



Fondazione ITS Academy per la Mobilità Sostenibile Aerospazio/Meccatronica e Servizi alle Imprese - Piemonte

Corso ITS 15^a Edizione

A.F. 2025-2026 - Torino

Tecnico Superiore per l'Automazione e la Robotica Industriale Embedded System e AI per l'Automazione – EMBT03

Indice

1 INTRODUZIONE.....	3
2 COMPONENTI E STRUMENTI.....	3
3 REALIZZAZIONE.....	4
3.1 Assemblaggio.....	4
NTC.....	4
DHT11.....	4
MAX6675 + Termocoppia.....	4
3.2 Codice.....	5
NTC.....	5
DHT11.....	5
MAX6675 + Termocoppia.....	5
Altre variabili.....	6
Setup.....	6
Delay.....	6
Lettura temperature.....	6
Stampa dati.....	7
4 SPERIMENTAZIONE.....	7
4.1 Acquisizione dati.....	7
4.2 Analisi dati.....	7
5 CONCLUSIONE.....	8
Codice Completo.....	9

Analisi Temperatura

Acquisizione e analisi della temperatura rilevata da tre sensori

1 INTRODUZIONE

Il progetto descrive la progettazione e l'implementazione di un sistema di monitoraggio della temperatura basato sulla scheda Arduino Uno R4 WiFi. L'obiettivo è l'acquisizione simultanea di dati attraverso l'utilizzo di tre diverse tecnologie di rilevamento: un termistore NTC, un sensore digitale DHT11 e una termocoppia con convertitore MAX6675.

L'architettura del sistema è finalizzata a verificare l'accuratezza delle singole tecnologie confrontando le letture ottenute, analizzare i margini di errore di ognuna.

2 COMPONENTI E STRUMENTI

Hardware:

- Arduino Uno R4 WiFi,
- termistore NTC,
- sensore DHT11,
- convertitore A/D MAX6675
- termocoppia K,
- cavi jumper,
- resistenza 10k Ω ,
- cavo USB-C.

Software:

- Arduino IDE,
- libreria esterna 'DHT.h' di Adafruit,
- libreria esterna 'max6675.h' di Adafruit,
- PuTTY,
- Microsoft Excel.



Immagine 1: Termistore NTC

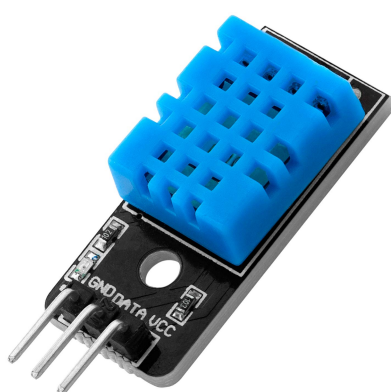


Immagine 2: Sensore DHT11

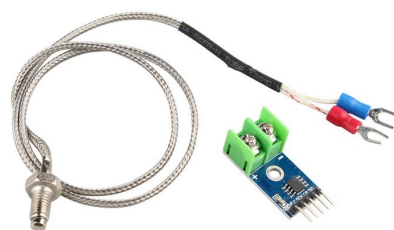


Immagine 3: MAX6675 + Termocoppia

3 REALIZZAZIONE

3.1 Assemblaggio

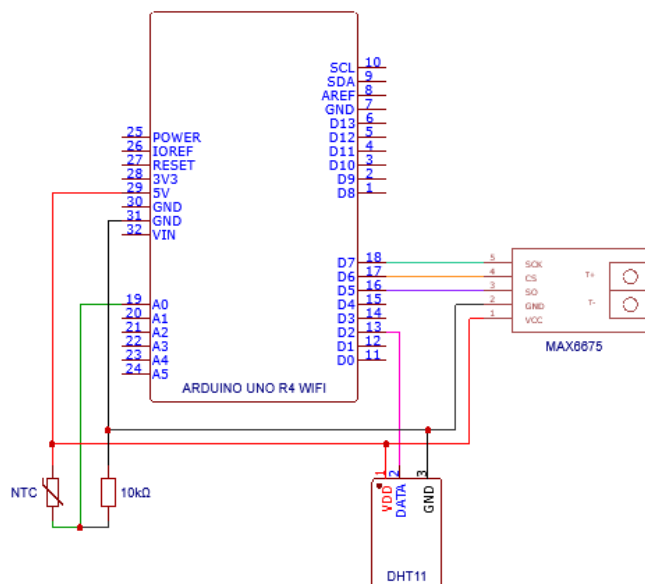


Immagine 4: Schema elettrico collegamenti

NTC

Collegare il termistore NTC all'Arduino:

- portare un'estremità al 5V,
- l'altra va collegata al pin analogico A0 in parallelo a GND con una resistenza di 10kΩ.

DHT11

Collegare il sensore di temperatura DHT11 all'Arduino:

- VDD a 5V,
- GND a GND,
- DATA al pin digitale D2.

MAX6675 + Termocoppia

Collegare prima di tutto la termocoppia al convertitore MAX6675 avvitando le estremità della termocoppia al convertitore. Collegare ora il MAX6675 all'Arduino:

- VCC a 5V,
- GND a GND,
- SO al pin digitale D5,
- CS al pin digitale D6,
- SCK al pin digitale D7.

3.2 Codice

NTC

Sapendo che il funzionamento del termistore NTC segue la formula semplificata:

$$R_T = R_{25} * e^{B * (\frac{1}{T_0} - \frac{1}{T})}$$

posso ricavare la formula per trovare la temperatura:

$$T_0 = \frac{B}{\ln(\frac{R_T}{R_{25}}) + \frac{B}{T_2}}$$

e quindi stabilire le variabili:

```
1 // for NTC
2 float Vout = 0;
3 float R1 = 10000;
4 float R25 = 10000;
5 const float Vi = 1023;
6 float Rt;
7 const float T0 = 273.15;
8 const float T2 = 298.15;
9 const float B = 3965;
```

DHT11

Per il sensore DHT11 si importa la libreria 'DHT.h' di Adafruit che permette di leggere i dati inviati dal sensore all'Arduino, si definiscono poi il pin utilizzato per leggere i dati e il tipo di sensore DHT utilizzato, in fine si definisce l'oggetto 'dht()' con i parametri specificati nelle linee precedenti, questo servirà per leggere i dati di temperatura dal sensore.

```
1 // for DHT11
2 #include "DHT.h"
3 #define DHTPIN 2
4 #define DHTTYPE DHT11
5 DHT dht(DHTPIN, DHTTYPE);
```

MAX6675 + Termocoppia

Per la termocoppia collegata al convertitore MAX6675 si importa la libreria 'max6675.h' di Adafruit e si dichiarano le variabili che identificano i pin digitali a cui è collegato, in fine si definisce l'oggetto 'thermocouple()', con i parametri specificati dalle variabili, che servirà per leggere la temperatura inviata dal sensore.

```
1 // for THERMO
2 #include "max6675.h"
3 int thermoDO = 4;
4 int thermoCS = 5;
5 int thermoCLK = 6;
6 MAX6675 thermocouple(thermoCLK, thermoCS, thermoDO);
```

Altre variabili

Si definiscono le variabili globali per memorizzare i valori letti e gestire il timing dell'acquisizione senza bloccare il processore (usando 'millis()' invece di 'delay()').

```
1 // temperature variables
2 float T_NTC;
3 float T_DHT;
4 float T_THERMO;
5
6 // for the delay
7 const unsigned long delta = 2000;
8 unsigned long now;
9 unsigned long old = 0;
10 unsigned long minutes = 0;
```

Setup

Inizializzazione della comunicazione seriale a 115200 baud per PuTTY e avvio del sensore DHT.

```
1 void setup() {
2     Serial.begin(115200);
3     dht.begin();
4 }
```

Delay

Per gestire il delay è stato usato un blocco 'if()' che permette di evitare la lettura della temperatura per i 2000ms necessari senza stoppare il programma. All'interno del blocco saranno presenti la lettura temperatura e la stampa dati.

```
1 now = millis();
2 if(now >= old + delta){
3     old = now;
4 }
```

Lettura temperature

All'interno del loop, ogni 2 secondi, vengono calcolati i valori: per l'NTC si usa l'equazione semplificata, per DHT e MAX6675 si interrogano le librerie.

```
1 // NTC
2 Vout = analogRead(A0);
3 Rt = (Vi/Vout - 1)*R1;
4 T_NTC = (B/(log(Rt/R25) + (B/T2))) - T0;
5
6 // DHT
7 T_DHT = dht.readTemperature();
8
9 // THERMOCOUPLE
10 T_THERMO = thermocouple.readCelsius();
```

Stampa dati

I dati vengono inviati al monitor seriale formattati come stringa CSV.

```
1 // print on CSV via PUTTY
2 Serial.print(now);
3 Serial.print(",");
4 Serial.print(T_NTC);
5 Serial.print(",");
6 Serial.print(T_DHT);
7 Serial.print(",");
8 Serial.println(T_THERMO);
```

4 SPERIMENTAZIONE

4.1 Acquisizione dati

Per raccogliere i dati è stato utilizzato il software PuTTY, che permette di leggere i valori inviati dall'Arduino sulla porta seriale e di salvarli automaticamente in un file CSV.

Il sistema è stato impostato per acquisire le temperature ogni 2000 millisecondi (2 secondi). Ogni riga del file salvato rappresenta una singola misurazione e i dati sono separati da una virgola per essere letti correttamente dai fogli di calcolo. Il formato della stringa salvata è: 'TIME, NTC, DHT, THERMO'.

In questo modo, si è ottenuto un database pronto per essere importato su Microsoft Excel e analizzato nel dettaglio.

4.2 Analisi dati

Il file in formato CSV, ottenuto tramite PuTTY, è stato importato in Microsoft Excel per l'analisi e la visualizzazione grafica.

Prima di tutto si sono rappresentati i dati grezzi su un grafico a linee per visualizzare l'andamento delle temperature rilevate da ciascuna tecnologia nel tempo.

Successivamente per i set di dati di ognuna, è stata calcolata la varianza dalla media, e sono state rappresentate su un istogramma dedicato per ogni tecnologia, evidenziando la precisione di ciascuno.

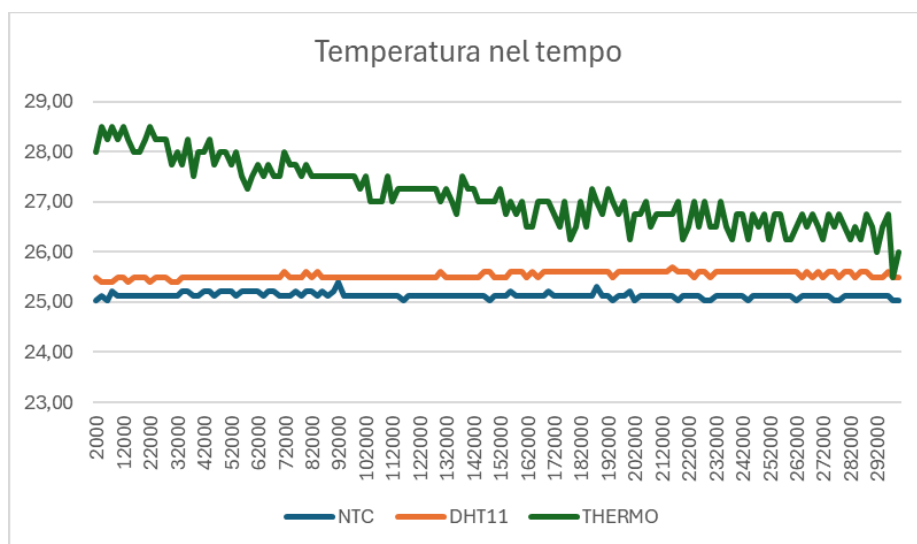


Immagine 5: Temperatura dei tre sensori in relazione al tempo

Infine, un istogramma comparativo tra i valori di tutte le tecnologie ha permesso di visualizzare la varianza di tutti i dati acquisiti.

5 CONCLUSIONE

L'analisi dei dati acquisiti durante il periodo di monitoraggio ha mostrato comportamenti diversi tra le tre tecnologie utilizzate per acquisire i dati.

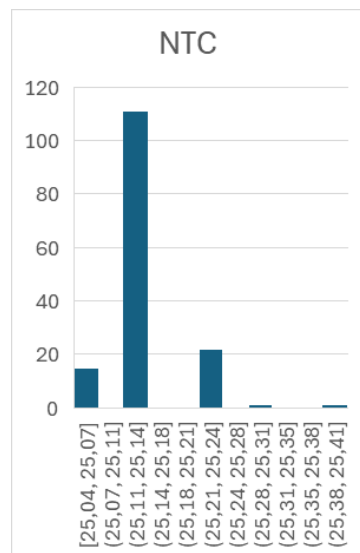


Immagine 6: Istogramma NTC

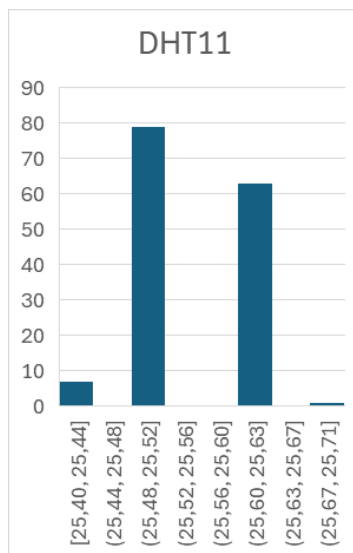


Immagine 7: Istogramma DHT11

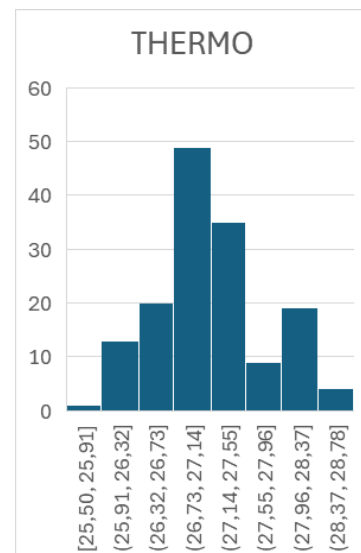


Immagine 8: Istogramma termocoppia

Il termistore NTC ha dimostrato la maggiore stabilità, con letture concentrate tra 25,04°C e 25,24°C. Questa precisione è probabilmente dovuta al calcolo della temperatura tramite l'equazione semplificata implementata nel codice.

Il DHT11 ha mostrato letture abbastanza stabili, concentrate principalmente sui valori di 25,50°C e 25,60°C. Questa caratteristica è tipica dei sensori digitali a basso costo, che hanno una risoluzione limitata ma comunque sufficiente per molte applicazioni.

La termocoppia con MAX6675 ha registrato la maggiore variabilità, con valori compresi tra 25,50°C e 28,50°C. Questa escursione più ampia è dovuta probabilmente ad una diversa posizione del sensore o ad una maggiore sensibilità.

È possibile notare una differenza di circa 2-3°C tra le letture medie della termocoppia e quelle del termistore e del DHT11. Questa differenza può essere attribuita alla diversa posizione di montaggio o a sensibilità diverse tra le varie tecnologie.

Il confronto tra le varie tipologie di misurazione della temperatura ha permesso di verificare che ogni metodo presenta caratteristiche specifiche: l'NTC offre maggiore precisione e costo contenuto, il DHT11 una spiccata semplicità d'uso, mentre la termocoppia risulta essere più sensibile alle variazioni rapide. L'utilizzo di tutte le tecnologie contemporaneamente sullo stesso periodo di tempo e nella stessa zona ha permesso di evidenziare le differenze, con pregi e difetti, di ognuna delle tecnologie usate.

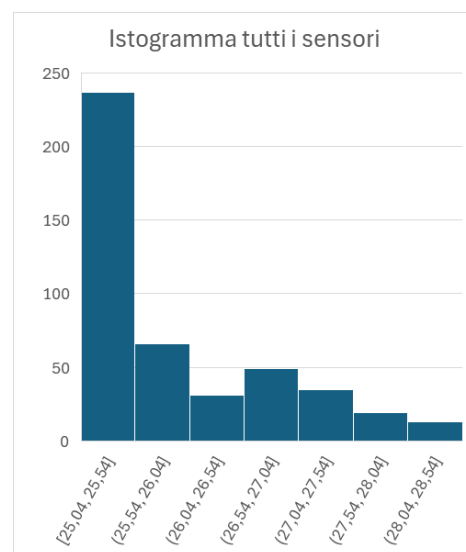


Immagine 9: Istogramma tutti i dati

Codice Completo

```

1 // for NTC
2 float Vout = 0;
3 float R1 = 10000;
4 float R25 = 10000;
5 const float Vi = 1023;
6 float Rt;
7 const float T0 = 273.15;
8 const float T2 = 298.15;
9 const float B = 3965;
10 float HUM;
11
12 // for DHT11
13 #include "DHT.h"
14 #define DHTPIN 2
15 #define DHTTYPE DHT11
16 DHT dht(DHTPIN, DHTTYPE);
17
18 // for THERMO
19 #include "max6675.h"
20 int thermoDO = 4;
21 int thermoCS = 5;
22 int thermoCLK = 6;
23 MAX6675 thermocouple(thermoCLK, thermoCS, thermoDO);
24
25 // temperature variables
26 float T_NTC;
27 float T_DHT;
28 float T_THERMO;
29
30 // for the delay
31 const unsigned long delta = 2000;
32 unsigned long now;
33 unsigned long old = 0;
34 unsigned long minutes = 0;
35
36
37 void setup() {
38     Serial.begin(115200);
39     dht.begin();
40 }
41
42 void loop() {
43     now = millis();
44     if(now >= old + delta){
45
46         // debug delay
47         /*
48         Serial.print("o: ");
49         Serial.print(old);
50         Serial.print(" - n: ");
51         Serial.println(now);
52         */
53
54         old = now;
55
56         // NTC
57         Vout = analogRead(A0);
58         Rt = (Vi/Vout - 1)*R1;
59         T_NTC = (B/(log(Rt/R25) + (B/T2))) - T0;
60
61         // DHT
62         T_DHT = dht.readTemperature();
63
64         // THERMOCOUPLE
65         T_THERMO = thermocouple.readCelsius();
66
67         // print on CSV via PUTTY
68         Serial.print(now);
69         Serial.print(",");
70         Serial.print(T_NTC);
71         Serial.print(",");
72         Serial.print(T_DHT);
73         Serial.print(",");
74         Serial.println(T_THERMO);
75     }
76 }
77

```