RRH46410

Digital Gas Sensor Module for Indoor Air Quality Applications

1. Introduction

The RRH46410 Gas Sensor Module is an easy to integrate sensor for measuring indoor pollutants with high accuracy. This document describes the general program flow to set up RRH46410 Gas Sensor Modules for gas measurements in a customer's environment. It also describes the functionality of example code provided as C code, which can be executed using the RRH46410 evaluation kit (EVK), Arduino® and Raspberry Pi® hardware.

The corresponding firmware package is provided on the Renesas RRH46410 product page under the Software Downloads section.

For instructions on assembly, connection, and installation of the EVK hardware and software, see the document titled *RRH46410 Evaluation Kit User Manual* on the RRH46410 EVK product page.

The RRH46410 has three modes of operation:

- IAQ 2nd Gen – This mode with embedded artificial intelligence (AI) algorithm ("iaq_2nd_gen") derived from machine learning, outputs total volatile organic compounds (TVOC), equivalent ethanol (EtOH) concentration, estimated carbon dioxide level (eCO2), a rating for the indoor air quality (IAQ), and a relative IAQ index (Rel IAQ) for threshold-based controls like air ventilation. **This is the recommended operation mode for IAQ**.
- IAQ 2nd Gen Ultra Low Power – This mode with embedded artificial intelligence (AI) algorithm ("iaq_2nd_gen_ulp") derived from machine learning, outputs total volatile organic compounds (TVOC), equivalent ethanol (EtOH) concentration, estimated carbon dioxide level (eCO2), a rating for the indoor air quality (IAQ), and a relative IAQ index for threshold-based controls like air ventilation (Rel IAQ). This mode of operation offers a much lower power consumption while keeping accurate and consistent sensor readings.
- Public Building Air Quality (PBAQ) – This mode with embedded artificial intelligence (AI) algorithm derived from machine learning, outputs total volatile organic compounds (TVOC) and equivalent ethanol (EtOH) concentration. This mode of operation is for highly accurate and consistent sensor readings to fulfill public building standards.

All operation modes can be adjusted in the firmware example.

Recommendation: Before using this document, read the *RRH46410 Datasheet* and corresponding documentation on the RRH46410 product page.

Contents

Figures

Tables

2. Structure of the RRH46410 Firmware

The RRH46410 firmware contains four code blocks as displayed in Figure 1:

1. The “Target Specific I²C and Low-Level Functions” block is the hardware-specific implementation of the I²C interface. This block contains read and write functions to communicate with the RRH46410 and a delay function. If the Renesas EVK is used, files for the EVK ESCom Communication Board are provided with the RRH46410 firmware package. Using the user’s own target hardware requires implementing the user’s target-specific I²C and low-level functions (this is highlighted in light blue in Figure 1).
2. The “Hardware Abstraction Layer (HAL)” block contains hardware-specific initialization and de-initialization functions and hardware error handling. Files for the EVK ESCom Communication Board are provided with the RRH46410 firmware package, and need to be adjusted to the user’s target hardware. The HAL is described in the documents *RRH46410-Firmware-Documentation.pdf* and *.html*, which are included in the firmware package.
3. The “Application Programming Interface (API)” block contains the functions needed to operate the RRH46410. Renesas recommends using this API. A detailed description of the API is located in the documents *RRH46410-Firmware-Documentation.pdf* and *.html*, which are included in the firmware package. As the usage of a humidity sensor is recommended, an API for Renesas HSxxxx sensors is also provided.
4. The “Example” block provides a code example as example.c file that is used to initialize the RRH46410, start the cleaning mode, select the operation mode, perform measurements, and display the data output. More information is provided in “Description of the Firmware Example”.

All these files are part of the downloadable firmware package.

To avoid naming conflicts, all API function names start with the prefix “RRH46410” in the RRH46410 code. The Arduino library and Raspberry Pi example have a similar structure but have some other features that facilitate operation with the corresponding hardware (see “RRH46410 Example for Arduino” and “RRH46410 example for Raspberry Pi”).

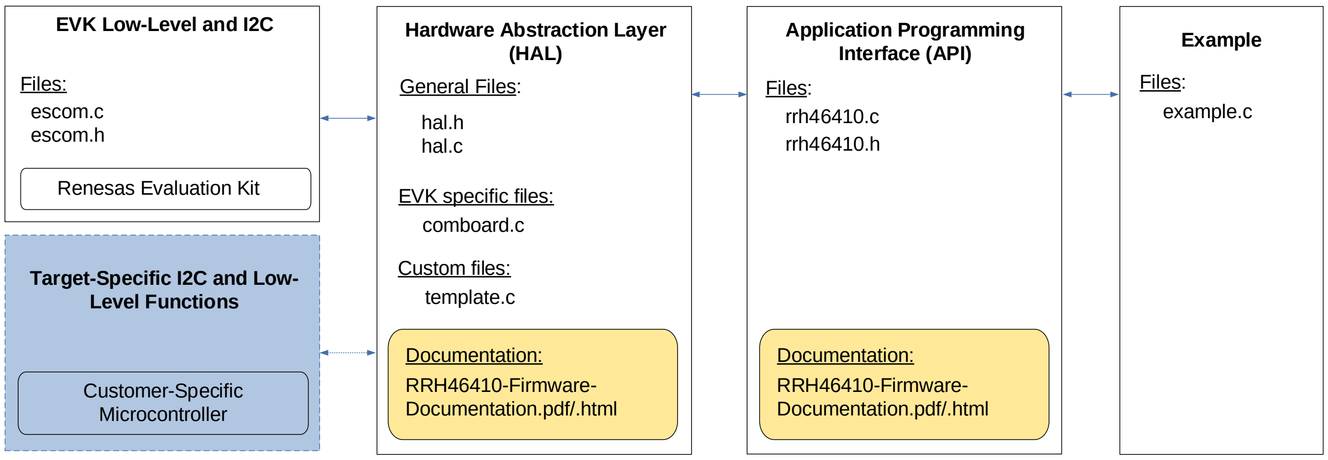

EVK Low-Level and I2C
Files:
escom.c
escom.h
Renesas Evaluation Kit
Target-Specific I2C and Low-Level Functions
Customer-Specific Microcontroller
Hardware Abstraction Layer (HAL)
General Files:
hal.h
hal.c
EVK specific files:
comboard.c
Custom files:
template.c
Documentation:
RRH46410-Firmware-Documentation.pdf/.html
Application Programming Interface (API)
Files:
rrh46410.c
rrh46410.h
Documentation:
RRH46410-Firmware-Documentation.pdf/.html
Example
Files:
example.c

Figure 1. File Overview for RRH46410 Firmware

3. Description of the Firmware Example

This section describes the structure of the firmware example and the steps needed to operate the sensor module. The example is intended to work on a Windows® computer in combination with the Renesas Gas Sensor EVK but can easily be adjusted to operate on other platforms (see “Adapting the Firmware Example for Target Hardware”).

To run this example using the EVK without further configuration, start the file *rrh46410-example.exe,* which is included in the firmware package. In addition, examples for Arduino and Raspberry Pi hardware are provided.

The RRH46410 can be operated in different modes. The precompiled example uses IAQ 2nd Gen mode. For information on how to change the operating mode, see “RRH46410 Example for EVK”.

3.1 RRH46410 Example for EVK

The *example.c* file contains the main program flow. First, the target-specific initializations are performed. The RRH46410 and humidity sensor is initialized and the RRH46410 operation mode is set. An endless measurement loop continuously checks the status of the RRH46410 and reads its data. All values are printed in the command line window. To stop the loop, press Ctrl + C, which releases the hardware and stops the program. For more information, refer to the example code.

Note: The blue colored lines in the following table can be run in an endless loop.

Table 1. RRH46410 Example Program Flow

Line	Program Actions	Notes	API and Algorithm Functions
1	Reset the sensor.	Before configuring the sensor, reset the sensor by powering it off/on or toggling the reset pin.	-
2	Check if all functions are available and check if sensor is accessible.	Function tries to read sensors operation mode for maximum 1 second.	RRH46410_Init
3	Read the product ID, firmware version and tracking number.	This step is optional.	RRH46410_GetSensorInfo
4	Run the cleaning cycle once.	Cleaning runs during first sensor operation. It takes 1 minute.	RRH46410_PerformCleaning
5	Select the operation mode and start measurement.	This function must be called after every RRH46410 startup if the startup operation mode was not adjusted permanently.	RRH46410_SetOperatingMode
6	Set the humidity default value.		RRH46410_SetHumidity
7	Check for humidity data and pass them.		RRH46410_SetHumidity
8	Read the results after the end of the measurement.	The end of the measurement will also be signaled on the interrupt pin with a falling signal. First samples are used for minimal, hard-coded sensor warm-up. Actual warm-up can take longer (up to 48 hours).	RRH46410_ReadResults

To change the operating mode permanently, open example.c and change the line

```
# define OP_MODE  omIAQ2
```

to the desired operating mode defined in *rrh46410.h*. Alternatively, the operating mode can be changed during compilation, by calling make:

```
make –DRRH46410_OP_MODE=<OP_MODE>
```

3.2 How to Compile for EVK Hardware

The EVK firmware example is designed to work with the EVK hardware. To evaluate the impact of code changes on sensor performance, it is possible to use the EVK as reference. This section provides guidelines for compiling the adapted source code into an executable file. This executable can be used with the EVK on a Windows platform. For compiling, “skeeto/w64devkit” must be downloaded and unzipped. The firmware folder structure should be identical to that in the download package.

Install skeeto/w64devkit:

1. Download the latest version of w64devkit and unzip it.
2. Add it to the system path in command line by using:
   `PATH=%PATH%;<PATH_TO_W64DEVKIT>\bin`

Compiling:

1. Go to the Command Prompt and change to the following directory of the firmware folder:
   [...]\Renesas-RRH46410-Firmware\src
2. Execute the following command:
   `make`

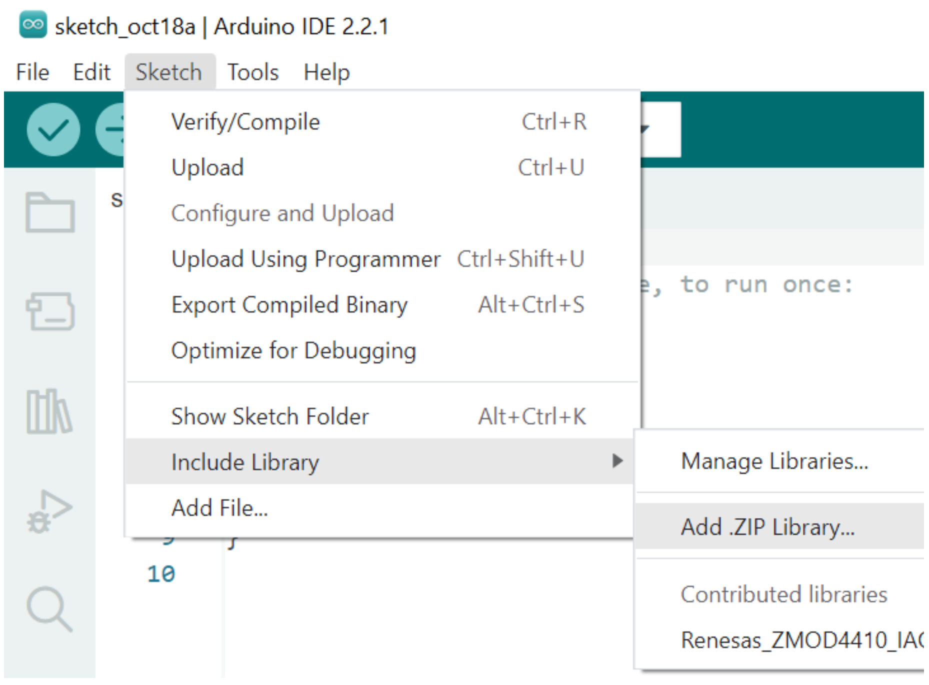
An executable file called *rrh46410-example.exe* will be created in folder *build*.

3.3 RRH46410 Example for Arduino

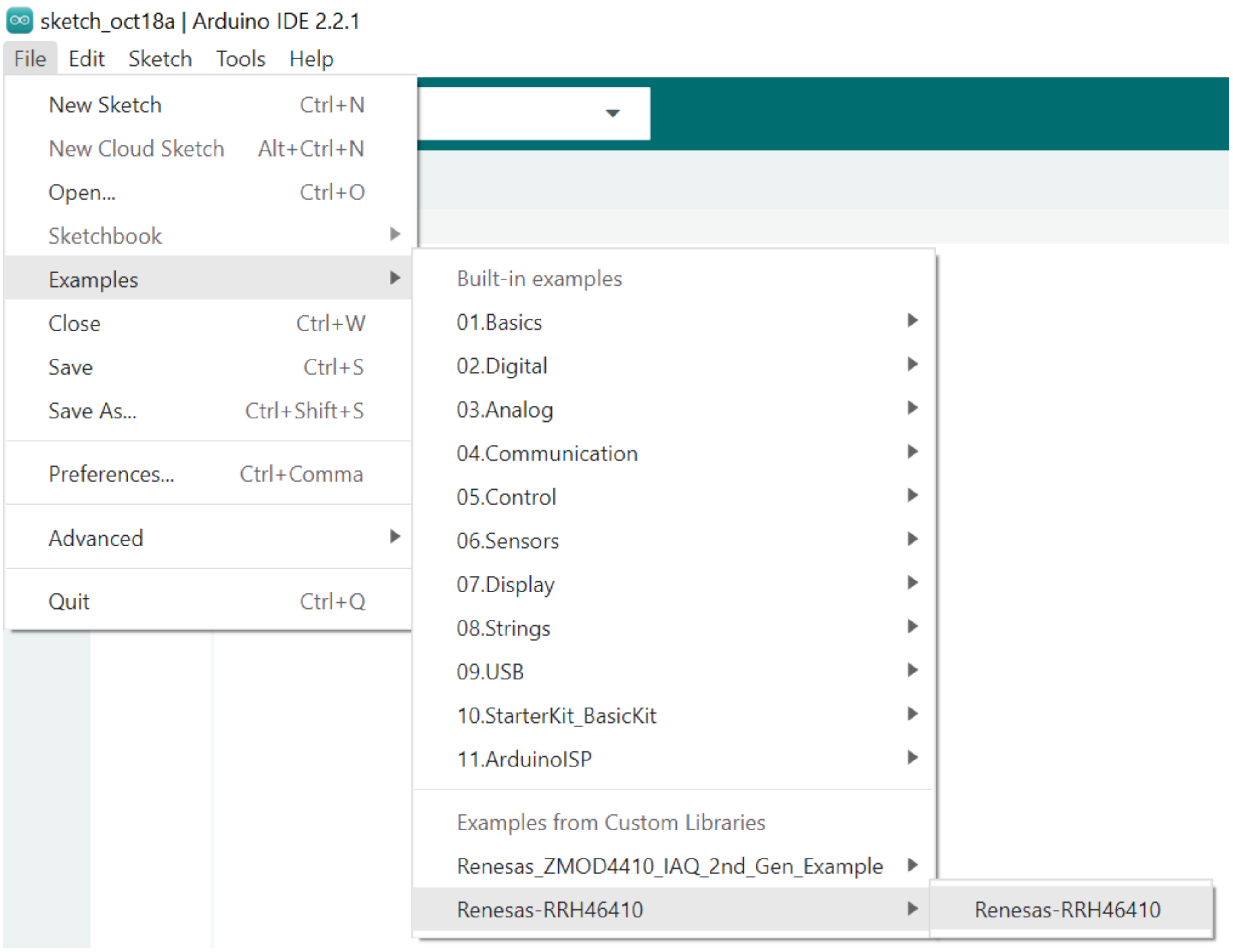
To set up a firmware for an Arduino target, Renesas provides the above-mentioned EVK example also as an Arduino example. This example is a high-level Arduino library and has a similar structure as shown in Figure 1 but with a HAL dedicated for Arduino, an Arduino-compatible structure, and Arduino-specific files. An Arduino IDE with version 1.8.13 and higher is needed.

The program flow corresponds to the EVK example. To get the Arduino example started, complete the following steps (example shown for SAMD 32-bit ARM Cortex-M0+ based Arduino-Hardware):

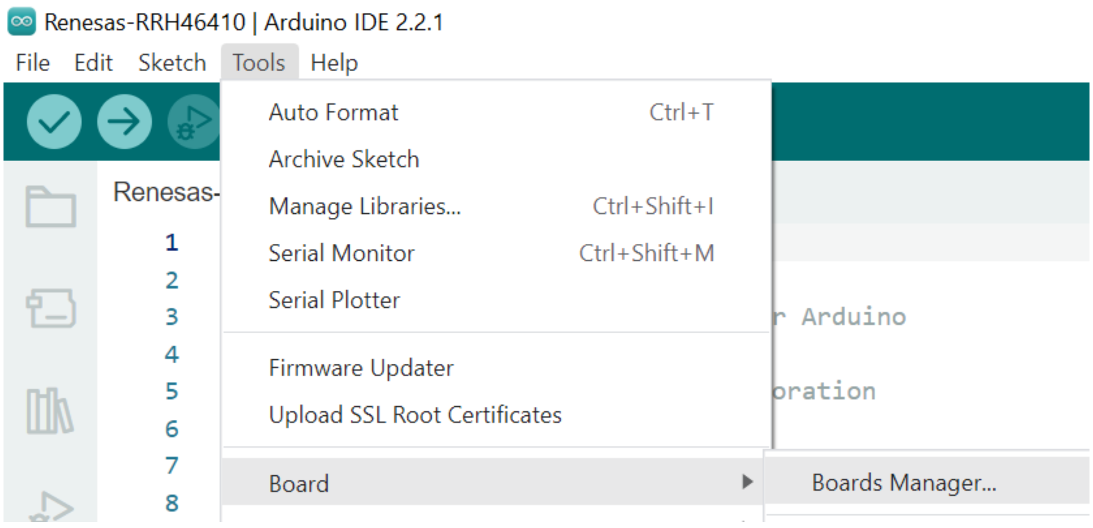
1. Connect the RRH46410 to the Arduino board. To connect the EVK Sensor Board, check the pin configuration on connector “X1” in the *RRH46410 EVK User Manual* on the EVK product page.
2. Go to the Arduino example path (e.g., […]\Documents\Arduino\libraries) and check if a RRH46410 example exists. Old example folders must be deleted.
3. Open Arduino IDE. Select “Sketch > Include Library > Add .ZIP library”.



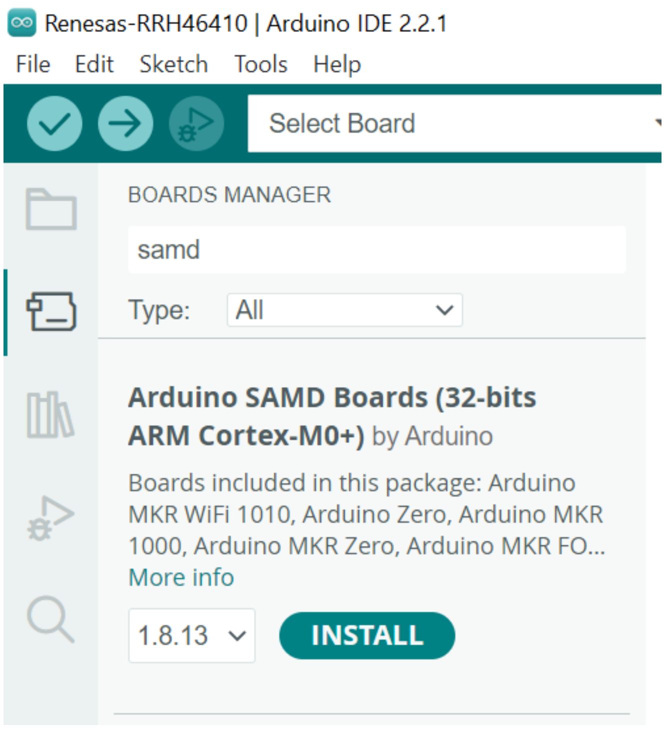
4. Select the Renesas-RRH46410-Arduino.zip file.
5. Select "File > Examples > Renesas-RRH46410." A new Arduino IDE window opens automatically with the example file.



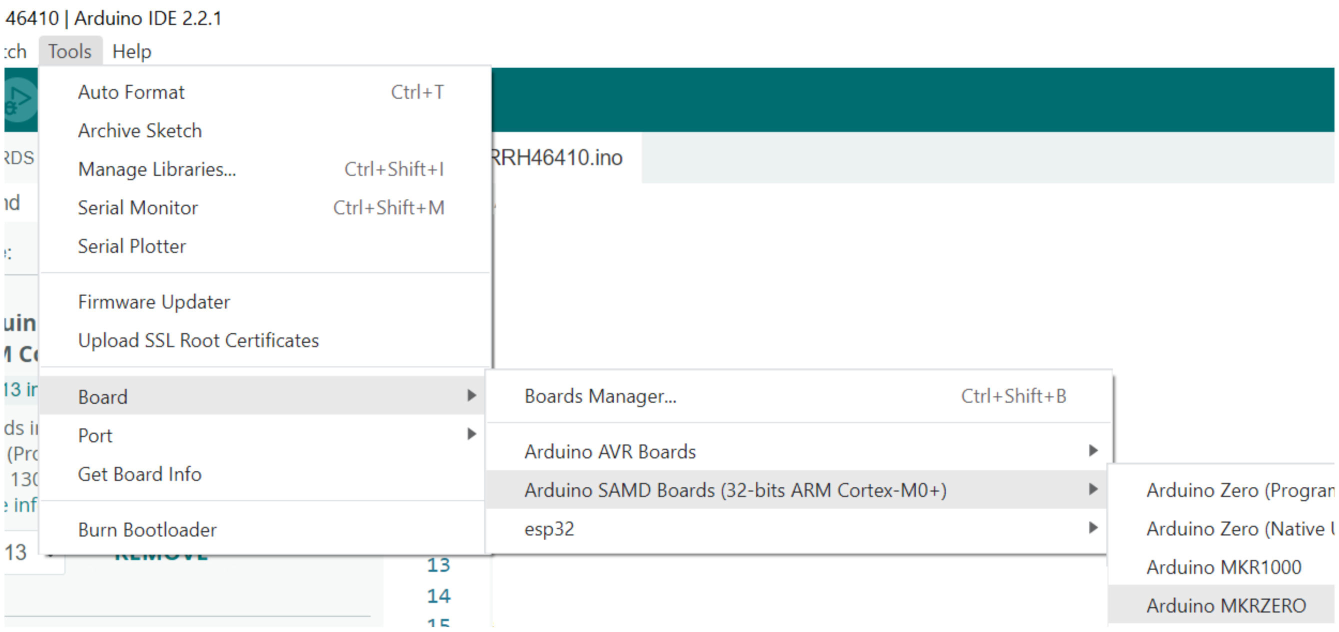
6. (This step may not be required for other Arduino hardware.) Install the "Arduino SAMD (32-bit ARM Cortex-M0+)" Boards library under "Tools > Board > Board Manager".


Renesas-RRH46410 | Arduino IDE 2.2.1
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
Firmware Updater
Upload SSL Root Certificates
Board
Boards Manager...
Renesas-
r Arduino
oration

If it already exists, skip this step. Type "Arduino SAMD Boards" in the search field and click "Install" button in "Arduino SAMD (32-bit ARM Cortex-M0+)" field.

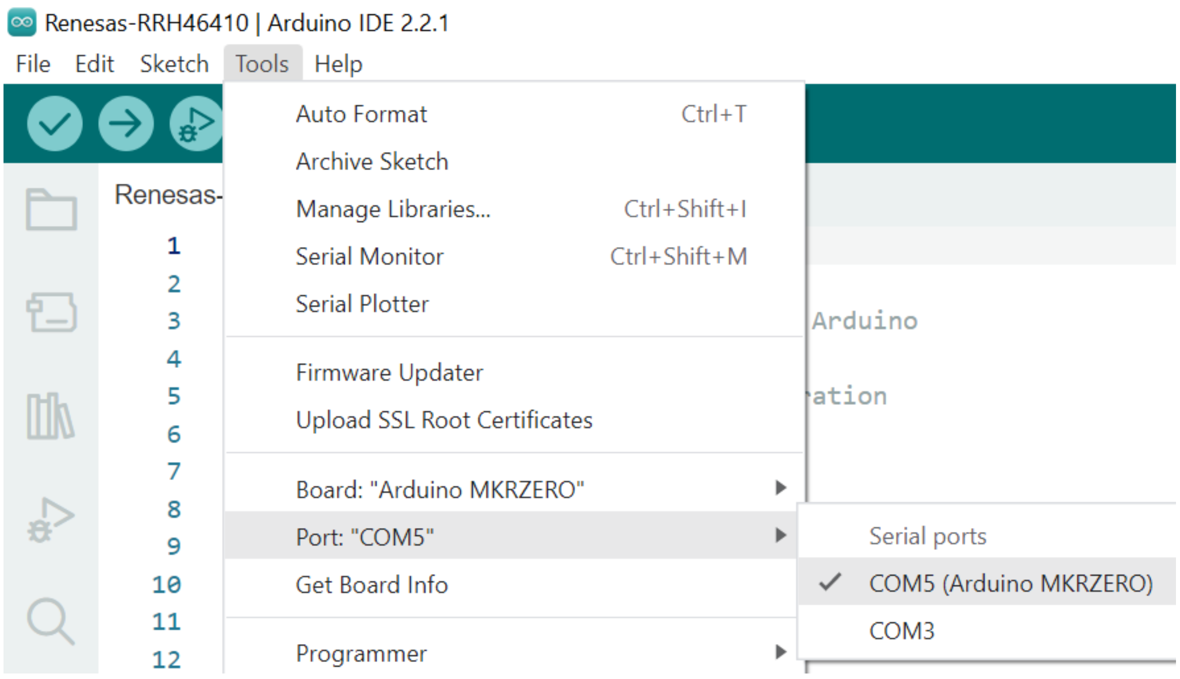

Renesas-RRH46410 | Arduino IDE 2.2.1
File Edit Sketch Tools Help
Select Board
BOARDS MANAGER
samd
Type: All
Arduino SAMD Boards (32-bits ARM Cortex-M0+) by Arduino
Boards included in this package: Arduino MKR WiFi 1010, Arduino Zero, Arduino MKR 1000, Arduino MKR Zero, Arduino MKR FO...
More info
1.8.13
INSTALL

7. Select the target board under "Tools > Board > Arduino SAMD (32-bits ARM Cortex-M0+) > Arduino MKRZERO".

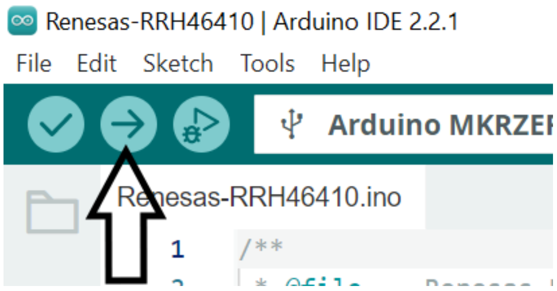


8. Compile the example with the "Verify" icon.

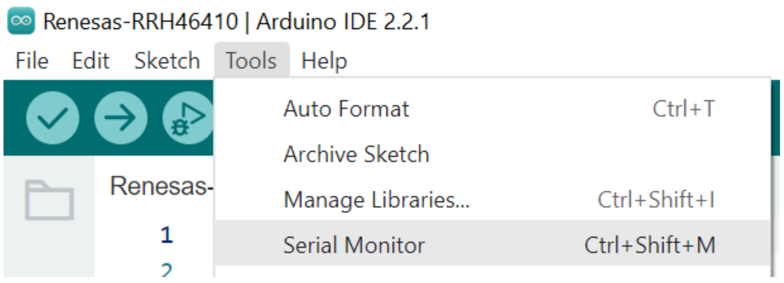
9. Select the connected port with “Tools > Port > (Connected Port)”. The correct COM-Port should show your Arduinos board name.



10. Load the program into the target hardware with the “Upload” icon.


Renesas-RRH46410 | Arduino IDE 2.2.1
File Edit Sketch Tools Help
Arduino MKRZEI
Renesas-RRH46410.ino

11. Check the results with the Serial Monitor (Tools > Serial Monitor).


Renesas-RRH46410 | Arduino IDE 2.2.1
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M

3.4 RRH46410 Example for Raspberry Pi

To set up firmware for a Raspberry Pi based target, Renesas provides the above-mentioned EVK example also as a Raspberry Pi example. This example has a similar structure as shown in Figure 1 but with a HAL dedicated for Raspberry Pi and a Makefile to easily compile the code. The example is based on the pigpio library and Raspberry Pi OS (previously called Raspbian).

The example is tested on the following Raspberry Pi models on Raspberry Pi OS (32-bit):

- Raspberry Pi 3B, B+
- Raspberry Pi 4B

The following table describes the connection of the Sensor Board connector “X1” and the Raspberry Pi GPIO Connector. Documentation of “X1” can be found in the *RRH46410 EVK User Manual*. Documentation of Raspberry Pi GPIO can be found on command line typing “pinout” or online.

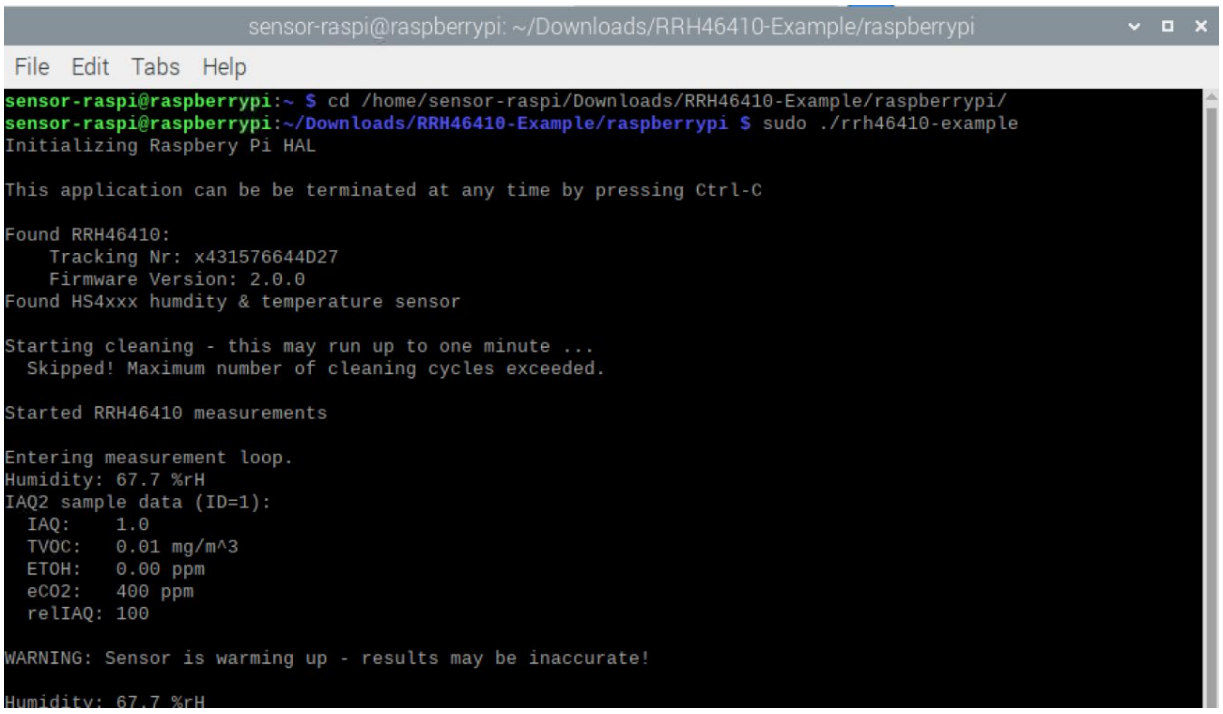
Table 2. Connection of Sensor Board to Raspberry Pi

Sensor Board Pin (X1)	Sensor Board Description	Raspberry Pi Pin	Raspberry Pi Description
1	VDD	1, 17	3V3 power
5	SDA	3	GPIO 2 (SDA)
7	SCL	5	GPIO 3 (SCL)
14	GND	6, 9, 14, 20, 25, 30, 34, 39	Ground

The program flows correspond to the EVK example. To get the Raspberry Pi example started, complete the following steps.

1. Install the Raspberry Pi operating system on the Raspberry Pi. An imager tool is available to easily flash the operating system to SD card.
2. Establish an internet connection via Wi-Fi or LAN and install updates on the Raspberry Pi using the command `sudo apt update && sudo apt upgrade -y` on the Terminal.
   Note: The updates may take some time to finish.
3. To configure the I^{2}C interface go to */boot* directory and open the configuration file:
   ```
   sudo nano config.txt
   ```
4. Enable the I^{2}C interface and change the baud rate by uncommenting the *line #dtparam=i2c_arm=on* and changing it to:

   dtparam=i2c_arm=on,i2c_arm_baudrate=200000
5. Reboot the Raspberry Pi to complete the initial setup. When done, the example code can be started.
6. Copy the whole Renesas firmware package to your Raspberry Pi and extract it to your preferred location (e.g., Downloads).
7. Open the Terminal and go to the directory containing the executable, e.g.:
   ```
   cd ~/Downloads/Renesas-RRH46410-Firmware/raspberrypi/
   ```
8. Start the example with the following command (sudo is required for pigpio package):
   ```
   sudo ./rrh46410-example
   ```



sensor-raspi@raspberrypi: ~/Downloads/RRH46410-Example/raspberrypi
File Edit Tabs Help

```
sensor-raspi@raspberrypi:~ $ cd /home/sensor-raspi/Downloads/RRH46410-Example/raspberrypi/
sensor-raspi@raspberrypi:~/Downloads/RRH46410-Example/raspberrypi $ sudo ./rrh46410-example
Initializing Raspbery Pi HAL

This application can be be terminated at any time by pressing Ctrl-C

Found RRH46410:
    Tracking Nr: x431576644D27
    Firmware Version: 2.0.0
Found HS4xxx humdity & temperature sensor

Starting cleaning - this may run up to one minute ...
  Skipped! Maximum number of cleaning cycles exceeded.

Started RRH46410 measurements

Entering measurement loop.
Humidity: 67.7 %rH
IAQ2 sample data (ID=1):
  IAQ:     1.0
  TVOC:    0.01 mg/m^3
  ETOH:    0.00 ppm
  eCO2:    400 ppm
  relIAQ:  100

WARNING: Sensor is warming up - results may be inaccurate!

Humidity: 67.7 %rH
```

Note: You may have to give yourself execute permissions with `chmod 544 rrh46410-example`. If you get an error “Can't lock /var/run/pigpio.pid”, run the command, `sudo killall pigpiod`.

3.5 How to Compile for Raspberry Pi Hardware

This section provides guidelines for compiling the adapted source code into an executable file. This executable can be used on the Raspberry Pi like the original provided executable file. For compiling, make must be installed, which is a standard package in Raspberry Pi OS. The folder structure should be identical to that in the downloaded package.

1. Complete your code changes in the source code of the firmware package.
2. Open the Terminal and go to the directory containing the example code. For example,
   `cd ~/Downloads/Renesas-RRH46410-Firmware/src`
3. Type “make”, and a file called *rr46410-example* will be generated in the folder named *build*.
4. Go to the folder /build and start the example with the following command (sudo is required for pigpio package). Make sure to have the I²C interface enabled (for instructions, see “RRH46410 Example for Raspberry Pi”).

   ```
   sudo ./rrh46410-example
   ```

3.6 Error Codes

All API functions return a code to indicate the success of the operation. If no error occurred, the return code is zero. If an error occurs, a number is returned. The API has predefined symbols *RRH46410_ErrorCodes_t* for the error codes defined in *rrh46410.h*. If an error occurs, check the following table for solutions.

Table 3. Error Codes

Error Code	Return value	Description	Solution
RRH46410 sensor module related errors			
0	ecSuccess	No error.	
1	ecWarmup	Sensor is in stabilization phase, results may be inaccurate.	Wait for the amount of warm-up samples before checking results. Amount of warm-up samples depend on operation mode, see table "Sample Rates and Warm-Up Samples for RRH46410 Operation Modes" in the *RRH46410 Datasheet*.
8	ecCleaningCountExceeded	The maximum numbers of cleaning cycles ran on this sensor. *RRH46410_PerformCleaning* function has no effect anymore.	To protect from damage the cleaning cannot be used anymore on this sensor module. This error informs that execution has already taken place and can be ignored after first run.
9	ecPOR	An unexpected reset of the sensor occurred. Power or pin reset.	Check stability and noise sources of power supply and power/reset lines (e.g., for cross-talk). After a power-on reset the sensor module may have lost its configuration of operation mode. To avoid this, change the operation mode after startup. For more information, see the "Data Flash" section in the *RRH46410 Datasheet*.
10	ecDamage	Sensor may be damaged.	1. For information on how to handle sensor damage error, see the "Conditioning, Sensor Self-Check Status, and Stability" section in the *RRH46410 Datasheet*.
16	ecDataNotReady	Trying to read-out data before first sample is available.	1. Check the implementation of the timing function (delay). 2. Check if operation mode is set.
32	ecInternalComm	Internal communication error between module's MCU and ASIC.	Check stability and noise sources on power supply line (e.g., for cross-talk).
64	ecHostToSensorCheckusm	Checksum error when sending command to sensor module.	1. Check the low-level I^{2}C implementation. It is best to analyze the voltage levels of the SDA/SCL line and check if they match the pattern described in figure "I^{2}C Interface" in the *RRH46410 Datasheet*. Check your implementation also with multiple data bytes. 2. Check stability and noise sources on SDA/SCL line (e.g., for cross-talk).
128	ecInvalidCommand	An invalid command was sent to the sensor module.	Check your API implementation.
API related errors			
256	ecSampleNotNew	The sample ID was not updated	Check the implementation of the timing function (delay).
257	ecSensorToHostChecksum	Checksum error when receiving data from sensor module.	1. Check the low-level I^{2}C implementation. It is best to analyze the voltage levels of the SDA/SCL line and check if they match the pattern described in figure "I^{2}C Interface" in the *RRH46410 Datasheet*. Check your implementation also with multiple data bytes. 2. Check stability and noise sources on SDA/SCL line (e.g., for cross-talk).
258	ecCleaningTimeout	Timeout while cleaning is executed.	Check the implementation of the timing function (delay).

Error Code	Return value	Description	Solution
259	ecGPIOConfigInvalid	Wrong GPIO configuration.	Not all GPIOs can be configured as input and output. Check your bitmask used for GPIO configuration. For more information, see “Config GPIO” section in the *RRH46410 Datasheet*.

4. Adapting the Firmware Example for Target Hardware

4.1 System Hierarchy and Implementation Steps

The Renesas RRH46410 C API is located between the application and the hardware level.

Customer Application
Application-Specific Configuration of the Firmware Example
RRH46410 API
Hardware Abstraction Layer (HAL)
Hardware Level (RRH46410 and Target)

Figure 2. System Hierarchy

The RRH46410 example code uses a Hardware Abstraction Layer to separate target hardware implementation details from the actual sensor interface. To transfer the example to a different hardware platform, the following steps are recommended:

1. Establish I²C communication and conduct a register test. For more information, see the “Data Transmission Protocols” section in the *RRH46410 Datasheet*.
2. Use the file template.c in folder \src\hal\custom to implement hardware-specific functions and to adjust the HAL. A detailed description of the HAL interface is provided in the *RRH46410-Firmware-Documentation.pdf* and *.html*. For the _I2CRead should point to the I²C implementation of the hardware used (see example in comboard.c and escom.c files). HSxxx sensors will also need _I2CWrite. Define the _Sleep and optional _Reset function (see example in comboard.c file) and test them with a scope plot.
3. Copy the filled template and all unmodified RRH46410 API files into your project. Copy and adapt content of the *example.c* into your code and use the example code. Test if the adapted example runs and outputs changing values in your main measurement loop.

   Note: It is necessary to wait the amount of warm-up samples, see section “Sample Rates and Warm-Up Samples for RRH46410 Operation Modes” in *RRH46410 Datasheet*.

4.2 Interrupt Usage and Measurement Timing

The firmware example is written with delays. The microcontroller is blocked during these time periods. Depending on target hardware and the application, this can be avoided by using interrupts. The measurement intervals (sample rates) for each operation mode are described in the “Sample Rates and Warm-Up Samples for RRH46410 Operation Modes” section in the *RRH46410 Datasheet*.

The following interrupt usages are possible to detect the end of an active measurement:

- Using RRH46410s Interrupt pin (INT) – This pin is high when data is available and low when data was read out.
- Using Timer-based interrupts – As an alternative a timer interrupt can be used to wait until the end of the active measurement phase. The variable *measurementInterval* of the *rrh46410* structure contains the right time.

5. Revision History

Revision	Date	Description
1.00	Nov 8, 2023	Initial release.

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