**Introduction**

This project emulates an Amphenol Advanced Sensors T6700 series sensor using the I2C protocol. The Board can be driven by the 5V circuit as the sensor would be.

**Bill of Materials**

Arduino UNO compatible board or similar

0.96” OLED SPI Screen if visible output required

A potentiometer (10kΩ min, 50kΩ max, I used a 22kΩ)

Switches to change range

Box and terminals to mount the emulator

**Connection Detail**

Connection detail is for an Arduino UNO or compatible, adjust if you are using a different processor. The power for the screen in this setup is delivered and switched from output 13, so will need be accommodated elsewhere if changed.

I2C Output on Terminals SDA/SCL, responds as a T67x3 series CO2 Module

0.96" SPI OLED or similar wired as:

Screen Arduino

Gnd Gnd

Vcc 13

SCL 12

SDA 11

RES 10

DC 9

10kΩ ~ 50kΩ potentiometer wired between 5V and ground, output to A0

Multiplier Circuit Switches, standard output is 0 - 1024ppm

Take Digital Input 2 to ground to multiply by 2 (0 - 2048)

Take Digital Input 3 to ground to multiply by 4 (0 - 4096)

Take Digital Input 4 to ground to multiply by 8 (0 - 8192)

Power can either be derived from the Serial USB connection, or a regulated 5V supply to Vin and GND, or an unregulated supply into the power socket. Ground connections should be common.

**Loading the code in the Arduino**

Download and install the Arduino IDE from Arduino.cc

Set up a directory ‘Arduino’ in your ‘My Documents Folder.

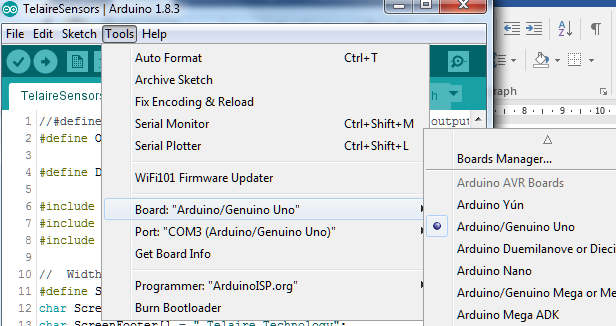
From https://github.com/AmphenolAdvancedSensors/Telaire/tree/T6700\_Series download all ‘T67xx\_Emulator‘ files into My Documents/Arduino/ T67xx\_Emulator directory.

From within the directory T67xx\_Emulator, double click the T67xx\_Emulator.ino file, this should open the file in Arduino IDE program.

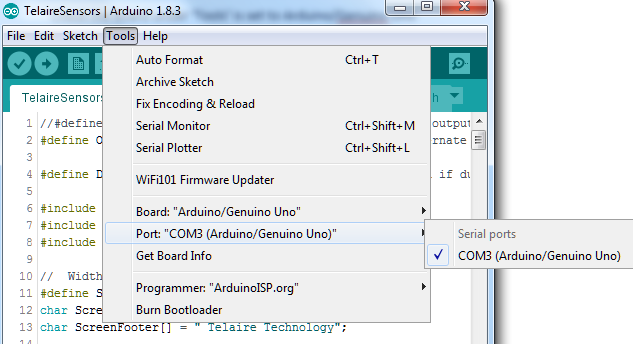
Plug in the Arduino Evaluation Kit to be modified using the USB connector.

In File >Preferences ensure your Sketchbook location is set to the folder you created (C:\Users\*your local name*\Documents\Arduino)

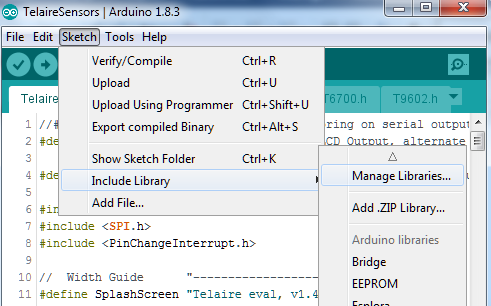
Check the Board under ‘Tools’ is set to Arduino/Genuino Uno



Check your port is connected.

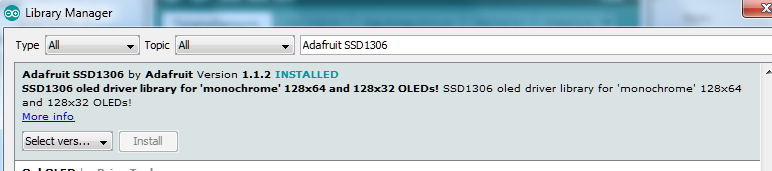


Check the Adafruit SSD1306 Library is installed. And install if it is not. Go to Sketch > Include Library > Manage Libraries

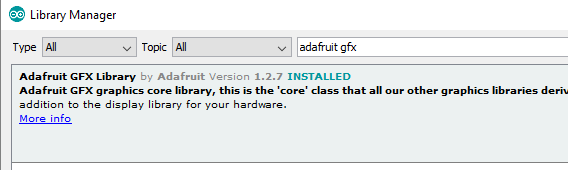


Index will be created (this may take a little time, watch bar at bottom)

Filter Adafruit SSD1306, if it is not there, add latest version.



Then repeat library addition process to add the Adafruit GFX library, sometimes this is suggested as part of SSD1306 library install.



The ‘Wire’ library also need to be added by the same method if not pre-installed.

Close the library window.



Upload file to UNO

If everything is set, the file should get uploaded and the UNO auto reboots, the I2C connection will respond with the displayed value.

**Copy of the Code**

/\* Arduino UNO or compatible Board

\* I2C Output on Terminals SDA/SCL, responds as a T67x3 series CO2 Module

\*

\* 0.96" SPI OLED or similar wired as:

\* Gnd - Gnd

\* Vcc 13

\* SCL 12

\* SDA 11

\* RES 10

\* DC 9

\*

\* 10kΩ ~ 50kΩ potentiometer wired between 5V and ground, output to A0

\*

\* Multiplier Circuit Switches, standard output is 0 - 1024ppm

\* Take Digital Input 2 to ground to multiply by 2 (0 - 2048)

\* Take Digital Input 3 to ground to multiply by 4 (0 - 4096)

\* Take Digital Input 4 to ground to multiply by 8 (0 - 8192)

\*/

// Include libraries for display and communications

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#include <Wire.h>

#define version "T67xx I2C Emulator"

#define I2CAddress 0x15 //default i2c for T67xx sensor

//(SCREEN\_WIDTH, SCREEN\_HEIGHT, OLED\_MOSI, OLED\_CLK, OLED\_DC, OLED\_RESET, OLED\_CS)

Adafruit\_SSD1306 display(128, 64, 11, 12, 9, 10, 8);

#define VccPin 13

void display\_prepare();

void displayReading(int V);

void requestEvent();

int CO2Value = 400;

byte multiplier = 1;

void setup(void) { //-------------------------SetUp--------------------------

pinMode(VccPin, OUTPUT); //Switches Screen On

digitalWrite(VccPin, HIGH);

pinMode(2, INPUT\_PULLUP); //2 x multiplier

pinMode(3, INPUT\_PULLUP); //4 x multiplier

pinMode(4, INPUT\_PULLUP); //8 x multiplier

display.setRotation(0);

display\_prepare();

display.setCursor(10, 30);

display.print(version);

display.display(); delay(1500);

Wire.begin(I2CAddress);// join i2c bus with address

Wire.onRequest(requestEvent); // register event

}

void loop(void) {//------------------------Loop---------------------------

multiplier = 1;

CO2Value = analogRead(A0);

if (!digitalRead(2)) multiplier = 2;

if (!digitalRead(3)) multiplier = 4;

if (!digitalRead(4)) multiplier = 8;

CO2Value = CO2Value \* multiplier;

displayReading(CO2Value);

delay(500);

}

//-----------------------------------------Display Routines--------------------

void display\_prepare(void) {

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

}

void displayReading(int V) {

float Variable;

//char SensorReading[4] = itoa(data[1] | data[0] << 8);

display\_prepare();

display.setCursor(0, 5); display.print(F("Emulated Value"));

display.setCursor(100, 55); display.print(F("ppm"));

display.setTextSize(4);

display.setCursor(25, 22); display.print(V);

display.display();

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_I2C RESPONSE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// function that executes whenever data is requested by master

// this function is registered as an event, see setup()

void requestEvent() {

byte data[4];

data[0] = (0); //MSB

data[1] = (0); //LSB

data[2] = ((CO2Value & 0xFF00) >> 8); //MSB

data[3] = (CO2Value & 0x00FF); //LSB

Wire.write(data[0]);

Wire.write(data[1]);

Wire.write(data[2]);

Wire.write(data[3]);

}

To verify all is OK press the tick. cid:image004.jpg@01D30243.D7BC90C0, the sketch (program) should compile without errors

To modify the sketch to send values to the PC rather than display them on the OLED then the top lines of code need be modified from default:

//#define USB //enable or disable monitoring on serial output, alternate with display, 115200 baud

#define OELCD\_OP //enable or disable LCD Output, alternate with serial output

To:

#define USB //enable or disable monitoring on serial output, alternate with display, 115200 baud

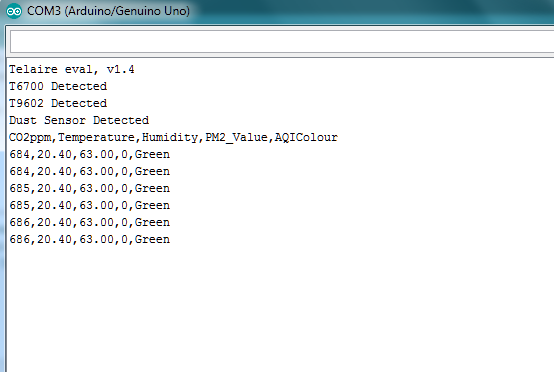
//#define OELCD\_OP //enable or disable LCD Output, alternate with serial output

This remarks ‘out’ the OLED and remarks ‘in’ the USB connection. Changing it back reinstates the OLED and switches off the serial output.

Upload (The Arrow pointing right on icon bar). Program should then compile and upload. cid:image004.jpg@01D30243.D7BC90C0

Open the serial window (magnifying glass icon top right) cid:image005.jpg@01D30243.D7BC90C0

Set the baud rate to 19200 in the bottom right, and you should see something like this:



The values are now being transmitted to the serial port and can either be data logged with program like coolterm, or operated on as you wish.

To set the evaluation kit to display values again reverse the commented lines to read as default:

//#define USB //enable or disable monitoring on serial output, alternate with display, 115200 baud

#define OELCD\_OP //enable or disable LCD Output, alternate with serial output

This remarks ‘in’ the OLED and remarks ‘out’ the USB connection. The software sketch then needs to be downloaded to Arduino.