

UNIVERSITÀ DEGLI STUDI DI BRESCIA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea Magistrale in Ingegneria Elettronica



Elettronica per Strumentazione, Sensori e Microsistemi

Datalogger e lettura tramite NFC

Professori:

prof. Vittorio Ferrari
prof. Marco Ferrari

Studenti:

Giulio Lucchi 86484
Eugenio Enrico Mainetti Gambera 86097

ANNO ACCADEMICO 2019/2020

Indice

1	Riassunto	1
2	Abstract	2
3	introduzione	3
4	Descrizione del sistema	5
4.1	Schema a blocchi	5
4.1.1	Microcontrollore	6
4.1.2	NFC	6
4.1.3	EEPROM e I2C	8
4.1.4	Sensore di temperatura e umidità	9
4.1.5	Circuito del microcontrollore	12
4.2	Software	14
4.2.1	Start Signal	14
4.2.2	Check Response	15
4.2.3	Read Data	15
4.2.4	I2C init	15
4.2.5	EEPageByteWrite	15
4.2.6	Cancella eeprom	16
4.2.7	Main	16
4.2.8	Interruzione	16
4.2.9	Software di lettura	17

INDICE

4.3 Utilizzo del datalogger	18
5 Risultati	19
5.1 Misure	19
5.1.1 Uscita da ventole di raffreddamento	19
5.1.2 Phon	21
5.2 Criticità affrontate	22
6 Conclusioni	24
Bibliografia	25

Capitolo 1

Riassunto

L'obiettivo di questo progetto è realizzare un datalogger di temperatura e umidità che raccolga misure ad intervalli regolari tramite un RTCC (Real time calendar/clock). I dati così raccolti devono poter essere letti tramite un lettore NFC anche in assenza di alimentazione del sistema, normalmente alimentato a batteria. Nella prima sezione del capitolo quattro vengono presentati i componenti e i protocolli utilizzati con particolare attenzione alle caratteristiche che li rendono adatti alla realizzazione del datalogger. Viene poi mostrato lo schema elettrico del circuito complessivo.

Nella seconda sezione viene descritta nello specifico ogni funzione che compone il programma caricato sul microcontrollore.

Nella terza sezione viene riassunta la procedura da eseguire per il corretto set up del datalogger e del suo utilizzo.

Nel capitolo cinque e sei infine viene testato il datalogger tramite misure di aria calda in uscita da un portatile e da un phon. Vengono analizzati i dati così ottenuti e si discutono le principali criticità riscontrate in fase di progetto, dando anche alcuni spunti nel caso si volesse approfondire ed estendere il progetto.

Capitolo 2

Abstract

The aim of this project is to create a temperature and humidity datalogger that collects measurements at regular intervals through an RTCC (Real time calendar / clock).

Data collected in this way can be read by an NFC reader even in the absence of power from the system, normally battery-powered.

In the first section of chapter four the components and protocols used are presented with particular attention to the characteristics that make them suitable for the realization of the data logger. The overall circuit diagram is then shown.

In the second section, each function that constitutes the program loaded on the microcontroller is described in detail.

The third section summarizes the procedure to be performed for the correct set of the datalogger and its use.

Finally, in chapter five and six the datalogger is tested by means of hot air measurements coming out from a laptop and a hairdryer. The data thus collected are then analyzed jointly with the main issues encountered during the project. Some ideas are given in case you want to deepen and extend the project.

Capitolo 3

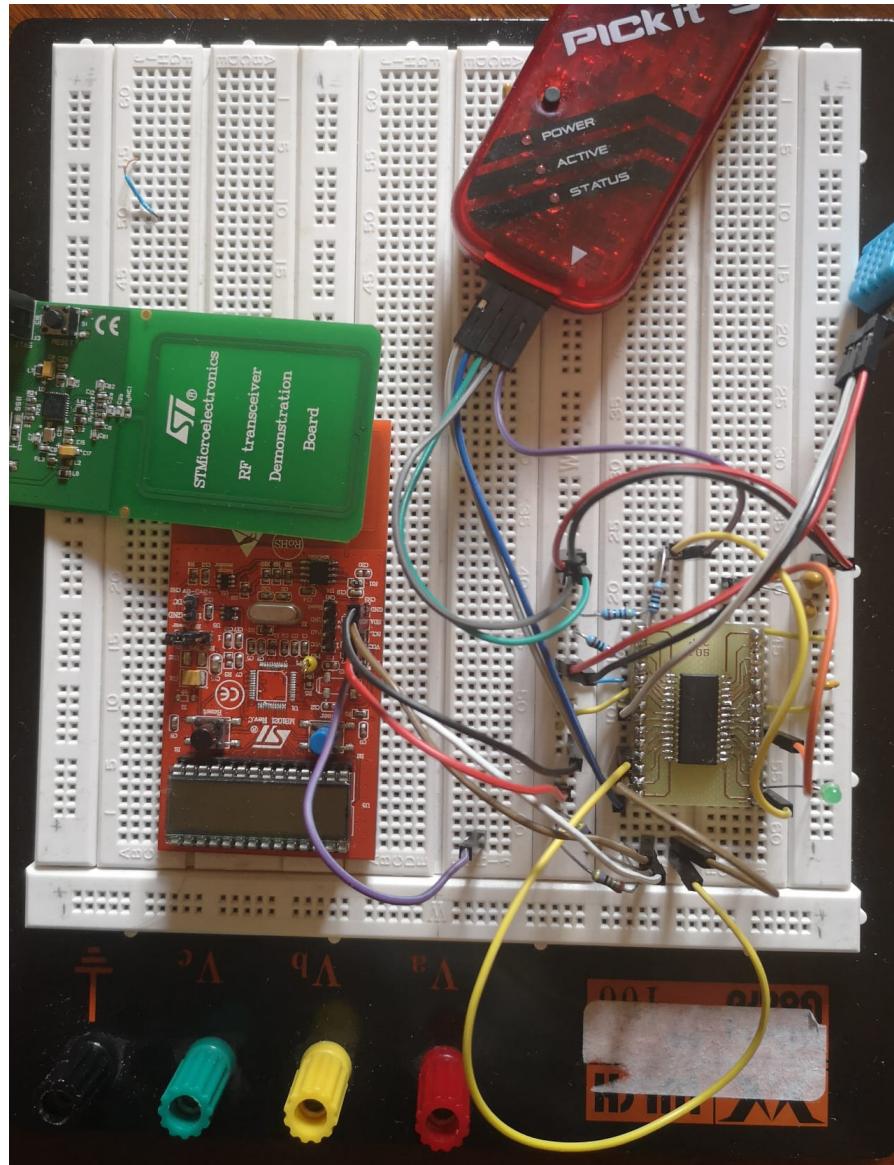
introduzione

Un datalogger, tradotto letteralmente in registratore di dati, è un dispositivo elettronico digitale, di solito di piccole dimensioni, che registra dei dati ad intervalli regolari attraverso un sensore.

In questo progetto viene usato un sensore di temperatura e umidità, una memoria in cui salvare i dati e un microcontrollore che gestisce il campionamento dei dati del sensore e la scrittura della memoria. Per quanto riguarda la lettura dei dati si utilizza un’interfaccia NFC che permette la lettura wireless della memoria anche in assenza di alimentazione del datalogger.

L’importanza dello sviluppo di sistemi di datalogging wireless spaziano dalle applicazioni in campo biomedico per il monitoraggio di dati biometrici o della salute degli alimenti [2] fino al controllo di dati ambientali per la prevenzione di incidenti [1]. Visti i campi applicativi dei datalogger la scelta dei componenti è principalmente legata al consumo di energia. Si utilizza quindi un PIC low power e una metodologia di lettura che non richiede alimentazione da parte della batteria.

CAPITOLO 3. INTRODUZIONE



Capitolo 4

Descrizione del sistema

4.1 Schema a blocchi

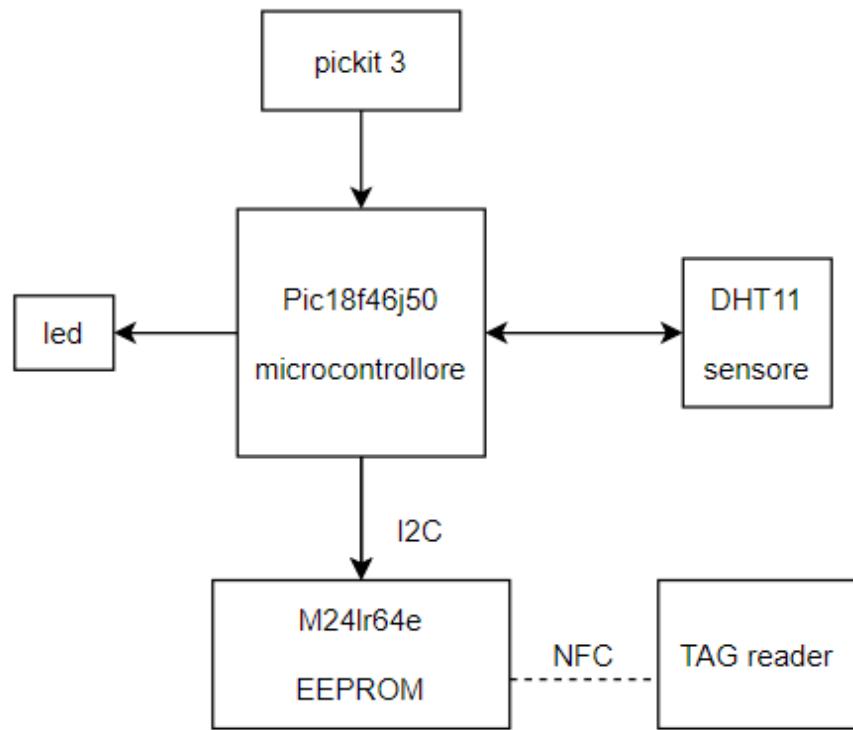


Figura 4.1: Schema a blocchi del sistema

4.1.1 Microcontrollore

Il cuore esecutivo del sistema è rappresentato dal microcontrollore pic18f26j50. Questo microcontrollore vanta tra le sue caratteristiche una bassa corrente di funzionamento e una tensione di alimentazione tra i 2 V e i 3.6 V, un RTCC (real time calendar clock) dotato di interrupt, un oscillatore interno a 8 MHz, alleggerendo così la necessità di un riferimento esterno per il clock, una periferica per I2C e dei pin IO digitali che tollerano fino ai 5.5 V. Il microcontrollore è il responsabile delle comunicazioni con il sensore di temperatura e umidità e della scrittura dell'EEPROM, e durante l'utilizzo normale viene alimentato a batteria.

Il pic18f26j50 è stato programmato da un pickit3 che oltre alle funzioni di programmazione, in fase di progetto ha svolto anche la funzione di debugging e di alimentazione per il circuito sostituendosi alla batteria.

4.1.2 NFC

La comunicazione in prossimità, anche chiamata near-field communication (NFC), è una tecnologia di ricetrasmissione che fornisce connettività senza fili (RF) bidirezionale a corto raggio. Nasce come naturale evoluzione del più datato RFID (radio frequency identification) tecnologia utilizzata soprattutto in ambito industriale come un'etichetta elettronica per fornire informazioni su oggetti e animali, che ha il limite di essere una comunicazione a senso unico.

Le comunicazioni nfc comunemente consistono in un initiator e un target. L'initiator crea un campo RF che alimenta il target, questo permette a quest'ultimo di avere dei form factor molto semplici come etichette e carte.

Fisicamente l'initiator e il target hanno delle antenne ad anello (loop antenna) che si accoppiano induttivamente creando un vero e proprio trasformatore in aria. Ciò è possibile perché le distanze fra le antenne sono nell'ordine dei centimetri e la lunghezza d'onda λ alla frequenza di 13,56 MHz vale:

$$c = \lambda * f \implies \lambda = \frac{3 * 10^8}{13,56 * 10^6} = 22,16 \text{ m} \quad (4.1)$$

CAPITOLO 4. DESCRIZIONE DEL SISTEMA

che è di circa tre ordini di grandezza superiore. Si puo' quindi considerare l'accoppiamento come puramente magnetico (near field) e si può affermare che solo una parte trascurabile di potenza viene irradiata sotto forma di onde elettromagnetiche.

Questo è importante non solo dal punto di vista energetico ma anche per le interferenze con onde radio o altri dispositivi nfc.

Lo standard di comunicazione NFC utilizzato da questo sistema è l'ISO15693 che prevede una portante di $13.56\text{ MHz} \pm 7\text{ kHz}$ e fino 53 kbit/sec di trasmissione dati tramite i comandi fast. La comunicazione NFC da computer fino all'EEPROM viene mediata tramite il reader CR95HF che converte i comandi impartiti via usb dal pc in comandi NFC. La trama di dati di comunicazione nel caso di lettura avviene tramite una stringa di dati che viene mandata dal reader ed è composta come in fig.4.2 in cui il primo byte è dedicato alla configurazione della metodologia

Request SOF	Request_flags	Read Multiple Block	UID ⁽¹⁾	First block number	Number of blocks	CRC16	Request EOF
-	8 bits	23h	64 bits	16 bits	8 bits	16 bits	-

Figura 4.2: Trama del comando READ

di comunicazione, il seguente byte è il comando effettivo di READ (in questo caso è 0x23), gli 8 byte successivi sono opzionali e servono nel caso si voglia interrogare un tag specifico con un ID identificativo proprio, onde evitare comunicazioni con dispositivi errati, i 2 byte successivi identificano l'indirizzo del primo blocco di byte che si vuole leggere (un blocco in questo caso si riferisce a 4 byte) seguito da un byte per identificare quanti blocchi di fila si vogliono leggere, con un massimo di 32 blocchi letti rapidamente in sequenza, gli ultimi 16 bit sono un cyclic redundancy check, ovvero controllo di ridondanza ciclico, un metodo di checksum per controllare l'integrità della comunicazione. In seguito a questo comando il tag risponde con una trama simile come raffigurata in fig.4.3 in cui le sostanziali differenze sono la presenza di un eventuale byte dedicato alla risposta sul security status della memoria e 32 bit di memoria che vengono ripetuti in base al numero di

blocchi che sono stati richiesti, fino ad un massimo di 32. Nel nostro caso specifico

Response SOF	Response_flags	Sector security status ⁽¹⁾	Data	CRC16	Response EOF
-	8 bits	8 bits ⁽²⁾	32 bits ⁽²⁾	16 bits	-

Figura 4.3: Trama di risposta del comando READ

la lettura tramite NFC deve essere eseguita con una certa prossimità e oltre i 3.5 cm il sistema non riesce ad effettuare nessuna operazione e non riconosce il tag.

4.1.3 EEPROM e I2C

EEPROM

La memoria utilizzata M24LR64E è fornita di 8192 byte divisi in 2048 blocchi da 4 byte, un'interfaccia contactless compatibile con ISO15693 e ISO 18000-3 a frequenza 13.56 MHz, utilizzata in questo caso per leggere i dati, e un'interfaccia I2C che collegata al pic permette la scrittura delle misure campionate dal sensore.

I2C

L'I2C è un protocollo seriale che utilizza due fili utilizzato per trasmettere informazioni tra due o più dispositivi. È un protocollo bidirezionale e half duplex.

Nonostante la sua semplicità una serie di vantaggi lo ha portato a essere supportato da tutti i maggiori produttori di circuiti integrati:

- si può comporre una rete i2c con più master e slave
- a seconda della modalità utilizzata può arrivare a una velocità di trasferimento unidirezionale di 5 Mbit/sec
- il fatto che sia una comunicazione a due fili,(teoricamente tre per il riferimento comune ai dispositivi spesso messo a massa), è molto importante

nel mondo dei sistemi embedded sempre in ricerca di un maggior numero di ingressi e uscite.

- è un protocollo sincrono, questo gli permette di trovare applicazioni in un gran numero di situazioni in cui è richiesto un soft real time.

I due fili utilizzati per la comunicazione vengono chiamati SDA (serial data line) e SCL (serial clock line). La transizione da alto a basso di SDA mentre SCL rimane alto determina l'inizio della comunicazione. Dopo la start condition il master manda l'indirizzo del dispositivo con il quale vuole interfacciarsi e un bit per definire se vuole leggere o scrivere. Lo slave che riconosce il suo indirizzo manda un Ack al master e si inizia a trasmettere un bit alla volta sulla linea SDA mandando un Ack ogni byte, da master a slave in caso di scrittura o viceversa in lettura. L'invio dei bit corrisponde temporalmente a quando SCL ha un livello basso. Quando l'operazione è finita il master manda la condizione di stop che consiste nella transizione da basso a alto sulla linea SDA mentre SCL rimane alto.

4.1.4 Sensore di temperatura e umidità

Il sensore di temperatura e umidità che è stato usato è il modulo DHT11 [4]. Il DHT11 è un sensore con un'uscita digitale calibrata e sfrutta una comunicazione tramite single-wire seriale.

Il sensore è presente su una scheda con inclusa una resistenza da 5k di pullup per la comunicazione fino a 20 mt e un led di segnalazione dell'alimentazione, che per questo chip può oscillare tra i 3 V e i 5.5 V .

La comunicazione avviene tramite il pin 23 del microcontrollore con una trama di dati (fig.4.5) ben precisa che è stata implementata via software senza l'utilizzo di librerie esterne, in cui il micro si comporta sia da trasmettitore che da ricevitore. Grazie a questa comunicazione si ottengono 40 bit divisi in 5 byte: 8bit parte intera RelativeHum data + 8bit parte decimale RelativeHum data + 8bit parte intera Temp data + 8bit parte decimale Temp data + 8bit checksum. Da notare che il modulo DHT11, nonostante la presenza di byte legati alla parte decimale,

CAPITOLO 4. DESCRIZIONE DEL SISTEMA

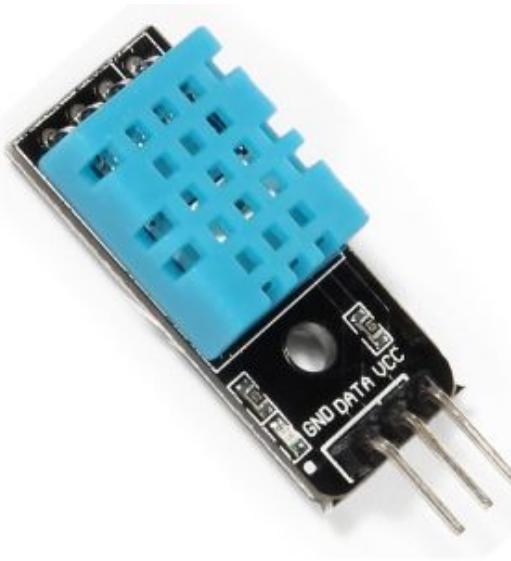


Figura 4.4: DHT11

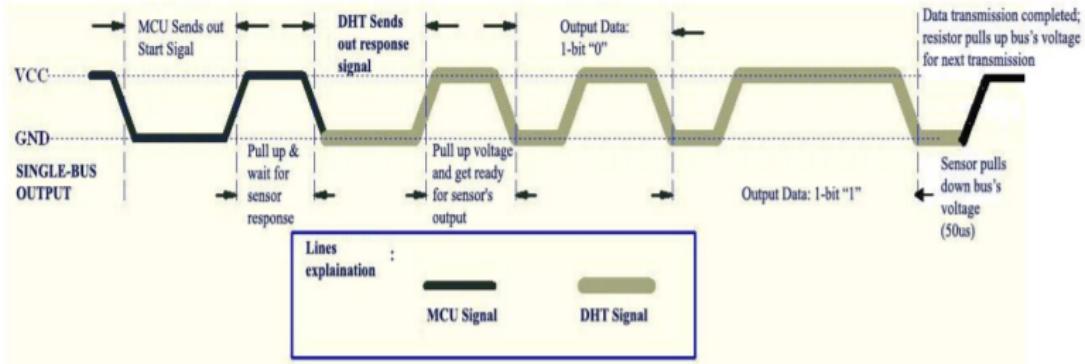


Figura 4.5: trama di comunicazione DHT11

ha precisione solo sulla parte intera e i byte sulla parte decimale risultano 0x00 per scelta costruttiva. Tuttavia vanno letti in ordine per ottenere le informazioni necessarie, questo perchè il DHT11 è imparentato con la versione superiore DHTT22 di cui condivide l'architettura ma che ha la possibilità di eseguire misure anche sulla parte decimale della temperatura. Bisogna inoltre notare che i dati di umidità, a differenza di quelli di temperatura che vengono trasmessi a partire dallo 0, vengono comunicati considerando un minimo di 20% che deve essere sommato al dato trasmesso.

CAPITOLO 4. DESCRIZIONE DEL SISTEMA

Parameters	Conditions	Minimum	Typical	Maximum
Humidity				
Resolution		1%RH	1%RH	1%RH
		8 Bit		
Repeatability			±1%RH	
Accuracy	25°C		±4%RH	
	0-50°C			±5%RH
Interchangeability	Fully Interchangeable			
Measurement Range	0°C	30%RH		90%RH
	25°C	20%RH		90%RH
	50°C	20%RH		80%RH
Response Time (Seconds)	1/e(63%)25°C , 1m/s Air	6 S	10 S	15 S
Hysteresis			±1%RH	
Long-Term Stability	Typical		±1%RH/year	
Temperature				
Resolution		1°C	1°C	1°C
		8 Bit	8 Bit	8 Bit
Repeatability			±1°C	
Accuracy		±1°C		±2°C
Measurement Range		0°C		50°C
Response Time (Seconds)	1/e(63%)	6 S		30 S

Figura 4.6: valori DHT11

4.1.5 Circuito del microcontrollore

Circuito del microcontrollore con tag esterno

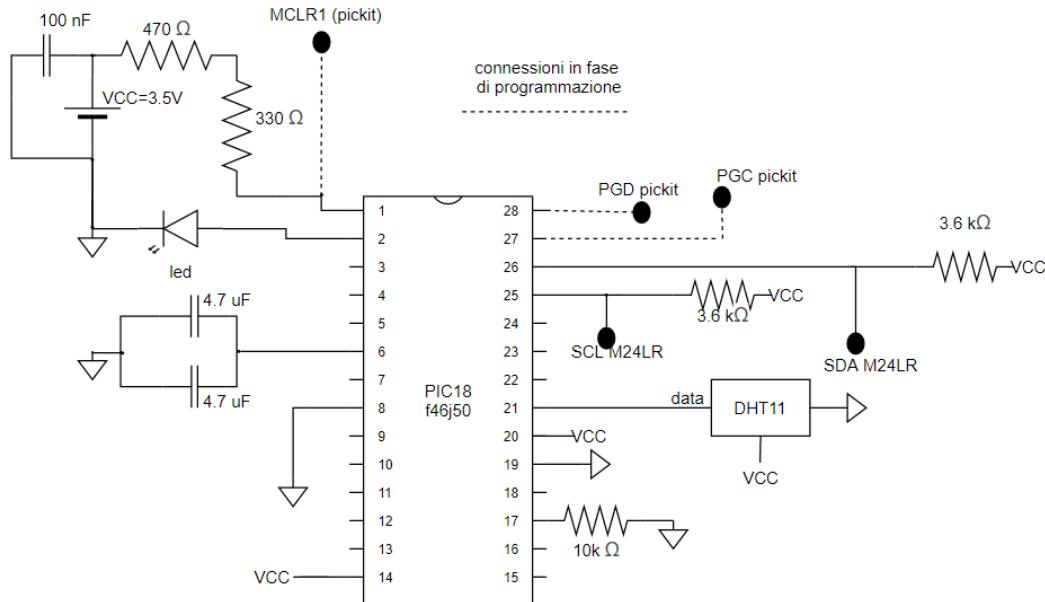


Figura 4.7: Circuito complessivo del microcontrollore

Il circuito utilizzato (fig.4.7) serve per l'interazione del microcontrollore con il sensore e con la linea I₂C, per cui un collegamento diretto non è possibile e servono dei resistori di pull up da 3.6 K. I restanti componenti servono principalmente per il corretto funzionamento del micro. Sono stati disposti inoltre condensatori da 100 nF tra alimentazione e massa nei punti prossimi all'alimentazione del PIC. Il circuito è stato realizzato e testato su una breadboard come si può vedere in fig.3.1, inoltre è stata progettata una scheda tramite Eagle per racchiudere in una scheda tutti i collegamenti del microcontrollore lasciando esterni il sensore e il tag.

CAPITOLO 4. DESCRIZIONE DEL SISTEMA

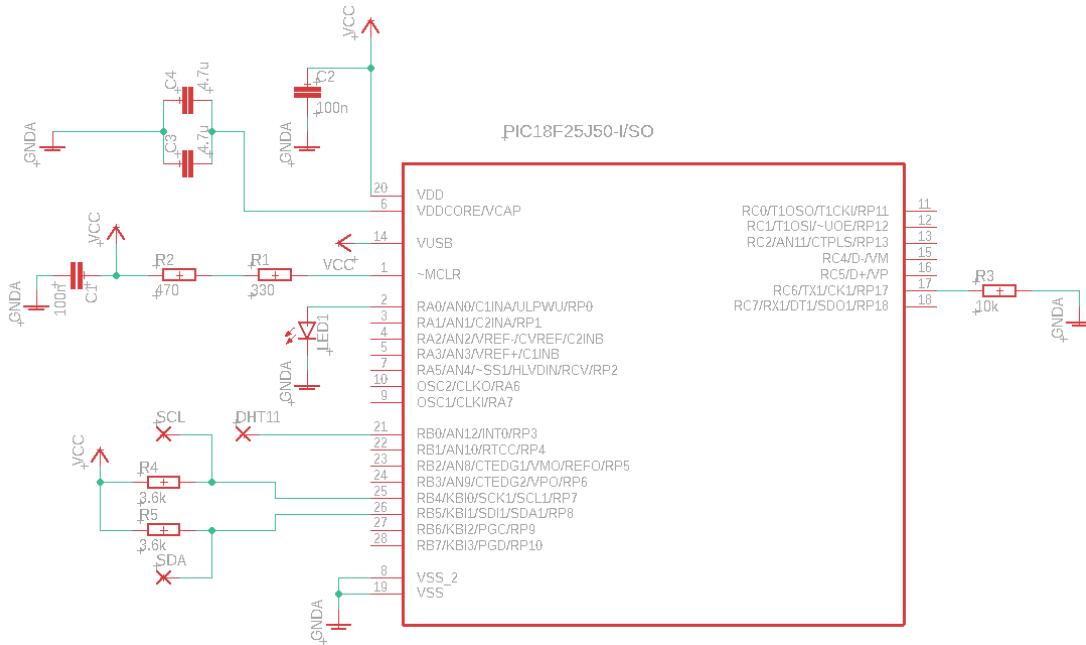


Figura 4.8: Circuito disegnato su Eagle

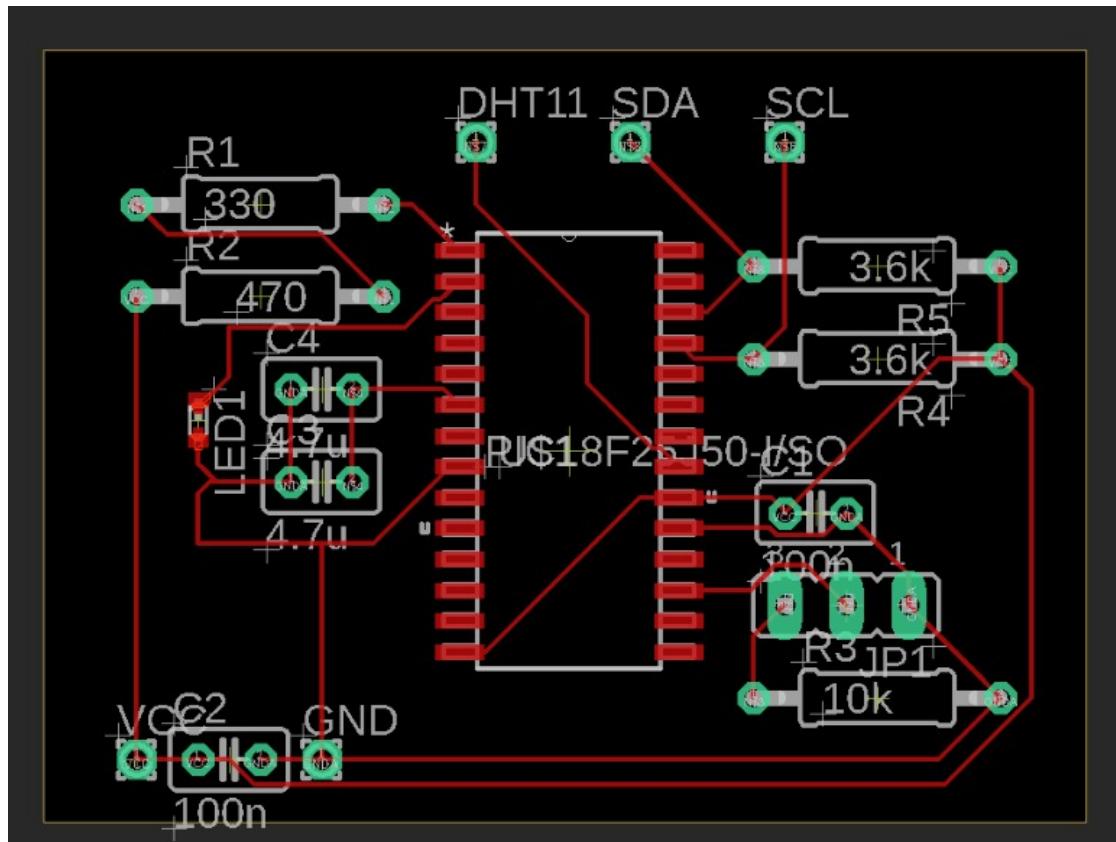


Figura 4.9: circuito stampato estratto da Eagle

PCB del microcontrollore con tag integrato

Per approfondire l'argomento progettazione su scheda è stato progettato un PCB che racchiudesse in maniera complessiva il circuito, lasciando esterno solo il sensore di temperatura in modo tale da essere collocato nel punto più opportuno. La progettazione dell'antenna è basato su un design commerciale con i disegni di layout liberamente disponibili [3].

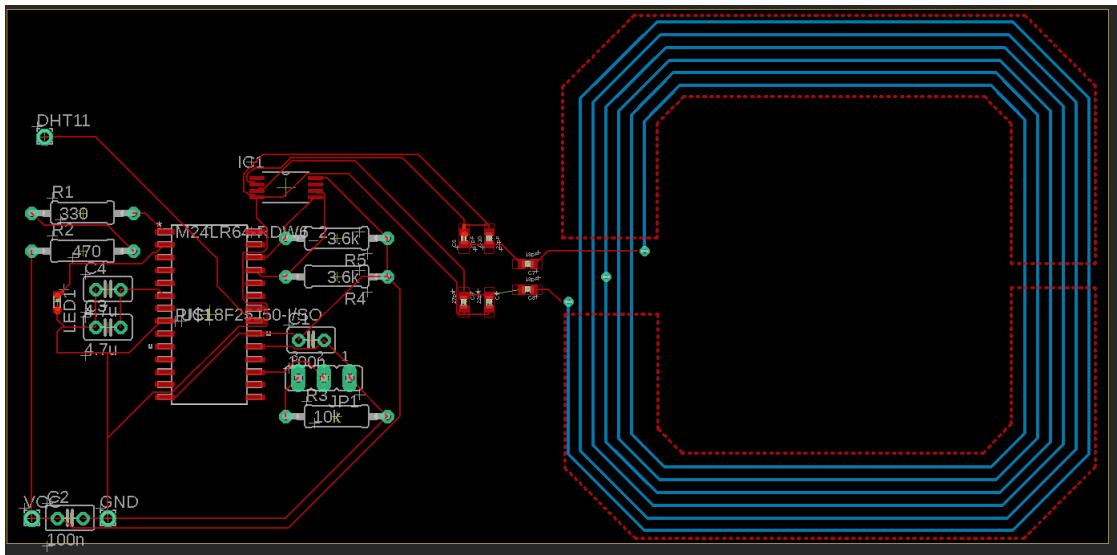


Figura 4.10: circuito stampato con antenna

4.2 Software

La parte principale del progetto si racchiude nel codice che è stato sviluppato per il PIC. Questo programma, sviluppato in c per il compilatore XC8, si compone di un singolo file in cui sono state scritte le funzioni principali, l'interrupt del RTCC e la funzione main.

4.2.1 Start Signal

Questa funzione viene richiamata per configurare l'inizio della comunicazione con il DHT11, mandando un segnale alto per 25 ms e mettendosi in attesa dopo aver impostato il pin di comunicazione in ricezione.

4.2.2 Check Response

Questa funzione ha il compito di verificare che il canale di comunicazione con il DHT11 sia stabile aspettando che il DHT11 risponda per 80 us con un segnale basso e per 80 us con un segnale alto. Se queste condizioni sono rispettate ritorna 1, al contrario se ci sono stati errori o timeout il return segna 0.

4.2.3 Read Data

Questa funzione ha il compito di leggere un byte di comunicazione proveniente dal DHT11, dopo aver aspettato che il sengale diventi alto per circa 80 us e successivamente basso per altri 80 us inizia a contare quanto dura la successiva transizione ad un valore alto: se dura meno di 50 us vuol dire che il sensore ha mandato uno 0, durate maggiori indicano un 1. Il procedimento viene ripetuto per 8 volte per riempire il byte

4.2.4 I2C init

Questa funzione serve semplicemente per settare i 2 pin usati per la comunicazione I2C.

4.2.5 EEPAGEBYTEWRITE

Questa funzione riceve in ingresso l'id I2C del dispositivo, due indirizzi e 4 byte da scrivere. La funzione aspetta che il bus sia libero per poi gestire i registri necessari per la comunicazione. La funzione prosegue scrivendo sul bus l'id del dispositivo I2C su cui si vuole scrivere, necessario per poter indicare a quale dispositivo si sta comunicando, per poi scrivere in successione 2 byte con l'indirizzo di memoria in cui si vuole scrivere aspettando tra un byte e l'altro l'acknowledge. Successivamente vengono mandati i 4 byte (che corrispondono alla page dell'EEPROM) e si attende il segnale di stop. In un qualsiasi di questi step se sorge un errore viene fermata la scrittura e si ritorna la tipologia dell'errore.

4.2.6 Cancellare eeprom

Questa funzione viene richiamata nel caso si voglia cancellare la memoria della EEPROM ponendo tutti i byte a 0. Dopo aver settato alcuni valori per porre il PIC come master I2C fa un ciclo per 2048 volte in cui richiama la funzione EEPAGEByteWrite_nuovo e incrementa l'indirizzo di 4. Una volta finito di scrivere l'ultimo byte, cosa che può impiegare quasi due minuti, accende un led e si mette in idle.

4.2.7 Main

Il main inizia settando l'oscillatore interno di riferimento a 8 MHz, setta il timer1 mettendo come fonte di alimentazione l'instruction clock, che ha una frequenza 4 volte più lenta dell'oscillatore di riferimento, e pone il prescaler a 1:2, in questo modo il timer1 incrementa con una frequenza di 1 MHz, molto utile se si vogliono avere misure precise di us. In seguito il programma verifica se il pin 7 è a massa, in caso negativo richiama la funzione cancella_eeprom. Dopo il main inizia ad operare sul RTCC impostando il clock di riferimento e abilitando la scrittura per poterlo resettare a 0, viene anche configurato l'alarm che ci permetterà di scandire gli intervalli di datalogging, nello specifico con ALRMCFGbits.AMASK = 0b0010 si pone un intervallo di datalogging di 10 secondi, inoltre è stato scelto l'utilizzo dell'RTCC interno perchè presenta una buona affidabilità con un errore di circa 2.64 secondi per mese. Vengono attivati gli interrupt e acceso un led per poi entrare in un ciclo while in attesa dell'interrupt.

4.2.8 Interruzione

La funzione principale che scandisce il tempo di scrittura è contenuta nell'interrupt. Questo interrupt viene invocato solo in caso di allarme del RTCC e ha come scopo due cose: la prima invertire lo stato del led per segnalare l'allarme avvenuto, la seconda richiamare le varie funzioni per la comunicazione con il DHT11 per recuperare informazioni sulla temperatura e sull'umidità relativa e poi scriverle

sul bus I2C all'EEPROM. Viene effettuato inoltre un controllo sull'indirizzo raggiunto, se sono stati superati i byte scrivibili sull'eprom si ferma per un periodo indeterminato.

4.2.9 Software di lettura

Per la lettura della EEPROM è stato utilizzato il software di lettura della ST: M24LRxx application software. Per utilizzare questo programma una volta avviato bisogna selezionare la tipologia di device che si vuole usare come lettore, in questo caso il CR95HF demo kit. In seguito, dopo aver avvicinato il reader al tag si può effettuare un riconoscimento automatico della tipologia di chip utilizzato come EEPROM dal menù reader application. Una volta identificato il chip utilizzato si apre un'interfaccia che permette la comunicazione con il tag sia in lettura che in scrittura.

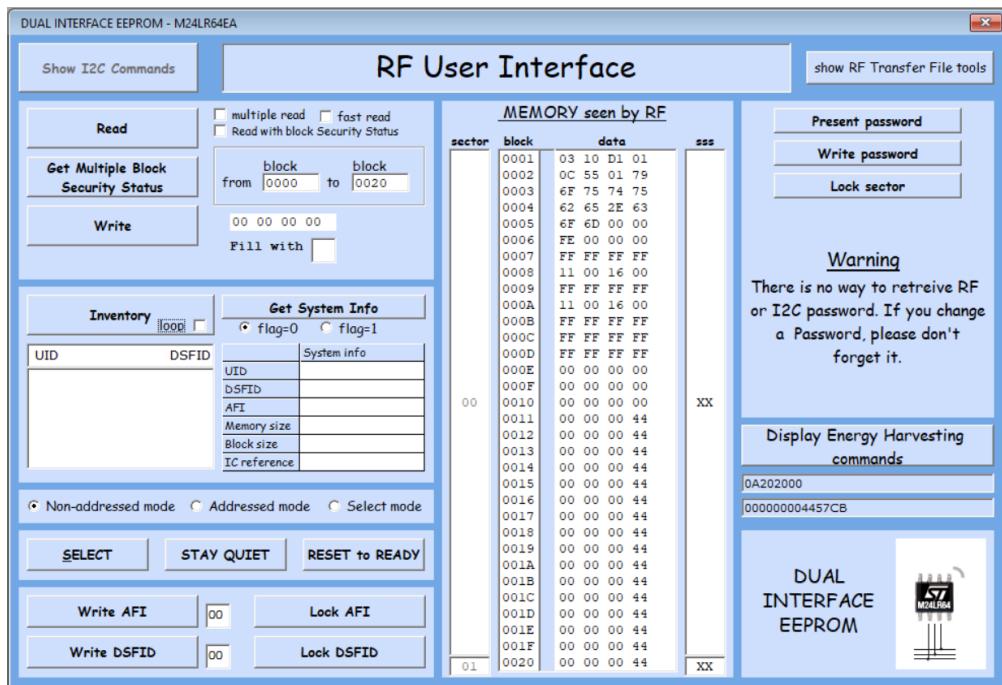


Figura 4.11: Interfaccia di lettura

4.3 Utilizzo del datalogger

L'utilizzo del datalogger prevede degli specifici passaggi, in primo luogo va spostato il terminale a massa della resistenza sul pin 17 a Vcc, in questo modo appena il micro viene alimentato entra in modalità di cancellazione e cancella l'EEPROM. Quando il led di segnalazione si accende, significa che la cancellazione della memoria è giunta a compimento, si può quindi spegnere il PIC. A questo punto bisogna riportare la resistenza a massa e rialimentare il micro, in questo modo il PIC entra in modalità datalogging e inizia a registrare a intervalli di tempo, in questo caso ogni dieci secondi, la temperatura e l'umidità relativa salvando il risultato nell'EEPROM, facendo cambiare di stato il led ad ogni scrittura. Si può leggere via NFC il tag per leggere i valori registrati anche in assenza di alimentazione del micro, tuttavia in caso di interruzione dell'alimentazione è necessario ripartire con la procedura se si vuole una serie storica dei valori sensata.

Capitolo 5

Risultati

5.1 Misure

5.1.1 Uscita da ventole di raffreddamento

Per la fase di testing il sistema è stato alimentato a batteria per circa 1 ora e il sensore di temperatura è stato avvicinato ad una fonte di aria calda da una ventola di un pc e si è effettuato un campionamento ogni 10 secondi. Dopo un'ora il microcontrollore è stato spento e si è letta la memoria del datalogger tramite il reader NFC. I dati sono stati raccolti e tramite Matlab si è creato un grafico per visionare i risultati che non si sono discostati da valori plausibili. Questo dimostra come il datalogger possa essere utile per mantenere letture di dati a lungo

CAPITOLO 5. RISULTATI

MEMORY seen by RF			
sector	block	data	sss
	0000	0F 00 19 00	
	0001	10 00 19 00	
	0002	0F 00 19 00	
	0003	10 00 19 00	
	0004	0F 00 19 00	
	0005	10 00 19 00	
	0006	0F 00 19 00	
	0007	10 00 19 00	
	0008	10 00 19 00	
	0009	10 00 19 00	
	000A	10 00 19 00	
	000B	10 00 19 00	
	000C	10 00 19 00	
	000D	11 00 19 00	
	000E	10 00 19 00	
	000F	11 00 19 00	
00	0010	10 00 19 00	XX
	0011	11 00 19 00	
	0012	10 00 19 00	
	0013	11 00 19 00	
	0014	10 00 19 00	
	0015	11 00 19 00	
	0016	10 00 19 00	
	0017	11 00 19 00	
	0018	11 00 19 00	
	0019	14 00 19 00	
	001A	15 00 1A 00	
	001B	0F 00 19 00	
	001C	0F 00 19 00	
	001D	0F 00 1A 00	
	001E	0F 00 1A 00	
	001F	0F 00 1A 00	

Figura 5.1: primi dati in ordine del datalogger

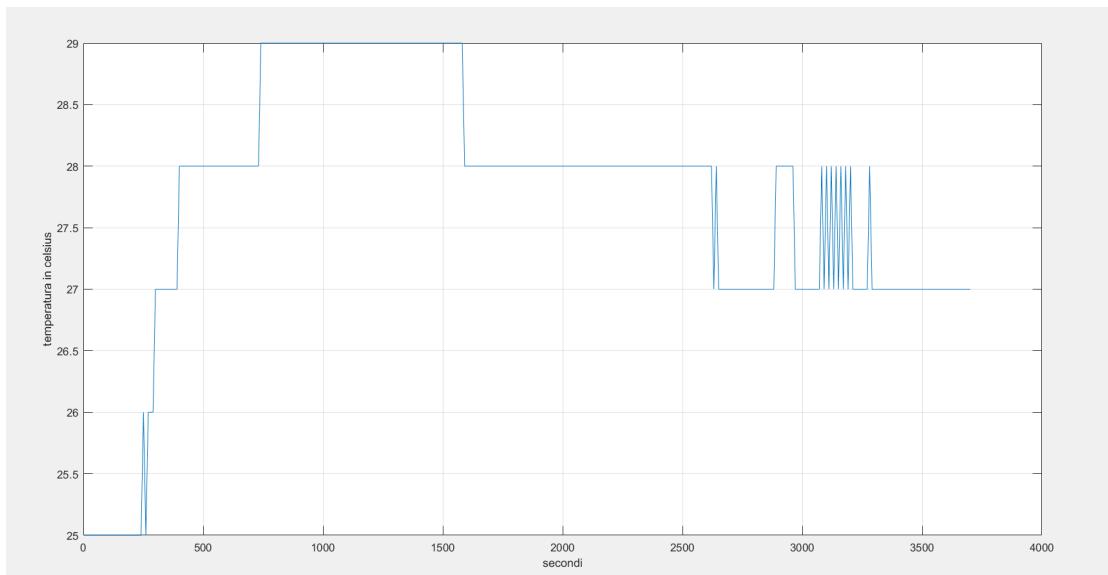


Figura 5.2: grafico della temperatura

5.1.2 Phon

Un'altra tipologia di misura più breve è stata fatta per testare i limiti di misura del sensore e la misura di umidità. In questo caso dopo aver fatto partire il sistema è stato avvicinato un phon di aria calda al sensore e lasciato operare per 30 secondi per poi lasciarlo raffreddare per un totale di quasi 10 minuti. In questo caso si può notare come in poco tempo il sensore saturi al suo valore massimo di 50 gradi e allo stesso tempo l'umidità relativa, quando è direttamente sotto il getto d'aria del phon, risulti molto bassa.

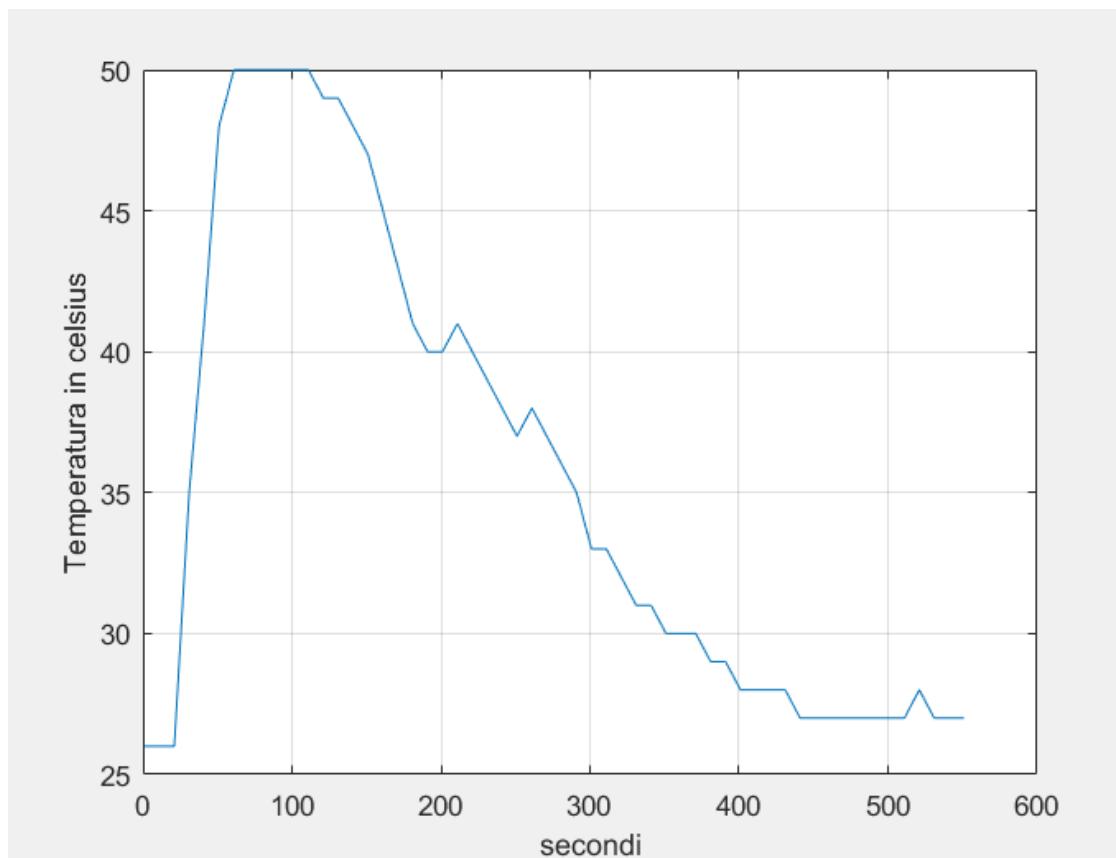


Figura 5.3: grafico della temperatura

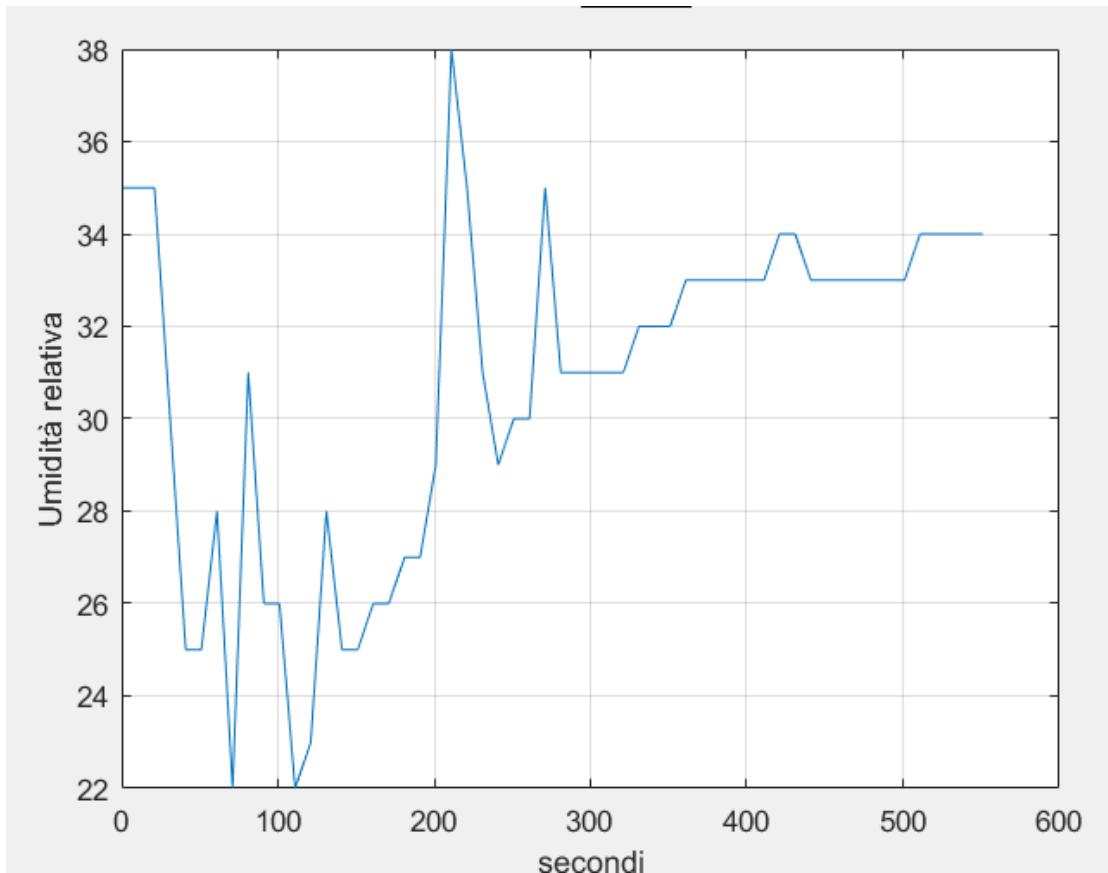


Figura 5.4: grafico della umidità relativa

5.2 Criticità affrontate

Per la creazione di un sistema stabile sono stati affrontati diversi problemi, in primo luogo pic18f26j50 non è tra i microcontrollori supportati dalle nuove caratteristiche dei plugin di configurazione di MPLAB X e quindi la scrittura del codice rispetto ad altri chip più recenti procede in maniera più lenta e tradizionale basandosi principalmente sul datasheet. Altro problema che è stato affrontato è la gestione della comunicazione con l'EEPROM, infatti nel caso di comunicazione di scrittura distanziata da almeno un secondo I2C risulta non avere particolari problemi tuttavia la problematica è sorta quando si è voluto ottimizzare il tempo di scrittura per cancellare la memoria nel tempo più breve possibile e quindi accedere ad una modalità di scrittura sequenziale rapida. In questo caso bisogna

CAPITOLO 5. RISULTATI

notare come una volta mandato il comando di write e i byte da scrivere la scrittura non è immediata e sequenziale con l'arrivo dei singoli bit ma avviene una volta ricevuto l'intero messaggio, in questo caso 4 byte, successivamente l'EEPROM entra in modalità di scrittura interna che può impiegare fino a 5 ms per byte. Per evitare questo problema dopo ogni scrittura è stato impostato un delay di 40 ms per essere sicuri che anche il messaggio successivo vada a buon fine.

Capitolo 6

Conclusioni

In questo progetto è stato realizzato un datalogger di temperatura che ha rispettato i parametri di progetto potendo eseguire senza problemi letture tramite NFC anche per un periodo di tempo esteso. Il lavoro ha visto un'analisi approfondita del materiale tecnico e lo sviluppo di un codice affidabile e utilizzo di funzioni di debug per testare l'affidabilità delle funzioni. Inoltre è stato testato il dispositivo in scenari diversi dando delle risposte in linea con le misure previsti senza mai acquisire dati nulli. Possibili sviluppi futuri vedono lo studio dei consumi del circuito per poter analizzare la durata massima di datalogging tramite batteria, la realizzazione effettiva su circuito stampato della soluzione su breadboard e lo sviluppo di software per PC specifico per la lettura, memorizzazione e gestione dei dati tramite l'utilizzo delle dll per c++ per interfacciarsi con il reader.

Bibliografia

- [1] John Wiley & Sons, Health Monitoring of Bridges (2009),
- [2] Engadget. Everyday RFID Tags Could Help Spot Food Contamination. <https://www.engadget.com/2018/11/15/rfid-food-contamination-detection-mit/>
- [3] SparkFun RFID Evaluation Shield - 13.56MHz, EAGLE FILE http://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/RFID_Eval_13.56MHz-v14.zip
- [4] Mouser technical data sheet DHT11, <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>
- [5] STMicroelectronics technical data sheet m24lr64e-r, <https://www.st.com/en/nfc/m24lr64e-r.html>
- [6] Microchip technical data sheet pic18f26j50, <http://ww1.microchip.com/downloads/en/devicedoc/39931d.pdf>
- [7] M24LR-DISCOVERY kit user guide, https://www.st.com/resource/en/user_manual/dm00068980-m24lrdiscovery-kit-user-guide-stmicroelectronics.pdf
- [8] J. Tan et al., "NISP: An NFC to I2C Sensing Platform With Supply Interference Reduction for Flexible RFID Sensor Applications," in IEEE Journal of Radio Frequency Identification, vol. 4, no. 1, pp. 3-13, March 2020, doi: 10.1109/JRFID.2020.2967862.
- [9] Patel, Sagar & Talati, Prachi & Gandhi, Saniya. (2019). Design of I2C Protocol.