Display OLED 128 x 64

VCC ai 5V

GND a GND

SCL a A5

SDA a A4

Il display comunica con Arduino col protocollo I2C. I piedini di Arduino UNO deputati alla comunicazione I2C sono A4 (SDA) e A5 (SCL), che quindi non potranno essere usati come ingresso analogico quando si usano periferiche I2C. Per altri modelli di scheda Arduino i piedini per SDA e SCL sono diversi (es: per la scheda Arduino MEGA2650 si usano i pin 20 (SDA) e 21 (SCL).

In gestione librerie cercare SSD1306 e installare la libreria **Adafruit SSD1306**

Sempre in gestione librerie cercare GFX e installare **Adafruit GFX library**

Come sempre, insieme alle librerie verranno installati dei programmi di esempi nel menù Esempi dell'IDE di Arduino.

Caricare nell'IDE l'esempio ssd1306_128x64_i2c

Nella riga in cui si inizializza il display, occorre cambiare l'indirizzo da 0x3D a **0x3C** (non è 0x78 come stampato nel retro del display):

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
```

Caricare l'esempio su Arduino e verificare che il display funzioni.

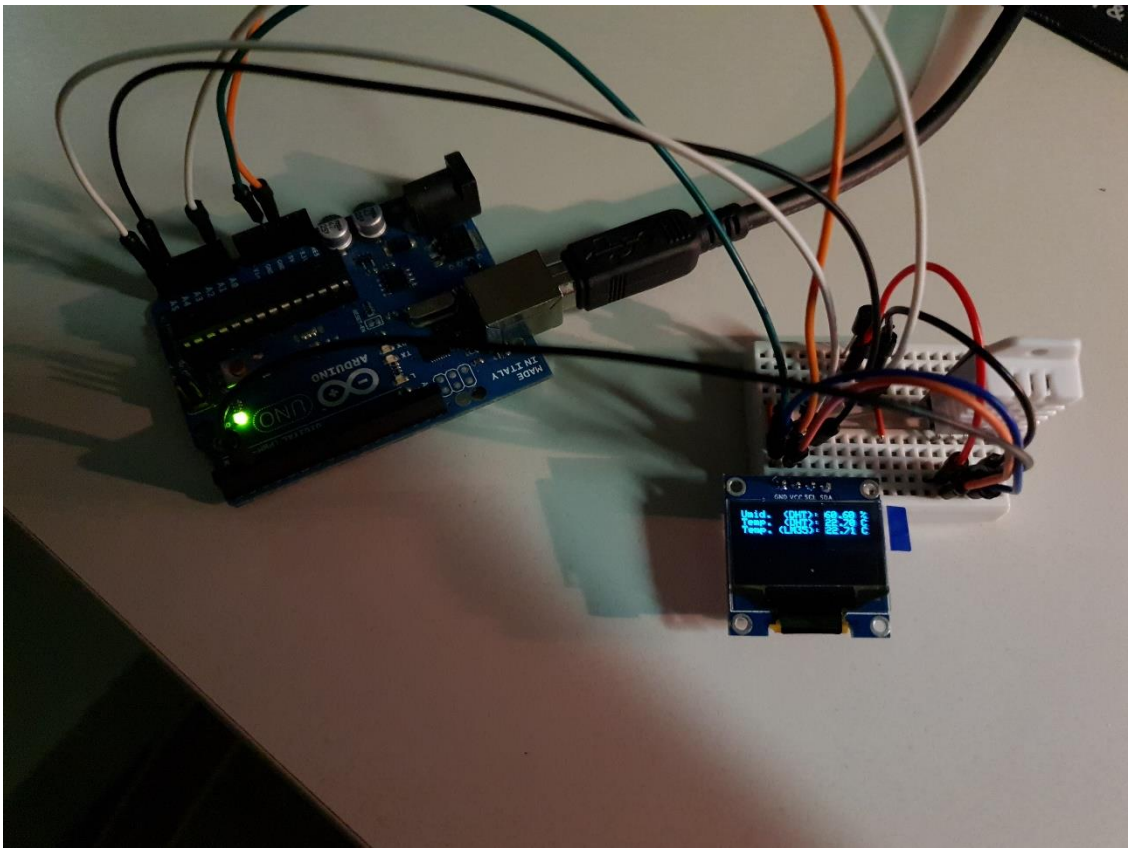
Lettura della temperatura dal **sensore umidità e temperatura digitale DHT22 (oppure DHT11)** e dal **sensore analogico di temperatura LM35** e scrittura dei valori sul display OLED

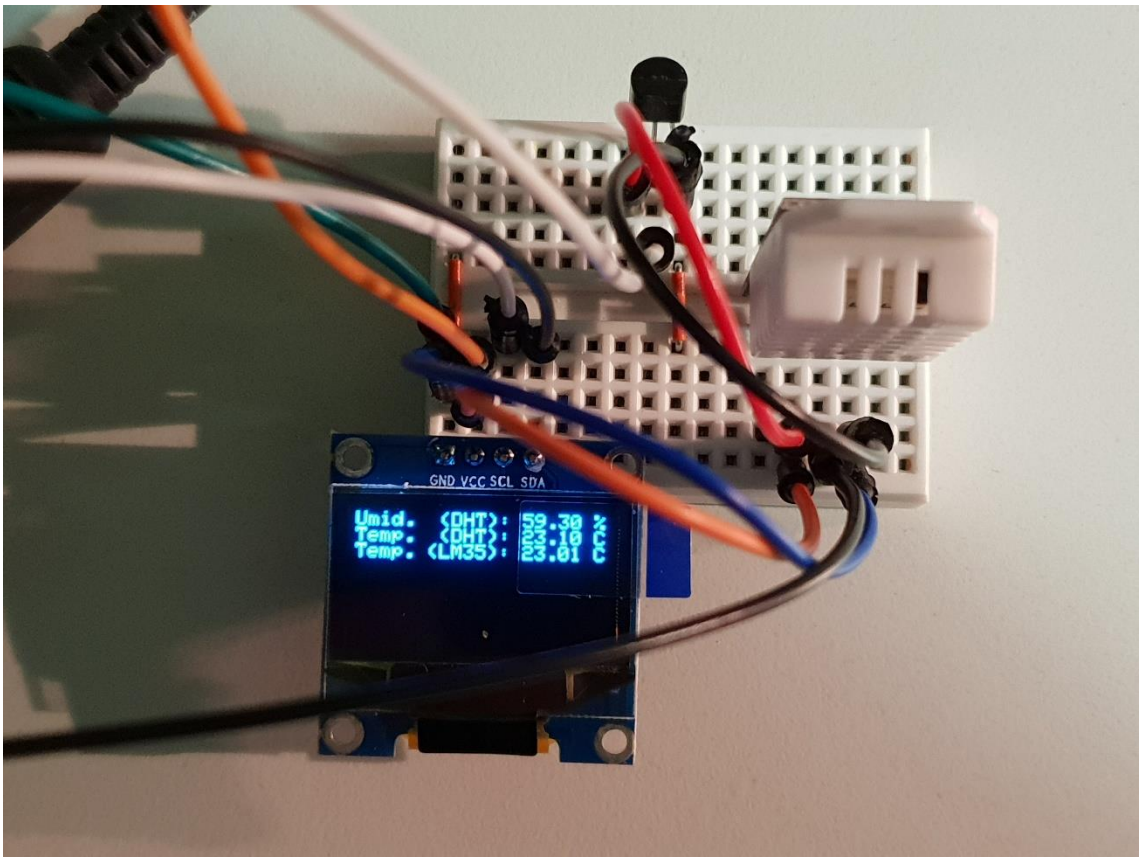
Collegamenti:

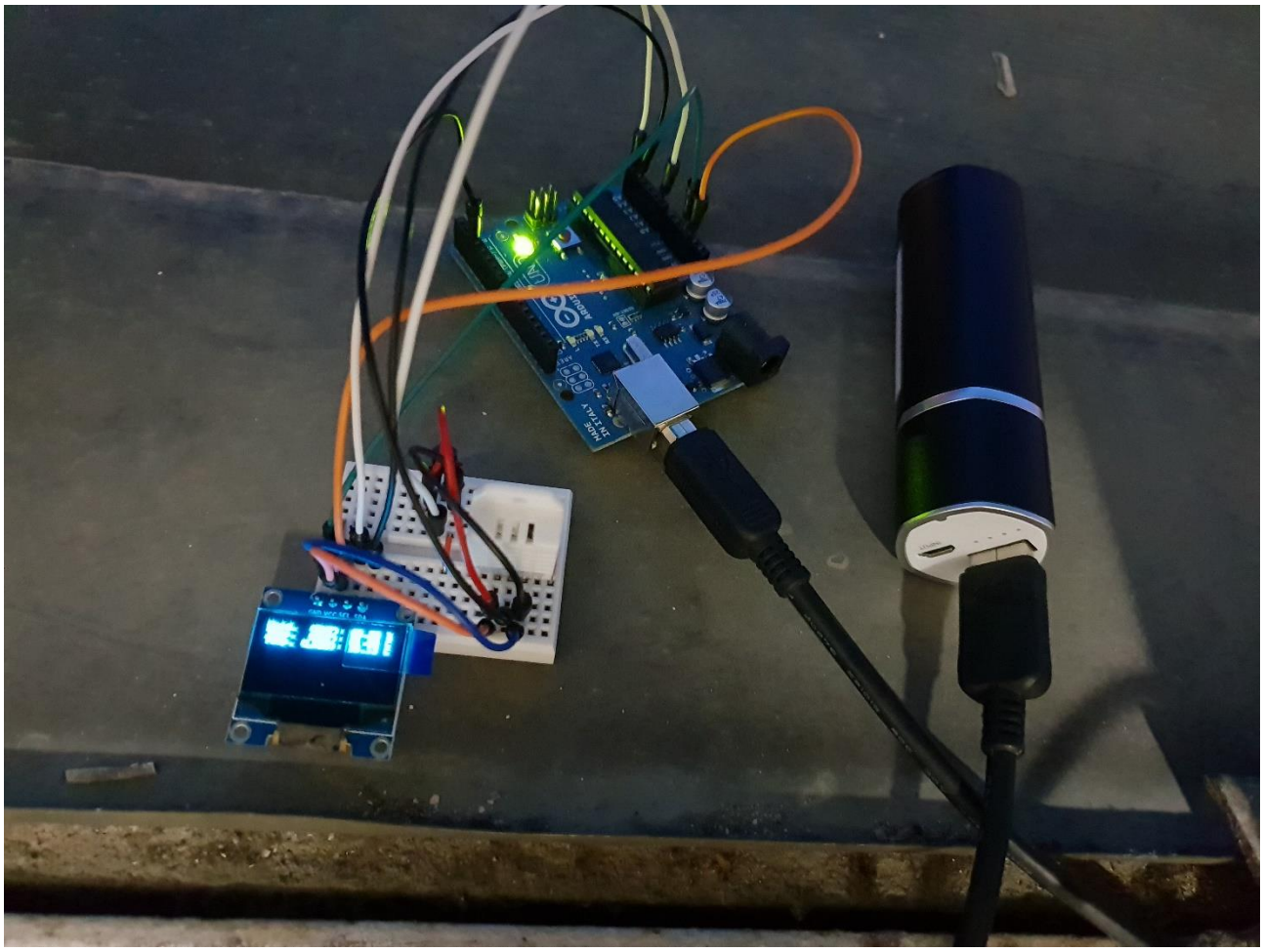
- Display OLED come sopra (I2C)

- Sensore DHT22 (o DHT11): VCC a 5V, GND a GND, pin Data a pin digitale 2 di Arduino (o altro pin, come definito nello sketch)

- Sensore LM35: guardando la parte piatta pin di sinistra a 5V, pin di destra a GND, pin centrale a ingresso analogico A0 di Arduino (o altro ingresso analogico definito nello sketch). La lettura dell'LM35 può essere effettuata o con risoluzione *standard*, se si lascia il riferimento dell'ADC a default (0..5V) o con risoluzione *aumentata* se si fissa il riferimento INTERNAL (1.1V). Eventualmente, è possibile "calibrare" leggermente il valore della tensione di riferimento nei calcoli se la temperatura letta dal LM35 risulta poco precisa.




GND VCC SCL SDA
Umid. (DHT): 59.30 %
Temp. (DHT): 23.10 °C
Temp. (LM35): 23.01 °C



```
// Comandi relativi al Display OLED
// Comandi relativi a DHT22 o 11
// Comandi relativi a LM35

#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
#define OLED_RESET     4 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#include "DHT.h"
#define DHTPIN 2     // what digital pin we're connected to
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321 // oppure DHT11, vedi esempio

DHT dht(DHTPIN, DHTTYPE);

#define LM35Pin  A0
```

```
void setup() {

  dht.begin();

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // Address 0x3D for 128x64

 // Show initial display buffer contents on the screen --
 // the library initializes this with an Adafruit splash screen.

 display.display();
 delay(500); // Pause for 2 seconds

  // Clear the buffer
  display.clearDisplay();

  // Draw a single pixel in white
  display.drawPixel(10, 10, WHITE);

  // Show the display buffer on the screen. You MUST call display() after
  // drawing commands to make them visible on screen!

  display.display();
  delay(200);

  // display.display() is NOT necessary after every single drawing command,
  // unless that's what you want...rather, you can batch up a bunch of
  // drawing operations and then update the screen all at once by calling
  // display.display(). These examples demonstrate both approaches...

  testdrawchar();   // Draw characters of the default font

  // Invert and restore display, pausing in-between
  display.invertDisplay(true);
  delay(100);
  display.invertDisplay(false);
  delay(100);


 analogReference(INTERNAL);

}


void loop() {

display.clearDisplay();


float h = dht.readHumidity();
float t = dht.readTemperature();
```

```
display.setTextSize(1);      // Normal 1:1 pixel scale
display.setTextColor(WHITE); // Draw white text
display.setCursor(0, 0);     // Start at top-left corner


display.print("Umid.  (DHT): ");
display.print(h);
display.println(" %");
display.print("Temp.  (DHT): ");
display.print(t);
display.println(" C");


int aRead = 0;
aRead = analogRead(LM35Pin);
float ref=1.02; // nominalmente 1.1
float tempC = aRead * ((ref/1024)*1000)/10;


display.print("Temp. (LM35): ");
display.print(tempC);
display.println(" C");

display.display();
delay(1000);

}


void testdrawchar(void) {

  display.clearDisplay();

  display.setTextSize(1);      // Normal 1:1 pixel scale
  display.setTextColor(WHITE); // Draw white text
  display.setCursor(0, 0);     // Start at top-left corner
  display.cp437(true);         // Use full 256 char 'Code Page 437' font


  // Not all the characters will fit on the display. This is normal.
  // Library will draw what it can and the rest will be clipped.

  for(int16_t i=0; i<256; i++) {
    if(i == '\n') display.write(' ');
    else          display.write(i);
  }

  display.display();
  delay(200);

}
```