UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea triennale

Moduli hardware per comunicazione e pilotaggio attuatori compatibili con il sistema ToLHnet

ToLHnet system-compatible hardware modules for communication and driving of actuators

Relatore: Chiar.mo
Prof. Biagetti Giorgio

Tesi di Laurea di:
Talucci Simone

A.A. 2018 /2019

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Moduli Hardware Per Comunicazione E Pilotaggio Attuatori Compatibili Con Il Sistema ToLHnet

(ToLHnet system-compatible hardware modules for communication and driving of actuators)

Indice

Introduzione **2**

Capitolo I **3**

Il protocollo ToLHnet 3

Capitolo II **7**

KiCad 7

Lo standard HAT 8

Progettazione dello schema elettrico 9

Capitolo III **13**

Progettazione dello schema elettrico 13

Realizzazione del PCB 18

Riferimenti **22**

Introduzione

In questa trattazione verrà illustrato lo sviluppo di due moduli hardware utilizzati all'interno di un progetto nel quale l'obiettivo è il controllo di un sistema di relè da remoto. Il protocollo impiegato per la comunicazione tra i vari componenti è il protocollo ToLHnet, che si presta particolarmente ad applicazioni di questo tipo per alcune caratteristiche che lo contraddistinguono.

Il protocollo ToLHnet prevede la presenza di un master in grado di comunicare con i vari nodi del sistema. In questa applicazione i vari nodi devono ricevere il comando dal master e devono essere connessi ai nodi terminali, ovvero alle schede relay. Essendo l'unica connessione possibile quella cablata, è necessario isolare le due porzioni di sistema qui descritte. È possibile ottenere l'isolamento tramite le schede di espansione progettate nella seconda parte della trattazione.

Nella prima parte è, dunque, presente il progetto dello schema elettrico di un PCB secondo lo standard HAT (Hardware Attached on Top) in grado di comunicare con una Raspberry Pi 3, un single-board computer sviluppato nel 2016 da Raspberry Pi Foundation. Per la comunicazione mediante protocollo ToLHnet è previsto l'inserimento di un modulo radio e di un'antenna che irradierà a 433 MHz. Il software del master deve infatti essere implementato nella Raspberry, che dovrà comunicare con i vari slaves proprio attraverso il modulo radio. Il componente selezionato per assolvere questa funzione è RFM23B. Il circuito integrato contiene inoltre una EEPROM, un'antenna e un connettore mini-din a 6 poli, in grado di comunicare attraverso un cavo con un modulo GPS, più altri componenti il cui inserimento è necessario al fine di ottenere prestazioni accettabili dal circuito.

Nella seconda parte sono illustrate la progettazione e la realizzazione del modulo da connettere ai vari nodi per realizzare l'isolamento descritto precedentemente. In particolare, per realizzare questo tipo di protezione bisogna separare la parte di potenza dalla parte di segnale della scheda, implementando inoltre tutte le precauzioni necessarie. La scheda ha quindi essenzialmente una funzione isolatrice, al fine di ottenere una connessione sicura tra i nodi del sistema ed i relè.

La trattazione inizierà con la descrizione del protocollo ToLHnet, per poi proseguire con una breve presentazione del software utilizzato per i progetti e concludersi con l'illustrazione dei due moduli hardware.

Capitolo I

Il protocollo ToLHnet

Il ToLHnet (Tree or Linear Hopping network) è un protocollo situato sul livello di rete dello standard ISO/OSI. Si presta ad essere utilizzato con una tipologia della rete di qualsiasi tipo, ed è fortemente asimmetrico. Il master è il nodo principale ed ha indirizzo 0, conosce la topologia della rete ed ha a disposizione gli indirizzi di tutti i nodi. Il software del master è abbastanza complicato e quindi è tipicamente un dispositivo differente rispetto ai dispositivi degli altri nodi, poiché necessita di capacità computazionali maggiori, mentre gli slaves hanno bisogno di una potenza di calcolo moderata. La comunicazione tra master e un nodo qualsiasi necessita di 4 byte oltre i bit informativi, e tra due nodi qualsiasi sono sufficienti 48 bit. Il protocollo è, quindi, molto leggero.

La topologia fisica della rete può avere una complessità arbitraria, ma, affinchè il routing del pacchetto venga semplificato, è opportuno costruire una topologia logica ad albero, la cui radice corrisponde con il master. Instradando i pacchetti lungo i rami della rete si ottengono infatti delle tabelle di routing più semplici, che conterranno una sola voce per ogni interfaccia fisica con allegati i figli e il route predefinito verso altre direzioni. Il compito di calcolare la rete logica a partire da quella fisica deve essere assolto dal master, che in alternativa può aver memorizzato una struttura ad albero pre-calcolata. Il master deve inoltre costruire le tabelle di routing, deve configurare i nodi durante la configurazione della rete, rilevare gli errori della rete ed interagire con l'utente o con altre interfacce. Ogni pacchetto contiene il livello di profondità del destinatario, che durante la trasmissione va confrontato con il livello di profondità del nodo (che esso stesso conosce) all'interno dell'albero. Per evitare eventuali collisioni durante la trasmissione e per garantire che la ritrasmissione avvenga lungo i rami della rete, si usa un flag che indica la direzione del pacchetto. I pacchetti che non sono destinati al nodo che li riceve vengono, banalmente, ignorati.

Come detto, il protocollo ToLHnet ha un'organizzazione logica ad albero, e la comunicazione avviene dal master a un nodo generico o da un nodo generico al master. Ogni pacchetto è indirizzato ad un singolo nodo identificato da un indirizzo unico. Esistono due tipi di indirizzi:

- Indirizzo Hardware (MAC): lungo 48 bit, che è unico ed identifica permanentemente ogni nodo. Tipicamente viene memorizzato in memorie non volatili. Viene utilizzato durante la procedura di configurazione iniziale dei nodi, ai quali deve essere assegnato un indirizzo di rete;
- Indirizzo di Rete: lungo 16 bit, viene assegnato dal master ad ogni nodo durante la configurazione della rete.

L'header di ogni pacchetto ha la seguente struttura:

- SRC: 16 bit, indirizzo di rete del nodo mittente; destinatario=0 (master);
- DST: 16 bit, indirizzo di rete del nodo ricevente; mittente=0 (master);
- HOPS: composto al massimo da 16 bit, i quali indicano la profondità corrente del pacchetto e vengono aggiornati ogni volta che il pacchetto transita da un nodo all'altro;
- DIR: bit di flag che indica la direzione del pacchetto.

Ogni nodo è caratterizzato dalle seguenti grandezze:

- address: 16 bit indicanti l'indirizzo di rete locale del nodo
- depth: profondità del nodo all'interno della rete.

Oltre a queste due grandezze ogni nodo è in possesso della tabella di routing, che viene assegnata dal master durante la configurazione iniziale.

Infine, ogni voce nella tabella di routing contiene i seguenti campi:

- span: intervallo di indirizzi comprendente gli indirizzi dei figli corrispondenti ad un nodo specifico;
- iface: l'interfaccia della rete fisica associata ad un determinato route;
- dir: la direzione del route, cioè il valore da aggiungere ad hops quando il pacchetto viaggia lungo il route.

Nel momento in cui il pacchetto arriva, il suo valore di hops viene confrontato con il valore di depth del nodo ricevente, e, nel caso in cui i due valori non corrispondano, il pacchetto viene scartato. Successivamente SRC e DST vengono cercati nella tabella di routing, e se i loro valori dir sono negativi, o se il valore dir di SRC coincide con DIR, il pacchetto viene scartato. In caso contrario il pacchetto viene accettato localmente o inoltrato a seconda del valore dell'indirizzo.

Il calcolo della struttura logica della rete ad albero viene effettuato mediante l'algoritmo di Dijkstra, che realizza un albero ottimale nel senso di ridurre al minimo il costo totale della trasmissione di un pacchetto da o verso la radice. Un esempio di applicazione è mostrato in *figura 1.1,* dove è raffigurata una catena di dieci nodi attaccati al master.

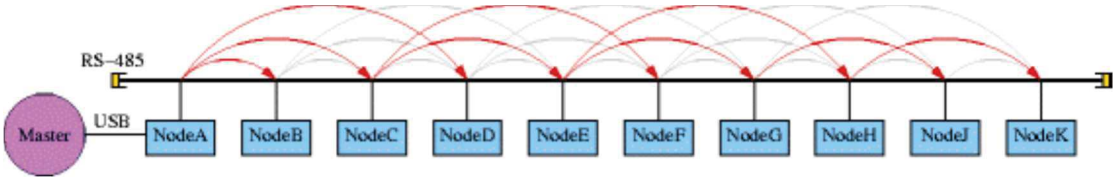
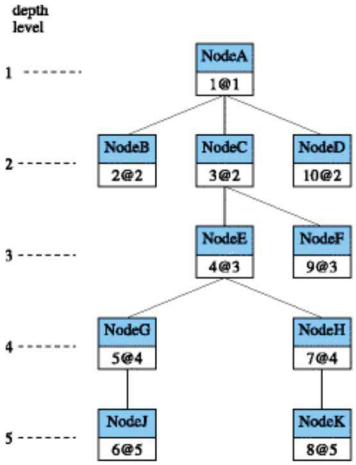


Figura 1.1, Fonte: [1]

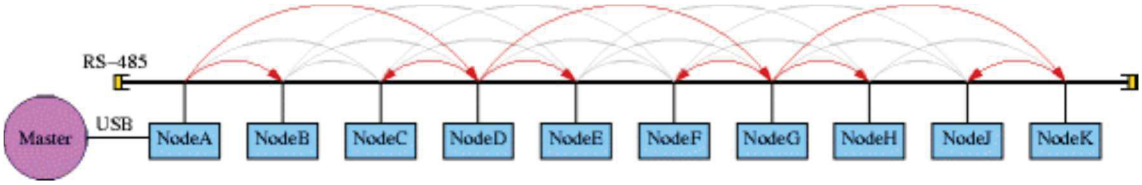
Tutti gli archi rappresentano i possibili percorsi della maglia, e quelli in rosso sono selezionati dall'algoritmo per appartenere all'albero. I nodi sono numerati in maniera tale che i figli di un nodo particolare condividano una parte dell'indirizzo. Per evidenziare i collegamenti in maniera migliore si può ridisegnare la rete come in *figura 1.2*:

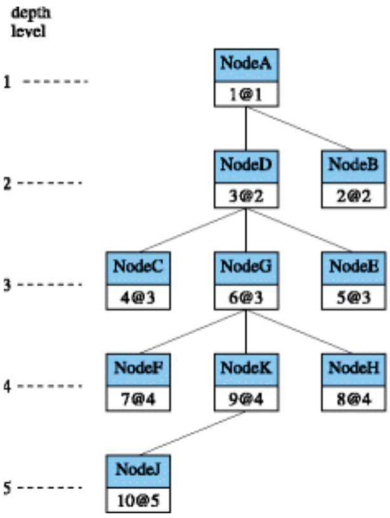


node name	node address	route #1 span	route #1 dir	default route span	default route dir
NodeA	1	2–10	+1	ALL	−1
NodeB	2			ALL	−1
NodeC	3	4–9	+1	ALL	−1
NodeD	10			ALL	−1
NodeE	4	5–8	+1	ALL	−1
NodeF	9			ALL	−1
NodeG	5	6–6	+1	ALL	−1
NodeH	7	8–8	+1	ALL	−1
NodeJ	6			ALL	−1
NodeK	8			ALL	−1

Figura 1.2, Fonte: [2]

Sebbene l'algoritmo di Dijkstra sia in grado di generare percorsi ottimali dal master a qualsiasi nodo e viceversa, per comunicazioni dirette tra due nodi potrebbe richiedere ai pacchetti percorsi inutilmente lunghi e, in casi estremi, potrebbe imporre al pacchetto di raggiungere il master nonostante sia richiesta una comunicazione tra due nodi vicini. Sono tuttavia implementabili alcune ottimizzazioni che permettono di evitare situazioni di questo tipo. A tal fine viene utilizzato un algoritmo specializzato, il quale assicura che l'albero sia realizzato in maniera tale da lasciare al massimo tre salti tra due nodi adiacenti fisicamente. Dal primo nodo viene selezionato il nodo più lontano raggiungibile per fungere da router, e il processo si ripete fino alla fine del collegamento. Tutti i nodi rimanenti vengono quindi assegnati come figli del router più vicino. Le tabelle di routing possono essere quindi estese con voci utilizzate solo per pacchetti generati localmente. Queste voci elencano tutti i nodi direttamente raggiungibili. Per fare in modo che la tabella di routing sia il più breve possibile, è richiesta una numerazione adeguata dei vari nodi. Realizzando la rete raffigurata nelle figure precedenti con l'algoritmo ottimizzato si ottiene una topologia di questo tipo:





node name	node addr	route #1 span	route #1 dir	route #2 span	route #2 dir	route #3 span	route #3 dir	route #4 span	route #4 dir	default route span	default route dir
NodeA	1	4–4	+2	2–10	+1					ALL	−1
NodeB	2	3–3	+0	4–5	+1					ALL	−1
NodeC	4	1–1	−2	2–2	−1	5–5	+0	7–7	+1	ALL	−1
NodeD	3	2–2	+0	7–7	+2	4–10	+1			ALL	−1
NodeE	5	2–2	−1	4–6	+0	7–8	+1			ALL	−1
NodeF	7	3–3	−2	4–5	−1	8–8	+0	10–10	+1	ALL	−1
NodeG	6	5–5	+0	10–10	+2	7–10	+1			ALL	−1
NodeH	8	5–5	−1	7–9	+0	10–11	+1			ALL	−1
NodeJ	10	6–6	−2	7–8	−1	11–11	+0			ALL	−1
NodeK	9	8–8	+0	10–10	+1					ALL	−1

Figure 1.3a, 1.3b, Fonti: [3] [4]

È possibile notare quanto l'algoritmo renda più efficiente tutto il procedimento di calcolo della struttura logica della rete: in figura 1 tra i nodi J e K, adiacenti, c'è una separazione pari a 4 hops, mentre con l'algoritmo modificato la separazione è pari ad 1 hops.

Durante la configurazione iniziale, il master deve quindi configurare tutti i nodi della rete assegnando loro, secondo le regole dell'algoritmo esposte precedentemente, l'indirizzo di rete, la profondità e la tabella di routing. La configurazione avviene inviando i pacchetti "CONFIG" ad

ogni nodo della rete. Dato che un nodo inizialmente non conosce questi parametri, l'assegnazione deve avvenire tramite il proprio indirizzo MAC, usando i pacchetti di rete con indirizzamento in modalità "DST+MAC/DEPTH". I successivi comandi di configurazione possono poi essere inviati utilizzando l'indirizzamento canonico. La ricezione del comando "CONFIG" da parte del nodo dà inizio al processo di configurazione e il nodo reimposta i suoi parametri di rete come segue:

- memorizza l'indirizzo di rete di destinazione come suo nuovo indirizzo di rete;
- memorizza il valore di depth del pacchetto come suo valore di profondità nell'albero;
- resetta la sua tabella di routing e la ricompila come indicato nel comando.

I pacchetti di configurazione ricevuti successivamente, nei quali viene usato l'indirizzo di rete e non l'indirizzo MAC, indicano le regole di routing che il nodo deve aggiungere, eliminare o modificare, senza che le precedenti informazioni vengano reimpostate. Quando un nodo reimposta la sua tabella di routing, per funzionare correttamente deve impostare due regole che non sono esplicitamente incluse nei comandi di configurazione inviati dal master.

La prima regola "implicita" è per i pacchetti che vengono accettati localmente, e possono essere codificati in questo modo:

XXXX 0 00

dove XXXX è l'indirizzo di rete del nodo assegnato dal comando di configurazione e il valore 00 indica all'interfaccia l'accettazione locale. Questa è la regola che verrà abbinata ai pacchetti diretti al nodo e tipicamente non ha valore di direzione.

La seconda regola è per i pacchetti che devono essere instradati al nodo genitore, che possono essere codificati come segue:

0000/0 (0000) -1 II

dove II è l'interfaccia dalla quale il comando di configurazione è stato ricevuto, identificando la connessione con il nodo genitore.

Si può notare come il master abbia bisogno di configurare i nodi con regole aventi sempre +1 come direzione, dato che gli altri casi sono coperti da regole implicite. Le regole con direzione positiva indicano ad un nodo quali sono i suoi nodi figli, indicando quindi come è strutturata la rete logica ad albero, e da quali interfacce raggiungerli. Dato che un nodo non può instradare pacchetti prima di essere configurato, il master dovrà, come prima cosa, inviare il comando di configurazione ai suoi nodi figli diretti, poi ai nodi di livello di profondità successivo e così via. È a questo proposito che si rende necessaria la conoscenza della topologia della rete da parte del master.

Capitolo II

Dopo una breve descrizione del programma utilizzato per questo progetto, il capitolo riguarderà la progettazione di un HAT per Raspberry Pi3. Sarà quindi illustrato lo standard HAT, per poi procedere con l'illustrazione dello schema elettrico progettato.

KiCad

Per realizzare questo progetto il software utilizzato è KiCad, una suite open-source[1] di software EDA (Electronic Design Automation) utilizzato per il disegno di uno schema elettrico e di un circuito stampato (Printed Circuit Board, o più comunemente PCB). Il programma è organizzato in quattro parti:

- kicad: il project manager;
- eeeschema, cvpcb: eeeschema è l'editor dello schema, in cui è compreso anche il cvpcb, che permette di associare le impronte ai vari simboli elettrici;
- pcbnew: permette di eseguire lo sbroglio del PCB;
- gerbview: permette di visualizzare il file gerber, un formato che permette la produzione del circuito stampato.

La versione di KiCad utilizzata in questa trattazione è la 5.1.5.

Esistono diversi software che permettono di realizzare progetti come quelli descritti. La scelta è ricaduta su KiCad in quanto quest'ultimo permette di gestire tutte le fasi del lavoro con la stessa interfaccia.

[1] Il termine open-source viene utilizzato per fare riferimento ad un tipo di software la cui licenza permette la pubblicazione, la modifica, l'utilizzo e la redistribuzione del codice sorgente.

Lo standard HAT

Un HAT è un PCB rettangolare di dimensioni 65x56 mm, che ha quattro fori tondi negli angoli, i quali si allineano con i fori della Raspberry Pi 3 affinché il montaggio risulti più agevole. Comprende poi un header di 40 fori, allineati anch'essi con i pin della scheda madre e supporta il sistema di autoconfigurazione che consente l'impostazione automatica dei GPIO e la configurazione del driver. La configurazione automatica si ottiene utilizzando due pin denominati ID_SD e ID_SC, riservati per una EEPROM I^2C. La memoria deve contenere, oltre le informazioni sul produttore della scheda, le informazioni sulla configurazione dei pin e una descrizione dell'hardware collegato, che consente a Linux di caricare automaticamente i drivers richiesti.

Per realizzare un PCB in grado di comunicare con Raspberry Pi 3 non è obbligatorio realizzare un modulo che segua gli standard HAT, ma, per un circuito stampato con queste caratteristiche, è garantita la compatibilità anche con modelli successivi della scheda, oltre che un'esperienza d'uso migliore.

Per garantire l'accesso a tutti i connettori della scheda, un HAT ha, in realtà, una forma leggermente modificata rispetto ad un normale rettangolo, come mostrato nella *figura 2.1*:

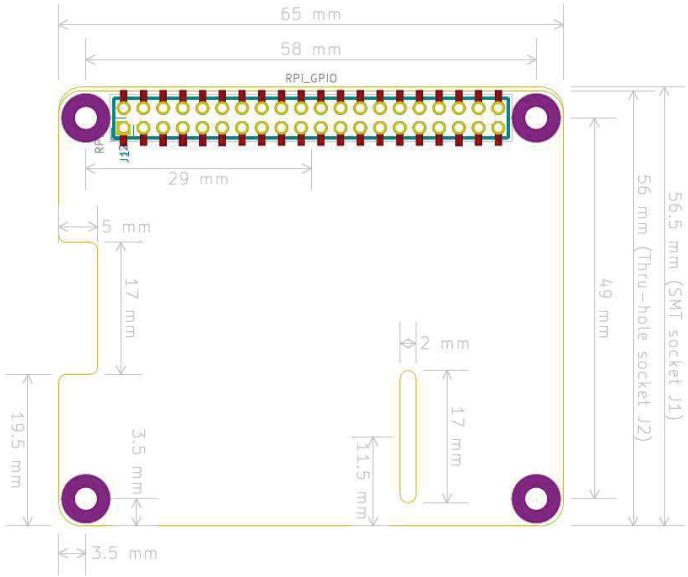


Figura 2.1

Progettazione dello schema elettrico

Come anticipato nell'introduzione, il componente principale del PCB è il modulo radio RFM23B-S1, il quale presenta 16 pin così definiti:

- 1: pin di input/output del segnale a radiofrequenza (impedenza di 50 Ω). È quindi il pin al quale è collegato lo switch che seleziona l'antenna da utilizzare;
- 2: riferimento di massa per l'antenna;
- 3, 4: pin non connessi;
- 5: pin di alimentazione. L'alimentazione raccomandata vale 3,3 V, ed è preferibile inserire un LDO tra la linea di alimentazione e il pin, dato che i 3,3 V erogati dalla Raspberry potrebbero essere troppo disturbati per alimentare un modulo radio;
- 6, 7: GPIO 0, GPIO 1, utilizzati per controllare lo switch SPDT;
- 8: GPIO 2, utilizzato per il segnale di clock del modulo radio;
- 9: riferimento di massa;
- 10, 11, 12: pin del bus seriale di comunicazione (SPI);
- 13: nSEL, pin che svolge la funzione di Slave Select del bus SPI;
- 14: nIRQ, pin di output che, se impostato sul livello logico basso, segnala un evento di interruzione;
- 15: SDN, pin di input che, se impostato sul livello logico alto, provoca l'arresto del modulo radio, perdendo di conseguenza tutto il contenuto dei registri. A questo pin è collegata una resistenza di pull-up dal valore di 100 kΩ;
- 16: riferimento di massa.

Come accennato precedentemente, per avere una linea di alimentazione immune dai disturbi, necessaria per ottenere un buon funzionamento del modulo radio, è raccomandato l'inserimento di un LDO (Low-Dropout Regulator). Il modello scelto è il "TPS73633" prodotto da Texas Instruments. Il LDO è connesso al pin di alimentazione a 5 V della Raspberry e deve abbassare la tensione in uscita fino ai 3,3 V necessari per alimentare il modulo radio. I pin dell'integrato sono 5:

- 1: tensione di ingresso, che in questo caso è prelevata direttamente dalla Raspberry e vale 5 V;
- 2: tensione di uscita;
- 3: GND, riferimento di massa;
- 4: NR/FB: nelle versioni a tensione regolabile è l'input per l'errore del circuito di controllo, e viene utilizzato per fissare la tensione di uscita del dispositivo;
- 5: pin di Enable, controllato direttamente dalla Raspberry.

Il circuito completo utilizzato per fissare la tensione di uscita a 3.3 V è illustrato in *figura 2.2* e comprende, oltre il LDO, due resistori e tre condensatori:

Il valore dei resistori è fissato in base alla formula fornita dal produttore:

$$V_{out} = \frac{(R_1+R_2)}{R_1} * 1{,}204$$

Il valore dei condensatori, invece, è fissato a 10, 100 nF o 1 μF, in base alle esigenze di filtraggio. Il range di valori all'interno del quale deve ricadere la capacità dei condensatori è fornito anch'esso dal costruttore. Per questo tipo di applicazione sono raccomandati condensatori con ESR[2] basso, dunque è preferibile utilizzare condensatori ceramici.

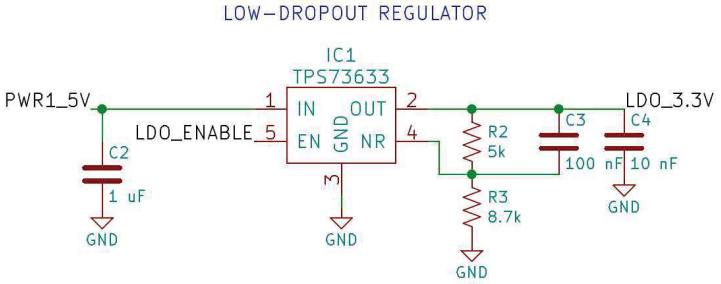


Figura 2.2

Per dare la possibilità all'utilizzatore di cambiare antenna in base alle esigenze di trasmissione, nella scheda è inserito uno switch SPDT. Il modello scelto è "AS169-73LF". Il componente ha 6 pin:

- J1: pin connesso al pin 1 del modulo radio. È presente un condensatore per realizzare un accoppiamento in AC, il cui valore è di 100 pF;
- GND: riferimento di massa;
- J2, J3: sono connessi direttamente alle antenne tramite condensatore in serie il cui valore è di 100 pF;
- V1, V2: sono connessi ai pin GPIO 0 e GPIO 1 del modulo radio, e sono le tensioni di controllo dello switch. Seguono le indicazioni della seguente tabella di verità:

V_1	V_2	J_1-J_2	J_1-J_3
V_{HIGH}	V_{LOW}	Isolation	Insertion loss
V_{LOW}	V_{HIGH}	Insertion loss	Isolation

È raccomandato di non utilizzare altre combinazioni oltre quelle indicate in tabella.

In condizioni tipiche (25°, Z_0=50 Ω) isolation vale 25 dB, mentre insertion loss vale 0,3 dB.

La EEPROM selezionata per la realizzazione del HAT è il modello "24LC64", nel formato SOT23. La memoria è in grado di immagazzinare 64 kbit ed è in grado di interfacciarsi con altri dispositivi mediante bus I^2C. In questo package il componente ha 5 pin:

- SCL, SDA: rispettivamente clock e linea dati della connessione seriale I^2C. La memoria è compatibile con clock da 100 kHz o 400 kHz;
- V_{CC}: pin di alimentazione. L'intervallo di valori nel quale può variare l'alimentazione va da 1,7 V a 5,5 V. In questo caso l'alimentazione è fissata a 3,3 V;
- V_{SS}: riferimento di massa;

[2] ESR è la resistenza serie del circuito equivalente di un condensatore reale.

- WP: Write-Protect Input. Se portato ad una tensione pari a V_{CC} le operazioni di scrittura sono inibite, mentre sono permesse le operazioni di lettura.

Le connessioni della EEPROM sono concordi con quanto richiesto dallo standard HAT.

Per ridurre i tempi di salita e di discesa (da qui in poi denominati rispettivamente t_r e t_f) è inserito un multiplexer. L'inserzione di questo componente permette l'utilizzo di un cavo più lungo (e quindi con capacità parassite maggiori) e/o una velocità di trasmissione più elevata. Il componente scelto è il "74LVC1G157". Il dispositivo ha 6 pin:

- 1,3: I1, I0, sono i due ingressi del multiplexer. I dati in trasmissione si trovano sull'ingresso I1, mentre i dati in ricezione sono collegati all'ingresso I0;
- 2: GND, riferimento di massa;
- 4: Y, uscita del multiplexer;
- 5: V_{CC}, alimentazione a 3,3 V, prelevata direttamente dalla Raspberry;
- 6: S, pin di selezione.

Il pin di selezione deve sempre rimanere alto, in modo da fissare l'uscita al valore di I1. Come specificato nel datasheet, per capacità parassite di 50 pF i tempi di salita e discesa vengono ridotti a 2,5 ns. Per capacità parassite più alte t_r e t_f aumentano proporzionalmente.

Infine è presente un connettore mini-din a 6 poli, così da permettere alla scheda di interfacciarsi con altri moduli hardware. I segnali collegati ai pin del connettore mini-din sono:

- 1: PPS: segnale elettrico che indica con precisione l'inizio di un secondo. Può essere utilizzato per sincronizzare in maniera molto precisa un dispositivo;
- 2, 4: rispettivamente i dati trasmessi e i dati ricevuti;
- 3: riferimento di massa;
- 5: alimentazione a 5 V;
- 6: alimentazione a 3.3 V in uscita dal LDO.

I pin di shield sono connessi a massa tramite un induttore. Questo accade per evitare che vengano indotte correnti ad alta frequenza troppo grandi sullo schermo, e quindi che quest'ultimo possa irradiare. La connessione avviene tramite un induttore in serie poiché le frequenze di interesse per questo fenomeno sono le alte frequenze, mentre non è necessario aumentare l'impedenza del componente a bassa frequenza. L'induttore svolge adeguatamente questa funzione, dato che la sua impedenza è direttamente proporzionale alla pulsazione.

Fatte queste considerazioni, lo schema elettrico risultante del modulo hardware è illustrato in *figura 2.3*:

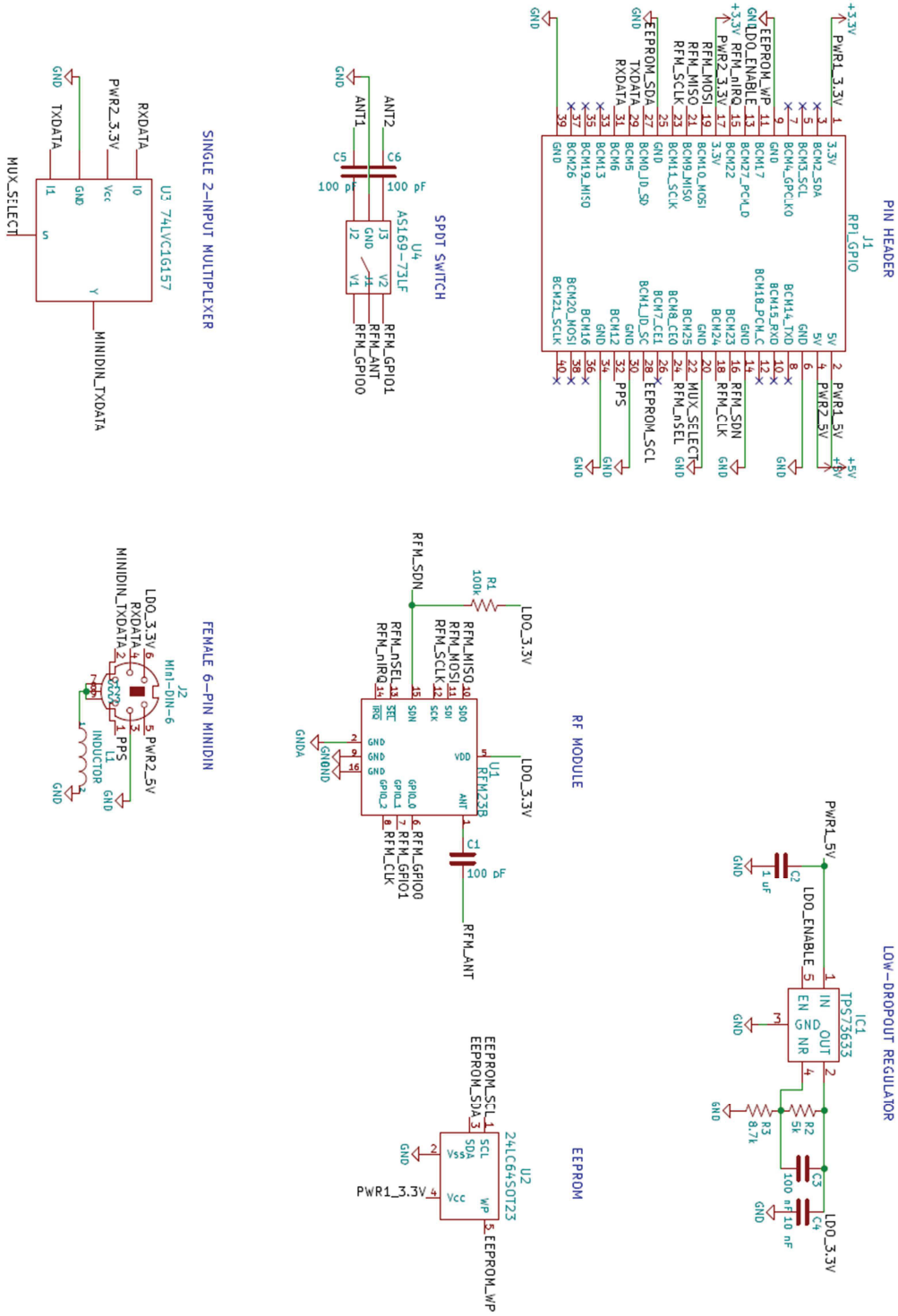


Figura 2.3

Capitolo III

In questo capitolo è descritto il progetto del secondo modulo hardware. Il PCB in questione è situato in una posizione intermedia tra il nodo generico ed i relè, ed è utilizzato per garantire l'isolamento galvanico tra le due parti, migliorando la sicurezza elettrica complessiva dell'intero apparato.

Progettazione dello schema elettrico

Per realizzare l'isolamento al quale si è accennato nell'introduzione i componenti chiave sono essenzialmente due: un convertitore DC-DC, utilizzato per realizzare l'alimentazione delle schede relay del sistema, isolandole dalla parte di segnale, ed un optoisolatore, utilizzato per trasferire il segnale tra le due parti mantenendo l'isolamento galvanico tra le stesse.

Il convertitore DC-DC scelto è il modello NMJ0505SAC nella versione Single Output. Il convertitore ha ingresso ed uscita a 5V, ha una efficienza del 68%, ed ha quattro pin, due di ingresso e due di uscita. Denominando la massa dell'alimentazione a 5V con GNDS i pin di ingresso sono rispettivamente il pin a 5V e il pin GNDS. I pin di uscita sono il pin a 5V con la sua massa, differente da quella di ingresso e denominata GND. Il convertitore è in grado di erogare una corrente di uscita che può arrivare fino a 200 mA, ma, per funzionare nella maniera adeguata, non può erogare meno di 20 mA. In questo caso infatti la tensione di uscita risulterà incrementata e potrà arrivare ad essere il doppio di quella prestabilita nel caso in cui la corrente erogata sia il 5% di quella massima, con conseguente danneggiamento dei componenti collegati a valle. Per realizzare questa condizione di consumo minimo vengono inseriti in uscita due resistori in parallelo.

In ingresso al convertitore non è richiesto alcun condensatore (che è già presente al suo interno), in uscita invece bisogna inserire un induttore ed un condensatore secondo lo schema in *figura 3.1*:

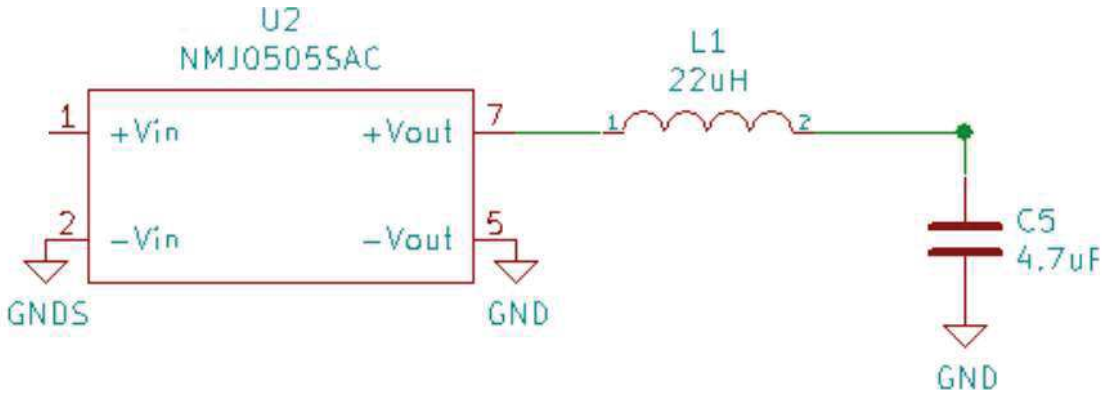


Figura 3.1

I valori fissati per l'induttore e per il condensatore sono rispettivamente 22μH e 4,7μF, come richiesto dal datasheet del convertitore. Questi due componenti sono entrambi SMD di dimensioni "0603"[3].

Il convertitore DC-DC è un componente dal consumo abbastanza elevato, dunque è opportuno realizzare un circuito che permetta di spegnere ed accendere il componente in base ad un segnale di Enable. Per realizzare questa funzione viene inserito a monte uno switch. A valle, per permettere all'utilizzatore di capire se il convertitore è acceso o meno, viene inserito un LED (anche questo "0603").

Lo switch scelto è il modello "TPS2553DBVT" prodotto da Texas Instruments. Lo switch ha 6 pin:

- 1: pin al quale è collegata l'alimentazione a 5V. Nel datasheet dello strumento è richiesto il collegamento tra questo pin e la massa (GNDS) di un condensatore da 100 nF, anche questo "0603".
- 2: pin di massa, denominata GNDS nello schema.
- 3: pin di Enable. È presente una resistenza di pull-down dal valore di 100 kΩ che consente di evitare comportamenti indesiderati durante l'accensione del sistema.
- 4: pin di Fault, quindi di output, che segnala eventuali anomalie come ad esempio sovracorrenti o riscaldamento eccessivo.
- 5: pin di output denominato Ilim, al quale va collegata una resistenza compresa tra i 15 kΩ e i 232 kΩ. La resistenza stabilisce la soglia limite di corrente[4]. Il valore del resistore è fissato a 47 kΩ.
- 6: pin di uscita. È richiesto un condensatore collegato tra questo pin e massa il cui valore viene fissato nel datasheet del componente e vale 4,7μF.

Per permettere all'utilizzatore di capire se il convertitore DC-DC è acceso o meno viene inserito un LED con in serie una resistenza da 330 Ω. Quando il convertitore è acceso, quindi, il LED si illumina. Infine sono inserite due resistenze in parallelo che hanno come unico scopo quello di consumare corrente cosicché si arrivi al consumo minimo per cui il funzionamento del convertitore è garantito. Le resistenze, dovendo consumare e quindi dissipare calore, vanno collocate abbastanza distanti nel circuito stampato sia tra di loro che tra altri componenti che potrebbero dissipare energia: essendo componenti piccoli il problema della dissipazione di energia è tutt'altro che irrilevante, poiché per unità di volume la quantità di energia dispersa è elevata. Il consumo dell'optoisolatore vale 2,5 mA, quello del LED è di 10 mA, e dunque sui due resistori deve scorrere una corrente complessiva di almeno 7,5 mA. Volendo eccedere leggermente questo valore (ad esempio del 10%) si può stabilire il valore della resistenza equivalente dei due resistori a 600 Ω, sui quali scorre una corrente complessiva di 5 V/0,6 kΩ= 8,3 mA, sufficiente per gli scopi prefissati. Inserire due resistori da 1 kΩ e 1,5 kΩ in parallelo è una soluzione migliore rispetto all'inserzione di un solo componente da 600 Ω poiché la dissipazione del calore è più facilmente gestibile su due componenti piuttosto che su uno solo. L'uscita del convertitore DC-DC è quindi illustrata in *figura 3.2*.

[3] La sigla fa riferimento alla dimensione in pollici (lunghezza x larghezza) 0,063" x 0,031", che corrisponde, nel sistema internazionale, ad 1.6 mm x 0.8 mm.

[4] Il valore nominale della corrente è stabilito dalla seguente equazione: $I_{OS}[mA]=\frac{23950\,V}{R_{ilim}{}^{0.977}k\Omega}$

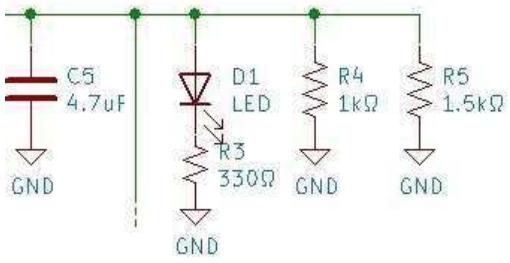


Figura 3.2

L'optoisolatore (chiamato più comunemente fotoaccoppiatore) viene utilizzato con lo scopo di tenere isolate galvanicamente le due parti del circuito trasferendo i segnali di trasmissione e ricezione da una parte all'altra. Il componente scelto è l'optoisolatore "FOD8012A", che ha 8 pin:

- 1, 8: pin di alimentazione di ingresso e di uscita. L'alimentazione di ingresso del componente vale 3,3 V, l'uscita invece è presa direttamente dal filtro in uscita del convertitore DC-DC e vale 5 V. Ad entrambi i pin sono connessi due condensatori "0603" collegati a loro volta alle rispettive masse dal valore di 100 nF, come richiesto nel datasheet del dispositivo;
- 2, 3: sono rispettivamente i pin di ricezione e trasmissione collegati al connettore che poi viene collegato a sua volta ai GPIO;
- 4: pin di massa riferito all'alimentazione da 3,3 V. Non c'è necessità di avere una massa comune con l'alimentazione a 5 V, dunque le due masse non sono poste necessariamente allo stesso potenziale e questa è denominata GNDA;
- 5: pin di massa riferito al lato di potenza del circuito, ovvero la GND introdotta in precedenza;
- 6, 7: pin di trasmissione e ricezione sul lato di potenza del PCB. Al pin di ricezione è collegata una resistenza di pull-down dal valore di 4,7 kΩ, anch'essa "0603".

Tutti i pin nel lato di potenza vengono collegati ad un connettore mini-din a 4-pin, che permetterà di trasportare con un cavo il segnale. Nel pin 1 del connettore mini-din è connesso il segnale di trasmissione, al pin 2 è connesso il segnale di ricezione, al pin 3 è connessa l'alimentazione a 5 V ed infine il pin 4 è connesso a GND, in analogia con il vecchio Apple Desktop Bus.

Restano infine i pin di shield, pin passivi indicati nello schema con i nomi S1, S2 ed S3. Questi tre pin, in teoria, vanno collegati a massa, ma potrebbe risultare utile inserire un induttore tra i tre pin e il nodo GND. L'induttore, anch'esso "0603", potrebbe tuttavia non essere necessario. Nel caso in cui non lo fosse sarà sufficiente inserire al suo posto un resistore da 0 Ω per realizzare il cortocircuito tra S1, S2, S3 e GND. Come spiegato per il modulo hardware precedente, l'induttore viene inserito al fine di aumentare l'impedenza dello schermo ad alte frequenze, in modo da impedire l'induzione di correnti che consentirebbero allo schermo di irradiare come se fosse un'antenna, causando problemi di EMI.

Il cavo che va collegato al connettore mini-din ha una capacità parassita abbastanza elevata, di circa 150 pF/m. Per trasmettere a 115.200 baud la capacità parassita massima sopportabile senza che le prestazioni degradino in maniera inaccettabile è di circa 2.400 pF e quindi, con una configurazione come quella descritta, non è consigliato impiegare un cavo troppo lungo. Alla luce delle considerazioni precedenti il limite massimo di lunghezza può essere fissato intorno a 16 m e, nel caso

in cui fosse necessario utilizzare cavi più lunghi e/o trasmettere a velocità maggiori, sarebbe opportuno inserire all'interno dello schema anche un buffer.

Per stabilire il valore della capacità parassita massima con la quale si può avere una trasmissione dalla qualità accettabile sono state fatte le seguenti considerazioni: il baud rate[5] minimo, come detto, è fissato a 115.200 baud. Questo significa che verranno trasmessi 92.160 bit/s. La larghezza d'impulso di un bit si può ricavare calcolando l'inverso di questa quantità, cioè 10.850 ns (il risultato è arrotondato per difetto per ottenere una condizione finale leggermente più restrittiva). Con una capacità di carico di 15 pF e con una larghezza di impulso pari a 66,67 ns, seguendo le indicazioni del datasheet, t_r e t_f valgono 6,5 ns, ovvero circa il 10,25% ognuno della larghezza di impulso complessiva. Volendo continuare a rispettare questa condizione si può scrivere una relazione come la seguente:

$$t_{r,f} = 0{,}1025 * PW^{(1)} = x * C_L^{(1)}$$

valida sia per il tempo di salita che per il tempo di discesa, dove x è un parametro che dipende dalla R_{ON} e da altri parametri dei quali si presuppone l'invarianza in questo range di variazione della C_L e di PW. In altre parole, l'ipotesi su cui si basa tutto questo ragionamento è la proporzionalità diretta tra il t_r o t_f e la capacità di carico. Per i parametri di progetto vale quindi una relazione identica (a meno che non si voglia ottenere un rapporto diverso tra $t_{r,f}$ e PW, nel qual caso la formula andrebbe generalizzata), ovvero:

$$0{,}1025 * PW^{(2)} = x * C_L^{(2)}$$

Facendo il rapporto tra le due equazioni si ottiene:

$$C_L^{(2)} = C_L^{(1)} \frac{PW^{(2)}}{PW^{(1)}}$$

Da questa equazione si può ricavare il valore massimo di $C_L^{(2)} = 15 * 162.74 * 10^{-12}$ F ≈ 2.441 pF

Il risultato è infine arrotondato per difetto a 2.400 pF. Con capacità parassite inferiori sono ovviamente possibili trasmissioni con baud rate più grande rispetto a quello minimo. Per calcolare il baud rate massimo in base alla lunghezza del cavo si può ripetere un ragionamento analogo a quello descritto, fissando la capacità parassita e procedendo a ritroso.

Per le applicazioni richieste non è necessario utilizzare un cavo molto lungo e non è necessario trasmettere con baud rate troppo elevati, e, di conseguenza, il buffer non è compreso.

[5] Il baud rate è la grandezza che misura il numero di simboli trasmessi ogni secondo in un sistema digitale. L'unità di misura di questa grandezza è il "Baud". Non è equivalente al bit-rate in quanto non necessariamente ad 1 bit corrisponde 1 baud.

Alla luce di tutte queste considerazioni, lo schema del modulo hardware è illustrato in *figura 3.3*:

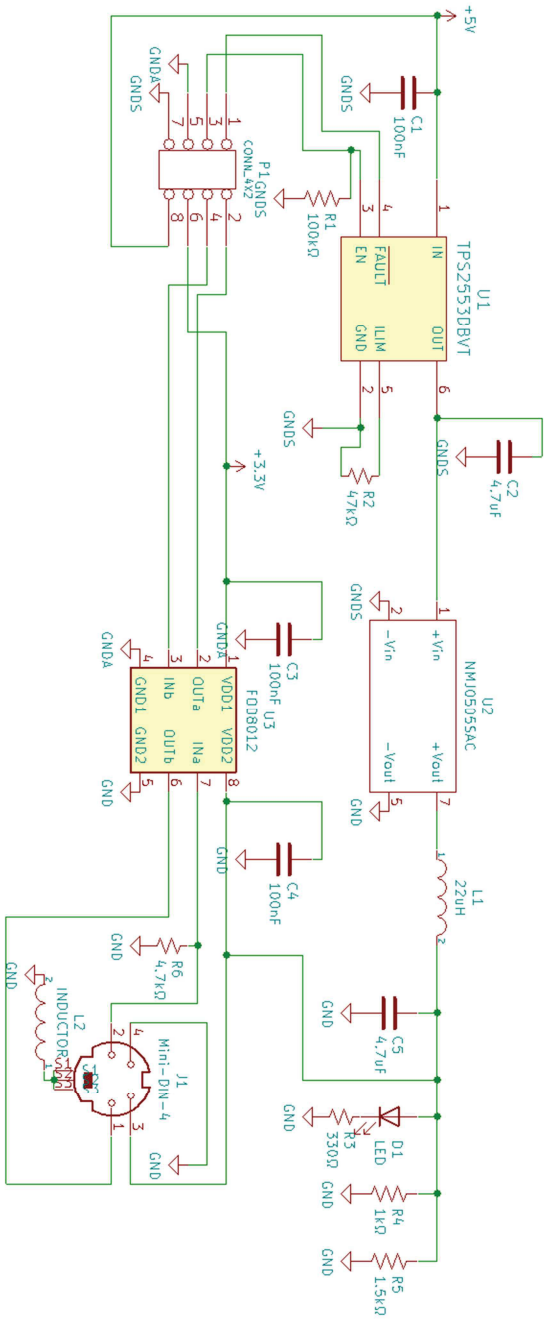


Figura 3.3

Realizzazione del PCB

Il circuito stampato è realizzato su due strati, fronte e retro, ha uno spessore di 1,6 mm e dimensioni compatibili con gli altri moduli hardware. Per evitare problemi legati ad un eventuale sovradimensionamento dovuto alle tolleranze del costruttore le dimensioni sono leggermente ridotte rispetto a quanto si evince dalle dimensioni degli altri PCB. Il modulo è quindi un rettangolo di 50,8 mm x 24,4 mm.

Nella parte frontale del PCB sono posti tutti i componenti descritti precedentemente, nel retro invece sono situati i piani di massa. La parte frontale del circuito stampato è mostrata in *figura 3.4*:

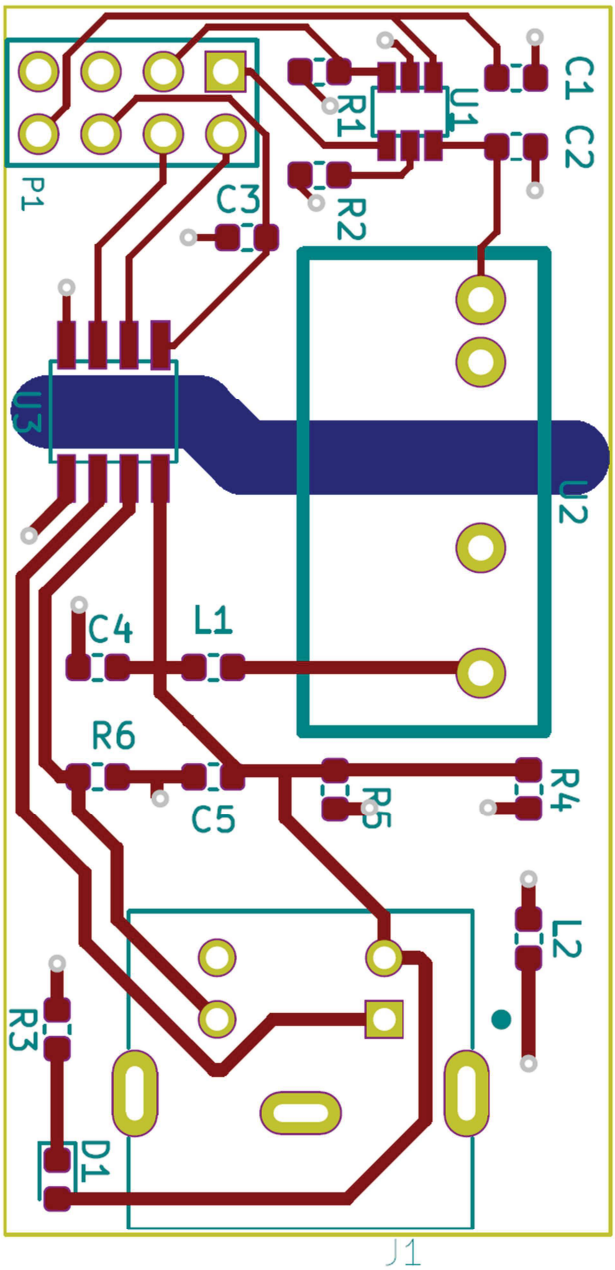


Figura 3.4

La parte in alto è quella che si interfaccia con il microcontrollore del nodo attraverso il connettore P1, posto in alto a sinistra. I centri dei fori relativi al connettore P1 sono posti ad una distanza di 2,54 mm rispetto ai fori adiacenti. Lo switch ed i relativi condensatori C1 e C2 sono posizionati vicini in accordo a quanto viene consigliato nel datasheet del componente. La larghezza delle piste in questa zona del modulo hardware corrisponde a 0,254 mm. La scelta è stata fatta in base alle indicazioni specificate nello standard IPC-2221[6]. In particolare la formula utilizzata per ottenere la corrente massima in funzione della larghezza delle piste è la seguente:

$$I = K * dT^{0,44} * (W * H)^{0,725}$$

I → Corrente massima (in A).

K → Parametro adimensionale che vale 0,024 per piste interne e 0,048 per piste esterne.

dT → Incremento di temperatura rispetto a quella ambientale (in °C o in K). Per il calcolo effettuato è ritenuto accettabile un incremento di temperatura pari a 2°.

W, H → Larghezza e spessore in mils (millesimi di pollice, sottomultiplo dello "yard", unità di misura del Sistema Imperiale detto più comunemente Sistema Anglosassone, e non appartenente al Sistema Internazionale). La H è stata fissata convenzionalmente a 0,035 mm, corrispondente ad 1,378 mils.

La formula ha validità entro certi limiti: le correnti esterne devono essere inferiori ai 35 A e le correnti interne devono essere inferiori ai 17,5 A, l'incremento di temperatura non può essere superiore ai 100 °C e le larghezze delle piste non possono andare oltre i 400 mil (10 mm). Tutti i limiti qui illustrati sono soddisfatti.

Realizzando piste di questa larghezza il limite massimo di corrente viene quindi posto a circa 0,2 A, più che sufficiente per gli scopi del progetto. Correnti più alte provocheranno un riscaldamento maggiore delle piste coinvolte nella conduzione.

Nella parte inferiore dell'immagine è invece illustrata la parte di circuito che si interfaccia con i relè attraverso il cavo che viene connesso al connettore mini-din, che in figura è denominato J1. In questa zona del circuito la larghezza delle piste è raddoppiata e vale 0,5 mm, anche se le correnti in gioco in condizioni normali non dovrebbero rendere comunque necessario questo accorgimento. I resistori R4 ed R5 sono posizionati ad una certa distanza tra di loro e ad una certa distanza da tutti gli altri componenti dato che, come si è già ampiamente detto, possono essere soggetti ad un surriscaldamento abbastanza elevato. Il LED è posizionato in un angolo, in maniera tale da essere più facilmente accessibile dall'operatore.

Resta ora da esaminare la parte posteriore del circuito stampato, che contiene i tre piani di massa di cui si è parlato nella descrizione dello schema elettrico. In *figura 3.5* è illustrato il layout di questo strato del circuito.

[6] L'IPC-2221 è uno standard che si propone di stabilire i requisiti generici per la progettazione dei circuiti stampati o di altre forme di strutture di interconnessione di questo genere.

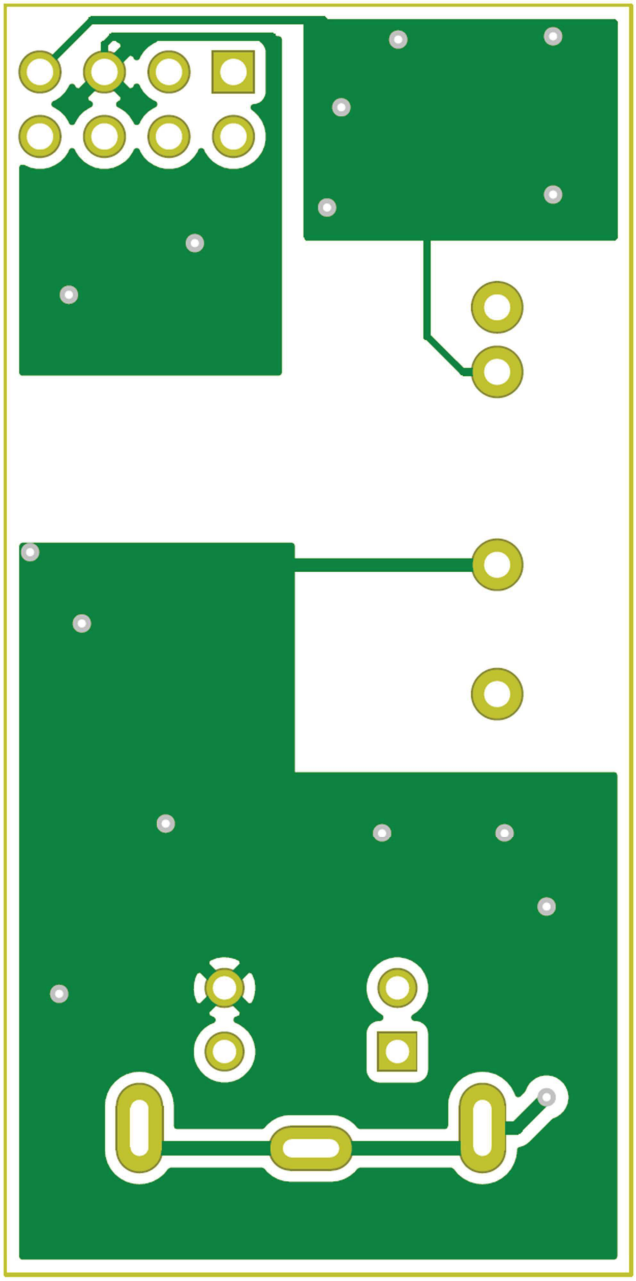


Figura 3.5

I due piani di massa relativi alla parte di circuito che si interfaccia con il nodo generico sono quelli in alto. I collegamenti a GNDA sono realizzati nel piano di massa a sinistra, mentre quelli relativi a GNDS sono realizzati nel piano di massa a destra. Nella parte bassa, invece, c'è il piano di massa denominato GND, molto più grande dei primi due. Al suo interno si può distinguere una pista non collegata al piano di massa che è relativa ai pin di shield del connettore mini-din. La pista, larga 0,5 mm, come tutte le piste presenti su questo lato del circuito, è collegata all'impronta dell'induttore L2 che si trova sull'altra faccia del circuito stampato, e poi torna a collegarsi al piano di massa. Tutte le vie (i fori colorati in grigio) sono i collegamenti a massa dei vari componenti situati sul piano superiore del circuito stampato. Tutti i piani di massa sono posti ad una certa distanza tra di

loro, in particolare i due piani relativi alla parte di segnale sono ben distanti dalla parte di potenza del PCB.

È possibile ed interessante, mediante rendering[7], avere una visuale in 3D del circuito stampato progettato. Alcune immagini tratte dal risultato del render sono le *figure 3.6a, 3.6b* e *3.6c*. Più specificamente, nella *figura 3.6a* è illustrato il lato frontale del PCB, nella *figura 3.6b* è illustrato il retro del PCB e nella *figura 3.6c* è mostrata un'immagine che permette di visualizzare lo spessore del circuito stampato.

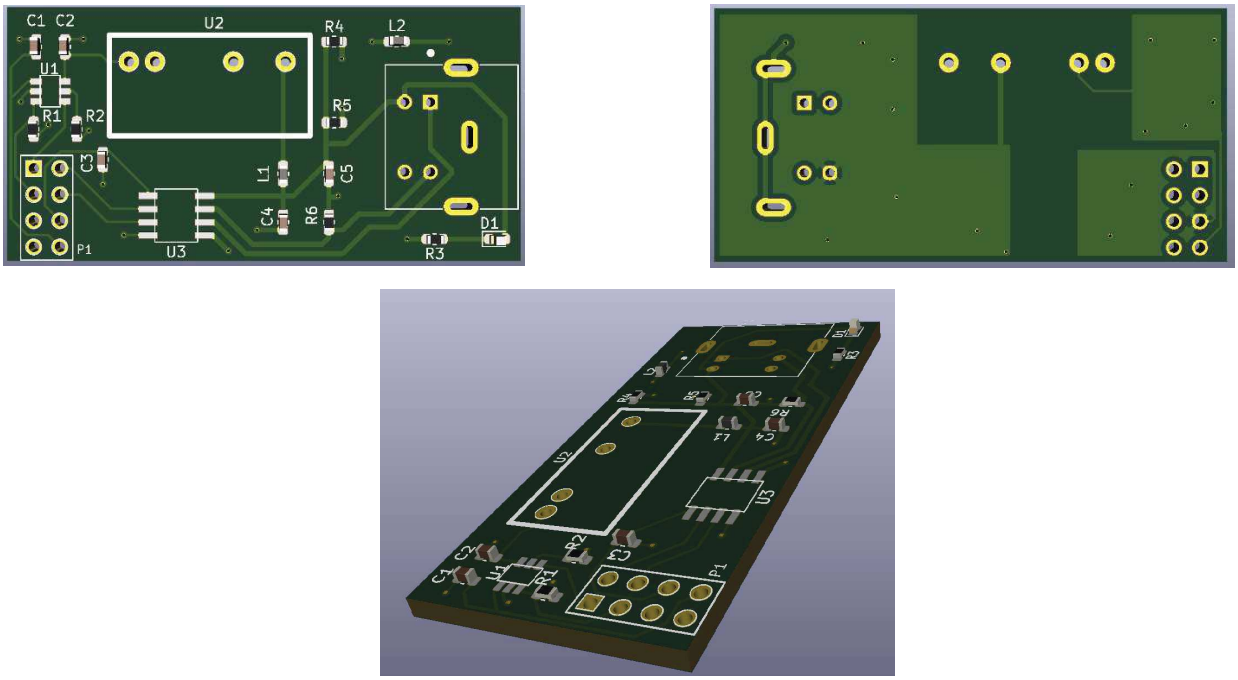


Figure 3.6a, 3.6b e 3.6c

È possibile notare che, nella parte di potenza del circuito stampato, la distanza minima tra le piste è maggiore, pari a 0,38 mm. Il dielettrico dovrebbe offrire un isolamento pari ad alcune migliaia di V_{DC}/mm, valore che assicura un buon funzionamento del circuito anche durante i transitori.

[7] Il rendering è un processo facente parte della computer grafica che permette di generare un'immagine in 3D a partire da una descrizione matematica di una scena tridimensionale. In KiCad è compresa la funzione che permette di "renderizzare" i circuiti stampati creati. Il software di modellazione utilizzato è Wings3D.

Riferimenti

ADB Pinout: https://old.pinouts.ru/Slots/apple_adb_pinout.shtml (2019)

Alessandrini, M., Biagetti, G., Crippa, P. *et al.* Optimizing linear routing in the ToLHnet protocol to improve performance over long RS-485 buses. *J Embedded Systems* 2017, 7 (2017). https://doi.org/10.1186/s13639-016-0042-x

Biagetti, G., Crippa, P., Curzi, A., Orcioni, S. and Turchetti, C., "ToLHnet: A low-complexity protocol for mixed wired and wireless low-rate control networks," *2014 6th European Embedded Design in Education and Research Conference (EDERC)*, Milano, 2014, pp. 177-181.

IPC -2221A. Generic Standard on Printed Board Desing: http://www.ipc.org/TOC/IPC-2221.pdf (2003)

Standard HAT's requirements: https://www.raspberrypi.org/blog/introducing-raspberry-pi-hats/ (2014)

[1] Immagine tratta da: https://link.springer.com/article/10.1186/s13639-016-0042-x/figures/1?shared-article-renderer

[2] Immagine tratta da: https://link.springer.com/article/10.1186/s13639-016-0042-x/figures/2?shared-article-renderer

[3] Immagine tratta da: https://link.springer.com/article/10.1186/s13639-016-0042-x/figures/3?shared-article-renderer

[4] Immagine tratta da: https://link.springer.com/article/10.1186/s13639-016-0042-x/figures/4?shared-article-renderer