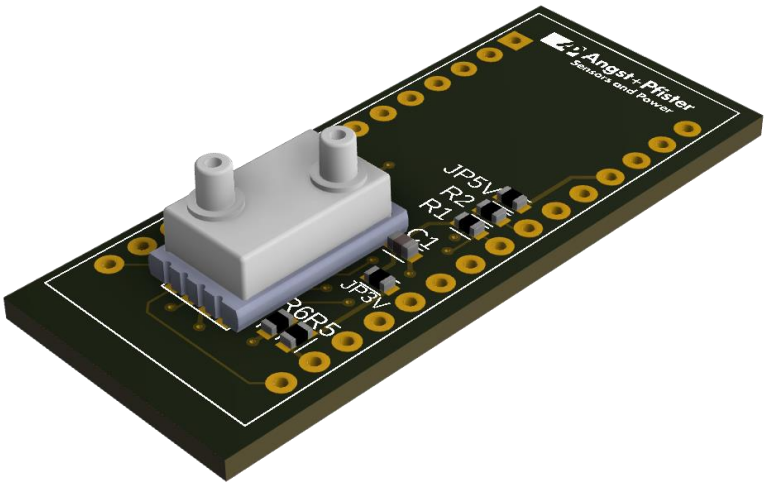
User Manual for the Sensor-Series PHPS-8500



Experts on Design-In
Angst+Pfister
Sensors and Power

Inhalt

1 Disclaimer

Products mentioned in this catalogue may possibly be trademarks used only for the purposes of identification. All rights reserved.
This document may only be used by the recipient for the purpose intended. It may not be copied in whole or in part in any manner, or translated into another language, without our explicit prior consent.
Subject to changes.

Copyright: Angst + Pfister Sensors and Power

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

Keanu Schmidt, Zurich, Release 03.2023

2 Introduction

The kit presented here is designed to help implement I²C Sensors in your design. To simplify this process, this kit provides you with a functioning I²C communication with an output through serial (USB). One version of the code for the I²C 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, analogue voltage output. However, the provided sensors only have two communication protocols, I²C and SPI. In this manual the focus lies on sensors with I²C protocols.

3 Package Content

1x Arduino Nano
1x PCB with sensors
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

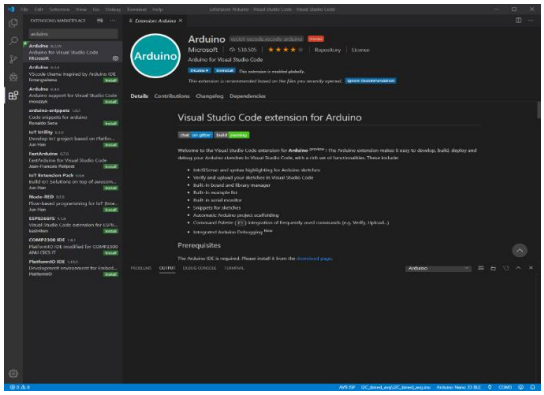


Figure 1: The "Arduino" extension must be installed for VSC to be used to upload code to an Arduino board.

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.

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”.

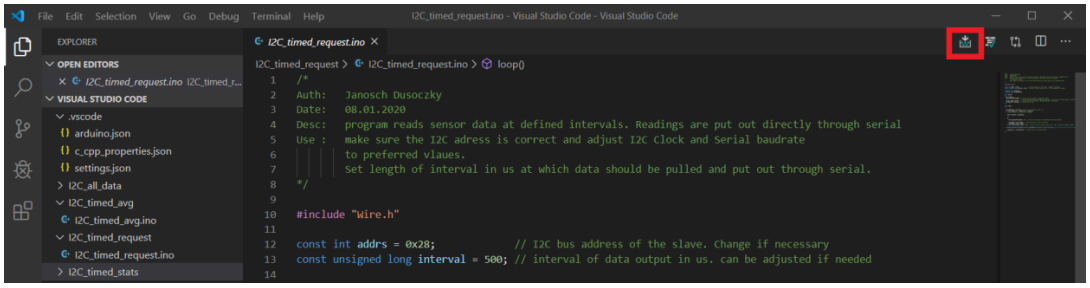


Figure 3: View of VCS with button to upload code to Arduino board highlighted.

4.3.3. Arduino IDE

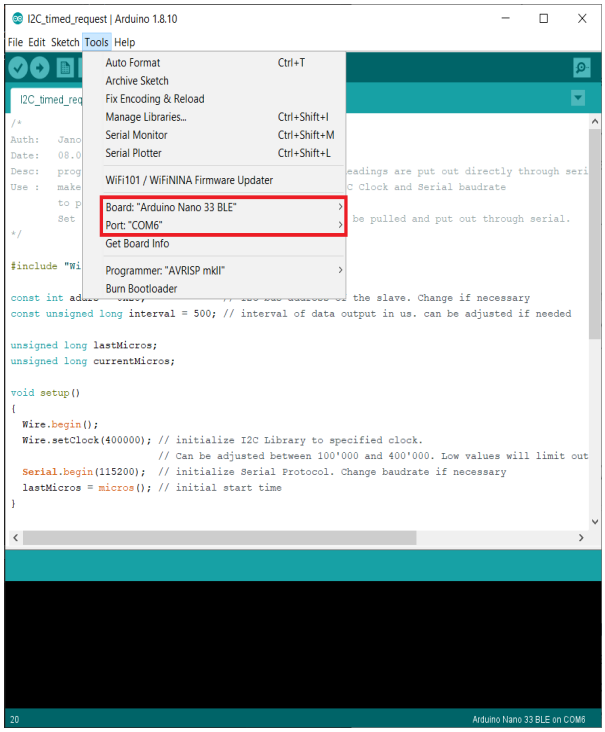


Figure 4: Settings that need to be checked and adjusted.

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.

Experts on Design-In
Angst+Pfister Sensors and Power

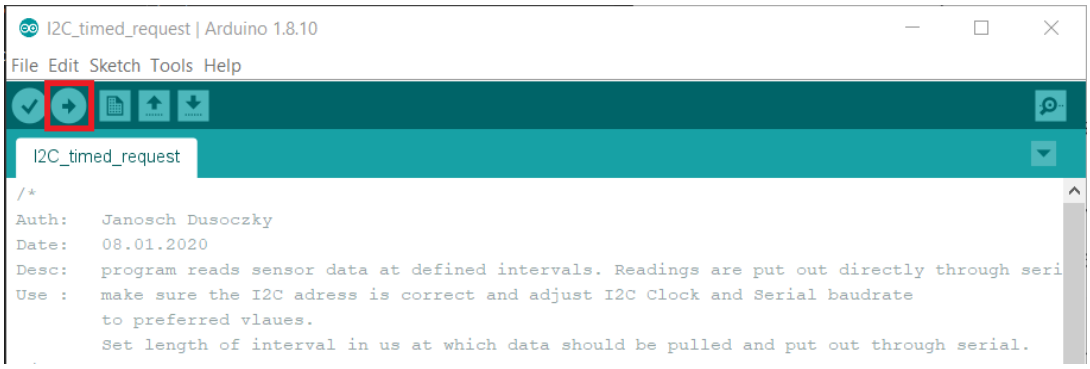


Figure 5: View of VCS with button to upload code to Arduino board highlighted.

4.4 Receiving Serial Data

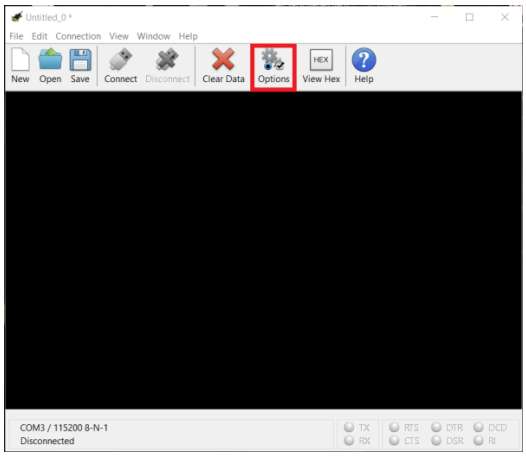


Figure 6: Under "Options", the correct port and baudrate can be set.

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.

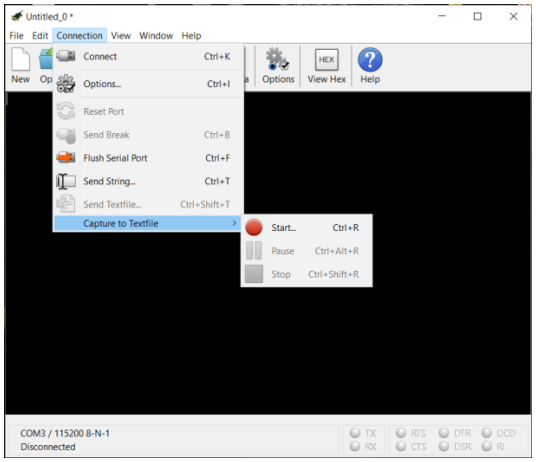


Figure 7: This way, the data that is received can be logged to a file.

Experts on Design-In
Angst+Pfister Sensors and Power

5 Equation for the Code

Definitions:

P= pressure (mbar)
Pmin= min pressure (mbar)
Pmax= max pressure (mbar)
D = digital pressure (counts)
Dmax = max digital pressure (counts)
Dmin = min digital pressure (counts)
S= sensitivity (count/mbar)

Equation:

$$S = \frac{\text{Dmax} - \text{Dmin}}{\text{Pmax} - \text{Pmin}} \qquad P = \frac{\text{D} - \text{Dmin}}{\text{S}} + \text{Pmin}$$

Example:

PHPS-8500 | Arduino 1.8.19

File Edit Sketch Tools Help

PHPS-8500

```
#include <Wire.h>  // Include Arduino Wire library for I2C, URL: https://github.com/PaulStoffregen/Wire

#define RxPin 0  // Define Serial communication pin Rx
#define TxPin 1  // Define Serial communication pin Tx


#define SLAVE_ADDR 120  // Define Slave I2C Address , Slave address: 0x78


// Define pressure range
const float Pmin = -0.5;                          // Max. pressure (mbar)
const float Pmax = 2.5;                           // Min. pressure (mbar)
const uint32_t DPmin = 3277;                      // Max. digital pressure
const uint32_t DPmax = 29491;                     // Min. digital pressure
const float S_P = (DPmax - DPmin) / (Pmax - Pmin);  // Pressure sensitivity


// Define temperature range
//const float Tmin = 0;                             // Max. temperature (°C)
//const float Tmax = 50.0;                          // Min. temperature (°C)
//onst uint32_t DTmin = 8192;                      // Max. digital temperature
//const uint32_t DTmax = 24576;                     // Min. digital temperature
//const float S_T = (DTmax - DTmin) / (Tmax - Tmin);  // Temperature sensitivity


void setup() {
  Wire.begin();        // Initialize I2C communications as Master
  Serial.begin(9600);  // Setup serial monitor
  delay(100);          // Wait 100ms

  pinMode(13, INPUT);
}


void loop() {
  byte rawdata[5];                // Define array for raw data
  Wire.requestFrom(SLAVE_ADDR, 4);  // Request 4 bytes from the sensor (Slave)
  for (uint8_t i = 0; i < 4; i++) {
    if (Wire.available()) {
      rawdata[i] = Wire.read();  // Store 1 byte at a time to data array
    } else {
      rawdata[i] = 0;  // Store value 0 if data no avaliable
    }
  }


  uint16_t x = (uint16_t)rawdata[0] << 8 | (uint16_t)rawdata[1];  // Convert pressure data from 2 bytes to unsigned integer
  uint16_t DPvalue = *(uint16_t*)&x;
  x = (uint16_t)rawdata[2] << 8 | (uint16_t)rawdata[3];  // Convert temperature data from 2 bytes to unsigned integer
  uint16_t DTvalue = *(uint16_t*)&x;
  float P = (DPvalue - DPmin) / S_P + Pmin;  // Calculate pressure value
  //float T = (DTvalue - DTmin) / S_T + Tmin;  // Calculate temperature value
  Serial.println(P);                         // Serial Monitor Pressure
  //Serial.println(T); // Serial Monitor Temperature
  delay(500);  // Wait 0.5s
}
```

Figure 8: Example of a code for the sensor PHPS-8502-2P5M-B-S-R

Experts on Design-In
Angst+Pfister Sensors and Power

6 Revision History

Rev. #	Date of change	Author	Changes made
1.0	11.04.2023	Keanu Schmidt	Document created

Angst+Pfister
Sensors and Power

We are here for you. Addresses and Contacts.

Headquarter Switzerland:

Angst+Pfister Sensors and Power AG
Thurgauerstrasse 66
CH-8050 Zurich
Phone +41 44 877 35 00
sensorsandpower@angst-pfister.com

Office Germany:

Angst+Pfister Sensors and Power Deutschland GmbH
Edisonstraße 16
D-85716 Unterschleißheim
Phone +49 89 374 288 87 00
sensorsandpower.de@angst-pfister.com

Scan here and get an overview of personal contacts!

sensorsandpower.angst-pfister.com