



Politecnico di Torino

Anno accademico 2022/2023

**Corso di Progettazione di Dispositivi Biomedici
Programmabili (Marco Knaflitz)**

Luxmetro con fotodiodo per luce continua

Gruppo 18:

316660 Lolli Alice
314760 Reviglio Luca
314946 Toska Robjona
314774 Traetta Giovanni

Indice

1.	Introduzione.....	3
2.	Specifiche di progetto	3
2.1	Specifiche richieste	3
2.2	Specifiche soddisfatte	3
2.3	Specifiche non soddisfatte	3
3.	Schemi elettrici e descrizione della basetta (descrizione hardware)	3
3.1	Schema a blocchi dell'hardware e suo funzionamento.....	3
3.2	Schema elettrico complessivo	4
3.3	Schemi elettrici singoli	5
3.3.1	Circuito di condizionamento sensore	5
3.3.2	Circuito di pilotaggio dei display a sette segmenti	6
3.3.3	Circuito di pilotaggio del LED di allarme	6
3.4	Foto basette montate	7
4.	Descrizione software.....	7
4.1	Diagramma a blocchi.....	7
4.2	Flowchart.....	8
4.2.1	Programma principale.....	8
4.2.2	Subroutine <i>measure_lux</i>	11
4.2.3	Subroutine <i>measure_al</i>	12
4.2.4	Subroutine di risposta ad interrupt: Overflow del Timer/Counter 0.....	13
5.	Calcoli progettuali	14
5.1	Tensione di uscita del circuito di condizionamento del fotodiodo.....	14
5.2	Taratura del fotodiodo BPW 34	15
5.3	Resistore di protezione di un diodo LED rosso	15
6.	Prove di verifica.....	16
6.1	LED di allarme	16
6.2	Fondo scala	17
6.3	Risoluzione	17
6.4	Accuratezza	17
6.5	Ripetibilità	18
6.6	Corrente assorbita	19
7.	Conclusioni.....	19
	Appendice	20
	Listato del software.....	20

1. Introduzione

Il progetto consiste nella realizzazione di un misuratore di intensità luminosa (luxmetro) attraverso il fotodiodo BPW34. Il segnale proveniente da quest'ultimo viene acquisito ed elaborato da una scheda "Arduino Uno R3", basata sul microcontrollore ATmega328P, programmato in linguaggio Assembly. Il valore di luminosità ottenuto viene visualizzato su tre display a sette segmenti.

2. Specifiche di progetto

2.1 Specifiche richieste

- Misura della tensione di alimentazione ogni 1000 ms e accensione di un diodo LED nel caso in cui questa tensione scenda sotto la soglia di 4.2 V.
- Misura continua dell'illuminazione e visualizzazione della misura in chilolux su tre display a sette segmenti. Aggiornamento dell'indicazione della misura ogni 1000 ms.
- Eventuale taratura del sensore di illuminazione.
- Variazione della tensione di alimentazione tramite potenziometro tra il cursore e il riferimento tra V_{al} e GND.

2.2 Specifiche soddisfatte

- Misura della tensione di alimentazione ogni 1000 ms.
Sistema di allarme (diodo LED rosso) a due soglie per tensione di alimentazione:
 $Val \geq 4.6 V \rightarrow$ led spento
 $Val < 4.2 V \rightarrow$ led acceso
 $4.2 V \leq Val < 4.6 V \rightarrow$ led lampeggiante (con periodo di 500 ms)
- Misura dell'illuminazione in chilolux ogni 1000 ms e relativa indicazione su tre display a sette segmenti.
- Variazione della tensione di alimentazione tramite potenziometro tra il cursore e il riferimento tra V_{al} e GND.
- Risoluzione del luxmetro pari a 10 lux

2.3 Specifiche non soddisfatte

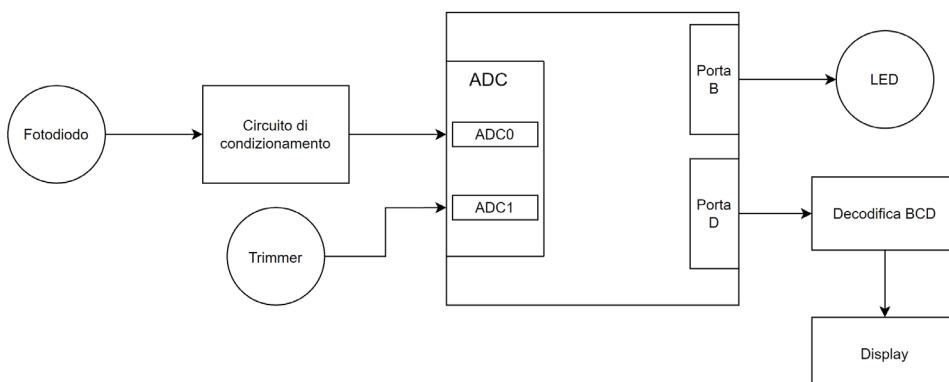
- Non potendo garantire una misura continua dell'illuminazione, abbiamo applicato un periodo di misura di 1000 ms, equivalente a quello dell'aggiornamento dell'indicazione di luminosità.

3. Schemi elettrici e descrizione della basetta (descrizione hardware)

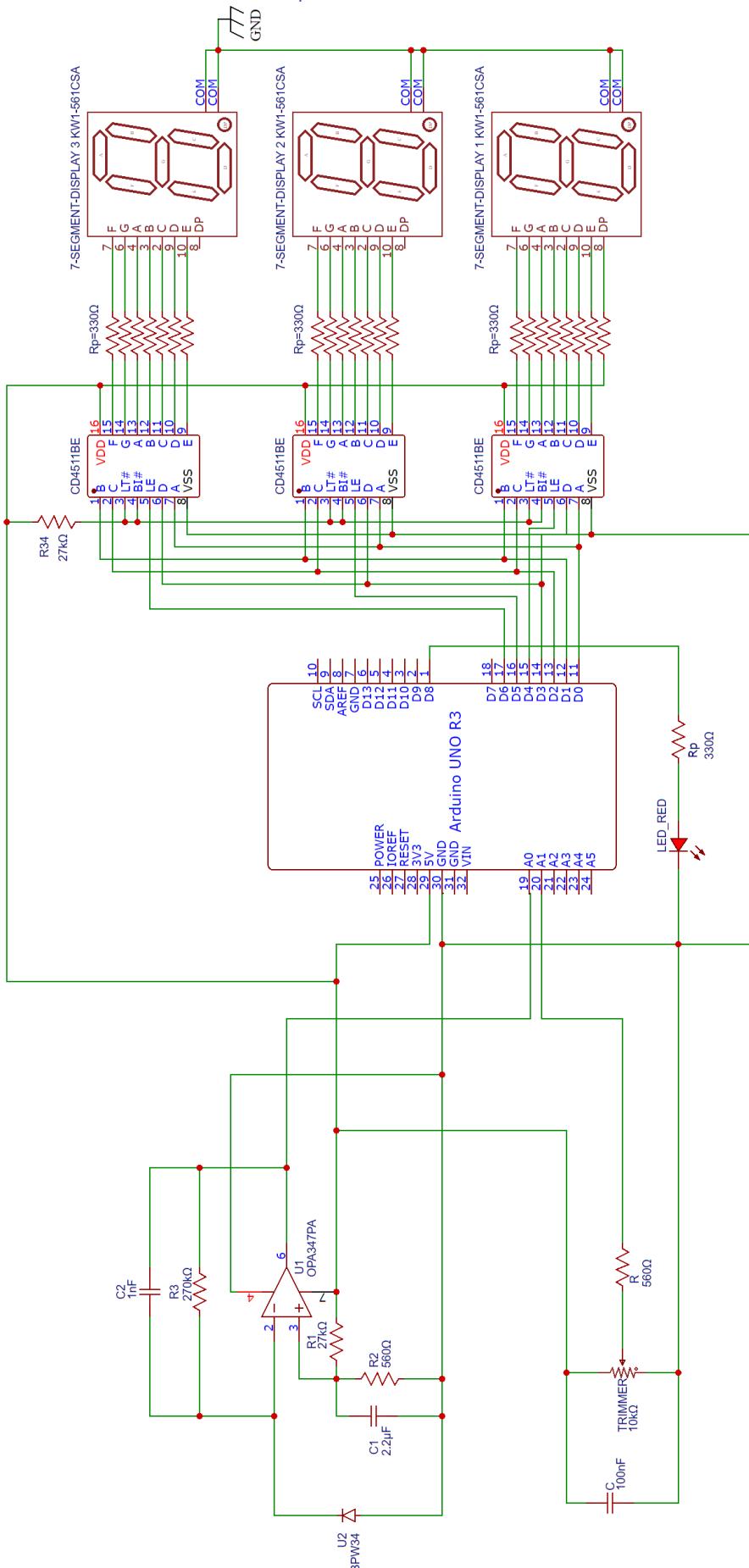
3.1 Schema a blocchi dell'hardware e suo funzionamento

Il circuito complessivo è formato da tre blocchi principali:

- Circuito di condizionamento del segnale prelevato dal sensore di intensità luminosa
- Circuito di pilotaggio dei display a sette segmenti
- Circuito di pilotaggio del LED di allarme della tensione di alimentazione

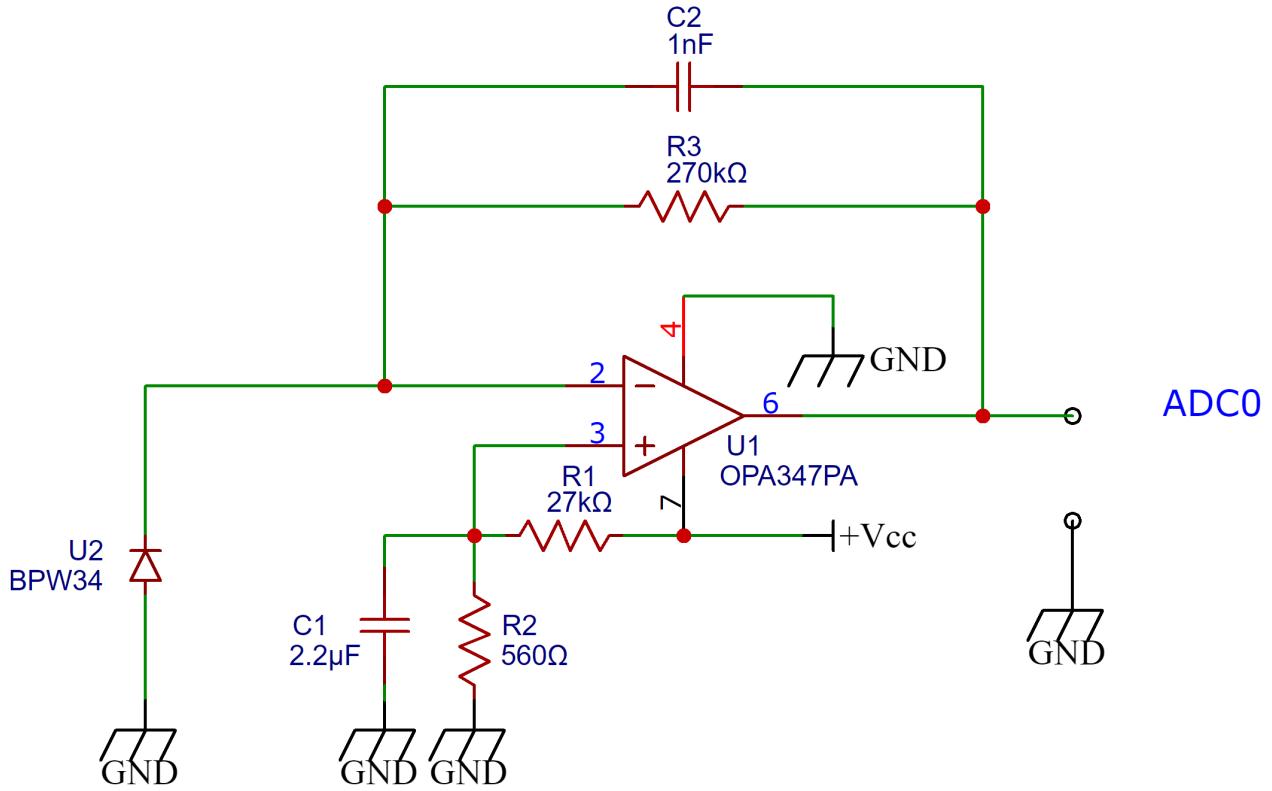


3.2 Schema elettrico complessivo



3.3 Schemi elettrici singoli

3.3.1 Circuito di condizionamento sensore



Nel fotodiodo BPW34 scorre una corrente proporzionale all'intensità luminosa. Attraverso un circuito di condizionamento, basato sull'amplificatore operazionale OPA347, si ottiene all'uscita una tensione anch'essa proporzionale all'intensità luminosa. Quest'ultima è collegata all'ingresso A0 del convertitore analogico/digitale della scheda "Arduino Uno R3".

La tensione di uscita calcolata utilizzando la sensibilità del fotodiodo è:

$$V_u = R_3(lux \cdot S) + V_{off}$$

$$\text{con } V_{off} = V_{R_2} = V_{al} \frac{R_2}{R_2 + R_1}.$$

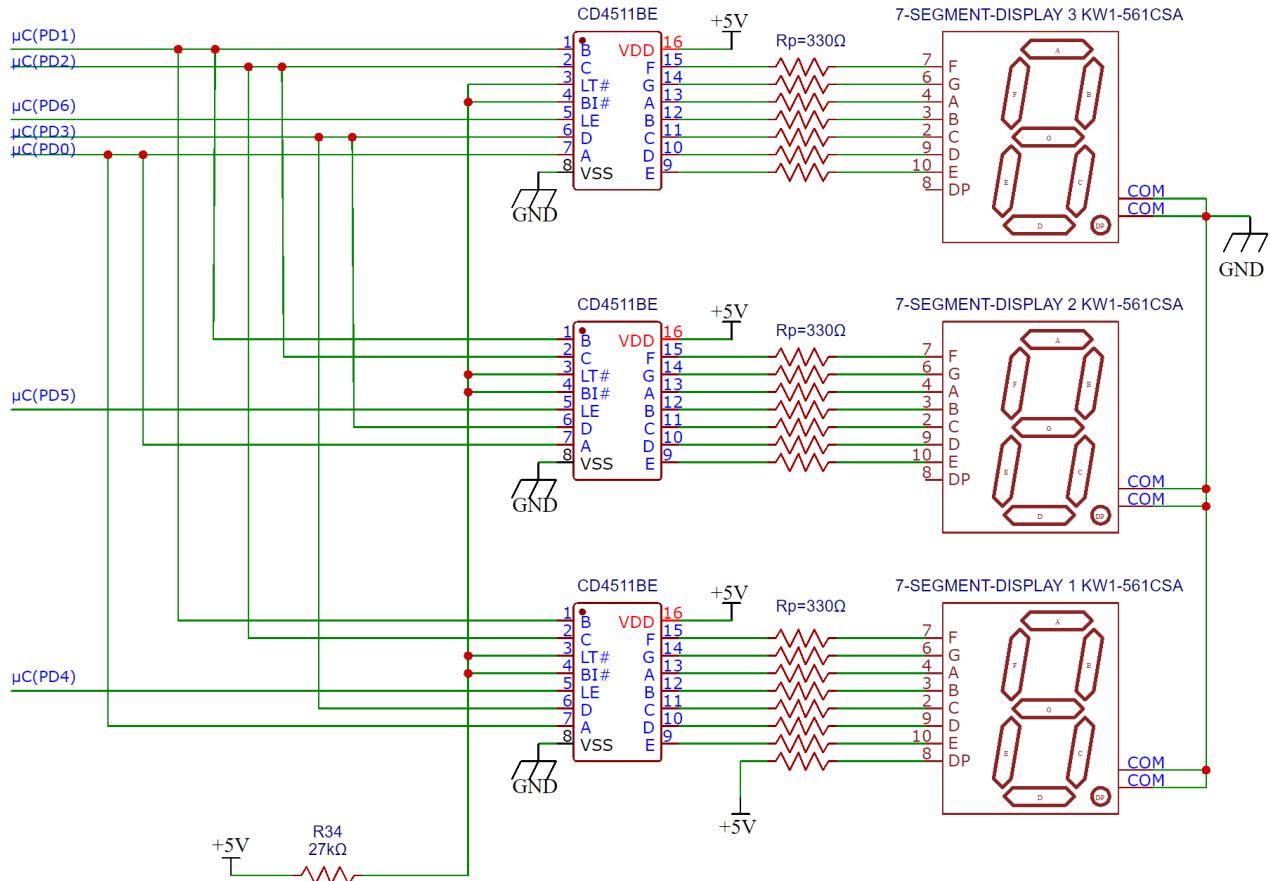
Quella effettiva sarà:

$$V_u = V_{R_3} + V_{R_2} = V_{R_3} + V_{off} = R_3 I_{ph} + V_{off}$$

con I_{ph} = corrente che scorre nel fotodiodo (variabile in base alla luce).

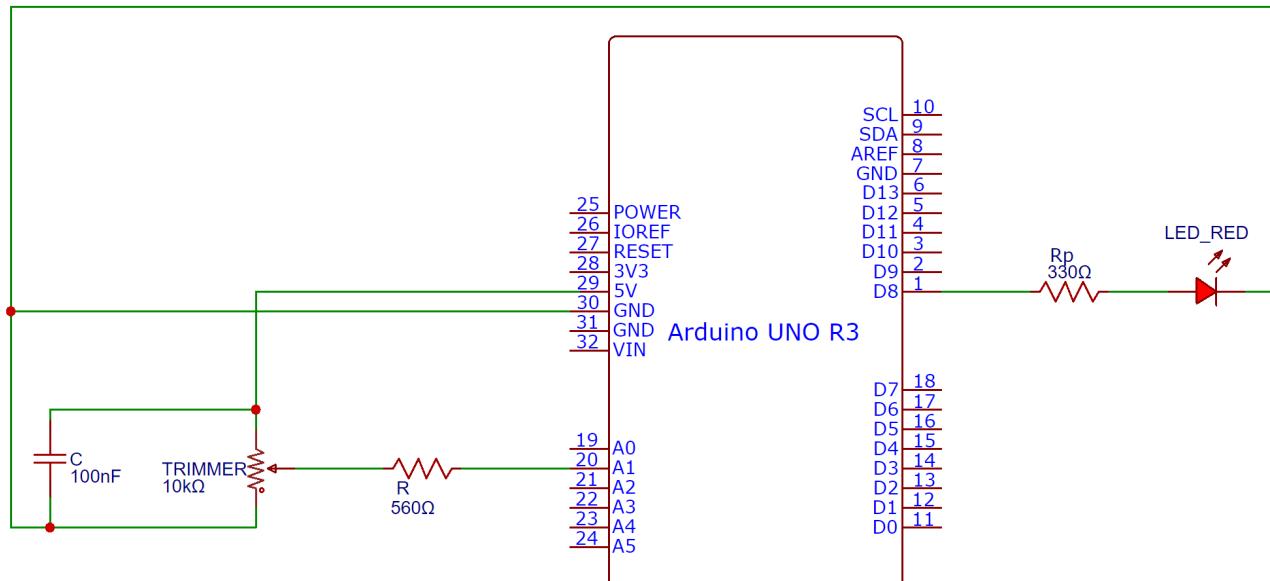
Nota: il condensatore C_1 era originariamente da $1 \mu F$, ma, non essendo presente nel kit, è stato cambiato con uno da $2.2 \mu F$.

3.3.2 Circuito di pilotaggio dei display a sette segmenti



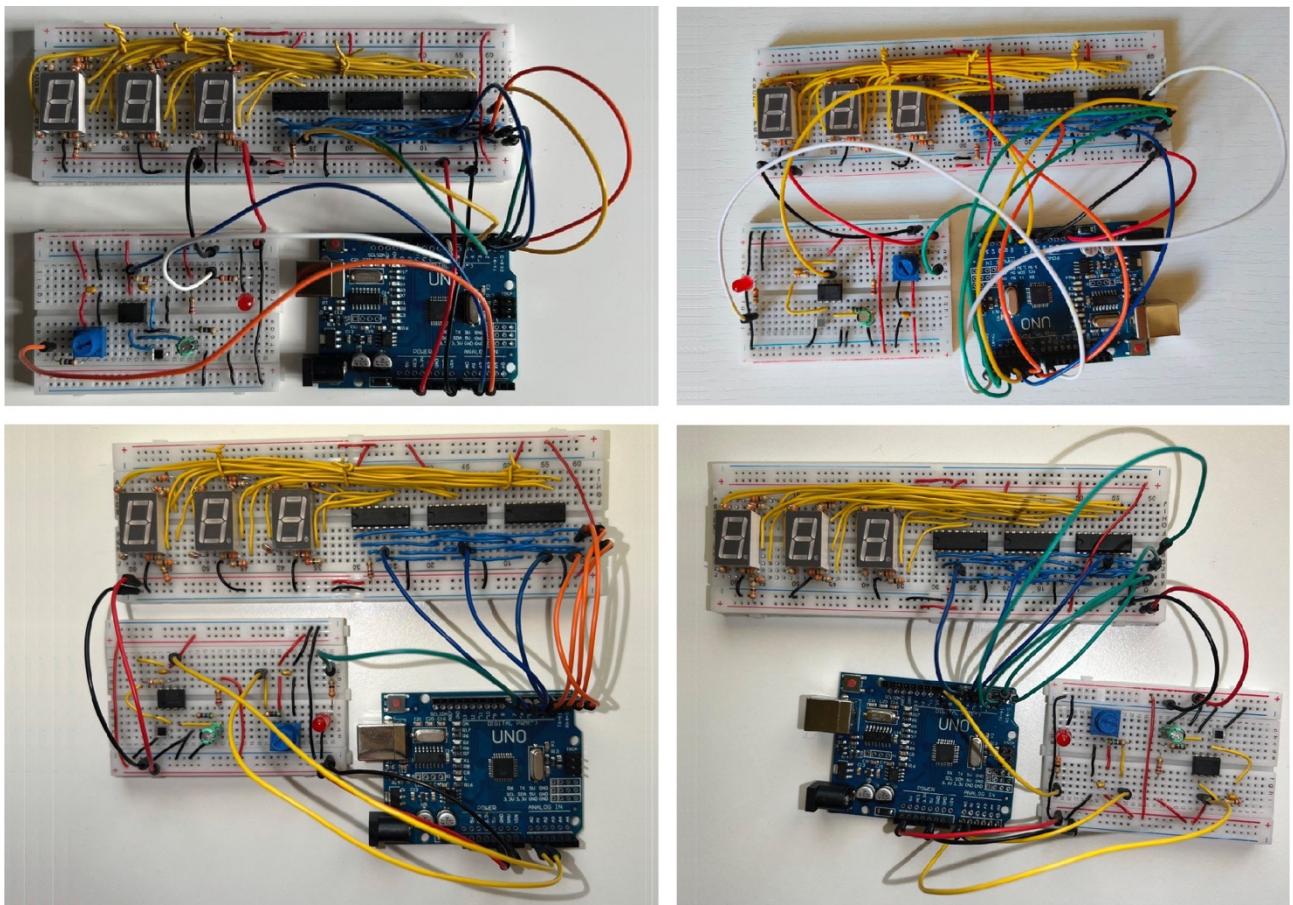
Il circuito di pilotaggio di ciascun display a sette segmenti è costituito dal circuito integrato CD4511, collegato ai pin dei segmenti del display tramite dei resistori di protezione. Esso permette, tramite programmazione, di effettuare la decodifica BCD (*Binary-Coded Decimal*).

3.3.3 Circuito di pilotaggio del LED di allarme



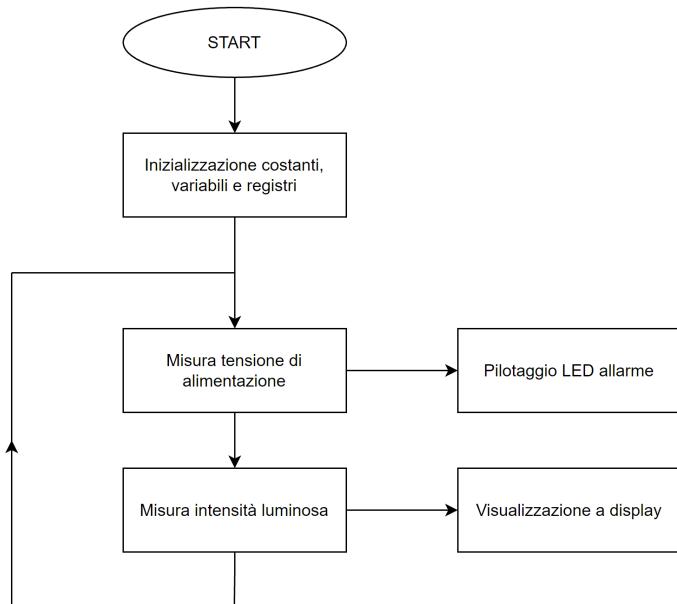
Il diodo LED di allarme è collegato all'alimentazione attraverso il resistore R_p di protezione. Non potendo variare effettivamente la tensione di alimentazione, sfruttiamo un potenziometro, collegato fra V_{al} e GND. La sua uscita arriva al pin A1 del convertitore analogico/digitale della scheda "Arduino Uno R3" attraverso il resistore R di protezione.

3.4 Foto basette montate



4. Descrizione software

4.1 Diagramma a blocchi

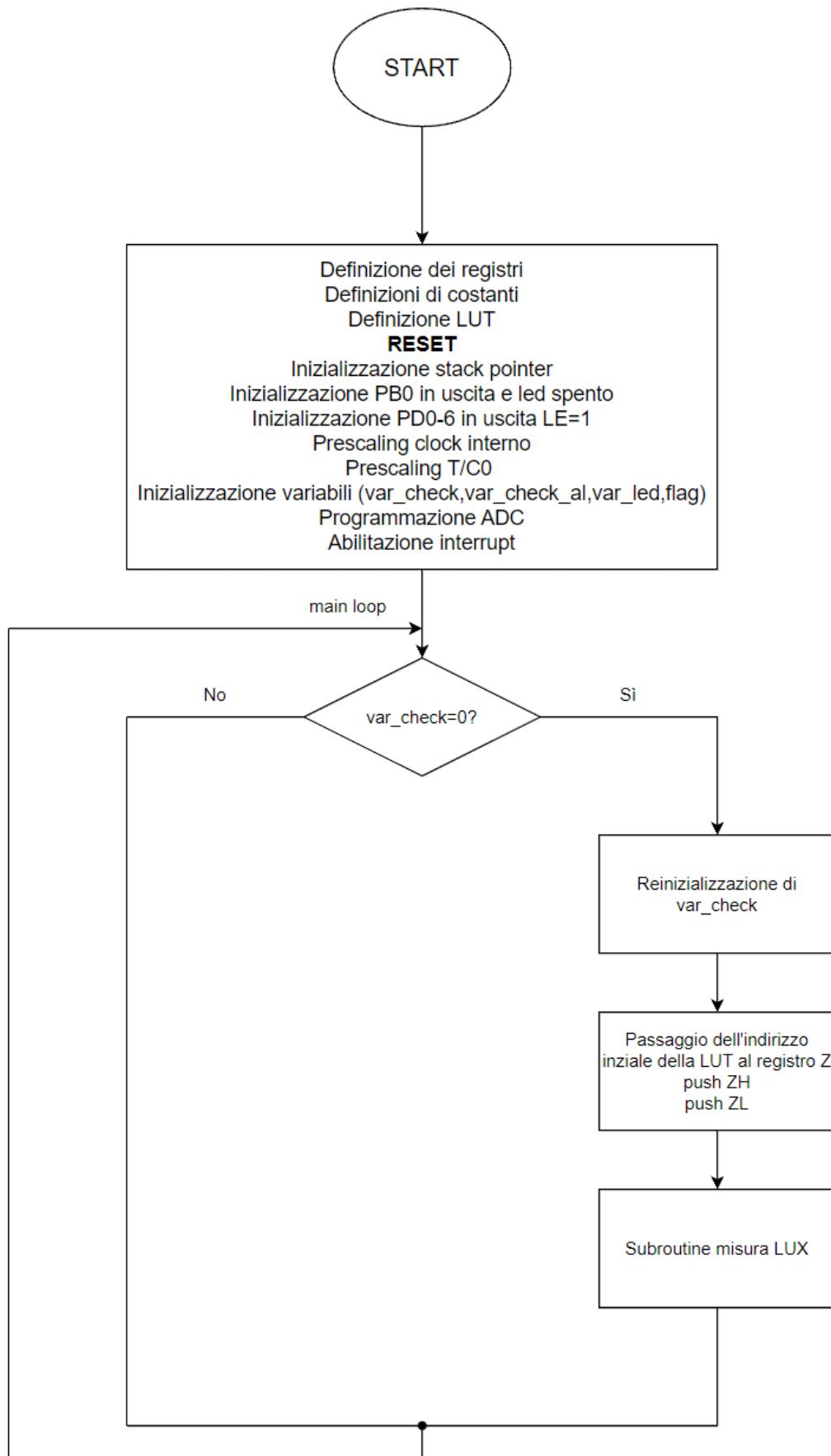


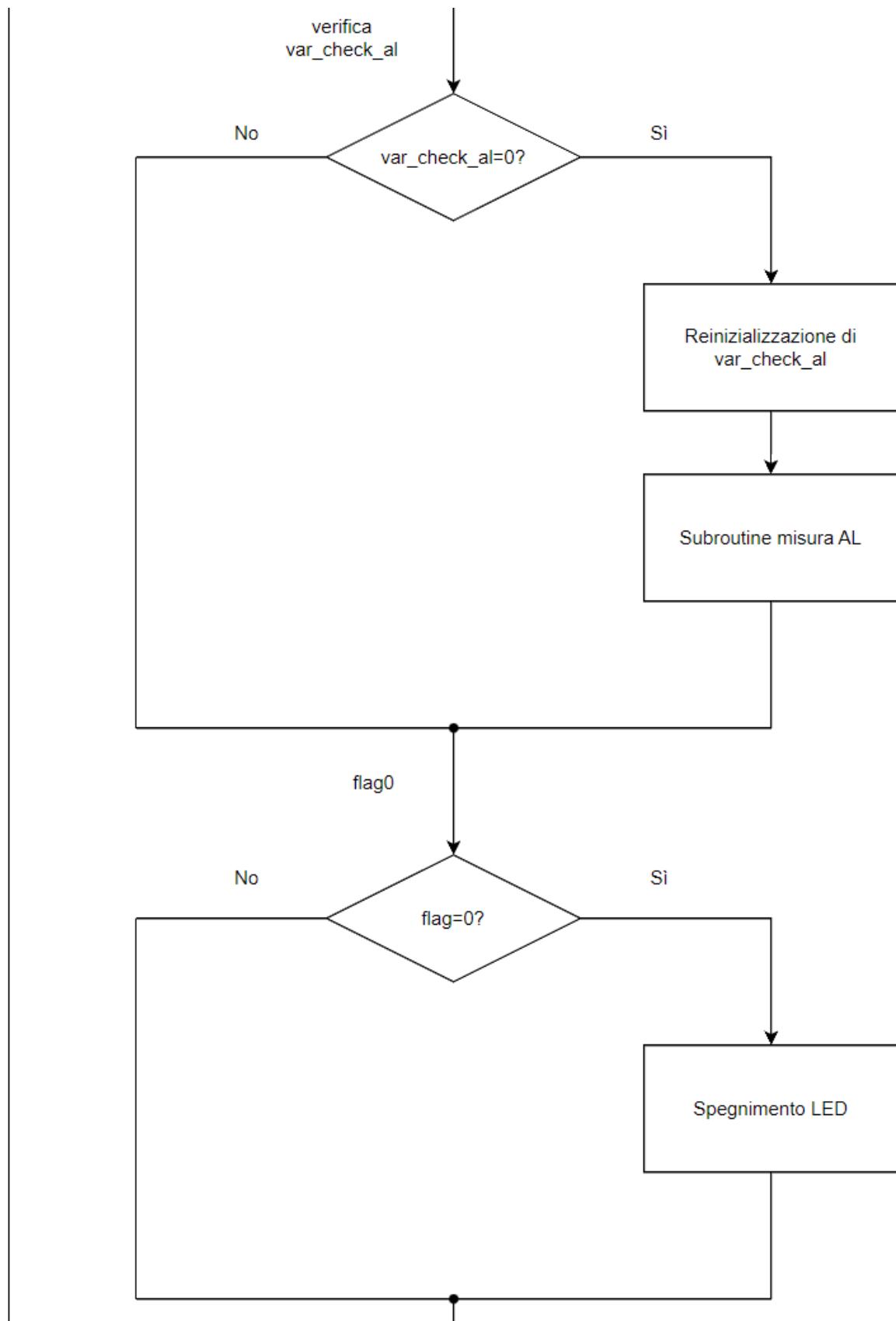
Il diagramma a blocchi del software si divide in tre parti principali:

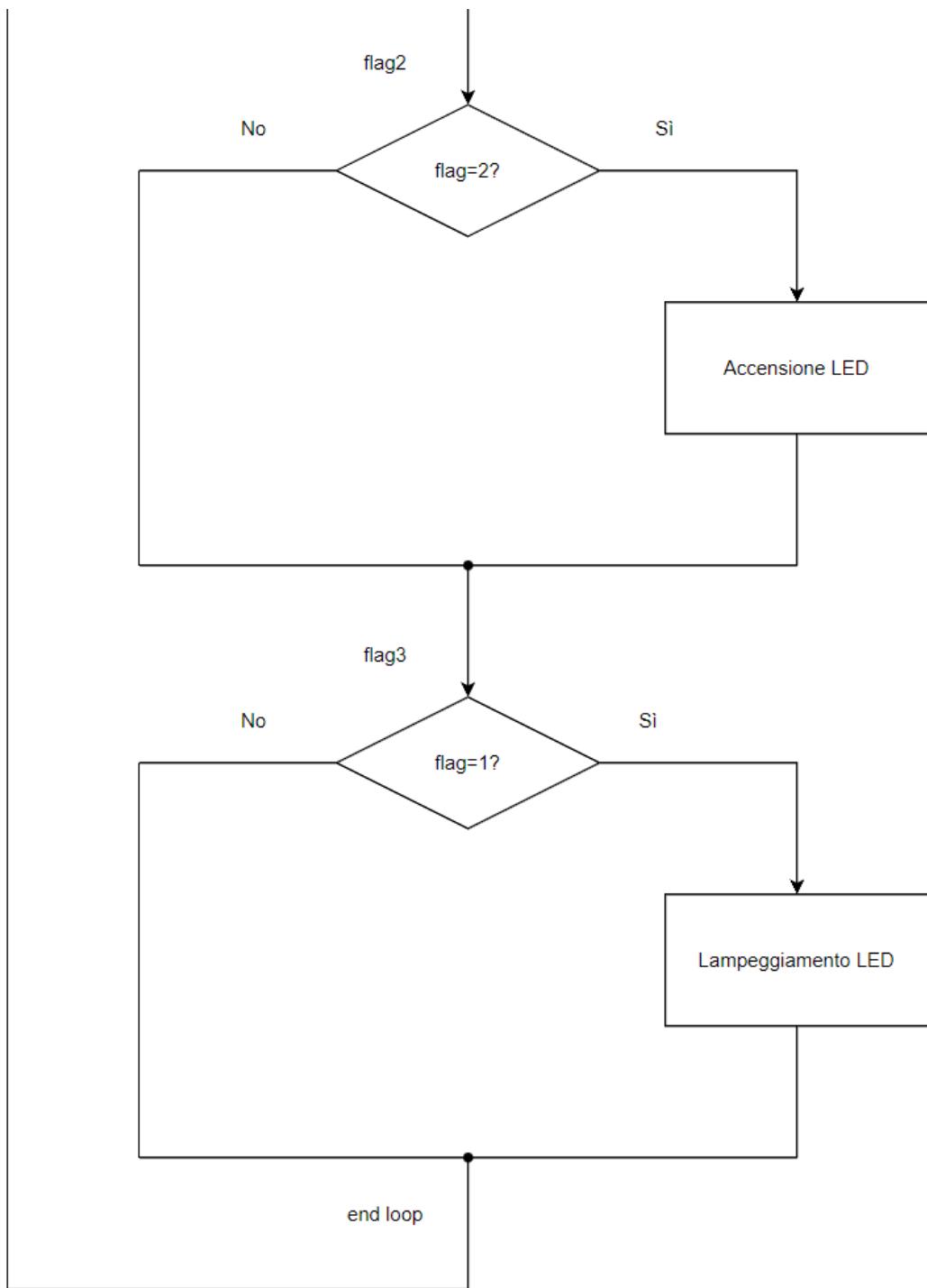
- Inizializzazione delle costanti, della LUT e dei registri utilizzati
- Misura della tensione di alimentazione e sistema di allarme
- Misura dell'intensità luminosa e visualizzazione a display

4.2 Flowchart

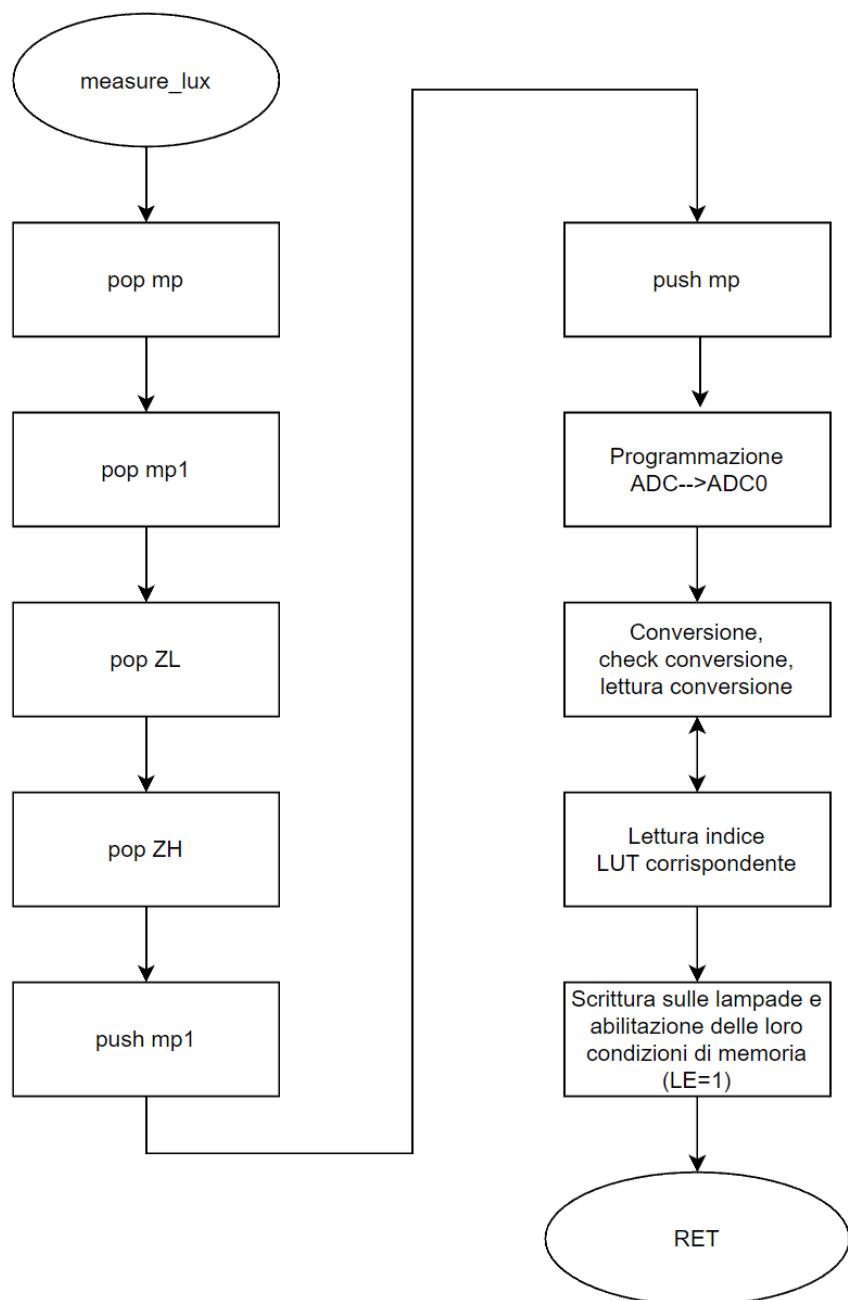
4.2.1 Programma principale



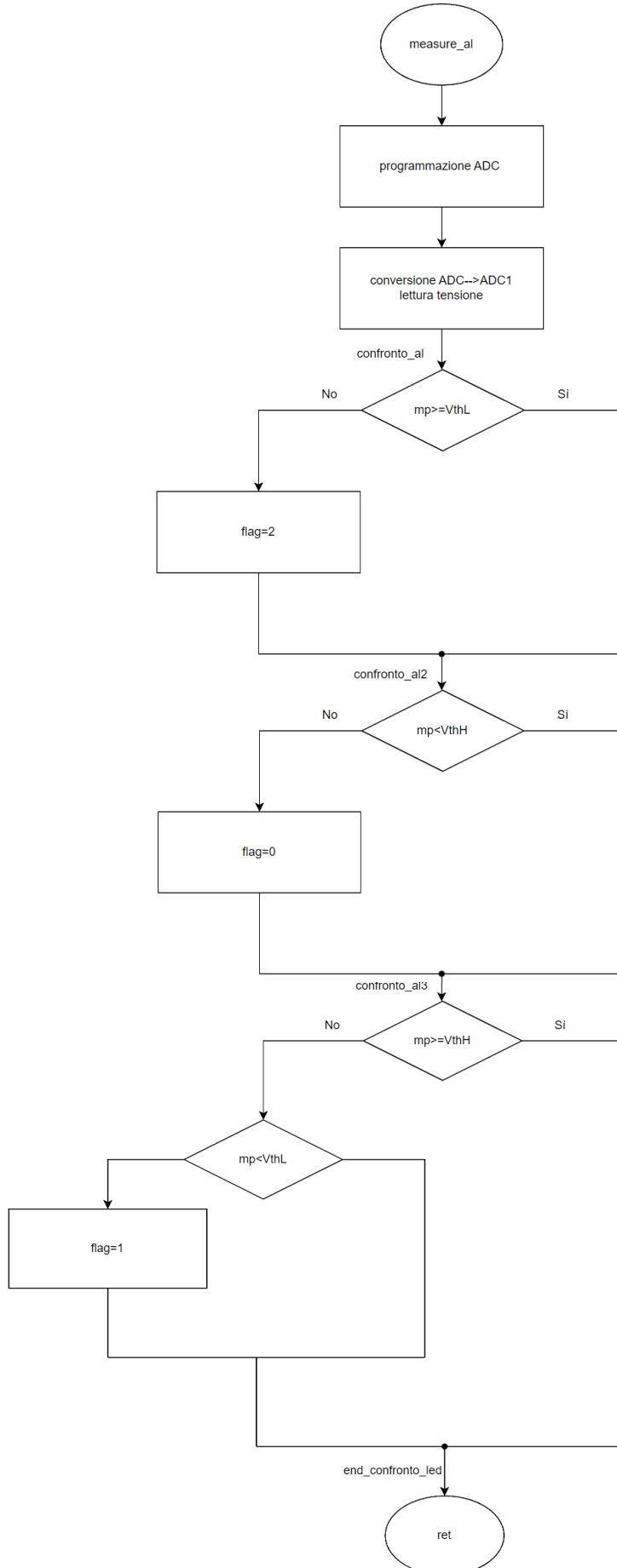




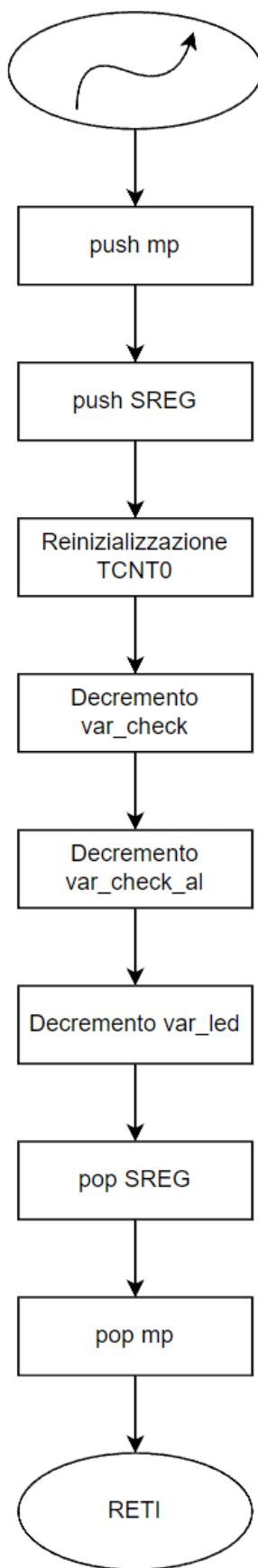
4.2.2 Subroutine *measure_lux*



4.2.3 Subroutine *measure_al*

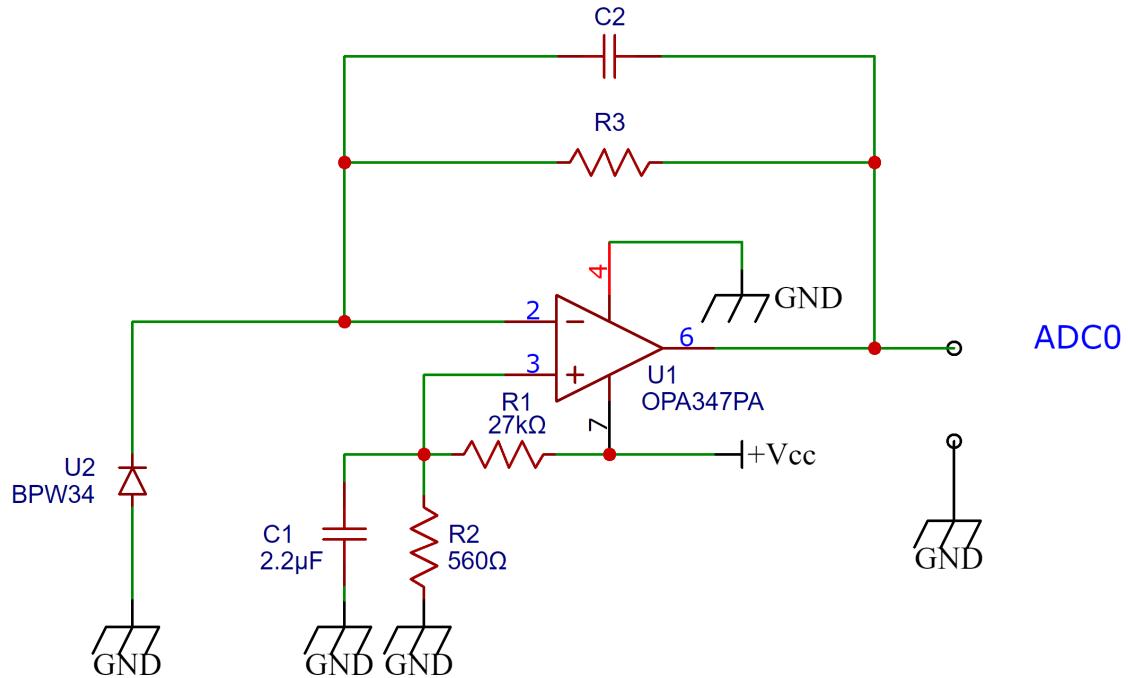


4.2.4 Subroutine di risposta ad interrupt: Overflow del Timer/Counter 0



5. Calcoli progettuali

5.1 Tensione di uscita del circuito di condizionamento del fotodiodo



Dati:

$$V_{al} = 5 \text{ V};$$

$$V_{off} = V_{al} \frac{R_2}{R_2 + R_1} \cong 100 \text{ mV};$$

$$R_3 = 1.5 \text{ M}\Omega;$$

$$S_{nomina} = 80 \frac{nA}{lux} \text{ (sensibilità del fotodiodo BPW34),}$$

la portata del circuito di condizionamento assegnato è calcolabile come:

$$portata_{lux} = \frac{V_{al} - V_{off}}{R_3 \cdot S_{nomina}} \cong 40.83 \text{ lux.}$$

Essa è relativamente bassa e non permette di visualizzare l'intensità luminosa in chilolux. Pertanto, è stato abbassato il valore della resistenza R_3 : $R_3 = 27 \text{ k}\Omega$. La portata del circuito diventa:

$$portata_{lux} = \frac{V_{al} - V_{off}}{R_3 \cdot S_{nomina}} \cong 2.27 \text{ klux.}$$

Tuttavia, sperimentalmente è stata notata una differenza tra la sensibilità nominale e quella reale di un ordine di grandezza: $S_{reale} \cong \frac{1}{10} S_{nomina}$.

Per questo motivo, è stato nuovamente cambiato il valore della resistenza R_3 : $R_3 = 27 \text{ k}\Omega \rightarrow 270 \text{ k}\Omega$.

Per non cambiare eccessivamente la frequenza del polo introdotta da C_2 originaria, è stato cambiato anche il valore di tale condensatore: $C_2 = 1 \mu F \rightarrow 1 nF$.

5.2 Taratura del fotodiodo BPW 34

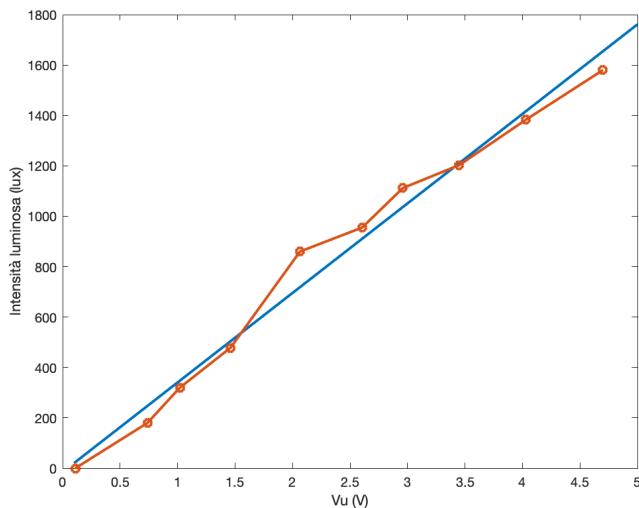
Siccome le misure ottenute utilizzando la sensibilità nominale del fotodiodo non corrispondono alle misure effettuate mediante lo strumento campione a disposizione, è stata effettuata la taratura del sensore.

Sono state raccolti dieci campioni di intensità luminosa e di relativa tensione di uscita dal circuito di condizionamento. Successivamente, sono stati interpolati linearmente questi campioni per ottenere una retta, da cui è stata ricavata un'approssimazione della sensibilità del sensore:

$$S_{reale} \cong 11 \frac{nA}{lux}.$$

La nuova portata risulta quindi essere:

$$portata_{lux} = \frac{V_{al}-V_{off}}{R_3 \cdot S_{reale}} \cong 1.65 \text{ klux}.$$



curva rossa: relazione Vu-lux sperimentale; curva blu: relazione Vu-lux approssimata

5.3 Resistore di protezione di un diodo LED rosso

Per funzionare, un diodo LED rosso necessita di una tensione di circa 1.8 V e di una corrente di circa 10 mA. È possibile calcolare il valore del resistore di protezione da inserire fra la tensione di alimentazione V_{al} e il LED come segue:

$$R_p = \frac{V_{al}-V_{LED\ rosso}}{I_{LED\ rosso}} = 320 \Omega.$$

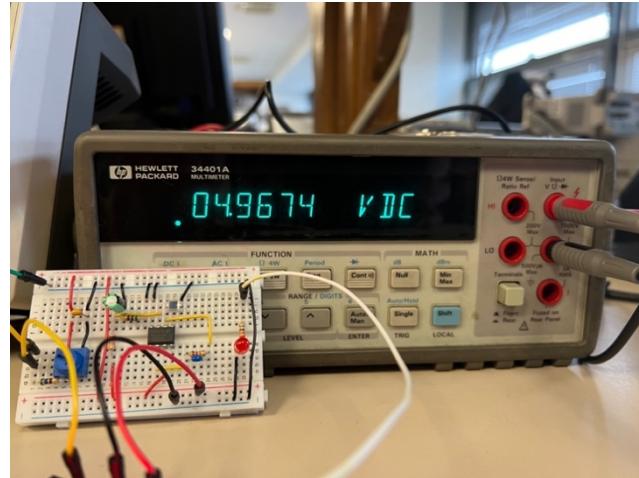
È stato scelto il valore commerciale più vicino alla R_p calcolata: $320 \Omega \rightarrow 330 \Omega$.

Sono stati utilizzati resistori di protezione con resistenza di tale valore per collegare sia il LED di allarme, sia i LED dei display a sette segmenti.

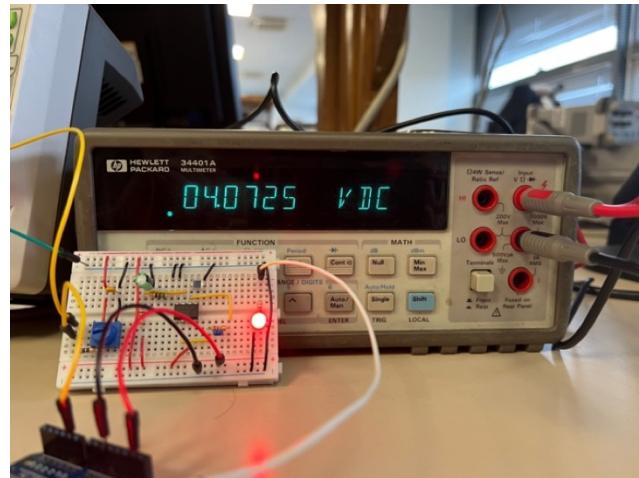
6. Prove di verifica

6.1 LED di allarme

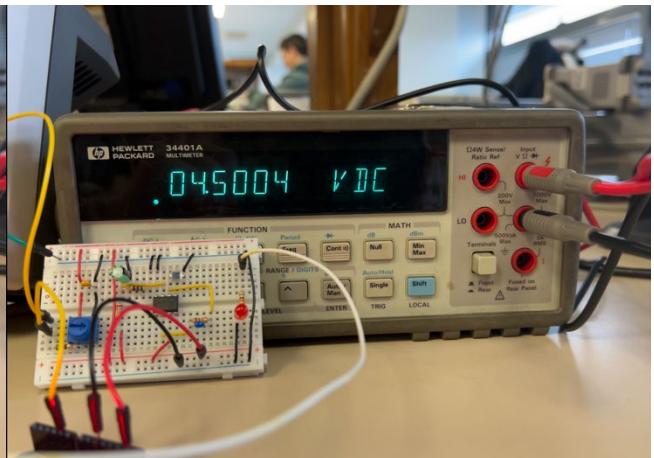
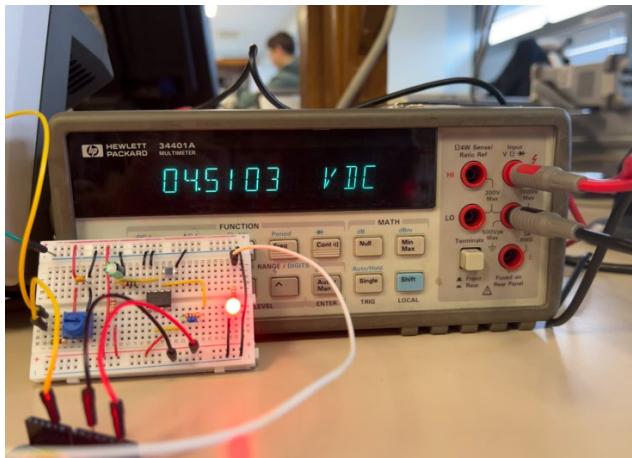
Il LED di allarme della tensione di alimentazione funziona correttamente. Infatti, fornendo alimentazione al circuito e ruotando la manopola del trimmer, che permette di modificare la sua resistenza e quindi la tensione sentita dal convertitore analogico digitale, è possibile osservare una variazione dello stato del LED, come visibile dalle immagini sottoriportate.



LED spento con $V > 4.6$



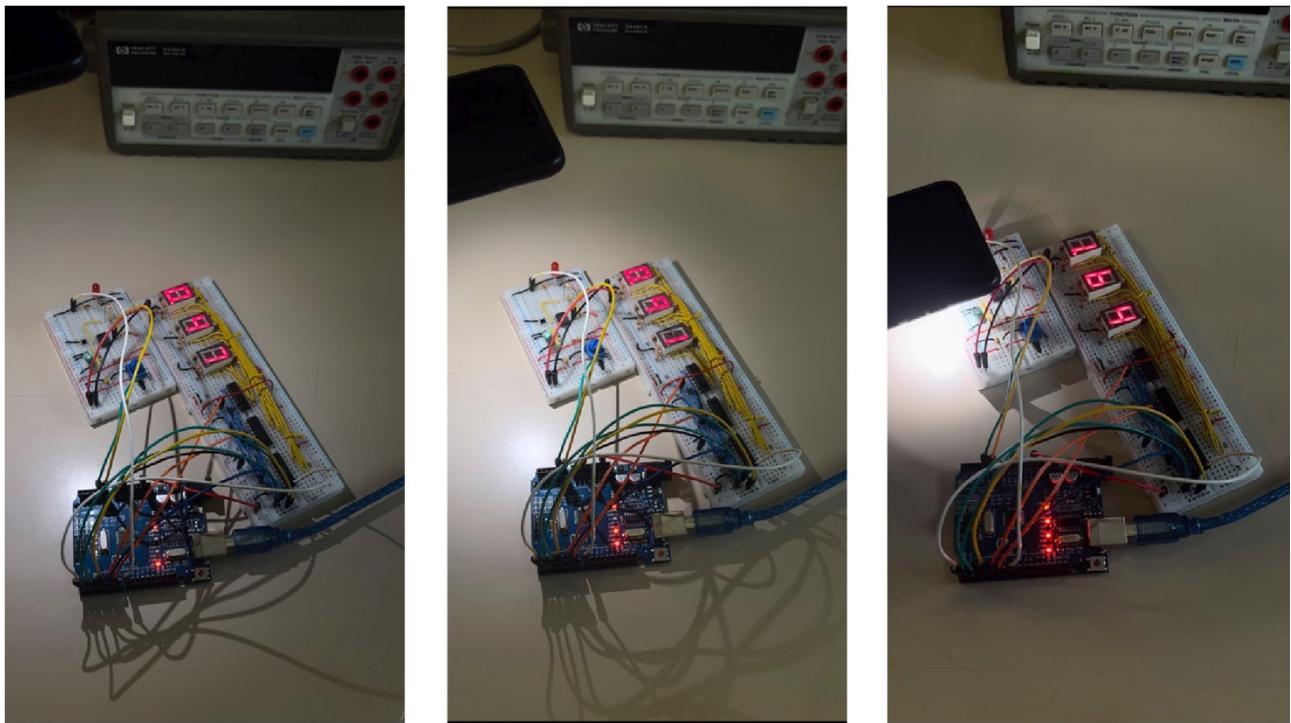
LED acceso con $V < 4.2$



LED lampeggiante con $4.2 \leq V < 4.6$

6.2 Fondo scala

Per verificare il raggiungimento della portata, è stata avvicinata progressivamente una sorgente luminosa al sensore: è stato mostrato un aumento graduale dei lux visualizzati sui display.



6.3 Risoluzione

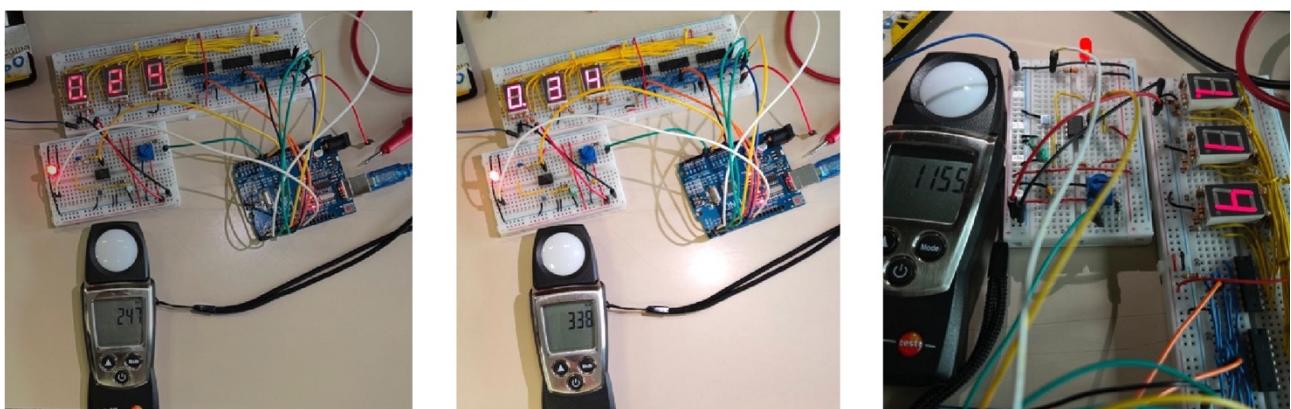
La risoluzione del segnale ottenuto all'uscita del convertitore analogico digitale di "Arduino Uno R3" è:

$$res = \frac{D_{A/D}}{2^N} \cdot \frac{1}{R_3 \cdot S_{reale}} \cong 6.6 \text{ lux.}$$

Tuttavia, utilizzando solo tre display a sette segmenti per indicare la misura dell'intensità luminosa, la risoluzione consentita dal luxmetro realizzato è di **10 lux**.

6.4 Accuratezza

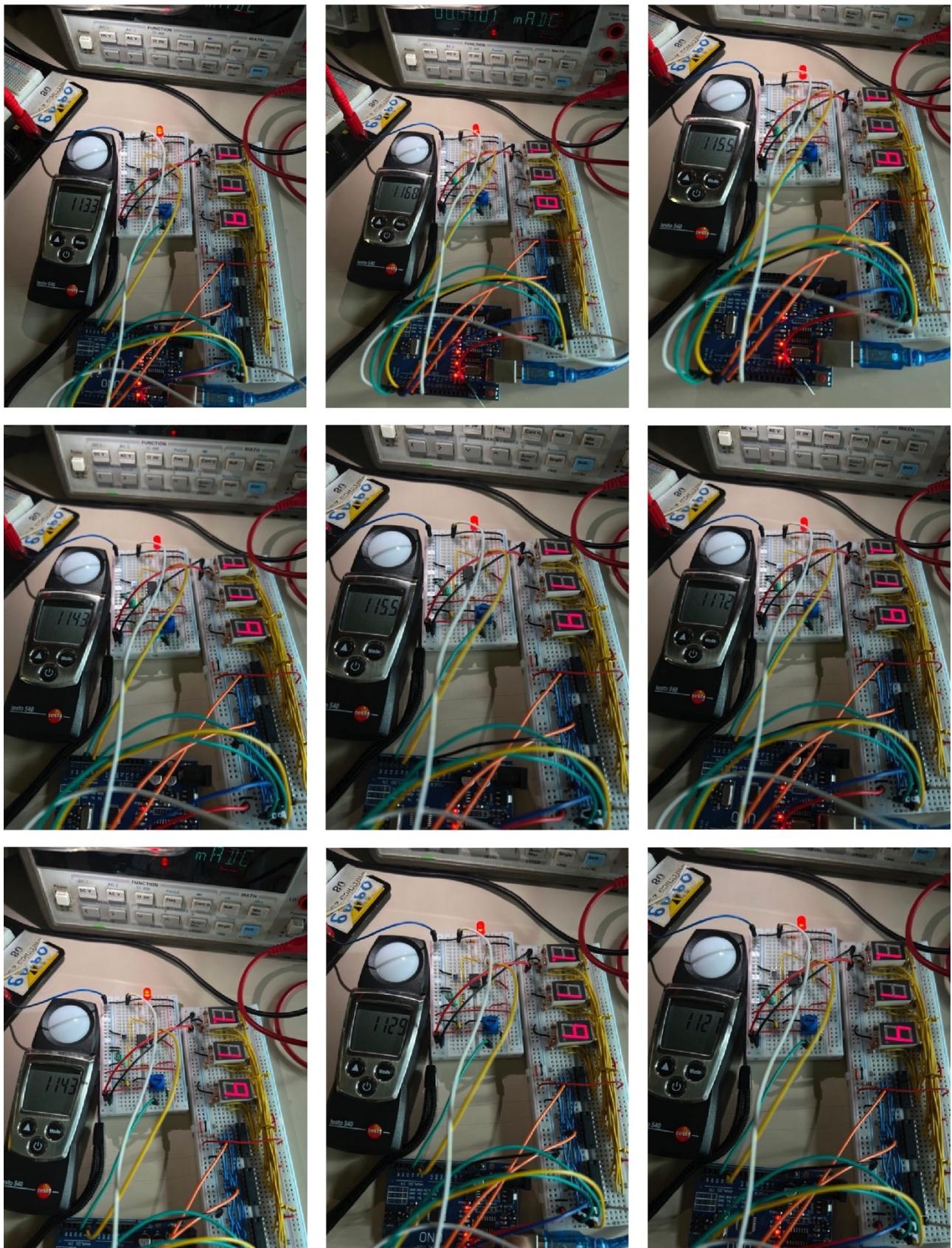
Effettuando misurazioni a campione, abbiamo verificato un'accuratezza soddisfacente, come possibile osservare da alcuni esempi riportati in seguito.



confronto misura luxmetro-luxmetro campione

6.5 Ripetibilità

A causa della variabilità delle condizioni di lavoro, ad esempio della condizione di luminosità, in cui è stata effettuata tale verifica, le misure ottenute non sono indicative della ripetibilità.



6.6 Corrente assorbita

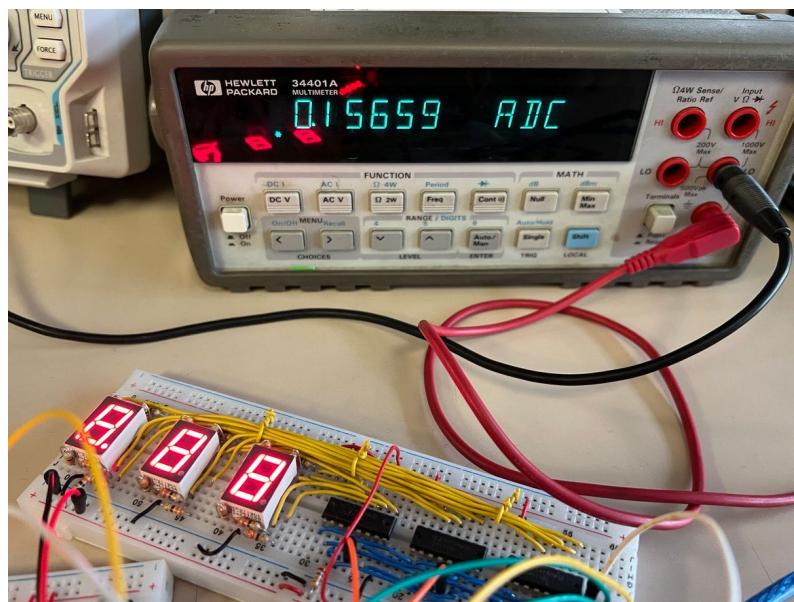
Un'ulteriore prova di verifica effettuata riguarda la misura della corrente assorbita dal dispositivo nelle condizioni di massimo consumo, visibile nella foto sottostante.

Il valore ottenuto è pari a circa 150 mA, che non rispecchia a pieno il valore teorico trovato.

Quest'ultimo è stato stimato tenendo conto dei componenti presenti nel dispositivo, quali, ad esempio, i display a sette segmenti, il LED di allarme, la scheda "Arduino Uno R3" e i circuiti integrati.

Considerando un assorbimento di circa 10 mA da parte dei LED rossi, di circa 12 mA da parte della scheda "Arduino Uno R3" (si ha un range tra circa 10 mA e circa 15 mA) e trascurando il consumo dei circuiti integrati, si ottiene un valore di circa 230 mA.

La discordanza fra il valore teorico e quello ottenuto potrebbe essere dovuta all'elevata incertezza associata all'effettivo assorbimento di corrente da parte dei segmenti del display, che potrebbe variare da segmento a segmento e con la temperatura.



7. Conclusioni

In seguito alle prove di verifica sostenute è possibile affermare che il luxmetro è funzionante. Sono state garantite una portata di 1.65 klux e una risoluzione di 10 lux.

Per quanto riguarda il sistema di allarme dell'alimentazione è stata scelta una soluzione a doppia soglia. Nel caso in cui si dovesse utilizzare una pila da 5 V per alimentare il circuito, la tensione da essa fornita diminuirebbe nel tempo.

Per valori di tensione inferiori a 4.2 V, il circuito potrebbe non funzionare correttamente. Per avvisare l'utente di tale problema il LED di allarme si accende.

È stato ritenuto utile avvisare l'utente prima di raggiungere il valore della tensione di alimentazione "critico" sopracitato. Per tale motivo, è stato programmato il LED affinché lampeggi per valori di tensione compresi fra 4.2 V e 4.6 V, in modo che l'utente abbia il tempo necessario per un eventuale cambio di pila.

Per valori superiori a 4.6 V, il circuito dovrebbe funzionare correttamente e quindi il LED è spento.

Appendice

Listato del software

```
;  
; Definizione dei registri utilizzati  
.DEF mp=R16 ; registro di lavoro (generico)  
.DEF mp1=R17 ; registro di lavoro secondario (generico)  
.DEF var_check=R18 ; var_check è la variabile decrementata dalla subroutine  
 ; di risposta ad interrupt  
 ; quando arriva a 0 viene attivato un campionamento dell'ADC.  
.DEF var_check_al=R19 ; variabile decrementata dalla subroutine di risposta ad  
 ; interrupt per il led di allarme dell'alimentazione  
.DEF var_led=R20 ; variabile decrementata dalla subroutine di risposta ad  
 ; interrupt per il lampeggiamento del LED di allarme  
 ; dell'alimentazione  
;  
.DEF flag = R21 ; flag che indica in che stato ci si trova:  
 ; Val <4.2V; 4.2V <= Val < 4.6 V; Val >= 4.6 V  
;  
.EQU vthL = 215 ; valore di soglia di 4.2 V della tensione di  
 ; alimentazione  
.EQU vthH = 235 ; valore di soglia di 4.6 V della tensione di  
 ; alimentazione  
;  
;  
.EQU T_lamp=50 ; costante che definisce l'intervallo di lampeggiamento  
 ; del LED di allarme in multipli di 10ms (in questo caso  
 ; 500ms)  
.EQU T_CHECK=100 ; costante che definisce l'intervallo di campionamento  
 ; della tensione di alimentazione e della tensione Vu in  
 ; multipli di 10ms (in questo caso 1s)  
.EQU TABLEN=255 ; lunghezza della tabella (in questo caso pari  
 ; all'intervallo di valori all'uscita di un ADC ad 8 bit)  
;  
; Crea tabella nella flash. In questo caso la tabella ha 255 celle raggruppate a gruppi  
 ; di 4.  
; La prima cella di ogni riga è l'indice della tabella, le altre tre rispettivamente i  
 ; valori delle unità,  
 ; della prima cifra decimale e della seconda cifra decimale.  
 ; Valori possibili della conversione su 8 bit da 0 a 255.  
;  
; Cambiando la LUT si può visualizzare la tensione o l'intensità luminosa.  
;  
.CSEG  
.ORG 0x1FFF ; definisce l'inizio della tabella che sarà scritta in flash  
 ; (bisogna essere certi di non sovrascriverla con il programma)  
;  
tabella: ; label che punta all'indirizzo di inizio della tabella  
.db 0,0,0,0  
.db 1,0,0,0  
.db 2,0,0,0  
.db 3,0,0,0  
.db 4,0,0,0  
.db 5,0,0,1  
.db 6,0,0,1  
.db 7,0,0,2  
.db 8,0,0,3  
.db 9,0,0,3  
.db 10,0,0,4  
.db 11,0,0,5  
.db 12,0,0,5  
.db 13,0,0,6  
.db 14,0,0,7
```

```
.db 15,0,0,7
.db 16,0,0,8
.db 17,0,0,9
.db 18,0,0,9
.db 19,0,1,0
.db 20,0,1,1
.db 21,0,1,1
.db 22,0,1,2
.db 23,0,1,2
.db 24,0,1,3
.db 25,0,1,4
.db 26,0,1,4
.db 27,0,1,5
.db 28,0,1,6
.db 29,0,1,6
.db 30,0,1,7
.db 31,0,1,8
.db 32,0,1,8
.db 33,0,1,9
.db 34,0,2,0
.db 35,0,2,0
.db 36,0,2,1
.db 37,0,2,2
.db 38,0,2,2
.db 39,0,2,3
.db 40,0,2,4
.db 41,0,2,4
.db 42,0,2,5
.db 43,0,2,6
.db 44,0,2,6
.db 45,0,2,7
.db 46,0,2,8
.db 47,0,2,8
.db 48,0,2,9
.db 49,0,3,0
.db 50,0,3,0
.db 51,0,3,1
.db 52,0,3,2
.db 53,0,3,2
.db 54,0,3,3
.db 55,0,3,4
.db 56,0,3,4
.db 57,0,3,5
.db 58,0,3,5
.db 59,0,3,6
.db 60,0,3,7
.db 61,0,3,7
.db 62,0,3,8
.db 63,0,3,9
.db 64,0,3,9
.db 65,0,4,0
.db 66,0,4,1
.db 67,0,4,1
.db 68,0,4,2
.db 69,0,4,3
.db 70,0,4,3
.db 71,0,4,4
.db 72,0,4,5
.db 73,0,4,5
.db 74,0,4,6
.db 75,0,4,7
.db 76,0,4,7
.db 77,0,4,8
.db 78,0,4,9
```

.db 79,0,4,9
.db 80,0,5,0
.db 81,0,5,1
.db 82,0,5,1
.db 83,0,5,2
.db 84,0,5,3
.db 85,0,5,3
.db 86,0,5,4
.db 87,0,5,5
.db 88,0,5,5
.db 89,0,5,6
.db 90,0,5,7
.db 91,0,5,7
.db 92,0,5,8
.db 93,0,5,9
.db 94,0,5,9
.db 95,0,6,0
.db 96,0,6,0
.db 97,0,6,1
.db 98,0,6,2
.db 99,0,6,2
.db 100,0,6,3
.db 101,0,6,4
.db 102,0,6,4
.db 103,0,6,5
.db 104,0,6,6
.db 105,0,6,6
.db 106,0,6,7
.db 107,0,6,8
.db 108,0,6,8
.db 109,0,6,9
.db 110,0,7,0
.db 111,0,7,0
.db 112,0,7,1
.db 113,0,7,2
.db 114,0,7,2
.db 115,0,7,3
.db 116,0,7,4
.db 117,0,7,4
.db 118,0,7,5
.db 119,0,7,6
.db 120,0,7,6
.db 121,0,7,7
.db 122,0,7,8
.db 123,0,7,8
.db 124,0,7,9
.db 125,0,8,0
.db 126,0,8,0
.db 127,0,8,1
.db 128,0,8,2
.db 129,0,8,2
.db 130,0,8,3
.db 131,0,8,3
.db 132,0,8,4
.db 133,0,8,5
.db 134,0,8,5
.db 135,0,8,6
.db 136,0,8,7
.db 137,0,8,7
.db 138,0,8,8
.db 139,0,8,9
.db 140,0,8,9
.db 141,0,9,0
.db 142,0,9,1

.db 143,0,9,1
.db 144,0,9,2
.db 145,0,9,3
.db 146,0,9,3
.db 147,0,9,4
.db 148,0,9,5
.db 149,0,9,5
.db 150,0,9,6
.db 151,0,9,7
.db 152,0,9,7
.db 153,0,9,8
.db 154,0,9,9
.db 155,0,9,9
.db 156,1,0,0
.db 157,1,0,1
.db 158,1,0,1
.db 159,1,0,2
.db 160,1,0,3
.db 161,1,0,3
.db 162,1,0,4
.db 163,1,0,5
.db 164,1,0,5
.db 165,1,0,6
.db 166,1,0,6
.db 167,1,0,7
.db 168,1,0,8
.db 169,1,0,8
.db 170,1,0,9
.db 171,1,1,0
.db 172,1,1,0
.db 173,1,1,1
.db 174,1,1,2
.db 175,1,1,2
.db 176,1,1,3
.db 177,1,1,4
.db 178,1,1,4
.db 179,1,1,5
.db 180,1,1,6
.db 181,1,1,6
.db 182,1,1,7
.db 183,1,1,8
.db 184,1,1,8
.db 185,1,1,9
.db 186,1,2,0
.db 187,1,2,0
.db 188,1,2,1
.db 189,1,2,2
.db 190,1,2,2
.db 191,1,2,3
.db 192,1,2,4
.db 193,1,2,4
.db 194,1,2,5
.db 195,1,2,6
.db 196,1,2,6
.db 197,1,2,7
.db 198,1,2,8
.db 199,1,2,8
.db 200,1,2,9
.db 201,1,2,9
.db 202,1,3,0
.db 203,1,3,1
.db 204,1,3,1
.db 205,1,3,2
.db 206,1,3,3

```

.db 207,1,3,3
.db 208,1,3,4
.db 209,1,3,5
.db 210,1,3,5
.db 211,1,3,6
.db 212,1,3,7
.db 213,1,3,7
.db 214,1,3,8
.db 215,1,3,9
.db 216,1,3,9
.db 217,1,4,0
.db 218,1,4,1
.db 219,1,4,1
.db 220,1,4,2
.db 221,1,4,3
.db 222,1,4,3
.db 223,1,4,4
.db 224,1,4,5
.db 225,1,4,5
.db 226,1,4,6
.db 227,1,4,7
.db 228,1,4,7
.db 229,1,4,8
.db 230,1,4,9
.db 231,1,4,9
.db 232,1,5,0
.db 233,1,5,1
.db 234,1,5,1
.db 235,1,5,2
.db 236,1,5,2
.db 237,1,5,3
.db 238,1,5,4
.db 239,1,5,4
.db 240,1,5,5
.db 241,1,5,6
.db 242,1,5,6
.db 243,1,5,7
.db 244,1,5,8
.db 245,1,5,8
.db 246,1,5,9
.db 247,1,6,0
.db 248,1,6,0
.db 249,1,6,1
.db 250,1,6,2
.db 251,1,6,2
.db 252,1,6,3
.db 253,1,6,4
.db 254,1,6,4
.db 255,1,6,5
;
;
.CSEG
.ORG 0x0000          ; definisce l'inizio del codice all'indirizzo
                      ; 0x0000 (obbligatorio)
;
; INTERRUPT VECTORS FOLLOW
;
    jmp RESET           ; vector 1:  Reset Handler
    jmp EXT_INT0         ; vector 2:  IRQ0 Handler
    jmp EXT_INT1         ; vector 3:  IRQ1 Handler
    jmp PCINTR0          ; vector 4:  PCINT0 Handler
    jmp PCINTR1          ; vector 5:  PCINT1 Handler
    jmp PCINTR2          ; vector 6:  PCINT2 Handler
    jmp WDT              ; vector 7:  Watchdog timer handler

```

```

        jmp TIM2_COMPA          ; vector 8:  Timer2 Compare A handler
        jmp TIM2_COMPB          ; vector 9:   Timer2 compare B handler
        jmp TIM2_OVF            ; vector 10:  Timer2 Overflow Handler
        jmp TIM1_CAPT           ; vector 11:  Timer1 Capture Handler
        jmp TIM1_COMPA           ; vector 12:  Timer1 CompareA Handler
        jmp TIM1_COMPB           ; vector 13:  Timer1 CompareB Handler
        jmp TIM1_OVF            ; vector 14:  Timer1 Overflow Handler
        jmp TIM0_COMPA           ; vector 15:  Timer 0 CompareA handler
        jmp TIM0_COMPB           ; vector 16:  Timer 0 CompareB handler
        jmp TIM0_OVF            ; vector 17:  Timer0 Overflow Handler
        jmp SPI_STC             ; vector 18:  SPI Transfer Complete Handler
        jmp USART_RXC           ; vector 19:  USART RX Complete Handler
        jmp USART_UDRE          ; vector 20:  USART UDR Empty Handler
        jmp USART_TXC            ; vector 21:  USART TX Complete Handler
        jmp ADC_conv             ; vector 22:  ADC Conversion Complete Handler
        jmp EE_RDY               ; vector 23:  EEPROM Ready Handler
        jmp ANA_COMP             ; vector 24:  Analog Comparator Handler
        jmp TWI                  ; vector 25:  Two-wire Serial Interface Handler
        jmp SPM_RDY              ; vector 26:  Store Program Memory Ready Handler

;
; END OF INTERRUPT VECTORS
;

RESET:
; Inizio del programma principale
;
; Inizializzazione dello stack pointer all'ultima cella della RAM
;
        ldi      mp,HIGH(RAMEND)
        out     SPH,mp
        ldi      mp,LOW(RAMEND)
        out     SPL,mp
;
; Inizializzazione del bit 0 di PORT B per il pilotaggio del LED di controllo della tensione
; di alimentazione
;
        ldi      mp,0b0000_0001
        out     DDRB,mp
;
        ldi      mp, 0b0000_0000          ; inizializzazione del LED di controllo
                                         ; dell'alimentazione spento
        out     PORTB, mp
;
; Inizializzazione in uscita dei primi 7 bit della porta D. PD0 - PD3 = ABCD; PD4 = LE
; unità; PD5 = LE prima cifra decimale; PD6 = LE seconda cifra decimale.
;
        ldi      mp,0b0111_1111
        out     DDRD,mp
;
        ldi      mp, 0b0111_0000          ; abilitazione dei LE (LE = 1) dei 4511
                                         ; (condizione di memoria)
        out     PORTD, mp
;
; Divisione della frequenza di clock interno per 16 (16 MHz)
        ldi      mp, 0b1000_0000
        sts     CLKPR, mp              ; abilitazione la programmazione del prescaler (deve
                                         ; sempre precedere l'istruzione seguente)
        ldi      mp, 0b0000_0100
        sts     CLKPR, mp              ; programmazione della divisione per 16 del clock interno
;
; Programmazione del Timer/Counter 0
;
; Selezione del prescaler con passo 1024 (1 MHz --> 1024 us)
        ldi      mp,0b0000_0101
        out     TCCR0B,mp

```

```

; Selezione del tempo tra interrupt pari a circa 10ms (10,24 ms)
    ldi      mp, 246
    out     TCNT0,mp
; Abilitazione dell'interrupt in caso di overflow di TCNT0
    ldi      mp,0b0000_0001
    sts     TIMSK0,mp
;
; Inizializzazione della variabile var_check che sarà decrementata dalla subroutine di
; risposta ad interrupt
;
    ldi      var_check, T_CHECK
;
; Inizializzazione della variabile var_check_al che sarà decrementata dalla subroutine di
; risposta ad interrupt
;
    ldi      var_check_al, T_CHECK
;
;
; Programmazione dell'ADC in modo da abilitarlo senza abilitare l'interrupt e selezione di
; un fattore di prescaling pari a 4.
; Deve SEMPRE precedere la selezione dell'ingresso: non si può selezionare l'ingresso
; dell'ADC se questo non è stato preventivamente abilitato
    ldi      mp,0b100_00010
    sts     ADCSRA,mp
;
; Programmazione dell'ADC perché lavori con riferimento interno pari alla tensione di
; alimentazione (AVcc), giustifichi a sinistra il risultato e senta l'input su PC0 (ADC0) -
; quindi la tensione analogica andrà collegata su PC0
;
    ldi      mp,0b0110_0000
    sts     ADMUX,mp
;
;
; Inizializzazione flag = 0
;
    ldi      flag,0b0000_0000
;
; Inizializzazione della variabile var_led che sarà decrementata dalla subroutine di
; risposta ad interrupt
;
    ldi      var_led,T_lamp
;
; Abilitazione degli interrupt a livello di SREG
;
    sei
;
;
main_loop:
;
; Verifica se è trascorso un tempo pari all'intervallo di campionamento della tensione
; (var_check = 0?)
;
    cpi      var_check,0
;
; Salto condizionato a verifica_var_check_al se var_check è diverso da 0 (cioè se non è
; trascorso l'intervallo di tempo programmato per campionare la tensione)
;
    brne  verifica_var_check_al
;
; Le istruzioni da qui a rjmp main_loop vengono eseguite solo se var_check è pari a 0
;
    ldi      var_check,T_CHECK          ; per prima cosa viene inizializzata nuovamente
                                         ; var_check
;

```

```

;      Prepara il passaggio dell'indirizzo iniziale della tabella come variabile locale
;      nello stack
;
    ldi    ZH,high(2*tabella)          ; inizializzazione di ZH con la parte alta
                                      ; dell'indirizzo della tabella.
    ldi    ZL,low(2*tabella)          ; inizializzazione di ZL con la parte bassa
                                      ; dell'indirizzo della tabella.
; l'indirizzo di tabella è moltiplicato per due perché le celle della flash sono da 16 bit
;
    push   ZH                      ; mette ZH nello stack per passarlo alla subroutine
                                      ; che lo utilizzerà come parametro
;
    push   ZL                      ; mette ZL nello stack per passarlo alla subroutine
                                      ; che lo utilizzerà come parametro
;
; Richiamo della subroutine measure_lux che campiona la tensione e visualizza il risultato
; (attenzione, la call memorizza nello stack i due byte dell'indirizzo di rientro)
;
    call  measure_lux
;
;
verifica_var_check_al:           ; verifica se è trascorso un tempo pari all'intervallo di
                                      ; campionamento della tensione di alimentazione
                                      ; (var_check_al = 0?)
;
    cpi      var_check_al,0
;
; Salto condizionato a flag0 se var_check_al è diverso da 0 (cioè se non è trascorso
; l'intervallo di tempo programmato per campionare la tensione)
;
    brne   flag0
;
; Le istruzioni da qui a rjmp main_loop vengono eseguite solo se var_check_al è pari a 0
;
    ldi    var_check_al,T_CHECK      ; per prima cosa inizializza nuovamente
                                      ; var_check_al
;
; Richiamo della subroutine measure_al che campiona la tensione e visualizza il
; risultato (la call memorizza nello stack i due byte dell'indirizzo di rientro)
;
    call  measure_al
;

flag0:
    cpi    flag,0
    breq  led_al_spento
flag2:
    cpi    flag,2
    breq  led_al_acceso
flag1:
    cpi    flag,1
    breq  lampeggiamento          ; Richiamo della subroutine che fa lampeggiare il LED se
                                      ; 4.2V <= Val < 4.6 V
    jmp   end_loop
;
led_al_acceso:                  ; Accendiamo il LED se Val < 4.2 V
    ldi  mp,0b0000_0001
    out  PORTB, mp
    jmp  flag1
;
led_al_spento:                  ; Spegniamo il LED se Val >= 4.6 V
    ldi  mp,0b0000_0000
    out  PORTB, mp
    jmp  flag2

```

```

;
; lampeggiamento:
;
;     cpi    var_led,0
;
; ; Salto condizionato a end_loop se var_led è diverso da 0 (cioè se non sono
; ; passati 0,5s)
;
;     brne  end_loop
;
; Reinizializzazione della variabile di temporizzazione (0.5s)
;     ldi    var_led,T_lamp
;
; Legge il valore di PORTB
;     in    mp, PORTB
;
; Prepara il valore di mp1 per ottenere la complementazione del bit 0 di PORTB
;     ldi    mp1,0b0000_0001
;
; Complementa PB0 (cambia stato al led)
; Esegue or esclusivo tra mp ed mp1 e mette il risultato in mp;
;     eor    mp,mp1
;
; Complementa la linea PB0
;     out   PORTB,mp
;
;
; end_loop:                                ; label che indica la fine del loop
;                                         ; principale
;
;     nop                                ; istruzione di comodo aggiunta per il
;                                         ; debugging (nop = no operation)
;
;     rjmp   main_loop                  ; ritorno a main_loop
;
; Segue la subroutine che legge l'uscita dell'ADC e mostra il valore letto su 8 bit
; sulle tre lampade a sette segmenti
; Riceve come parametri passati dal programma chiamante nello stack i byte basso ed
; alto dell'indirizzo iniziale della LUT preceduti
; dall'indirizzo di rientro (2 byte).
;
;
;
;
measure_lux:
;
;     pop    mp          ; estraе temporaneamente dallo stack l'ultimo byte dell'indirizzo
;                         ; di rientro messo nello stack dalla call
;
;     pop    mp1         ; estrarre temporaneamente dallo stack il primo byte
dell'indirizzo di rientro messo nello stack dalla call
;     pop    ZL          ; recupera il byte basso dell'indirizzo della tabella che è
;                         ; stato passato dal programma chiamante nello stack
;
;     pop    ZH          ; recupera il byte alto dell'indirizzo della tabella che è stato
;                         ; passato dal programma chiamante nello stack
;
;     push  mp1         ; ripristina nello stack il primo byte dell'indirizzo di rientro
;                         ; messo nello stack dalla call
;
;     push  mp          ; ripristina nello stack l'ultimo byte dell'indirizzo di rientro
;                         ; messo nello stack dalla call
;
; Programma l'ADC perché lavori con riferimento interno pari alla tensione di
; alimentazione (AVcc), giustifichi a sinistra il risultato e senta l'input su PC0 (ADC0) -
; quindi la tensione analogica andrà collegata su PC0
;
;     ldi    mp,0b0110_0000
;     sts    ADMUX,mp
;

```

```

;
; Inizia adesso la conversione portando a 1 il bit 6 di ADCSRA (ADSC) senza modificare
; null'altro in ADCSRA
;
    lds    mp,ADCSRA          ; legge il valore del registro dell'AD in mp
    ldi    mp1,0b0100_0000    ; prepara la maschera di programmazione
    or     mp,mp1              ; porta a 1 in mp il bit 6
    sts    ADCSRA,mp          ; scrive il valore modificato mediante l'or in ADCSRA
;
; Aspetta che la conversione sia pronta testando ADSC in ADCSRA: a conversione terminata
; ADSC torna a 0
;
check_conv:
    lds    mp,ADCSRA
    ldi    mp1,0b0100_0000
    and    mp,mp1              ; se il bit 6 di ADCSRA torna a 0 la conversione è
                                ; terminata
    brne   check_conv
;
;
; Legge il valore convertito su 8 bit (siccome è stata scelta la giustificazione a sinistra
; gli 8 bit più significativi del risultato sono in ADCH)
;
;
nop          ; istruzione di comodo per il debug
lds    mp, ADCH           ; legge il valore di ADCH (uscita ad 8 bit del
                          ; convertitore A/D giustificata a sinistra)
ldi    mp1,4               ; numero di cifre dalle quali è costituita una riga (prima
                          ; : cifra indice, le altre i valori)
mul   mp,mp1              ; ottiene l'indice del gruppo di numeri della tabella
                          ; (indicatore della "riga")
;
; ATTENZIONE: Il risultato dell'operazione mul è su 16 bit: il byte basso in R0 ed il byte
; alto in R1
;
;
add   ZL,R0              ; Punta alla cella della tabella che corrisponde al valore letto
                          ; dall'ADC comando all'inizio della
                          ; tabella lo spostamento (attenzione, somma su 16 bit fatta come
                          ; due somme ad 8 bit delle quali la seconda con riporto
adc   ZH,R1
;
; Prepara la visualizzazione su display
;
lpm   mp, Z+              ; legge primo valore della riga della tabella che contiene
                          ; il valore completo intero su 8 bit ed incrementa Z (cioè
                          ; leggo l'indice che non uso)
;
lpm   mp, Z+              ; legge il secondo valore della riga della tabella che
                          ; contiene la cifra delle centinaia (BCD, nibble basso) ed
                          ; incrementa Z
ori   mp,0b0111_0000      ; lascia a uno PD4 - PD6 e non modifica PD0 - PD3
out   PORTD, mp            ; scrive in uscita la cifra delle unità
andi  mp, 0b0110_1111     ; prepara PD4 a livello basso (LE unità) senza
                          ; modificare la cifra delle unità
out   PORTD, mp            ; abilita LE unità
nop
nop
nop
nop
ori   mp,0b0111_0000      ; mette a uno PD4 - PD7 (alza LE per fissare il
                          ; valore)

```

```

        out    PORTD, mp
;
;

        lpm    mp, Z+           ; legge il terzo valore della riga della tabella
                                ; che contiene la cifra della prima cifra decimale
                                ; (BCD, nibble basso) ed incrementa Z
;
        ori    mp,0b0111_0000
        out    PORTD, mp
        andi   mp, 0b0101_1111           ; lascia a 1 PD4 - PD7 e non modifica PD0 - PD3
                                ; scrive in uscita la prima cifra decimale
                                ; prepara PD5 a livello basso (LE prima cifra
                                ; decimale) senza modificare la prima cifra
                                ; decimale
        out    PORTD, mp           ; abilita LE prima cifra decimale
        nop               ; istruzioni di attesa necessarie per garantire
                                ; l'efficacia dello strobe
        nop
        nop
        nop
        nop
        ori    mp,0b0111_0000           ; mette a uno PD4 - PD7 (alza LE per fissare
                                ; il valore)
        out    PORTD, mp
;
        lpm    mp, Z+           ; legge il quarto valore della riga della
                                ; tabella che contiene la seconda cifra
                                ; decimale (BCD, nibble basso) ed incrementa
                                ; Z
;
        ori    mp,0b0111_0000
        out    PORTD, mp
        andi   mp, 0b0011_1111           ; lascia a 1 PD4 - PD7 e non modifica PD0 -
                                ; PD3
                                ; scrive in uscita la seconda cifra decimale
                                ; prepara PD6 a livello basso (LE seconda
                                ; cifra decimale)
                                ; senza modificare la seconda cifra decimale
                                ; abilita LE seconda cifra decimale
                                ; istruzioni di attesa necessarie per
                                ; garantire l'efficacia dello strobe
        out    PORTD, mp
        nop
        nop
        nop
        nop
        ori    mp,0b0111_0000           ; mette a uno PD4 - PD7 (alza LE per fissare
                                ; il valore)
        out    PORTD, mp
;
;
;
; Scrittura display terminata
;
;
        ret                     ; ritorna al programma chiamante
;
;
measure_al:
; Programma l'ADC perché lavori con riferimento interno pari alla tensione di
; alimentazione (AVcc), giustifichi a sinistra il risultato e senta l'input su PC1 (ADC1) -
; quindi la tensione analogica andrà collegata su PC1
;
        ldi    mp,0b0110_0001
        sts    ADMUX,mp
;
; Inizia adesso la conversione portando a 1 il bit 6 di ADCSRA (ADSC) senza modificare
; null'altro in ADCSRA
;
        lds    mp,ADCSRA          ; legge il valore del registro dell'AD in mp

```

```

    ldi    mp1,0b0100_0000      ; prepara la maschera di programmazione
    or     mp,mp1                ; porta a 1 in mp il bit 6
    sts    ADCSRA,mp            ; scrive il valore modificato mediante l'or in ADCSRA
;
; Aspetta che la conversione sia pronta testando ADSC in ADCSRA: a conversione terminata
; ADSC torna a 0
;
check_conv_al:
    lds    mp,ADCSRA
    ldi    mp1,0b0100_0000
    and    mp,mp1                ; se il bit 6 di ADCSRA torna a 0 la conversione è
                                    ; terminata
    brne   check_conv_al
;
;
; Legge il valore convertito su 8 bit (siccome è stata scelta la giustificazione a sinistra
; gli 8 bit più significativi del risultato sono in ADCH)
;
    nop                  ; istruzione di comodo per il debug
    lds    mp, ADCH          ; legge il valore di ADCH (uscita ad 8 bit del
                            ; convertitore A/D giustificata a sinistra)
;
;
confronto_al:
    cpi mp,vthL
    brsh confronto_a12
    ldi flag,2
;
confronto_a12:
    cpi mp,vthH
    brlo confronto_a13
    ldi flag,0
;
confronto_a13:
    cpi mp,vthH
    brsh end_confronto_led
    cpi mp,vthL
    brlo end_confronto_led
    ldi flag,1
;
end_confronto_led:
;
    ret
;
;
; INTERRUPT HANDLERS FOLLOW
;
;
EXT_INT0:
    reti                      ; vector 2:           IRQ0 Handler
;
EXT_INT1:
    reti                      ; vector 3:           IRQ1 Handler
;
PCINTR0:
    reti                      ; vector 4:           PCINT0 Handler
;
PCINTR1:
    reti                      ; vector 5:           PCINT1 Handler
;
PCINTR2:
    reti                      ; vector 6:           PCINT2 Handler
;
WDT:
    reti                      ; vector 7:           Watchdog timer handler

```

```

;
TIM2_COMPA:
    reti                                ; vector 8:      Timer2 compare A handler
;
TIM2_COMPB:
    reti                                ; vector 9:      Timer2 compare B handler
;
TIM2_OVF:
    reti                                ; vector 10:     Timer2 Overflow Handler
;
TIM1_CAPT:
    reti                                ; vector 11:     Timer1 Capture Handler
;
TIM1_COMPA:
    reti                                ; vector 12:     Timer1 CompareA Handler
;
TIM1_COMPB:
    reti                                ; vector 13:     Timer1 CompareB Handler
;
TIM1_OVF:
    reti                                ; vector 14:     Timer1 Overflow Handler
;
TIM0_COMPA:
    reti                                ; vector 15:     Timer 0 CompareA handler
;
TIM0_COMPB:
    reti                                ; vector 16:     Timer 0 CompareB handler
;
;
TIM0_OVF:
;
;      salva mp nello stack prima di utilizzarlo
    push   mp
;      salva SREG nello stack
    in    mp,SREG
    push   mp
;  inizializza nuovamente la variabile di conteggio del timer counter (10,24 ms)
    ldi   mp,246
    out   TCNT0,mp
;  decrementa le variabili di conteggio della temporizzazione dell'ADC
    dec  var_check
    dec  var_check_al
    dec  var_led
;  ripristina SREG
    pop   mp
    out   SREG,mp
;  ripristina mp
    pop   mp
;
    reti                                ; vector 17:     Timer0 Overflow Handler
;
;
;
SPI_STC:
    reti                                ; vector 18:     SPI Transfer Complete Handler
;
USART_RXC:
    reti                                ; vector 19:     USART RX Complete Handler
;
USART_UDRE:
    reti                                ; vector 20:     USART UDR Empty Handler
;
USART_TXC:
    reti                                ; vector 21:     USART TX Complete Handler

```

```
; ADC_conv:  
    reti ; vector 22: ADC Conversion Complete Handler  
;  
EE_RDY:  
    reti ; vector 23: EEPROM Ready Handler  
;  
ANA_COMP:  
    reti ; vector 24: Analog Comparator Handler  
;  
TWSI:  
    reti ; vector 25: Two-wire Serial Interface Handler  
;  
SPM_RDY:  
    reti ; vector 26: Store Program Memory Ready Handler  
;
```