Portable handheld drum machine

2024/2025
Sistemi elettronici

Introduzione e idea

Ho deciso di costruire un dispositivo portatile simile a una drum machine perché volevo uno strumento compatto per creare ritmi ovunque, senza dover dipendere da attrezzature ingombranti o costose. Spesso, i produttori musicali e principianti appassionati di beatmaking si trovano a dover lavorare su software complessi o hardware poco pratici da trasportare. Questo progetto mira a risolvere proprio questo problema, offrendo un'interfaccia semplice ma efficace per generare e modificare pattern ritmici in tempo reale. L'idea è quella di combinare portabilità, facilità d'uso e funzionalità essenziali per permettere a chiunque di sperimentare con la musica in qualsiasi momento e luogo.

Requisiti

Ho definito alcuni requisiti funzionali e li ho implementati passo per passo. I requisiti che mi sono posto sono i seguenti:

1. Sensori tattili per riprodurre i suoni
2. Un piccolo altoparlante integrato
3. Pulsanti/encoder/LED per interagire col dispositivo
4. Scheda micro-SD per salvare registrazioni
5. Header per impostare modalità output e gain dell'amplificatore
6. Connettore USB-C per alimentazione/programmazione/output
7. Regolatore Lineare per convertire da 5V a 3.3V

Implementazione

Ho scelto di usare un ESP32-S3 come microcontrollore perché si può comprare un modulo integrato con tanta memoria e antenna Bluetooth/WiFi integrata.

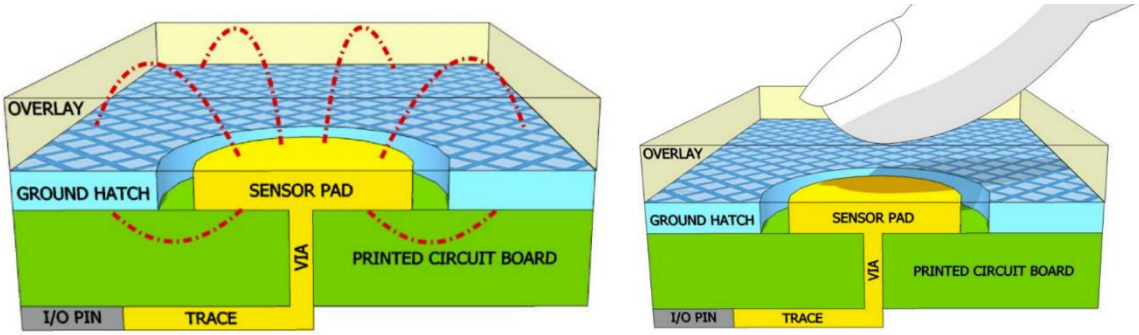
Nei prossimi capitoli spiegherò come ho implementato questi requisiti.

1. Sensori tattili

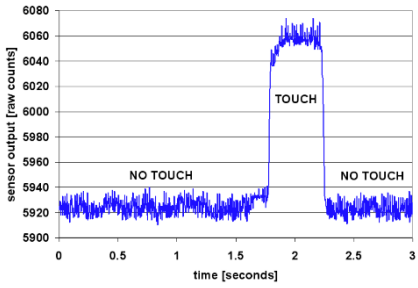
Per implementare questo, ho deciso di usare un circuito integrato dedicato alla lettura dei sensori, anche se l'ESP32 ha dei pin integrati che possono essere usati come sensori tattili di capacità. Ho preso questa decisione perché ci sono molti dispositivi collegati al microcontrollore e non ha tantissimi pin disponibili.

Nella prima versione del dispositivo ho usato il CY8CMBR3116, un circuito integrato di Cypress (Infineon) che può leggere fino a 16 sensori tattili e comunica con il microcontrollore tramite l'interfaccia I^2C.

Il modo in cui il sensore rileva un tocco è misurando la capacità. Quando un sensore viene toccato, il dito e l'elettrodo formano un condensatore a piatti paralleli.

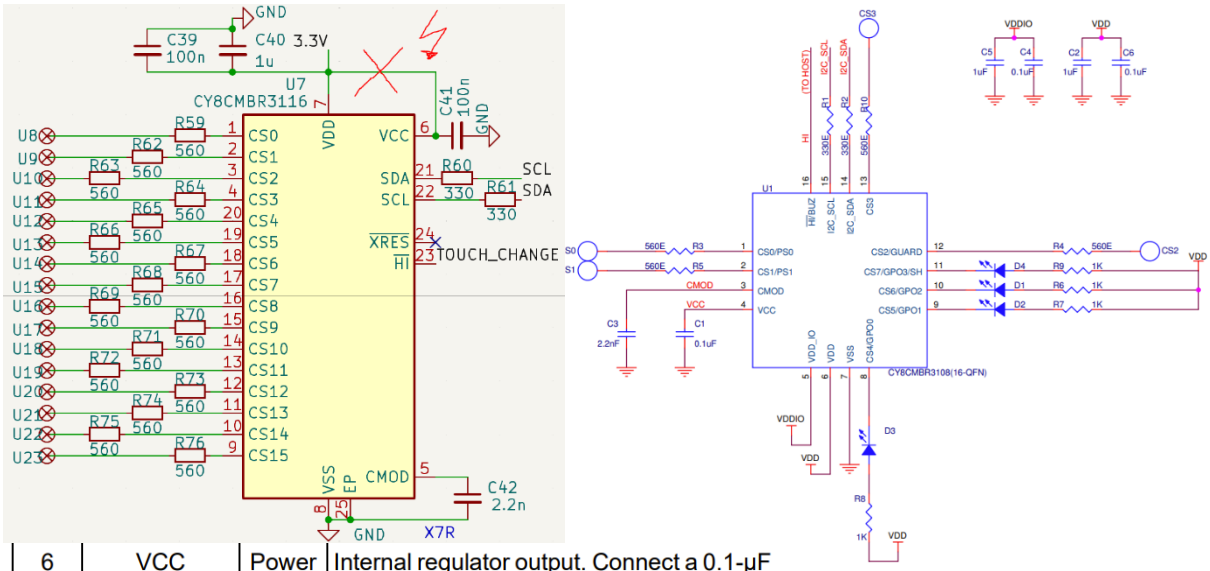


La capacità misurata è composta da due componenti: la capacità parassita e la capacità del dito. L'equazione è: $C_S = C_P + C_F$. Per aumentare la sensibilità, è necessario massimizzare il rapporto $\frac{C_F}{C_P}$.



Aumentando questo rapporto, si ottiene una differenza più alta nei valori misurati quando il sensore viene toccato rispetto a quando non viene toccato. Il modo in cui viene misurata la capacità è tramite un ADC di corrente, simile a un convertitore Sigma Delta. Il produttore chiama questo metodo CAPSENSE™ Sigma Delta (CSD) per il rilevamento della self-capacitance. È descritto più in dettaglio in questo documento [1].

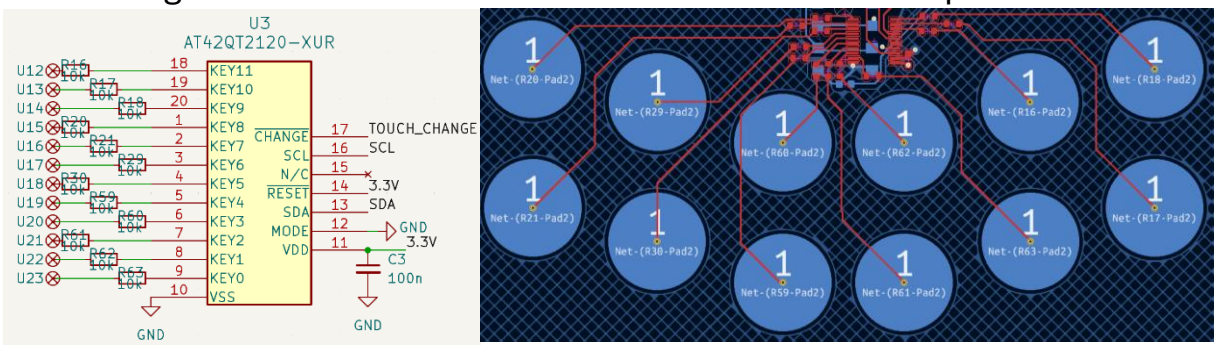
Purtroppo, nella prima versione del progetto ho fatto un errore nell'alimentazione del circuito integrato, che lo ha distrutto. È stato un errore che non doveva accadere, poiché nel datasheet era fornito il circuito di riferimento, e bisognava solo copiare.



6	VCC	Power	Internal regulator output. Connect a 0.1-µF decoupling capacitor if VDD > 1.8 V. If VDD is 1.71 V to 1.89 V, short this pin to VDD.

I pin VDD e VCC non vengono connessi se la tensione di alimentazione VDD non è 1.8V. Il motivo è che il pin VCC è connesso a un regolatore interno a 1.8V. In quel caso, bisogna collegare il pin solo a un condensatore bypass.

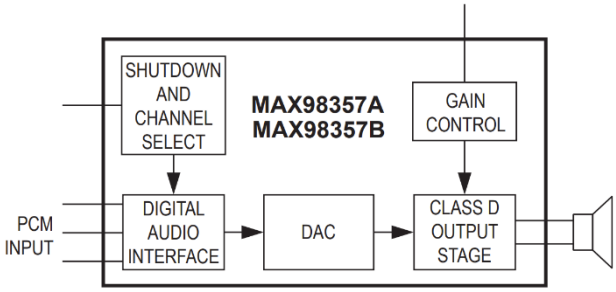
Nella seconda versione del progetto ho scelto un altro circuito integrato, l'AT42QT2120. Questo supporta solo 12 sensori tattili, che sono sufficienti per questa applicazione, ed è molto più facile sviluppare il software per questo IC. La tecnologia utilizzata da questo IC si chiama QtouchADC, ed è descritta più in dettaglio in questo documento [2]. Comunica anche tramite I^{2}C. Il disegno finale sul PCB nella versione attuale si vede qui.



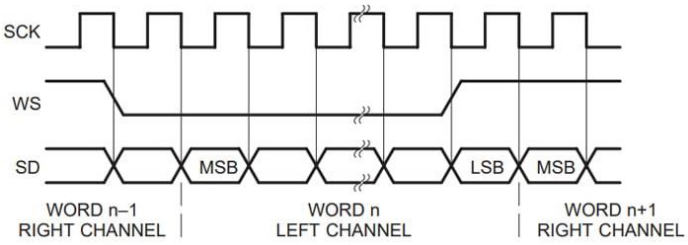
Ho cercato di rendere gli elettrodi molto accessibili con i pollici quando si tiene il dispositivo con due mani.

2. Amplificatore integrato

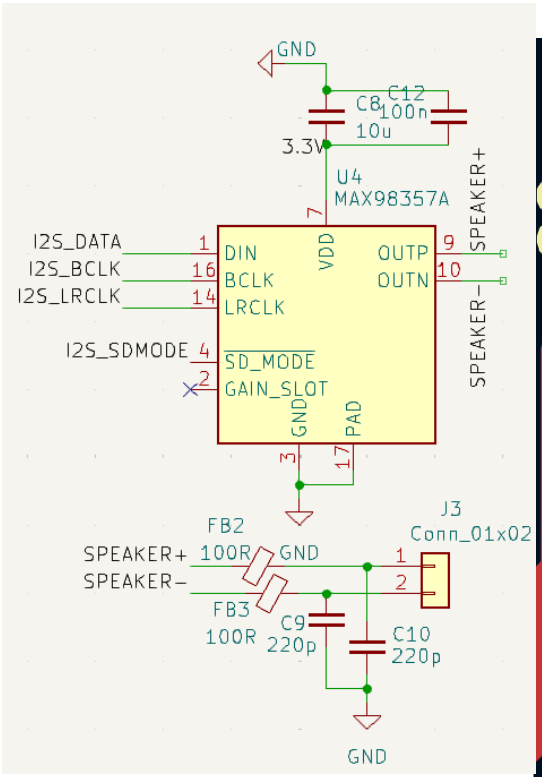
Ho scelto di usare un circuito integrato che combina DAC e amplificatore, poiché non volevo spendere troppo tempo nel disegnare un amplificatore specifico e intendevo utilizzare l'altoparlante solo durante il processo di sviluppo del software. La mia intenzione è poi di usare l'antenna Bluetooth dell'ESP32 come sorgente di audio Bluetooth. Così diventa possibile collegare il dispositivo a un altoparlante di alta qualità.
Ho trovato il componente MAX98357A [3]. È un DAC che supporta un segnale audio digitale I^2S, un protocollo compatibile con l'ESP32.
Per usare il protocollo I^2S bisognano tre collegamenti.

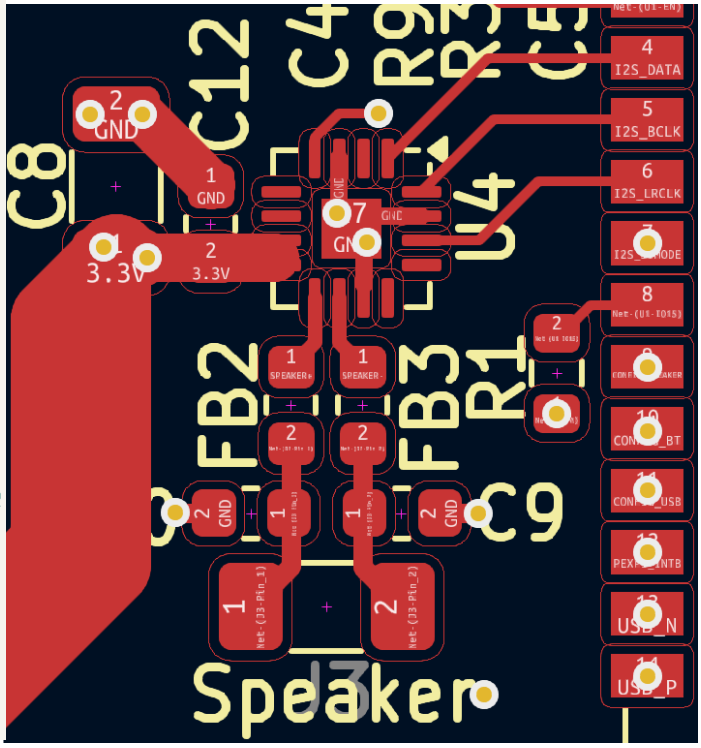


SD: Serial data
SCK: Serial clock
WS: Word select (Stereo)



Serial clock è il clock di base, e a ogni ciclo viene trasferito un bit. Word select determina a quale canale il dato appartiene. Serial data trasferisce i singoli bit a ogni ciclo clock.





La frequenza di clock dipende dalla frequenza di campionamento dell'segnale digitale, dalla precisione e dal numero di canali. Nel mio caso ho scelto i seguenti parametri: 16 Bits/sample, 44.1 kHz. Sono valori standard. Risulta la frequenza di clock: 44.1 kHz × 16 × 2 = 1.4112 MHz

L'altoparlante viene collegato al dispositivo attraverso un filtro passa basso con 2 perline di ferrite e 2 condensatori piccoli. Ho trovato un circuito di referenza, ma non ho fatto nessuna simulazione sull'impatto di questo filtro.

3. Pulsanti/Encoder/LED

Per rendere il funzionamento del dispositivo molto interattivo ho deciso di mettere una serie di pulsanti e anche degli encoder incrementali.

Le funzioni dei pulsanti sono i seguenti:

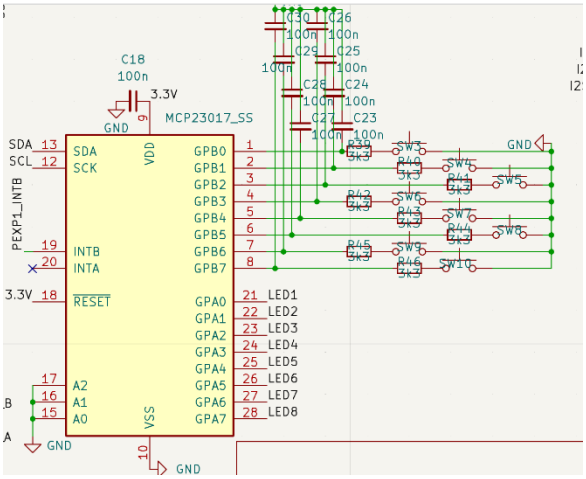
- Avvia/ferma registrazione con LED di indicazione
- Cancella tutta la registrazione
- Cancella la ultima sequenza registrata
- Pulsante per selezionare da 3 "track" con 3 LED
- Pulsante per selezionare da 3 "profili" con 3 LED
- 5 pulsanti per selezionare effetto con LED
- 3 pulsanti per avviare registrazione di una sequenza con LED

Gli encoder hanno queste funzioni:

- Impostare frequenza del metronomo
- Impostare volume della sequenza registrata
- Impostare volume del "track" selezionato
- Impostare volume master

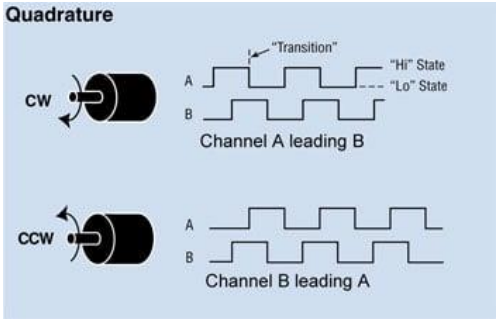
Gli ultimi due encoder hanno pulsanti integrati per silenziare il track/tutti i track.

Siccome non c'è un numero eccessivo di pin nel ESP32, ho deciso di collegare i pulsanti a 2 IO Expander. Anche questi comunicano tramite I^{2}C. Ho scelto il modello MCP23017. É un expander a 16 canali. Hanno 3 Pin per impostare l'addresso I^{2}C, quindi in teoria è possibile avere 8 expander sullo stesso bus. Ci sono due Pin di interrupt per gestire gli eventi di input.



I pulsanti sono collegati ai expander con un filtro a passa basso R=3.3k e C=100n e risulta τ=0.3ms. Non ho trovato valori suggeriti, ma ho trovato delle informazioni che i pulsanti piccoli hanno un tempo di bounce di 10-100us, quindi i valori di R e C dovrebbero essere abbastanza larghi.

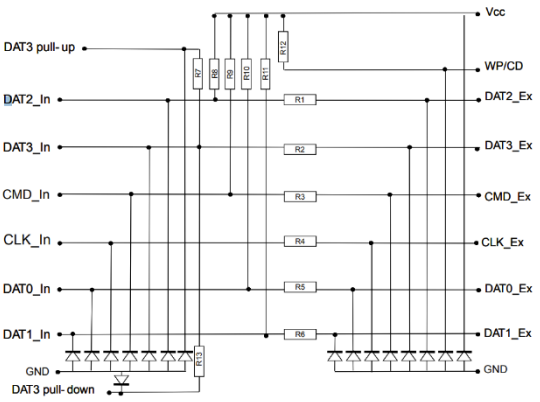
Siccome il segnale del encoder ha una frequenza più alta rispetto ai pulsanti, non li ho collegati ai expander, ma direttamente al ESP32. Il ESP ha 4 pulse counter integrati che sono in grado di contare i pulsi degli encoder senza usare interrupt, e hanno anche un debouncing digitale integrato che dovrebbe rendere il conto più stabile. Gli encoder sono a 24 Pulsi per rotazione, e il segnale è un segnale in quadratura con 2 canali. Ho assunto una frequenza massima di commutazione di 48 Hz, che corrispondono a 2 rotazioni completi in un secondo.

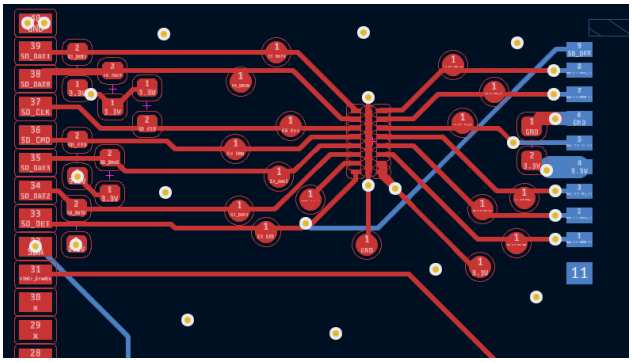


Ho scelto il modello PEC12R-4020-S0024 per gli encoder con pulsante integrato e PEC12R-4020-N0024 per quelli senza pulsante.

4. Scheda Micro SD

Per collegare la scheda Micro SD basta uno slot, ho scelto il modello DM3AT-SF-PEJM5. Per proteggere il ESP32 da eventuali connessioni difettosi o spike tensione ho messo un circuito integrato fatto apposta per i l'applicazione con Micro SD, e ho scelto il modello EMIF06-MSD02N16 [6] di STMicroelectronics. Conveniva scegliere uno con un package più grande perché era abbastanza difficile saldarlo senza corto circuiti. Il circuito interno include resistenze di serie,



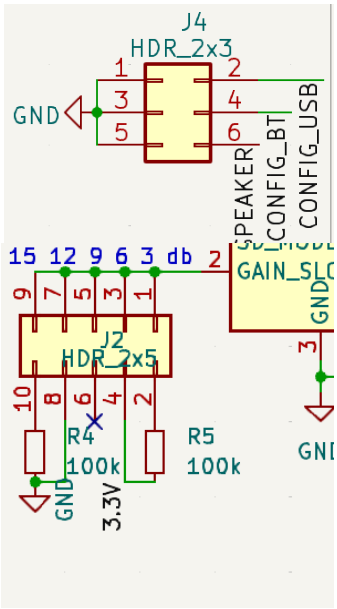


delle di pull up, e diodi TVS.

Siccome nella prima versione del PCB era molto difficile verificare la assenza di corto circuiti e la saldatura corretta, dato dalla larghezza del package e dal fatto che il ESP e lo slot Micro SD sono su layer opposti del PCB, nella seconda versione ho aggiunto dei test point.

5. Header Per Output e gain

Per selezionare tra 3 modalità di output ho deciso di mettere un Header 3x2. In quel modo, dopo il Boot il ESP32 vede quale modalità è selezionata e la Firmware fa tutto il resto. Si può selezionare tra Bluetooth, altoparlante e USB. In modalità bluetooth, il ESP32 fa da Server HTTP con un sito dove si può impostare quale dispositivo bluetooth da collegare. Se si seleziona altoparlante, l'output è un segnale I^2S all'amplificatore MAX98357A. Se si seleziona USB, l'output sarà o un segnale tipo Midi, o un segnale Audio tramite USB, devo ancora verificare quale dei 2 sarebbe più utile.



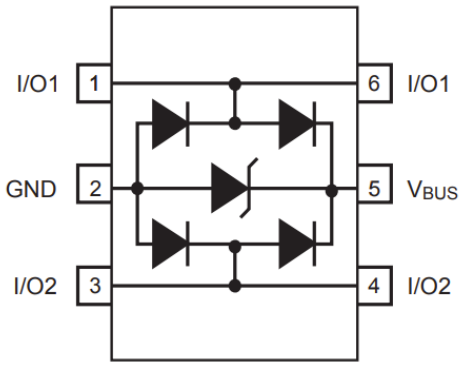
Nella seconda versione del PCB ho deciso di togliere i header Gain, perché sono molto grandi e non tanto utili.

6. Connettore USB-C per alimentazione/programmazione/output

Il ESP32 può essere programmato direttamente dall'interfaccia USB senza un convertitore USB-Serial. Questo rende il processo di disegno più facile. Dallo standard USB è richiesta un'impedenza di 90Ohm. Ho ottenuto valori di Track width/Track spacing da un calcolatore di JLCPCB che usa i parametri per dati layer-stackup venduti da JLCPCB. Poi bastava collegare le connessioni e far sì che non c'è una differenza troppo grande tra la lunghezza dei conduttori. Anche qui ho messo un circuito integrato per la protezione del ESP32, e il modello che ho scelto è il USBLC6-2P6 [7] da STM.

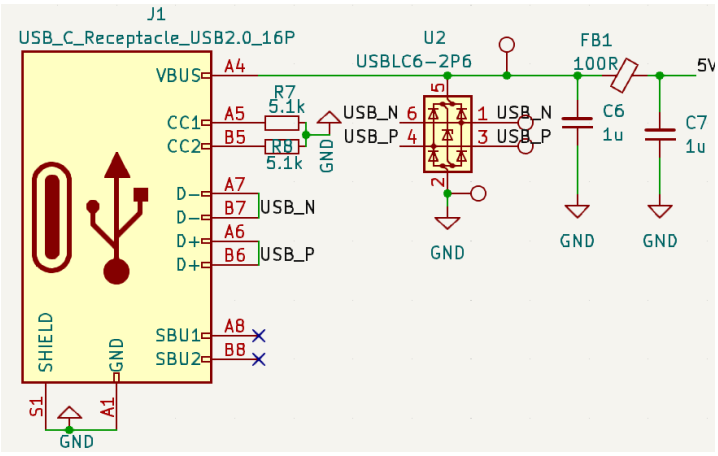
I valori che ho ottenuto sono:

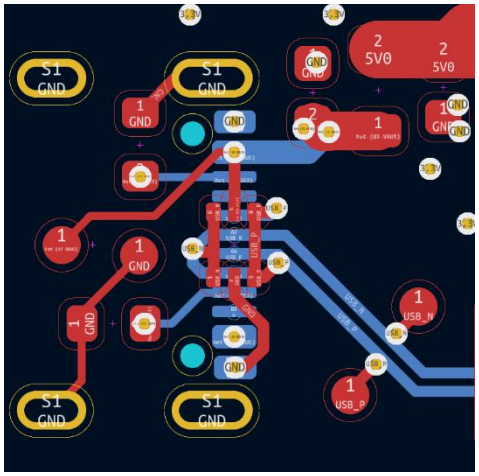
Track width: 0.284mm
Track spacing: 0.2mm



USB-C ha anche 2 conduttori CC1 e CC2 che si usano per configurare che tipo di dispositivo è connesso.

Bisogna collegare questi 2 conduttori a resistenze pull down con valori 5.1kOhm, per indicare che si tratta di un dispositivo downstream e per fare la comunicazione USB PD. Ho anche messo un PI filter per filtrare il rumore a frequenze elevate.



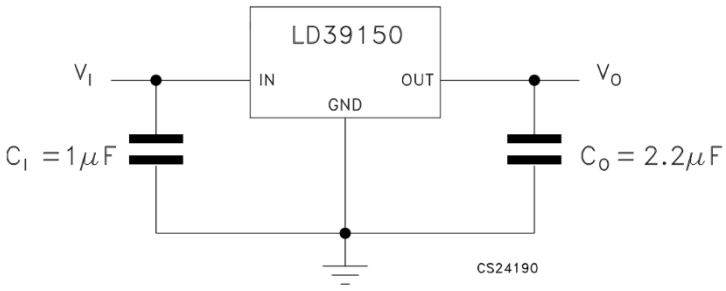


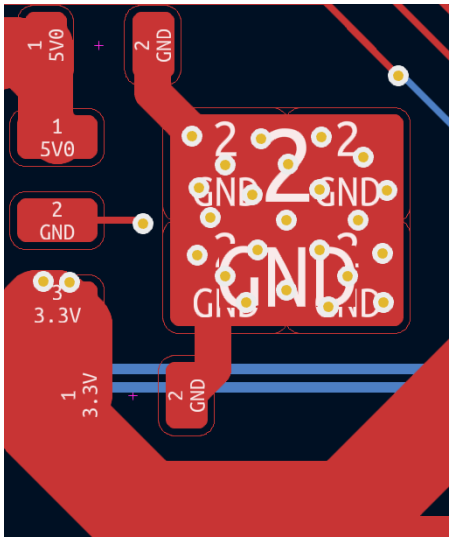
7. Regolatore lineare

Ho stimato che in totale, usando tutti dispositivi contemporaneamente la corrente totale sarà circa 1.5A. Per questo ho selezionato il regolatore lineare

LD39150DT33 [8]. Ha una tensione output fissa a 3.3V e una corrente massima di 1.5°, e ha anche una protezione interna riscaldamento elevato.

Siccome la tensione output è fissa, il circuito per alimentare il LDO è estremamente piccolo, consistendo solamente di tre componenti. I valori di





capacità nel datasheet sono valori minimi, e valori più elevati di capacità generalmente aumentano le prestazioni ho messo dei condensatori a 10uF.

Per aumentare la dissipazione di calore del regolatore, ho messo tanti via e all'interno del PCB il regolatore è connesso a due piani di Ground.

Obiettivi

Generalmente la prima versione del progetto è stato un successo, e poteva andare molto peggio. L'unica cosa che non funziona ancora è il sensore touch. Nella seconda versione del progetto, usando un altro sensore, che è anche più facile da programmare, vorrei risolvere questo problema. Risolvendo questo problema mi permetterà di iniziare a lavorare sulla software, che diventerà anche abbastanza complessa.

[1]https://www.infineon.com/dgdl/Infineon-AN90071_CY8CMBR3xxx_CapSense%28R%29_Design_Guide-ApplicationNotes-v09_00-EN.pdf?fileId=8ac78c8c7cdc391c017d07236b6e4698&utm_source=cypress&utm_medium=referral&utm_campaign=202110_globe_en_all_integration-application_note

[2]https://www.mouser.com/pdfDocs/QTouch_QTAN0079.pdf

[3]https://www.analog.com/media/en/technical-documentation/data-sheets/max98357a-max98357b.pdf

[4]https://www.allaboutcircuits.com/technical-articles/introduction-to-the-i2s-interface/

[5]https://www.dynapar.com/technology/encoder_basics/quadrature_encoder/

[6]https://www.st.com/resource/en/datasheet/emif06-msd02n16.pdf

[7]https://www.st.com/resource/en/datasheet/usblc6-2.pdf

[8]https://www.st.com/resource/en/datasheet/ld39150.pdf