Erriez MH-Z19B CO2 sensor library for Arduino

1.0.0

Generated by Doxygen 1.8.13

Contents

Chapter 1

Erriez MH-Z19B CO2 sensor library for Arduino

This is a MH-Z19B CO2 sensor library for Arduino. It has been built from scratch to support hardware and software serial with a small footprint.

The MH-Z19B is a NDIR (Non-Dispersive Infrared) type gas sensor with built-in temperature compensation to measure CO2 concentration in air.

Library features

- Small code/memory footprint
- Hardware and software serial interface at 9600 baud 8N1
- Read CO2 concentration 400..5000 ppm +/-50ppm+3% minimum 5 seconds interval
- Chip detection
- Smart warming-up detection
- Read firmware version
- Set/get range 2000 or 5000 ppm
- Set/get auto calibration (Automatic Baseline Correction 24h interval)
- Manual 400ppm calibration command
- CRC checks on communication protocol and timeout handling
- Interface for sending undocumented commands

Tested Hardware

The following targets are supported and tested:

- AVR: UNO, MINI, Pro Mini 8/16 MHz, ATMega2560, Leonardo
- ARM: DUE
- ESP8266: Mini D1 & D2, NodeMCU
- ESP32: Lolin D32

Examples

- ErriezMHZ19BGettingStarted
- ErriezMHZ19BSerialPlottter
- ErriezMHZ19B7SegmentDisplay

Documentation

- Online HTML
- Doxygen PDF
- Datasheet PDF

CO2 Concentrations

The table below displays the human impact of CO2:

CO2 ppm	Description
0..399	Incorrect values. Minimum value starts at 400ppm outdoor fresh air.
400..1000	Concentrations typical of occupied indoor spaces with good air exchange.
1000..2000	Complaints of drowsiness and poor air quality. Ventilation is required.
2000..5000	Headaches, sleepiness and stagnant, stale, stuffy air. Poor concentration, loss of attention, increased heart rate and slight nausea may also be present.
>5000	Higher values are extremely dangerous and cannot be measured by this sensor.

Usage

- Operating voltage is between 4.5 and 5VDC, 150mA peak current (average < 60mA).
- UART pins are compatible with processors running at 3.3V without level converters.
- Keep sensor outside direct sunlight.

Calibration

The sensor requires an internal calibration regularly. Without it, the minimum value drifts away which is noticeable after a few weeks of operation. With my experiments, the minimum value was drifted to 800ppm after 3 months continues operation without a calibration.

There are two calibration options:

1. Automatic calibration, performed every 24 hours (default).
2. Manual calibration.

1. Automatic Calibration

Automatic calibration is recommended when the sensor cannot be moved outdoor with fresh air. This calibration method requires a regularly ventilated room at 400ppm, at least once in 1..3 weeks. Additionally, it requires continues power-up without interruptions, otherwise the calibration data will not be updated correctly.

Automatic calibration configuration:

- Set auto calibration on: `setAutoCalibration(true)` (Default from manufacture).
- Set auto calibration off: `setAutoCalibration(false)`.

The status can be read with function `getAutoCalibration()`.

Note: For simplicity, this library uses the terminology `Automatic Calibration` which is identical to the `ABC (Automatic Baseline Correction) logic on/off` mentioned in the datasheet.

2. Manual Calibration (400ppm)

Procedure for manual calibration at 400ppm:

- Turn automatic calibration off.
- Power the sensor up outdoor in fresh air for at least 20 minutes. (Not in a forest or a farm which produces background CO2)
- Call `manual400ppmCalibration()` once. This will send command `0x87 Zero Point Calibration`, but is not a zero calibration as stated in the datasheet. There is no nitrogen needed as this calibration is performed at 400ppm.

Now the sensor is calibrated. Repeat the sequence more often for higher accuracy.

3. Manual Calibration (SPAN)

The datasheet also mentions a command `0x88 Span Point Calibration`. The calibration procedure is not clear and therefore not implemented in this library.

MH-Z19B API

Initialization Software Serial

Use a Software Serial when no hardware serial is available. Sometimes a 3rd party library is required, for example for ESP32 targets by installing `ESPSoftwareSerial`. It must be installed into `.arduino15/packages/esp32/hardware/esp32/<version>/libraries/EspSoftware↩ Serial`, because the library contains a naming conflict with existing `SoftwareSerial.h` built-in libraries.

```
{c++}
#include <ErriezMHZ19B.h>
#include <SoftwareSerial.h>

// Pin defines
#define MHZ19B_TX_PIN        4
#define MHZ19B_RX_PIN        5

// Create software serial object
SoftwareSerial mhzSerial(MHZ19B_TX_PIN, MHZ19B_RX_PIN);

// Create MHZ19B object with software serial
ErriezMHZ19B mhz19b(&mhzSerial);
```

Initialization Hardware Serial

Any hardware serial like `Serial`, `Serial1`, `Serial2` etc can be used when supported by the CPU. Multiple hardware serial ports are only available on targets like ATMEGA2560, Leonardo and SAM DUE boards:

```
{c++}
#include <ErriezMHZ19B.h>

// Create MHZ19B object with hardware serial
ErriezMHZ19B mhz19b(&Serial1);
```

General initialization

The optional items of the initialization sequence can be omitted.

```
{c++}
void setup()
{
    // Initialize serial
    Serial.begin(115200);
    Serial.println(F("\nErriez MH-Z19B CO2 Sensor example"));

    // Initialize software serial at fixed baudrate
    mhzSerial.begin(9600);

    // Optional: Detect MH-Z19B sensor (check wiring / power)
    while ( !mhz19b.detect() ) {
        Serial.println(F("Detecting MH-Z19B sensor..."));
        delay(2000);
    };

    // Sensor requires 3 minutes warming-up after power-on
    while (mhz19b.isWarmingUp()) {
        Serial.println(F("Warming up..."));
        delay(2000);
    };
}
```

Read CO2 loop

Read CO2 with minimum interval asynchronous function `isReady()`. A good practice is to check error returns $<$ `0`.

```
{c++}
void loop()
{
    int16_t result;

    // Minimum interval between CO2 reads
    if (mhz19b.isReady()) {
        // Read CO2 from sensor
        result = mhz19b.readCO2();

        // Print result
        if (result < 0) {
            // Print error code
            switch (result) {
                case MHZ19B_RESULT_ERR_CRC:
```

```
                    Serial.println(F("CRC error"));
                    break;
                case MHZ19B_RESULT_ERR_TIMEOUT:
                    Serial.println(F("RX timeout"));
                    break;
                default:
                    Serial.print(F("Error: "));
                    Serial.println(result);
                    break;
            }
        } else {
            // Print CO2 concentration in ppm
            Serial.print(result);
            Serial.println(F(" ppm"));
        }
    }
}
```

Print internal settings

All tests are performed with sensor version string "0443".

```
{c++}
char firmwareVersion[5];

// Optional: Print firmware version
Serial.print(F("  Firmware: "));
mhz19b.getVersion(firmwareVersion, sizeof(firmwareVersion));
Serial.println(firmwareVersion);

// Optional: Print operating range
Serial.print(F("  Range: "));
Serial.print(mhz19b.getRange());
Serial.println(F("ppm"));

// Optional: Print Automatic Baseline Calibration status
Serial.print(F("  Auto calibrate: "));
Serial.println(mhz19b.getAutoCalibration() ? F("On") : F("Off"));
```

Set automatic calibration

Turn automatic calibration on or off once at startup:

```
{c++}
// Optional: Set automatic calibration on (true) or off (false) once
mhz19b.setAutoCalibration(true);
```

Documented commands

The following commands are documented, used and tested by the library:

Command	Description
0x79	Set auto calibration on/off
0x86	Read CO2 concentration
0x87	Calibration zero point at 400ppm (not 0 ppm)
0x88	Calibrate span point (NOT IMPLEMENTED)
0x99	Set detection range

Not documented commands (tested)

The following commands are **not documented**, are used and tested by the library:

Command	Description
0x7D	Get auto calibration status (NOT DOCUMENTED)
0x9B	Get range detection (NOT DOCUMENTED)
0xA0	Get firmware version (NOT DOCUMENTED)

More information about undocumented commands: https://revspace.nl/MH-Z19B.

NOTE: Sending untested commands may damage the sensor permanently! Use at your own risk.

```
{c++}
uint8_t response[MHZ19B_RESPONSE_LENGTH];
int16_t result;

mhz19b.sendCommand(MHZ19B_CMD_NOT_DOCUMENTED, 0x00, 0x00, 0x00, 0x00, 0x00);
result = mhz19b.receiveResponse(response, sizeof(response));
```

Library configuration

Unfortunately, the sensor has no possibility to read warming-up status, so the library must wait at least 3 minutes after reset or power-on. To speedup the boot process, macro `MHZ19B_SMART_WARMING_UP` can be enabled in `ErriezMHZ19B.h` to enable smart warming-up when the MCU is reset and MH-Z19B powered $>$ 3 minutes.

Library installation

Please refer to the Wiki page.

Other Arduino Libraries and Sketches from Erriez

Erriez Libraries and Sketches

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 ErriezMHZ19B Class Reference

Class ErriezMHZ19B.

`#include <ErriezMHZ19B.h>`

Public Member Functions

- ErriezMHZ19B (Stream *serial)
  Constructor with serial Stream.
- ~ErriezMHZ19B ()
  Destructor.
- bool detect ()
  Detect MHZ19B sensor.
- bool isWarmingUp ()
  Check if sensor is warming-up after power-on.
- bool isReady ()
  Check minimum interval between CO2 reads.
- int16_t readCO2 ()
  Read CO2 from sensor.
- MHZ19B_Result_e getVersion (char *version, uint8_t versionLen)
  Get firmware version (NOT DOCUMENTED)
- MHZ19B_Result_e setRange (MHZ19B_Range_e range)
  Set CO2 range in PPM.
- MHZ19B_Range_e getRange ()
  Get CO2 range in PPM (NOT DOCUMENTED)
- MHZ19B_Result_e setAutoCalibration (bool calibrationOn)
  Enable or disable automatic calibration.
- int8_t getAutoCalibration ()
  Get status automatic calibration (NOT DOCUMENTED)
- MHZ19B_Result_e manual400ppmCalibration ()
  Manual 400ppm calibration (Zero Point Calibration)
- void sendCommand (uint8_t cmd, byte b3=0, byte b4=0, byte b5=0, byte b6=0, byte b7=0)
  Send serial command to sensor.
- MHZ19B_Result_e receiveResponse (uint8_t *rxBuffer, uint8_t rxBufferLength)
  Receive serial response from sensor.

4.1.1 Detailed Description

Class ErriezMHZ19B.

Definition at line 92 of file ErriezMHZ19B.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 ErriezMHZ19B()

```
ErriezMHZ19B::ErriezMHZ19B (
            Stream * serial )
```

Constructor with serial Stream.

Parameters

serial	Serial Stream pointer.

Definition at line 49 of file ErriezMHZ19B.cpp.

4.1.2.2 ~ErriezMHZ19B()

```
ErriezMHZ19B::~ErriezMHZ19B ( )
```

Destructor.

The serial Stream pointer is cleared and requires a new constructor to reuse it again.

Definition at line 58 of file ErriezMHZ19B.cpp.

4.1.3 Member Function Documentation

4.1.3.1 detect()

```
bool ErriezMHZ19B::detect ( )
```

Detect MHZ19B sensor.

Return values

true	Sensor detected.
false	Sensor not detected.

Definition at line 70 of file ErriezMHZ19B.cpp.

4.1.3.2 getAutoCalibration()

```
int8_t ErriezMHZ19B::getAutoCalibration ( )
```

Get status automatic calibration (NOT DOCUMENTED)

Return values

true	Automatic calibration on.
false	Automatic calibration off.

Definition at line 300 of file ErriezMHZ19B.cpp.

4.1.3.3 getRange()

```
MHZ19B_Range_e ErriezMHZ19B::getRange ( )
```

Get CO2 range in PPM (NOT DOCUMENTED)

This function verifies valid read ranges of 2000 or 5000 ppm.
Note: Other ranges may be returned, but are undocumented and marked as invalid.

Return values

MHZ19B_RANGE_INVALID	Invalid range.
MHZ19B_RANGE_2000	Range 2000 ppm.
MHZ19B_RANGE_5000	Range 5000 ppm (default).

Definition at line 255 of file ErriezMHZ19B.cpp.

4.1.3.4 getVersion()

```
MHZ19B_Result_e ErriezMHZ19B::getVersion (
            char * version,
            uint8_t versionLen )
```

Get firmware version (NOT DOCUMENTED)

This is an undocumented command, but most sensors returns ASCII "0430 or "0443".

Parameters

version	Character pointer to version (must be at least 5 Bytes).
versionLen	Number of characters including NULL of version buffer.

Returns

MH-Z19B response error codes.

Definition at line 187 of file ErriezMHZ19B.cpp.

4.1.3.5 isReady()

```
bool ErriezMHZ19B::isReady ( )
```

Check minimum interval between CO2 reads.

Not described in the datasheet, but it is the same frequency as the built-in LED blink.

Returns

Definition at line 130 of file ErriezMHZ19B.cpp.

4.1.3.6 isWarmingUp()

```
bool ErriezMHZ19B::isWarmingUp ( )
```

Check if sensor is warming-up after power-on.

The datasheet mentions a startup delay of 3 minutes before reading CO2. Experimentally discovered, the sensor may return CO2 data earlier. To speed-up the boot process, it is possible to check if the CO2 value changes to abort the warming-up, for example when the MCU is reset and keep the sensor powered. Recommended to disable this option for deployment by disabling macro MHZ19B_SMART_WARMING_UP in header file.

Return values

true	Sensor is warming-up.
false	Sensor is ready to use.

Definition at line 96 of file ErriezMHZ19B.cpp.

4.1.3.7 manual400ppmCalibration()

```
MHZ19B_Result_e ErriezMHZ19B::manual400ppmCalibration ( )
```

Manual 400ppm calibration (Zero Point Calibration)

The sensor must be powered-up for at least 20 minutes in fresh air at 400ppm room temperature. Then call this function once to execute self calibration. Note: This function is useful when auto calibrate is turned off.

Returns

MH-Z19B response error codes.

Definition at line 331 of file ErriezMHZ19B.cpp.

4.1.3.8 readCO2()

```
int16_t ErriezMHZ19B::readCO2 ( )
```

Read CO2 from sensor.

Return values

<0	MH-Z19B response error codes.
0..399	ppm Incorrect values. Minimum value starts at 400ppm outdoor fresh air.
400..1000	ppm Concentrations typical of occupied indoor spaces with good air exchange.
1000..2000	ppm Complaints of drowsiness and poor air quality. Ventilation is required.
2000..5000	ppm Headaches, sleepiness and stagnant, stale, stuffy air. Poor concentration, loss of attention, increased heart rate and slight nausea may also be present. Higher values are extremely dangerous and cannot be measured.

Definition at line 155 of file ErriezMHZ19B.cpp.

4.1.3.9 receiveResponse()

```
MHZ19B_Result_e ErriezMHZ19B::receiveResponse (
            uint8_t * rxBuffer,
            uint8_t rxBufferLength )
```

Receive serial response from sensor.

Parameters

rxBuffer	Receive buffer (must be 9 Bytes).
rxBufferLength	Receive buffer size.

Returns

MH-Z19B response error codes.

Definition at line 378 of file ErriezMHZ19B.cpp.

4.1.3.10 sendCommand()

```
void ErriezMHZ19B::sendCommand (
            uint8_t cmd,
            byte b3 = 0,
            byte b4 = 0,
            byte b5 = 0,
            byte b6 = 0,
            byte b7 = 0 )
```

Send serial command to sensor.

Send command to sensor. Then retrieve response from sensor with receiveResponse().

Parameters

cmd	Command Byte
b3	Byte 3
b4	Byte 4
b5	Byte 5
b6	Byte 6
b7	Byte 7

Definition at line 355 of file ErriezMHZ19B.cpp.

4.1.3.11 setAutoCalibration()

```
MHZ19B_Result_e ErriezMHZ19B::setAutoCalibration (
            bool calibrationOn )
```

Enable or disable automatic calibration.

Parameters

calibrationOn	true: Automatic calibration on. false: Automatic calibration off.

Returns

MH-Z19B response error codes.

Definition at line 284 of file ErriezMHZ19B.cpp.

4.1.3.12 setRange()

```
MHZ19B_Result_e ErriezMHZ19B::setRange (
            MHZ19B_Range_e range )
```

Set CO2 range in PPM.

This function only accepts documented range values.

Parameters

range	Valid ranges in PPM: 2000, 5000 (default).

Returns

MH-Z19B response error codes.

Definition at line 226 of file ErriezMHZ19B.cpp.

The documentation for this class was generated from the following files:

- src/ErriezMHZ19B.h
- src/ErriezMHZ19B.cpp

Chapter 5

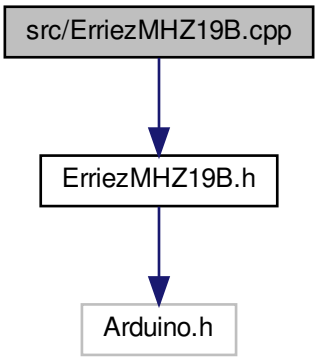
File Documentation

5.1 src/ErriezMHZ19B.cpp File Reference

MH-Z19B CO2 sensor library for Arduino.

```
#include "ErriezMHZ19B.h"
```

Include dependency graph for ErriezMHZ19B.cpp:



5.1.1 Detailed Description

MH-Z19B CO2 sensor library for Arduino.

This sensor library is re-build from scratch.

Design choices:

- Keep code and memory size as small as possible.
- Use documented functions as much as possible for reliability and to prevent bricking the sensor.
- PWM not implemented in this library, because it is not accurate and reduces code size.

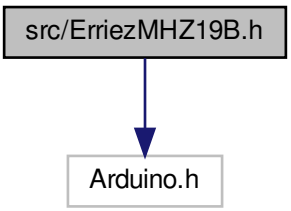
Source: https://github.com/Erriez/ErriezMHZ19B Documentation: https://erriez.↩
github.io/ErriezMHZ19B

5.2 src/ErriezMHZ19B.h File Reference

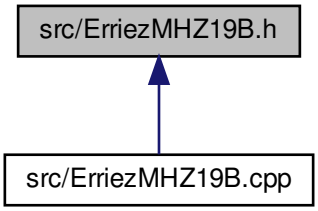
MH-Z19B CO2 sensor library for Arduino.

```
#include <Arduino.h>
```

Include dependency graph for ErriezMHZ19B.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ErriezMHZ19B

  Class ErriezMHZ19B.

Macros

- #define MHZ19B_WARMING_UP_TIME_MS (3UL ∗ 60000UL)

  3 minutes warming-up time after power-on before valid data returned
- #define MHZ19B_READ_INTERVAL_MS (5UL ∗ 1000UL)

  Minimum response time between CO2 reads (EXPERIMENTALLY DEFINED)
- #define MHZ19B_RESPONSE_LENGTH 9

  Fixed 9 Bytes response.
- #define MHZ19B_SERIAL_RX_TIMEOUT_MS 120

Response timeout between 15..120 ms at 9600 baud works reliable for all commands.

- #define MHZ19B_CMD_SET_AUTO_CAL 0x79

  Command set auto calibration on/off.
- #define MHZ19B_CMD_READ_CO2 0x86

  Command read CO2 concentration.
- #define MHZ19B_CMD_CAL_ZERO_POINT 0x87

  Command calibrate zero point at 400ppm.
- #define MHZ19B_CMD_CAL_SPAN_PIONT 0x88

  Command calibrate span point (NOT IMPLEMENTED)
- #define MHZ19B_CMD_SET_RANGE 0x99

  Command set detection range.
- #define MHZ19B_CMD_GET_AUTO_CAL 0x7D

  Command get auto calibration status (NOT DOCUMENTED)
- #define MHZ19B_CMD_GET_RANGE 0x9B

  Command get range detection (NOT DOCUMENTED)
- #define MHZ19B_CMD_GET_VERSION 0xA0

  Command get firmware version (NOT DOCUMENTED)

Enumerations

- enum MHZ19B_Result_e { MHZ19B_RESULT_OK = 0, MHZ19B_RESULT_ERROR = -1, MHZ19B_RESULT_ERR_CRC = -2, MHZ19B_RESULT_ERR_TIMEOUT = -3, MHZ19B_RESULT_ARGUMENT_ERROR = -4 }

  Response on a command.
- enum MHZ19B_Range_e { MHZ19B_RANGE_INVALID = -1, MHZ19B_RANGE_2000 = 2000, MHZ19B_RANGE_5000 = 5000 }

  PPM range.

5.2.1 Detailed Description

MH-Z19B CO2 sensor library for Arduino.

Source: https://github.com/Erriez/ErriezMHZ19B Documentation: https://erriez.github.io/ErriezMHZ19B

5.2.2 Macro Definition Documentation

5.2.2.1 MHZ19B_WARMING_UP_TIME_MS

```
#define MHZ19B_WARMING_UP_TIME_MS (3UL * 60000UL)
```

3 minutes warming-up time after power-on before valid data returned

Enable smart warming-up to return false when CO2 value changes within 3 minutes pre-heating time. Can be used when MCU is reset and sensor powered-up for >3 minutes. Recommended to disable for deployment to ensure warming-up timing.

Definition at line 45 of file ErriezMHZ19B.h.

5.2.3 Enumeration Type Documentation

5.2.3.1 MHZ19B_Range_e

`enum MHZ19B_Range_e`

PPM range.

Enumerator

MHZ19B_RANGE_INVALID	Invalid range.
MHZ19B_RANGE_2000	Range 2000 ppm.
MHZ19B_RANGE_5000	Range 5000 ppm (Default)

Definition at line 82 of file ErriezMHZ19B.h.

5.2.3.2 MHZ19B_Result_e

`enum MHZ19B_Result_e`

Response on a command.

Enumerator

MHZ19B_RESULT_OK	Response OK.
MHZ19B_RESULT_ERROR	Response error.
MHZ19B_RESULT_ERR_CRC	Response CRC error.
MHZ19B_RESULT_ERR_TIMEOUT	Response timeout.
MHZ19B_RESULT_ARGUMENT_ERROR	Response argument error.

Definition at line 71 of file ErriezMHZ19B.h.

Index