RENESAS

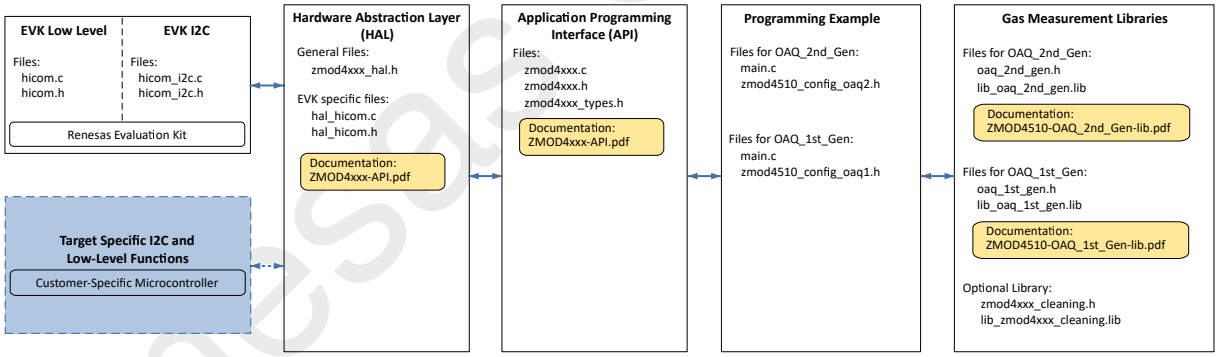
Outdoor Air Quality 2nd Gen. Library Documentation

Contents

Chapter 1

ZMOD4510 Application Programming Interface Overview

This document describes the libraries for the ZMOD4510 gas sensor module using the second-generation algorithms for outdoor air quality measurements (OAQ 2nd Gen) in ultra-low power mode (ULP). This algorithm is recommended for accurate and consistent Air Quality Index (AQI) measurement. Refer to the ZMOD4510 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 OAQ 2nd Gen algorithm libraries for ultra-low power and an optional cleaning procedure after product assembly.



Chapter 2

How to Read Library Version

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

To access the library version;

```
#include <oaq_2nd_gen.h>

extern algorithm_version oaq_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", oaq_2nd_gen_ver.major);
    printf("minor = %u", oaq_2nd_gen_ver.minor);
    printf("patch = %u", oaq_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 "oaq_2nd_gen.h"`
- Copy the library file into user's project folder
- Call the intended function in the user's program

Example for OAQ:

```
#include "oaq_2nd_gen.h"

int main() {
    int8_t ret;
    oaq_2nd_gen_handle_t algo_handle;
    oaq_2nd_gen_inputs_t algo_input;
    oaq_2nd_gen_results_t algo_results;
    zmod4xxx_dev_t dev;
    uint8_t adc_result[ZMOD4510_ADC_DATA_LEN];
    ...

    // User's functionality
    ...

    // Hardware initialization
    ...

    // Algorithm initialization
    ret = init_oaq_2nd_gen(&algo_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 recommented 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 OAQ outputs
        ret = calc_oaq_2nd_gen(&algo_handle, &dev, &algo_input, &algo_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 can be run only once during the modules lifetime and takes 10 minutes.

```
#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 OAQ_2ND_GEN_OK (0)
- #define OAQ_2ND_GEN_STABILIZATION (1)

8.1.1 Detailed Description

8.1.2 Macro Definition Documentation

8.1.2.1 OAQ_2ND_GEN_OK

```
#define OAQ_2ND_GEN_OK (0)
```

everything okay

8.1.2.2 OAQ_2ND_GEN_STABILIZATION

```
#define OAQ_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 <oaq_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:

- oaq_2nd_gen.h

9.2 oaq_2nd_gen_handle_t Struct Reference

Variables that describe the sensor or the algorithm state.

```
#include <oaq_2nd_gen.h>
```

Data Fields

- uint32_t sample_cnt
- float **smooth_rmox**
- float gcda
- float **o3_conc_ppb**
- float **o3_1h_ppb**
- float **o3_8h_ppb**

9.2.1 Detailed Description

Variables that describe the sensor or the algorithm state.

9.2.2 Field Documentation

9.2.2.1 gcda

```
float gcda
```

baseline conductance.

9.2.2.2 sample_cnt

```
uint32_t sample_cnt
```

Sample counter. Will satturate at 0xFFFFFFFF.

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

- oaq_2nd_gen.h

9.3 oaq_2nd_gen_inputs_t Struct Reference

Variables that are needed for algorithm.

```
#include <oaq_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:

- oaq_2nd_gen.h

9.4 oaq_2nd_gen_results_t Struct Reference

Variables that receive the algorithm outputs.

`#include <oaq_2nd_gen.h>`

Data Fields

- float rmox [8]
- float O3_conc_ppb
- uint16_t FAST_AQI
- uint16_t EPA_AQI

9.4.1 Detailed Description

Variables that receive the algorithm outputs.

9.4.2 Field Documentation

9.4.2.1 EPA_AQI

`uint16_t EPA_AQI`

EPA_AQI stands for the Air Quality Index according to the EPA standard based on ozone.

9.4.2.2 FAST_AQI

`uint16_t FAST_AQI`

FAST_AQI stands for a 1-minute average of the Air Quality Index according to the EPA standard based on ozone

9.4.2.3 O3_conc_ppb

```
float O3_conc_ppb
```

O3_conc_ppb stands for the ozone concentration in part-per-billion

9.4.2.4 rmox

```
float rmox[8]
```

MOx resistance.

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

- oaq_2nd_gen.h

Chapter 10

File Documentation

10.1 oaq_2nd_gen.h File Reference

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

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

Data Structures

- struct algorithm_version

  Variables that describe the library version.
- struct oaq_2nd_gen_handle_t

  Variables that describe the sensor or the algorithm state.
- struct oaq_2nd_gen_results_t

  Variables that receive the algorithm outputs.
- struct oaq_2nd_gen_inputs_t

  Variables that are needed for algorithm.

Macros

- #define OAQ_2ND_GEN_OK (0)
- #define OAQ_2ND_GEN_STABILIZATION (1)

Functions

- int8_t init_oaq_2nd_gen (oaq_2nd_gen_handle_t *handle)

  Initializes the OAQ algorithm.
- int8_t calc_oaq_2nd_gen (oaq_2nd_gen_handle_t *handle, zmod4xxx_dev_t *dev, const oaq_2nd_gen_↩ inputs_t *algo_input, oaq_2nd_gen_results_t *results)

  calculates OAQ results from present sample.

10.1.1 Detailed Description

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

Author

Renesas Electronics Corporation

Version

4.0.0

The library contains an algorithm to calculate an ozone concentration and various air quality index values from the ZMOD4510 measurements.

10.1.2 Function Documentation

10.1.2.1 calc_oaq_2nd_gen()

```
int8_t calc_oaq_2nd_gen (
            oaq_2nd_gen_handle_t * handle,
            zmod4xxx_dev_t * dev,
            const oaq_2nd_gen_inputs_t * algo_input,
            oaq_2nd_gen_results_t * results )
```

calculates OAQ 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.

10.1.2.2 init_oaq_2nd_gen()

```
int8_t init_oaq_2nd_gen (
            oaq_2nd_gen_handle_t * handle )
```

Initializes the OAQ 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 Detailed Description

This file contains the cleaning function definition for ZMOD4xxx.

Version

2.5.1

Author

Renesas Electronics Corporation

The library contains the function that starts the cleaning procedure. **The procedure takes 10 minutes.** After successful cleaning, the function returns 0. **The procedure can be run only once.**

10.2.2 Function Documentation

10.2.2.1 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 of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.