Contents

List of Figures

List of Tables

1. Introduction

The ZMOD4450 Gas Sensor Module is highly configurable in order to meet various application needs for refrigeration air quality (RAQ) detection. This document describes a general program flow to set up ZMOD4450 gas sensor modules for RAQ measurements. In addition, this document describes the function of an example code provided as a C file by Renesas for the ZMOD4450, which can be operated using the ZMOD4450 Evaluation Kit (EVK). The ZMOD4450 operates in continuous operation and has the ability to set an external control signal based on the RAQ changes.

Recommendation: Before using this manual, read the *ZMOD4450 Datasheet* available on the Renesas product webpage for the ZMOD4450 at www.IDT.com/ZMOD4450.

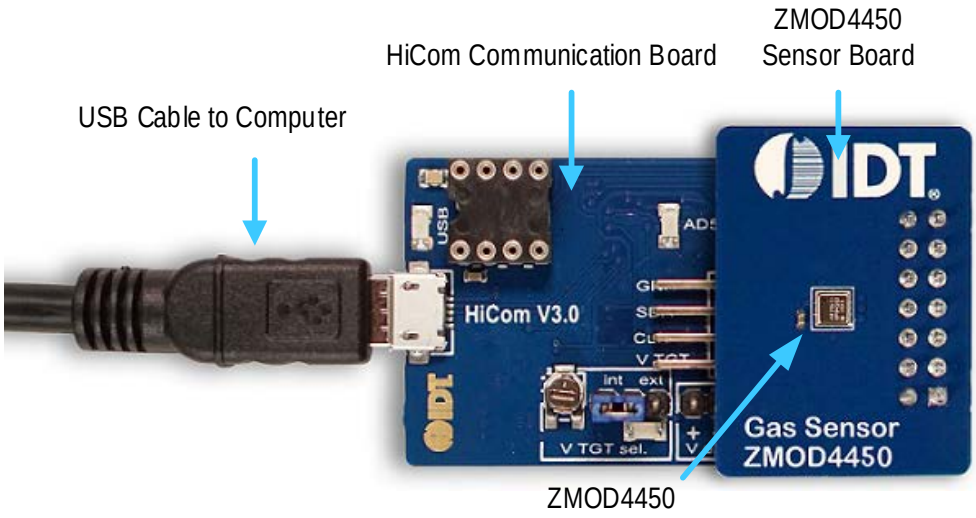
2. Hardware Setup for the ZMOD4450

To implement the ZMOD4450 sensor with the user's program, specific hardware is needed. The example below shows the ZMOD4450 EVK, which consist of three components:

- HiCom Communication Board
- ZMOD4450 Sensor Board (Daughter Board) with the ZMOD4450 Gas Sensor Module
- Micro-USB cable

Refer to the *ZMOD4450 Evaluation Kit Description* (available on www.idt.com/zmod4450-evk) for instructions for assembly, connections, and installation of the hardware and software. Figure 1 shows the assembled evaluation kit.

Figure 1. ZMOD4450 Evaluation Kit



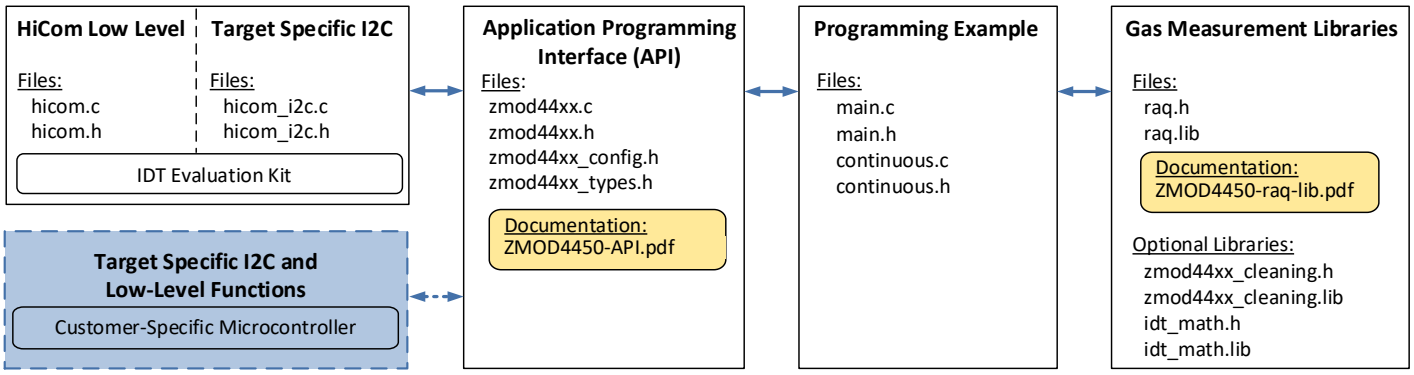
3. General Program Flow for Setting up ZMOD4450 Gas Measurements

To operate the ZMOD4450 in the user's hardware and use its full functionality, four code blocks are needed as illustrated in Figure 2:

- The "Target Specific I2C and Low-Level Functions" block is the hardware-specific implementation of the I2C interface; it contains read and write functions to communicate with the ZMOD4450. If the EVK is used, files for the HiCom Communication Board are provided with this Read-me manual. 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 light blue in Figure 2).
- The "Application Programming Interface (API)" block contains the functions needed to work with the ZMOD4450. A detailed description of the API can be found in the document *ZMOD4450-API.pdf*, which is included with the download of the code file.
- The "Programming Example" block provides a code example that is used to initialize the ZMOD4450 Gas Sensor Module, operate it at a temperature of 300°C, display the data for an air quality change, and a cleaning procedure. Further details can be found in section 4 of this document.
- The "Gas Measurement Libraries" block contains the functions and data structures needed to actually calculate the Air Quality Change Rate-, as well as for a cleaning procedure. The libraries are described in more detail in the document *ZMOD4450-raq-lib.pdf* and are available for download with the code files from the Renesas website.

To avoid naming conflicts, all function names start with the prefix "zmod44xx_" in the ZMOD4450 code. This naming applies to all operation methods of the ZMOD4450.

Figure 2. File Overview for ZMOD4450



4. Description of the Programming Example Code

The following section describes the example code and how to use the ZMOD4450 Gas Sensor Module. In the example, the ZMOD4450 is initialized and configured for operation at 300°C in Continuous Operation, and then it displays measured values. The example is intended to function on a Windows® (trademark of Microsoft, Inc.) computer in combination with the ZMOD4450 EVK. However, the example can be adjusted to operate on other platforms such as ARM® (trademark of ARM, Ltd.) and Linux® (trademark of Linus Torvalds).

To run the example using the EVK without further configuration, run *ZMOD4450_example.exe,* which is included in the kit software.

4.1 *main* Files

The *main.c* / *main.h* files contain the main program flow.

The ZMOD4450 RAQ algorithms are configured by setting the parameters according to the ZMOD4450 libraries. Then the target-specific initializations are performed. The ZMOD4450 is configured by reading the Final Module Test parameters from the sensor's nonvolatile memory (NVM) and initializing it to run at its operation temperature.

An endless loop continuously checks the status of the ZMOD4450 and reads its data. The raw data is subsequently processed, and the ZMOD4450 algorithm for RAQ air quality change is calculated. The value is shown in a command line window. To stop the loop, press any key, which releases the hardware and stops the program.

4.2 Program Flow to Operate the Sensor

Refer to the example code for more details.

Table 1. Program Flow

Note: In the following table, lines that are shaded blue can be run in an endless loop with polling or interrupt usage.

Line	Program Actions	Notes	API Functions
1	Reset the sensor.	Before configuring the sensor, reset the sensor by powering it off/on or toggling the reset pin.	-
2	Read device parameters from the nonvolatile memory (NVM).	This step is required to select the correct configuration for the sensor.	**zmod44xx_read_sensor_info**
3	Initial initialization.	This function must be called after every startup.	**zmod44xx_init_sensor**
4	Initialize the sensor for ZMOD4450 measurements.	Initialize the sensor to run at 300°C.	**zmod44xx_init_measurement**
5	Start the measurement.	Start the measurement.	**zmod44xx_start_measurement**
6	Read status register.	Wait until the initialization is ready.	**zmod44xx_read_status**
7	Read status register.	Wait until the measurement is done. This will also be signaled by an interrupt.	**zmod44xx_read_status**
8	Get the MOx resistance value.		**zmod44xx_read_rmox**
9	Calculation algorithm.	Get ZMOD4450 control signal	
10	Read status register.	Wait until the sensor is ready.	**zmod44xx_read_status**

4.3 Cleaning Procedure

The cleaning procedure is recommended after product assembly. It takes 10 minutes and helps to clean the metal oxide surface from assembly residues.

Important note: The cleaning procedure can be run only once during the modules lifetime.

5. Using the Example on a Different Hardware Platform

To incorporate this programming example into a different hardware platform, it is necessary to set the device's struct pointers *read*, *write,* and *delay_ms*. The type definitions of the function pointers can be found in *zmod44xx_types.h* (see Figure 2). The functions *read* and *write* should point to the I2C implementation of the hardware used.

In addition, Renesas provides precompiled algorithm libraries that are hardware-platform dependent. Renesas offers a download of these libraries for x86/x64 (Linux/Windows) as well as for Arm® Cortex®-M Series, MSP430 Series, 8051, and RL78 (GCC) microcontrollers.

5.1 Adaptation for the Target System

Renesas' ZMOD4450 API, which is written in the C programming language, is located between the application and the target hardware.

Figure 3. System Hierarchy

<table>
<tr><td colspan="2">Application</td></tr>
<tr><td colspan="2">Application-Specific ZMOD4450 Configuration</td></tr>
<tr><td colspan="2">ZMOD4450 API and Libraries (Algorithm)</td></tr>
<tr><td>Low-Level I2C Communication</td><td>Low-Level Hardware Functions</td></tr>
<tr><td colspan="2">Target Hardware</td></tr>
</table>

The low-level I2C functions are implemented in the file *hicom_i2c.c* (see Figure 2) for the EVK hardware running on a Windows-based computer and the HiCom Communication Board.

5.2 Error Codes

Most of the API functions return a code to indicate the success of the operation. If no error occurred, the return code is 0. In the event of an error, a number not equal to zero is returned. The API has predefined symbols ZMOD44XX_ERROR_* for the error codes defined in *zmod44xx_types.h*.

6. Revision History

Revision Date	Description of Change
March 4, 2020	▪ Update library names. ▪ Update to Renesas.
November 18, 2019	▪ Cleaning procedure added and explained. ▪ File overview figure updated.
January 3, 2019	Initial release.