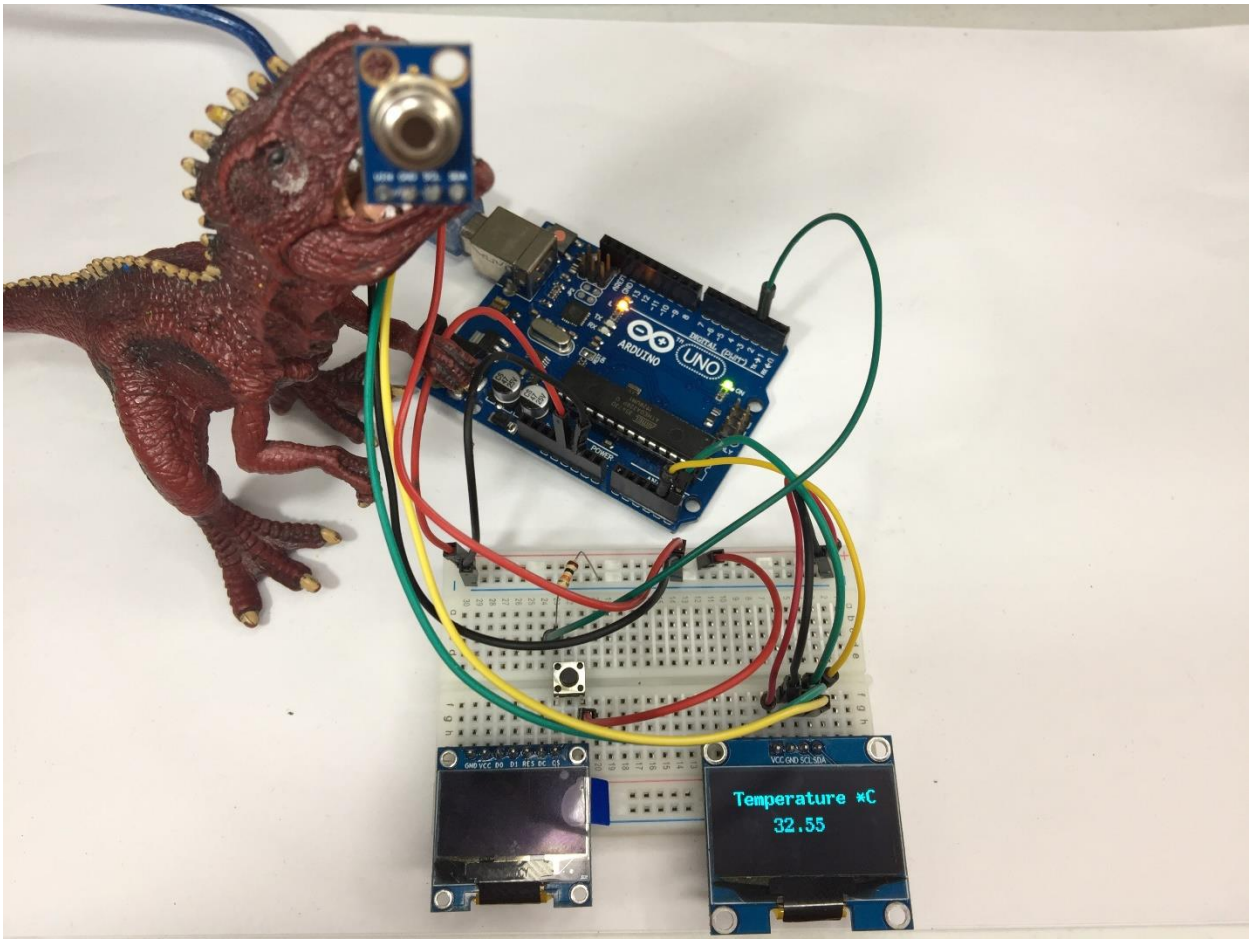
Arduino IR Temperature (Thermometer) with Display



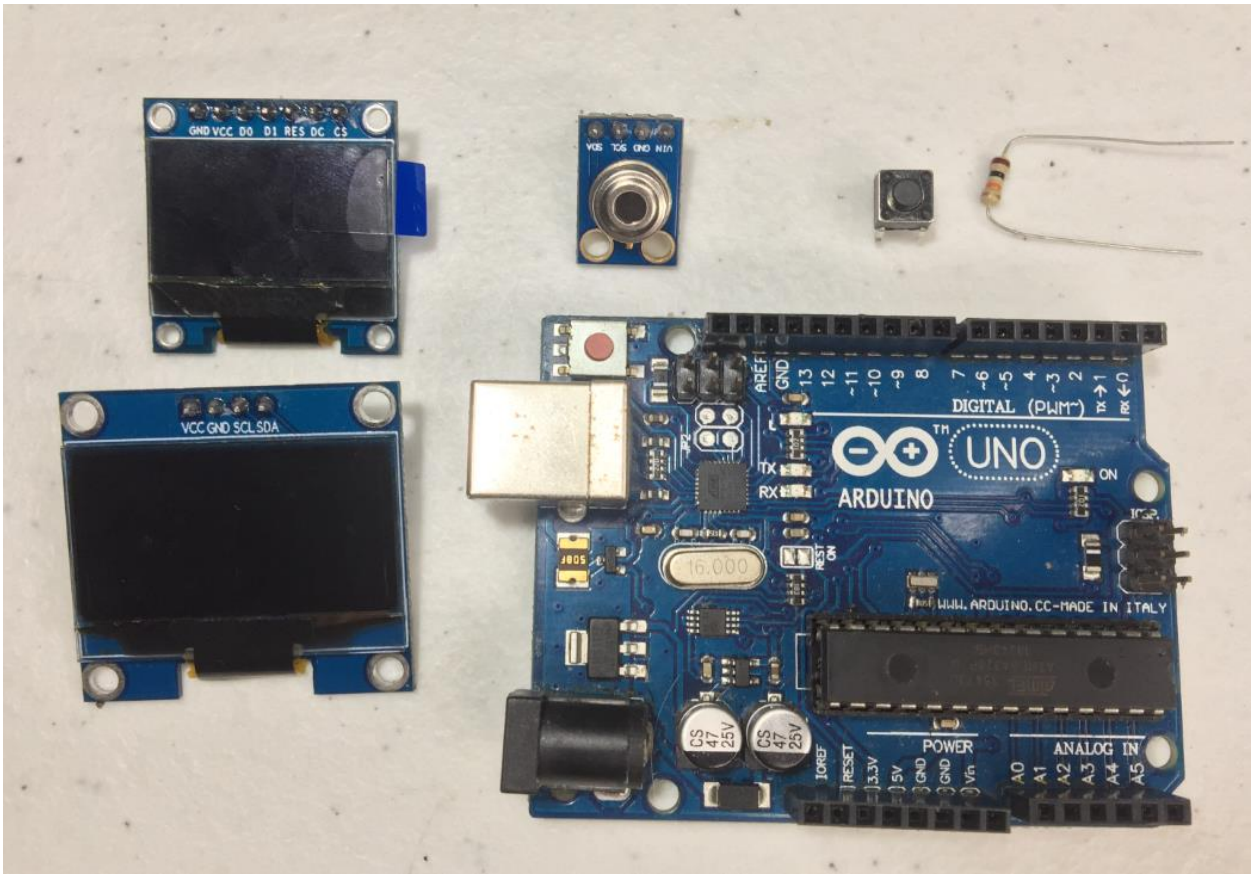
Introduction

In this project we will be making a contactless temperature monitoring circuit using an Arduino UNO, MLX90614 IR Temperature Thermometer Sensor Module and OLED Display.

Parts you will need

To create this Arduino contactless temperature monitoring project we need the following parts:

- Arduino UNO R3
- MLX90614 IR Temperature (Thermometer) Sensor module
- OLED i2c 1.3 inch - 128x64 pixel
- Small breadboard
- Tactile Switch and 10K Resistor (for pull-down)
- Connecting wires male to male
- Connecting wires male to female



Hardware Overview

The **MLX90614 IR Temperature (Thermometer) Sensor module** is very easy to use and connect to any microcontroller like an Arduino. It only has 4pins: The **VIN** which goes to 5V supply, **GND** which goes to Ground and **SDA/SCL** i2c pins. Check out the additional specification and characteristics of the sensor below:

Main Specifications:

- I2C Address **0x5A**
- Power rating: 3.3V - 5Volts
- Built-in Resistors for I2C Pull Up 10K
- Temperature -40 to 125 C
- Accuracy 0.5 degrees

Features / Specifications

- BAA range is 2cm
- BCC ranging is 10CM
- DCI range is 1M
- Using a Chip: MLX90614
- Power supply: 3-5v (internal low dropout regulator)
- Small size, low cost
- Measurement resolution of 0.02 ° C
- Temperature measurement range: -70℃～382.2℃
- Temperature measurement error: ±0.5℃ (at room temperature), resolution 0.02℃
- Working voltage: 3.3V～5V
- Operating environment temperature -40～125℃
- High precision non-contact temperature measurements
- Thermal Comfort sensor for Mobile Air Conditioning control system
- Temperature sensing element for residential, commercial and industrial building airc onditioning Windshield defogging
- Automotive blind angle detection
- Industrial temperature control of moving parts
- Temperature control in printers and copiers
- Home appliances with temperature control

- Healthcare
- Livestock monitoring
- Movement detection
- Multiple zone temperature control - up to 127sensors can be read via common 2 wires
- Thermal relay / alert
- Body temperature measurement

If you would like to explore more of this module you can go to Adafruit-MLX90614-Library link to explore it's Arduino library.

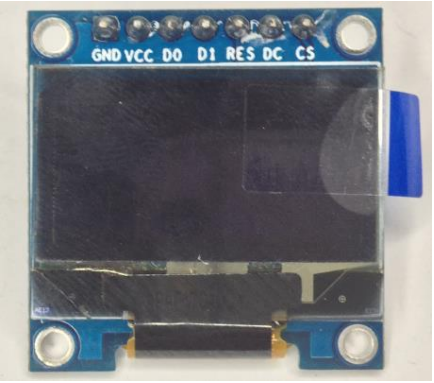
OLED Module 1.3" I2C (Inter-Integrated Circuit)

- Address 0x78
- 3.3V - 5Volts

Unlike Wire.h, the library don't need to declare the address of the OLED. Once you've declared the object below, the Arduino will know what address and communicates with it automatically.

I2C OLED is 1.3" and can be used upon declaring the object

U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);
SDA – Pin A4
SCLOCK – Pin A5
There is no need to have a 10K pull up



OLED Module .96" SPI (Serial Peripheral Interface)

Clock – Pin 13
Data – Pin 11
Chip Select – Pin 10
DC/Data Command – Pin 9
Reset – Pin 8

SPI OLED is .96” and can be used upon declaring the object

U8G2_SSD1306_128X64_NONAME_F_4W_SW_SPI u8g2(U8G2_R0, /* clock=*/ 13, /* data=*/ 11, /* cs=*/ 10, /* dc=*/ 9, /* reset=*/ 8);

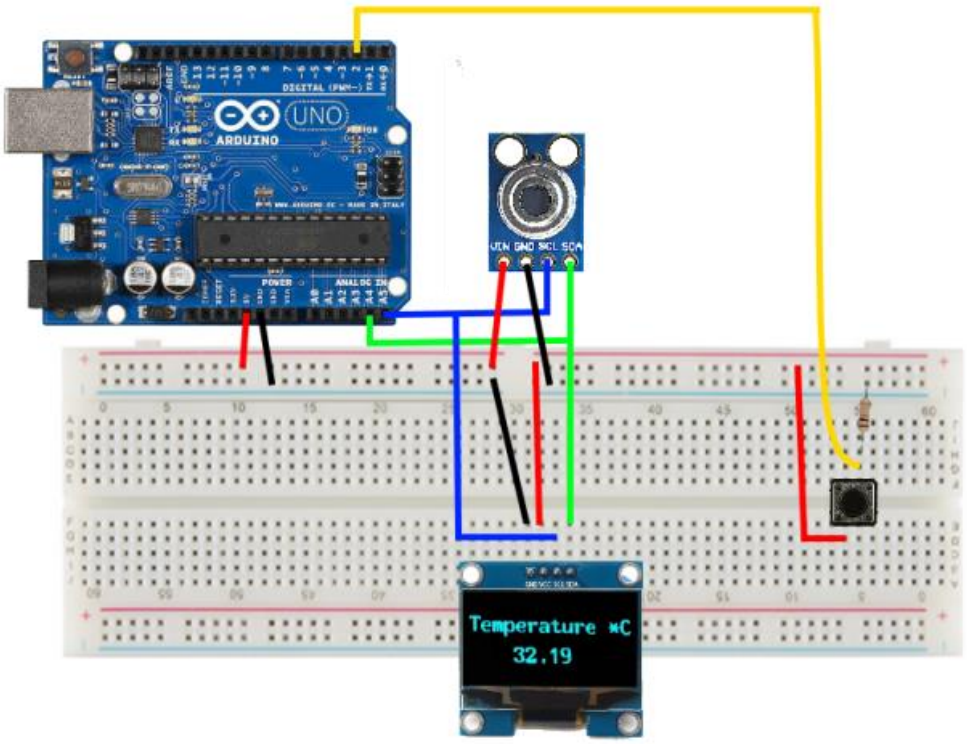
If you would like to explore more about the Arduino library
https://github.com/olikraus/u8g2

You can also install it using Library Manager

For the font, you can experiment by replacing the parameter
u8g2.setFont(u8g2_font_t0_16b_tf); // choose a suitable font

Fonts can be located at /libraries/U8g2/src/clib/u8g2_fonts.c

Wiring Diagram



Arduino Code

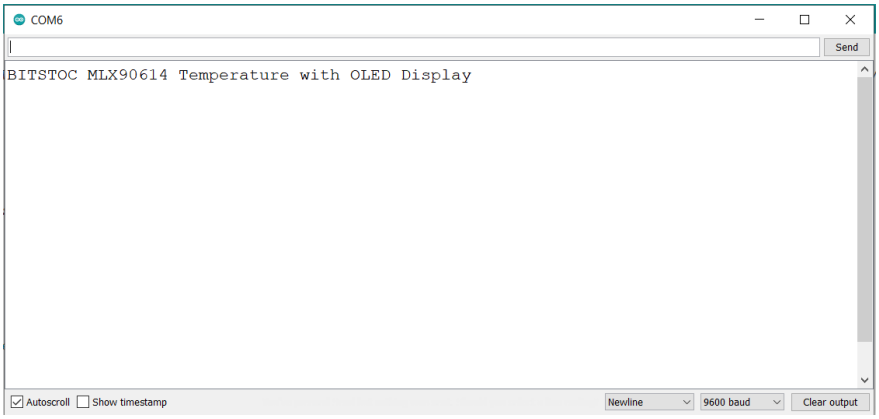
Code can be downloaded at
https://github.com/mracal/Arduino_IR_OLED_Display

```
#include <Arduino.h>
#include <U8g2lib.h>
#include <Wire.h>
#include <Adafruit_MLX90614.h>

//Wiring I2C
//  A4 - SDA
//  A5 - SCLK
//
//Wiring SPI
//  CLK   - 13
//  DATA  - 11
//  CS    - 10
//  DC    - 9
//  RESET – 8

//IR Temperature
Adafruit_MLX90614 mlx = Adafruit_MLX90614();

// Choose between 2 objects depending on the OLED Module you have
//I2C OLED Object
U8G2_SH1106_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);

//SPI OLED Object
// U8G2_SSD1306_128X64_NONAME_F_4W_SW_SPI u8g2(U8G2_R0, /* clock=*/ 13, /* data=*/ 11, /* cs=*/ 10, /* dc=*/ 9, /* reset=*/ 8);

//Button
const int buttonPin = 2;

//variable
int buttonState = 0; // push button state

void setup() {
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);

  Serial.println("BITSOC MLX90614 Temperature with OLED Display");


  u8g2.begin();
  initializeDisplay();

}
void loop() {
  buttonState = digitalRead(buttonPin);


  if (buttonState == HIGH) {
    // read temperature and display
    readAndDisplay();
  }
}

void initializeDisplay () {
  u8g2.clearBuffer();         // clear the internal memory
  u8g2.setFont(u8g2_font_t0_16b_tf);  // choose a suitable font
  u8g2.drawStr(20,20,"Temperature"); // write something to the internal memory
  u8g2.drawStr(40,40,"----");
  u8g2.sendBuffer();          // transfer internal memory to the display
  delay(500);
}
```

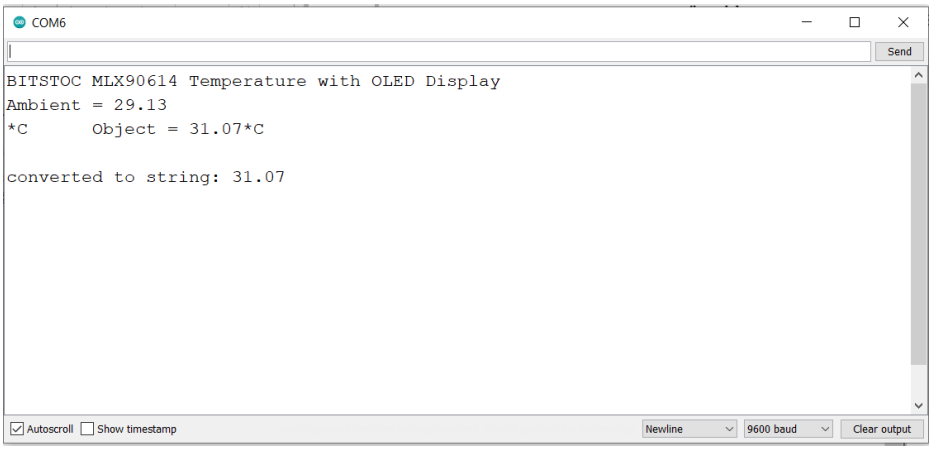
```
void readAndDisplay(){
  mlx.begin(); // This declaration must be inside the reader function
  char tempC [10];
  float tempValue = 0;
  char *pointer_to_created_string;

  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempC());
  Serial.println();
  tempValue = mlx.readObjectTempC();
  Serial.print("*C\tObject = "); Serial.print(mlx.readObjectTempC()); Serial.println("*C");
  Serial.println();
  pointer_to_created_string = dtostrf(tempValue, 4, 2, tempC);
  Serial.print("converted to string: "); Serial.println(pointer_to_created_string);

  u8g2.clearBuffer();         // clear the internal memory
  u8g2.setFont(u8g2_font_t0_16b_tf);  // choose a suitable font
  u8g2.drawStr(10,20,"Temperature *C"); // write something to the internal memory
  u8g2.drawStr(40,40,pointer_to_created_string);  // write something to the internal memory
  u8g2.sendBuffer();          // transfer internal memory to the display
  delay(500);
}
```

Output

Open Serial monitor.

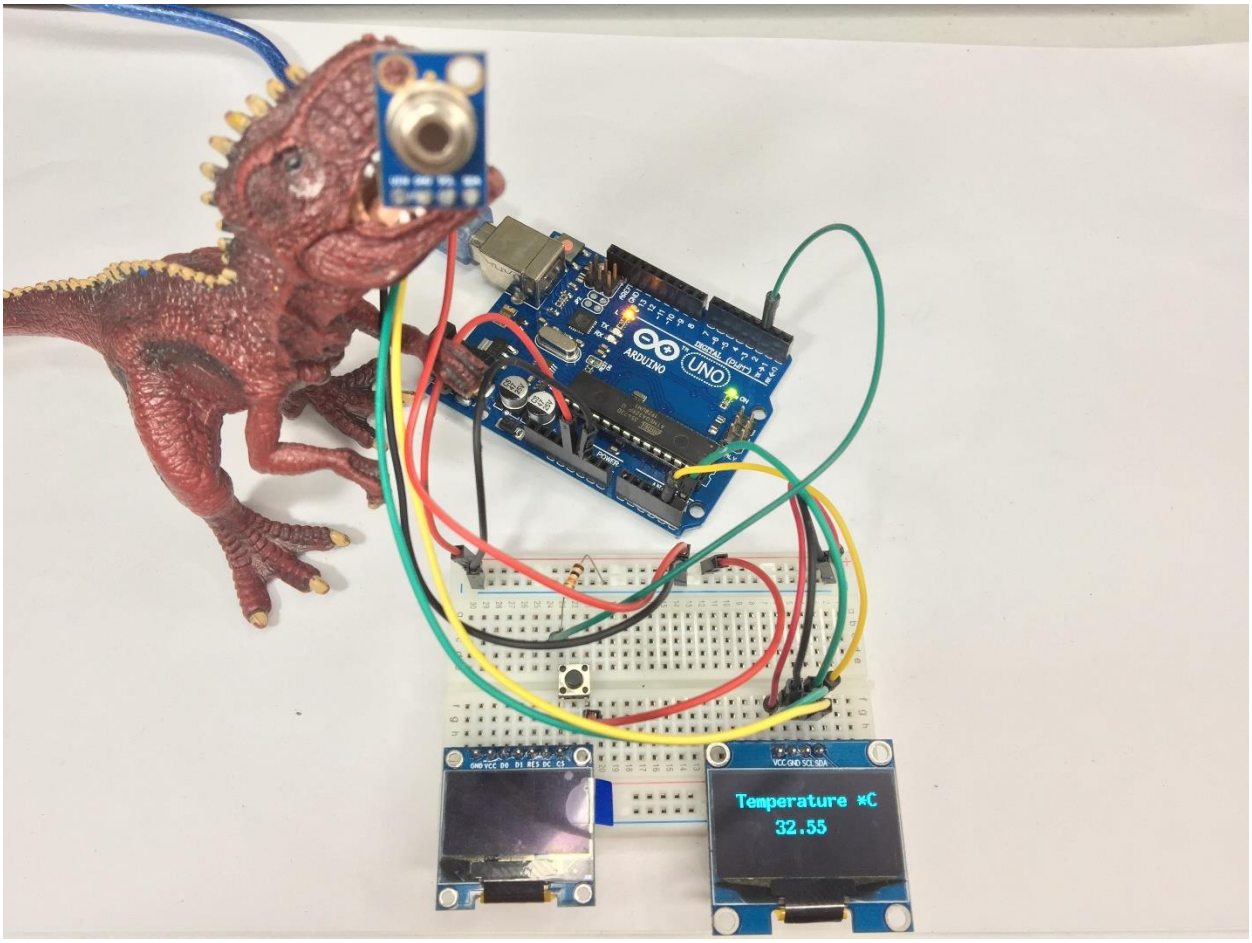


Click the button on the breadboard, the Serial Monitor should display the same as the picture below. *Converted to string is a debugging code that is being used to monitor the conversion from char * to String*.



Every time you push the button, the system will read a new temperature.

The output should look like this (*minus the Dinosaur since it is being used to hold the MLX90614 Temperature Module*).



Going Further

You can build a frame and make your own Thermal Gun Scanner or you can connect this with ESP-based Microcontroller and send the reading to Blynk or Cayenne for full IOT implementation.