**Istituto Professionale per L'industria L'Artigianato**
"Antonio Guastaferro"

**63074 SAN BENEDETTO DEL TRONTO (AP)**

Classe : 4A_IPAI 5A_IPAI
A.S. : 2021-2022
Docente : Tufoni Franco
Disciplina : Tecnologie elettriche-elettroniche, dell'automazione e applicazioni

# Arduino

# Moduli I2C

## 1) Display LCD 20x4


## 2) BME680


# Hardware - Software

*Vers. 2.7*

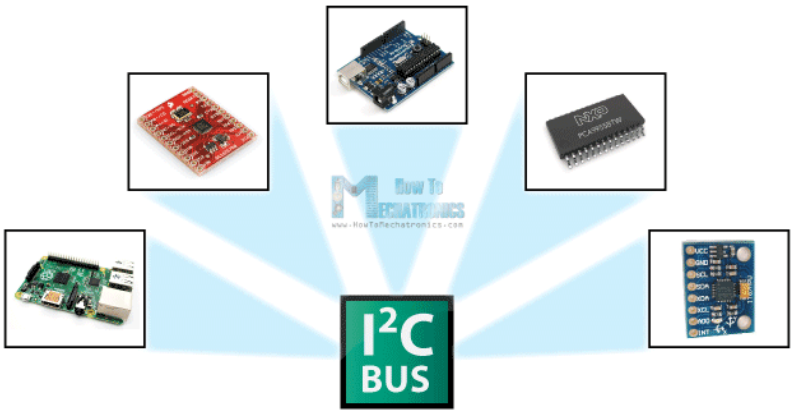
*Open Source*

## Protocollo I2C con Arduino

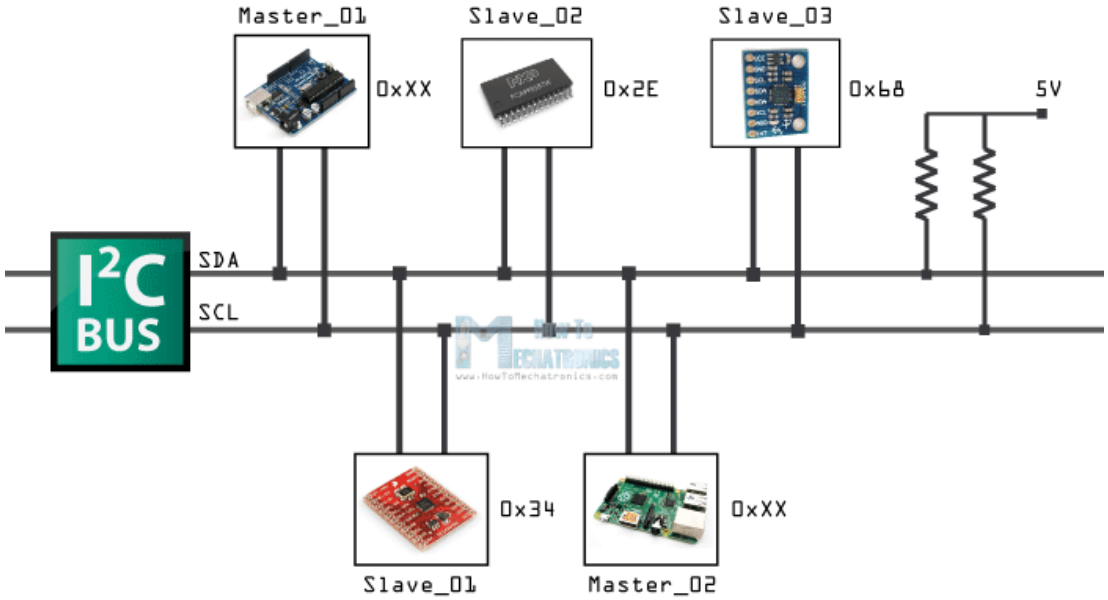
Il protocollo I2C, acronimo di **I**nter **I**ntegrated **C**ircuit, è un metodo di comunicazione seriale ideato dalla Philips che permette il trasferimento dati utilizzando solo due fili, nello specifico uno trasmette i dati e l'altro, tramite un Clock, sincronizza la trasmissione.
Il BUS è composto da 4 fili, due per l'alimentazione e due per i dati:

- Alimentazione (Vcc, GND)
- Una linea di scambio dati (SDA = Serial Data)
- Una linea per la sincronizzazione (SCL = Serial Clock)

Il bus di comunicazione I2C è molto popolare e ampiamente utilizzato da molti dispositivi elettronici perché può essere facilmente implementato in molti progetti elettronici che richiedono la comunicazione tra un **MASTER** e più dispositivi **SLAVE** o anche più dispositivi **MASTER**. Le facili implementazioni derivano dal fatto che sono necessari solo due fili per la comunicazione tra un massimo di 128 (max 112) dispositivi quando si utilizza l'indirizzamento a 7 bit e fino a quasi 1024 (max 1008) dispositivi quando si utilizza l'indirizzamento a 10 bit.



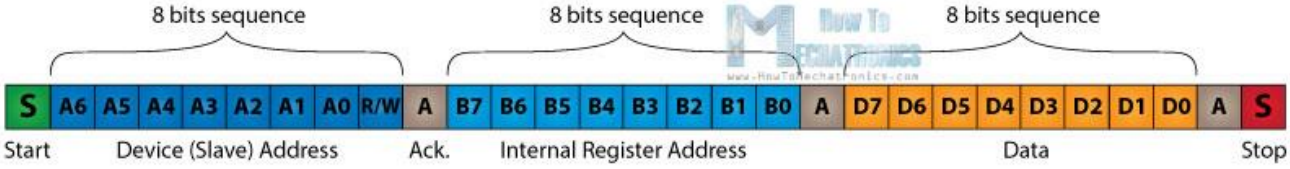

I due fili o linee sono chiamati Serial Clock (o SCL) e Serial Data (o SDA). La linea SCL è il segnale di clock che sincronizza il trasferimento dati tra i dispositivi sul bus I2C ed è generato dal dispositivo master. L'altra linea è la linea SDA che trasporta i dati.
I dispositivi con BUS I2C sono dotati di un indirizzo scritto in esadecimale e vengono collegati tutti in parallelo, come indicato in figura.
I moduli non possono avere lo stesso indirizzo.



Le due linee sono "open-drain", il che significa che devono essere collegate resistenze di pull up in modo che le linee siano alte perché i dispositivi sul bus I2C sono attivi bassi. I valori comunemente usati per i resistori vanno da 2K per velocità più elevate a circa 400 kbps, a 10K per velocità inferiori a circa 100 kbps. Spesso i resistori di Pull_Up sono inserite all'interno dei moduli, quindi non occorre effettuare il collegamento esterno.

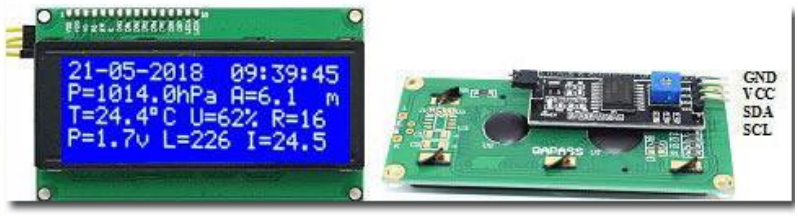
## Protocollo I2C

Il segnale dati viene trasferito in sequenze di 8 bit. Quindi, dopo che si verifica una condizione di avvio speciale, arriva la prima sequenza di 8 bit che indica l'indirizzo dello slave a cui vengono inviati i dati. Dopo ogni sequenza di 8 bit segue un bit chiamato Riconoscimento. Dopo il primo bit di riconoscimento, nella maggior parte dei casi, arriva un'altra sequenza di indirizzamento, ma questa volta per i registri interni del dispositivo slave. Subito dopo le sequenze di indirizzamento seguono le sequenze di dati altrettante fino a quando i dati non vengono completamente inviati e si conclude con una condizione di stop speciale.




## Display LCD I2C 20x4

Il Modulo Display LCD 20x4 dispone di 4 righe e 20 colonne è basato sul controller HD44780; dotato di retroilluminazione blu, caratteri bianchi, regolazione del contrasto e di un'interfaccia di comunicazione I2C.




Lo schermo a cristalli liquidi (LCD) è un display a schermo piatto, una visualizzazione elettronica o un video che utilizza le proprietà modulanti della luce dei cristalli liquidi.

I cristalli liquidi non emettono direttamente la luce.

Come mostrato nell'immagine, un'importante caratteristica di questo modulo LCD è l'interfaccia di comunicazione I2C integrata posta sul retro che ne rende estremamente semplice l'utilizzo con Arduino.

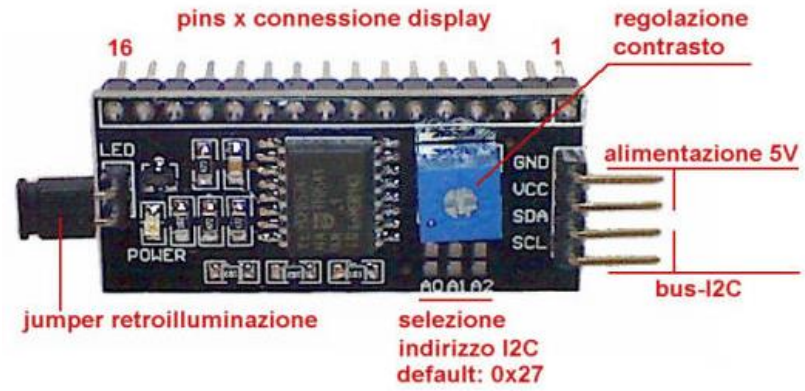
Il modulo è dotato di **4 pin** (Alimentazione: **Vcc, Gnd**) e (Dati: **SDA, SCL**)

Questo modulo permette di comunicare con un display LCD mediante il protocollo I2C che impegna quindi due sole porte. (**SDA (**dati**), SCL(**Clock**)**) oltre ai due terminali di alimentazione (**VCC, GND**).

L'**I2C** è un Bus bidirezionale di tipo seriale-Multi Master in cui più dispositivi possono assumere il controllo del Bus. La trasmissione dei dati avviene, come accennato, per mezzo di due sole linee denominate rispettivamente SDA (Serial Data pin A4) e SCL (Serial Clock pin A5). Sulla prima viaggiano effettivamente i Bit d'informazione che microcontrollore e Display si scambiano; sulla seconda linea viaggia il segnale di Clock, generato sempre ed esclusivamente da un Master, il cui scopo è quello di sincronizzare i dispositivi stabilendo la validità e il significato dei Bit presenti sulla linea dati.

## Adattatore/convertitore - Settaggio dell'indirizzo del modulo

Sul retro del display è presente l'adattatore/convertitore dal bus seriale I2C al bus parallelo utilizzato dai Display LCD con standard HD44780 e matrice (colonne x righe): 4x2, 8x2, 16x2, 20x2, 16x4, 20x4 ecc




Ad esempio un display LCD 2004 impegna per il suo controllo, a prescindere dall'alimentazione, almeno 6 porte del Microcontroller di sistema per cui il numero delle porte restanti può risultare insufficiente per le necessità del progetto che si vuol realizzare.
Questo modulo permette di comunicare con un display LCD mediante il protocollo I2C che impegna quindi due sole porte.
Sul modulo è presente un potenziometro per la regolazione del contrasto ed un jumper rimovibile per l'attivazione della retroilluminazione. Se il jumper viene rimosso e tra i due pin viene inserita una resistenza, si può modificare l'intensità di retroilluminazione (ad esempio con una resistenza da 470 ohm 1/4w l'intensità si dimezza).
Sono inoltre presenti tre connessioni denominate A1, A2 e A3 per il settaggio (a 3-bit) dell'indirizzo I2C tra 0x20 e 0x27.

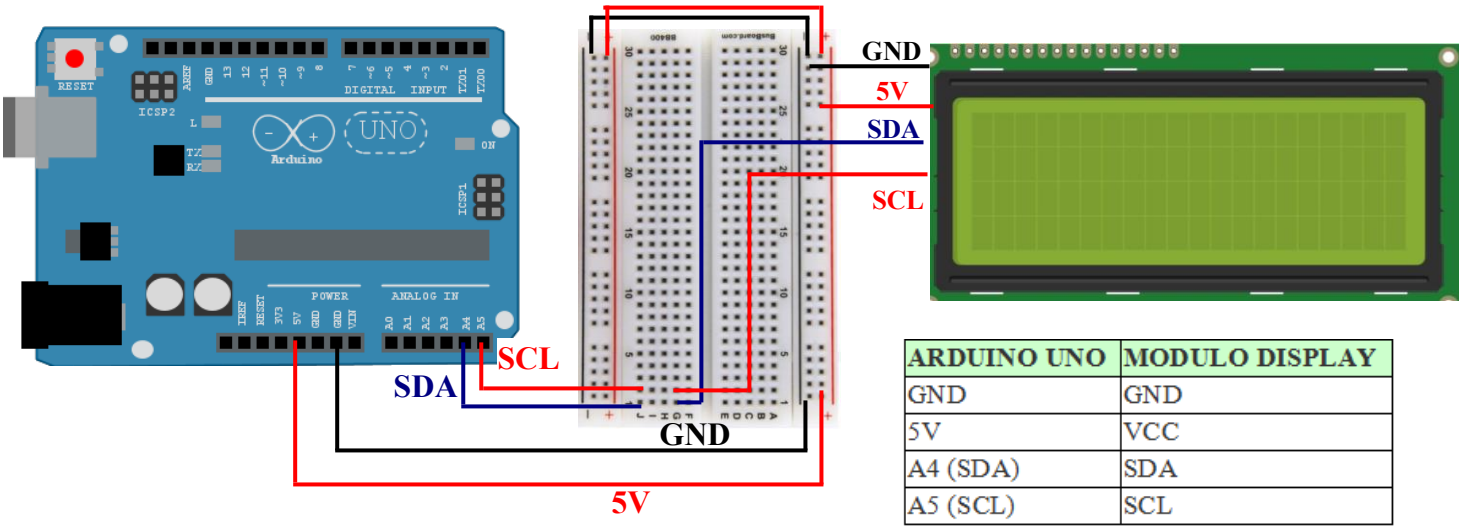
selezione indirizzo I2C

| indirizzi | A0 | A1 | A2 |
|---|---|---|---|
| 0x20 | chiuso | chiuso | chiuso |
| 0x21 | aperto | chiuso | chiuso |
| 0x22 | chiuso | aperto | chiuso |
| 0x23 | aperto | aperto | chiuso |
| 0x24 | chiuso | chiuso | aperto |
| 0x25 | aperto | chiuso | aperto |
| 0x26 | chiuso | aperto | aperto |
| 0x27 | aperto | aperto | aperto |

L'indirizzo di default è **0x27** (A0, A1 e A2 lasciati aperti).
In tabella sono riportate le possibili combinazioni ed il relativo indirizzo in esadecimale.
La chiusura di una coppia si effettua collegando tra loro le due rispettive piazzole. E' ovvio che tale indirizzo hardware deve coincidere con l'indirizzo I2C nel software/libreria di gestione del Modulo.
**Il display viene gestito tramite la libreria *LiquidCrystal_I2C.h*.**

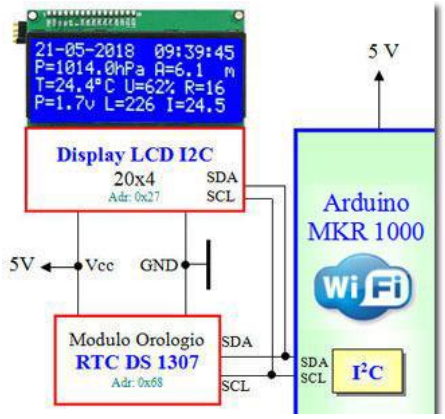
## Collegamento modulo con Arduino UNO

Si utilizzano i piedini A4 e A5 ai quali sono collegati rispettivamente i terminali SDA e SCL del bus $I^2C$. I collegamenti sono quelli indicati in tabella.



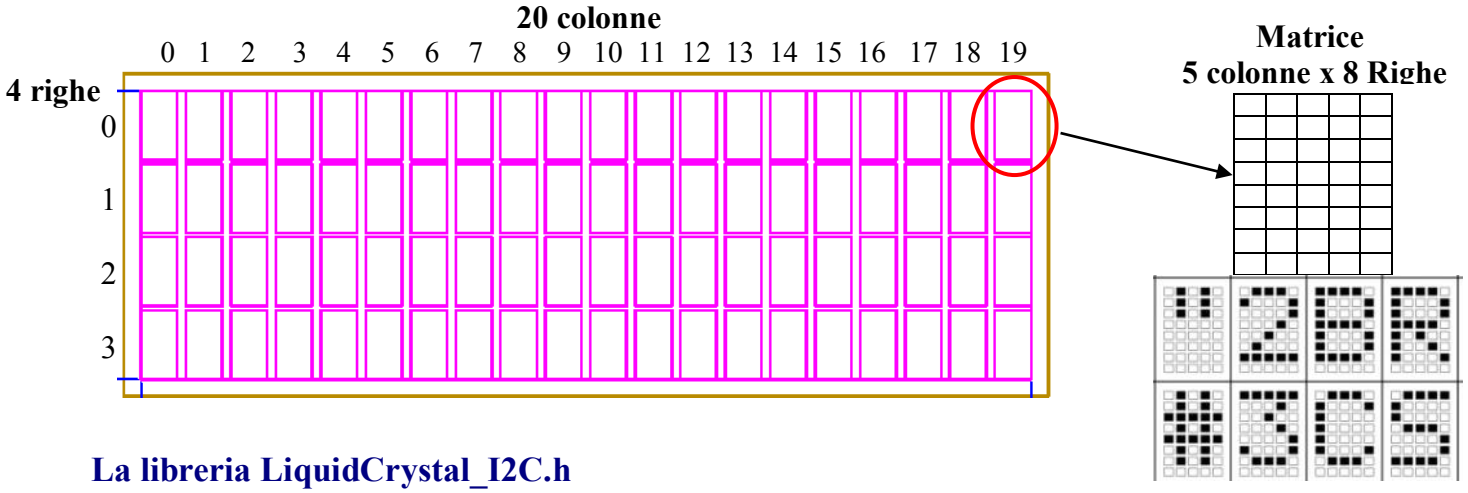

| ARDUINO UNO | MODULO DISPLAY |
|---|---|
| GND | GND |
| 5V | VCC |
| A4 (SDA) | SDA |
| A5 (SCL) | SCL |

La posizione dei pin SDA (Linea Dati) e SCL (Clock) dipende da modello di Arduino. In alcuni ci sono i pin dedicati denominati SDA e SCL.




**Esempio con Arduino MKR1000**

**N.B.:** Alcune tipologie di display escono dalla fabbrica con un'impostazione della luminosità dei caratteri sbagliata e per questo non risultano visibili. Se una volta caricato lo sketch il display si spegne e si riaccende, ma non appare nessun carattere, è necessario agire sul potenziometro posizionato nella parte posteriore (visibile in figura) per regolare la luminosità dei caratteri.




## La libreria LiquidCrystal_I2C.h

Operativamente si fa uso della libreria *LiquidCrystal_I2C.h* che è necessario aver caricato nell'IDE di Arduino; con queste librerie è possibile gestire tutta la comunicazione.
Per includere la libreria nell'IDE di sviluppo di Arduino è sufficiente utilizzare il Gestore Library (disponibile da IDE versione 1.6.2). Aprire l'IDE e fare clic sul menu "Sketch" e poi: #include libreria > Gestione librerie... . Si aprirà una finestra di dialogo che come prima opzione avrà "gestione librerie". Cliccare su tale opzione e cercare " ***LiquidCrystal_I2C*** ". Se non è installata, scegliere la versione (senza dubbio è meglio la più recente) e successivamente cliccare sul tasto "installa".
Se non è disponibile la libreria dalla stessa finestra di dialogo è possibile specificare il file .zip contenete la libreria da installare. N.B. Nel caso vi siano aggiornamenti, l'IDE automaticamente vi informerà e chiederà se si vogliono installare o meno.
Ulteriori informazioni sull'uso della libreria sono reperibili al seguente
link https://github.com/johnrickman/LiquidCrystal_I2C

### Impostazione programma e comandi della libreria

#### Prima di void setup()

```
// Includere le librerie:
// LiquidCrystal_I2C.h: https://github.com/johnrickman/LiquidCrystal_I2C
#include <Wire.h> // Libreria per la comunicazione I2C
#include <LiquidCrystal_I2C.h> // Libreria per LCD I2C
// Cablaggio: il pin SDA è collegato ad A4 e il pin SCL ad A5.
// Connessione LCD tramite I2C, indirizzo predefinito 0x27 (A0-A1-A2 non ponticellati)
LiquidCrystal_I2C lcd ( 0x27 , 20, 4 ); // imposta oggetto lcd: Indirizzo Hex (0x27) Colonne (20), Righe (4).
```

La libreria LiquidCrystal_I2C funziona in combinazione con la libreria **Wire.h** che consente di comunicare con i dispositivi I2C. Questa libreria è preinstallata con l'IDE di Arduino. * *Quando si utilizza l'ultima versione della libreria LiquidCrystal_I2C, non è più necessario includere la libreria wire.h nel proprio Sketch. L'altra libreria importa wire.h automaticamente.*

L'istruzione **LiquidCrystal I2C lcd ( 0x27 , 20, 4 ) ;** crea l'oggetto **lcd** ed imposta i parametri del display Indirizzo Hex (**0x27**) , Colonne (**20**), Righe (**4**).

**All'interno di void setup()**

```
// Inizializza il display LCD:
lcd.init();          //Attiva il display
lcd.backlight(); // Attiva la retroilluminazione
```

**All'interno di void setup() e di void loop()** è possibile inserire i comandi per la visualizzazione di costati e variabili.
I comandi principali sono **lcd. setCursor** e **lcd.print** il primo permette di posizionare il cursore il secondo la stampa di un dato (Costante, Variabile) .

**Comandi della libreria**
I comandi devono essere preceduti dal nome dell'oggetto e separati con un punto.
nel nostro esempio abbiamo imostato il nome dell'oggetto pari a **lcd**

**Esempio: lcd. setCursor(1,3);**

Link dei comandi

**https://www.arduino.cc/en/Reference/LiquidCrystal**

**clear()**
permette di cancellare tutto quello che c'è sul display.

**home()**
posiziona il cursore nella coordinata (0,0), cioè in alto a sinistra. Infatti la numerazione delle colonne e delle righe parte da zero.

**setCursor(numCol, numRig)**
posiziona il cursore nella coordinata (numCol, numRig), considerando i seguenti valori:

- colonne : valori da 0 a N-1 colonne del display
- righe : valori da 0 a N-1 righe del display

**cursor()**
visualizza il cursore mediante un trattino basso (tipo underscore) nel punto in cui dovrebbe scrivere.
**noCursor()**
interrompe la visualizzazione del cursore mediante il trattino basso. Alternando con un ritardo cursor() e noCursor() si ottiene il trattino basso lampeggiante nel punto in cui dovrebbe scrivere.

**blink()**

visualizza il cursore accendendo tutti i puntini del reticolo, solitamente 5x8 punti.

**noBlink()**
interrompe la visualizzazione del cursore con tutti i puntini del reticolo.

**display()**
attiva la visualizzazione dei caratteri inviati al display.

**noDisplay()**
disattiva la visualizzazione dei caratteri inviati al display.

**scrollDisplayLeft ()**
Sposta tutti i caratteri e il cursore sul display di una posizione a sinistra

**scrollDisplayRight ()**
Sposta tutti i caratteri e il cursore sul display di una posizione a destra

**leftToLeft()**
Permette di scrivere da destra a sinistra, cioè il primo carattere viene scritto nella posizione specificata, gli altri a sinistra.

**leftToRight()**
Permette di scrivere da sinistra a destra, cioè il primo carattere viene scritto nella posizione specificata, gli altri a destra (selezione di default).

**autoscroll()**
sposta tutto lo schermo di una posizione verso la direzione di scrittura specificato da leftToLeft o LeftToRight.

**I caratteri standard del display:**

È possibile creare fino a 8 caratteri mediante il reticolo 5x8 nel seguente modo:

**Step 1**

Creare un array tipo byte specificando per ognuna delle 8 righe quali punti devono essere accesi e quali spenti. Per accendere un punto va messo in coincidenza della posizione il bit a 1 mentre per spegnerlo il bit va messo a 0.

I valori di ognuno degli 8 elementi dell'array può essere espresso in binario o in esadecimale, quest'ultimo però è meno chiaro nel risultato.

```
byte nomeCarattere[]= {
B00000,
B00000,
B00000,
B00000,
B00000,
B00000,
B00000,
B00000};
```

**Link per generare caratteri**

https://mikeyancey.com/hamcalc/lcd_characters.php

## Esercizio 1_i2C

```
/* Esercizio 1_I2c
  * Lettura tensione sul PinA0 e
 * visualizzazione sul Display I2C
 */
int val = 0; float volt=0;

#include<Wire.h>;// Libreria per la comunicazione I2C
#include<LiquidCrystal_I2C.h>;// Libreria per LCD I2C
LiquidCrystal_I2C lcd (0x27,20,4);//Hex (0x27) 20 col. 4 righe

void setup ()
{
  lcd.init();              //Attiva il display
  lcd.backlight();        // Attiva la retroilluminazione
  lcd.setCursor(0,0);lcd.print("Lettura Volt pin A0");
}
void loop()
{
  val = analogRead(A0); //Lettura valore Analogico e conv. A/D
  volt = val * (5.00 / 1024.0); // Conversione Numero in Volt
lcd.setCursor(0,1);lcd.print("N=     ");
lcd.setCursor(3,1);lcd.print(val);
lcd.setCursor(0,2);lcd.print("Volt=     ");
lcd.setCursor(6,2);lcd.print(volt);
}
```

## Esercizio 2_I2C – Rilevazione temperatura con il TMP 36 e Display LCD I2C

Dimensionare e programmare un circuito in grado di rilevare la temperatura ambiente con il sensore TMP36 e visualizzare il valore tramite il Monitor Seriale dell'IDE

1. Disegnare lo schema elettrico
2. Disegnare lo schema di cablaggio
3. Scrivere il programma

### Soluzione

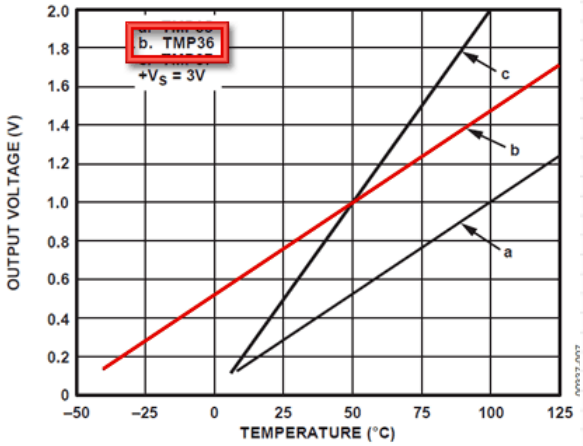
Il TMP36 è un sensore di temperatura analogico, fornisce un'uscita lineare pari a 10mv/°C con un offset di 500mV nel range -40°C +125°C, alla temperatura di **0°C** il sensore eroga una tensione di **500mV**.
Il legame Temperatura misurata (°C) e tensione di uscita è dato dalla seguente caratteristica di trasferimento:

$V_{out}=K \bullet T+0,5$ dove K=10mv/°C, T=temperatura espressa in °C e 0,5 rappresenta l'Offset.

Trasformazione in:

T(°F)=1,8*T+32 T(K)=273,15+T

In figura 1 Grafico (*Vout in funzione della Temperatura*) e Caratteristiche tecniche.




| | |
|---|---|
| Altezza | 5.2mm |
| Profondità | 4.19mm |
| Lunghezza | 5.2mm |
| Package fornitore | TO-92 |
| Funzione | Sensore di temperatura |
| Numero pin | 3 |
| Precisione | ± 2 ° C |
| Linearità | ± 0,5 ° C |
| Sensibilità | 10 mV / °C fattore di scala |
| Campo temperatura minima | -40 ° C a +125 ° C, funzionamento a +150 ° C |
| Tensione tipica di funzionamento | 2,7 V a 5,5 V |
| Tipo uscita | Analogica |
| Corrente di riposo | Meno di 50 µA |
| | Basso auto-riscaldamento |

Fig. 1

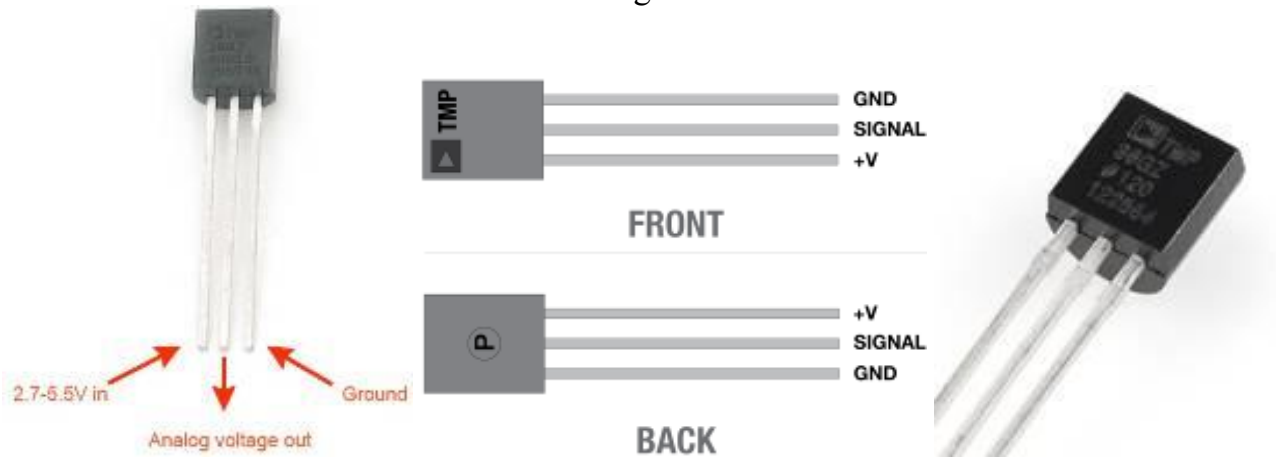



Fig. 2 (Piedinatura e foto)

In Fig. 3 è riportato lo schema elettrico

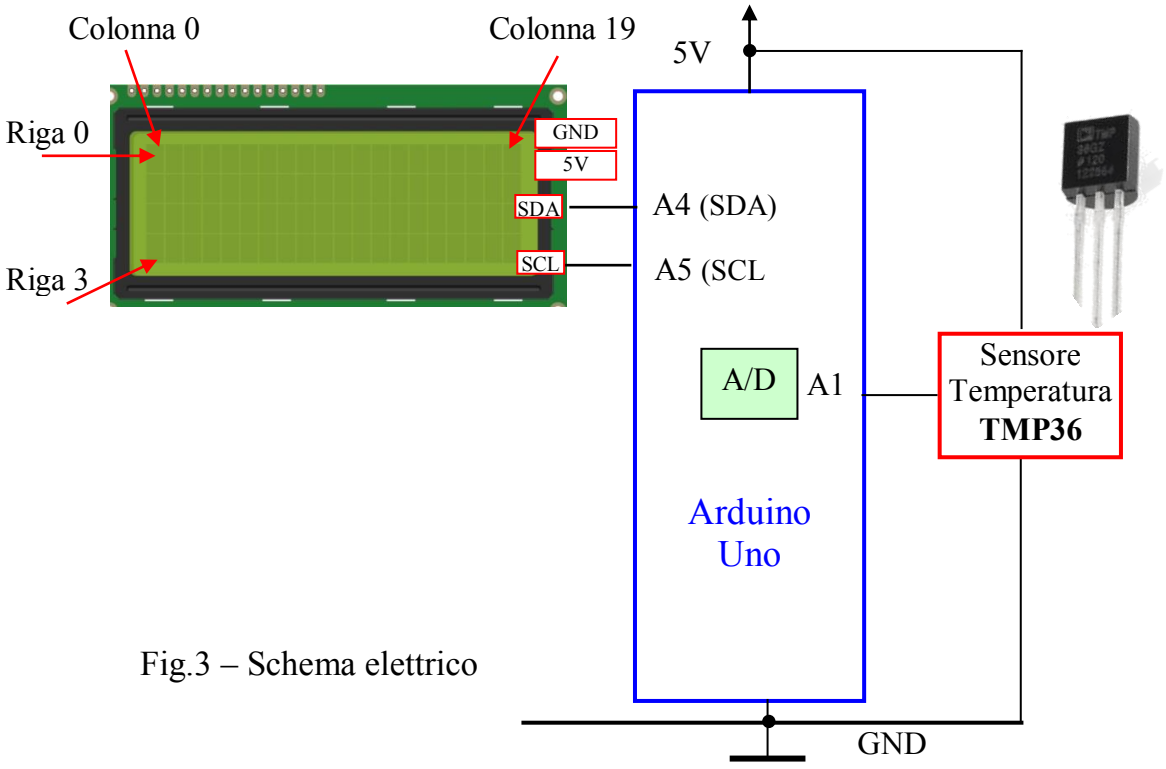



Fig.3 – Schema elettrico

Il segnale analogico fornito dal sensore ($V_{out}$) viene inviato al convertitore A/D interno (Pin A1) che lo trasforma in un numero nel range (0÷1023).
Tramite software si trasforma il numero nel valore di Tensione e nel valore di Temperatura .

**Formule:**

**$V_{out}$=K•T+0,5 [V]** dove K=10mv/°C, T=temperatura espressa in °C e 0,5 rappresenta l'Offset.

$$N = \frac{Vout}{\frac{5}{1024}} \text{ [N. out A/D]} \qquad Vout = \frac{N \bullet 5}{1024} \text{ [V]} \qquad T = \frac{Vout - 0{,}5}{K} \text{ [°C]}$$

**Programma**

```
/* Esercizio 2_I2c
* Rilevazione temperatura con il Sensore TMP 36 e Display LCD I2C
*/
int valoreADC = 0;
float volt=0; float temp=0;

#include<Wire.h> // Libreria per la comunicazione I2C
#include<LiquidCrystal_I2C.h> // Libreria per LCD I2C
LiquidCrystal_I2C lcd (0x27,20,4);//Hex (0x27) 20 col. 4 righe

void setup ()
{
  lcd.init();       //Attiva il display
  lcd.backlight(); // Attiva la retroilluminazione
  lcd.clear();       // Pulisce lo schermo LCD
}
```

```
void loop ()
{
valoreADC = analogRead(A1);
volt = (valoreADC/1024.0)* 5.0;
temp=(volt-0.5)/0.01;
// Invio dati al Display LCD
lcd.setCursor(0,0;lcd.print(" Temperatura - TMP36");
lcd.setCursor(0, 1); lcd.print("Vout=");
lcd.setCursor(5, 1); lcd.print(volt,3);
lcd.setCursor(10, 1); lcd.print("V");
lcd.setCursor(0, 2); lcd.print("Temp=");lcd.print(temp,2);
lcd.setCursor(10, 2); lcd.print("C");
delay (2000); //Pausa di 2 secondi
}
```

## Esercizio 3_I2C – Rilevazione temperatura, umidità, pressione atmosferica con il Modulo BME680

Dimensionare e programmare un circuito in grado di rilevare la temperatura, l'umidità e la pressione atmosferica con il modulo BME680 e visualizzare il valori sul display LCD I2C 20x04.

1. Disegnare lo schema elettrico
2. Scrivere il programma

### Soluzione

In Fig. 1 è riportato lo schema elettrico.

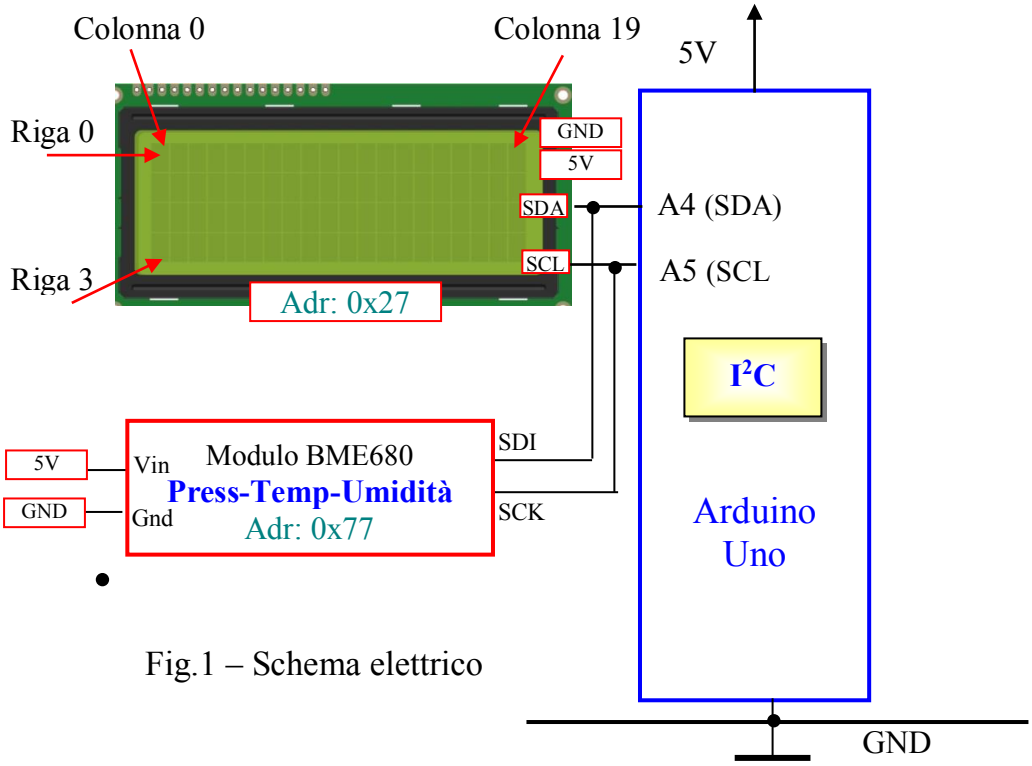



Fig.1 – Schema elettrico

### Modulo BME680

**BME 680: *Temperatura, Umidità, Pressione Atmosferica***

Il modulo viene utilizzato per il rilevamento ed acquisizione delle seguenti grandezze


**1) Pressione Atmosferica, Temperatura, Umidità**

***- rilevamento diretto***

**2) Temperatura di rugiada, indice di calore**

***- calcolo tramite algoritmi***

**Descrizione BME680**

Il Sensore di Gas Pressione Temperatura Umidità BME680 è in grado di rilevare diversi parametri ambientali come la temperatura, l'umidità, la pressione barometrica ed i composti organici volatili (VOC).

Questo sensore è in grado di misurare l'umidità con precisione del ± 3%, la pressione barometrica con precisione assoluta di ± 1 hPa e la temperatura con precisione di ± 1.0 °C, poiché la pressione cambia con l'altitudine è anche possibile utilizzare il sensore come altimetro con una precisione di ± 1 metro.

Il BME680 contiene anche un piccolo sensore MOX, che cambia la propria resistenza in funzione dei composti organici volatili (VOC) presenti nell'aria.

Come tutti i sensori VOC dovrà essere calibrarlo utilizzando una fonte note per ottenere misurazioni precise

Al primo utilizzo è consigliabile stabilizzare il sensore lasciandolo acceso per 48 ore e poi per 30 minuti nella modalità desiderata ogni volta che viene utilizzato, questo perché i livelli di sensibilità del sensore cambieranno durante l'uso iniziale e la resistenza aumenterà lentamente nel tempo man mano che il MOX si scalda fino alla sua lettura di base.
Il sensore può essere alimentato con una tensione compresa tra 3,3Vcc e 5Vcc, i dati in uscita sono disponibili sia su I2C sia su SPI.
BME680 è un sensore ambientale integrato sviluppato in modo specifico per applicazioni mobili e dispositivi indossabili in cui le dimensioni e il basso consumo energetico sono requisiti fondamentali.

***Caratteristiche:***

- Alimentazione compresa tra 3,3Vcc e 5Vcc
- Uscita I2C o SPI
- Caratteristiche sensore di umidità: precisione ± 3%, tempo di risposta 8s
- Caratteristiche sensore di pressione: campo di misura compreso tra 300 hPa e 1100 hPa, precisione ± 1 hPa (da 950 hPa a 1050 hPa @25°C)
- Caratteristiche sensore di temperatura: campo di misura compreso tra -40°C e +85°C, precisione ± 1°C (da 0°C a +65°C)
- Caratteristiche sensore di gas: tempo di risposta 1s

***Applicazioni***

Misurazione della qualità dell'aria
Stazione meteo personalizzata
Consapevolezza del contesto, ad esempio rilevamento dell'umidità della pelle, delle variazioni nell'ambiente
Monitoraggio fitness/benessere
Avvertimento in caso di aria secca o temperature elevate
Misurazione del volume e del flusso d'aria
Controllo automazione domestica (ad es. HVAC)
Potenziamento del GPS (ad es., miglioramento del Time To First Fix (TTFF), determinazione del punto stimato, rilevazione di pendenza)

***Pin di alimentazione**:*

- **Vin** : Pin alimentazione. Poiché il chip del sensore utilizza 3 VCC, abbiamo incluso un regolatore di tensione a bordo che prenderà 3-5 VCC e lo convertirà in sicurezza. Per alimentare la scheda, fornisci la stessa potenza del livello logico del tuo microcontrollore - ad es. Per un micro 5V come Arduino, usa 5V
- **3Vo** - uscita 3.3V del regolatore di tensione interno ( max 100mA)
- **GND** - questo è il pin di messa a terra dell'alimentazione e del segnale.

***Pin logici SPI:***

Tutti i pin che entrano nel breakout hanno circuiti di spostamento di livello per renderli sicuri a livello logico 3-5V. Usa qualunque livello logico su **Vin!**

- **SCK** - Questo è il pin **Clock** , è un input per il chip
- **SDO** - questo è il **S**erial **D**ata **O**ut / **M**aster **I**n **S**lave **O**ut pin, per i dati inviati dal BME680 al tuo processore
- **SDI** - questo è il **S**erial **D**ata **I**n / **M**aster **O**ut **S**lave **I**n pin, per i dati inviati dal processore al BME680
- **CS** - questo è il **C**hip **S**elect pin, livello basso per avviare una transazione SPI. È un input per il chip

Se si desidera collegare più BME680 a un microcontrollore, farli condividere i pin SDI, SDO e SCK. Quindi assegnare a ciascuno un pin CS univoco.

### *Pin logici I2C: Adr: 0x77*

- **SCK** - Pin di clock I2C, corrisponde al segnale SCL.
- **SDI** - Pin dati I2C, corrisponde al segnale SDA.

### *Gestione software*

Per la gestione del modulo installare le librerie ***Adafruit_Sensor.h*** e ***Adafruit_BME680.h***

### Programma

```
/* Esercizio 3_BME680
* Rilevazione temperatura, Umidità, Pressione Atmosferica
* con il modulo BME680
*/
// Dichiarazione variabili: Umidità(um),Temperatura (temp), Pressione
atmosferica (pat)
float um=0; float temp=0;float pat=0;
#include<Wire.h>// Libreria per la comunicazione I2C
#include<LiquidCrystal_I2C.h>  // Libreria per LCD I2C
#include <Adafruit_Sensor.h>  //Libreria Moduli sensori Adafruit
#include <Adafruit_BME680.h> //Libreria Moduli BME680
// Dichiarazione oggetti
LiquidCrystal_I2C lcd (0x27,20,4);// Oggetto lcd - Hex (0x27) 20
col. 4 righe
Adafruit_BME680 bme; // Oggetto bme per BME680  Hex (0x77)

void setup ()
{
  // Set Parametri delmodulo BME680
  bme.begin();
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320*C for 150 ms
  lcd.init();    //Attiva il display
  lcd.backlight();        // Attiva la retroilluminazione
  lcd.clear(); // Pulisce lo schermo LCD
}
 void loop ()
{
temp=bme.readTemperature(); // Lettura Temperatura BME680
um=bme.readHumidity();      // Lettura Umidità BME680
pat=bme.pressure / 100.0;  //Lettura pressione Atmosferica in hPa
// Invio dati al Display LCD
lcd.setCursor(0,0);lcd.print(" Modulo I2C BME680");
lcd.setCursor(0, 1);
lcd.print("Temp=");lcd.print(temp,2);lcd.setCursor(12, 1);
lcd.print(" C");
lcd.setCursor(0, 2);  lcd.print("Umid=");lcd.print(um,2);
lcd.setCursor(12, 2); lcd.print(" %");
lcd.setCursor(0, 3);  lcd.print("PAtm=");lcd.print(pat,2);
lcd.setCursor(12, 3); lcd.print(" hPa");
delay (1000); //Pausa di 1 secondo
}
```