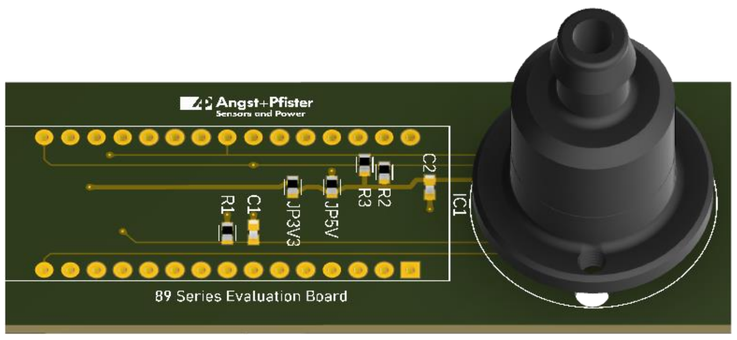
User Manual for the 89 Series Pressure Sensors



Content

1 Disclaimer

Angst + Pfister Sensors and Power
Thurgauerstrasse 66
8050 Zürich
Switzerland

Phone +41 44 877 35 00
sensorsandpower@angst-pfister.com
https://sensorsandpower.angst-pfister.com

Michel Krapf, Zurich, Release 07.2023

2 Introduction

The kit presented here is designed to help implement 89 series sensors in your design. To simplify this process, this kit provides you with functioning communication with the sensor through serial (USB). One version of the code for the I^{2}C communication is being provided to help understand possibilities to interact with sensors. The code can be altered and rewritten to test different scenarios.

Additional Information:
It is possible to use other communication protocols with the PCB, such as SPI, one wire interface, analog voltage output. However, in this manual the focus lies on sensors with I^{2}C protocols.

3 Package Content

1x Arduino Nano
1x PCB with sensor
1x Micro – USB-A Cable
1x Software Pack

4 Setup

4.1 Assembly

The sensor must be soldered correctly onto the PCB. The orientation is indicated by a header with the name “IC1”. Additionally, there is a wider white line outlining the place where the sensor should be soldered on.

4.2 Connection

Connect the PCB to the Arduino board. Then connect the Arduino to a computer.

4.3 Arduino Code Editor

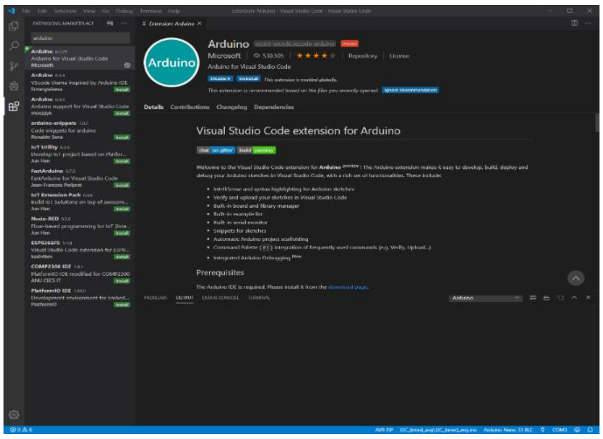
The Arduino board can be programmed in multiple ways. For a quick start three options are listed here, while others will work just as well.

4.3.1 Arduino Create

An easy way to program the board is https://create.arduino.cc/ , since there is no full installation needed. The code can be accessed directly through the following links:
I2C_timed_avg:
https://create.arduino.cc/editor/JaPew/2c243570-574d-40c9-8020-c827dd7b3be0/preview
I2C_timed_request:
https://create.arduino.cc/editor/JaPew/23f06c91-94aa-4057-9dbe-ba5980208e5d/preview

The Website guides through how to install a plugin and then code can be run on the Arduino. However, there are limits on free compilation-time.

4.3.2 Visual Studio Code



Visual Studio Code extension for Arduino

Welcome to the Visual Studio Code extension for **Arduino** preview ! The Arduino extension makes it easy to develop, build, deploy and debug your Arduino sketches in Visual Studio Code, with a rich set of functionalities. These include:

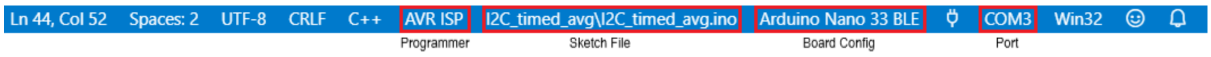
- IntelliSense and syntax highlighting for Arduino sketches
- Verify and upload your sketches in Visual Studio Code
- Built-in board and library manager
- Built-in example list
- Built-in serial monitor
- Snippets for sketches
- Automatic Arduino project scaffolding
- Command Palette (F1) integration of frequently used commands (e.g. Verify, Upload...)
- Integrated Arduino Debugging New

Prerequisites

The Arduino IDE is required. Please install it from the download page.

Figure 1: The "Arduino" extension must be installed for VSC to be used to upload code to an Arduino board.

Another option is “Microsoft Visual Studio Code” (not to be confused with “Microsoft Visual Studio”). It offers more of a classical IDE. Visual Studio Code (VSC) can be downloaded here: https://code.visualstudio.com/download
After VSC is installed, an extension needs to be added. The extension allows VSC to communicate directly with the Arduino and updates the Arduino Libraries.

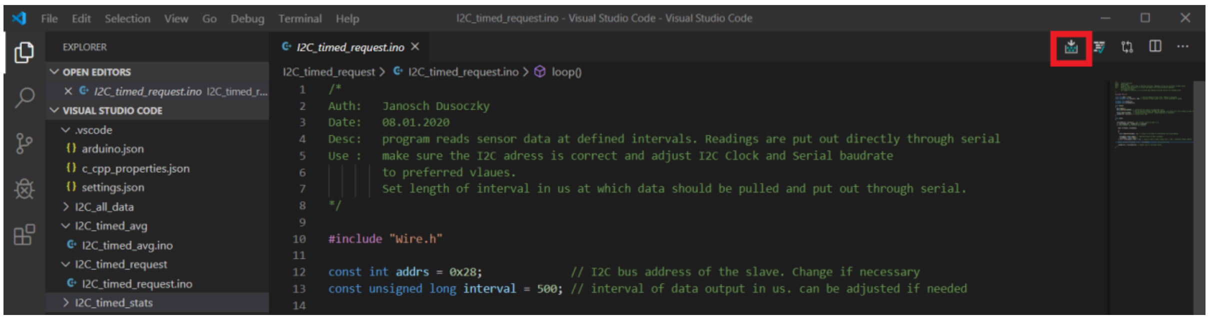


```
Ln 44, Col 52   Spaces: 2   UTF-8   CRLF   C++   AVR ISP   I2C_timed_avg\I2C_timed_avg.ino   Arduino Nano 33 BLE   COM3   Win32
                                                 Programmer          Sketch File                   Board Config         Port
```

Figure 2: Settings in VSC that need to be checked and adjusted.

After installing the extensions, the blue bottom ribbon allows for some new settings. As “programmer” we recommend “AVR ISP”. When multiple Code Files are opened, “Sketch File” can be changed to whatever Sketch needs to be uploaded to the Arduino. Board Config needs to be changed to the Arduino Board used. The serial port needs to be changed to the port the Arduino is connected to.

Finally, code can be compiled and uploaded to the Arduino by pressing “Arduino: Upload” in the top right corner or pressing “Ctrl+Alt+U”.



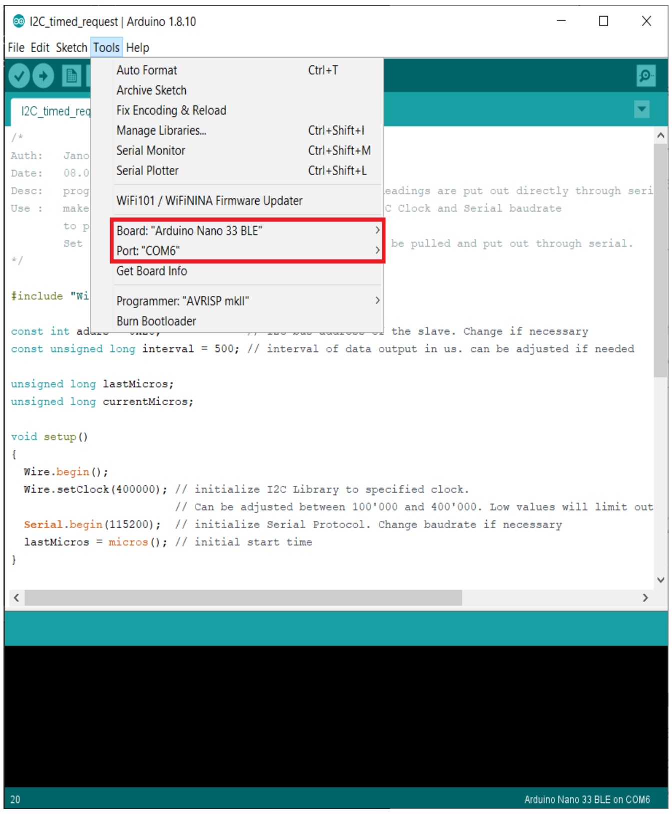
```
/*
Auth:   Janosch Dusoczky
Date:   08.01.2020
Desc:   program reads sensor data at defined intervals. Readings are put out directly through serial
Use :   make sure the I2C adress is correct and adjust I2C Clock and Serial baudrate
        to preferred vlaues.
        Set length of interval in us at which data should be pulled and put out through serial.
*/

#include "Wire.h"

const int addrs = 0x28;               // I2C bus address of the slave. Change if necessary
const unsigned long interval = 500; // interval of data output in us. can be adjusted if needed
```

Figure 3: View of VCS with button to upload code to Arduino board highlighted.

4.3.3. Arduino IDE


I2C_timed_request | Arduino 1.8.10
File Edit Sketch Tools Help
I2C_timed_req
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Get Board Info
Burn Bootloader

Board: "Arduino Nano 33 BLE"
Port: "COM6"
Programmer: "AVRISP mkII"

```
/*
Auth:   Jano
Date:   08.0
Desc:   prog                                        eadings are put out directly through seri
Use :   make                                        C Clock and Serial baudrate
        to p
        Set                                          be pulled and put out through serial.
*/

#include "Wi

const int ad                                  the slave. Change if necessary
const unsigned long interval = 500; // interval of data output in us. can be adjusted if needed

unsigned long lastMicros;
unsigned long currentMicros;

void setup()
{
  Wire.begin();
  Wire.setClock(400000); // initialize I2C Library to specified clock.
                         // Can be adjusted between 100'000 and 400'000. Low values will limit out
  Serial.begin(115200);  // initialize Serial Protocol. Change baudrate if necessary
  lastMicros = micros(); // initial start time
}
```

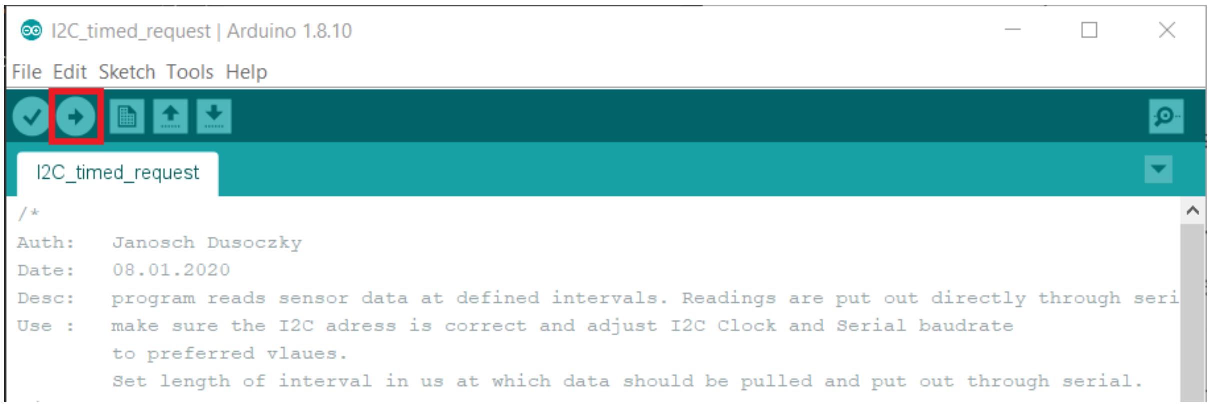
```
20                                                        Arduino Nano 33 BLE on COM6
```

Figure 4: Settings that need to be checked and adjusted.

The last option to be presented here is the Arduino IDE. It can be downloaded here: https://www.arduino.cc/en/Main/Software

After installation, the Arduino board has to be chosen and the serial port the Arduino is connected to needs to be defined.

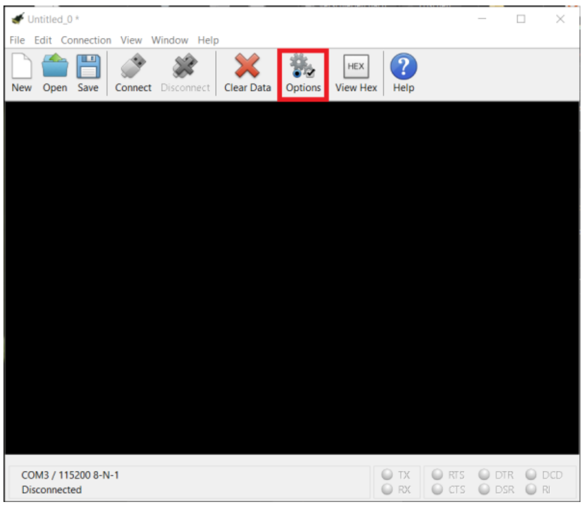
Finally, code can be compiled and uploaded to the Arduino by pressing “Upload” in the top left corner.


I2C_timed_request | Arduino 1.8.10
File Edit Sketch Tools Help
I2C_timed_request

```
/*
Auth:    Janosch Dusoczky
Date:    08.01.2020
Desc:    program reads sensor data at defined intervals. Readings are put out directly through seri
Use :    make sure the I2C adress is correct and adjust I2C Clock and Serial baudrate
         to preferred vlaues.
         Set length of interval in us at which data should be pulled and put out through serial.
```

Figure 5: View of VCS with button to upload code to Arduino board highlighted.

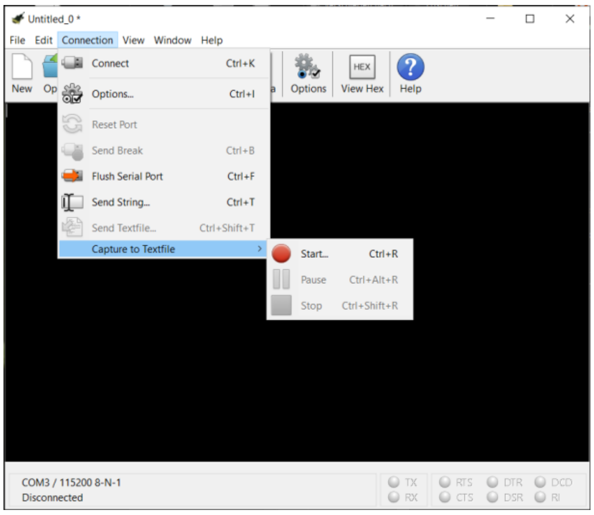
4.4 Receiving Serial Data



```
COM3 / 115200 8-N-1
Disconnected
```

Figure 6: Under "Options", the correct port and baudrate can be set.

The Arduino can now be programmed and will send data through serial to the computer. Any program that can receive and log serial data can be used. For ease of use, we recommend “CoolTerm”. CoolTerm can be downloaded here: https://freeware.the-meiers.org/
After installation, the correct port and baudrate have to be selected.
The data received can be captured to a file by clicking “connection” -> “Capture to Textfile” -> “Start” or by pressing “Ctrl+R”. An explorer Window will open where location a name of the log file can be selected.
When naming the file, the extension can be adjusted, for example to *.csv for easier analyzation.



```
COM3 / 115200 8-N-1
Disconnected
```

Figure 7: This way, the data that is received can be logged to a file.

5 Converting raw sensor values

$\frac{\text{raw}-0.1*\text{adc\_fs}}{0.8*\text{adc\_fs}} * \text{pressure\_fs}$ $\text{adc\_fs} = 2^{\text{adc\_resolution}}$ $\text{adc\_resolution} = 16\text{bit}$

Check our GitHub repository for a concrete example of an implementation of the communication with the sensor and the conversion of raw data.

6 Revision History

Rev. #	Date of change	Author	Changes made
1.0	05.07.2023	Michel Krapf	Document created