RENESAS

Indoor Air Quality 2nd Gen. Library Documentation

Library Version: 3.2.0

Contents

Chapter 1

ZMOD4410 Application Programming Interface Overview

This document describes the libraries for the ZMOD4410 gas sensor module using the second generation algorithms for indoor air quality measurements (IAQ 2nd Gen). This algorithm is the recommended for accurate and consistent TVOC, IAQ and eCO2 measurements. Refer to the *ZMOD4410 Programming Manual - Read Me* for further information regarding sample code. The figure below shows an overview of the ZMOD4xxx API, programming example and libraries. Custom microcontrollers can be used to establish I2C communication. Using the user's own microcontroller requires implementing the user's own target-specific I2C and low-level functions (highlighted in blue). The following sections describe in detail the IAQ 2nd Gen algorithm libraries and the cleaning procedure that is mandatory to be applied after product assembly.
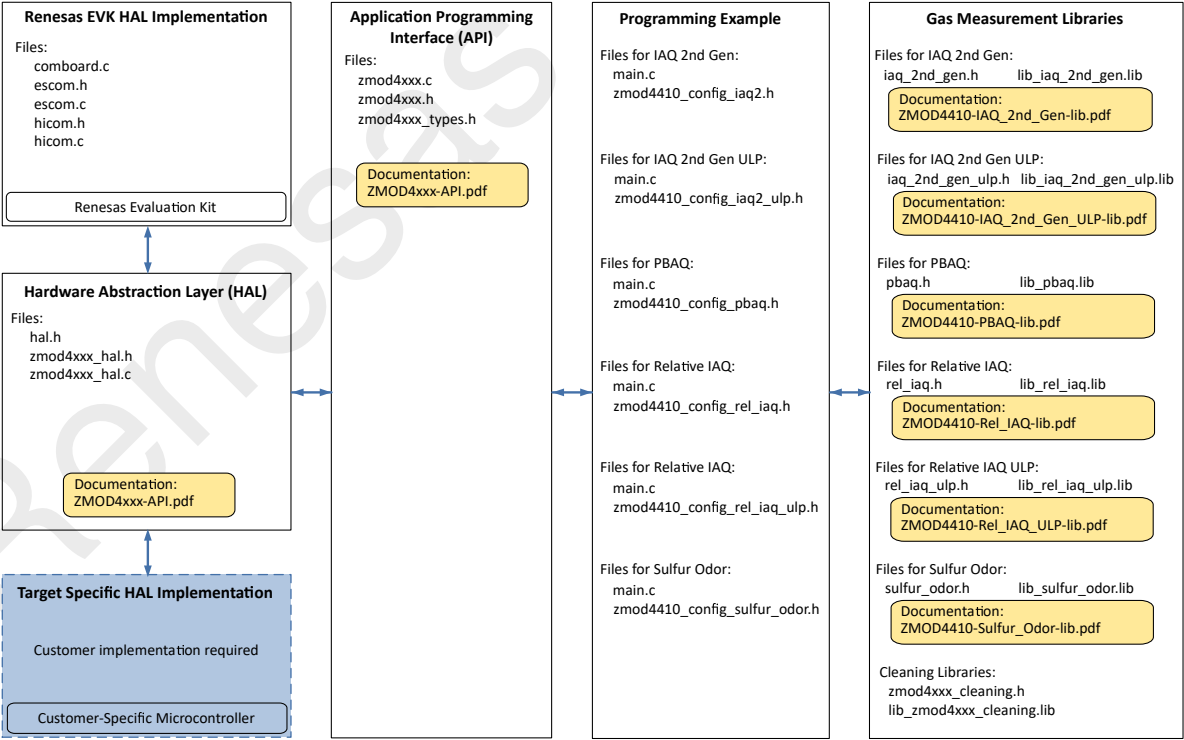
Chapter 2

How to Read Library Version

Libraries have a library version as variable that can be accessed during run-time.

To access the library version:

```c
#include <iaq_2nd_gen.h>

extern algorithm_version iaq_2nd_gen_ver;

int main()
{
    int8_t ret;
    zmod4xxx_dev_t dev;

    /* Sensor target variables */
    uint8_t zmod4xxx_status;
    .
    .

    // Debug library version
    printf("major = %u", iaq_2nd_gen_ver.major);
    printf("minor = %u", iaq_2nd_gen_ver.minor);
    printf("patch = %u", iaq_2nd_gen_ver.patch);
}
```

Chapter 3

How to Work with the Renesas Gas Algorithm Libraries

- Include the intended header file in the user's program for gas sensor module control; for example:
  `#include "iaq_2nd_gen.h"`

- Copy the library file into user's project folder

- Call the intended function in the user's program

Example for IAQ:

```c
#include "iaq_2nd_gen.h"

int main() {
    int8_t ret;
    iaq_2nd_gen_handle_t iaq_handle;
    iaq_2nd_gen_inputs_t algo_input;
    iaq_2nd_gen_results_t results;
    zmod4xxx_dev_t dev;
    uint8_t adc_result[ZMOD4410_ADC_DATA_LEN];
    ...

    // User's functionality
    ...

    // Hardware initialization
    ...

    // Algorithm initialization
    ret = init_iaq_2nd_gen(&iaq_handle);

    // User's functionality
    ...

    while(1) {
        // start sensor measurement
        ...

        // wait until end of sensor measurement
        ...

        // check that measurement sequence completed without errors
        ...

        // get adc_result with API and use it as algorithm input
```

```
        ...
        algo_input.adc_result = adc_result;

        // verify validness of sensor results
        ...

        // measure ambient humidity and temperature
        // Humidity and temperature measurements are needed for ambient compensation.
        // It is highly recommended to have a real humidity and temperature sensor
        // for these values!
        algo_input.humidity_pct = 50.0; // 50% RH
        algo_input.temperature_degc = 20.0; // 20 degC
        ...

        // calculate algorithm outputs
        ret = calc_iaq_2nd_gen(&iaq_handle, &dev, &algo_input, &results);
    }

    return 0;
}
```

Chapter 4

Example for zmod4xxx_cleaning:

- Include the intended header file in the user's program for cleaning;
  `#include "zmod4xxx_cleaning.h"`

- Copy the library file into user's project folder

- Call the `zmod4xxx_cleaning_run` function in the user's program

- `IMPORTANT NOTE:` The cleaning procedure takes 1 minute (blocking) and can be executed once only.

```c
#include "zmod4xxx_cleaning.h"

int main() {
    // initialization of the device structure(dev)
    zmod4xxx_dev_t dev;

    // User's functionality

    zmod4xxx_cleaning_run(&dev);

    // User's functionality

    return 0;
}
```

Chapter 5

Module Index

5.1 Modules

Here is a list of all modules:

Chapter 6

Data Structure Index

6.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 7

File Index

7.1 File List

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

Chapter 8

Module Documentation

8.1 Return codes of the algorithm functions.

Macros

- #define IAQ_2ND_GEN_OK (0)
- #define IAQ_2ND_GEN_STABILIZATION (1)
- #define IAQ_2ND_GEN_DAMAGE (-102)

8.1.1 Detailed Description

8.1.2 Macro Definition Documentation

8.1.2.1 IAQ_2ND_GEN_DAMAGE

```
#define IAQ_2ND_GEN_DAMAGE (-102)
```

sensor damaged

8.1.2.2 IAQ_2ND_GEN_OK

```
#define IAQ_2ND_GEN_OK (0)
```

everything okay

8.1.2.3 IAQ_2ND_GEN_STABILIZATION

```
#define IAQ_2ND_GEN_STABILIZATION (1)
```

sensor in stabilization

Chapter 9

Data Structure Documentation

9.1 algorithm_version Struct Reference

Variables that describe the library version.

```
#include <iaq_2nd_gen.h>
```

Data Fields

- uint8_t **major**
- uint8_t **minor**
- uint8_t **patch**

9.1.1 Detailed Description

Variables that describe the library version.

The documentation for this struct was generated from the following file:

- iaq_2nd_gen.h

9.2 iaq_2nd_gen_handle_t Struct Reference

Variables that describe the sensor or the algorithm state.

```
#include <iaq_2nd_gen.h>
```

Data Fields

- float [log_rcda](#) [9]
- float **rh_cda**
- float **t_cda**
- uint32_t **sample_counter**
- float **tvoc_smooth**
- float **tvoc_deltafilter**
- float **acchw**
- float **accow**
- float **eco2**
- float **etoh**
- float **iaq**
- uint8_t **need_filter_init**

9.2.1   Detailed Description

Variables that describe the sensor or the algorithm state.

9.2.2   Field Documentation

9.2.2.1   log_rcda

```
float log_rcda[9]
```

log10 of CDA resistances.

The documentation for this struct was generated from the following file:

- [iaq_2nd_gen.h](#)

9.3   iaq_2nd_gen_inputs_t Struct Reference

Variables that are needed for algorithm.

```
#include <iaq_2nd_gen.h>
```

Data Fields

- uint8_t ∗ **adc_result**
- float **humidity_pct**
- float **temperature_degc**

9.3.1   Detailed Description

Variables that are needed for algorithm.

Parameters

in	*adc_result*	Value from read_adc_result function
in	*humidity_pct*	relative ambient humidity (%)
in	*temperature_degc*	ambient temperature (degC)

The documentation for this struct was generated from the following file:

- iaq_2nd_gen.h

9.4 iaq_2nd_gen_results_t Struct Reference

Variables that receive the algorithm outputs.

```
#include <iaq_2nd_gen.h>
```

Data Fields

- float rmox [13]
- float log_rcda
- float rhtr
- float temperature
- float iaq
- float tvoc
- float etoh
- float eco2

9.4.1 Detailed Description

Variables that receive the algorithm outputs.

9.4.2 Field Documentation

9.4.2.1 eco2

```
float eco2
```

eCO2 concentration (ppm).

9.4.2.2 etoh

```
float etoh
```

EtOH concentration (ppm).

9.4.2.3 iaq

```
float iaq
```

IAQ index.

9.4.2.4 log_rcda

```
float log_rcda
```

log10 of CDA resistance.

9.4.2.5 rhtr

```
float rhtr
```

heater resistance.

9.4.2.6 rmox

```
float rmox[13]
```

MOx resistance.

9.4.2.7 temperature

```
float temperature
```

ambient temperature (degC).

9.4.2.8 tvoc

```
float tvoc
```

TVOC concentration (mg/m^3).

The documentation for this struct was generated from the following file:

- iaq_2nd_gen.h

Chapter 10

File Documentation

10.1 iaq_2nd_gen.h File Reference

This file contains the data structure definitions and the function definitions for the 2nd generation IAQ algorithm.

```
#include <stdint.h>
#include <math.h>
#include "zmod4xxx_types.h"
```

Data Structures

- struct algorithm_version

  Variables that describe the library version.
- struct iaq_2nd_gen_handle_t

  Variables that describe the sensor or the algorithm state.
- struct iaq_2nd_gen_results_t

  Variables that receive the algorithm outputs.
- struct iaq_2nd_gen_inputs_t

  Variables that are needed for algorithm.

Macros

- #define IAQ_2ND_GEN_OK (0)
- #define IAQ_2ND_GEN_STABILIZATION (1)
- #define IAQ_2ND_GEN_DAMAGE (-102)

Functions

- int8_t calc_iaq_2nd_gen (iaq_2nd_gen_handle_t ∗handle, const zmod4xxx_dev_t ∗dev, const iaq_2nd_gen↩ _inputs_t ∗algo_input, iaq_2nd_gen_results_t ∗results)

  calculates algorithm results from present sample.
- int8_t init_iaq_2nd_gen (iaq_2nd_gen_handle_t ∗handle)

  Initializes the algorithm.

10.1.1 Function Documentation

10.1.1.1 calc_iaq_2nd_gen()

```
int8_t calc_iaq_2nd_gen (
                iaq_2nd_gen_handle_t * handle,
                const zmod4xxx_dev_t * dev,
                const iaq_2nd_gen_inputs_t * algo_input,
                iaq_2nd_gen_results_t * results )
```

calculates algorithm results from present sample.

Parameters

in	*handle*	Pointer to algorithm state variable.
in	*dev*	Pointer to the device.
in	*algo_input*	Structure containing inputs required for algo calculation.
out	*results*	Pointer for storing the algorithm results.

Returns

error code.

10.1.1.2 init_iaq_2nd_gen()

```
int8_t init_iaq_2nd_gen (
                iaq_2nd_gen_handle_t * handle )
```

Initializes the algorithm.

Parameters

out	*handle*	Pointer to algorithm state variable.

Returns

error code.

10.2 zmod4xxx_cleaning.h File Reference

This file contains the cleaning function definition for ZMOD4xxx.

```
#include "zmod4xxx.h"
```

Functions

- int8_t zmod4xxx_cleaning_run (zmod4xxx_dev_t ∗dev)

  Start a cleaning procedure.

10.2.1 Function Documentation

10.2.1.1 zmod4xxx_cleaning_run()

```
int8_t zmod4xxx_cleaning_run (
          zmod4xxx_dev_t * dev )
```

Start a cleaning procedure.

Parameters

in	*dev*	pointer to the device

Returns

Error code

Return values

0	Success
!= 0	Error

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES ("RENESAS") PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use o any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.0 Mar 2020)