

# Arduino

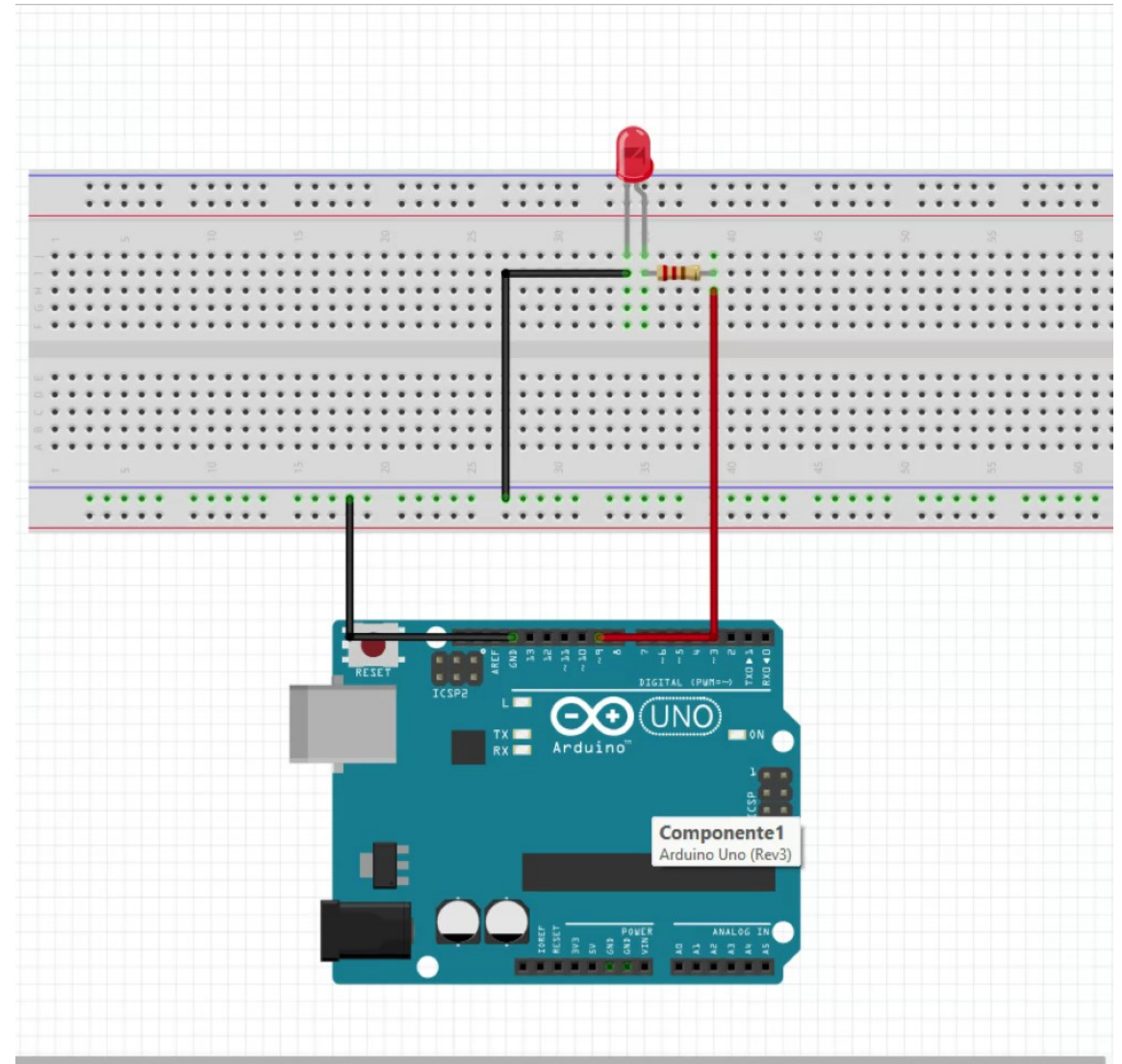
Fatto da Valentino Hu

classe 1A

Anno 2021/2022

# Primo laboratorio: primo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un led rosso
- Una resistenza da 330 Ohm



# Codice primo circuito

Con questo codice il led dovrebbe man mano aumentare di luminosità, arrivare al massimo e poi diminuire sfruttando un ciclo for



```
1
int led = 9; // the PWM pin the LED is attached to
int brightness = 0; // how bright the LED is
int fadeAmount = 5; // how many points to fade the LED by
// the setup routine runs once when you press reset:

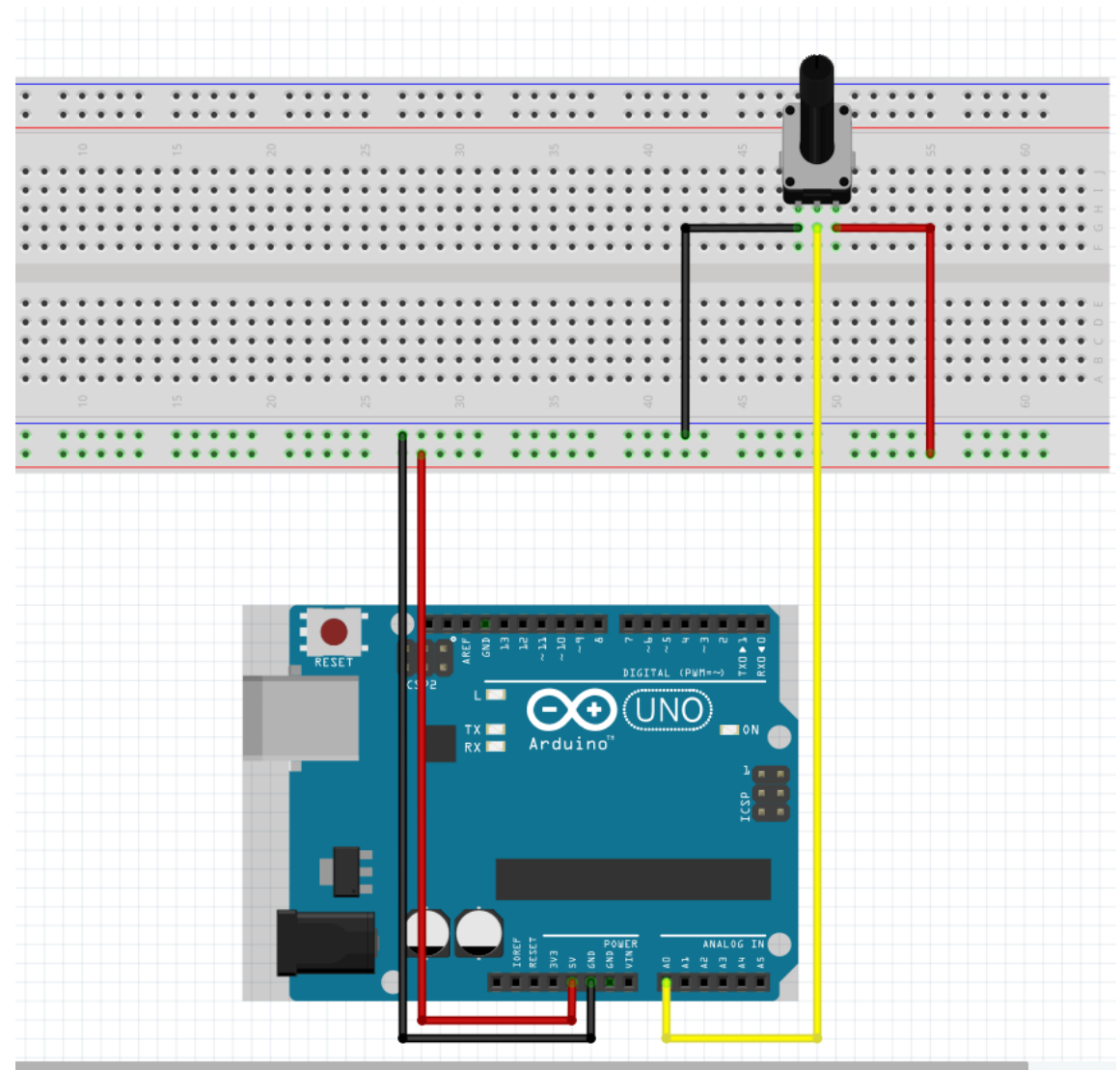
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:

void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);
  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;
  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(300);
}
```

# Primo laboratorio: secondo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un resistore variabile 5k (potenziometro)



# Codice secondo circuito

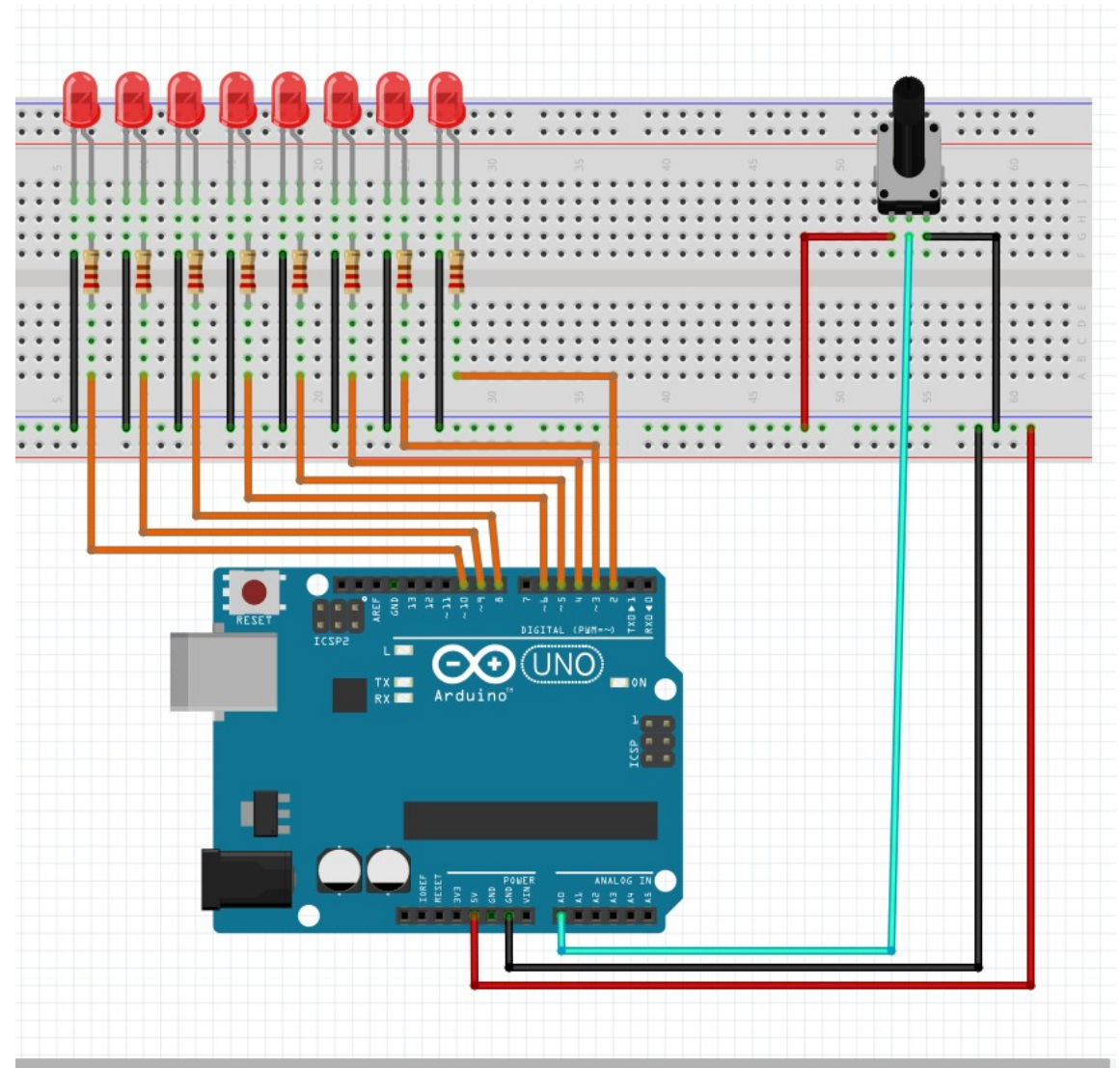
Con questo codice andando su IDE e cliccando in alto a destra su monitor seriale, uscirà uno schermo dove vi farà vedere il voltaggio letto dal PIN. La tensione aumenta e diminuisce girando il potenziometro



```
void setup() {  
  // initialize serial communication at 9600 bits per second:  
  Serial.begin(9600);  
}  
  
// the loop routine runs over and over again forever:  
  
void loop() {  
  // read the input on analog pin 0:  
  int sensorValue = analogRead(A0);  
  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):  
  float voltage = sensorValue * (5.0 / 1023.0);  
  // print out the value you read:  
  Serial.println(voltage);  
}
```



- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un resistore variabile 5k (potenziometro)
- Otto led rossi
- Otto resistenze da 330 ohm



# Codice terzo circuito

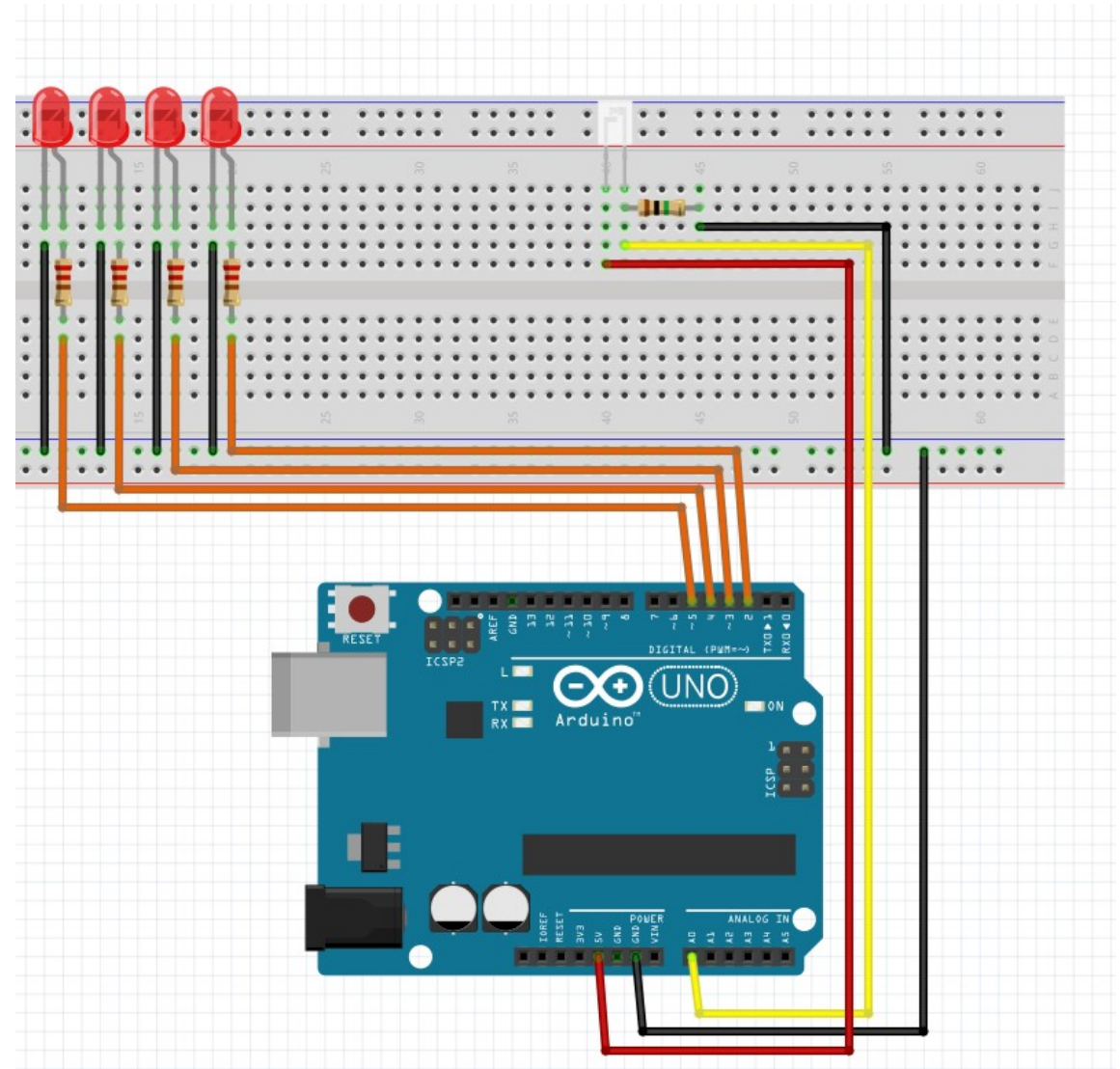
Con questo codice girando il potenziometro i led si accenderanno uno dopo l'altro in proporzione alla tensione sentita dal PIN



```
sketch_rdv18a$  
  
const int analogPin = A0; // the pin that the potentiometer is attached to  
const int ledCount = 8; // the number of LEDs in the bar graph  
int ledPins[] = {2, 3, 4, 5, 6, 7, 8, 9}; // an array of pin numbers to which LEDs are attached  
  
void setup() {  
  // loop over the pin array and set them all to output:  
  for (int thisLed = 0; thisLed < ledCount; thisLed++) {  
    pinMode(ledPins[thisLed], OUTPUT);  
  }  
}  
  
void loop() {  
  // read the potentiometer:  
  int sensorReading = analogRead(analogPin);  
  // map the result to a range from 0 to the number of LEDs:  
  int ledLevel = map(sensorReading, 0, 1023, 0, ledCount);  
  // loop over the LED array:  
  for (int thisLed = 0; thisLed < ledCount; thisLed++) {  
    // if the array element's index is less than ledLevel,  
    // turn the pin for this element on:  
    if (thisLed < ledLevel) {  
      digitalWrite(ledPins[thisLed], HIGH);  
    } else { // turn off all pins higher than the ledLevel:  
      digitalWrite(ledPins[thisLed], LOW);  
    }  
  }  
}
```

## secondo laboratorio: primo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Resistenza da 1M ohm
- Quattro led rossi
- Quattro resistenze da 330 ohm
- phototransistor





# Codice primo circuito

Con questo codice dopo la fase di calibrazione e la scelta della resistenza adatta, più si fa buio e più LED si accendono



```
File Modifica Sketch Strumenti Aiuto
[Icons]
phototransistor_come_sensore_di_led_e_led_bar_graph$
const int analogPin = A0; // pin that the sensor is attached to
// variables:
const int ledCount = 4; // the number of LEDs in the bar graph
int ledPins[] = {2, 3, 4, 5}; // an array of pin numbers to which LEDs are attached
int sensorValue; // the sensor value
int sensorMin = 1023; // minimum sensor value
int sensorMax = 0; // maximum sensor value

void setup() {
  Serial.begin(9600);
  for (int thisLed = 0; thisLed < ledCount; thisLed++) {
    pinMode(ledPins[thisLed], OUTPUT);
  }
  while (millis() < 5000) {
    sensorValue = analogRead(analogPin);
    // record the maximum sensor value
    if (sensorValue > sensorMax) {
      sensorMax = sensorValue;
    }
    // record the minimum sensor value
    if (sensorValue < sensorMin) {
      sensorMin = sensorValue;
    }
  }
}

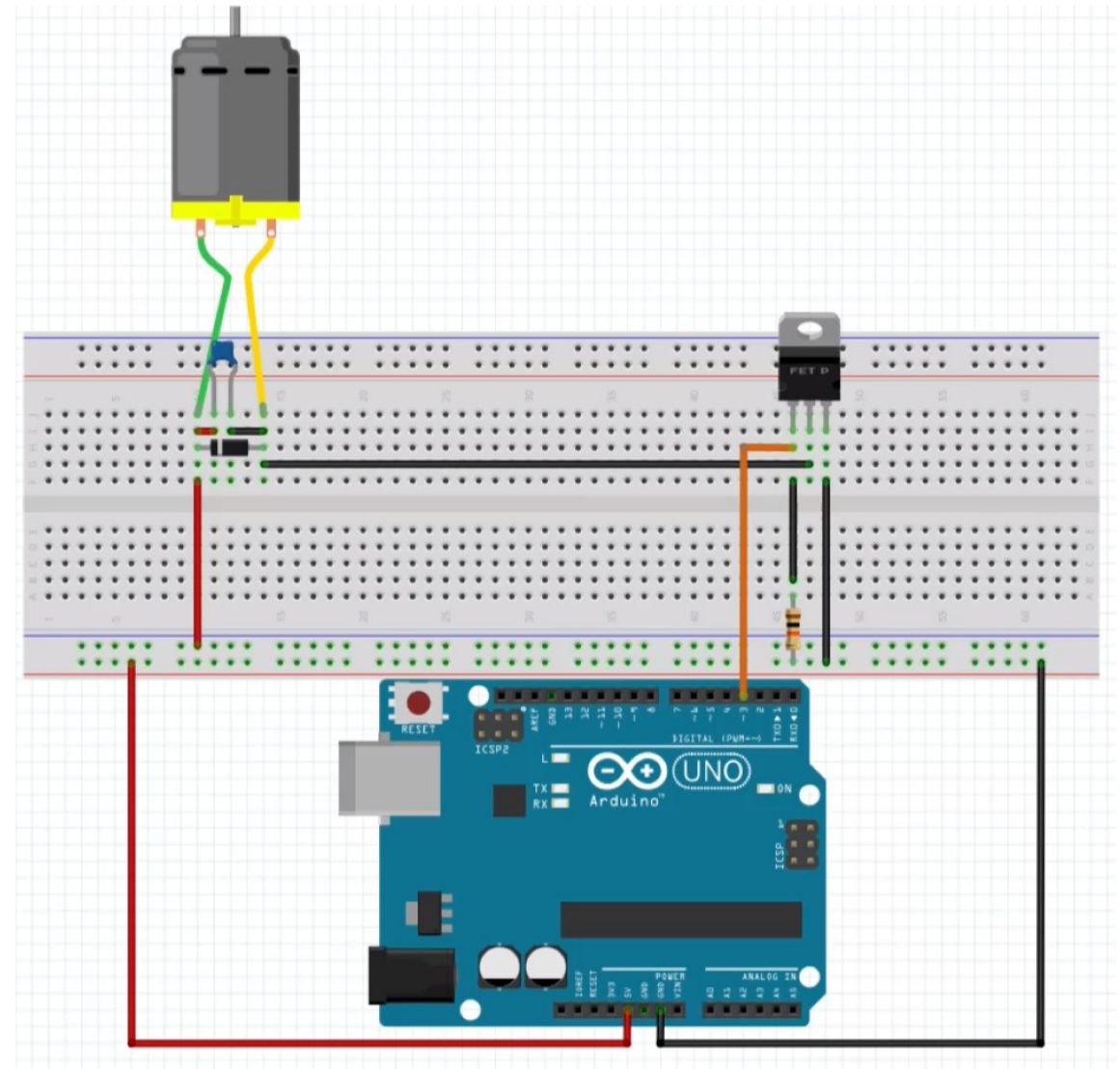
void loop() {
  // read the sensor:
  sensorValue = analogRead(analogPin);
  Serial.println("sensorMin");
  Serial.println(sensorMin);
  Serial.println("sensorMax");
  Serial.println(sensorMax);
  Serial.println("sensorValue");
  Serial.println(sensorValue);
}

Salvataggio annullato

31 Arduino Uno su COM3
```

# Terzo laboratorio: primo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un DC motor
- Un diodo
- Un mosfet
- Un condensatore
- Resistenza



# Codice primo circuito

Con questo codice il motore si azionerà e si fermerà a intermittenza per intervalli di un secondo, ripetendosi all'infinito



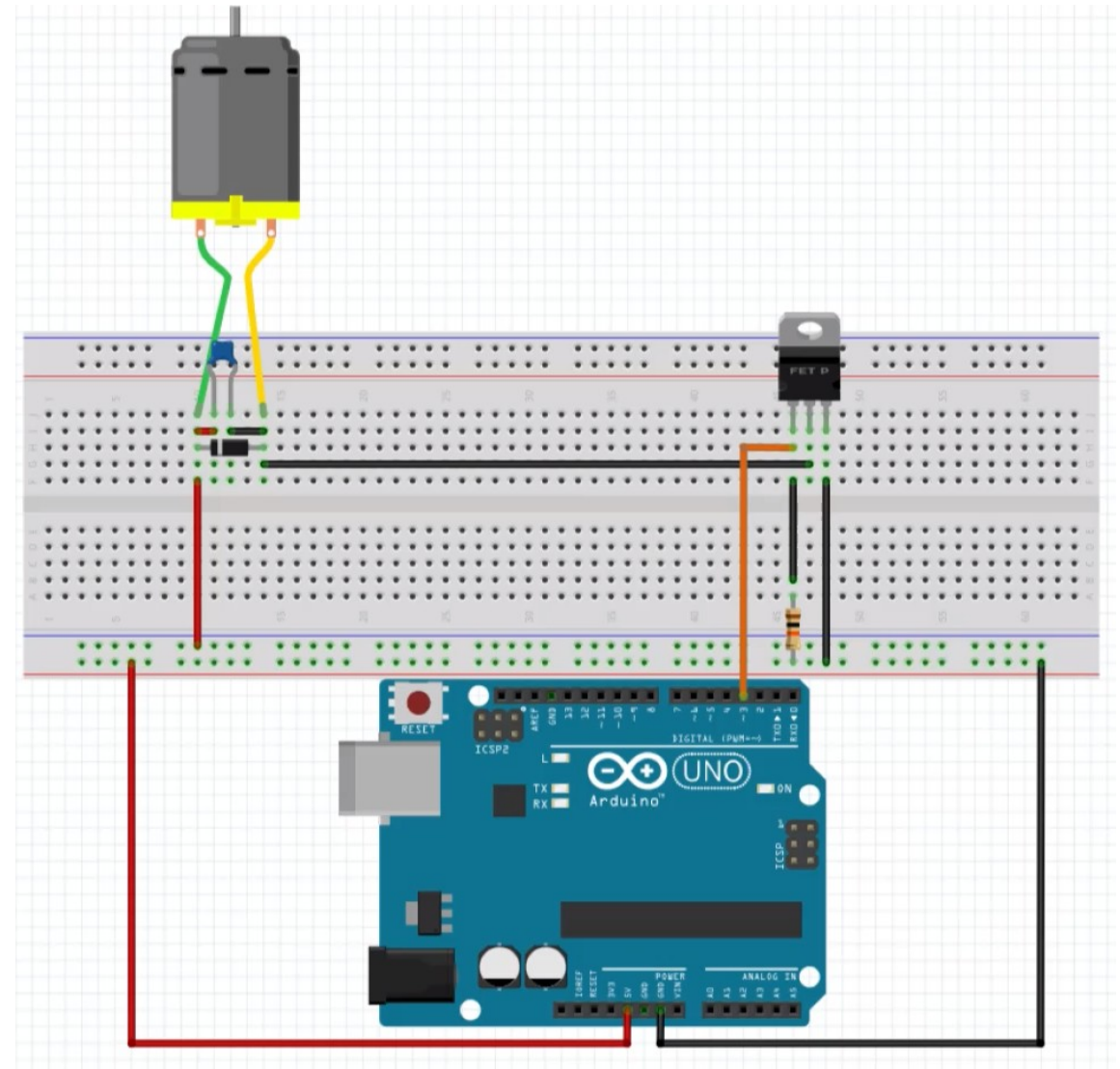
```
motore_a_secondi
int pinMot = 3;

void setup() {
  // put your setup code here, to run once:
  pinMode(pinMot, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  //analogWrite(pinMot, 0);
  //delay(1000);
  digitalWrite(pinMot, HIGH);
  delay(1000);
  digitalWrite(pinMot, LOW);
  delay(1000);
}
```

## Terzo laboratorio: secondo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un DC motor
- Un diodo
- Un mosfet
- Un condensatore
- Una resistenza



# Codice secondo circuito

Con questo codice entrando su monitor seriale possiamo decidere la velocità del motore mettendo un numero tra 0 e 255



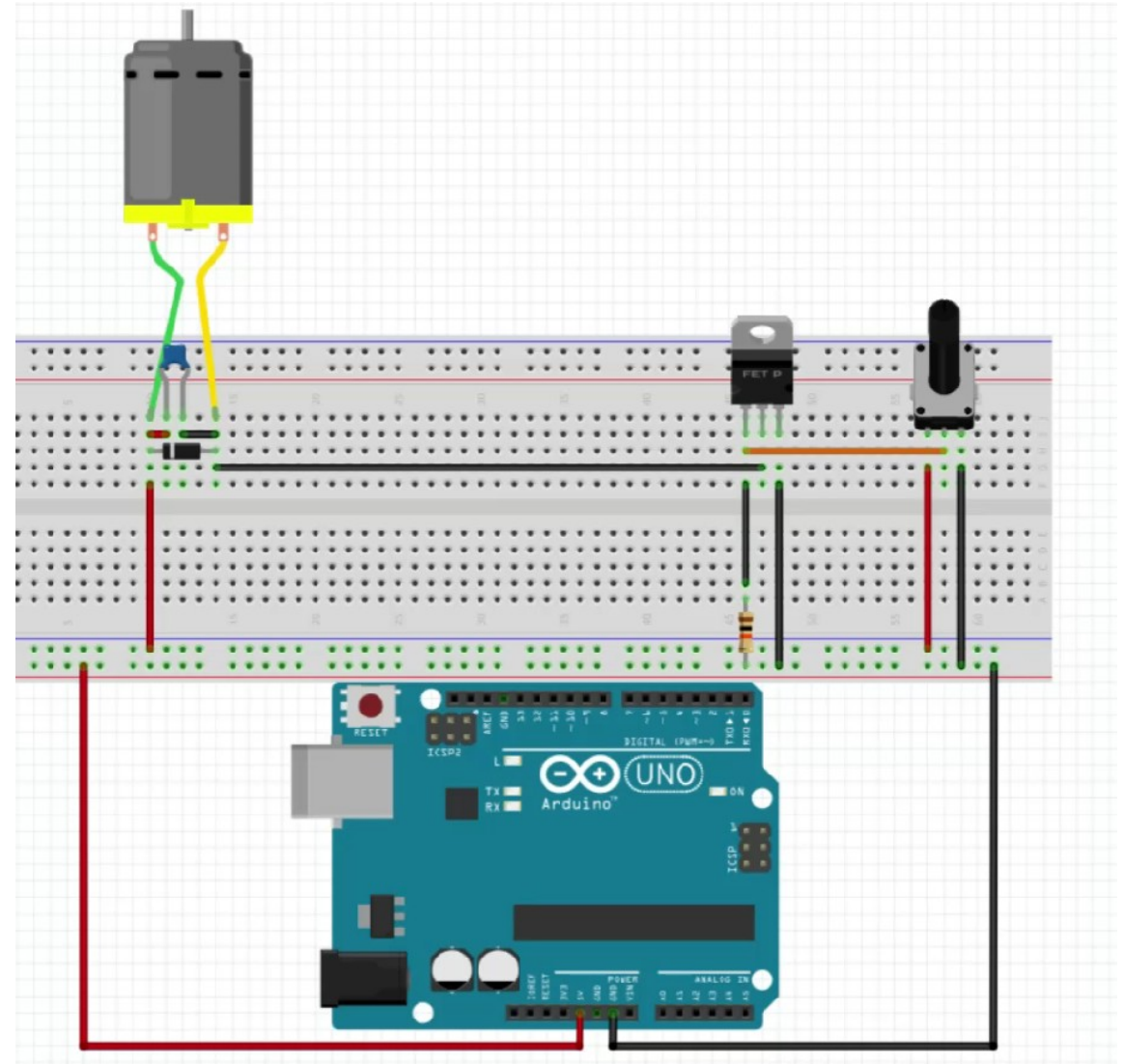
```
int motore_cambio_velocit_
int motorPin = 3;
void setup()
{
  pinMode(motorPin, OUTPUT);
  Serial.begin(9600);
  while (! Serial);
  Serial.println("Speed 0 to 255");
}

void loop()
{
  if (Serial.available())
  {
    int speed = Serial.parseInt();
    if (speed >= 0 && speed <= 255)
    {
      analogWrite(motorPin, speed);
    }
    delay(5000);
  }
}
```



# Terzo laboratorio: terzo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un DC motor
- Un diodo
- Un mosfet
- Un condensatore
- Un resistore variabile 5k (potenziometro)
- Una resistenza



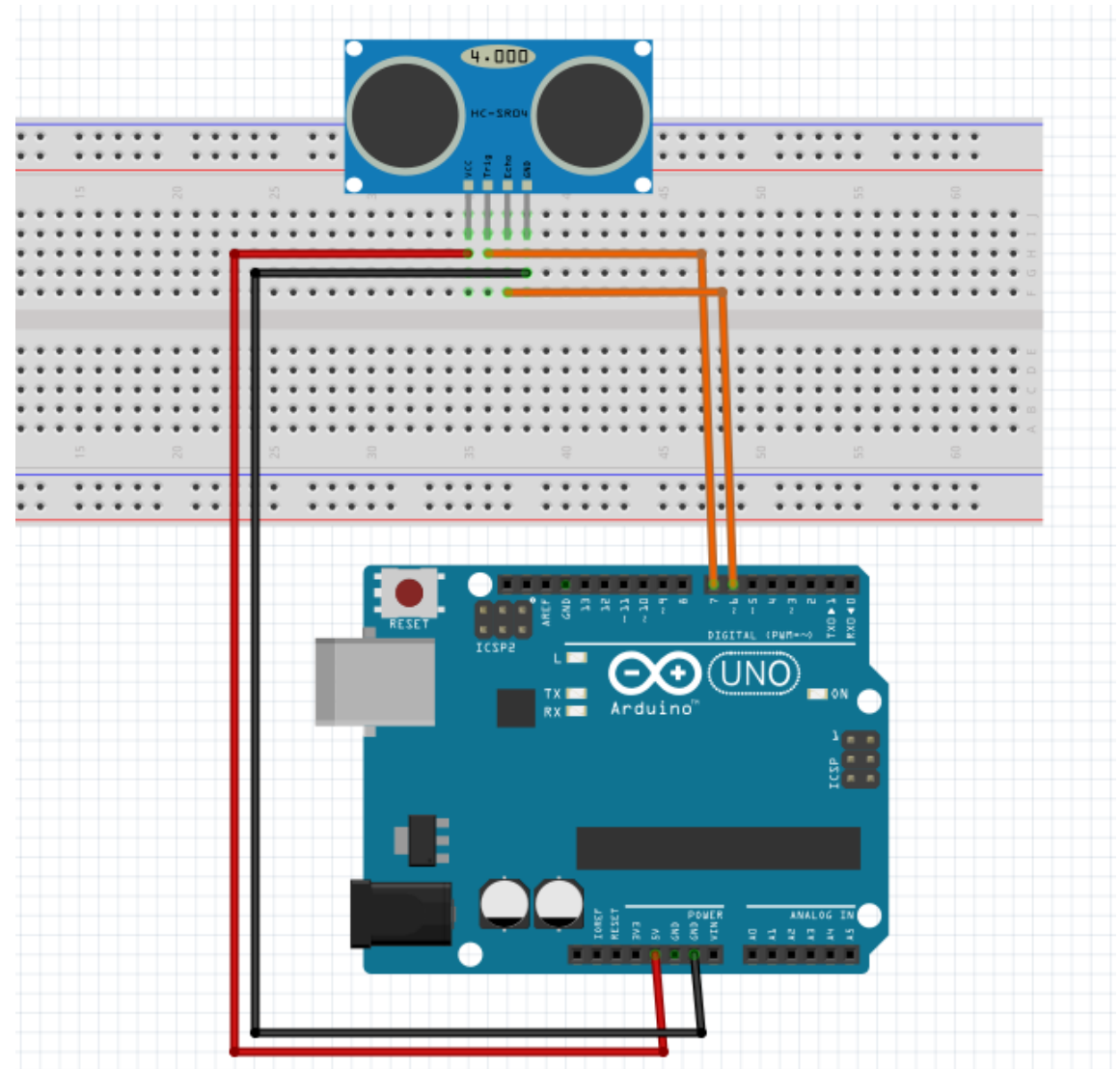
# Nel terzo circuito non serve il codice

Questo circuito girando il potenziometro il motore comincerà ad aumentare la  
velocità

---

# Quarto laboratorio: primo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un sensore di distanza(ultrasonic sensor)



# Codice primo circuito

Andando su monitor seriale ci farà vedere le distanze in cm



```
ultrasonic_sensor

const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 6; // Echo Pin of Ultrasonic Sensor

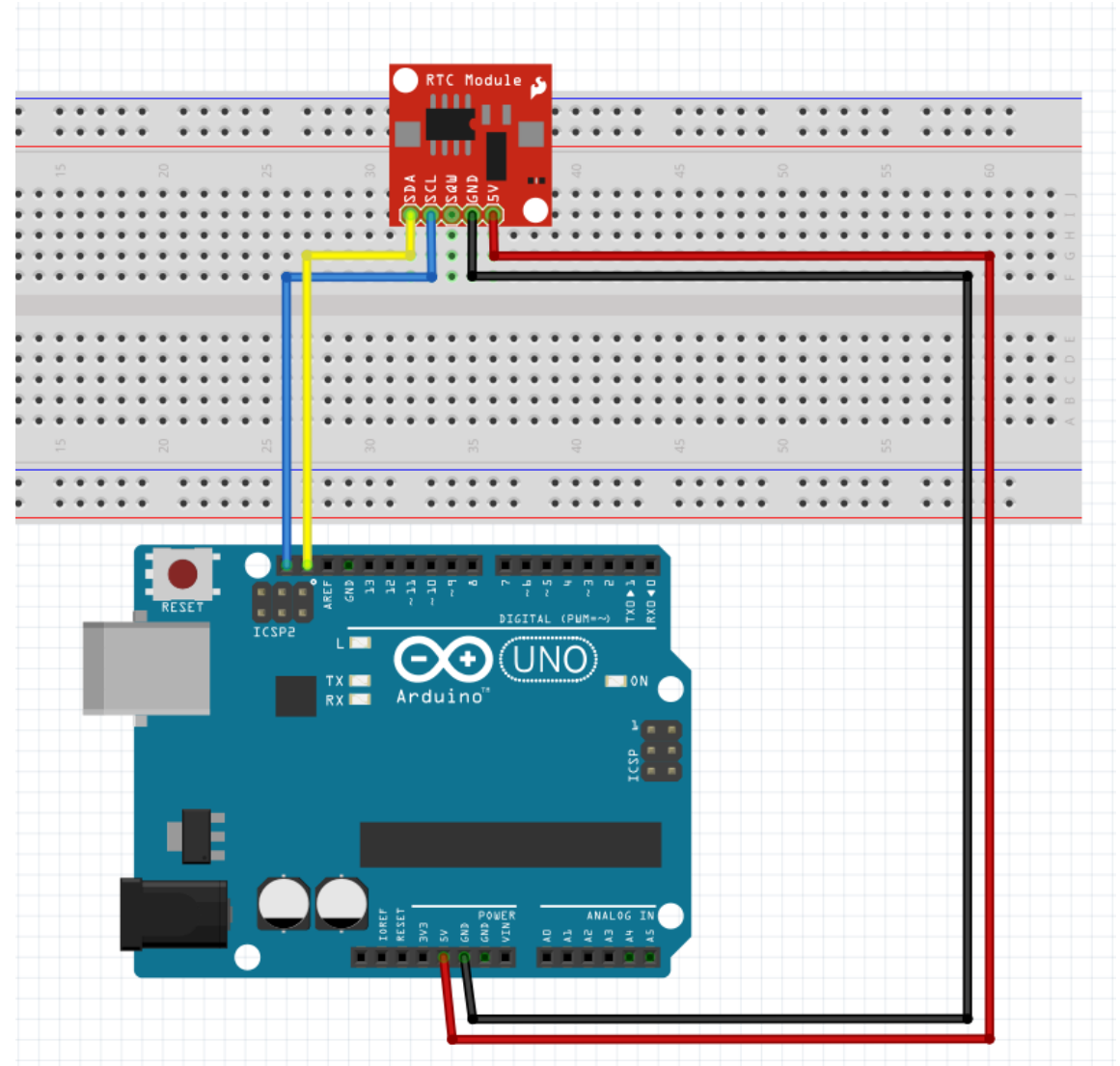
void setup() {
  Serial.begin(9600); // Starting Serial Terminal
}

void loop() {
  long duration, cm;
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(pingPin, LOW);
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
  cm = microsecondsToCentimeters(duration);
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  delay(100);
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

# Quarto laboratorio: secondo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un rtc(real time clock) module





# Codice secondo circuito

Mettendo questo codice impostiamo data e ora e successivamente li si potrà vedere sul monitor seriale.

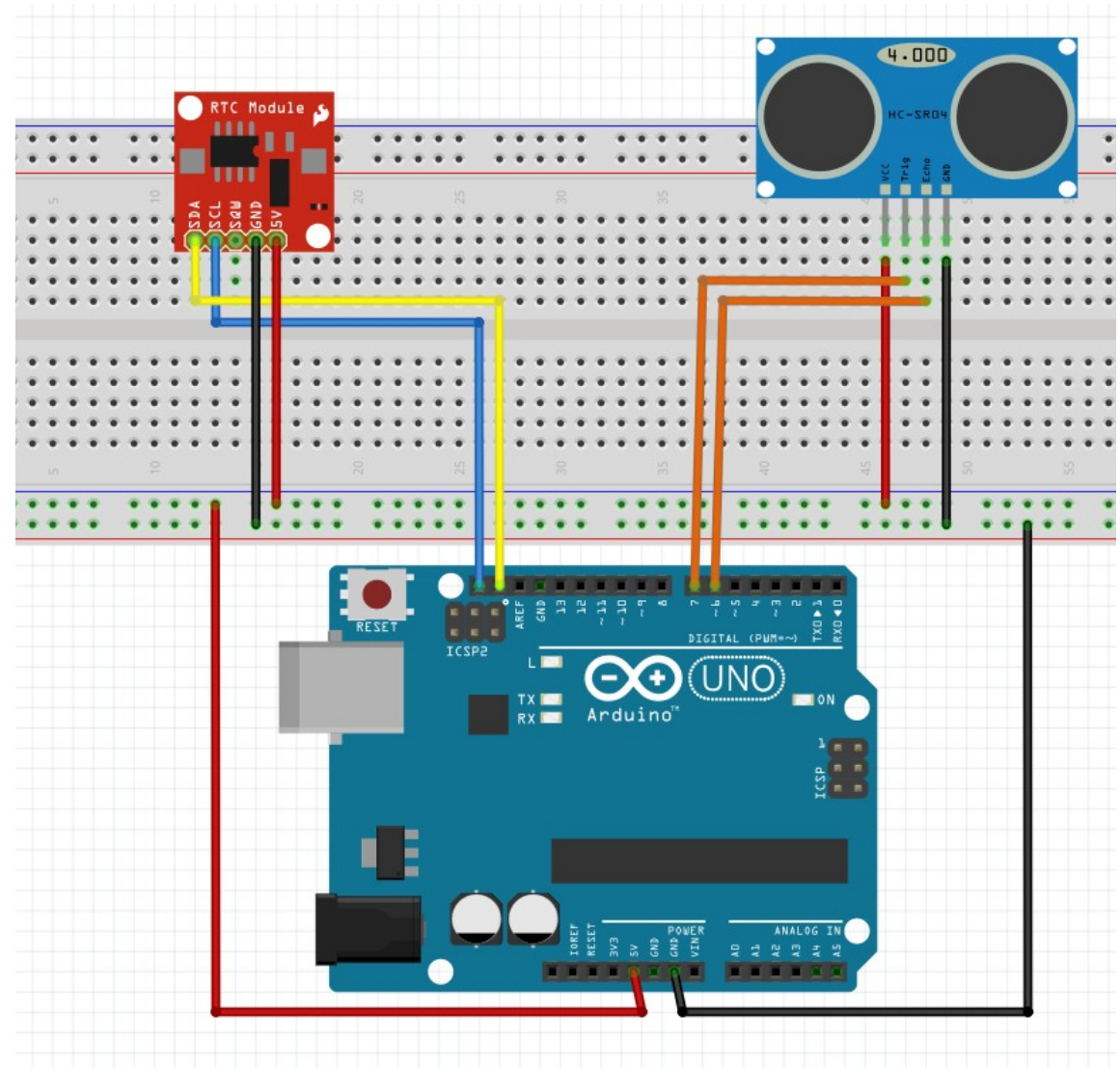
Per usare questo codice bisogna scaricare la libreria "ds3231FS"



```
#include <Wire.h>
#include <ds3231.h>
struct ts t;
void setup() {
  Serial.begin(9600);
  Wire.begin();
  t.hour=11;
  t.min=12;
  t.sec=0;
  t.mday=06;
  t.mon=03;
  t.year=2022;
  DS3231_set(t);
}
void loop() {
  DS3231_get(&t);
  Serial.print("");
  Serial.print(t.mday);
  Serial.print(",");
  Serial.print(t.mon);
  Serial.print(",");
  Serial.print(t.year);
  Serial.print(",");
  Serial.print(t.hour);
  Serial.print(",");
  Serial.print(t.min);
  Serial.print(",");
  Serial.println(t.sec);
  delay(1000);
```

# Quarto laboratorio: terzo circuito

- Componenti:
- Una Breadboard
- Un Arduino Uno
- Un sensore di distanza(ultrasonic sensor )
- Un rtc(real time clock) module



# Codice terzo circuito

Questo codice è l'unione dei due codici precedenti



```
File Modifica Sketch Strumenti Aiuto
ultrasonic_sensor_rtc_module $
#include <Wire.h>

#include <Wire.h>
#include <ds3231.h>

const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 6; // Echo Pin of Ultrasonic Sensor
struct ts t;

void setup() {
  Serial.begin(9600);
  Wire.begin();
  t.hour=9;
  t.min=30;
  t.sec=0;
  t.mday=15;
  t.mon=03;
  t.year=2022;

  DS3231_set(t);
}

void loop() {
  DS3231_get(&t);
  long duration, cm;
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delay(10);
  digitalWrite(pingPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  cm = duration * 0.0343 / 2;
  Serial.print("Distance: ");
  Serial.println(cm);
  delay(1000);
}
```

26 Arduino Uno su COM3

Grazie per la visione

Fine