

MATERIA	ANNO SCOLASTICO	INSEGNANTI
TPSIT	2022/2023	ZANELLA SIMONE DE ROSSI MARCO
LUOGO E DATA	CLASSE	ALUNNO/ALUNNI
05/12/2022	4° B	SAPPIA FULVIO

TITOLO DELLA PROVA/PROGETTO/LAVORO

TELEMETRO AD ULTRASUONI

OBIETTIVI

Realizzare un circuito elettrico per la misurazione della distanza di un oggetto dal punto di osservazione, attraverso una rilevazione acustica.

STRUMENTAZIONE UTILIZZATA

- ARDUINO UNO R3

Scheda elettronica con un microcontrollore ([Atmel ATmega328](#)), programmabile con il suo ambiente (IDE di arduino).



- LINGUAGGIO C/C++

Utilizza per scrivere il codice sorgente da inserire all'interno dell'Arduino IDE, che lo caricherà sulla scheda.



- IDE ARDUINO

L'IDE di arduino sta per "Integrated Development Environment", è un'applicazione multiplatforma basata su Java e Electron, hostata su [GitHub](#).

Last Version: 2.0.2
Software: Open Source
Licence: LGPL o GPL
Available: Windows | Linux | Mac OS



- TINKERCAD

Software di simulazione e modellazione 3D di proprietà di Autodesk



- DISPLAY LCD

Modello 1: LCD 16x2

Modello 2: LCD 16x2 I2C

Il display LCD 16x2 permette la visualizzazione fino a 32 caratteri alla volta, disposti su 2 righe, 16 caratteri l'una. Caratteri di colore bianco, su sfondo blu, dotato di retroilluminazione (Default disattiva)

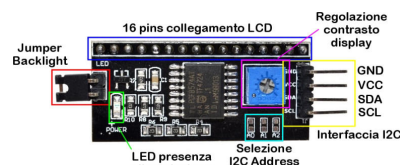
Tensione: 5V
Formato: 16x2
Peso: ~30g
Caratteri: 32
Temperatura: -10°C / +60°C
Driver: SPLC780



LCD 16x2 I2C è composto in più dal protocollo di comunicazione/interfaccia I2C

Protocollo di comunicazione/interfaccia I2C permette l'utilizzo di soltanto 2 pin del microcontrollore, sul modulo sono presenti i pin di alimentazione, quelli per l'analogico, retroilluminazione e contrasto.

Tensione: 2,5V / 6V
Protocollo: I2C
Peso: 5g
Chip: PCF8574
Risoluzione: 8 bit



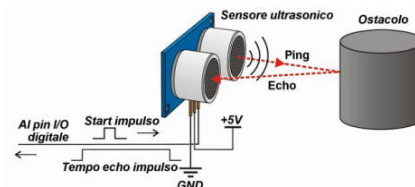
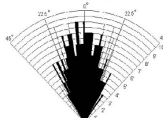
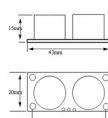
- SENSORE HC-SR04

Il suo campo di misura si estende da due centimetri a quattro metri, la precisione arriva a 1 cm, il modulo comprende il trasmettitore a ultrasuoni, il ricevitore e il circuito di controllo.

Questi sensori ad ultrasuoni non misurano direttamente la distanza, forniscono il tempo impiegato da un segnale sonoro per raggiungere un ostacolo e ritornare di nuovo al sensore.

Prodotto dalla ELECROW, può essere trovato in altri modelli e fornitori.

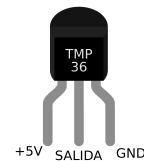
Tensione: 5V
Angolo: <30°
Max lettura: 400 cm
Min lettura: 2 cm
Frequenza: 40 kHz
Risoluzione: 1 cm



- SENSORE TPM36

Questo sensore permette di acquisire temperature comprese tra l'intervallo di -40°C e $+125^{\circ}\text{C}$ restituendo in uscita valori di tensione lineari tra 0.1Vdc e 1.7Vdc.
Il sensore ad una temperatura pari a 0°C eroga una tensione di 500mV, una variazione di grado produce una variazione della tensione pari a 10mV.

Composto da 3 PIN:
+V (2.7V to 5.5V)
Analog Output
Ground (GND)



Attenzione: il sensore non restituisce i valori in $^{\circ}\text{C}$ quindi bisogna convertirlo.

$$^{\circ}\text{C} = ((\text{valoreADC} * \text{PrecisioneADC}) - \text{TensioneZeroGradi}) / \text{stepGradoTensione}$$

$^{\circ}\text{C}$ — Temperatura in gradi Centigradi

valoreADC — Valore restituito dalla lettura

PrecisioneADC — Ottenuto dividendo le tensioni di riferimento dell'ADC (5Vdc default) e il numero massimo restituito dalla conversione (1024)
($5\text{Vdc} / 1024 = 0.00488$)

Tensione 0° — Tensione quando la temperatura rilevata è 0°

stepGradoTensione — variazione di tensione per ogni variazione di grado ($0.01 = 10 \text{ mV}$)

Codice Sorgente per questa operazione:

```
//variabili globali
int val_ADC = 0;
float temperatura = 0;

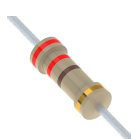
void setup(){
  //init seriale
  Serial.begin(9600);
}
```

```
//Temperatura dal sensore in celsius
//leggo dalla porta A0
val_ADC = analogRead(0);
/*converto il segnale acquisito in un valore
 espresso in gradi centigradi*/
temperatura = ((val_ADC * 0.00488) - 0.5) / 0.01;
//invio il dato sulla seriale
Serial.print( "Temperatura: " );
Serial.print(temperatura);
Serial.println( " C" );
```

Si può inoltre passare da Centigradi a Kelvin:

```
//Temperatura in Kelvin
float tempK = temperatura + 273;
Serial.print( "Temperatura: " );
Serial.print(tempK);
Serial.println( " K" );
```

- RESISTENZE (200 Ohm)



- POTENZIOMETRO (250 KOhm)



INTRODUZIONE

Prima di iniziare a entrare nel vivo dell'esercitazione e nel codice, bisogna includere e inizializzare delle librerie e dei parametri.

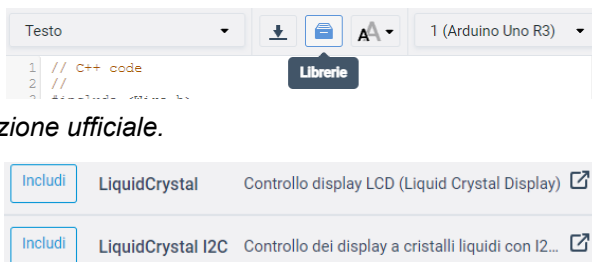
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

Notiamo subito quando andiamo a includere le 2 librerie che la libreria `LiquidCrystal_I2C.h` non esiste, o meglio viene segnalata la mancanza di questa libreria. Per includerla andiamo nel sito generale di Arduino cerchiamo la libreria da includere, ci apparirà una pagina con le specifiche e l'utilizzo di quella libreria, a fine pagina ci saranno 2 link uno dove scaricare la versione più recente e il secondo la versione precedente. Vengono date 2 versioni in caso facessimo uso di altri componenti e librerie non aggiornate.

la libreria `Wire.h` è già inclusa nell'IDE di arduino e anche nel simulatore TINKERCAD, se non lo fosse eseguiamo lo stesso procedimento elencato per l'altra libreria.

Su tinkercad c'è un modo più veloce per trovare e scaricare le librerie basta recarsi nel codice sorgente del nostro arduino simulato, guardare in alto e troveremo un pulsante denominato "Librerie":

Ci mostrerà un elenco di librerie, quando troviamo quella giusta possiamo decidere sia se includerla nel progetto oppure di andare nella sua documentazione ufficiale.



In fine scarichiamo il .zip e lo andiamo ad includere nelle librerie del nostro IDE, chiudiamo tutto e siamo pronti a lavorare con l'ausilio di queste librerie.

Dopo aver incluso queste librerie inizializziamo la libreria `LiquidCrystal_I2C.h` e gli richiamiamo il display LCD che andiamo ad utilizzare,

```
LiquidCrystal_I2C lcd(0x20,16,2);
// connessione display (pin)
```

inizializziamo i pin da utilizzare e il void setup con tutte le specifiche per i pin, monitor seriale e display.

```
int triggerPort = 7;
int echoPort = 8;

void setup() {
  // modalità dei pin
  pinMode( triggerPort, OUTPUT );
  pinMode( echoPort, INPUT );

  // inizializzazione seriale
  Serial.begin( 9600 );
  Serial.println( "Sensore ultrasuoni: " );

  // inizializzazione display lcd
  lcd.init();
  lcd.backlight();
  // stampa non modificabile
  lcd.setCursor(0,0);
  lcd.print("Distanza:");
  lcd.setCursor(0,1);
  lcd.print("Velocita:");
}
```

I sensori ad ultrasuoni, come HC-SR04, per misurare la distanza tra il sensore stesso e l'oggetto più vicino sfruttano i suoni.

Sono composti da un Emettitore e da un Ricevitore di onde sonore, lavorano a frequenze molto alte non udibili dall'uomo.

L'Emettitore genera un'onda sonora con una frequenza specifica, la emette, quando quest'onda incontra un oggetto/ostacolo rimbalza su di esso variando la sua direzione e tornando indietro verso il sensore, a quel punto il Ricevitore capta il segnale sonoro.

A questo punto il sensore misurerà il tempo che è trascorso da quando il segnale è stato generato a quando è stato ricevuto (durata).

Conoscendo il tempo impiegato dall'onda a tornare e la sua velocità, si può calcolare la distanza percorsa dall'onda:

```
//(Distanza) = (Velocità) X (Tempo)  
float D = V * T;
```

Con il microcontrollore Arduino la nostra variabile "T", il tempo di durata dell'onda per ritornare da quando generata a quando captata, la possiamo ottenere con il seguente codice:

```
int triggerPort = 7;  
int echoPort = 8;  
  
void setup() {  
  // modalità dei pin  
  pinMode( triggerPort, OUTPUT );  
  pinMode( echoPort, INPUT );  
  
  // inizializzazione seriale  
  Serial.begin( 9600 );  
  Serial.println( "Sensore ultrasuoni: " );  
}
```

```
void loop() {  
  // porta bassa l'uscita del trigger  
  digitalWrite( triggerPort, LOW );  
  // invia un impulso di 10microsec su trigger  
  digitalWrite( triggerPort, HIGH );  
  delayMicroseconds( 10 );  
  digitalWrite( triggerPort, LOW );  
  // legge il ritorno e calcola la durata  
  long duration = pulseIn( echoPort, HIGH );  
  // calcola la distanza  
  long r = 0.034 * duration / 2;  
  // stampa in seriale di durata e distanza  
  Serial.print( "durata: " );  
  Serial.print( duration );  
  Serial.print( " , " );  
  Serial.print( "distanza: " );  
  // se non nella portata minima e massima del sensore  
  if( duration > 38000 ) Serial.println( "fuori portata" );  
  // se all'interno della portata  
  else { Serial.print( r ); Serial.println( "cm" );}
```

L'onda è possibile che non rimbalzi su nessuna superficie o oggetto senza tornare verso il sensore.

La velocità del suono può essere calcolata tenendo conto delle variabili atmosferiche, come Temperatura, Umidità e Pressione.

Per esempio nell'aria il suono a una velocità che varia al variare della temperatura (aumenta all'aumento della temperatura).

Nell'esperienza abbiamo utilizzato un sensore di temperatura per compensare la precisione della distanza, così ottenendo una misura più accurata della velocità del suono nell'aria dell'ambiente.

I sensori di temperatura sono dispositivi che rilevano e misurano la variazione di temperatura, sono spesso trasduttori perché trasformano una grandezza fisica (temperatura) in una elettrica.

Solitamente sono composti da un transistor di tipo PNP in saturazione.

I transistor sono 2 diodi con un terminale comune, la tensione del terminale comune è funzione della temperatura, quindi la temperatura si ricava da quella alla quale si trova il transistor.

Il TMP36 ha diversi transistor all'interno per una misura più precisa e accurata.

DESCRIZIONE DELLE FASI DI LAVORO/PROGETTO

L'esperienza si suddivide principalmente in 3 partizioni:

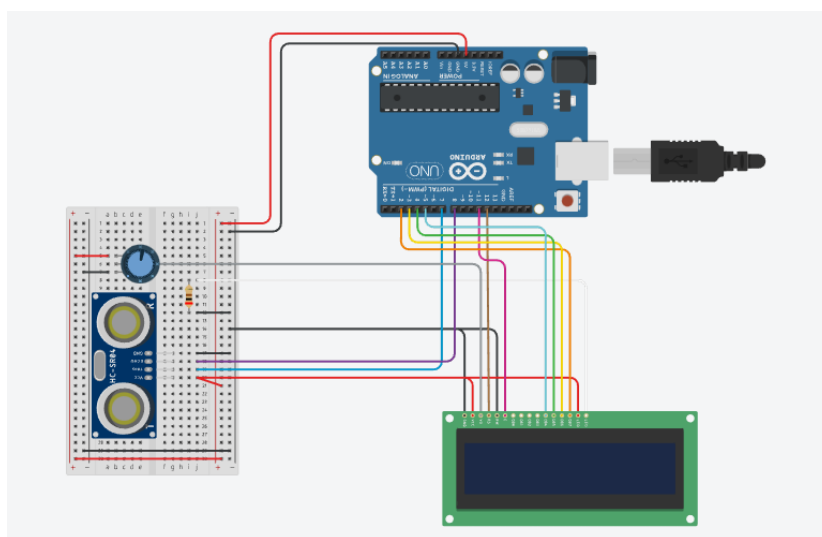
- 1: Realizzazione di un telemetro con LCD e regolazione del contrasto esterna (potenziometro)
- 2: Realizzazione di un telemetro ma con LCD ad interfaccia I2C
- 3: Realizzazione di un telemetro dotato di un sensore di temperatura

Parte 1: Telemetro con LCD 16x2

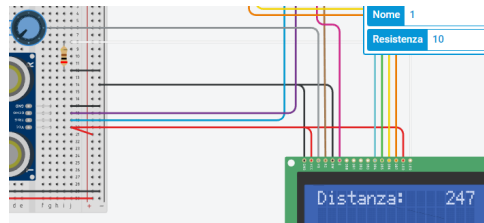
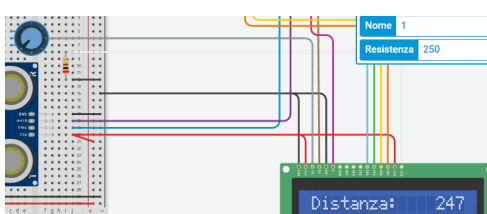
Inizialmente abbiamo realizzato su Tinkercad il telemetro con Arduino, il circuito e il display per vedere i valori dati a video.

Questo circuito è formato da:

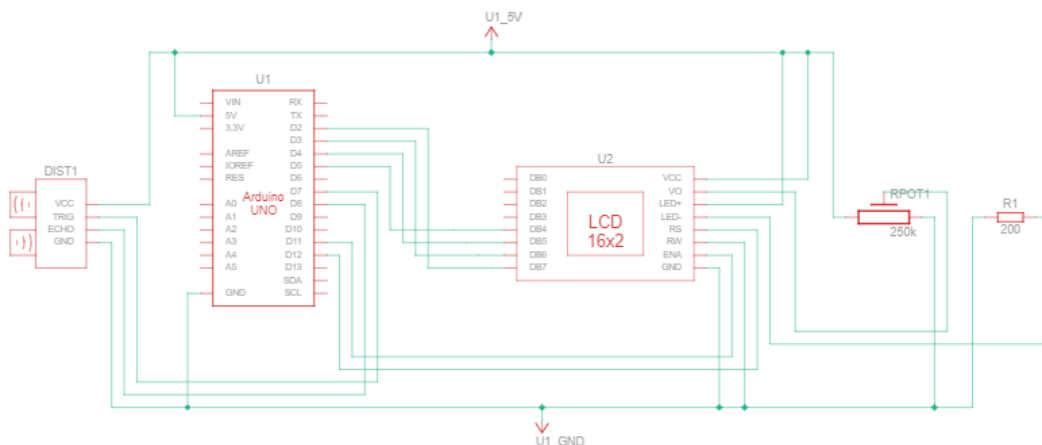
- Arduino Uno r3 programmato per gestire tutti i componenti e i dati del circuito
- LCD 16x2 per mostrare i valori calcolati e ricavati da Arduino e il circuito
- il sensore di distanza ad ultrasuoni HC-SR04
- una resistenza da 220 Ohm di "pull-down", che produce la caduta di tensione per pilotare correttamente la retroilluminazione del LCD
- un potenziometro da 250 KOhm per regolare il contrasto del LCD



Si nota che nella regolazione del contrasto con il potenziometro si trova una certa difficoltà, si ottiene una regolazione accettabile solo una minima partizione della rotazione che ci consente il potenziometro. Cambiando potenziometro e quindi diminuendo la resistenza di esso si può ottenere un arco maggiore di intervento nella regolazione del contrasto.



Schema elettrico del circuito risultante



Codice all'interno del microcontrollore
Nota: la libreria da usare è LiquidCrystal.h

```
#include <LiquidCrystal.h>

int triggerPort = 7;
int echoPort = 8;
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
// connessione display (pin)

void setup() {
  pinMode( triggerPort, OUTPUT );
  pinMode( echoPort, INPUT );
  Serial.begin( 9600 );
  Serial.println( "Sensore ultrasuoni: " );
  lcd.begin(16, 2);
  // imposto il tipo di display (colonne, righe)
  lcd.print("Distanza:");
}
```

Per visionare il circuito e vedere quanto detto
cliccare il seguente carattere ■

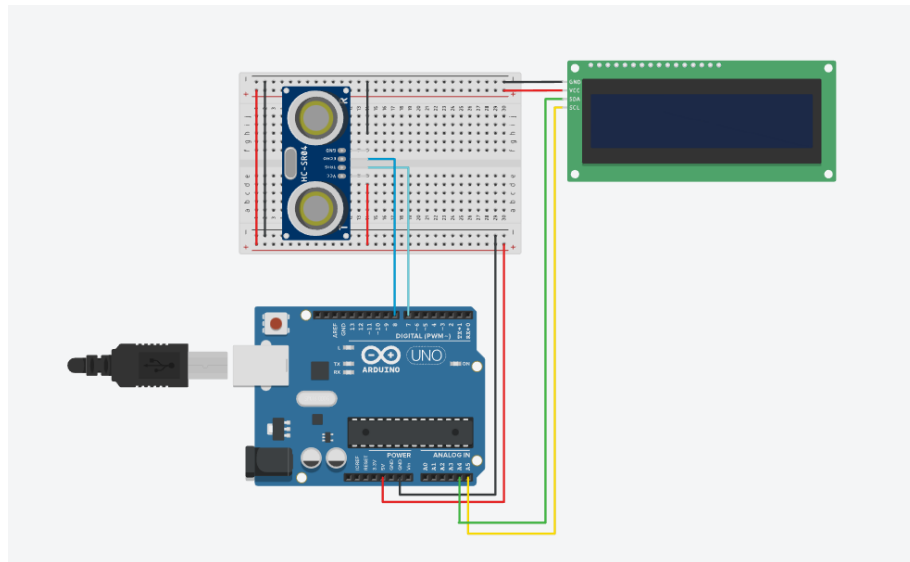
```
void loop() {
  //porta bassa l'uscita del trigger
  digitalWrite( triggerPort, LOW );
  //invia un impulso di 10microsec su trigger
  digitalWrite( triggerPort, HIGH );
  delayMicroseconds( 10 );
  digitalWrite( triggerPort, LOW );
  long duration = pulseIn( echoPort, HIGH );
  long r = 0.034 * duration / 2;
  Serial.print( "durata: " );
  Serial.print( duration );
  Serial.print( " , " );
  Serial.print( "distanza: " );
  if( duration > 38000 ) Serial.println( "fuori portata");
  else { Serial.print( r ); Serial.println( "cm" );}
  // posiziono il cursore alla colonna 12 e riga 0
  if(r > 99){
    lcd.setCursor(13, 0);
    lcd.print(r);}
  if(r <= 99 && r > 9){
    lcd.setCursor(13, 0);
    lcd.print(r);
    lcd.setCursor(15, 0);
    lcd.print(" ");}
  if(r <= 9){
    lcd.setCursor(13, 0);
    lcd.print(r);
    lcd.setCursor(14, 0);
    lcd.print(" ");}
  delay(300);
}
```


Parte 2: Telemetro con LCD 16x2 e interfaccia I2C

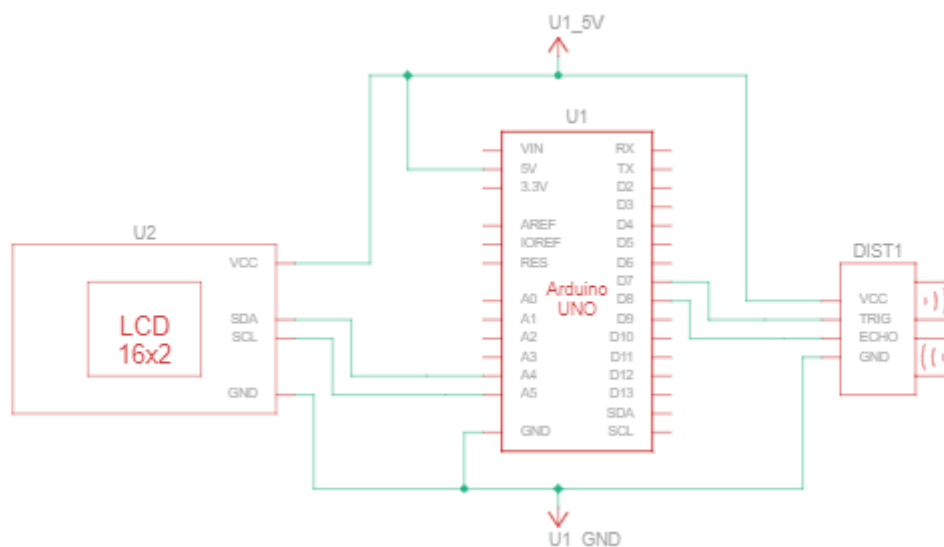
Come nel primo caso andiamo a creare il circuito sul simulatore Tinkercad.

Questo circuito è costituito dalla stessa componentistica del primo a parte LCD che ha in più l'interfaccia I2C che ci aiuta nella gestione dell'LCD quindi semplificando il circuito.

Notiamo subito che vengono usati un numero nettamente inferiore di cavi tra il display LCD e arduino, tutti i pin dell'LCD che utilizzavamo prima sono racchiusi nell'interfaccia, che ci dispone dei 2 pin di alimentazione (vcc e gnd), del pin SDA (Serial Data) e SCL (Serial Clock), in questo modo si ottiene comunque il pilotaggio completo del display, ma comporta una differente programmazione e libreria.

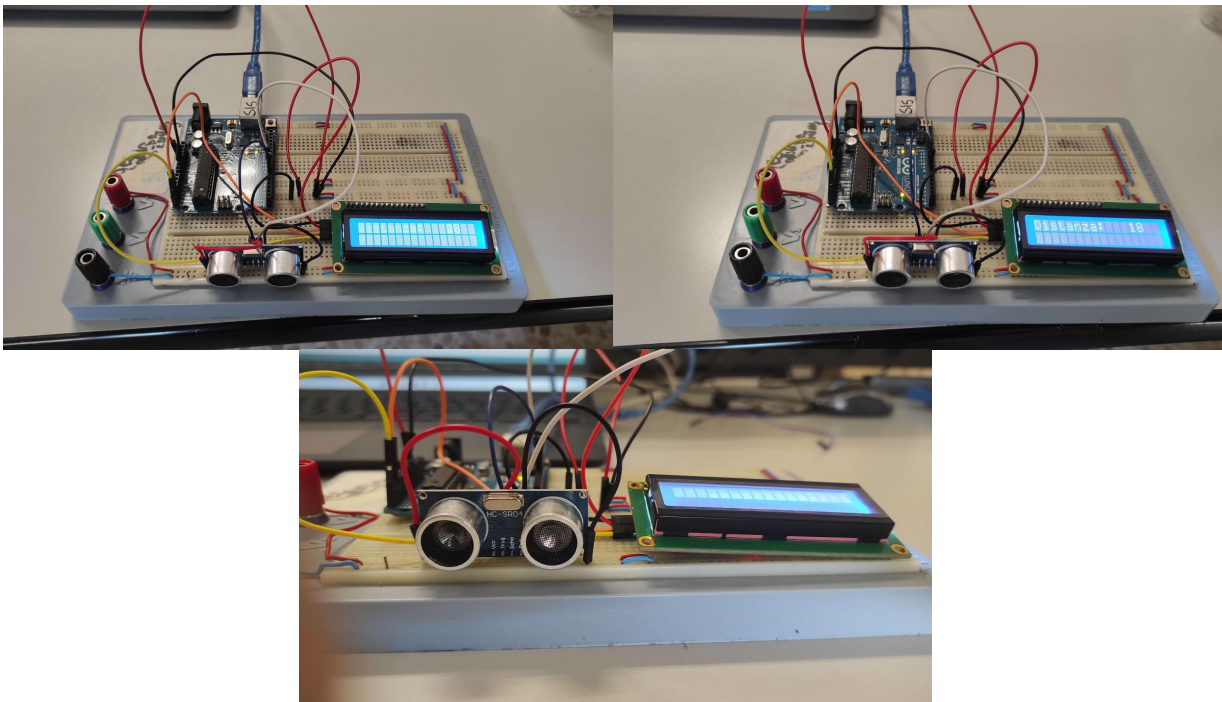


Schema elettrico del circuito risultante

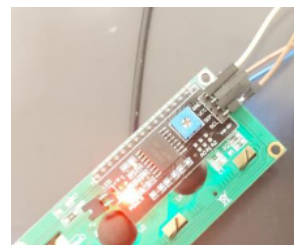
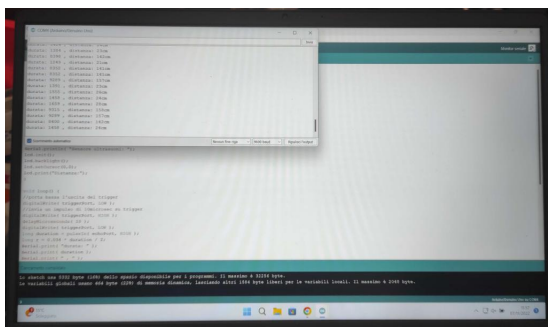


Notiamo sostanziali differenze grazie allo schema elettrico, sono stati utilizzati meno componenti e cavi, semplificando il circuito, questo è dovuto all'ausilio del display con interfaccia I2C.

Andiamo quindi a realizzare il circuito nella realtà, ricreando tutto il circuito elettrico e caricando il codice nel microcontrollore.



In quest'immagine si può vedere l'interfaccia I2C dell' LCD, e in quella sottostante l'IDE con il monitor seriale aperto.



Codice all'interno del microcontrollore

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x20,16,2);
// connessione display (pin)

int triggerPort = 7;
int echoPort = 8;

void setup() {
  // modalita dei pin
  pinMode( triggerPort, OUTPUT );
  pinMode( echoPort, INPUT );

  // inizializzazione seriale
  Serial.begin( 9600 );
  Serial.println( "Sensore ultrasuoni: " );

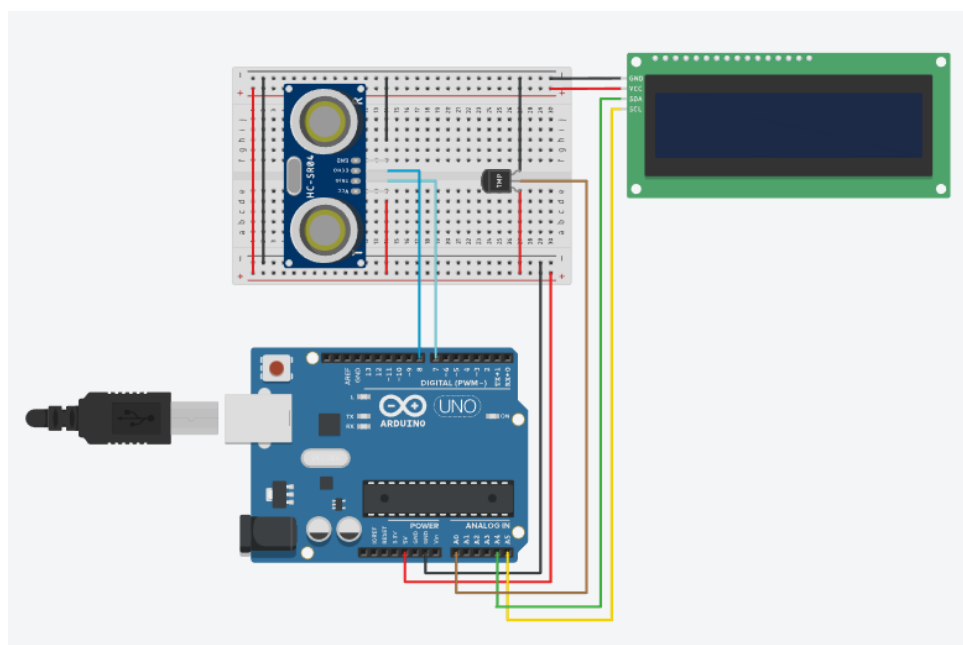
  // inizializzazione display lcd
  lcd.init();
  lcd.backlight();
  // stampa non modificabile
  lcd.setCursor(0,0);
  lcd.print("Distanza:");
}
```

Per visionare il circuito e vedere quanto detto
cliccare il seguente carattere ■

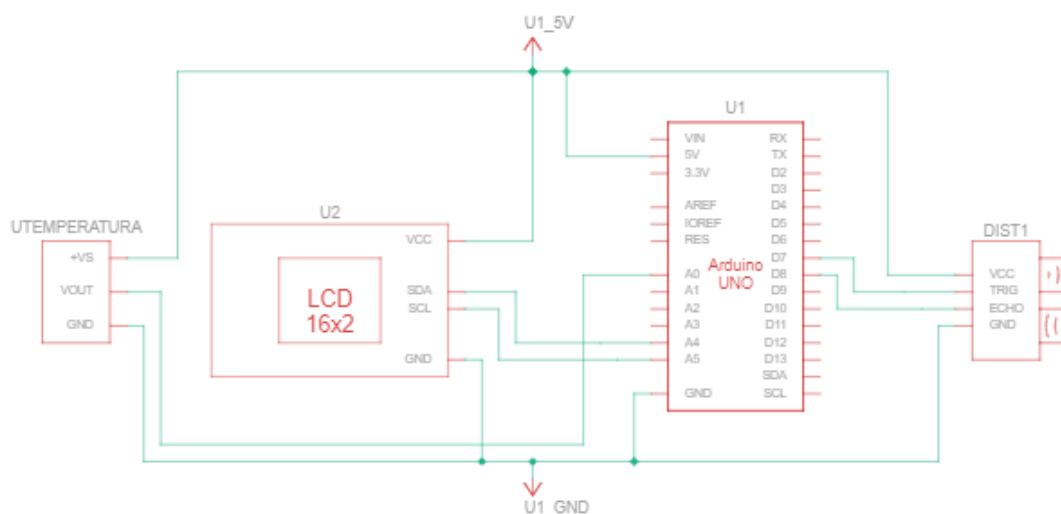
```
void loop() {
  //porta bassa l'uscita del trigger
  digitalWrite( triggerPort, LOW );
  //invia un impulso di 10microsec su trigger
  digitalWrite( triggerPort, HIGH );
  delayMicroseconds( 10 );
  digitalWrite( triggerPort, LOW );
  long duration = pulseIn( echoPort, HIGH );
  long r = 0.034 * duration / 2;
  Serial.print( "durata: " );
  Serial.print( duration );
  Serial.print( " , " );
  Serial.print( "distanza: " );
  if( duration > 38000 ) Serial.println( "fuori portata");
  else { Serial.print( r ); Serial.println( "cm" );}
  //Display con la distanza ultrasuoni
  if(r > 99){
    lcd.setCursor(11,0);
    lcd.print(r);}
  if(r <= 99 && r > 9){
    lcd.setCursor(11,0);
    lcd.print(r);
    lcd.setCursor(11,0);
    lcd.print(" ");}
  if(r <= 9){
    lcd.setCursor(11,0);
    lcd.print(r);
    lcd.setCursor(11,0);
    lcd.print(" ");}
  delay(300);
}
```

Parte 3: Telemetro con LCD 16x2 e interfaccia I2C e sensore di temperatura

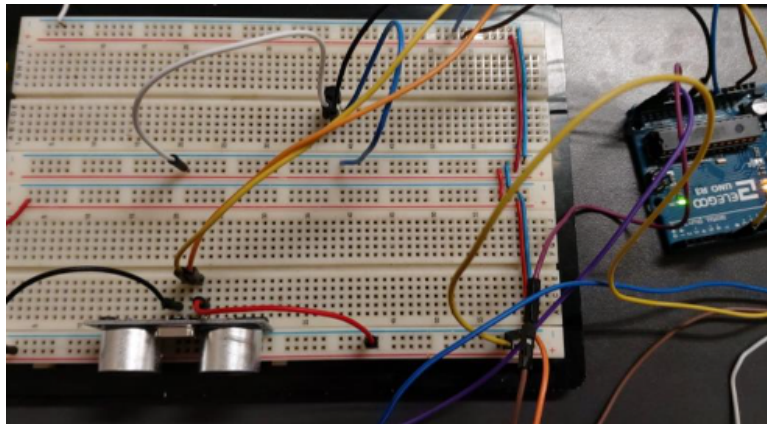
Sempre su Tinkercad abbiamo ricostruito lo stesso circuito, ricreando lo stesso telemetro ma con compensazione termica utilizzando il sensore di temperatura TPM36 in dotazione al laboratorio.



Schema elettrico del circuito risultante



Andiamo quindi a realizzare il circuito nella realtà, ricreando tutto il circuito elettrico e caricando il codice nel microcontrollore.



Codice all'interno del microcontrollore

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x20,16,2);
// connessione display (pin)

int triggerPort = 7;
int echoPort = 8;


void setup() {
  // modalità del pin
  pinMode( triggerPort, OUTPUT );
  pinMode( echoPort, INPUT );

  // inizializzazione seriale
  Serial.begin( 9600 );
  Serial.println( "Sensore ultrasuoni: " );

  // inizializzazione display lcd
  lcd.init();
  lcd.backlight();
  // stampa non modificabile
  lcd.setCursor(0,0);
  lcd.print("Distanza:");
  lcd.setCursor(0,1);
  lcd.print("Velocita:");
}
```

```
//Display con la distanza ultrasuoni
if(r > 99){
  lcd.setCursor(11,0);
  lcd.print(r);}
if(r <= 99 && r > 9){
  lcd.setCursor(11,0);
  lcd.print(r);
  lcd.setCursor(11,0);
  lcd.print(" ");}
if(r <= 9){
  lcd.setCursor(11,0);
  lcd.print(r);
  lcd.setCursor(11,0);
  lcd.print(" ");}
delay(300);
//Display con la velocità del suono
lcd.setCursor(11,1);
lcd.print(V);
}
```

```
void loop() {
  //invia un impulso di 10microsec su trigger
  digitalWrite( triggerPort, LOW );
  //invia un impulso di 10microsec su trigger
  digitalWrite( triggerPort, HIGH );
  delayMicroseconds( 10 );
  digitalWrite( triggerPort, LOW );
  long duration = pulseIn( echoPort, HIGH );
  long r = 0.034 * duration / 2;
  Serial.print( "durata: " );
  Serial.print( duration );
  Serial.print( " , " );
  Serial.print( "distanza: " );
  if( duration > 30000 ) Serial.println( "fuori portata");
  else { Serial.print( r ); Serial.println( "cm" );}
  //Velocità del suono
  //Temperatura dal sensore in celsius
  int val_ADC = analogRead(0);
  float temperatura = ((val_ADC * 0.00488) - 0.5) / 0.01;
  // oppure float temperatura = (((val_ADC*5.0)/1023.0)-0.5)*100;
  Serial.print( "Temperatura: " );
  Serial.print(temperatura);
  Serial.println( " C" );
  //Temperatura in Kelvin
  float tempK = temperatura + 273;
  Serial.print( "Temperatura: " );
  Serial.print(tempK);
  Serial.println( " K" );
  //calcolo velocità del suono
  float V = sqrt(tempK*1.41*287);
  Serial.print( "Velocita: " );
  Serial.print(V);
  Serial.println( " m/s" );
}
```

Per visionare il circuito e vedere quanto detto
cliccare il seguente carattere 

POLO TECNOLOGICO IMPERIESE ITI "G. Galilei" Informatica e Telecomunicazioni	
RELAZIONE TECNICA	Pagina 13 di 13

CONCLUSIONI E OSSERVAZIONI
<p><i>Grazie ai 3 circuiti consecutivi, si può notare che dal primo al secondo circuito utilizzando l'interfaccia I2C per il display si semplifica molto il circuito, si ottiene una gestione più semplice e pulita sia nel codice che nel circuito stesso, con una notevole riduzione di cavi e componenti.</i></p> <p><i>Inoltre si nota che nel terzo circuito con l'introduzione del sensore termico (quindi dall'aggiunta della compensazione termica), si ottiene una maggior precisione nelle misurazioni del calcolo della distanza.</i></p> <p><i>La velocità varia in proporzione della temperatura, questo è determinato dal fatto che la velocità dipende dalle condizioni atmosferiche come la temperatura, quindi la velocità è direttamente proporzionale al valore della temperatura letto dal TMP36.</i></p>