



Laboratorio 6

Michele Ferrero 268542, Emanuele Falci 268617,
Gabriele Ferro 268510, Riccardo Drago 269663

A.A. 2021-2022

1 Strumentazione

Sono stati utilizzati i seguenti strumenti di laboratorio:

- Arduino Uno
- Breadboard
- Sensore di temperatura *LM335*
- Resistenze varie
- Multimetro digitale *Agilent – 34401A*
- Alimentatore *Rigol – DP832*

2 Funzionamento sensore

Il sensore digitale *LM335* è un diodo Zener. Esso permette la misura grazie alla tensione di breakdown, la quale è proporzionale alla temperatura. Sono qui raccolti i dati utili:

- Sensibilità nominale $S = 10mV/K$
- Incertezza strumentale a $T_{ambiente}$ $\delta T = \pm 2^{\circ}C$
- Corrente inversa nel campo da $0.4mA$ a $5mA$
- Resistenza termica $202^{\circ}C/W$

3 Arduino

3.1 Dati di riferimento

Si riportano qui alcuni dati utili che saranno utilizzati in seguito.

- $N_B = 10bit$ (8 bit per $f_S > 15kSa/s$)
- Tensione di riferimento Vcc o interna ($1.1V$)
- Non Linearità Integrale(INL): $0.5LSB$
- Incertezza assoluta: $2LSB$

3.2 Codice

Per utilizzare l'Arduino è stato scritto tramite l'utilizzo dell'omonimo software un programma detto “sketch”.

Dopo aver scritto il programma è possibile avviare il compilatore tramite il bottone con la spunta in alto a sinistra, cliccando l'icona a destra il codice viene inviato al micro-controllore e infine il tasto in alto a destra permette la comunicazione tra PC e micro-controllore, avviando anche l'esecuzione del programma.

Per quanto riguarda la struttura di uno sketch, è possibile notare tutte le variabili globali, definite e inizializzate prima delle funzioni. Tra queste sono presenti la funzione di *setup* e la funzione di *loop*, le quali devono essere sempre incluse anche se vuote.

4 Circuito 1

Inizialmente si utilizza come tensione di riferimento della scheda la V_{cc} , nel nostro caso pari all'ingresso in tensione di una porta USB2.0. Dunque si ha $V_{cc} = 5.0 \pm 0.25V$.

Si misura la V_{out} direttamente sul sensore, considerando che la corrente che passa sul diodo è pari a quella che passa sul sensore (quindi quella che entra nell'ADC è trascurabile).

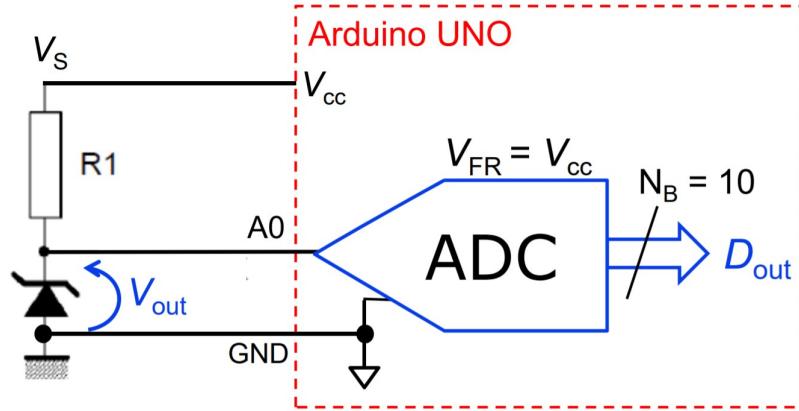


Figure 1: Prima funzione di taratura

4.1 Stima R1

In seguito si stima il valore nominale di R_1 calcolando il suo valore minimo e massimo. Considerando un campo di temperatura fra 5 e 50°C la V_{out} sul diodo Zener è compresa fra 2.78 e 3.23, mentre la corrente inversa dovrà essere $I_d \in [0.4, 5]mA$.

Dalla formula della corrente del diodo ricaviamo i valori minimo e massimo $R_{1,min}$ e $R_{1,max}$

$$I_D = \frac{V_s - V_{out}}{R_1}$$

$$R_{1,min} = \frac{V_s - V_{out,min}}{I_{D,max}} = \frac{5 - 2.78}{5 \cdot 10^{-3}} = 444\Omega$$

$$R_{1,max} = \frac{V_s - V_{out,max}}{I_{D,min}} = \frac{5 - 3.23}{4 \cdot 10^{-4}} = 4425\Omega$$

Scegliamo in questo range una resistenza nominalmente alta, pari a $3.9k\Omega$. In questo modo la corrente che transita nel circuito è minore e si riduce l'effetto dell'auto-riscaldamento.

La sua misura effettiva (effettuata con multmetro digitale) è $R1 = 3.840k\Omega$

4.2 Costruzione circuito

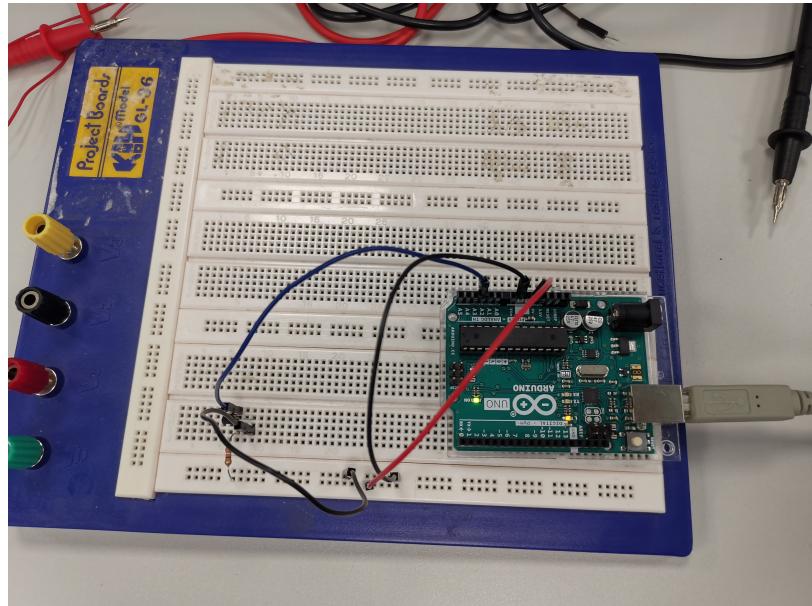


Figure 2: Implementazione del circuito 1

4.3 Sketch

Per utilizzare il primo circuito è stato scritto il seguente sketch. All'inizio vengono definite e inizializzate le variabili globali S (sensibilità sensore), V_{fr} (tensione di full range) e res (risoluzione) e nella funzione di *setup* viene inizializzata la comunicazione tramite *Serial.begin(9600)*, nella quale il valore inserito indica il *boderate*, ossia il numero di caratteri scambiati al secondo tra PC e Arduino. Nella funzione di *loop* viene letto il valore del sensore dal canale *A2* e vengono utilizzate le funzioni *vout(...)* e *temp(...)* per calcolare i rispettivi valori. Vengono poi stampati a schermo i risultati, e impostato un *delay* di *1s* per ripetere la lettura.

```
arduino | Arduino 1.8.15
File Edit Sketch Tools Help
arduino
float S = 0.01;
int Vfr = 5;
int res = 1024;

float vout(int Dout);
float temp(float Vout);

void setup() {
    // initialize serial communication
    Serial.begin(9600);
}

void loop() {
    // read the input on analog pin 0:
    int sensorValue = analogRead(A2);
    // print out the read value:
    float Vout = vout(sensorValue);
    float t = temp(Vout);
    char s[100];
    sprintf(s, "Temperature: %d.%02d, Vout: %d.%02d, Dout: %d", (int)t, (int)(t*100)%100, (int)Vout, (int)(Vout*100)%100, (int)Dout);
    Serial.println(s);
    delay(1000); // stop 1 s and repeat
}

float vout(int Dout){
    return ((float)(Dout * Vfr) / res);
}

float temp(float Vout){
    return (Vout / S) - 273.15;
}

Done uploading.
Sketch uses 4536 bytes (14%) of program storage space. Maximum is 32256 bytes.
Global variables use 234 bytes (1%) of dynamic memory, leaving 1814 bytes for local variables
32
Arduino Uno on COM3
```

Figure 3: Sketch 1

```
COM3
temperature: 17.37, Vout: 2.90, Dout: 595
Temperature: 16.88, Vout: 2.90, Dout: 594
Temperature: 17.37, Vout: 2.90, Dout: 595
Temperature: 16.88, Vout: 2.90, Dout: 594
Temperature: 17.37, Vout: 2.90, Dout: 595
Temperature: 16.88, Vout: 2.90, Dout: 594

Autoscroll  Show timestamp
Newline  9600 baud  Clear output
```

Figure 4: Output 1

4.4 Risultati e studio errore

La funzione di taratura utilizzata è:

$$T = D_{out} * \frac{V_{FR}}{2^{N_B}} * \frac{1}{S}$$

Per calcolare gli errori abbiamo utilizzato la formula generica per l'errore deterministico su T. I dati necessari sono:

- $\delta D_{out} = 2LSB$ ricavato dal manuale di Arduino
- $\delta V_{FR} = 0.25V$ preso dallo standard USB 2.0
- $\delta T_{sensor} = 2K$ preso dal manuale del sensore utilizzato
- $V_{FR} = 5V$ preso dallo standard USB 2.0
- $N_b = 10bit$ preso dal manuale di Arduino
- $S = 10 * 10^{-3}mV/K$ preso dal manuale del sensore
- $1LSB = \frac{V_{USB}}{1024} = 4.88 * 10^{-3}V$
- $D_{out} = 662$

$$\delta T_{V_{FR}} = \frac{V_{FR}}{S * 2^N} * \delta D_{out} + \frac{D_{out}}{S * 2^N} * \delta V_{FR} + \delta T_{sensor} = 18.16K$$

5 Circuito 3

In questo circuito viene utilizzato come riferimento interno la tensione dell'ADC di valore $(1.1 \pm 0.1)V$, il quale ha una tolleranza più alta rispetto a V_{cc} . Per rientrare nel fondo scala bisogna applicare un'attenuazione > 3 avendo come massimo di input la tensione del sensore pari a $3.3V$.

Infine per minimizzare l'effetto di carico le resistenze sono state scelte abbastanza grandi così da ridurre la corrente in esse.

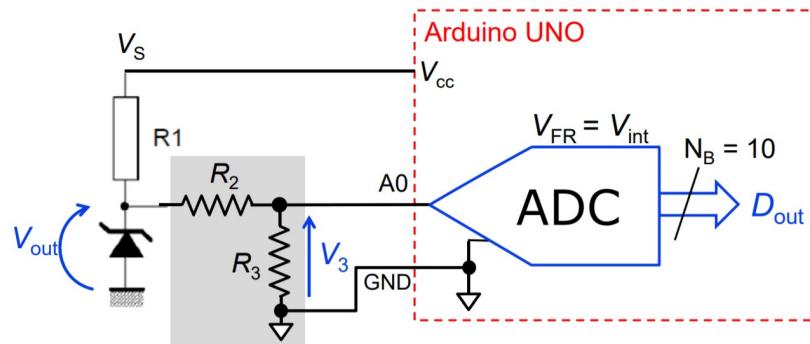


Figure 5: Circuito 3

5.1 Scelta di R₂ e R₃

Per la scelta delle resistenze R_2 e R_3 si considera che, utilizzando come riferimento di tensione dell'ADC la tensione interna, si deve avere un'attenuazione ≥ 3 e si cerca di minimizzare gli effetti di carico sulle resistenze.

$$3 \leq \frac{R_2+R_3}{R_3}$$

$$2 * R_3 \leq R_2$$

Vengono scelte $R_2 = 22k\Omega$ e $R_3 = 10k\Omega$ per rispettare la relazione ricavata e inoltre per ridurre la corrente che scorre all'interno in modo da ridurre gli effetti di carico, in particolare per evitare di portare fuori dinamica il sensore.

5.2 Caratterizzazione scheda Arduino

Per migliorare l'incertezza si procede a caratterizzare il sistema. In particolare si correggeranno gli errori di offset e di guadagno.

Preliminarmente calcoliamo D_{out} che sarà utile sia per la misura finale della temperatura, sia per calcolare il fattore di guadagno G.

$$D_{out} = V_{out} * \frac{R_3}{R_2+R_3} * \frac{2^{N_B}}{V_{int}} + D_{off}$$

$$G = \frac{R_3}{R_2+R_3} * \frac{2^{N_B}}{V_{int}}$$

5.2.1 Errore offset

Si misura inizialmente l'offset del sistema verificando il D_{out} a vuoto = D_{OFF} .

L'offset rilevato è 0. (Potrebbe essere presente un offset negativo, ma l'ADC utilizzato non è in grado di rilevarlo dunque si legge un'uscita pari a 0).

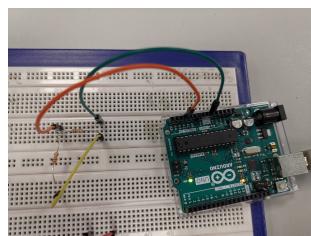


Figure 6: Circuito per misurare D_{OFF}

5.2.2 Errore di gain

In seguito si calcola l'errore di gain leggendo il valore di $D_{out,REF}$ quando è collegato un generatore esterno di tensione di valore noto. Si è utilizzata la tensione di uscita dell'alimentatore, di circa 3V. Si ha che:

$$D_{out,REF} = V_{REF} * G + D_{OFF}$$

$$G = \frac{D_{out,REF} - D_{OFF}}{V_{REF}} = \frac{879 - 0}{2.998} = 293,20$$

5.2.3 Errore integrale

L'errore integrale non è correggibile, rimarrà dunque un contributo di incertezza insieme all'errore di quantizzazione.

5.3 Costruzione circuito

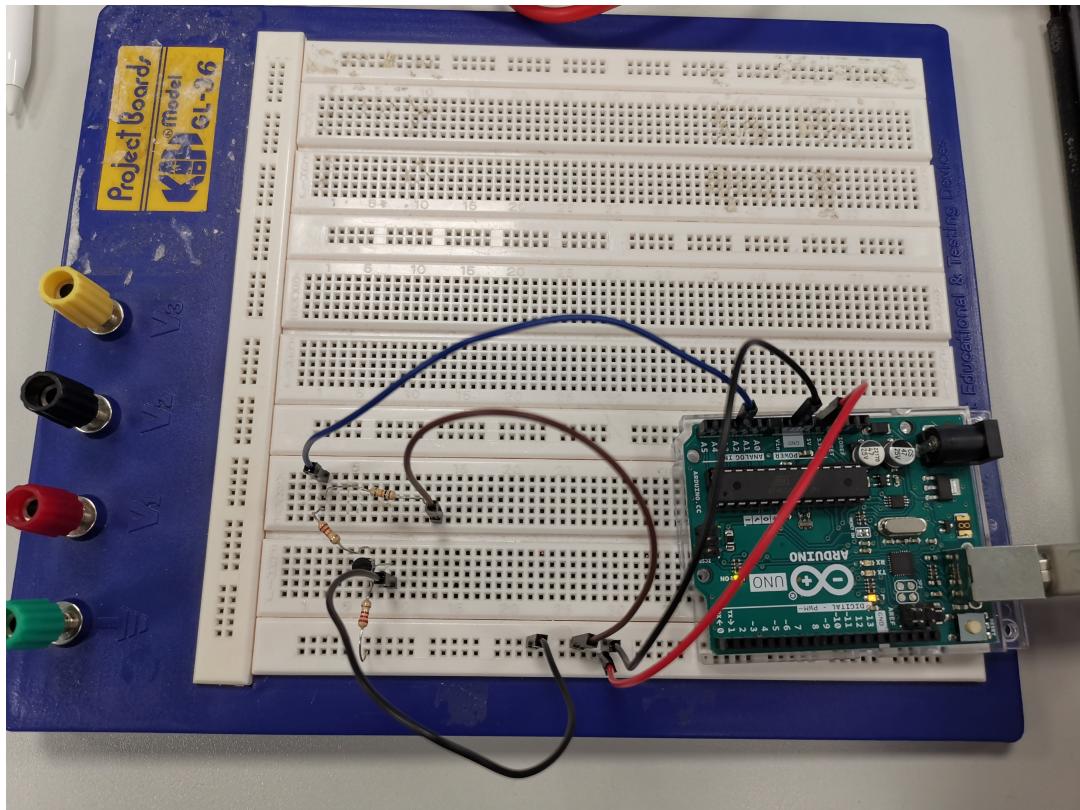


Figure 7: Implementazione del circuito 3

5.4 Sketch

Sono state aggiunte in cima tutte le variabili globali necessarie e si può notare che nella funzione di *setup* è stato modificato il riferimento della DC impostandolo a *INTERNAL* e viene calcolato il valore di *G* (correzione dell'errore di guadagno).

La funzione di loop non è molto differente rispetto al caso precedente, ma la funzione *temp* viene modificata adattando la formula al caso in esame.

The screenshot shows the Arduino IDE interface with the following code:

```
arduino | Arduino 1.8.15
File Edit Sketch Tools Help
arduino
float S = 0.01;
int Vfr = 5;
int res = 1024;
int D0=0;
float G;
int DoutG=879;
float Vref=2.998;

float temp(int Dout);
float Gcalc(int Dout);

void setup() {
    // initialize serial communication
    Serial.begin(9600);
    analogReference(INTERNAL);
    G=Gcalc(DoutG);
}

void loop() {
    int sensorValue = analogRead(A2);
    Serial.println(sensorValue);
    float t = temp(sensorValue);
    char s[100];
    Serial.println(G);
    sprintf(s, "Temperature: %d.%02d, Dout: %d", (int)t, (int)(t*100)%100, sensorValue);
    Serial.println(s);
    delay(1000); // stop 1 s and repeat
}

float temp(int Dout){
    return 1/S*(float)(Dout-D0)/G-(273.15);
}

Sketch uses 5202 bytes (16%) of program storage space. Maximum is 32256 bytes.
Global variables use 236 bytes (11%) of dynamic memory, leaving 1812 bytes for local variables
30
```

Arduino Uno on COM3

Figure 8: Circuito 3 - continua

The screenshot shows the Arduino IDE interface with the following code:

```
float Gcalc(int Dout){
    return (float)(Dout-D0)/Vref;
}

Sketch uses 5202 bytes (16%) of program storage space. Maximum is 32256 bytes.
Global variables use 236 bytes (11%) of dynamic memory, leaving 1812 bytes for local variables
30
```

Arduino Uno on COM3

Figure 9: Circuito 3

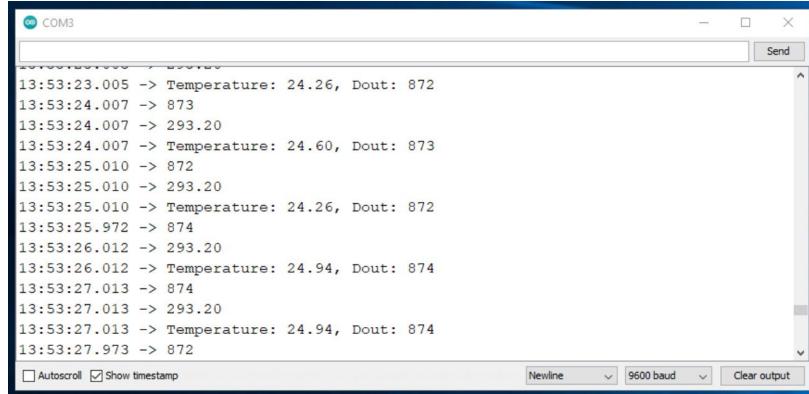


Figure 10: Output 3

5.5 Risultati e studio errore

In questo caso la funzione di taratura è invece:

$$T = \frac{1}{S} * \frac{D_{out} - D_{REF}}{G}$$

Per arrivare a calcolare l'incertezza sulla misura bisogna prima andare a trovare quella di G . Il coefficiente, che minimizza l'errore di guadagno ha come errore:

$$\delta G = \frac{1}{V_{REF}} * \delta D_{out,REF} + \frac{1}{V_{REF}} * \delta D_{out,0} = 5 * 10^{-4} V$$

con i seguenti dati:

- $\delta D_{out,REF} = 1LSB$ (linearità integrale + quantizzazione)
- $\delta D_{out,0} = 0.5LSB$ (solo quantizzazione)
- $V_{REF} = 2.998V$
- $1LSB = \frac{V_{REF}}{2^{Nb}} = \frac{1.08345}{1024} = 1 * 10^{-3} V$

L'incertezza sul circuito, esente da errore di offset e dall'errore di guadagno, è pari a:

$$\delta T = \frac{1}{S} * \frac{D_{out,MAX} - D_{off}}{G^2} * \delta G + \frac{1}{G*S} * \delta D_{out,MAX} + \frac{1}{G*S} * \delta D_{out,0} + \delta T = 2.0011 K$$

con, in aggiunta ai precedenti dati, anche:

- $D_{out,MAX} = 954$
- $G = 293.20$
- $\delta D_{out,MAX} = 1LSB$