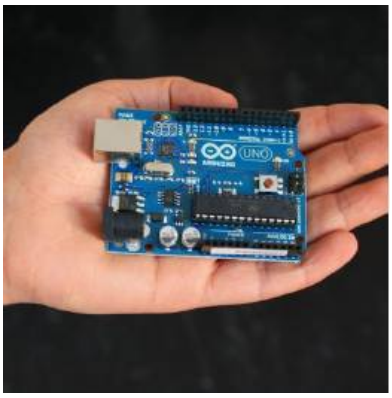
Hackathon SMD 24

Secondaria di I grado

Arduino

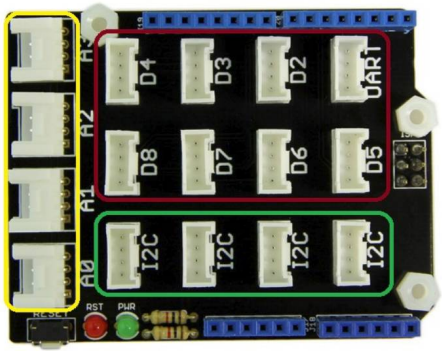
Arduino è una piattaforma di prototipizzazione elettronica open-source basata su un hardware ed un software flessibile e di facile utilizzo.E' pensato per creativi, progettisti, hobbysti e per chiunque sia interessato a creare oggetti o ambienti interattivi.



Arduino può percepire l'ambiente ricevendo gli ingressi da una varietà di sensori e può influenzare l'ambiente circostante controllando luci, motori e altri attuatori. Il microcontrollore della scheda è programmato usando il linguaggio di programmazione "Arduino" e l'ambiente di sviluppo "Arduino". I progetti basati su Arduino possono essere a se stanti (stand-alone), oppure possono comunicare con software in esecuzione su un computer. Le schede possono essere autocostruite oppure acquistate pre-assemblate, il software può essere scaricato gratuitamente. Gli schemi di progettazione hardware (file CAD) sono disponibili sotto una licenza open-source e possono essere adattati alle proprie necessità. Arduino è stato sviluppato da Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis nell'ambito dell' "Interaction Design Institute" di Ivrea.

Grove Starter Kit IoT Edition

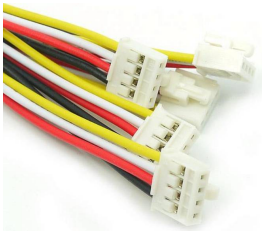
La scheda Grove si collega a un Arduino e costituisce la base del sistema Grove. Tutte le porte I/O di Arduino sono esposte e adattate a 22 connettori Grove che includono I/O digitale, I/O analogico e porte specializzate.



Al centro, circondate dalla linea rossa, ci sono 13 porte I/O digitali. Queste possono essere utilizzate per leggere e controllare i moduli Grove digitali, come il sensore di luce e i LED. Alcune delle porte I/O digitali possono essere utilizzate anche come uscite PWM (modulazione di larghezza di impulso). Generando onde PWM, Arduino può controllare il movimento di un motore o accendere un LED con luminosità variabile. All'interno della linea verdei, sul lato sinistro, ci sono 5 porte di ingresso analogiche. Gli ingressi analogici vengono generalmente utilizzati per leggere sensori analogici, come un potenziometro o un sensore di temperatura, ma queste porte possono essere utilizzate anche come porte I/O digitali. Infine, le porte specializzate sono delineate inverde: sono quattro porte I2C da utilizzare con moduli Grove più complessi come l'accelerometro a 3 assi.

Collegamenti

I sensori ed attuatori compatibili Grove contenuti nel kit utilizzano una porta digitale, una porta analogica o una porta I2C. Nel kit sono inclusi dieci cavi Grove. Collegare i cavi Grove ai sensori e alla scheda di base nella sezione corrispondente al tipo di sensore (digitale, analogico, I2C). Ciò consente operazioni plug and play senza saldatura.



Sensori ed attuatori

Grove- Button: pulsante normalmente aperto. Fornisce un ingresso digitale che normalmente vale 0 e diventa 1 quando viene premuto. Va collegato ad una porta digitale tra D2 e D8

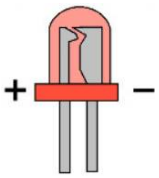
Grove - Touch Sensor: funzionalmente identico al precedente ma sensibile al tocco.

Grove - Rotary Angle Sensor: potenziometro rotante con un angolo di 270°. Fornisce un ingresso analogico proporzionale all'angolo di rotazione. Il valore analogico viene convertito in un valore digitale compreso tra 0 e 1023. Va collegato ad un ingresso analogico da A0 ad A3.

Grove - Red LED: LED rosso. Va collegato ad una uscita digitale tra D2 e D8. Quando l'uscita vale 0 il LED è spento; quando l'uscita vale 1 il LED è accesso. Collegando il LED alle porte D3, D5 e D6 è possibile accendere il LED con una intensità proporzionale al valore inviato in uscita compreso tra 0 e 255.

Grove - Blue LED: come sopra di colore blu

Grove - Green LED: come sopra di colore verde



Attenzione! I LED sono smontati e vanno montati con il corretto orientamento: osservando in trasparenza il LED si vedono due elettrodi: quello più piccolo è il polo positivo e va inserito nel pin "+" della scheda.

Grove – Buzzer: cicalino che va collegato ad una uscita tra D2 e D8. Emette un tono fisso quando l'uscita vale 1 mentre è silezioso quando l'uscita vale 0.

Grove - Sound Sensor: sensore analogico di suono. Fornisce un ingresso analogico proporzionale all'intensità di suono rilevato. Il valore analogico viene convertito in un valore digitale compreso tra 0 e 1023. Va collegato ad un ingresso analogico da A0 ad A3.

Grove - Light Sensor:sensore analogico di luce. Fornisce un ingresso analogico proporzionale all'intensità di luminosità rilevata. Il valore analogico viene convertito in un valore digitale compreso tra 0 e 1023. Va collegato ad un ingresso analogico da A0 ad A3.

Grove - Temperature Sensor: sensore analogico di temperatura. Fornisce un ingresso analogico correlato ma non proporzionale alla temperatura ambiente rilevata. Il valore analogico viene convertito in un valore digitale compreso tra 0 e 1023. Per ottenere la temperatura si deve applicare la formula di linearizzazione. Vedi esempio. Va collegato ad un ingresso analogico da A0 ad A3.

Grove - LCD RGB Backlight: display alfanumerico (2 righe, 16 colonne) retroilluminato con colori RGB. Va collegato ad una porta I2C. Si può impostare il colore di sfondo, posizionare il cursore in una qualsiasi riga/colonna, scrivere o sovrascrivere un testo, cancellare il display.

Grove - 3-Axis Digital Accelerometer(±1.5g): accelerometro a 3 assi. Va collegato ad una porta I2C. Fornisce l'accelerazione lungo tre assi x,y,z. Quando è in moto accelerato fornisce il valore di accelerazione su ciascun asse espresso in "g" cioè valore rispetto all'accelerazione di gravità. Quando è a risposo solo l'asse che punta verso il basso restituisce 1 g e gli altri danno 0 g.

Grove – Servo motor: servo motore analogico. Va collegato ad una delle porte D3, D5, D6 che possono fornire in uscita un valore analogico PWM. In corrispondenza di tali valori in motore ruota di un angolo proporzionale al valore che può andare da -180° a +180°

Tensione di alimentazione

Durante lo sviluppo non è necessario collegare l'alimentatore perché la scheda Arduino+Grove prende l'alimentazione dalla porta USB utilizzata per caricare i programmi.

Lo switch 3.3V-VCC-5V sulla scheda Grove deve essere in posizione 5V.

Ambienti di sviluppo

I programmi di Arduino vanno caricati nella memoria della scheda ma vanno scritti sul computer utilizzando un programma che si chiama "ambiente di sviluppo".
Sì possono sviluppare programmi sia usando un linguaggio di programmazione testuale si un programma di codifica a blocchi.
Per lo sviluppo in linguaggio testuale è consigliato l'ambiente di sviluppo IDE di Arduino mentre per la programmazione a blocchi è consigliato l'ambiente di sviluppo Codecraft.
L'IDE di Arduino va scaricato all'url: https://www.arduino.cc/en/software
ed installato.
Codecraft è un'applicazione web: https://ide.tinkergen.com/
tuttavia il caricamento del programma sulla scheda richiede l'installazione di un programma che va scaricato dalla pagina stessa.

TEST

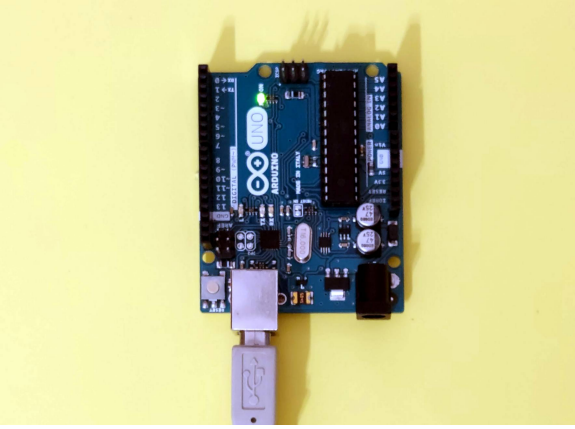
La programmazione viene anche chiamata "sviluppo" perché per evitare errori si costruisce il programma un componente alla volta testando ogni parte separatamente introducendola nel progetto solo quando è sicuramente priva di errori.
I seguenti sono test che verificano il funzionamento di tutte le parti del kit.
Dopo aver testato i componenti è possibile integrarli tra loro costruendo applicazioni complesse.

TEST01 – Salve mondo: solo scheda Arduino; test della funzionalità dell'hardware
TEST02 – Pulsante – LED – Stato
TEST03 – Pulsante – LED – Evento
TEST04 – Potenziometro – LED PWM
TEST05 – Pulsante – Buzzer attivo
TEST06 – Sensore di suono – LED - Soglia
TEST07 – Sensore di luminosità – LED - Soglia
TEST08 – Sensore di temperatura - Seriale
TEST09 – Display – Salve mondo - Timer
TEST10 – Accelerometro – Seriale
TEST11 – Servo sweep

TEST01 – Salve mondo

Il test "salve mondo" serve per verificare la funzionalità dell'hardware facendogli eseguire un codice elementare. Nel caso della scheda Arduino il test fa lampeggiare il LED integrato nella scheda accessibile sulla porta 13.

Schema di montaggio



Effettuare il test senza montare la scheda Grove.

Versione C++

```
void setup() {
  pinMode(13, OUTPUT);     //Inzializza la porta 13 come uscita
}
void loop() {
  digitalWrite(13, HIGH); //Accende il LED (HIGH=1)
  delay(1000);            //Aspetta 1 secondo
  digitalWrite(13, LOW);  //Spegne il LED  (LOW=0)
  delay(2000);            //Aspetta 2 secondi
}
```

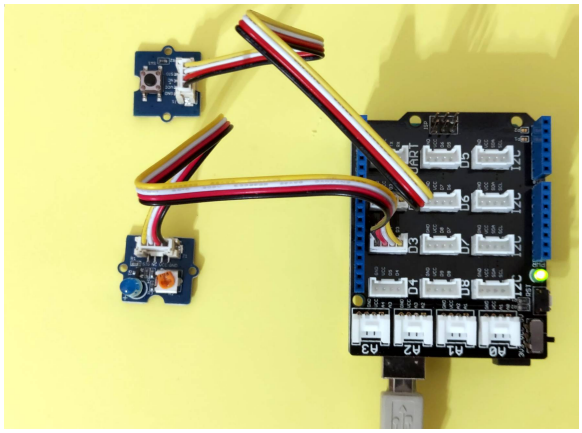
Versione Codecraft

N.B.: in Codecraft non è necessario inizializzare la porta come uscita perché il compilatore provvede a farlo automaticamente in base al tipo di operazione fatta sulla porta.
Si può verificare il codice generato facendo click sul bottone **</>**

TEST02 – Pulsante – LED - stato

Comando di un LED con un pulsante sensibile allo stato
Quando il pulsante sulla porta 2 è premuto (HIGH) il LED sulla porta 3 è acceso (HIGH)

Schema di montaggio



Montare la scheda Grove sulla scheda Arduino facendo attenzione che tutti i pin siano correttamente inseriti.
Collegare il pulsante alla porta D2 ed il LED alla porta D3 facendo attenzione al verso del LED (l'elettrodo più piccolo visto in trasparenza va nel pin +).

Versione C++

```
void setup() {
  pinMode(2, INPUT); //Inizial.la porta 2 come ingresso (pulsante)
  pinMode(3, OUTPUT);//Inizial.la porta 3 come uscita (LED)
}
void loop() {
  if(digitalRead(2)==HIGH) {     //se pulsante premuto
    digitalWrite(3, HIGH);       //Accende il LED
  }
  else {
    digitalWrite(3, LOW);        //Spegne il LED
  }
}
```

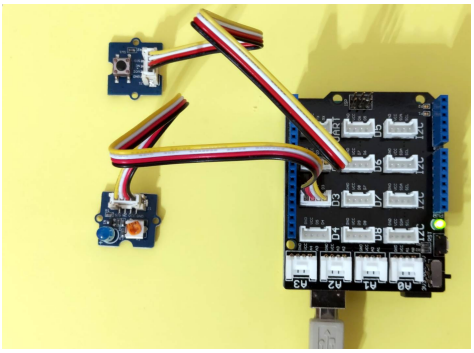
Versione Codecraft

N.B. Sebbene sia possibile utilizzare in Codecraft anche le istruzioni standard di Arduino, per maggior chiarezza, è preferibile quando possibile utilizzare le istruzioni delle librerie Grove.

TEST03 – Pulsante – LED - evento

Comando di un LED con un pulsante sensibile all'evento di pressione (passaggio da LOW ad HIGH)
Quando il pulsante sulla porta 2 viene premuto il LED effettua un lampeggio di mezzo secondo.
Si deve rilasciare il pulsante e premere nuovamente per un successivo lampeggio.
Il riconoscimento di un evento è importante quando si devono fare dei conteggi.

Schema di montaggio

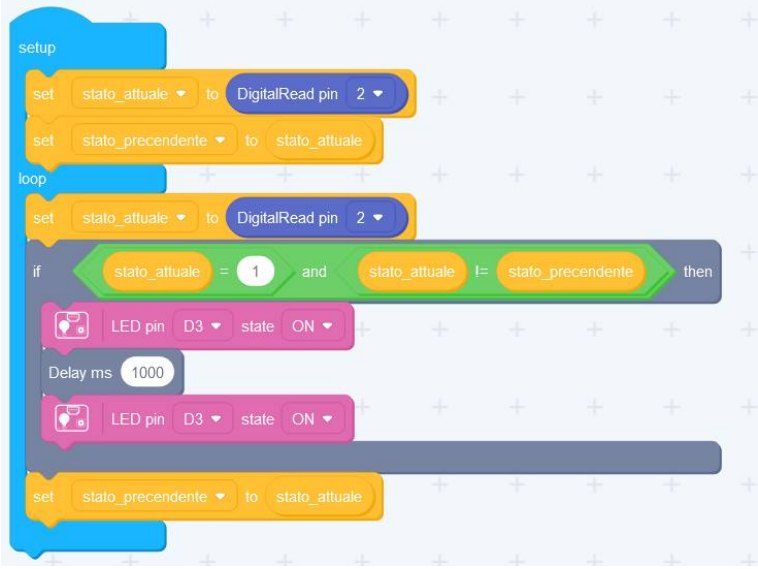


Lo schema di montaggio è identico al precedente test 02.

Versione C++

```
int stato_attuale;    //stato attuale del pulsante
int stato_precedente; //stato del pulsante nel loop precedente
void setup() {
  pinMode(2, INPUT);  //Iniz.la porta 2 come ingresso (pulsante)
  pinMode(3, OUTPUT); //Iniz.la porta 3 come uscita (LED)
  stato_attuale=digitalRead(2);     //inizializza lo stato attuale
  stato_precedente=stato_attuale;   //forza "nessun evento"
}
void loop() {
  stato_attuale=digitalRead(2); //legge il nuovo stato attuale
  if((stato_attuale==HIGH)&&(stato_precedente==LOW)) {
     //se stato att.HIGH E stato prec.LOW: evento pressione
    digitalWrite(3, HIGH);      //Accende il LED
    delay(500);
    digitalWrite(3, LOW);       //Spegne il LED
  }
  stato_precedente=stato_attuale; //agg.stato prec. Con stato att.
}
```

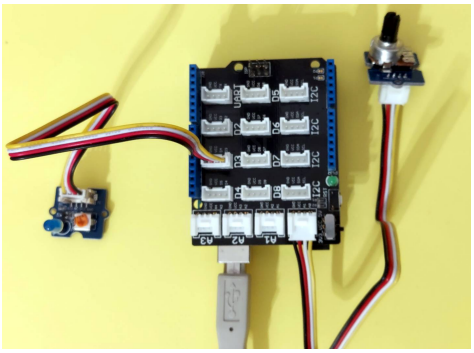
Versione Codecraft



TEST04 – Potenziometro - LED

comando di un LED in PWM con un potenziometro
Il potenziometro collegato ad A0 fornisce in ingresso una tensione variabile che viene trasformata in un valore digitale tra 0-1023
il valore viene convertito in PWM in range tra 0-255 dividendolo per 4 ed inviato al LED che si illumina con una intensità variabile

Schema di montaggio

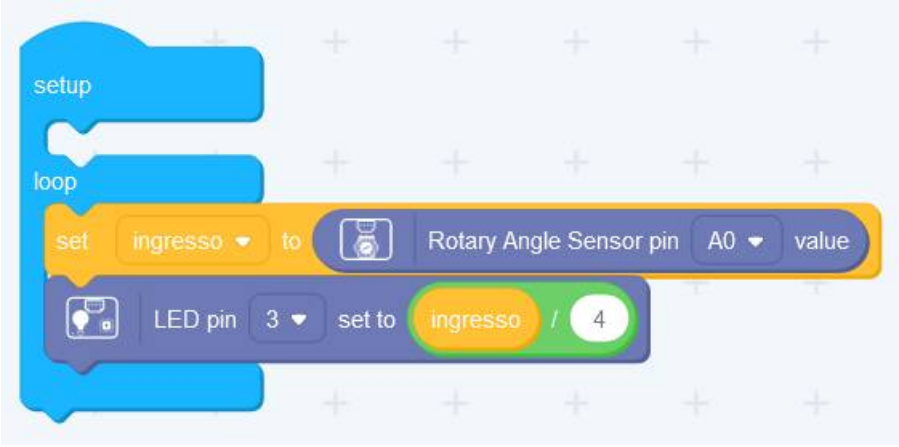


Il LED rimane collegato a D3.
Al posto del pulsante si monta il potenziometro (Rotary angle sensor) collegato ad A0.

Versione C++

```
int ingresso;           //valore letto dal potenziometro
void setup() {
  pinMode(3, OUTPUT);  //Inizializza la porta 3 come uscita (LED)
}
void loop() {
  ingresso=analogRead(A0);   //legge val. ingresso (0-1023)
  analogWrite(3,ingresso/4); //emette valore PWM su LED (0-255)
}
```

Versione Codecraft

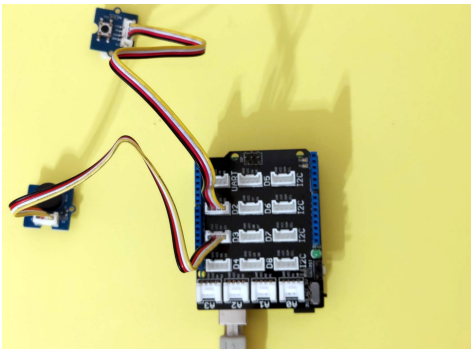


N.B. il comando LED è di colore diverso rispetto a quelli usati in precedenza perché in questo caso si tratta di un comando PWM con valori da 0 a 255
Il range del dato in ingresso è 0-1023 mentre quello del dato in uscita 0-255 quindi il valore va compresso dividendo per 4.

TEST05 – Pulsante – buzzer attivo

Comando di un buzzer attivo con un pulsante sensibile allo stato
Quando il pulsante sulla porta 2 è premuto il buzzer suona con un tono predefinito

Schema di montaggio



Il pulsante va collegato a D2.
Il buzzer va collegato a D3

Versione C++

```
void setup() {
  pinMode(2, INPUT);  //Iniz.la porta 2 come ingresso (pulsante)
  pinMode(3, OUTPUT); //Iniz.la porta 3 come uscita (buzzer)
}
void loop() {
  if(digitalRead(2)==HIGH) {     //se pulsante premuto
    digitalWrite(3, HIGH);       //Accende il buzzer
  }
  else {
    digitalWrite(3, LOW);        //Spegne il buzzer
  }
}
```

Versione Codecraft

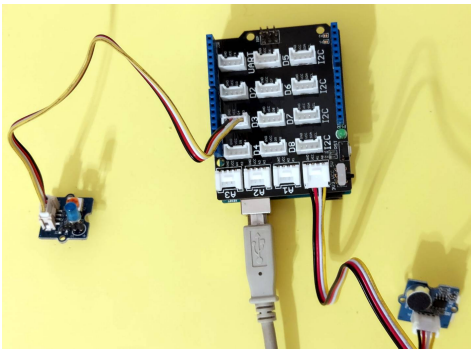
TEST06 – sound sensor-LED

comando di un LED con un sensore di suono

Il sensore di suono collegato ad A0 fornisce in ingresso una tensione variabile che viene trasformata in un valore digitale tra 0-1023

Se il valore supera una soglia regolabile in base alla rumorosità dell'ambiente viene attivato il LED.

Schema di montaggio



Il LED va collegato a D3.

Il sensore di suono va collegato ad ad A0.

Versione C++

```
int suono;
void setup() {
  Serial.begin(9600);
  pinMode(3, OUTPUT); //Iniz. la porta 3 come uscita (LED)
}
void loop() {
  suono=analogRead(A0);     //legge l'intensità del suono (0-1023)
  Serial.println(suono);
  if (suono>120) {          //se suono supera soglia accende il LED
    digitalWrite(3,HIGH);
  }
  else {                    //altrimenti spegne il LED
    digitalWrite(3,LOW);
  }
}
```

Versione Codecraft

N.B. Per regolare la soglia è opportuno aprire un canale con il monitor seriale ed inviare al monitor seriale il valore letto dal sensore.

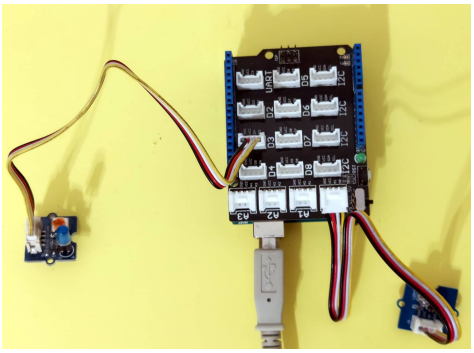
TEST07 – light sensor-LED

comando di un LED con un sensore di luminosità

Il sensore di luminosità collegato ad A0 fornisce in ingresso una tensione variabile che viene trasformata in un valore digitale tra 0-1023

Se il valore supera una soglia regolabile in base alla luminosità dell'ambiente viene attivato il LED.

Schema di montaggio



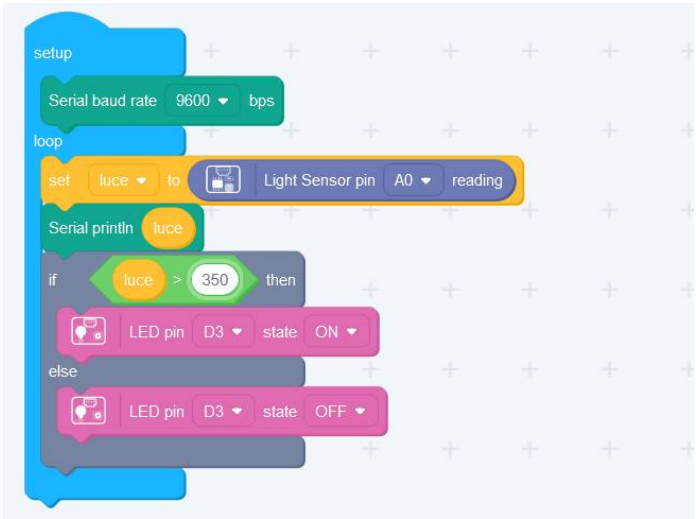
Il LED va collegato a D3.

Il sensore di luminosità va collegato ad ad A0.

Versione C++

```
int luce;
void setup() {
  Serial.begin(9600);
  pinMode(3, OUTPUT);    //Inizializza la porta 3 come uscita (LED)
}
void loop() {
  luce=analogRead(A0);  //legge l'intensità della luce (0-1023)
  Serial.println(luce);
  if (luce>350) {        //se supera soglia accende il LED
    digitalWrite(3,HIGH);
  }
  else {                          //altrimenti spegne il LED
    digitalWrite(3,LOW);
  }
}
```

Versione Codecraft

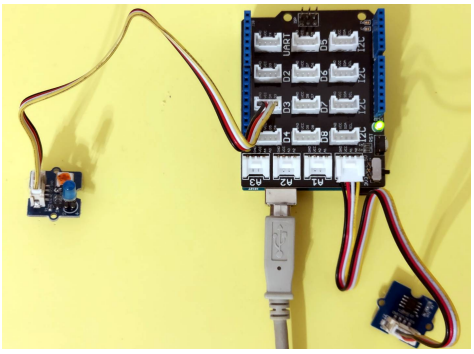


N.B. Per regolare la soglia è opportuno aprire un canale con il monitor seriale ed inviare al monitor seriale il valore letto dal sensore.

TEST08 – temperature sensor-serial

raccolta valori di temperatura inviati sul canale seriale
Il sensore di di temperatura NTC collegato ad A0 fornisce in ingresso una tensione variabile che viene trasformata in un valore digitale tra 0-1023.
Il legame tra temperatura e valore letto non è lineare quindi il dato grezzo va convertito con la formula 1/T = 1/T0 + 1/B * ln(R/R0)
Il sensore di luminosità collegato ad A0 fornisce in ingresso una tensione variabile che viene trasformata in un valore digitale tra 0-1023
Se il valore supera una soglia regolabile in base alla luminosità dell'ambiente viene attivato il LED.
Il dato convertito e messo in scala viene inviato al monitor seriale

Schema di montaggio

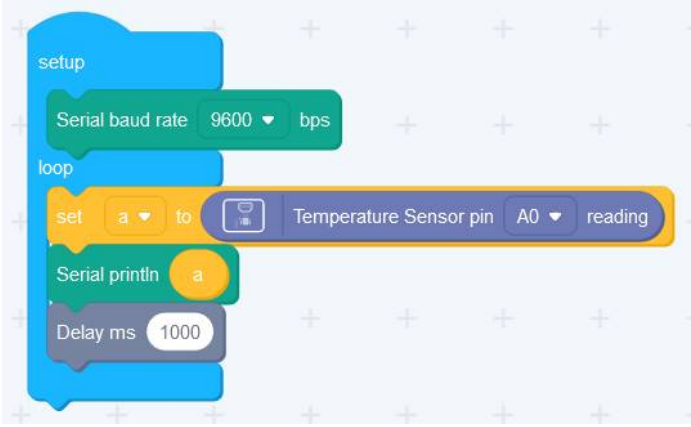


Il LED va collegato a D3.
Il sensore di temperatura va collegato ad ad A0.

Versione C++

```
const int B = 4275;     // valore B value del termisore
const int R0 = 100000; // R0 = 100k partitore di tensione
int a;                  //lettura analogica
float R;                //valore convertito
float t;                //temperatura in °C
void setup() {
  Serial.begin(9600);
  pinMode(3, OUTPUT);   //Inizializza la porta 3 come uscita (LED)
}
void loop() {
  a = analogRead(A0);   //legge la temperatura ambiente (0-1023)
  R = (1023.0/a-1.0)*R0;
  t = 1.0/(log(R/R0)/B+1/298.15)-273.15;
  Serial.println(t);
  delay(1000);
}
```

Versione Codecraft



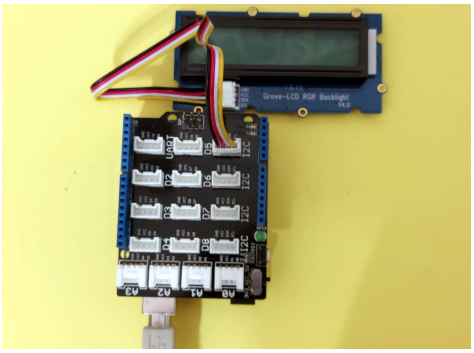
N.B. nella versione a blocchi non è necessario fare la linearizzazione perchè questa operazione è già fatta dal blocco.

TEST09 – display salve mondo timer

test di funzionalità del display I2C

Mostra sulla prima riga del display il messaggio Hello, world e sulla seconda riga il tempo passato dal reset in secondi.

Schema di montaggio



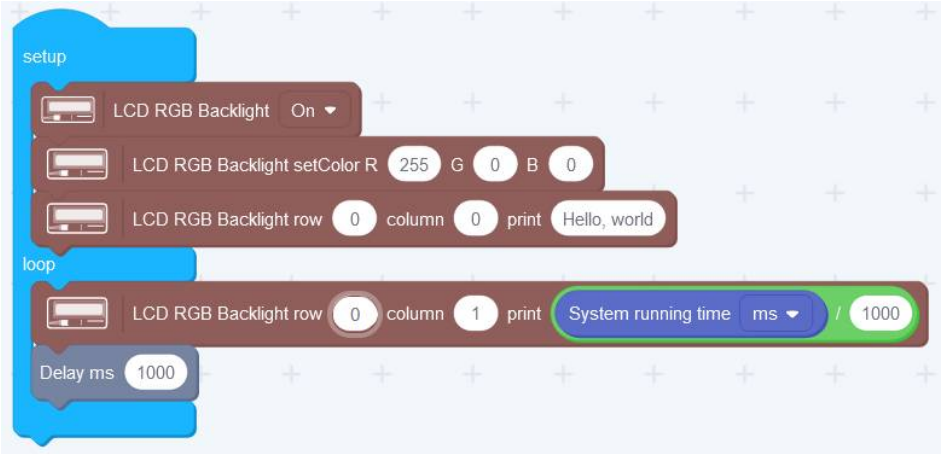
Il Display va collegato ad una qualsiasi porta I2C..

Versione C++

```
#include <Wire.h>          //libreria di base I2C
#include "rgb_lcd.h"       //libreria Grove LCD
rgb_lcd lcd;               //oggetto LCD
void setup() {
    lcd.begin(16, 2); //inizializza il display: 16 col., 2 righe
    lcd.setRGB(255,0,0); //imposta colore di sfondo RGB (rosso)
    lcd.print("Hello, world!");   //messaggio nella prima riga
    delay(1000);
}

void loop() {
    lcd.setCursor(0, 1);    //sposta cursore nella seconda riga
    lcd.print(millis()/1000); //stampa n° di secondi dal reset
    delay(1000);
}
```

Versione Codecraft

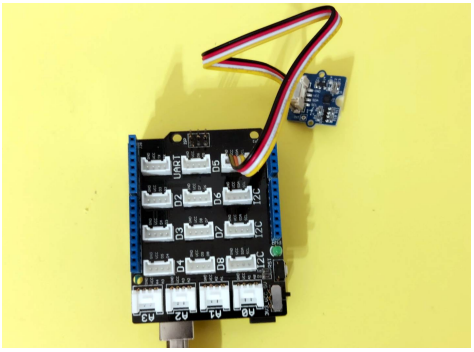


N.B. nel blocco che posiziona il testo in riga e colonna le righe e colonne sono invertite quindi l'istruzione utilizzata per scrivere nella prima colonna della seconda riga che dovrebbe essere: row 1 column 0 va invece scritta come row 0 column 1.

TEST10 – accelerometro seriale

test di rilevamento dell'orientamento con accelerometro
rileva l'accelerazione sui tre assi x,y,z e la mostra, espressa in g, sulla seriale.
con il dispositivo a riposo i dati indicano l'orientamento del dispositivo (l'asse z attraversa il piano del dispositivo)

Schema di montaggio



L'accelerometro va collegato ad una qualsiasi porta I2C..

Versione C++

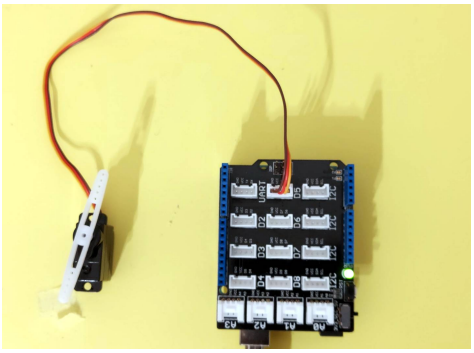
```
#include <Wire.h>
#include "MMA7660.h"
MMA7660 accelerometro;
float ax,ay,az; //accelerazione sui tre assi
void setup(){
  accelerometro.init();
  Serial.begin(9600);
}
void loop(){
  accelerometro.getAcceleration(&ax,&ay,&az);
  Serial.print("ax = ");
  Serial.print(ax);
  Serial.print(" ay = ");
  Serial.print(ay);
  Serial.print(" az = ");
  Serial.print(az);
  Serial.println();
  delay(1000);
}
```

Versione Codecraft

TEST11 – servo sweep

sweep di un servo collegato al pin 5
il servo si porta a 0°, fa una rotazione fino a 180° a passi di 1° poi torna a 0° a passi di 1°rileva l'accelerazione sui tre assi x,y,z e la mostra, espressa in g, sulla seriale.

Schema di montaggio



Il servo va collegato alla porta D5

Versione C++

```
#include <Servo.h>
Servo servo;           //oggetto servo
int pos = 0;           //posizione del servo

void setup() {
  servo.attach(5);     //servo collegato al pin 5
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { //da 0 a 180 a passi di 1
    servo.write(pos);                   //sposta di 1°
    delay(15);                          //tempo di rotazione
  }
  for (pos = 180; pos >= 0; pos -= 1) { //da 180 a 0 a passi di 1
    servo.write(pos);
    delay(15);
  }
}
```

Versione Codecraft