UNIVERSITÀ DEGLI STUDI DI CATANIA

Dipartimento di Fisica e Astronomia "Ettore Majorana"

Corso di Laurea in Fisica

Relazioni di Laboratorio di Fisica 3

Le quattro cose

Anno Accademico 2024 – 2025

Indice

Sommario

In questo documento sono raccolte le quattro relazioni brevi da svolgere durante il corso annuale di *Laboratorio di Fisica 3* del Corso di Laurea in *Fisica* presso l'Università degli Studi di Catania.

Le esperienze sono esposte nei quattro capitoli seguenti:

1. *Implementazione numerica della formula di Bethe–Bloch.* Attraverso un codice in C che implementa numericamente la formula di Bethe–Bloch ho simulato il passaggio di una particella α a 5 MeV attraverso un sottile foglio di alluminio, realizzando un grafico che rappresenta l'energia della particella e la quantità di energia ceduta in funzione della distanza percorsa dentro il materiale.

2. *Misura di temperature con Arduino.* Attraverso l'uso di un microcontrollore Arduino, un sensore di temperatura e un semplice codice ho misurato la variazione di temperatura di una stanza in seguito all'accensione del riscaldamento. Nella relazione analizzo qualitativamente i dati raccolti ed estrapolo una possibile funzione che ne modelli l'andamento.

3. *Misura di resistenze con un multimetro digitale.*

4. *Accettanza geometrica di un rivelatore.*

1 Implementazione Numerica della Formula di Bethe–Bloch

1.1 Il modello

Come descritto dal Particle Data Group [2], la formula di Bethe–Bloch è un modello sperimentale che descrive la perdita di energia di particelle cariche pesanti di media energia—come protoni e particelle α—nella materia:

$$\left\langle -\frac{\mathrm{d}E}{\mathrm{d}x} \right\rangle = K z^2 \frac{Z}{A} \frac{1}{\beta^2} \left[\frac{1}{2} \log \frac{2 m_e c^2 \beta^2 \gamma^2 W_{\max}}{I^2} - \beta^2 - \frac{\delta(\beta\gamma)}{2} \right], \tag{1.1}$$

dove β e γ sono le usuali quantità relativistiche mentre il resto dei simboli sono esplicitati in Tab. 1.1. La perdita di energia media data dalla (1.1) è misuarta in $\mathrm{MeV\,g^{-1}\,cm^2}$ ma può essere portata in $\mathrm{MeV\,cm^{-1}}$ moltiplicando entrambi i membri per la densità volumica ρ del bersaglio misurata in $\mathrm{g\,cm^{-3}}$. La quantità $W_{\max}$ è la massima energia che una particella carica può cedere a un elettrone e si

Simbolo	Definizione	Valore o unità di misura
$m_e c^2$	massa a riposo dell'elettrone $\times c^2$	$0.510\,998\,950\,00(15)\,\mathrm{MeV}$
r_e	raggio classico dell'elettrone $e^2/4\pi\epsilon_0 m_e c^2$	$2.817\,940\,322\,7(19)\,\mathrm{fm}$
N_A	numero di Avogadro	$6.022\,140\,857(74) \times 10^{23}\,\mathrm{mol^{-1}}$
ρ	densità	$\mathrm{g\,cm^{-3}}$
x	massa per unità di area	$\mathrm{g\,cm^{-2}}$
M	massa della particella incidente	$\mathrm{MeV}\,c^{-2}$
E	energia della particella incidente $\gamma M c^2$	MeV
$W_{\max}$	massima energia trasferibile per collisioni	MeV
z	numero di carica della particella incidente	
Z	numreo atomico del bersaglio	
A	numero di massa atomica del bersaglio	
K	$4\pi N_\mathrm{A} r_e^2 m_e c^2$	$0.307\,075\,\mathrm{MeV\,mol^{-1}\,cm^2}$
I	energia media di eccitazione	eV
$\delta(\beta\gamma)$	correzione di ionizzazione	

Tabella 1.1: Notazione e unità di misura per la formula di Bethe–Bloch. Si tratta di un riassunto della tabella del PDG [2].

esprime come

$$W_{\max} = \frac{2m_e c^2 \beta^2 \gamma^2}{1 + \gamma\, m_e/M + (m_e/M)^2}\,. \tag{1.2}$$

La (1.1) e la (1.2) sono valide nell'approssimazione $0.1 \lesssim \beta\gamma \lesssim 1000$ poiché al limite inferiore la velocità del proiettile diventa confrontabile con la "velocità" degli elettroni atomici mentre al limite superiore gli effetti radiativi non sono più trascurabili.

1.2 La simulazione

Ho scelto di simulare la perdita di energia di una particella α a 5 MeV attraverso un foglio di alluminio di spessore[1] 0.020 mm. Per realizzare la simulazione ho scritto un codice in C che implementa la (1.1) in modo approssimato.

1.2.1 Descrizione del codice e dei calcoli

Il programma inizia richiedendo da tastiera il numero di intervalli N in cui suddividere lo spessore del materiale; gli altri dati sono tutti precompilati come costanti.

Assumiamo che in ciascun intervallo di spessore[2] $\mathrm{d}s$ tutte le quantità variabili di nostro interesse siano costanti—velocità, energia, perdita di energia *et cetera*. Dalla teoria della relatività ristretta scriviamo per l'n-esimo intervallo

$$\begin{cases} E_n = T_n + Mc^2 \\ E_n = \gamma_n Mc^2 \end{cases},$$

dove M è la massa a riposo della particella incidente, T_n la sua energia cinetica ed E_n la sua energia totale. Note queste ultime due quantità, è possibile calcolare il fattore di Lorentz γ_n e β_n^2 come

$$\gamma_n = \frac{T_n + Mc^2}{Mc^2}\,, \quad \beta_n^2 = 1 - \frac{1}{\gamma_n^2}\,.$$

Le quantità appena descritte sono calcolate attraverso il codice nelle righe 45–48.

Noti γ_n e β_n, è possibile calcolare la perdita di energia media per unità di lunghezza attraverso la (1.1) moltiplicata per ρ. Possiamo in particolare calcolare l'energia cinetica con cui la particella α entra nello strato successivo, T_{n+1} come

$$T_{n+1} = T_n + \mathrm{d}T_n = T_n + \rho \left\langle -\frac{\mathrm{d}E}{\mathrm{d}x} \right\rangle_n \mathrm{d}s\,,$$

[1] Dopo aver provato con uno spessore di 0.016 mm e aver constatato che la curva risultava tagliata—la particella non cedeva tutta l'energia all'alluminio—ho scelto di aumentare di poco lo spessore per il gusto di un grafico più completo.

[2] Ricordo che in questo caso la x non indica una distanza.

essendo naturalmente $\mathrm{d}T_n \leq 0$. Il calcolo di $\rho\,|\langle -\,\mathrm{d}E\,/\,\mathrm{d}x\rangle_n|$ è svolto dalla funzione `bethe()` chiamata alla riga 49 e definita alla riga 75, con l'ausilio della funzione `wmax()` che in particolare calcola il termine $W_{\max}$.

Il calcolo viene quindi ripetuto a partire dalla nuova energia T_{n+1} per ottenere la perdita di energia attraverso il successivo strato. Per ogni reiterazione il codice stampa su un file la distanza totale percorsa, l'energia cinetica della particella e la quantità $|\mathrm{d}T_n| = \rho\,|\langle -\,\mathrm{d}E\,/\,\mathrm{d}x\rangle_n|$.

1.2.2 Risultati della simulazione

2 Misura di Temperature con Arduino

Tra le esperienze svolte con Arduino Uno riporto in particolare la misura della variazione della temperatura della mia stanza da letto in seguito all'accensione del riscaldamento in casa.

2.1 L'esperimento

L'obiettivo dell'esperienza è quello di valutare qualitativamente l'andamento della temperatura della stanza per fare una stima di quanto velocemente si riscaldi e a quale temperatura tenda asintoticamente.

2.1.1 Preparazione della stanza

Per massimizzare l'escursione termica ho effettuato la misura durante una sera invernale avendo preventivamente aperto le finestre per abbassare la temperatura della stanza.

Per migliorare la circolazione dell'aria ed evitare un eccessivo gradiente di temperatura—il radiatore caldo si trova in un angolo della stanza mentre i vetri freddi della finestra si trovano dal lato opposto—ho acceso dei ventilatori: uno a soffitto per limitare la raccolta dell'aria calda in alto e un più piccolo ventilatore da tavolo per allontanare l'aria calda dal radiatore e facilitare il riscaldamento dell'aria fredda.

Infine, per isolare il più possbile il sistema, ho chiuso le tende sulla finestra per ridurre la dispersione di calore attraverso il vetro freddo e mantenuto la porta chiusa per non disperdere calore nel resto della casa.

2.1.2 Strumenti utilizzati

Gli strumenti utillizzati per la presa dei dati sono:

- Una microcontrollore Arduino Uno con un sensore di temperatura TMP36;
- Un computer per compilare ed eseguire il codice sulla scheda Arduino e prelevare i dati.

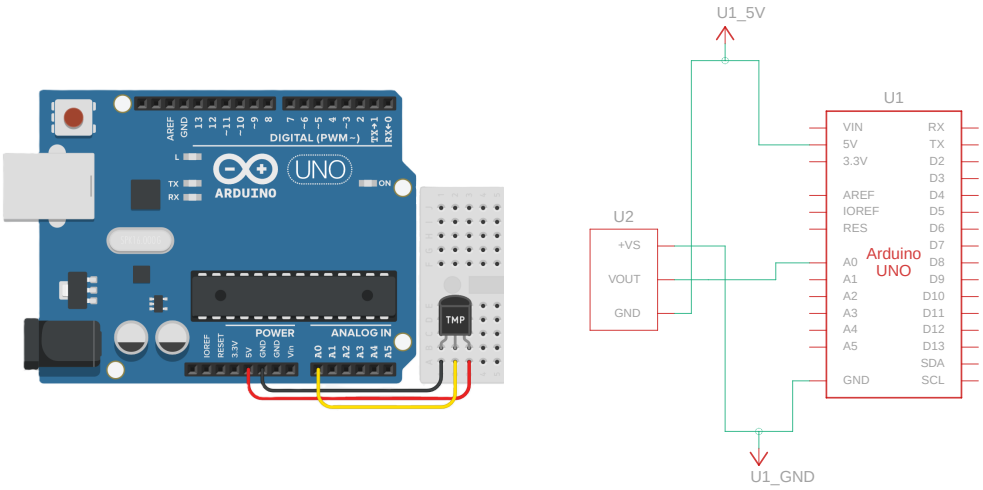

Figura 2.1: A sinistra una rappresentazione digitale del circuito realizzato per l'esperimento. A destra lo schema del circuito. Entrambe le illustrazioni sono state realizzate con Tinkercad®.

Il sensore TMP36 è un sensore di temperatura a semiconduttore pensato per operare in un range di temperature che va da $-40\,°\mathrm{C}$ a $+125\,°\mathrm{C}$. Esso presenta tre pin: `+vs`, `vout` e `gnd`. Il primo e l'ultimo servono per l'alimentazione che deve essere compresa tra $2.7\,\mathrm{V}$ e $5.5\,\mathrm{V}$ con una corrente inferiore ai $50\,\mu\mathrm{A}$, che garantisce un surriscaldamento per effetto Joule trascurabile. Il secondo pin invece sestituisce una differenza di potenziale rispetto al `gnd` proporzionale alla temperatura misurata. La sensibilità del sensore fornita dal costruttore è di $\pm 1\,°\mathrm{C}$ e il suo fattore di scala è di $10\,\mathrm{mV}\,°\mathrm{C}^{-1}$ [1].

La scheda Arduino attraverso i pin analogici accetta in input delle differenze di potenziale che vanno da $0\,\mathrm{V}$ a $5\,\mathrm{V}$ che vengono convertite in un segnale digitale che assume valori discreti da 0 a 1023.

2.1.3 Circuito e codice

Il circuito realizzato per l'esperimento è quello rappresentato in Fig. 2.1. L'alimentazione al sensore è fornita tramite i pin `5v` e `gnd` mentre il segnale in uscita dal sensore viene letto dal pin `A0` della scheda.

Per effettuare le misure ho usato il codice riportato in appendice B. A intervalli di $30\,\mathrm{s}$ la lettura discreta di tensione data dal sensore[1] e la converte in un numero decimale tra $0\,\mathrm{V}$ e $5\,\mathrm{V}$ attraverso la formula

$$\frac{\texttt{(float)analogRead(SENSOR\_PIN)}}{\texttt{MAX\_READ}} * \texttt{MAX\_V},$$

[1] Come detto prima si tratta di un valore tra 0 e 1023

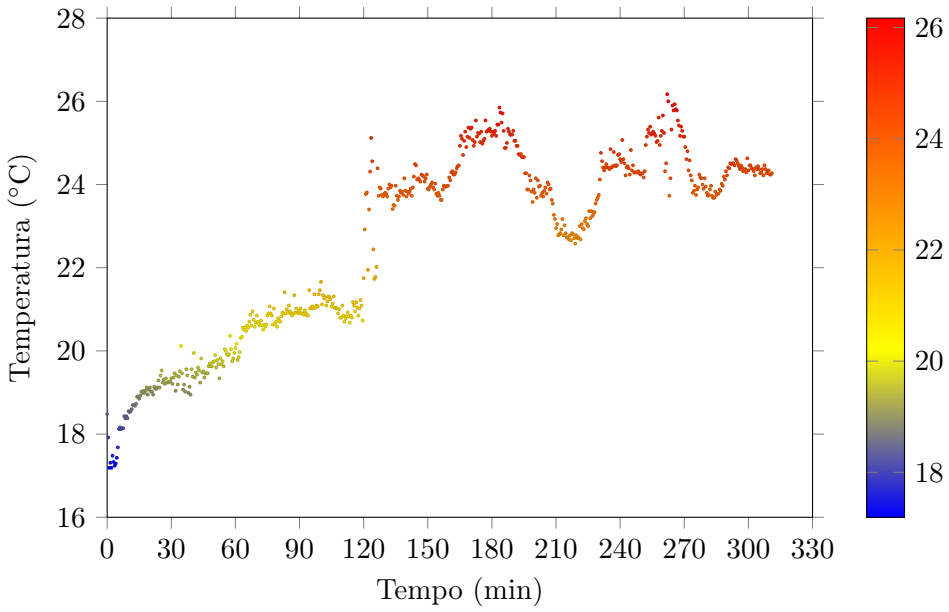

Figura 2.2: Andamento della temperatura nel tempo

essendo `MAX_READ` = 1023 e `MAX_V` = 5 V. Sapendo che a una tensione di 0 V corrisponde una temperatura di 0.5 °C e a 4.5 V corrispondono 100 °C, la conversione della lettura in gradi Celsius è data da

$$\left[\frac{\texttt{(float)analogRead(SENSOR\_PIN)}}{\texttt{MAX\_READ}} * \texttt{MAX\_V} - \texttt{A}\right] * \texttt{B}\,, \tag{2.1}$$

dove `A` = 0.5 V e `B` = 100 °C V^{-1} sono i fattori di scala.

La conversione dei valori discreti in temperatura è eseguita dal codice tra le righe `23` e `26` applicando la (2.1). Vengono eseguite `N` = 20 misure consecuitive di cui è contestualmente calcolata la media che viene a sua volta stampata a schermo. Infine il codice attende il tempo mancante per raggiungere i 30 s dall'inizio del `loop`.

2.2 Dati

Attraverso il codice riportato al punto §2.1.3 ho misurato la temperatura della stanza ogni 30 s per circa cinque ore e mezza; in Fig. 2.2 sono riportati tutti i dati raccolti, avendo convertito il tempo in minuti.

2.2.1 Prime considerazioni

Osservando il grafico si nota una evidente crescita di temperatura che, dopo una crescita regolare, oscilla fino a stabilizzarsi poco sopra i 24 °C.

I primi punti a temperatura più elevata possono essere dovuti a un precedente contatto con le mani e un riscaldamento del sensore che è poi tornato a temperatura

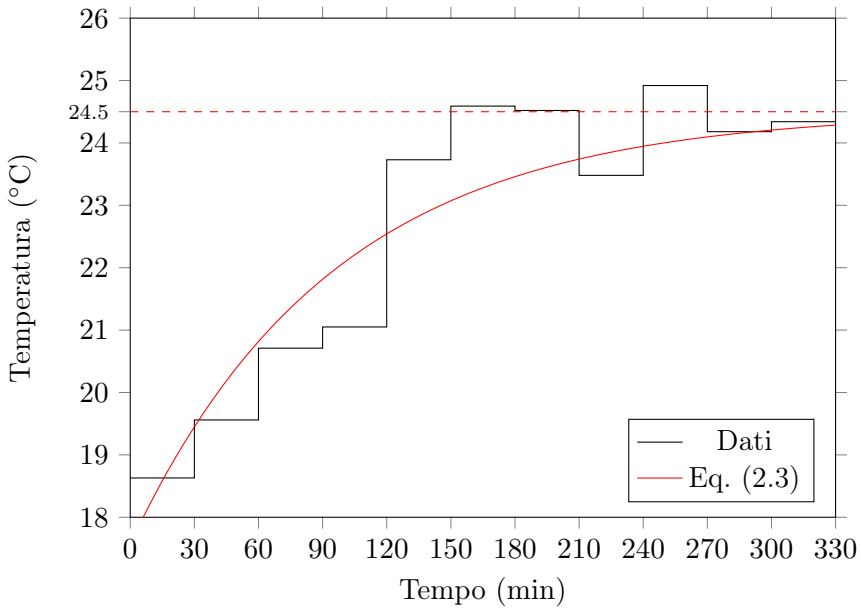

Figura 2.3: Temperature mediate in intervalli di 30 min.

ambiente. L'ampia oscillazione dei dati intorno ai 130 min deve essere dovuta a un movimento del sensore, che ho dovuto spostare di qualche centimetro. È quindi possibile che anche le successive oscillazioni siano dovute alla nuova posizione del sensore in un punto con un flusso d'aria più dinamico.

2.2.2 Breve analisi dei dati

Per per visualizzare il macro-andamento della temperatura tamponando il rumore, ho deciso di suddividere le misure in intervalli temporali di 30 min e riportare nel grafico in Fig. 2.3 la temperatura media per ciascun intervallo.

Anche in questo caso si notano l'andamento crescente della temperatura e il salto, ma le oscillazioni intorno alla temperatura finale di circa 24 °C risultano smorzate. Si osserva inoltre che la temperatura si stabilizza intorno a questo valore 150 min—oppure 2 h 30 min—dopo l'accensione del riscaldamento.

Svolgiamo adesso un fit dei dati di carattere prettamente qualitativo. Dal momento che il radiatore a regime avrà una temperatura T^* fissata e tenendo conto della dispersione del calore verso l'esterno, è ragionevole assumere che la temperatura della stanza $T(t)$ tenda asintoticamente a un valore finito $T_\mathrm{f} \leq T^*$ per $t \to +\infty$. Una funzione crescente che ha un comportamento simile è

$$T(t) = T_\mathrm{f}\left[1 - \exp\left(-\frac{t - t_0}{\tau}\right)\right], \tag{2.2}$$

per qualche valore di T_f, t_0 e τ. Supponiamo arbitrariamente che la temperatura

limite[2] sia $T_\mathrm{f} = 24.5\,^\circ\mathrm{C}$; per quanto detto prima supponiamo inoltre che il tempo caratteristico[3] sia $\tau = 95\,\mathrm{min} \approx 63\,\%$ di $150\,\mathrm{min}$. Imponendo che sia $T(0) = 18.63\,^\circ\mathrm{C}$ troviamo

$$t_0 = \tau \log\left[1 - \frac{T(0)}{T_\mathrm{f}}\right] \approx -135\,\mathrm{min}\,.$$

Infine per centrare meglio la funzione rispetto agli intervalli di 30 min sommiamo a t_0 un valore di 15 min, pari a metà dell'intervallo di tempo. La funzione finale che si ottiene sostituendo questi valori nella (2.2) e si trova rappresentata in Fig. 2.3 è

$$T(t) = 24.5\left[1 - \exp\left(-\frac{t + 120}{95}\right)\right]. \tag{2.3}$$

2.3 Conclusioni

I dati presi possono essere migliorati facendo maggiore attenzione a non toccare gli strumenti o uscendo dalla stanza e assicurandosi che la porta non venga mai aperta.

Per avere informazioni più dettagliate sulla bontà del fit si potrebbe invece procedere applicando il metodo dei minimi quadrati per determinare i tre parametri T_f, t_0 e τ. Per eseguire un fit con due parametri invece che tre, si può migliorare l'accuratezza su T_f continuando a prendere dati il più a lungo possibile. Questo permetterebbe di fissare la temperatura limite con più sicurezza e di determinare t_0 e τ attraverso la linearizzazione della (2.2) in

$$\log\left[1 - \frac{T(t)}{T_\mathrm{f}}\right] = \frac{t_0}{\tau} - \frac{1}{\tau}t \quad \Longleftrightarrow \quad y(t) = c_1(t_0, \tau) + c_2(t_0, \tau)t\,.$$

Si potrebbe inoltre provare a utilizzare funzioni diverse dalla (2.2) nel fit per provare a modellare quantitativamente le oscillazioni che si verificano in prossimità della temperatura limite, possibilmente dovute a moti convettivi non del tutto smorzati dai ventilatori indicati al punto §2.1.1.

[2] I dati in Fig. 2.2 oscillando superano i 25 °C ma le ultime misure sono tutte comprese tra 24 °C e 24.5 °C, motivo per cui assumo quest'ultimo valore come temperatura limite.

[3] Dal momento che "a occhio" la temperatura di 24.5 °C viene già raggiunta dopo 150 min, scegliamo il tempo caratteristico come il tempo necessario a raggiungere il 63 %, ovvero $(1/e)$-*esimo* della temperatura finale.

A Codice per la formula di Bethe–Bloch

Riporto in questa appendice il codice utilizzato per l'esperienza descritta al Capitolo §1. Come descritto nel capitolo dedicato, il programma simula il passaggio di particelle α attraverso un foglio di alluminio spesso 0.020 mm.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#define STRLEN  1024
#define ME      0.511   /* mass of electron in MeV / c^2 */
#define K       0.307   /* constant K in Bethe--Bloch's formula */
#define I       173.0   /* mean excitation energy in Al */

/* INC = projectile, TAR = target */
#define Z_INC   2.0     /* number of protons in projectile */
#define M_INC   2807.0  /* mass of projectile in MeV / c^2 (4He) */
#define T_INC   5.0     /* kynetic energy of projectile (MeV) */
#define Z_TAR   13.0    /* number of protons in target (Al) */
#define A_TAR   27.0    /* number of nucleons in target (Al) */
#define R_TAR   2.7     /* target density */
#define X_TAR   0.0020  /* target thickness (cm) */

#define FILE_PATH       "path/to/file"

double wmax(double m_e, double beta2, double gamma, double gamma2,
double m, double m2);
double bethe(double k, double m_e, double z_inc, double z_tar, double
a_tar, double r_tar, double beta2, double gamma2, double w_max, double
i);

int main() {
    double x, dx, E_tot, T, dT, stop;
    double gamma, gamma2, beta2, m ,m2;
    int step;
    char buffer[STRLEN];
    FILE *pf;

    printf("Step number: ");
```

```
33      scanf("%s", buffer);
34
35      if (0 >= (step = atoi(buffer))) goto err_step;
36      if (NULL == (pf = fopen(FILE_PATH, "w"))) goto err_file;
37
38      dx  = X_TAR / step;
39      m   = ME / M_INC;
40      m2  = m*m;
41
42      for (T = T_INC, x = 0; (x < X_TAR) && (T >= 0); x += dx) {
43          if (T <= 0) fprintf(pf, "%.8f %.8f %.8f\n", x, 0.0, 0.0);
44          else {
45              E_tot = T + M_INC;
46              gamma = E_tot / M_INC;
47              gamma2 = gamma * gamma;
48              beta2 = 1.0 - (1.0 / (gamma2));
49
50              stop = bethe(K, M_E, Z_INC, Z_TAR, A_TAR, R_TAR, beta2,
gamma2, wmax(ME, beta2, gamma, gamma2, m, m2), I);
51
52              dT = stop * dx;
53              fprintf(pf, "%.8f %.8f %.8f\n", x, T, dT);
54              T -= dT;
55          }
56      }
57
58      fclose(pf);
59      return EXIT_SUCCESS;
60
61      err_step:
62          printf("Step number musts be positive\n");
63          return EXIT_FAILURE;
64
65      err_file:
66          printf("File at '%s' not found or could not be open\n",
FILE_PATH);
67          return EXIT_FAILURE;
68  }
69
70  double wmax(double m_e, double beta2, double gamma, double gamma2,
double m, double m2) {
71      return (2.0 * ME * beta2 * gamma2) / (1.0 + (2.0 * gamma * m) + m2)
;
72  }
73
74  double bethe(double k, double m_e, double z_inc, double z_tar, double
a_tar, double r_tar, double beta2, double gamma2, double w_max, double
i) {
75      return r_tar * (((k * z_inc * z_inc * z_tar) / (a_tar * beta2)) *
(0.5 * log((2.0 * m_e * beta2 * gamma2 * w_max) / (i * i * 1.0e-12)) -
beta2));
76  }
```

B Codice per Arduino

Riporto in questa appendice il codice utilizzato per l'esperienza descritta al Capitolo §2.

```
#define SENSOR_PIN  A0
#define MAX_READ    1023.0
#define MAX_V       5.0
#define N           20
#define A           0.5
#define B           100

void setup() {
    // set SENSOR_PIN (A0) as analog input pin and initialize the
serial monitor
    pinMode(SENSOR_PIN, INPUT);
    Serial.begin(9600);
}

float         tmp;
int           i = 0;
unsigned long dt = 0;

void loop() {
    // print the time since the program started
    Serial.print(dt = millis());

    // measure the temperature 20 times and calculate the average
    for (i = 0, tmp = 0; i < N; i++) {
        tmp += (((float)analogRead(SENSOR_PIN) / MAX_READ) * MAX_V - A)
 * B;
    }
    tmp = tmp / N;

    // print the measured temperature next to the time associated with
that measurement
    Serial.print(", ");
    Serial.println(tmp);

    // wait for a total of 30 seconds, taking the time up to here into
account
    dt = millis() - dt;
    delay(30000 - dt);
}
```

Bibliografia

[1] Analog Devices. *TMP35/TMP36/TMP37 Data Sheet*. Accessed: 2025-04-27. Analog Devices. 2015. URL: https://www.analog.com/media/en/technical-documentation/data-sheets/tmp35_36_37.pdf.

[2] S. Navas e C. et al. Amsler. «Review of Particle Physics». In: *Phys. Rev. D* 110 (3 2024), p. 030001. DOI: 10.1103/PhysRevD.110.030001. URL: https://link.aps.org/doi/10.1103/PhysRevD.110.030001.