = 0;
int max1=0, max2=0, max3=0;
int ser1 = 80, ser2=0;
void setup() {
myservo1.attach(9);
myservo2.attach(8);
Serial.begin(9600);
myservo1.write(ser1);
myservo2.write(100);
}
```

```
void loop() {
rRDL1 = analogRead(LDR1) / 100;
rRDL2 = analogRead(LDR2) / 100;
rRDL3 = analogRead(LDR3) / 100;
rRDL4 = analogRead(LDR4) / 100;
max1 = max(rRDL1, rRDL2);
max2 = max(rRDL3, rRDL4);
max3 = max(max1, max2);
//Serial.println(String(max3));
//Serial.println(String(rRDL1) +", "+String(rRDL2) +", "+String(rRDL3) +", "+String(rRDL4));
if(rRDL1<max3 && rRDL2<max3)
{
if(ser1<140)
ser1+=1;
myservo1.write(ser1);
}
if(rRDL3<max3 && rRDL4<max3)
{
if(ser1>0)
ser1-=1;
myservo1.write(ser1);
}
if(rRDL2<max3 && rRDL3<max3)
{
Serial.println("servo2 +" + String(ser2));
if(ser2<180)
ser2+=1;
myservo2.write(ser2);
}
if(rRDL1<max3 && rRDL4<max3)
{
Serial.println("servo2 -" + String(ser2));
if(ser2>0)
ser2-=1;
myservo2.write(ser2);
}
delay(15);
}
```

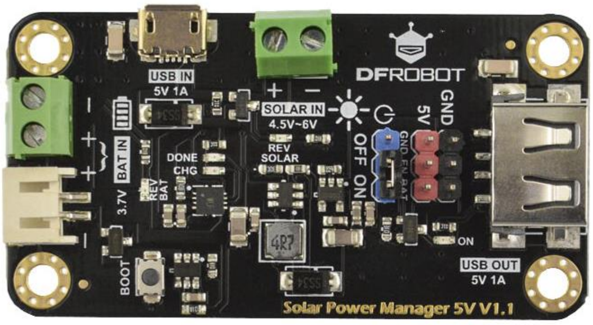
Step 3: Connect solar charger components to solar panel

For that we need next parts:

DFRobot Solar Power Manager 5V

Solar panel

USB cable for charging power bank or phone

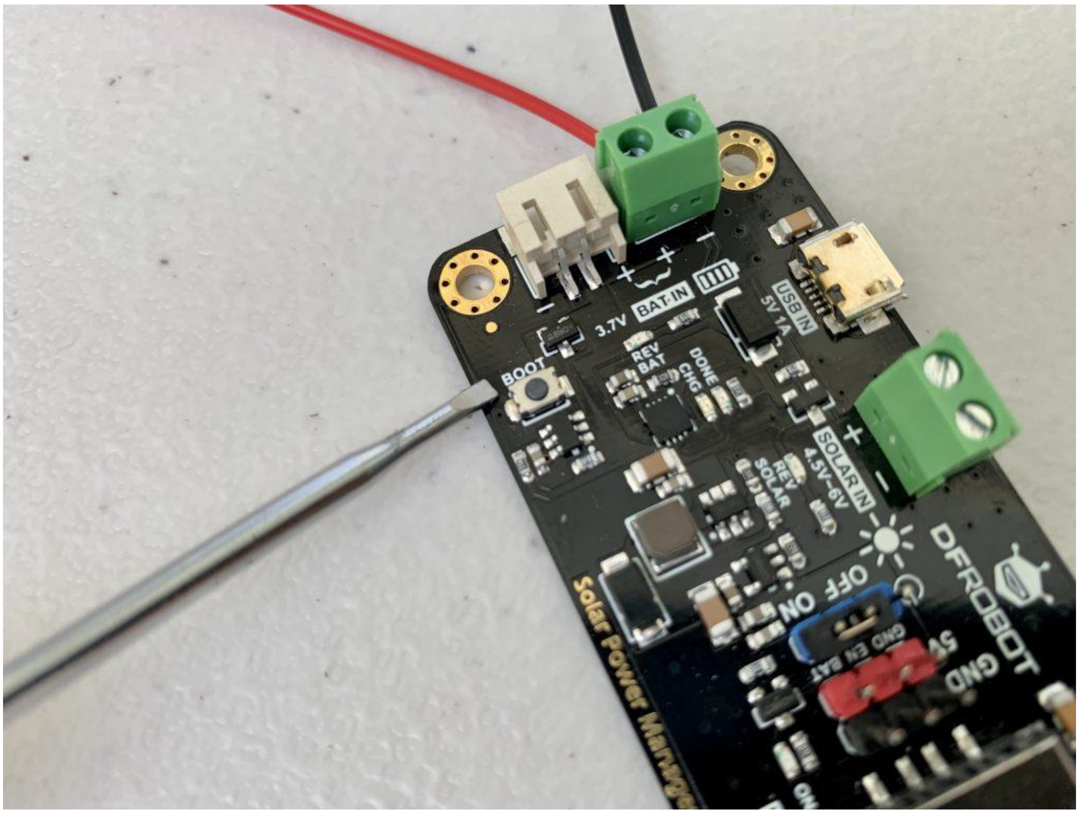


FRobot Solar Power Manager series are designed for IoT projects and renewable energy projects, providing safe and high-efficiency embedded solar power management modules for makers and application engineers.

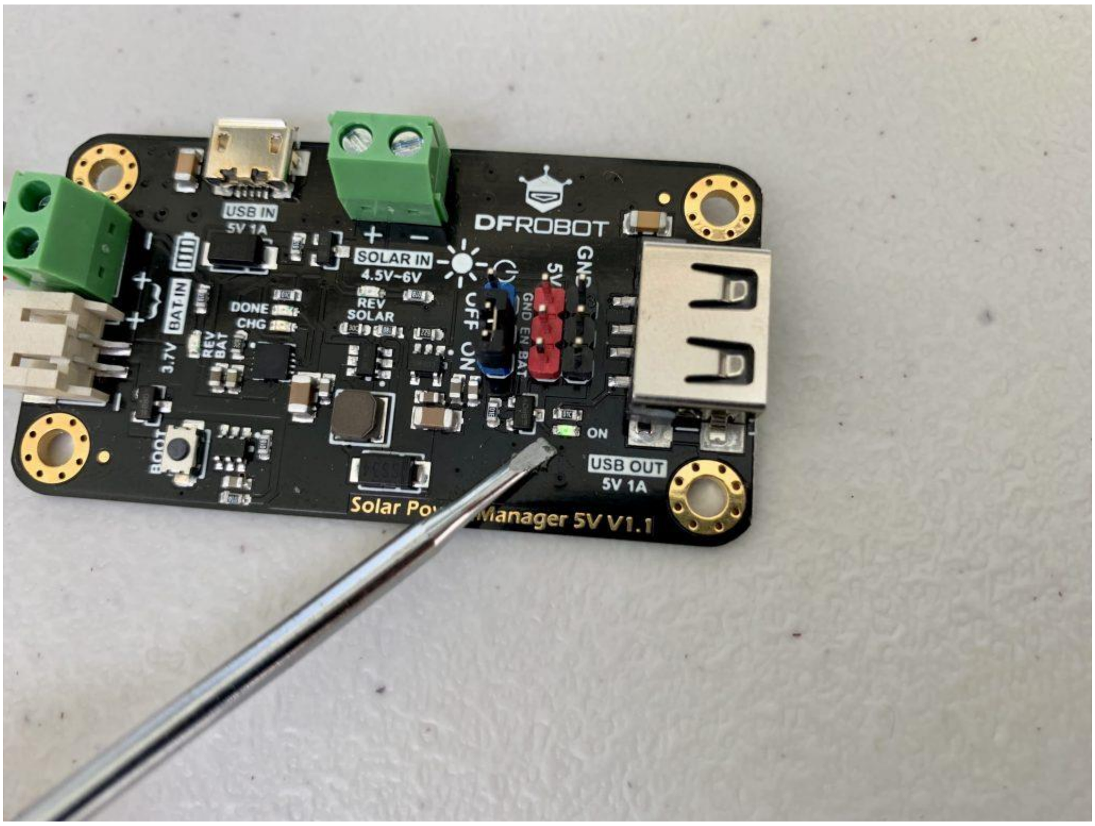
Solar Power Manager 5V is a small power and high-efficiency solar power management module designed for 5V solar panel. It features as MPPT (Maximum Power Point Tracking) function, maximizing the efficiency of the solar panel. The module can provide up to 900mA charging current to 3.7V Li battery with USB charger or solar panel.

The ON/OFF controllable DC-DC converters with 5V 1A output satisfies the needs of various solar power projects and low-power applications. The module also employs various protection functions for battery, solar panel and output, which greatly improves the stability and safety of your solar projects.

Now Locate the BOOT button on the board. It’s next to the JST port.

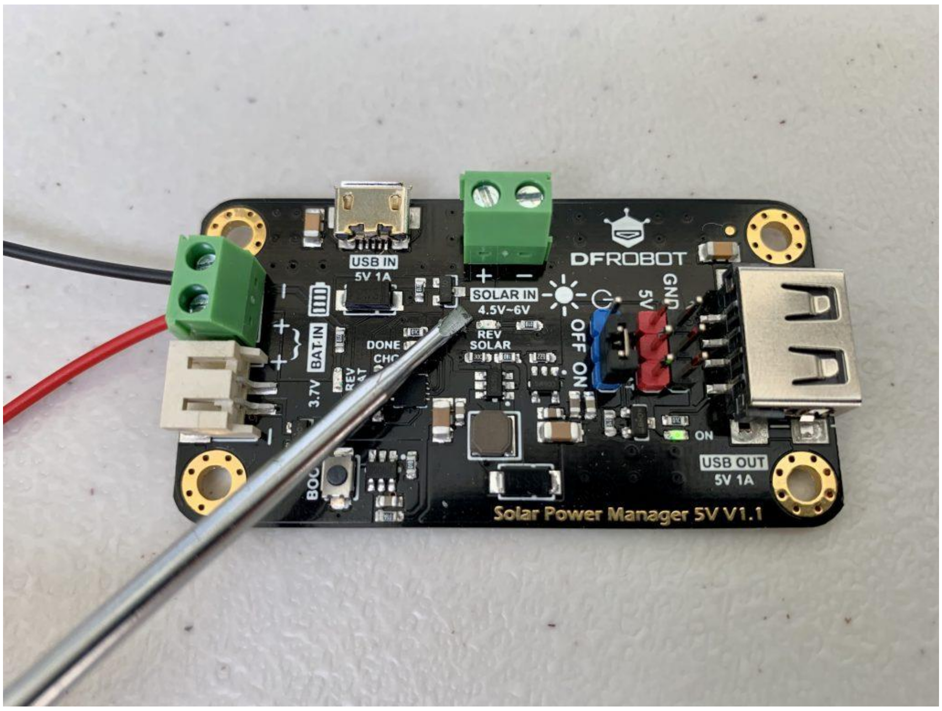


Press the BOOT button to turn the board on. Once you do, you should see the ON light turn on.

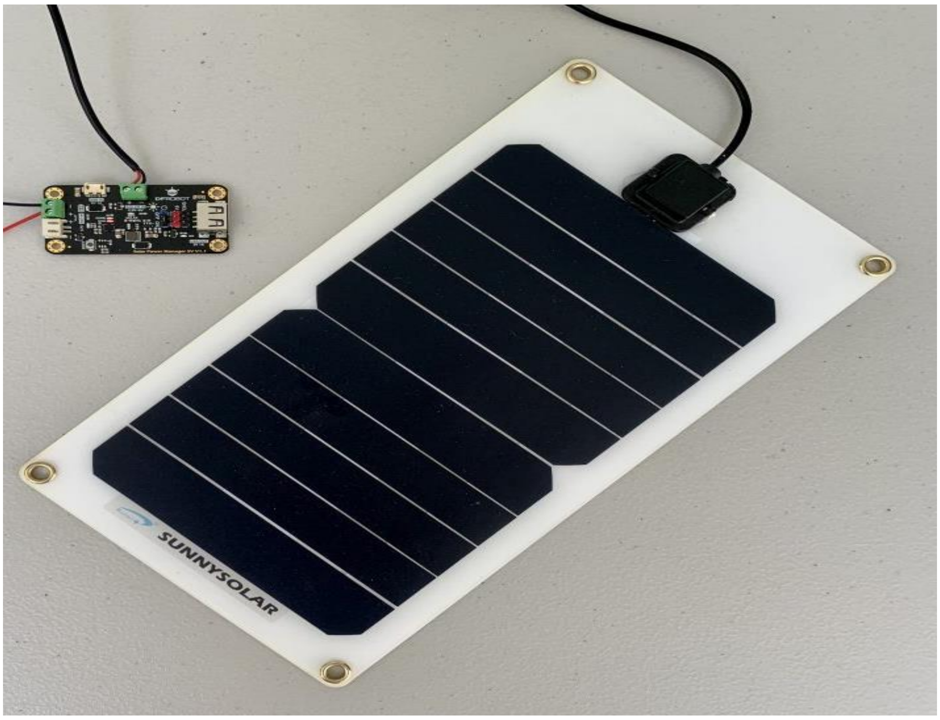


4: Connect the Solar Panel to the Solar Power Manager

Locate the solar terminals on the Solar Power Manager. They’re the other set of green screw terminals.



Connect the solar panel leads to the solar terminals.



Now you need to plug usb chrging cable into USB port and start with charging power bank or mobile phone.