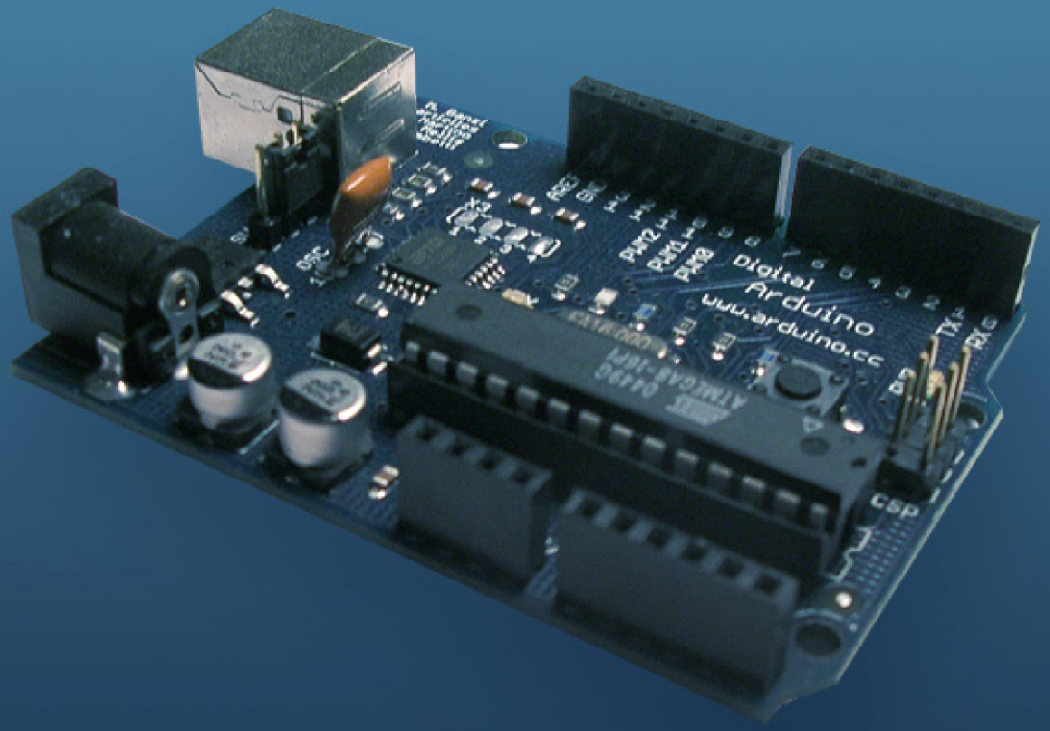
Arduino

Physical Computing I/O board



5^ parte : Controllare un LED RGB con 3 potenziometri

Author: Ing. Sebastiano Giannitto (ITIS “M.BARTOLO” –PACHINO)

Esperienza n° 3

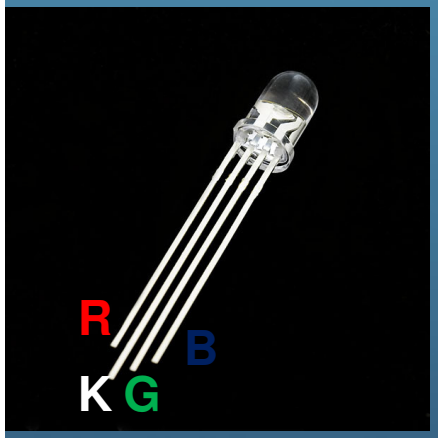
Lo scopo è riuscire a controllare i tre canali di un led RGB attraverso dei potenziometri connessi alle porte analogiche del nostro Arduino.

Occorrente:

1. Arduino Uno
2. Breadboard
3. Un led RGB
4. Resistenze
5. 3 potenziometri (uno per ogni colore emesso dal led)

cavi di collegamento

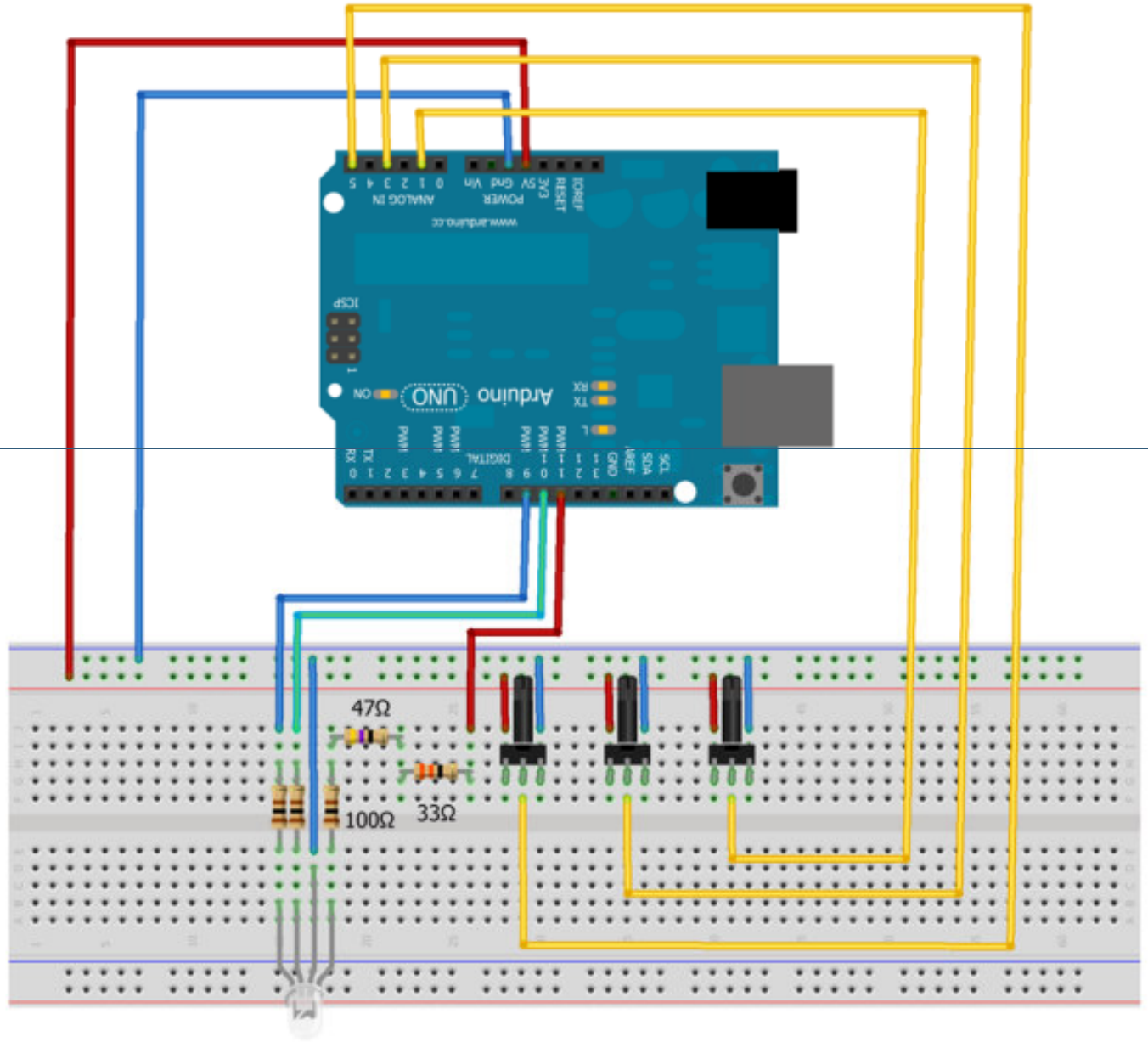
Soluzione



Un led RGB ha quattro pin: il più lungo è come al solito il catodo, mentre gli altri tre sono per i rispettivi colori. Sfruttando la PWM (modulazione di larghezza di impulso) è possibile controllare l'intensità di ogni singolo canale, in modo da poter creare tutti i colori possibili mischiando tra di loro rosso, verde e blu.

Anche in questo caso vale lo stesso discorso per il potenziometro fatto nella guida precedente. I pin analogici leggono valori da 0 a 1023, mentre la funzione **analogWrite (LED,val)** con cui impostiamo la PWM, assume come secondo parametro un numero da 0 a 255. Bisogna dunque rapportare questi due intervalli in modo che girando il potenziometro il valore di fondo scala combacia con il valore massimo o minimo di intensità del colore. Costruiamo il circuito!

Schema e Circuito elettrico su breadboard

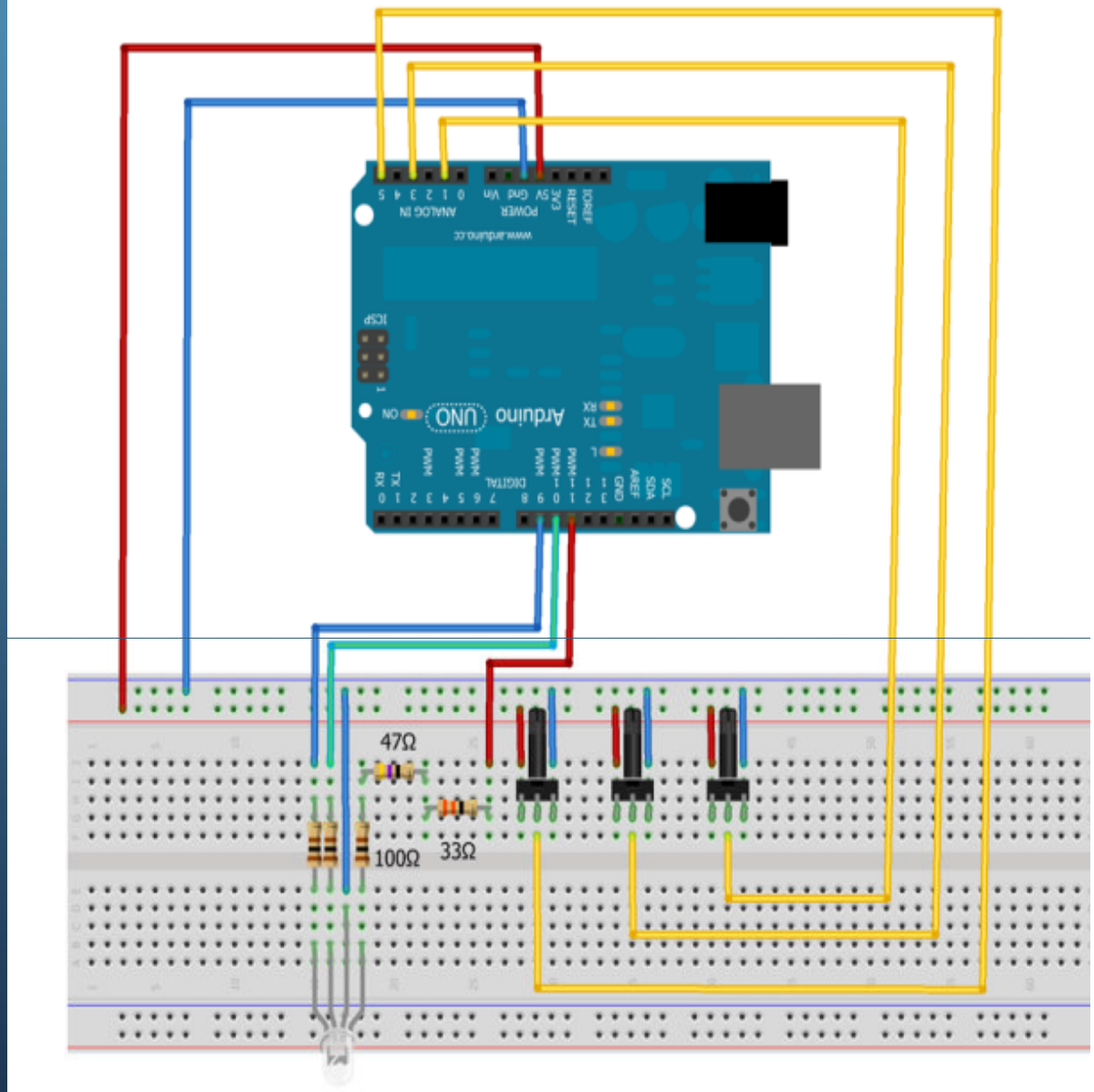


Schema creato con Fritzing

http://fritzing.org/

Collegate i potenziometri ai +5V del vostro Arduino e alla GND (terra) e poi a tre pin analogici a vostra scelta.

Come si può vedere dallo schema i pin dei colori del led RGB sono collegati alle porte digitali inserendo in serie resistenze da 100Ω per il blu e per il verde, mentre per il rosso sono state inserite 3 resistenze in serie per un totale di 180Ω = 100Ω + 47Ω + 33Ω (la resistenza singola da 180Ω è poco comune).



Il codice

```
// dichiariamo la posizione dei pin digitali per il led RGB
    #define RED 11
    #define GREEN 10
    #define BLUE 9
// e anche di quelli analogici per i potenziometri
    #define POT1 1
    #define POT2 3
    #define POT3 5
// inizializziamo tre variabili intere che conterranno i valori
// letti sui potenziometri
    int val1 = 0;
    int val2 = 0;
    int val3 = 0;
```

Il codice

```
void setup()
{
// rendiamo i pin digitali degli output
pinMode(RED, OUTPUT);
pinMode(GREEN, OUTPUT);
pinMode(BLUE, OUTPUT);
}
```

Il codice

```
void loop()
{
// leggiamo la posizione attuale di ogni potenziometro
val1 = analogRead(POT1);
val2 = analogRead(POT2);
val3 = analogRead(POT3);
// dividiamo il valore letto sui potenziometri per 4.01176 = 1023 / 255
//(int) serve per troncare il risultato della divisione e considerare solo la
//parte intera senza arrotondamento, questo tipo di operazione si
//chiama CAST
val1 = (int) (val1/4.01176);
val2 = (int) (val2/4.01176);
val3 = (int) (val3/4.01176);
// impostiamo l'intensità dei tre colori del led RGB sfruttando la PWM
analogWrite(RED, val1);
analogWrite(GREEN, val2);
analogWrite(BLUE, val3);
}
```

A partire da questo semplice schema è ora possibile trasformare la scheda Arduino in una lampada, costruendo magari un struttura in carta, tipo origami, per diffondere meglio la luce del led RGB, che come potete vedere dal video è molto luminoso.
Oppure è possibile implementare degli algoritmi nel codice per cambiare il colore della luce seguendo un pattern o modulare la luce in base alla temperatura o qualsiasi altra cosa vi passi per la mente.
Guardare il video su Youtube

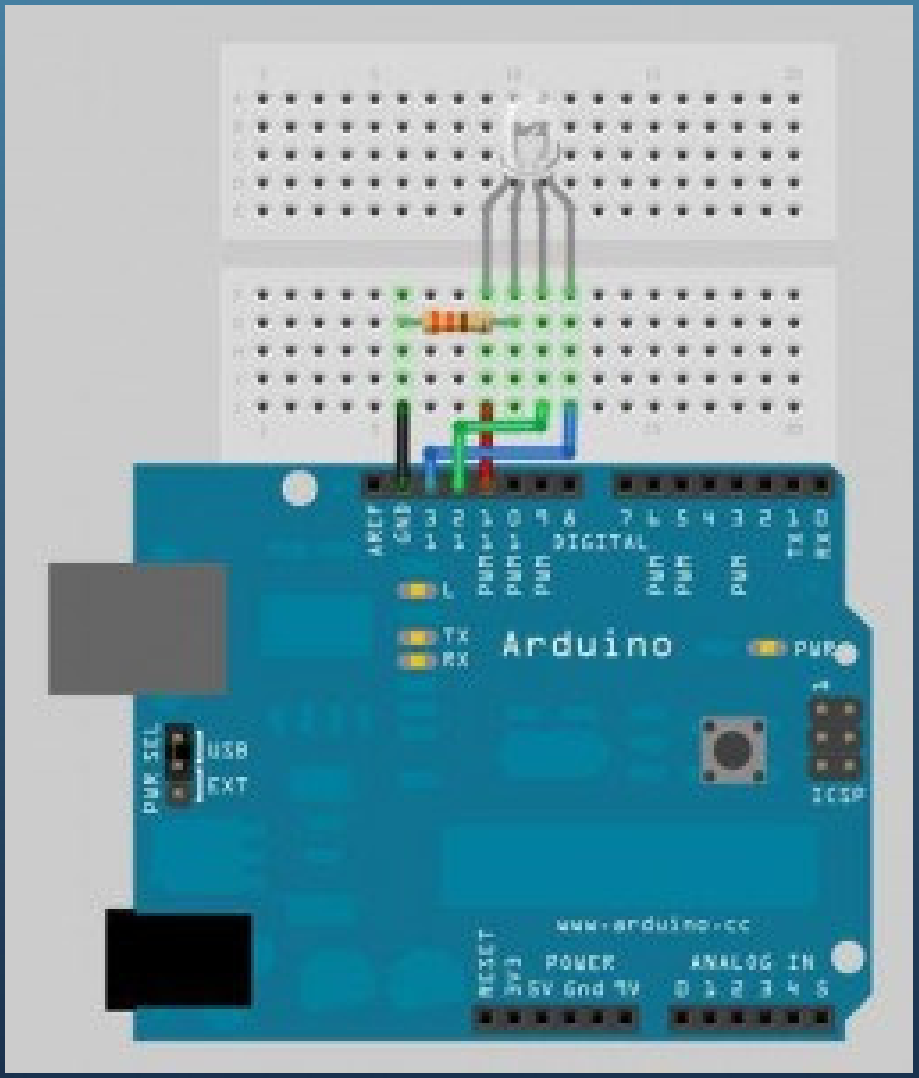
http://www.youtube.com/watch?feature=player_embedded&v=8nRMDVXZ0W4

Esperienza n° 4

Dal seguente sketch ricavare il circuito per pilotare un led RGB avente al catodo un resistore di 330 Ω

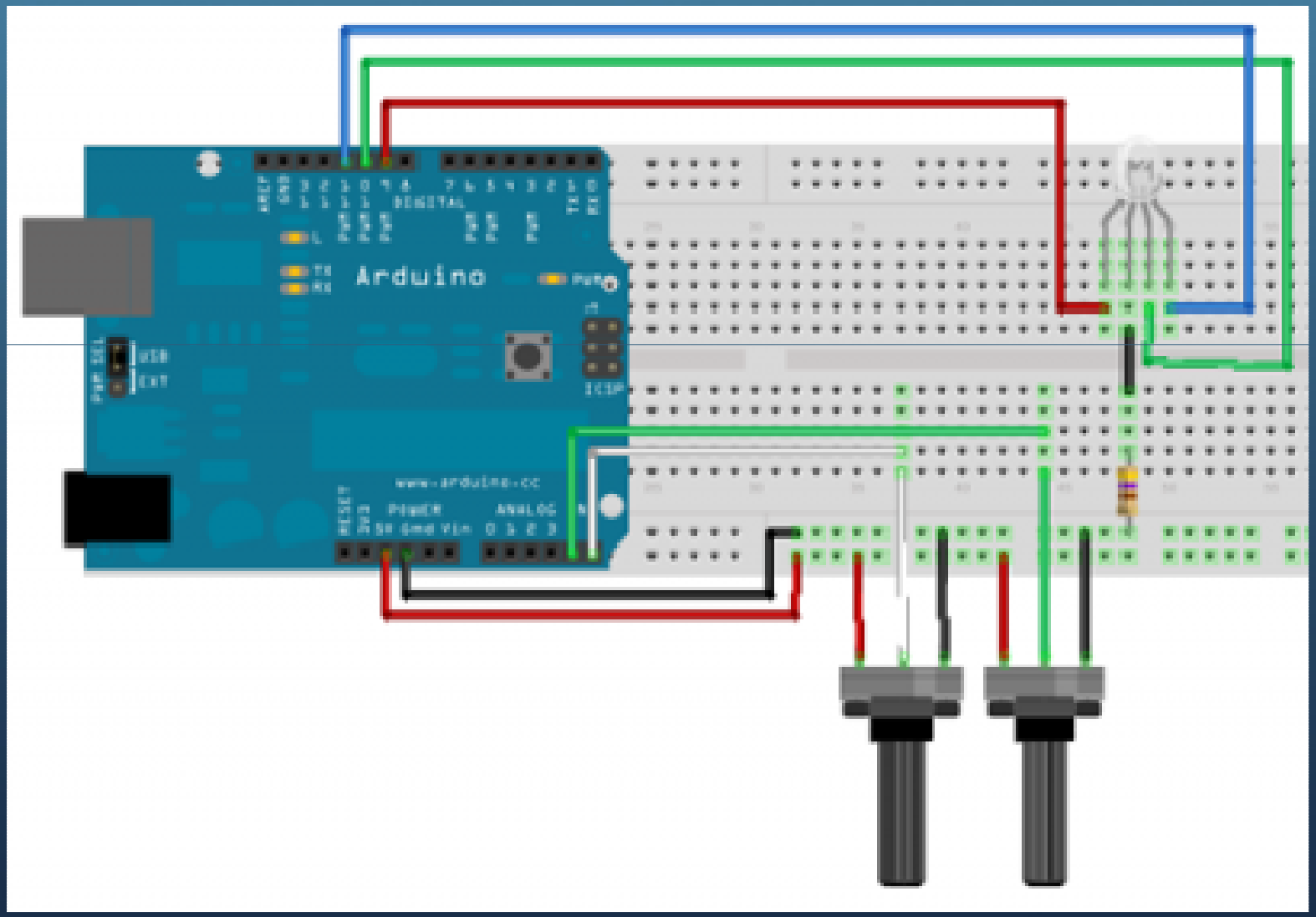
```
int a = 1000; //tempo millisecondi di //mantenimento colore
int red = 11; //pin per il led rosso
int green = 12; //pin per il led verde
int blue = 13; //pin per il led blu
void setup()
{
//impostazione pin come uscita
pinMode(red, OUTPUT);
pinMode(green, OUTPUT);
pinMode(blue, OUTPUT);
}
void loop() {
//ciclo loop
analogWrite(blue, random(255)); //la funzione random genera dei numeri casuali in modo da creare dei colori casuali
analogWrite(red, random(255));
analogWrite(green, random(255));
 delay(a);
}
```

Soluzione


DIGITAL
Arduino
PWR
USB
EXT
PWR SEL
ICSP
www.arduino.cc
POWER
ANALOG IN

Esperienza n° 5

Provare il programma per pilotare un led RGB con 2 potenziometri



Codice

```
int ledR = 9;
int ledG = 10;
int ledB = 11;
```

definisci 3 variabili di tipo *integer* per indicare i pin di OUTPUT con cui piloterai i 3 colori del led distintamente;

```
int potA = A5;
int potB = A4;
```

definisci altre due variabili di tipo *integer* per indicare i pin di INPUT analogico A5 e A4 a cui sono collegati i due potenziometri;

```
void setup()
{
  pinMode(ledR,OUTPUT);
  pinMode(ledG,OUTPUT);
  pinMode(ledB,OUTPUT);
```

Si indica ad arduino che i tre pin assegnati alle variabili ledR, ledG e ledB sono da utilizzare come OUTPUT;

```
  Serial.begin(9600);
}
```

imposta la comunicazione seriale tra Arduino e il tuo computer o mac a 9600 baud, userai questa connessione per vedere a video i valori letti dai potenziometri e tradotti in scala 0-255 per i led;

Codice

Con il comando *analogRead(pin)*, leggi il valore dal potenziometro.Come sai il valore letto varia da 0 a 1023, ossia 1024 step, tuttavia al led questa informazioni sarebbe superflua perchè il valore in OUTPUT possibile sui pin PWM di arduino è 0-255, se non facessi nulla otterresti che nel primo quarto, circa, di rotazione il led varia la sua luminosità e dal valore 255 in poi resterebbe sempre alla massima intensità luminosa.;

```
void loop()
{
  int valA = map(analogRead(potA),0,1023,0,255);
  int valB = map(analogRead(potB),0,1023,0,255);

  Serial.print(" Potenziometro A: ");
  Serial.print( valA );
  Serial.print(" Potenziometro B: ");
  Serial.println( valB );

  analogWrite(ledR,valA);
  analogWrite(ledG,valB);
  analogWrite(ledB,valB);
}
```

Per correggere questa differenta tra INPUT ed OUTPUT arduino ti mette a disposizione la funzione *map(valore,minInput,maxInput,minOutput,maxOutput)* con cui traslare, più o meno in modo lineare, i valori letti su un pin analogico dalla scala 0-1023 alla scala 0-255; potresti fare questa proporzione con dei calcoli e crearti tu una funzione, ma arduino ti semplifica la vita e te ne da una già fatta e collaudata;

Con la funzione *analogWrite()* scrivi il vaore del potenziometro A sul led Rosso ed il valore del potenziometro B sui led Green e Blu.