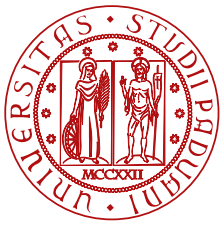


UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA ELETTRONICA

Realizzazione di una Source Measurement Unit (SMU) per la caratterizzazione delle celle solari

Relatore
Prof. Meneghini Matteo

Laureando
Semenzato Nicolò

Correlatore
Dott. Piva Francesco

ANNO ACCADEMICO 2024-2025

Data di laurea 12/11/2024

Sommario

L'attività di tirocinio descritta in questo elaborato riguarda la progettazione e realizzazione di una Source Measurement Unit (SMU) per la caratterizzazione di una cella solare. L'obiettivo principale è stato lo sviluppo di un sistema in grado di fornire corrente e tensione controllate alla cella e di misurare i suoi parametri in risposta alla luce. Il cuore del progetto è un PCB progettato appositamente per gestire l'accensione di un LED che illumina la cella e per raccogliere i dati di misura, corrente e tensione, ai capi della cella. La SMU è gestita da un microcontrollore Arduino DUE, che ne controlla il funzionamento e l'acquisizione dei dati. Scopo ultimo è stato quello di creare uno strumento dai costi contenuti e di facile realizzazione utile per fini didattici.

Indice

Elenco delle figure

Capitolo 1

Introduzione

1.1 Obiettivi del tirocinio

L'obiettivo del lavoro svolto durante questa attività di tirocinio è stato la progettazione di un sistema, che andrà realizzato su PCB, in grado di effettuare misure di corrente e tensione su di una cella solare, un componente elettronico che costituisce l'elemento costitutivo dei pannelli fotovoltaici.

La strumentazione elettronica necessaria per effettuare queste misure può rivelarsi complicata da utilizzare e dai costi considerevoli, l'idea dietro questo progetto è stata quello di realizzare una Source Measurement Unit (SMU) dai costi contenuti, agevolmente replicabile e di facile programmazione, utilizzando pochi componenti e affidando il controllo del sistema a una scheda Arduino, una delle piattaforme più usate per la prototipazione elettronica per la sua semplicità e praticità d'uso.

Tutto queste caratteristiche sono ideali per realizzare una scheda a scopo didattico che può fornire un'introduzione a :

- Design e progettazione di semplici circuiti
- Realizzazione di Printed Circuit Board (PCB)
- Utilizzo e programmazione di un microcontrollore (Arduino DUE)
- Studio delle caratteristiche elettriche di un qualsiasi diodo, LED, cella solare
- Funzionamento di una cella solare

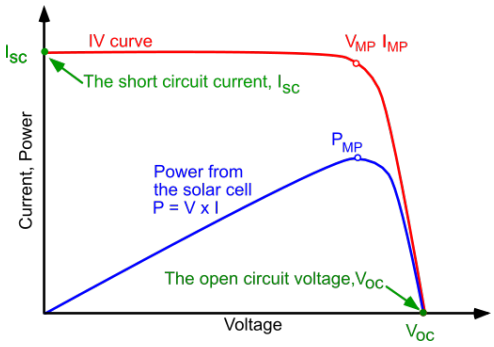


Figura 1.1: Caratteristica IV [1]

1.2 Cenni sulla caratterizzazione di una cella solare

La cella solare è un dispositivo a semiconduttore che converte l'energia della luce solare (o artificiale) incidente in elettricità tramite l'effetto fotovoltaico. Rappresenta l'elemento costitutivo dei pannelli fotovoltaici.

I fotoni incidenti generano coppie elettrone-lacuna sullo strato di semiconduttore, drogato in modo da costituire una giunzione *p*-*n*. La ricombinazione di tali coppie è impedita dal campo elettrico che esiste nella regione di svuotamento tra gli strati *p* e *n*.

Se l'emettitore e la base della cella solare sono collegati insieme (cioè, se la cella solare è in cortocircuito), gli elettroni generati dalla luce fluiscono attraverso il circuito esterno, costituendo così una corrente elettrica.

Attraverso il circuito che verrà presentato nel capitolo successivo, questa Source Measuring Unit si prefigge il compito di misurare la tensione ai capi della cella e la corrente che l'attraversa, variando allo stesso tempo l'intensità luminosa che la colpisce, regolando il LED che illumina la cella. Da questi dati è possibile ricavare alcuni dei parametri principali che caraterizzano una cella:

- **Corrente di corto circuito** I_{SC}: la corrente che scorre attraverso la cella quando la tensione ai suoi capi è nulla (cioè è corto circuitata). In condizioni ideali la corrente di corto circuito indica anche la corrente massima che può essere prelevata dalla cella.

- **Tensione a vuoto** V_{OC}: La tensione a vuoto è la tensione massima ottenibile dalla cella solare quando non è collegato alcun carico, ovvero quando la corrente generata è nulla.

- **Curva IV**: La caratteristica corrente-tensione permette di analizzare il comportamento complessivo della cella (Figura 1.1). I punti rilevanti della curva sono la corrente di corto circuito e la tensione a vuoto. Nella figura è possibile vedere anche il punto a cui una cella tipicamente dovrebbe operare per dare la massima potenza in uscita.

Capitolo 2

Componenti utilizzati

2.1 Arduino DUE

Arduino DUE è una scheda basata sul microcontrollore ARM Cortex-M3, l'Atmel SAM3X8E, che offre una serie di caratteristiche avanzate, le principali sono:

- Architettura a 32 bit
- Clock a 84 Mhz
- Memoria flash 512 KB
- SRAM 96 KB
- Tensione di funzionamento 3.3 V
- Tensione di ingresso 7-12 V (Limiti: 6-20 V)
- 54 pin digitali I/O (di cui 16 con PWM)
- 12 pin analogici di input (ADC a 12 bit)
- 2 pin analogici di output (DAC a 12 bit)
- corrente massima per tutti gli I/O: 130 mA
- corrente massima dai pin a 3.3 V e 5 V: 800 mA
- supporto per i protocolli di comunicazione I2C, SPI

Per la sua semplicità, versatilità e costo contenuto è una piattaforma molto utilizzata in ambito educativo e sperimentale e per queste caratteristiche è stata scelta per il progetto ivi descritto. Presenta tuttavia alcune limitazioni:

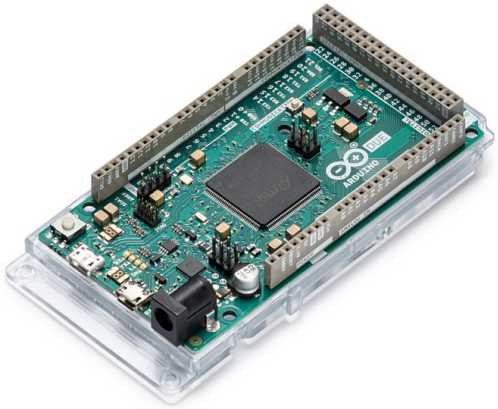


Figura 2.1: Scheda Arduino DUE

- Tensione 3.3V, poco stabile quando è richiesta molta corrente
- Limiti di corrente erogabile dalla scheda
- La tensione minima di uscita del DAC è 0.55 V

Perciò si è ritenuto opportuno utilizzare un PCB esterno con dei componenti aggiuntivi, invece di una semplice shield, in modo da avere prestazioni migliori e maggiore risoluzione per le misure.

2.1.1 SPI

Il **Serial Peripheral Interface (SPI)** è un protocollo di comunicazione seriale sincrona, cioè basata sulla sincronizzazione del clock del trasmettitore e del ricevitore e sulla comunicazione simultanea (full-duplex ovvero su due fili). Tipicamente è costituito da quattro linee di comunicazione:

- **MISO** (Master In Slave Out): Dati in ingresso per il master, uscita delle periferiche slave
- **MOSI** (Master Out Slave In): Dati in uscita per il master, ingresso per lo slave
- **SCLK** (Serial Clock): Clock generato dal master per sincronizzare la comunicazione
- **CS** (Chip Select): Selezione dello slave, attivo basso per indicare quale slave deve comunicare con il master

Arduino DUE supporta questo protocollo tramite degli appositi pin dedicati per i segnali di SCK, MOSI, MISO, mentre i CS possono essere associati a dei pin digitali. Arduino possiede una libreria SPI integrata, con cui è possibile configurare e gestire le comunicazioni SPI tramite apposite funzioni. [2] I parametri configurabili sono:

- Frequenza di clock: programmabile fino a 42 MHz (ovvero metà del clock del microcontrollore)
- Shifting dei bit: l'ordine con cui vengono inviati i bit, a seconda della periferica può dover essere inviato prima il bit più significativo (MSB) oppure quello meno significativo (LSB)
- Modalità SPI: supporta quattro modalità a seconda del tipo di campionamento e polarità del clock:
  - Modalità 0: Clock idle a basso livello, dati campionati al fronte di salita
  - Modalità 1: Clock idle a basso livello, dati campionati al fronte di discesa
  - Modalità 2: Clock idle ad alto livello, dati campionati al fronte di discesa
  - Modalità 3: Clock idle ad alto livello, dati campionati al fronte di salita

Questo protocollo è stato utilizzato per gestire le comunicazioni tra il microcontrollore e i componenti aggiuntivi esterni, introdotti nei paragrafi successivi.

2.2 DAC AD5752

E' un convertitore digitale-analogico con le seguenti caratteristiche principali:

- 2 canali con risoluzione programmabile dai 12 fino ai 16 bit
- Frequenza di campionamento 1MS/s
- Alimentazione analogica sia singola che duale, da $\pm$4.5 V fino a $\pm$16.5 V
- Tempo di assestamento 10 μs
- Total unadjusted error (TUE): massimo 0.1% FSR
- Alimentazione digitale minima 2.7 V massima 5.5 V
- Interfaccia seriale compatibile con SPI che può operare con clock fino a 30 MHz
- Tipo di montaggio SMD

Il DAC supporta la comunicazione seriale tramite SPI, perfettamente compatibile con Arduino, attraverso il pin SYNC usato come Chip Select.

Il dispositivo ha altri due pin in ingresso per avere ulteriori funzionalità: LDAC che permette di aggiornare simultaneamente i due canali e CLR per fare un reset asincrono del DAC e azzerare le sue uscite.

Si è optato per scelta di questo integrato per in modo da avere una risoluzione maggiore, 16 bit contro i 12 del DAC già presente in Arduino, e per ovviare alla limitazione nel range di tensione di quest'ultimo, Il DAC scelto raggiunge gli 0V e ciò permette di sfruttare pienamente tutto il range di tensione 0-5V. [3]

2.3 ADC AD7734

E' un convertitore analogico digitale di tipo Sigma-Delta con le seguenti caratteristiche principali:

- 4 canali con risoluzione programmabile a 16 bit o 24 bit
- Frequenza di campionamento 15.4kS/s
- Alimentazione analogica 5V
- Alimentazione digitale da 2.7-3.6 V, 4.75V-5.25V
- Tensione di input 5V/10V, $\pm$5V/$\pm$10V
- Total unadjusted error (TUE): massimo 0.1% FSR
- INL - Integral Nonlinearity: +/- 0.006 % FSR
- Interfaccia seriale compatibile con SPI con freq di clock dino a 10 MHz
- Tipo di montaggio SMD

Questo ADC funziona a una frequenza di clock di 6.144MHz, il clock è dato tramite apposito oscillatore esterno connesso all'integrato. Possiede diverse modalità di funzionamento come la possibilità di funzionare tramite conversione continua o singola. Il tempo di conversione dell'ADC dipende dalle impostazioni avanzate, quali ad esempio il *chopping*, tecinca che permette di filtrare le acquisizioni per ridurre il rumore e avere misurazioni più accurate. Da datasheet comunque il tempo di conversione di un dato non eccede i 27μs.

Oltre al pin CS per la comunicazione SPI, possono essere collegati al microcontrollore altri due pin: un pin di input RESET per effettuare un rest dell'integrato, e il pin output RDY, che

può essere letto per verificare che il convertitore abbia finito la conversione del dato sul registro selezionato.

Si è optato per questo dispositivo per avere una risoluzione maggiore per le misure di tensione, infatti l'ADC interno ad Arduino ha una risoluzione di 12 bit contro i 24 dell'AD7734. [4]

2.4 Cella solare

La cella solare su cui vengono effettuate le misure è l' ANYSOLAR KXOB25-04X3F-TB[5]. Presenta le seguenti caratteristiche elettriche (valori tipici, da datasheet):

- Tensione a vuoto: 2.07 V
- Corrente di corto circuito: 14 mA
- Tensione al punto di massima potenza: 1.67 V
- Corrente al punto di massima potenza: 13.2 mA
- Potenza massima: 22 mW
- Efficienza della cella η: 25%

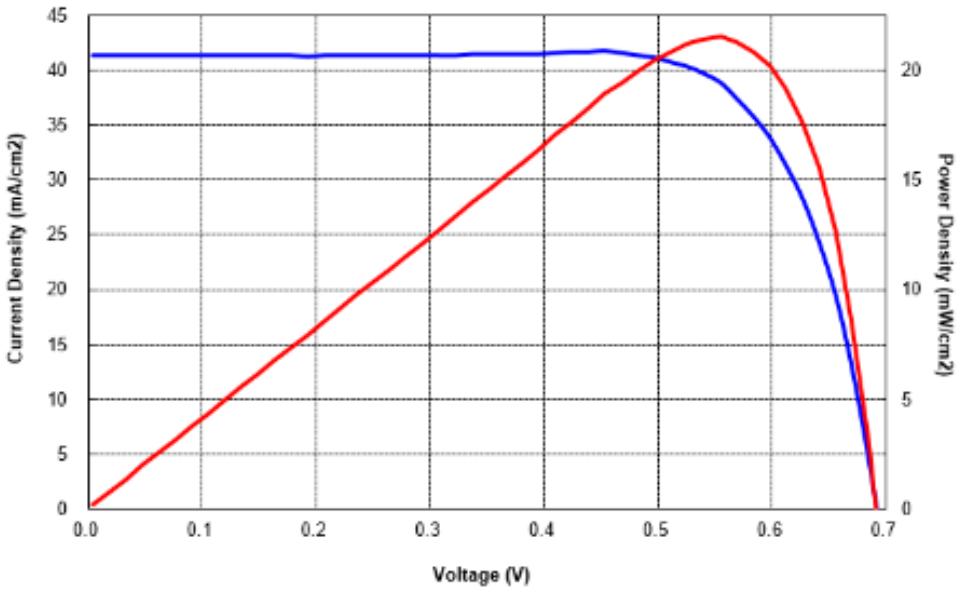
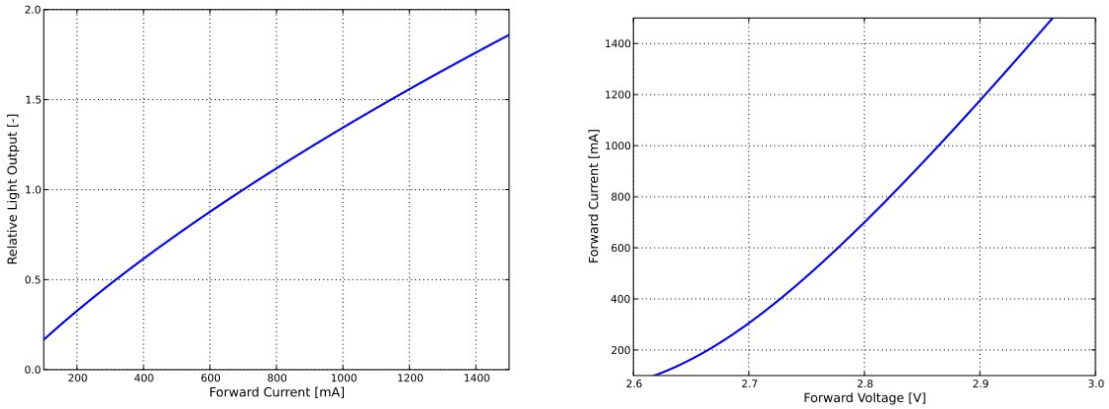


Figura 2.2: Caratteristiche corrente-tensione della cella ANYSOLAR

2.5 Altri componenti

- Relè a stato solido AQY211EH[6]:
  - Photo MOSFET
  - Corrente di carico fino a 1 A
  - Tensione di carico fino a 30 V
  - Alto isolamento: fino a 5000 Vrms
  - T_{on} tipico 1.5 ms, max 5 ms
  - T_{off} tipico 0.1ms, max 1 ms
- Transistor BJT TIP41C [7]
  - Transistor npn di potenza
  - Corrente massima di collettore 6 A
  - Tensione massima collettore-emettitore 100 V
  - Guadagno di corrente base-collettore minimo 15, massimo 75
- LED Luxeon TX [8]
  - Tensione diretta tipica 2.8 V
  - Corrente massima fino a 1.2 A
- 2x Amplificatori operazionali TL082
- Resistenze, condensatori e componentistica varia



(a) Curva Tensione diretta-luce emessa (b) Curva Tensione diretta-corrente

Figura 2.3: Curve di funzionamento del LED

Capitolo 3

Progettazione della SMU

3.1 Schemi del circuito

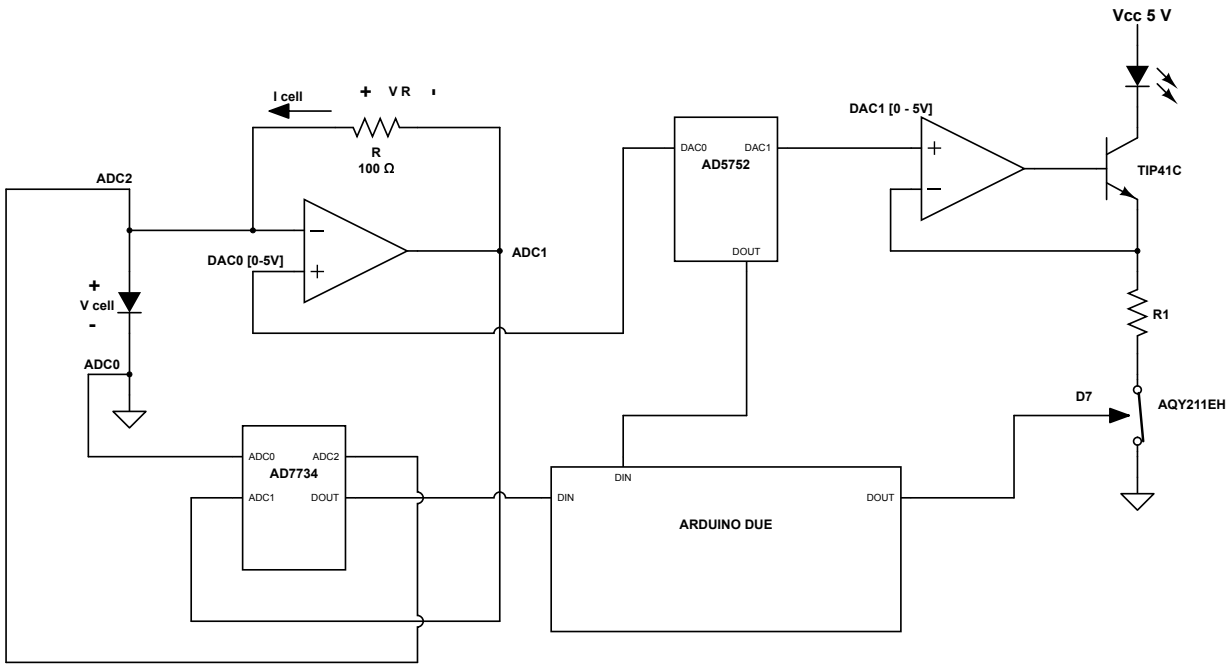


Figura 3.1: Schema a blocchi generale del circuito

Nella figura precedente (Figura 3.1) è mostrato una schema a blocchi (semplificato) del circuito realizzato nel PCB. Esso può essere riassunto in due componenti analogiche principali:

- La prima è costituita dal circuito che effettivamente realizza la SMU ed effettua le misure sulla cella solare.
- La seconda, invece, dal circuito che svolge la funzione di LED Driver, regolando la luminosità dei LED che illuminano la cella.

La parte digitale è invece costituita dagli integrati (ADC, DAC, relè a stato solido) collegati alla scheda Arduino tramite i pin digitali di input/output e i pin dedicati all'SPI, che sono stati omessi per semplicità.

3.1.1 SMU

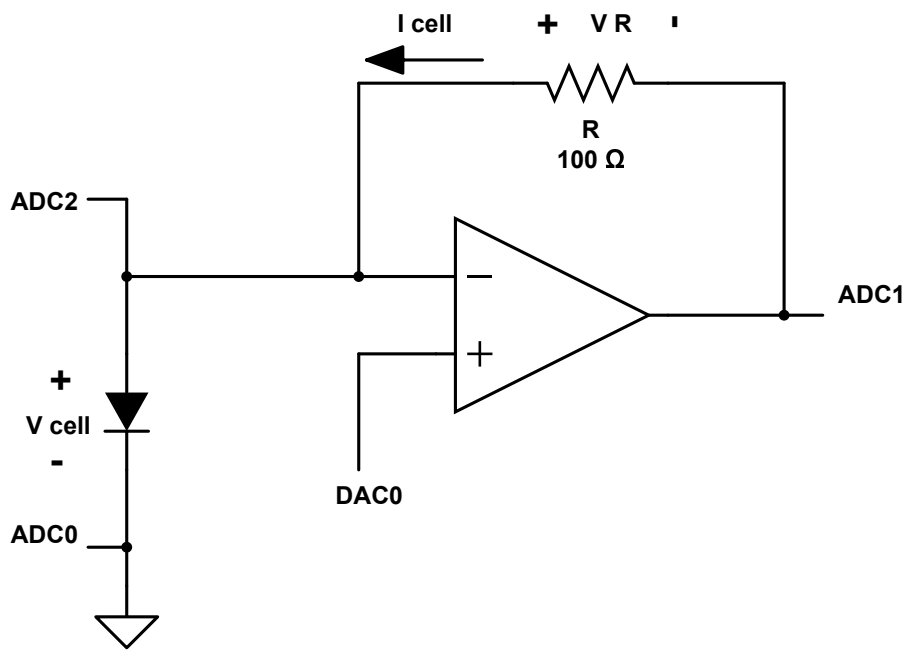


Figura 3.2: Circuito SMU

In questa porzione di circuito abbiamo un amplificatore operazionale (op-amp) con una resistenza connessa tra l'uscita e il morsetto invertente, al morsetto non invertente è invece collegato il DAC che può fornire una tensione che va dai 0 ai 5 V.

La cella solare è approssimata come un diodo, il cui catodo è collegato al morsetto non invertente dell'op-amp mentre l'anodo a massa.

L'op-amp impone una tensione di uscita nel tentativo di mantenere la tensione sul morsetto invertente uguale a quello non invertente, imponendo così una corrente che scorre attraverso la resistenza e il diodo (nel caso in trattazione la cella solare).

La tensione sulla cella è misurata dai due ingressi analogici ADC0 e ADC2:

$$V_{cell} = V_{ADC2} - V_{ADC0} \tag{3.1}$$

La corrente sulla cella è invece calcolata come tensione sulla resistenza R conosciuta :

$$V_R = V_{ADC2} - V_{ADC1} \tag{3.2}$$

$$I_{cell} = \frac{V_{ADC2} - V_{ADC1}}{R} \tag{3.3}$$

3.1.2 LED Driver

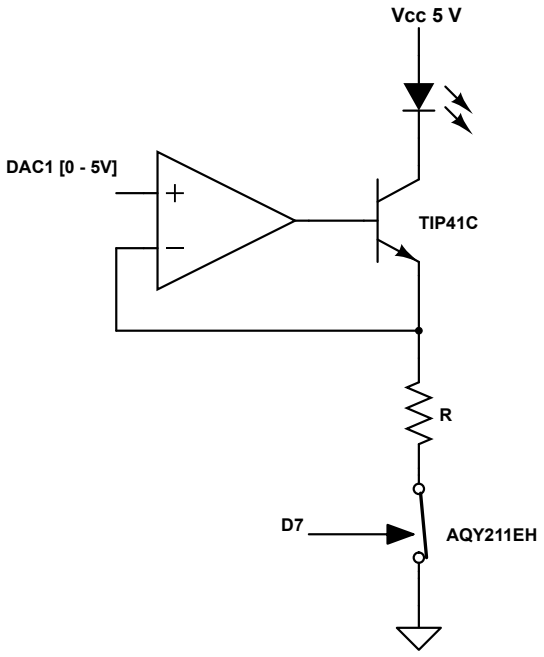
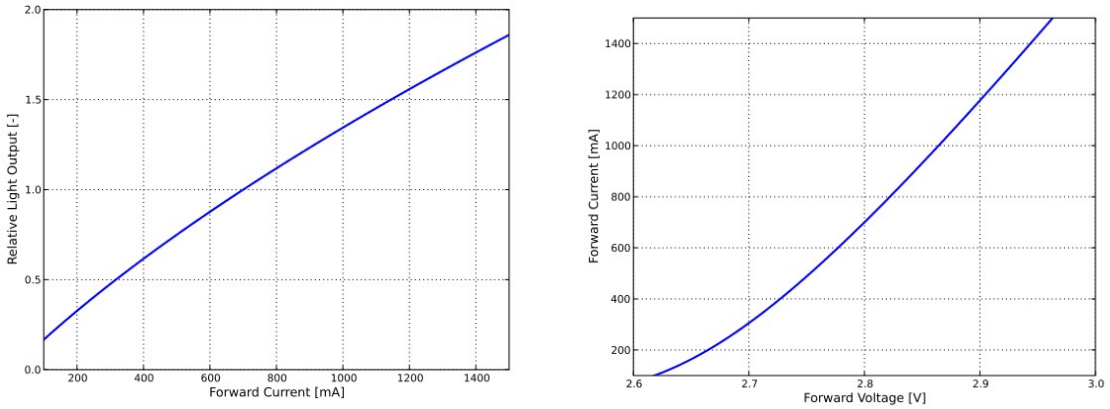


Figura 3.3: Circuito LED driver

In questa porzione di circuito viene controllata la luminosità del LED che illumina la cella. Il DAC fornisce la tensione al morsetto non invertente dell'op-amp, che all'uscita fornisce una

tensione alla base del BJT per eguagliare l'ingresso invertente, collegato all'emettitore del transistor, a quello non invertente, accedendo così il BJT, il quale opera come amplificatore di corrente, permettendo così di regolare in modo lineare la corrente che scorre attraverso il LED e di conseguenza la luce che viene emessa.

La tensione applicata dal DAC all'ingresso non invertente viene applicata dall'opamp anche ai capi della resistenza R tramite l'ingresso invertente, conoscendo la tensione su R è facilmente ricavabile la corrente che scorre nel LED e, consultando le curve caratteristiche, la luce emessa dal diodo.



(a) Curva Tensione diretta-luce emessa (b) Curva Tensione diretta-corrente

Figura 3.4: Curve di funzionamento del LED

La massima tensione che può fornire il DAC per far funzionare il circuito è fornita dalla seguente equazione:

$$V_{DAC1_{MAX}} = V_{dd} - V_{LED} - V_{CE_{sat}} = 5 - 2.75 - 0.06 = 2.19V \tag{3.4}$$

Dove V_{LED} è la tensione di funzionamento del LED in polarizzazione diretta, mentre $V_{CE_{sat}}$ è la tensione del transistor quando giunge in saturazione (dati ricavati da datasheet).

Il relè a stato solido funge da interruttore, controllato dalla scheda Arduino tramite un apposito pin digitale.

3.2 PCB design

3.2.1 Progettazione elettronica

Una volta delineato il circuito finale e scelti i componenti, è stato il momento di progettare il PCB sul quale realizzarlo, tramite l'utilizzo di un apposito software di progettazione elettronica, ovvero *Autodesk Fusion 360*.

I passaggi principali di quest'attività sono stati innanzitutto la creazione di un nuovo progetto, successivamente si è partiti dallo*Schematic Design*, ovvero il disegno dello schema elettrico del circuito.

Una volta creato lo schematico del progetto si possono posizionare tutti i componenti necessari (integrati, resistenze, condensatori, morsettiere etc) ed effettuare i collegamenti elettrici. Ogni componente può essere posizionato tramite un'apposita libreria che contiene le informazioni necessarie, ad esempio layout dei pin e dimensioni del componente.

Nel caso le librerie non fossero presenti già in Fusion di default, queste sono facilmente reperibili in rete, quindi è bastato scaricarle e importarle su Fusion in una libreria personalizzata. In questo modo è possibile avere la footprint del dispositivo sia nello schematico che sulla sezione dedicata al PCB. Completato lo schematico ed essersi assicurati che le connessioni siano corrette, la fase successiva è stata quella che riguarda il *PCB Layout* , nel quale si passa a disegnare il layout di quello che diventerà il PCB fisico.

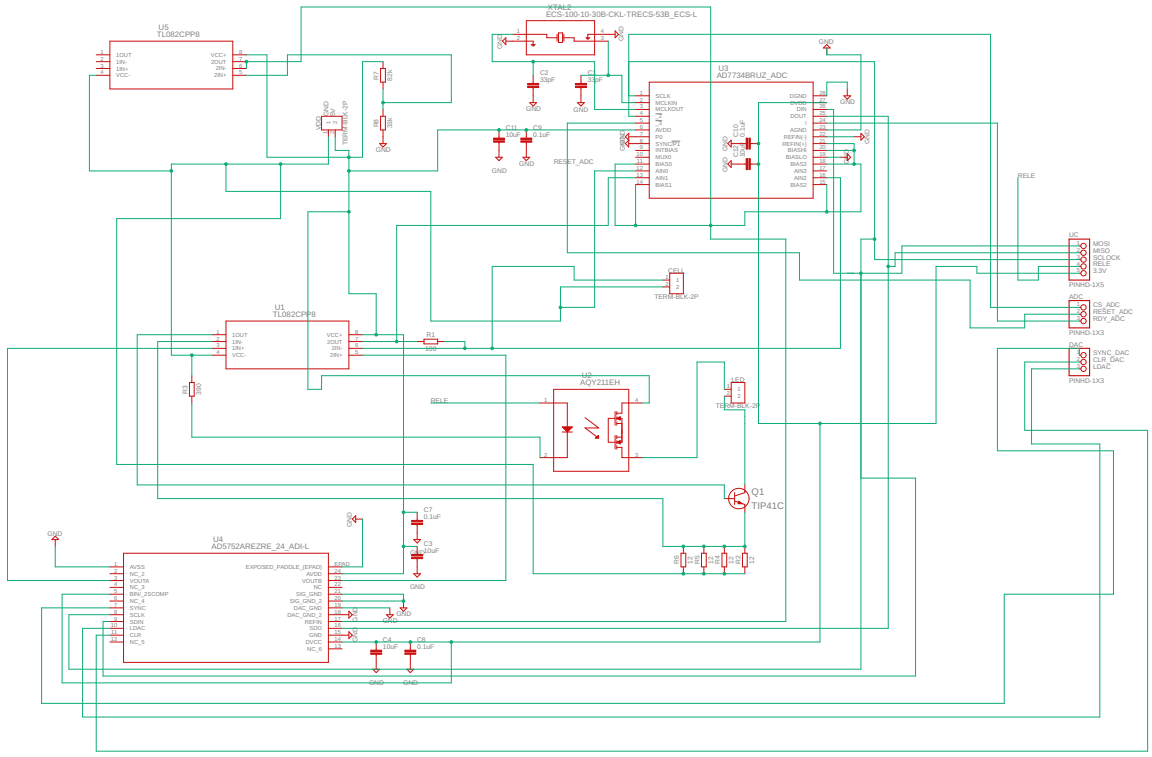
Le fasi seguenti sono stati il delineamento delle dimensioni della scheda, la creazione dei poligoni (ovvero i piani di massa) sia sulla parte superiore (Topside) che sulla parte inferiore (Bottomside), e il posizionamento di tutti i componenti necessari sulla scheda.

I componenti sono stati posizionati sulla scheda in mondo che fosse agevole tracciare le piste per collegarli ma allo stesso cercando di ottimizzare gli spazi, infatti è buona norma cercare di ridurre le dimensioni per contenere il costo del PCB, che cresce in base alle dimensioni.

I collegamenti vengono effettuati tramite un apposita funzione che permette di disegnare le piste, secondo le indicazioni fornite dallo schematico realizzato in precedenza. Le piste devono essere correttamente distanziate e non devono passare in mezzo alle piazzole nelle quali saranno connessi i pin dei componenti, in modo da evitare errori nel processo di stampa della scheda.

La grandezza delle piste e dei pad sul quale poggiano i componenti devono soddisfare i requisiti che permettano alla scheda di essere prodotta in laboratorio, in particolare le piste di 0.5 mm, il diametro dei fori a 40 mils (1.016 mm) e il diametro dei pad a 85 mils (2.159 mm).

Per le connessioni dei componenti con package SMD (Surface Mount Device) si è avuto cura che le piste, ridotte a 0.3048 mm, fossero correttamente allineate ai pad che connettono i pin degli integrati, in modo da evitare difetti di saldatura.



30/10/2024 16:39 f=0.60 C:\Users\nicol\Desktop\Progettazione elettronica2 v4.pdf (Sheet: 1/1)

Figura 3.5: Schematico del PCB completo

3.2.2 Risultato finale

La scheda presenta delle morsettiere a lato per collegare rispettivamente l'alimentazione a 5 V e il riferimento a massa, il PCB che ospita il led e la cella solare su cui effettuare le misurazioni. Dei connettori in alto forniscono invece i pin per collegare la basetta alla scheda Arduino.

In particolare i pin per la comunicazione SPI (Clock, MOSI, MISO, e i Chip Select di DAC e ADC), per fornire la tensione 3.3 V da fornire all'alimentazione digitale degli integrati, e i pin per eventualmente sfruttare le funzionalità aggiuntive degli integrati, i pin LDAC e CLR per l'AD5752 e i pin RDY e RST per l'AD7734. Il PCB qui presente soddisfa le regole di design per essere realizzato in un laboratorio dell'Università, una volta generati i file Gerber con Fusion.

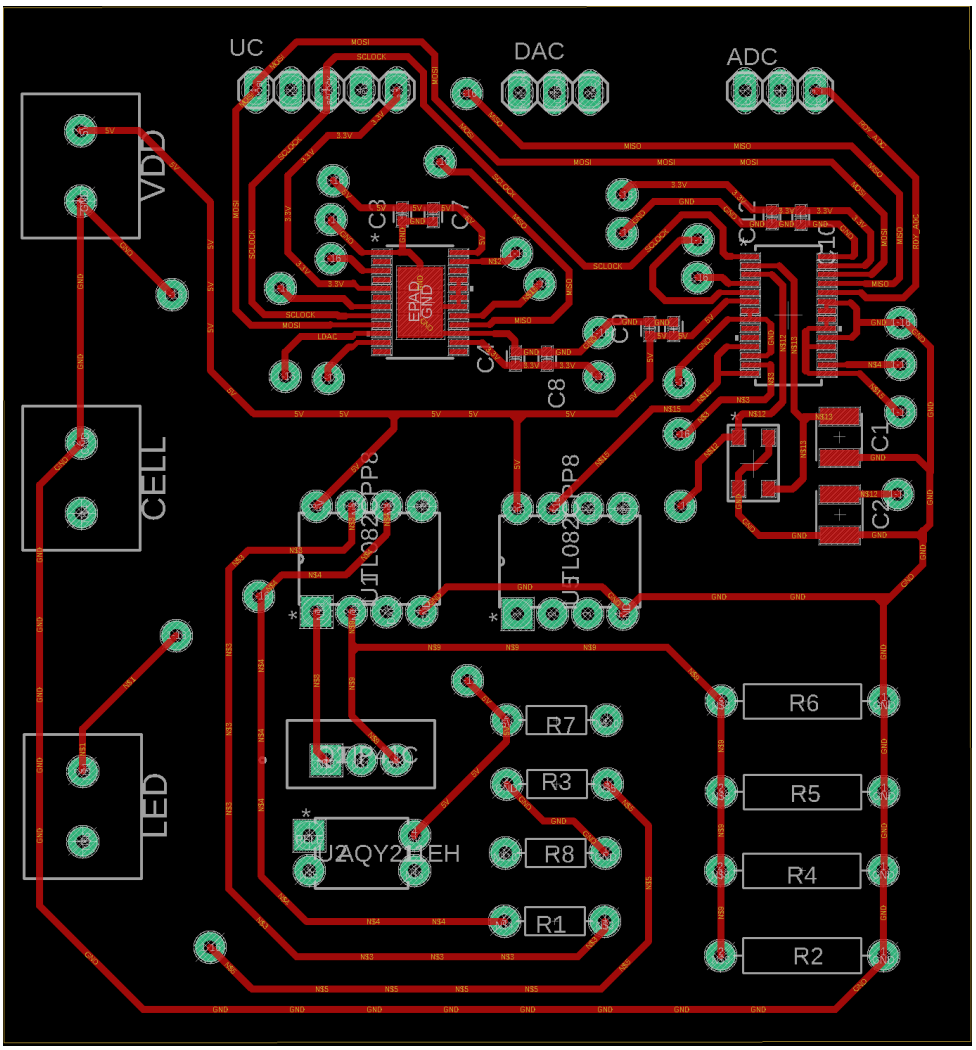


Figura 3.6: Parte superiore del PCB

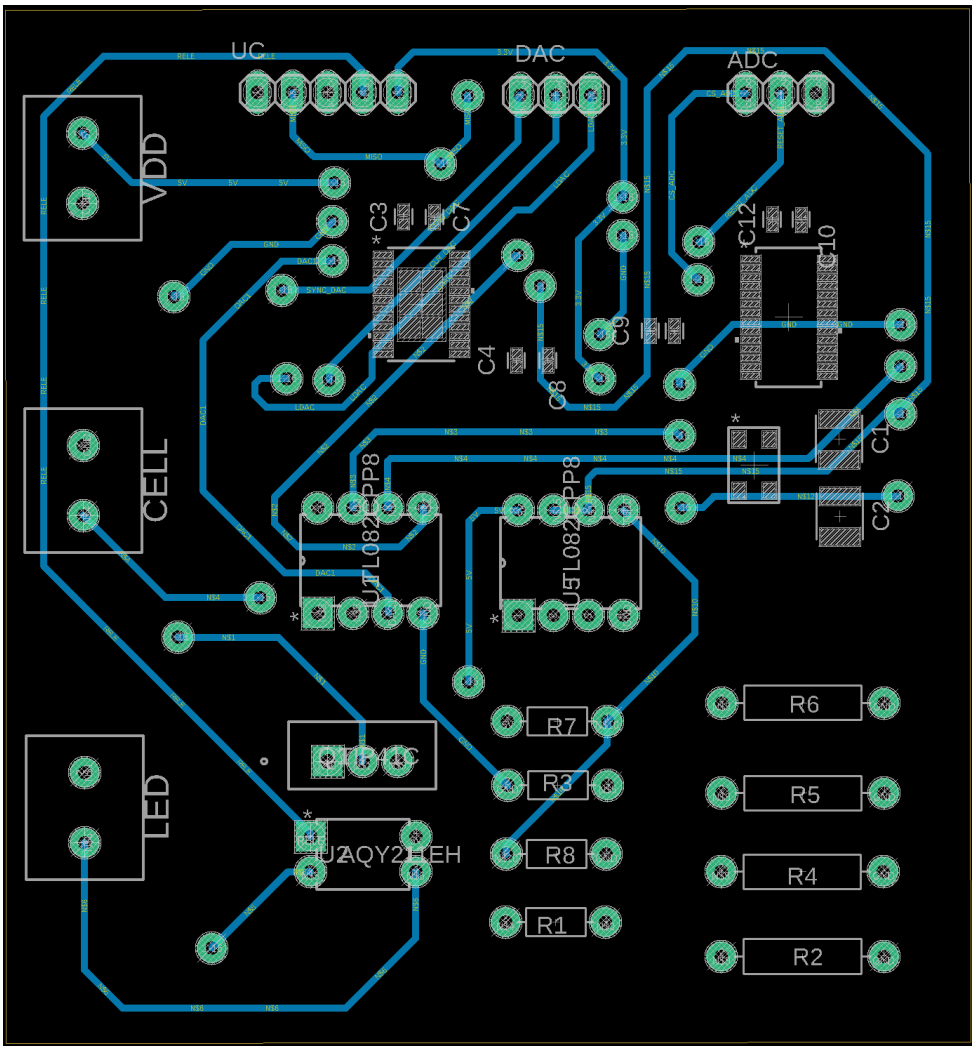


Figura 3.7: Parte inferiore del PCB

Capitolo 4

Scrittura del codice

4.1 Introduzione al software

Il codice per utilizzare la scheda descritta nei capitoli precedenti è stato scritto utlizzando l'IDE di Arduino. Il programma è composto da una funzione di setup che contiene le istruzioni di per inizializzare gli ingressi e le uscite del microcontrollore e poi di un ciclo che esegue in loop le funzioni necessarie a fare le misurazioni. Attraverso il monitor seriale è possibile richiamare le funzioni richieste e di visualizzare i risultati. Per gestire la comunicazioni con gli integrati vengono utilizzate due librerie che contengono le istruzioni per le funzioni di inizializzazione e di lettura/scrittura sul DAC e sull'ADC. Le librerie sono state modificate per gestire correttamente la comunicazione SPI secondo la documentazione di Arduino.

4.1.1 Comunicazione SPI

La comunicazione SPI su Arduino è gestita tramite un'apposita libreria che fornisce tutte le funzioni necessarie. Per comunicare con un dispositivo tramite protocollo SPI i passaggi chiave sono:

- Definire le impostazioni SPI, in base alle caratteristiche del dispositivo, creando un oggetto `SPISettings` con i tre parametri necessari: frequenza del clock, ordine dei bit, e la modalità di trasmissione (2.1.1). Due possibili configurazioni sono le seguenti:
  - DAC: `SPISettings(SPI_CLOCK_DIV64, MSBFIRST, SPI_MODE0)` per avere una frequenza di clock di 656 kHz, bit più significativo per primo e la modalità 0 (clock idle a basso livello, dati campionati al fronte di salita)
  - ADC: `SPISettings(SPI_CLOCK_DIV64, MSBFIRST, SPI_MODE3)` per avere una frequenza di clock di 656 kHz, bit più significativo per primo e la modalità 3 (clock idle ad alto livello, dati campionati al fronte di salita)

Siccome la velocità di comunicazione non è una priorità per l'applicazione presa in esame si è optato per una frequenza di clock "bassa", anche se i dispositivi potrebbero comunicare anche con frequenze di clock più elevate.

Le modalità di comunicazione scelte sono state ricavate dai datasheet, che contengono i diagrammi per capire come i dati vengono inviati e campionati nei bus seriali.

- Chiamare la funzione `SPI.beginTransaction(SPISettings)` per inizializzare la porta SPI con le impostazioni fornite.
- Portare basso il bit CS del dispositivo con cui si intende comunicare, tipicamente con la funzione `digitalWrite(CS, LOW)`.
- Chiamare `SPI.transfer()` quante volte si desidera per trasferire i dati. La funzione `SPI.transfer()` permette di inviare o di ricevere singoli byte oppure array buffer (specificando nome del buffer e dimensione nell'argomento della funzione).
- Portare alto il bit CS (`digitalWrite(CS, HIGH)`) per comunicare al dispositivo slave la fine della trasmissione.
- Chiamare la funzione `SPI.endTransaction()` per terminare di usare il bus SPI, e permettere ad altre librerie di usare la comunicazione seriale.

4.2 Codice utilizzato

Nella sezione di setup del programma per far funzionare la SMU viene inizializzata la comunicazione SPI della scheda Arduino , il pin di output per controllare il relè a stato solido, che fa da interruttore nel circuito del LED Driver, il monitor seriale e, tramite le funzioni apposite delle librerie dedicate, vengono inizializzati il DAC e l'ADC.

In particolare viene impostato il range in cui devono operare, nel nostro caso 0-5 V, i pin Chip Select per la comunicazione seriale, e i pin di input/output che possono essere usati per interagire con i due componenti.

```
void setup() {

 SPI.begin();

 pinMode(7, OUTPUT);  //relé,

 //setup DAC
```

```
  init_DAC(DAC_AB, p5V);  //impostare entrambi i canali da 0 a 5 V
 //setup ADC
  adc.SetupAD7734(CS_ADC, RDY_ADC, RESET_ADC);
  adc.ChannelSetup(ADC_A, P_05); //impostare i canali usati da 0 a 5 V
  adc.ChannelSetup(ADC_B, P_05);
  adc.ChannelSetup(ADC_C, P_05);
  Serial.begin(9600); //monitor seriale
 }
```

Tramite il monitor seriale è possibile richiamare le funzioni necessarie, in particolare principali sono quella che permettono di impostare la luminosià del led, utilizzando la funzione `set_DAC_value(unsigned int DAC_value, char DAC)` dalla libreria dell'AD5752 [9]:

```
 void set_DAC_value(unsigned int DAC_value, char DAC)
{
 SPI.beginTransaction(settingsDAC);
 digitalWrite(DAC_CS_pin, LOW);
 delay(1);
 SPI.transfer(DAC);
 SPI.transfer((DAC_value & 0xFF00) >> 8);
 SPI.transfer(DAC_value & 0x00FF);
 delay(1);
 digitalWrite(DAC_CS_pin, HIGH);
 delay(1);

 digitalWrite(DAC_CS_pin, LOW);
 delay(1);
 SPI.transfer(0x1D);
 SPI.transfer(0x00);
 SPI.transfer(0x00);
 delay(1);
 digitalWrite(DAC_CS_pin, HIGH);
 delay(1);
 SPI.endTransaction();

}
```

Nella funzione viene passato il valore di output desiderato come intero senza segno, e il come char il numero corrispondente al canale che si vuole impostare, per impostare la luminosità del led verrà utilizzato DAC1 che corrisponde al canale B. [9]

Tramite comunicazione seriale vengono inviate le istruzioni necessarie, il DAC AD5752 comunica tramite pacchetti di 16 bit, quindi per ogni comunicazione `SPI.transfer()` viene chiamata tre volte per inviare tre byte, che contengono le informazioni per scrivere nel registro di output il valore desiderato.

Una volta impostata la luminosità del LED, si può chiamare la funzione IV() che imposta il valore del canale A del DAC (DAC0) a valori crescenti, applicando quindi valori diversi di tensione ai capi della cella, per ognuno di questi valori di tensioni differenti vengono effettuate le misurazioni tramite gli input analogici dell'ADC.

Viene fatta una media aritmetica per ridurre gli effetti di eventuale rumore e disturbi e avere risultati più vicini a quelli reali, dopo la media il dato di misurazione viene moltiplicato per il FSR (Full Scale Range) dell'ADC (in questo caso 5 V) e diviso per la risoluzione $(2^{16} - 1)$, ottenendo così il valore di tensione cercato.

I valori ottenuti possono venire stampati poi sul monitor seriale con le tensioni in Volt e la corrente in milliampere.

```
 void IV() { //misura della corrente variando la tensione ai capi della cella
for(int ii=0; ii<65530; ii=ii+320)
{
  int valA0=0; int valA1=0; int valA2=0; int valA3=0;
  set_DAC_value(ii, DAC_A);
  for(int jj=0; jj<medie; jj++) {
    delay(20);
    adc.StartSingleConversion(ADC_A); //inizio conversione su A0 - A1 -A2
    delay(5); //delay per attendere che la conversione finisca
    adc.StartSingleConversion(ADC_B);
    delay(5);
    adc.StartSingleConversion(ADC_C);
    delay(5);
    valA0 += adc.GetConversionData(ADC_A);
    delay(5);
    valA1 += adc.GetConversionData(ADC_B);
    delay(5);
    valA2 += adc.GetConversionData(ADC_C);
    delay(5);
```

```
    }
  float V0 = ((valA0/medie)*5)/65535;
  float V1 = ((valA1/medie)*5)/65535;
  float V2 = ((valA2/medie)*5)/65535;

  float V = V2-V0;
  float I = (V1-V2)/R*1000; //in mA

  Serial.print(V,3); Serial.print(", "); Serial.print(I,2); Serial.print(", ");
  Serial.print(V0,3); Serial.print(", "); Serial.print(V1,3);
  Serial.print(", "); Serial.print(V2,3); Serial.print(", ");
}
Serial.println();
return;
```

Per le misure di tensione sono state utilizzate dalla libreria dell'AD7734 [10] le funzioni `StartSingleConversion()` per comunicare all'ADC di effettuare una singola conversione sul canale desiderato e `getConversionData()` per ottenere il dato convertito.

Per ricevere il dato convertito si può usare un delay per aspettare che l'integrato faccia la conversione oppure controllare il bit RDY che diventa basso una volta che il dato è pronto per essere letto.

In questo codice è stato l'utilizzato l'ADC in modalità conversione singola a 16 bit. Per comunicare con l'ADC si indica con il primo byte se l'operazione è di lettura o scrittura e l'indirizzo del registro scelto. Successivamente si può trasferire, a seconda dei casi, i byte da scrivere nel registro oppure si ricevono i byte in lettura che andranno salvati in una variabile o in un array.

Ad esempio, la funzione per far partire una conversione singola sul canale desiderato è la seguente:

```
  void AD7734::StartSingleConversion(int adc_channel) {
  uint8_t data_array[2];

  //imposta il registro di comunicazione per scrittura  nel mode register
  data_array[0] = WRITE | ADDR_MODE(adc_channel);

  //setup mode register
```

```
  data_array[1] = SINGLE_CONV_MODE;

  //invio delle istruzioni
  SPI.beginTransaction(settingsADC);
  delay(1);
  digitalWrite(_cs, LOW);
  delay(1);
  SPI.transfer ( data_array, 2);
  delay(1);
  digitalWrite(_cs, HIGH);
  SPI.endTransaction();
 }
```

Mentre l'istruzione per ottenere i dati convertiti dai canali dell'ADC è implementata nel seguente metodo

```
  uint16_t AD7734::GetConversionData(int adc_channel) {
  uint8_t data_array, upper, lower;
 //Imposta il communication register per la lettura dei dati del canale.
  data_array = READ | ADDR_CHANNELDATA(adc_channel);

  //scrive nel communication register
  SPI.beginTransaction(settingsADC);
  delay(1);
  digitalWrite(_cs, LOW);
  delay(1);
  SPI.transfer( data_array);

  //Leggi i byte superiore e inferiore del registro dati del canale
  //(modalità 16 bit)
  upper = SPI.transfer(0);
  lower = SPI.transfer(0);
   delay(1);
  digitalWrite(_cs, HIGH);
  SPI.endTransaction();
  uint16_t result = upper << 8 | lower;
  return result;
  }
```

Capitolo 5

Conclusioni

5.1 Riepilogo

In questa attività di tirocinio è stato sviluppato e progettato un circuito stampato per effettuare misure per la caratterizzazione di piccole celle solari.

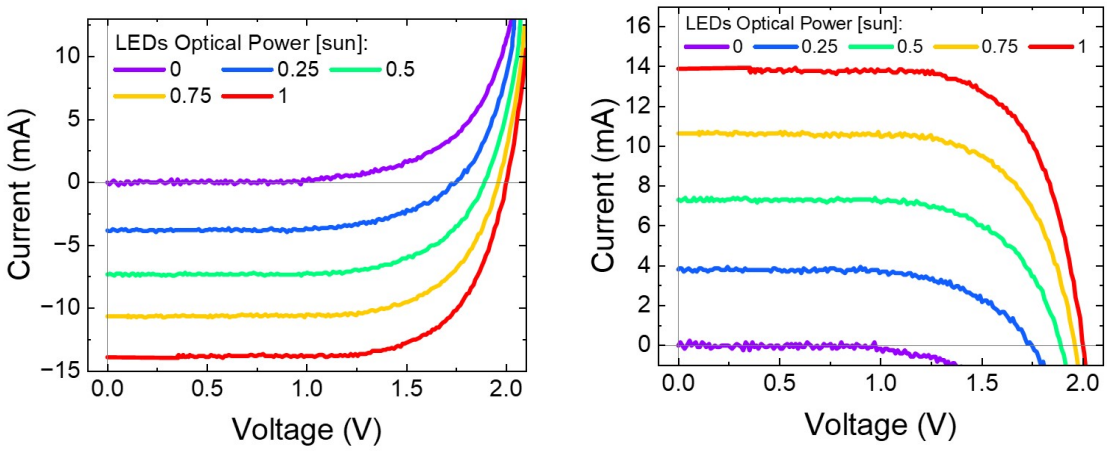
Sono stati toccati vari ambiti della progettazione elettronica, tra cui lo studio di un semplice circuito per effettuare misure di tensione e di corrente con l'ausilio di un amplificatore operazionale e un convertitore analogico-digitale , di un circuito per controllare l'accensione e l'intensità luminosa di un LED tramite l'utilizzo di convertitore digitale-analogico insieme a un operazione e a un transistor.

Si è applicato anche lo studio delle caratteristiche di circuiti integrati comuni come DAC e ADC, su come comunicano con un microcontrollore e quali operazioni effettuare per leggere e scrivere nei registri di questi componenti. In particolare sono state approfondite le caratteristiche del protocollo di comunicazione seriale SPI , dalla sua implementazione a livello hardware a quella software nel caso particolare della piattaforma Arduino.

Nella fase successiva si è utilizzato un software CAD (Autodesk Fusion) per la progettazione della scheda, disegnando lo schematico e il layout del PCB , creando i file necessari affinché possa essere stampato e realizzato fisicamente.

5.2 Risultati del progetto

Una volta pronta, questa sistema all'atto pratico permette di ottenere i valori di tensione e corrente caratteristici di una cella solare, con cui poi è possibile, ad esempio con l'ausilio di software come LabView, tracciare in tempo reale le curve che ne descrivono il funzionamento, a condizioni di luminosità variabili a piacimento.



(a) (b)

Figura 5.1: Esempio di funzionamento

In Figura 5.1 è illustrato il tipo di misurazioni che si possono ottenere con la SMU ivi descritta , una volta realizzata. Si ottengono dunque i grafici della caratteristica della cella a seconda della tensione ai applicati ai capi e a seconda della luminosità del LED. Da queste curve è poi possibile ricavare con una buona accuratezza, altri dati di interesse come la corrente di corto-circuito e la tensione a vuoto, e di come esse variano a seconda delle condizioni di illuminazione in cui opera la cella. L'utilizzo di componenti aggiuntivi come il DAC e l'ADC di alta precisione come quelli scelti in questo progetto, permette di ricavare delle misurazioni con una risoluzione migliore e quindi tracciare in maniera più accurata il comportamento, in particolare per bassi valori di tensione e luminosità.

Possibili miglioramenti suggeriti per sviluppi futuri possono essere ad esempio sostituire la scheda Arduino con il solo microcontrollore, per avere tutto il necessario su un unico PCB, una soluzione più pratica, e che può ridurre anche eventualmente rumore e distorsioni causati dai collegamenti tra due schede. Un ulteriore possibile miglioria è quella di aggiungere ulteriori componenti per una caratterizzazione ancora più dettagliata, ad esempio una serie di amplificatori operazionali con guadagno crescente (10, 100, 1000) per poter leggere meglio le correnti più basse.

Per concludere, il lavoro svolto permette di poter produrre, testare e applicare un piccolo sistema di misurazione di celle solari e diodi, i cui punti di forza risiedono nella facilità di riproduzione e di utilizzo, e nell'ampio valore dal punto di vista istruttivo nella sua possibilità di fornire un'introduzione a tutti gli aspetti toccati nei capitoli precedenti, dal design alla realizzazione di circuiti stampati, alla programmazione di un microcontrollore passando per lo studio delle caratteristiche elettriche di componenti elettronici.

Bibliografia

[1] C.B.Honsberg e S.G.Bowden, «Photovoltaics Education Website,» *www.pveducation.org*, 2019. indirizzo: www.pveducation.org.

[2] Arduino, *Arduino Documentation*, indirizzo: https://docs.arduino.cc/.

[3] A. Devices, «AD5752 Datasheet,» indirizzo: https://www.analog.com/en/products/ad5752.html.

[4] A. Devices, «AD7734 Datasheet (Rev. B),» indirizzo: https://www.analog.com/media/en/technical-documentation/data-sheets/ad7734.pdf.

[5] Anysolar, «KXOB25-04X3F-TB Datasheet,» indirizzo: https://waf-e.dubudisk.com/anysolar.dubuplus.com/techsupport@anysolar.biz/O18Adzm/DubuDisk/www/Gen2/KXOB25-04X3F%20DATA%20SHEET%20202007.pdf.

[6] Panasonic, «AQY211EH Datasheet,» indirizzo: https://www3.panasonic.biz/ac/e_download/control/relay/photomos/catalog/semi_eng_ge1a_aqy21_e.pdf.

[7] STMicroelectronics, «TIP41C Datasheet,» indirizzo: https://www.st.com/resource/en/datasheet/tip41c.pdf.

[8] Lumileds, «Luxeon TX Datasheet,» indirizzo: https://otmm.lumileds.com/adaptivemedia/00e48058eea0534796263328f6cc63e0555557bb.

[9] «AD5752 library,» *github*, indirizzo: https://github.com/kasskas/AD5752_lib/tree/master.

[10] «AD7734 library,» *github*, indirizzo: https://github.com/ucd-squidlab/AD7734Lib/tree/master.