PROGETTO CSD

Sistema di elaborazione per gestione della climatizzazione

Realizzato da:

- Stefano Marano
- Paolo Russo
- Vito Romano
- Marco Cimmino

1 Introduzione

In questo progetto, ci proponiamo di sviluppare un sistema che consenta di gestire la temperatura in diverse zone, utilizzando un nodo master e vari nodi slave.

L'obiettivo principale del progetto è quello di creare un sistema efficace ed efficiente che permetta di mantenere una temperatura ottimale all'interno di ogni zona, aumentando il comfort degli utenti e riducendo il consumo energetico.

Il nodo master sarà responsabile della gestione del sistema nel suo complesso, infatti sarà in grado di raccogliere i dati relativi alla temperatura all'interno di ciascuna zona tramite i nodi slave e di regolarne i parametri secondo le esigenze specifiche. Sarà anche in grado di gestire diverse **modalità** per il controllo della temperatura, ad esempio: un modalità dedicata al mantenimento di una certa temperatura oppure una modalità automatica 'eco'.

I nodi slave, invece, saranno posizionati in ciascuna zona e saranno dotati di sensori che rileveranno la temperatura ambiente e invieranno i dati al nodo master. Riceveranno anche le istruzioni dal utente per regolare la temperatura in base alle necessità.

Il sistema di elaborazione si baserà su una rete di comunicazione tra il nodo master e i nodi slave. Questo consentirà la trasmissione dei dati in tempo reale e la possibilità di ricevere le istruzioni in modo tempestivo.

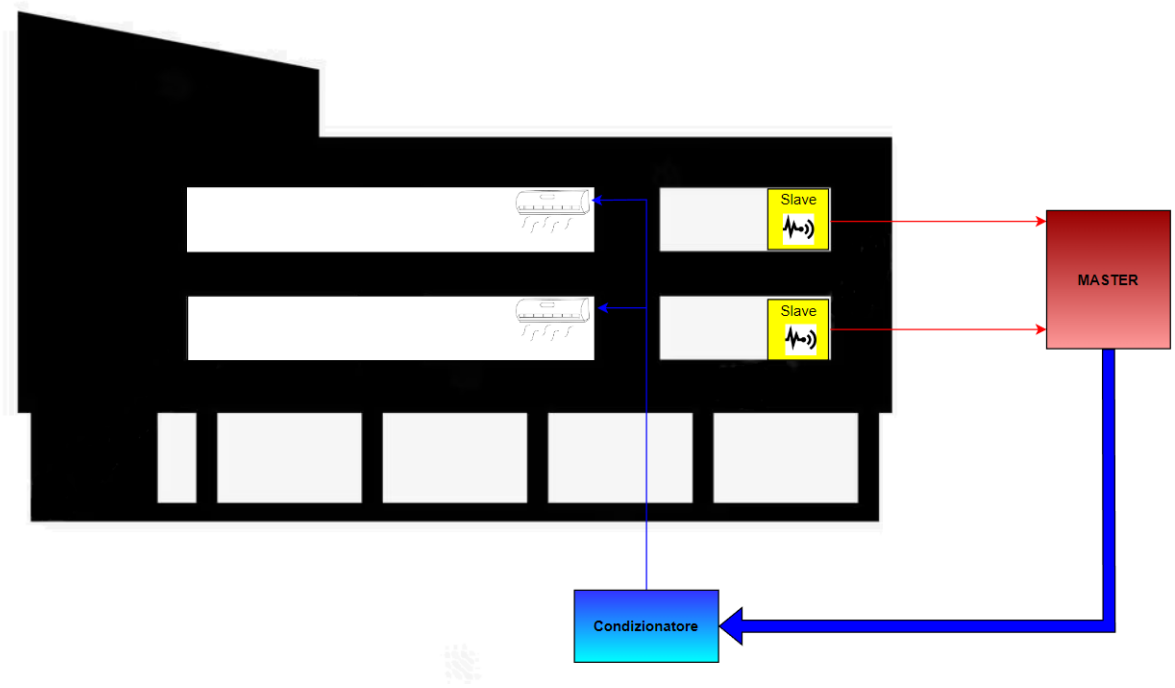

Figure 1: Schema introduttivo al progetto

2 Lettura dei campioni

Il DHT11 è un sensore di umidità e temperatura economico e facile da usare, comunemente utilizzato in progetti di elettronica e IoT, combina un termistore per la misurazione della temperatura e un sensore capacitivo per la misurazione dell'umidità, inoltre in uscita fornisce il dato il dato direttamente in digitale. Il DHT11 utilizza un protocollo di comunicazione proprietario su un solo filo per trasmettere i dati, la sequenza di comunicazione tra il sensore e il microcontrollore è la seguente:

1. Il microcontrollore invia un segnale di start al DHT11 mettendo il pin LOW per almeno 18 ms.
2. Il DHT11 risponde con un segnale di risposta (LOW per $80\mu s$, seguito da HIGH per $80\mu s$).
3. HT11 invia 40 bit di dati tramite il pin dei dati, la trasmissione di ogni bit è caratterizzato dalla forma d'onda nella figura 3

Per utilizzare il DHT11 con stm32 abbiamo utilizzato una libreria, che presenta dei metodi che implementano il comportamento appena descritto.

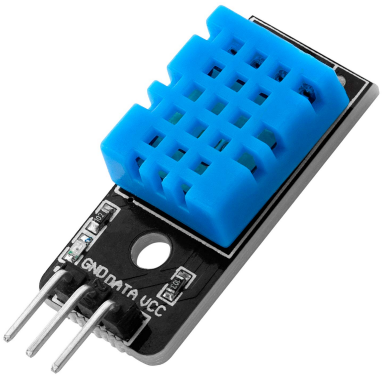

Figure 2: Sensore DHT11

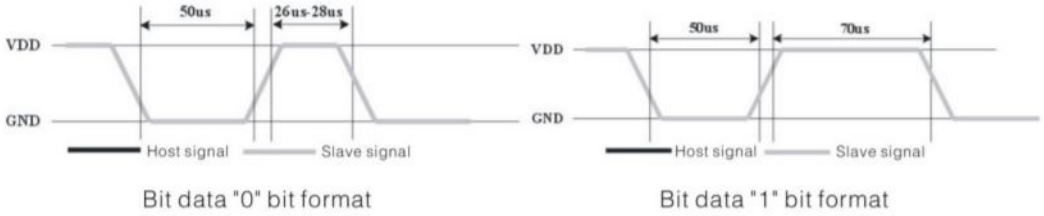

Figure 3: Data format del sensore DHT11

3 Comunicazione tra nodi

La comunicazione in questo sistema avviene tramite protocollo USART: in ogni zona è presente un nodo slave collegato al sistema principale. Per ogni slave è presente una porta USART dedicata che permette la trasmissione e la ricezione di dati tra i dispositivi.

3.1 Comunicazione Master-Slave

Per facilitare il testing e la realizzazione di un prototipo, è stato utilizzato un modulo Bluetooth basato sulla porta USART. Questo modulo consente di rendere wireless la comunicazione tra i dispositivi, migliorando la flessibilità e la facilità di connessione.

Il Bluetooth HC-05 permette di trasformare una porta UART-USART in una porta Bluetooth, generalmente con profilo SPP(Serial Port Profile), diventando cosi una seriale over Bluetooth. Abbiamo scelto questo modulo perché è uno dei moduli più popolari e poco costosi utilizzati per le comunicazioni, inoltre ha una portata di 10 mt e si imposta facilmente tramite comandi AT ed è programmabile sia come master che come slave.

La configurazione base di trasmissione vede coinvolti due dispositivi, uno assume il ruolo di master e l'altro di slave. Il master ha il compito di gestire la comunicazione (inizio, sincronizzazione, termine) mentre lo slave si limita a seguire le istruzioni del master. Un master può collegarsi con più slave, dando così origine ad una configurazione detta piconet, tali possono essere collegate tra loro per formare una rete chiamata scatternet.

Nel nostro caso si è deciso di introdurre due moduli in modalità Master nel unità centrale per controllare al più 14 slave, in questo modo è stato possibile controllare e realizzare un sistema scalabile e piuttosto modulare. Il modulo Bluetooth HC-05 ha due diverse modalità di funzionamento:

- In modalità **dati** per trasmettere o ricevere dati a/da un altro modulo/dispositivo Bluetooth.
- In modalità **comando AT**, quando è necessario modificare alcune delle configurazioni predefinite ed i relativi valori. Con qualsiasi software per terminale seriale è possibile ad esempio settare il modulo in modalità master ed effettuare il binding con uno slave, in questo modo quando saranno attivi s'interfacceranno automaticamente.

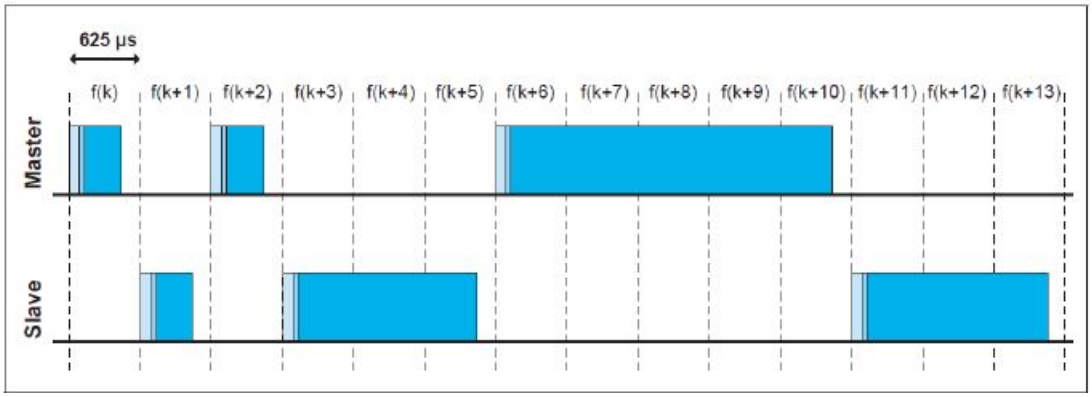

Figure 4: Comunicazione Master-Slave con HC-05

I canali solo suddivisi in slots da $625\mu s$ e ciascun slot è riservato al master o allo slave alternativamente.

Nella fase di ricerca e connessione sono coinvolti due dispositivi e lo scambio di pacchetti in queste fasi avviene nei canali inquiry scan e page scan, che utilizzano le stesse frequenze ed hanno un periodo di hopping di $325\mu s$. Il pacchetto utilizzato nella fase di ricerca è detto ID (identity message) ed è composto da 68 bits contenenti informazioni per la sincronizzazione.

Se la ricerca ha successo, i dispositivi si preparano per la connessione ed il master trasmette un pacchetto, detto FHS, che contiene l'indirizzo di dispositivo e il clock del master. Con l'invio del pacchetto FHS, i dispositivi possono finalmente sincronizzarsi e connettersi.

In generale, il master può trasmettere due pacchetti ogni slot (su canali diversi) ma la risposta dello slave sarà inviata dopo $625\mu s$ dall'invio del pacchetto da parte del master. Nella fase di trasmissione i dispositivi master e slave utilizzano i canali basic piconet o adapted piconet che utilizzano le stesse 79 frequenze determinate a livello fisico. Sia il master che lo slave possono inviare al massimo un pacchetto per slot ma l'invio di un pacchetto può richiedere anche più slots fino ad un massimo di cinque.

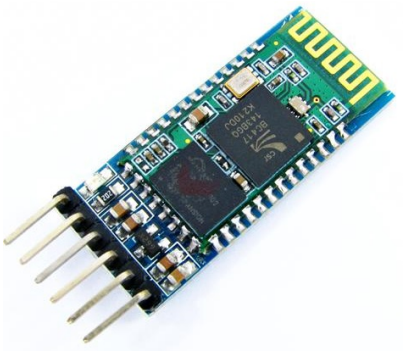

Figure 5: Modulo HC-05

3.2 Comunicazione Master - Sistema

Figure 6: Modulo CH340.

Per emulare l'interfacciamento con il condizionatore, abbiamo deciso di utilizzare un pc, dove visualizzare gli andamenti con una piccola applicazione python.

Per far ciò abbiamo utilizzato il CH340G, un chip convertitore seriale USB-TTL, esso è un dispositivo economico e molto utilizzato che fornisce una soluzione semplice e affidabile per la conversione di segnali USB in segnali TTL (Transistor-Transistor Logic), ed è ampiamente utilizzato in una varietà di applicazioni di comunicazione seriale. Risulta compatibile con l'interfaccia USB 2.0 Full Speed e con standard USB inferiori, ma è anche compatibile con UART, infatti il dispositivo supporta comuni segnali di comunicazione UART (Universal Asynchronous Receiver/Transmitter), inclusi 5V, 3.3V e segnali invertiti, inoltre ha una vasta gamma di baud rate, da 50 bps a 2 Mbps.

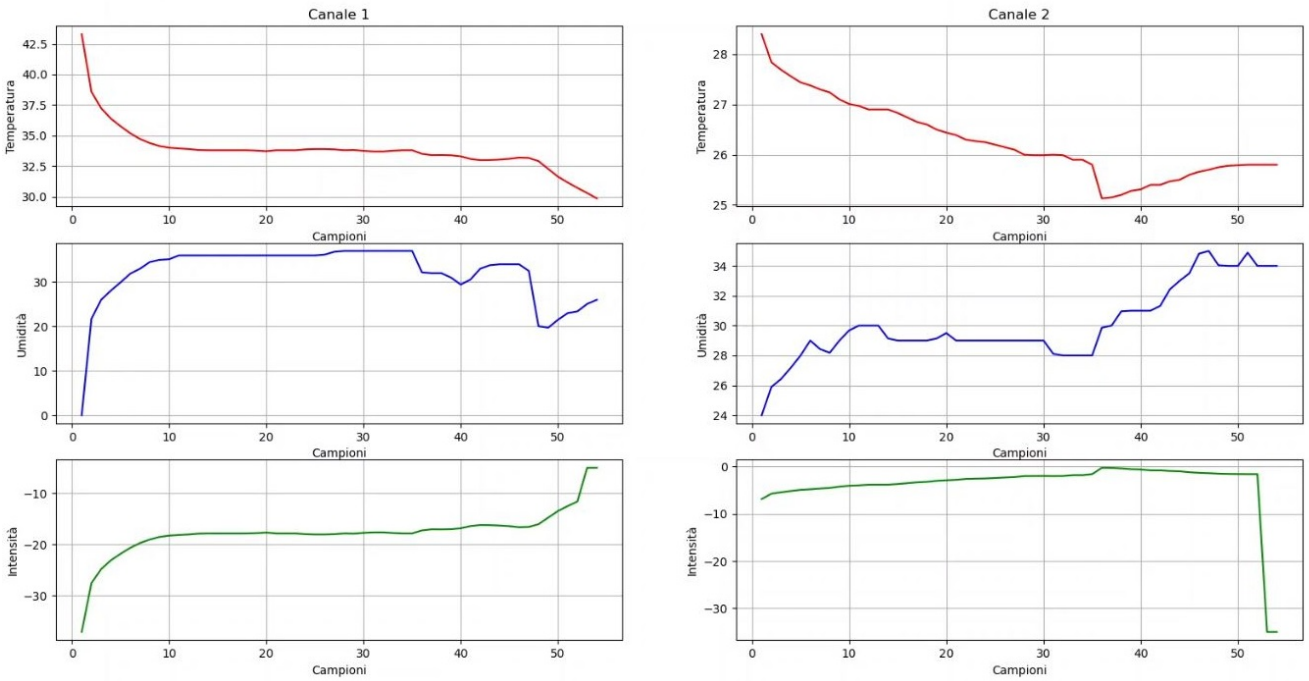

Figure 7: Interfaccia python.

4 Nodo Master

Nel contesto descritto, il nodo master possiede un insieme di isr (Interrupt Service Routine) dedicate alla ricezione dei dati dai nodi slave, ed anche di un buffer, composto da uno spazio di memoria dedicato per ogni slave, per memorizzare i dati ricevuti da ciascun di essi. Quando un dato arriva tramite un'isr, questo viene memorizzato nel buffer corrispondente. Continuativamente, il nodo master elabora i dati contenuti nel buffer per calcolare il valore di uscita desiderato in base alla modalità di funzionamento espressa dallo slave. Una volta che il valore di uscita viene calcolato, invia il valore di uscita ad un dispositivo esterno, in teoria il climatizzatore, nel nostro caso per motivi di test una semplice USART.

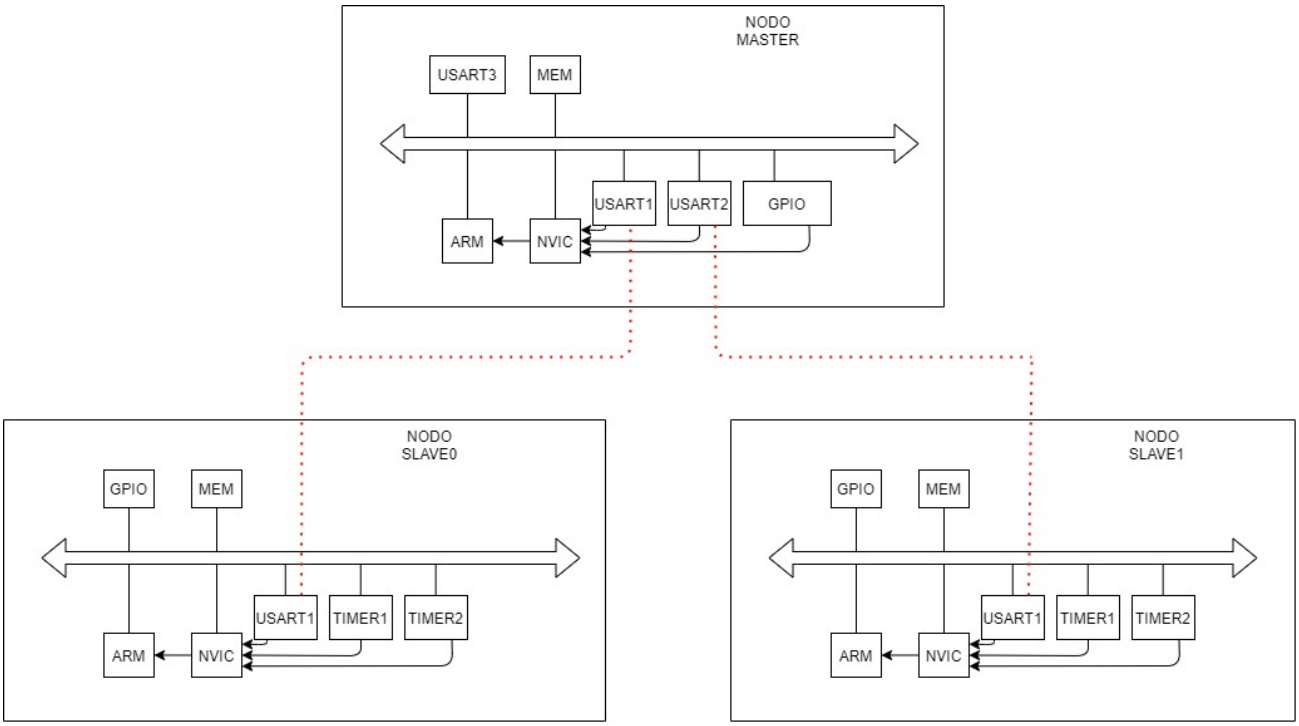

Figure 8: Configurazione di sistema.

4.1 Modalità di Funzionamento

Inizialmente il sistema può innescare una fase di INIT, dove ogni slave effettua velocemente delle catture e invia un valore medio al master, in questo modo si può avere velocemente un valore della temperatura ed iniziare prontamente l'esecuzione; Tale modalità è innescata utilizzato il tasto fornito dallo schedino.

L'esecuzione standard prevede invece due modalità di funzionamento: la prima modalità è basata su una logica **PID** (Proporzionale-Integrativo-Differenziale), che utilizza valori di temperatura e umidità nel tempo per mantenere le condizioni desiderate nella zona gestita. La logica PID calcola un valore di controllo in base alla differenza tra la misura attuale e il setpoint desiderato, e lo utilizza per regolare l'attuatore responsabile del riscaldamento o del raffreddamento. Questo tipo di controllo è molto preciso e permette di mantenere una temperatura e un'umidità stabili nel tempo.

Temp Umid	freddo	moderato	caldo
bassa	alta	media	bassa
media	media	media	media
alta	bassa	bassa	alta

Table 1: Tabella della logica Fuzzy.

La seconda modalità di funzionamento del sistema utilizza una logica **fuzzy** per realizzare una modalità automatica che mira a mantenere il comfort nella zona gestita, garantendo al contempo un risparmio energetico ottimale. La logica fuzzy considera una serie di variabili di input e assegna loro un grado di appartenenza a diverse categorie (ad esempio: freddo, fresco, mite, caldo, molto caldo). In base a queste categorie e a una serie di regole predefinite, la logica fuzzy determina i valori di controllo per l'attuatore, adattando il sistema in modo intelligente al fine di ottimizzare il comfort e il consumo energetico.

Il sistema è in grado di gestire contemporaneamente entrambe le modalità di funzionamento, consentendo una gestione personalizzata di ogni zona. Ciò significa che è possibile impostare la modalità di mantenimento basata su logica PID per alcune zone che richiedono una temperatura e un'umidità costanti, come ad esempio una sala server, mentre è possibile utilizzare la modalità basata su logica fuzzy per altre zone, come un'area di lavoro con uffici, in modo da adattarsi in modo intelligente alle preferenze degli occupanti e ai risparmi energetici desiderati.

5 Nodo Slave

Il nodo slave è basato su ISR per la lettura di campioni di temperatura e umidità che raccoglie dai sensori, li trasmette al nodo master e comunica la modalità di funzionamento e il parametro desiderato di temperatura al master.

5.1 Comportamento del sistema

Sono definite due fasi per lo slave, una viene imposta dal master, mentre l'altra rappresenta il normale comportamento del nodo.

Nel primo caso, la cattura della temperatura avviene durante la fase di **Init**, ovvero durante l'avvio del sistema di elaborazione. Durante questa fase, vengono effettuate numerose catture della temperatura in breve tempo e successivamente processate per calcolare una media troncata, ovvero si tolgono eventuali valori anomali o estremi per ottenere una stima più accurata della temperatura effettiva dell'ambiente in un breve periodo.

Nella seconda fase, utilizzata durante il normale funzionamento del sistema, la temperatura viene letta ogni 20 secondi. Per ogni lettura, viene valutato se si è verificata una variazione sensibile della temperatura rispetto alla lettura precedente e se è rilevante, viene inviata l'informazione al master del sistema per aggiornare l'andamento del climatizzatore. Da notare che dopo un tempo prestabilito senza variazioni viene comunque inviato al master l'andamento in modo da mantenere stabile e aggiornato il sistema.

Questo permette al sistema di monitorare costantemente la temperatura e prendere le opportune azioni nel caso in cui si verifichi un cambiamento significativo che richieda una risposta immediata.

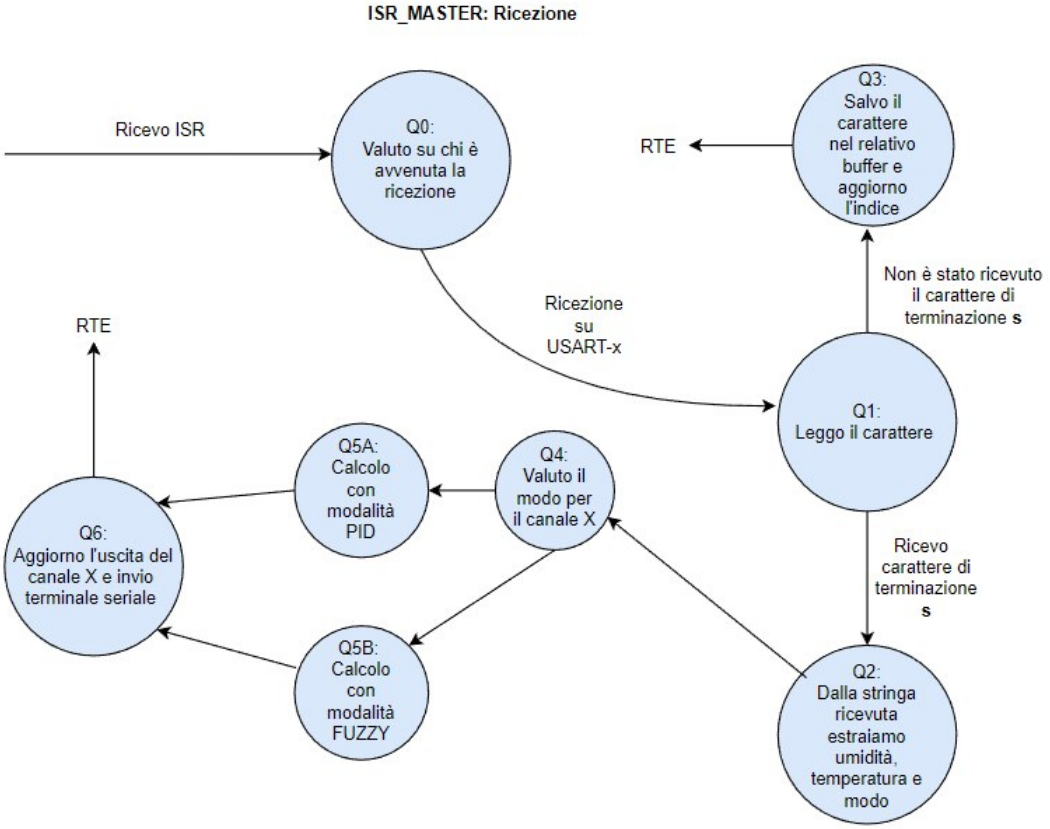

Figure 9: ISR di ricezione sul master.

6 Reference

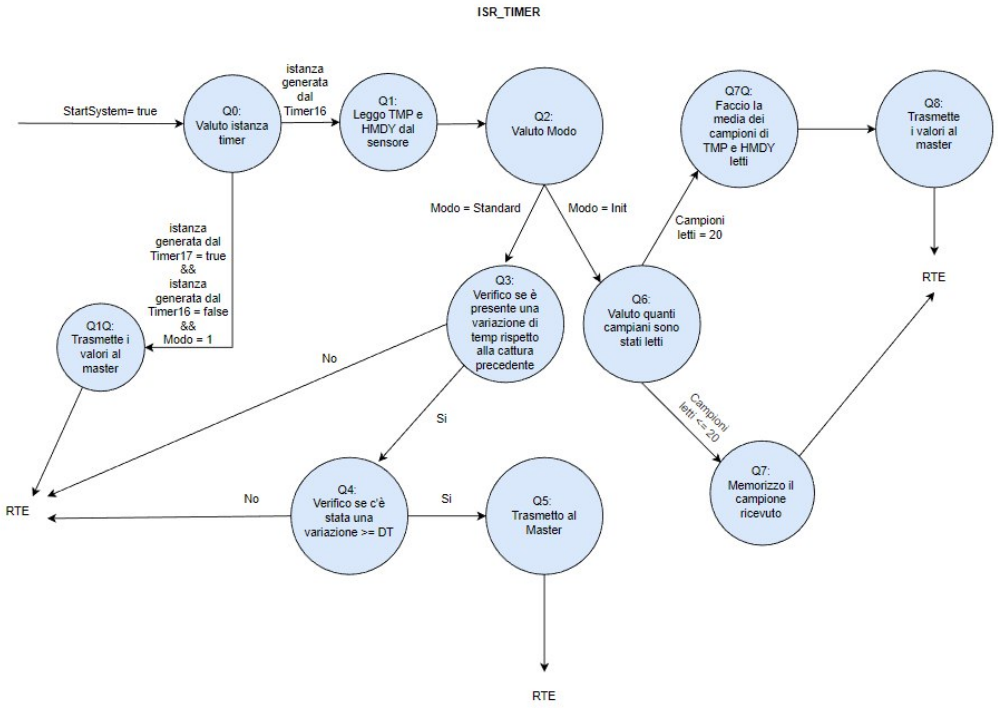

Figure 10: ISR dello slave per le letture.

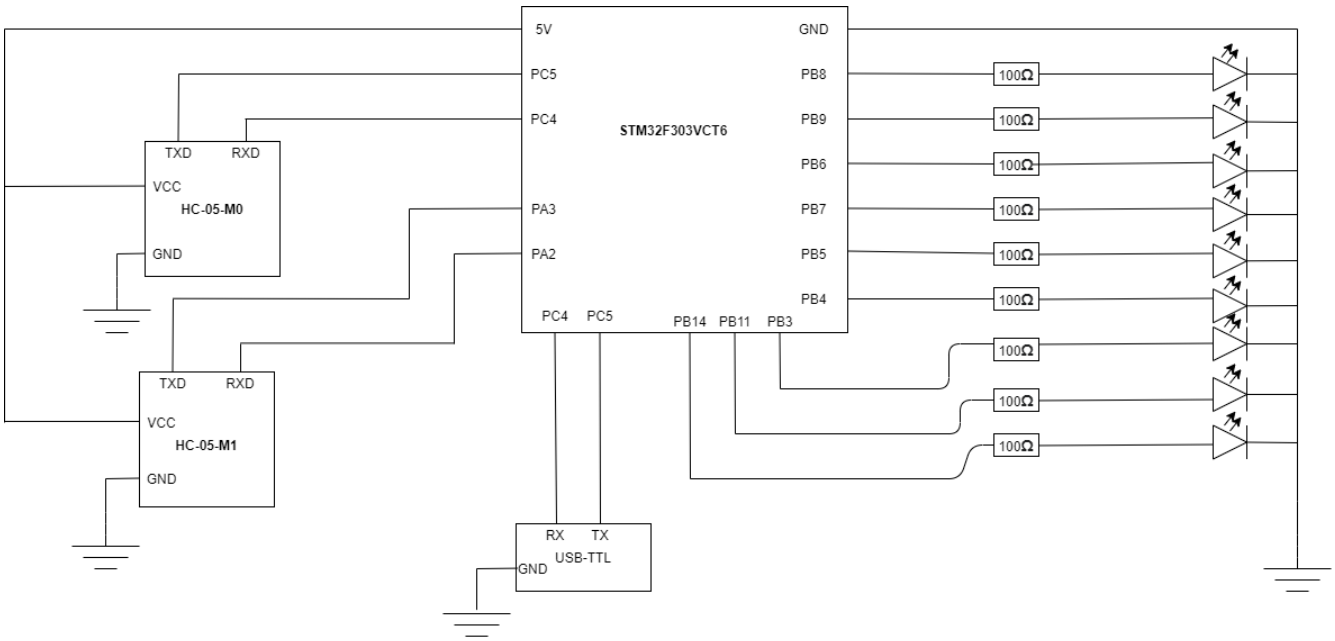

Figure 11: Schema dei collegamenti del master

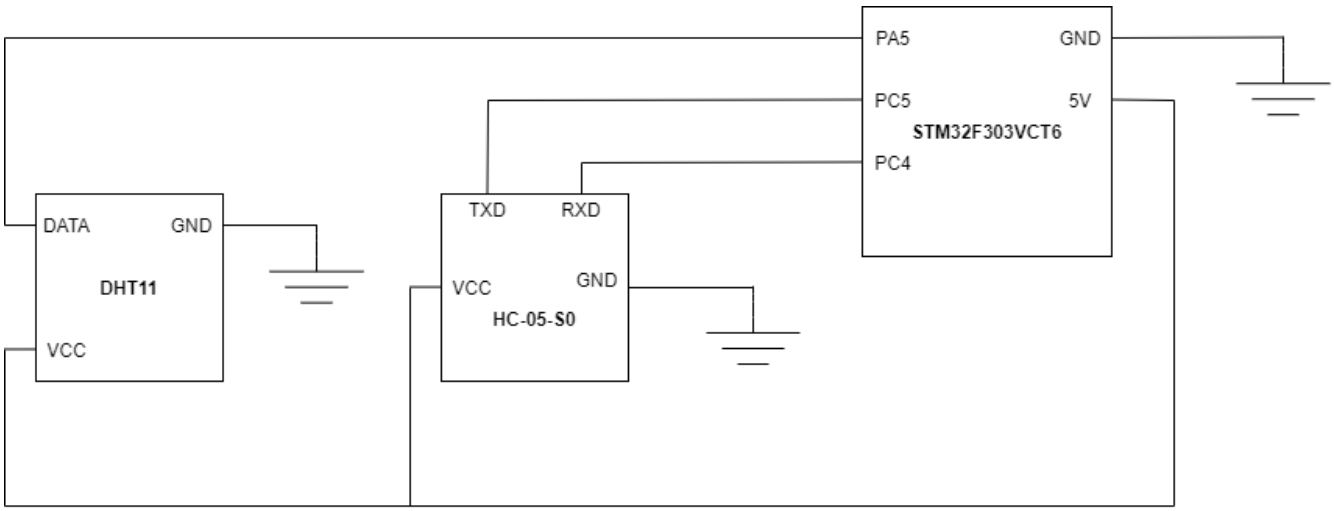

Figure 12: Schema dei collegamenti dello slave