Analisi della caratteristica V-I del LED

Scopo dell'esperimento

Obiettivo dell'esperienza è realizzare un esperimento con acquisizione dati via Arduino finalizzato a registrare la curva caratteristica V-I di un LED a giunzione p-n in arseniuro di gallio.

Esecuzione dell'esperimento

Nell'esperimento la *d.d.p. V (in polarizzazione diretta)* applicata al diodo uscirà da un circuito RC (in modalità di integrazione) che riceve in ingresso il treno di impulsi generato dalla porta ~ 5V (PWM) di Arduino e fornisce in uscita una d.d.p. quasi-costante, proporzionale al *duty-cycle* del treno di impulsi. In altre parole questa prima parte del circuito serve per fornire al LED una differenza di potenziale variabile da 0 a 5 V.

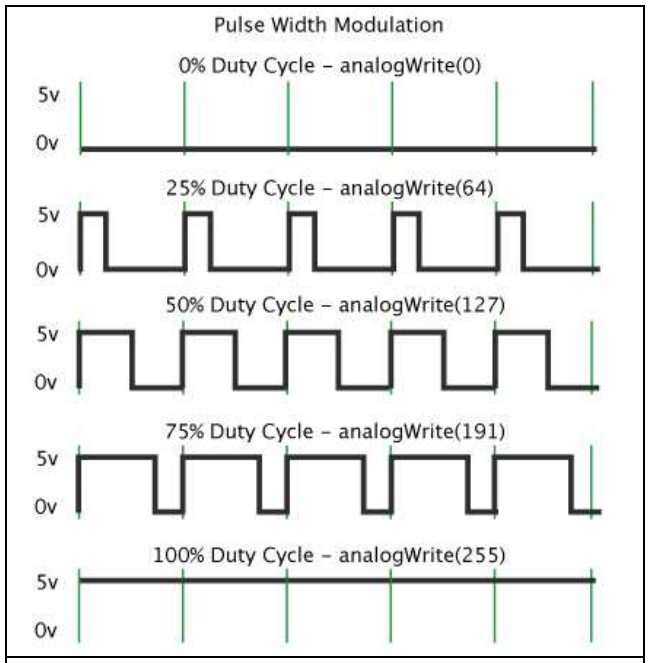

Figura 1

Nell'immagine di figura 1, prelevata dal sito ufficiale di Arduino, vediamo un diagramma chiarificatore. Il concetto è in realtà assai semplice: anziché utilizzare l'uscita 5 V continua di Arduino, si usano degli impulsi a 5 V più o meno ravvicinati andando perciò a simulare una tensione che va da 0 a 5V. Come vedete la linea temporale, è suddivisa in barrette verdi, che rappresentano il periodo. In Arduino la frequenza è di *1000 Hz* per cui ogni periodo dura circa *1 millisecondo*. All'interno del periodo abbiamo un impulso a 5 V che ha una durata temporale variabile (larghezza dell'impulso, *pulse width*), ed una parte in cui l'impulso si azzera.

Ultima cosa è il *duty cycle*, ossia la percentuale di tempo in cui il segnale è *acceso*. Un *duty cycle* del 100% indica una tensione continua a 5 V, un *duty cycle* allo 0% indica una tensione fissa a 0 V ed un duty cycle al 50% indica che per metà del periodo abbiamo una tensione a 5V e per l'altra metà a 0 V (in questo caso la tensione effettiva in uscita è 2.5 V).

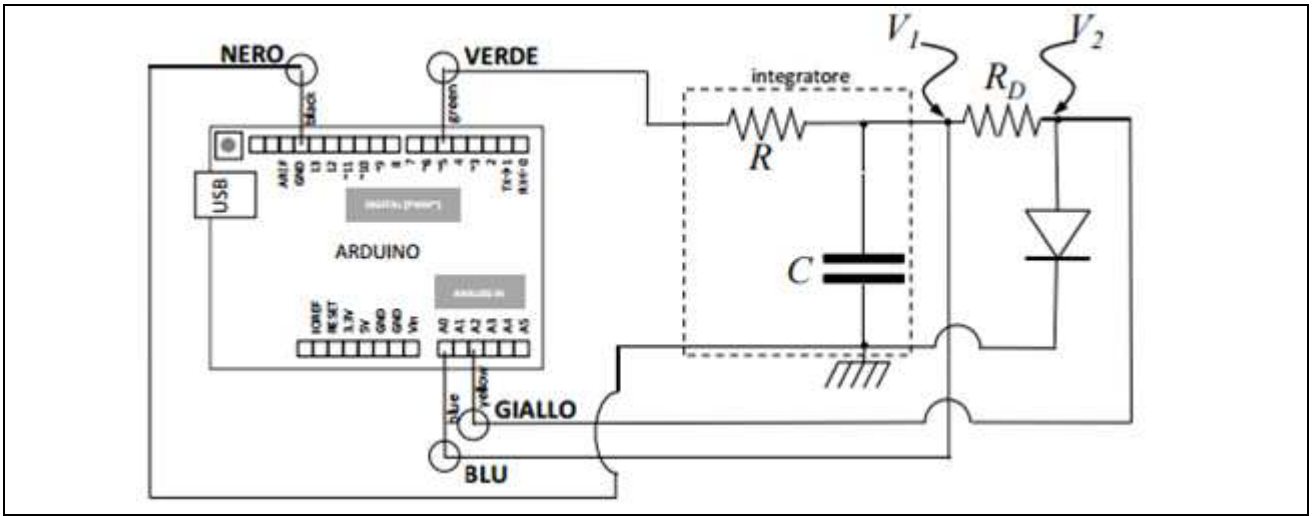

Figura 2 – Schema del circuito e collegamenti con Arduino (i fili a disposizione hanno colori diversi)

Nella scheda Arduino si può osservare che le uscite digitali 3, 5, 6, 9, 10, 11 sono contrassegnate dal simbolo ~. Questo sta ad indicare che quell'uscita è abilitata all'uso della PWM. Arduino

permette di scegliere il *duty cycle* con un valore compreso fra 0 e 255 dove il 255 corrisponde al 100% e 127 corrisponde al 50%, potendo perciò avere una precisione di circa lo 0.5%, sul valore della tensione. Per dimensionare opportunamente i valori di *R* e *C*, cioè la frequenza di taglio del filtro passa-basso ($f_{\text{taglio}} = 1/\tau = 1/(R \cdot C)$ da essi costituito, in modo tale che si comporti da (buon) integratore, si tenga presente che il treno di impulsi generato da Arduino in modalità PWM ha frequenza $f \sim 1$ kHz.

Nel nostro caso R = 200Ω (in realtà sono due resistori da 100Ω da collegare in serie) e C = 470μF, Il resistore di polarizzazione del LED è R_D =100Ω, in modo che la corrente che passa nel circuito sia al più dell'ordine di qualche decina di milliampere.
Avete a disposizione quattro LED: uno rosso, uno verde, uno giallo e uno blu. Per primo inserite nel circuito il LED rosso.

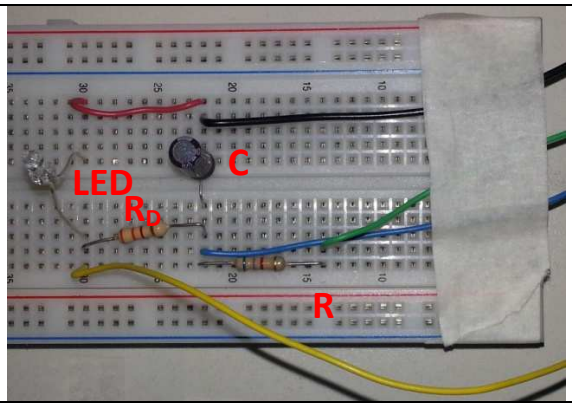

Figura 3 – Il circuito sulla breadboard. Attenzione i valori dei resistori riportati in figura e i colori dei fili sono diversi. Al posto di R vanno messi due resistori in serie.

La corrente *I* che circola nel led è valutata in modo indiretto a partire dalle tensioni V_1 e V_2 ai capi della resistenza di polarizzazione R_D: tali tensioni sono lette dalle porte A0 e A2 di Arduino; è necessario riferire i potenziali alla linea di massa (piedino GND) di Arduino. Fate attenzione a realizzare correttamente i collegamenti con la scheda (usate quattro fili di colori distinti, anche se quelli a disposizione non sono dello stesso colore di quelli nelle figure) e a controllare preliminarmente che le connessioni con i vari piedini della scheda siano corretti.
Per effettuare la misura dovete caricare lo sketch seguente nella memoria di Arduino utilizzando il programma Arduino nel computer di laboratorio.

Sketch di ARDUINO

```
const unsigned int RampPin = 5; //pin 5 uscita PWM per generare la rampa.
// Questo pin viene collegato all'integratore
const unsigned int analogPin_uno=0; //pin A0 per lettura V1
const unsigned int analogPin_due=2; //pin A2 per lettura V2
unsigned int i=0; //variabile che conta gli step durante la salita della rampa
int V1[256]; //array per memorizzare V1 (d.d.p, letta da analogPin_uno)
int V2[256]; //array per memorizzare V2 (d.d.p, letta da analogPin_due)
int start=1; //flag per fermare il ciclo
int rd = 100; // Il valore della resistenza RD = 100 Ohm
//Inizializzazione
void setup()
{
pinMode(RampPin, OUTPUT); //pin pwm RampPin configurato come uscita
Serial.begin(9600); //inizializzazione della porta seriale
}
//Ciclo di istruzioni del programma
void loop()
{
delay(1000); // attende 1s
```

```
Serial.println("INIZIO LA MISURA");
for(i=0;i<256;i++) //il valore che definisce il duty cycle dell'onda quadra è scrivibile su 8 bit
//cioè assume valori da 0 a 256
{
analogWrite(RampPin, i); //incrementa il duty cycle di uno step
V1[i]=analogRead(analogPin_uno); //legge il pin analogPin_uno
V2[i]=analogRead(analogPin_due); //legge il pin analogPin_due
delay(100); // tempo di attesa tra due misure successive 0.1 s
}
Serial.println("HO FINITO LA MISURA");
delay(2000);
Serial.println("INIZIO LA STAMPA DELLE MISURE");
delay(1000);
for(i=0;i<256;i++) //nuovo ciclo che scorre gli array di dati e li scrive sulla seriale
{
Serial.print(5.0*V2[i]/1024, 4); // il 4 dopo la virgola sta a significare che stampa con 4 cifre decimali
Serial.print(" ");
Serial.println((5.0/1.024)*(V1[i]-V2[i])/rd, 4); // si divide per 1.024 invece che per 1024 per avere la corrente in milliampere invece che in ampere
}
Serial.println("HO FINITO LA STAMPA DELLE MISURE");
analogWrite(RampPin, 0); // spegne il led
while (start = 1) {}; // mette Arduino in standby
}
```

I parametri che siete invitati a modificare secondo necessità sono l'intervallo di tempo Δt fra due campionamenti successivi, da scegliere in funzione della frequenza di taglio dell'integratore (la scelta non è critica), e la parte di *sketch* relativa alla misura della corrente, in base al valore del resistore R_D scelto.

Per far partire la misura basta accedere al Monitor seriale.

Installazione dello sketch su Arduino e inizio della misura.

Per caricare gli sketch su Arduino si usa il software *Arduino* che si trova sul *desktop*. Una volta aperto appare la schermata di figura 4 bisogna copiare lo sketch precedente al suo interno e poi salvare il file su una cartella (il comando salva si trova dentro il sottomenu File).
Utilizzare il nome LED_GNN (dove NN è il numero del vostro gruppo, per esempio il gruppo 1 salverà con il nome LED_G01). Se fate modifiche allo sketch e volete salvarle, aggiungete un numero progressivo dopo LED.

Una volta salvato si può verificare la corretta scrittura del codice con il pulsante *compila*  .
Effettuare i collegamenti dei fili con *Arduino*.
Ricordarsi di collegare anche la porta USB.
Se la compilazione è andata a buon fine si può caricare il programma sulla scheda *Arduino* con il pulsante *carica*  .

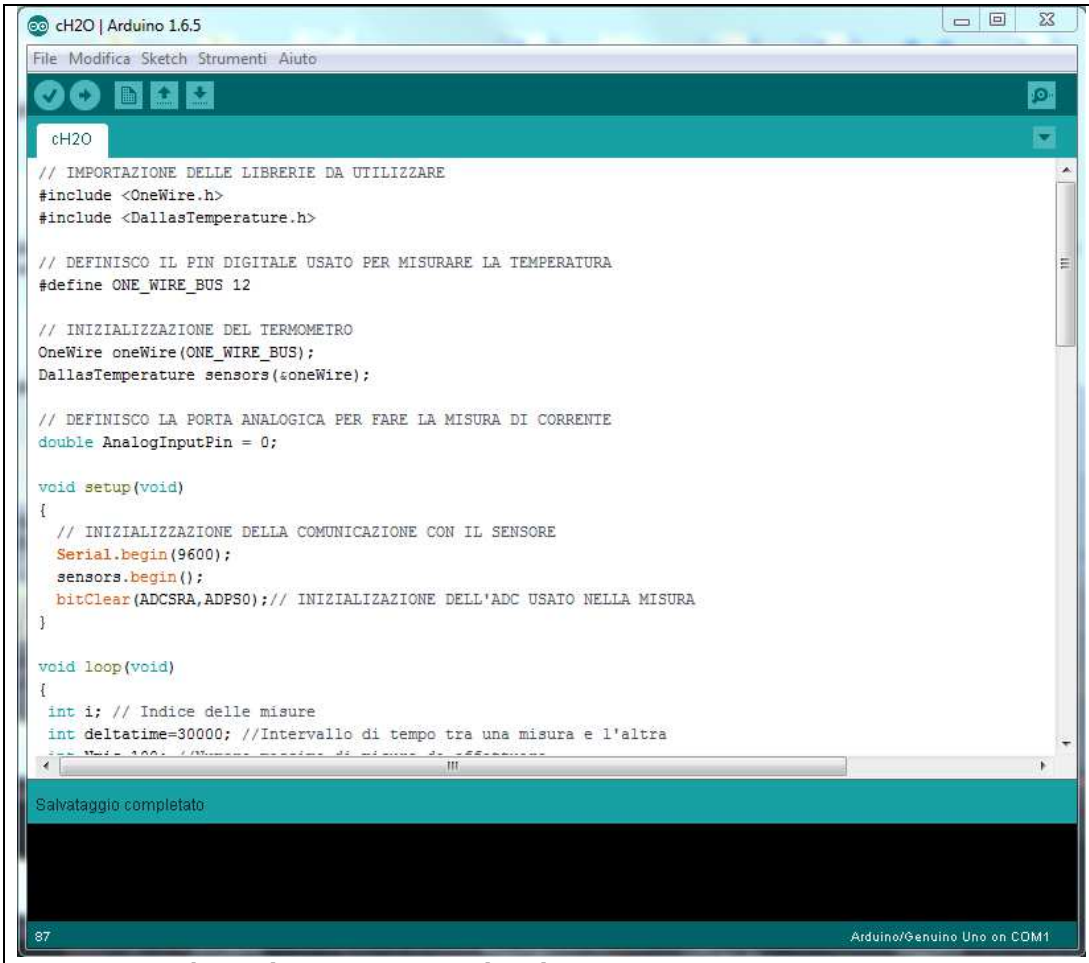

Figura 4 – L'ambiente di programmazione di *Arduino*

Per avviare l'esecuzione dello sketch cliccare su *Strumenti* e quindi su *Monitor seriale* (vedi figura 5)

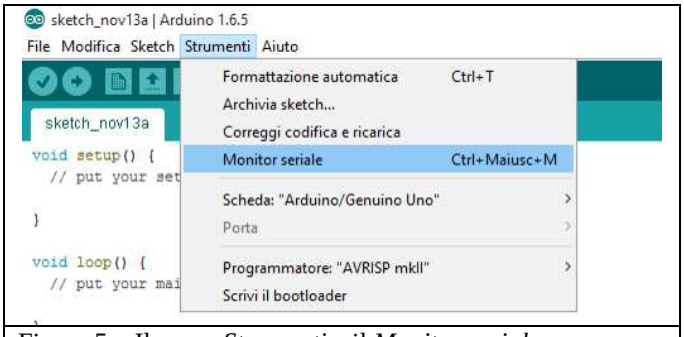

Figura 5 – Il menu *Strumenti* e il *Monitor seriale*

Sul monitor seriale appare la scritta INIZIO LA MISURA.
Attendere quindi le istruzioni che indicano la fine della stampa delle misure: HO FINITO LA MISURA, INIZIO LA STAMPA DELLE MISURE e HO FINITO LA STAMPA DELLE MISURE.

Il file che appare sul Monitor seriale ha 256 righe, corrispondenti a 256 misure realizzate al variare del duty-cycle del treno di impulsi, e due colonne che riportano nell'ordine le d.d.p. V_2 (in volt) e la corrente (in milliampere) separati da uno spazio.

I dati che appaiono sul Monitor seriale vanno salvati in un file di testo per la successiva rielaborazione. Per fare ciò:
1) Aprire *Blocco Note* di Windows.
2) Marcare i dati sul Monitor seriale iniziando dal primo e utilizzare il comando *copia* .
3) Posizionare il mouse nel Blocco Note e con il comando *incolla* trasferire i dati.
4) Salvare il file dandogli un nome (si consiglia GNN_LED_C_N.txt dove GNN è il numero del gruppo, C sta per il colore del LED: R = rosso, V = verde, G = giallo, B = blu, N è un numero progressivo, da mettere se si fanno più misure sul LED dello stesso colore)

Elaborazione dati

Per l'elaborazione dati:
1) Aprite un foglio Excel.
2) Importate i dati come indicato nel punto 2 (pag. 5) della scheda RAPPRESENTAZIONE GRAFICA E ANALISI DEI DATI SPERIMENTALI CON EXCEL.
3) Fate il grafico dei dati come indicato nel punto 1 (pag. 1) della stessa scheda.
4) Determinare la pendenza della parte in cui la corrente nel LED diventa lineare (vedi figura 6).
5) Fate la regressione lineare di questi valori come indicato nel punto 3 della scheda (pag. 8); si determinano il coefficiente angolare e il termine noto della retta di figura 6.
6) Determinare il valore di V_0 in cui la retta trovata interseca l'asse del potenziale; tale valore è detto potenziale di soglia o anche potenziale di innesco.
7) Salvare il foglio Excel con il nome GNN_LED_C_N con lo stesso significato dato sopra.

Ripetere le misure cambiando il LED. Secondo voi come varia il potenziale di soglia V_0 in funzione della lunghezza d'onda della luce emessa dal led?
SUGGERIMENTO: si tenga conto che per il LED vale la relazione $hf = eV_0$ (dove **h** è la costante di Planck e **f** è la frequenza della luce emessa)

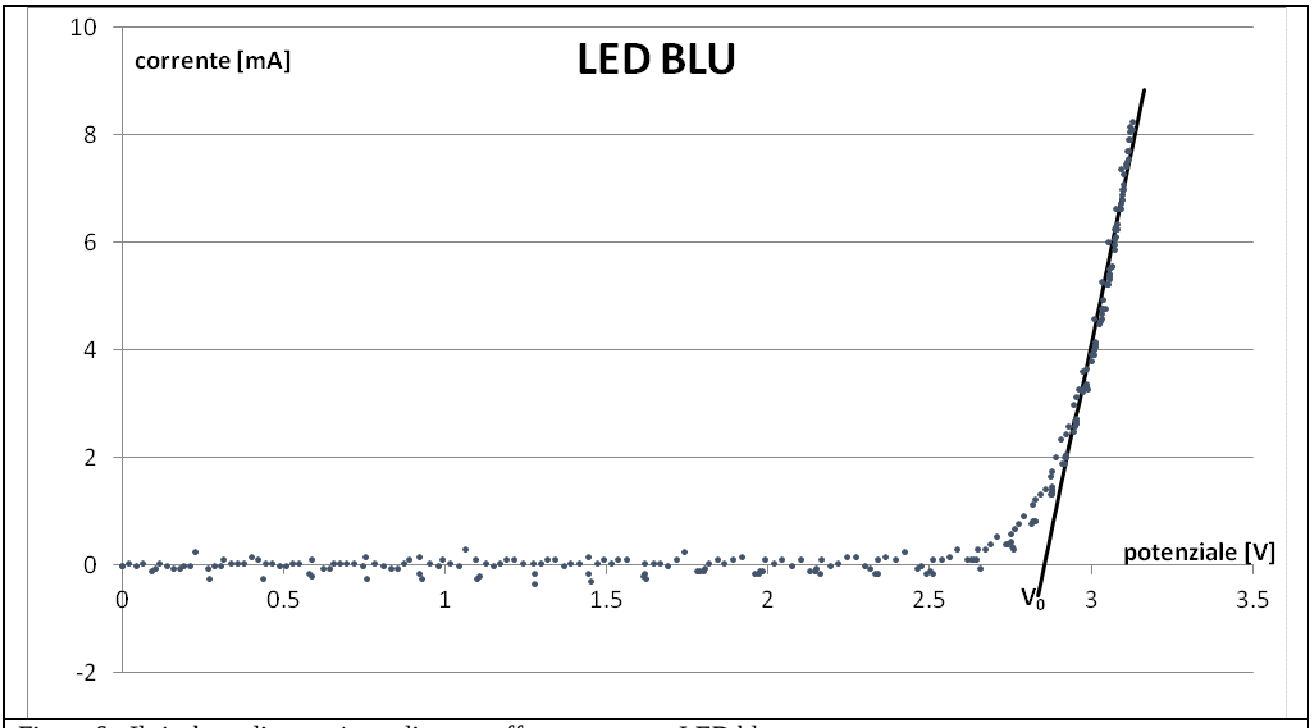

Figura 6 - Il risultato di una misura di prova effettuata con un LED blu.