SMART Home Real-Time IOT Device

Version 1.0

Technical Documentation

SMART Home

SMART Home Real-Time IOT Device
Technical Documentation

Purpose of Documentation

The purpose of this documentation is to give the reader a general picture about the *SMART Home Real-Time IOT Device*. Instead of listing every detail, the document is functioning as a *guide* to introducing the device.

Proprietary Notice

Product Status

The information in this document is not final. The device is under development status: Work-In-Progress.

Feedback on this product

If you have any comments or suggestions about this product, contact the owner of this document with the product name.

Contact Address

If you have any suggestions contact me:

horvathadamb@gmail.com

Contents

SMART Home Real-Time IOT Device
Technical Documentation

Introduction

This chapter describes the *SMART Home Real-Time IOT Device*. It contains the following sections:

- Introduction of *the SMART Home Real-Time IOT Device*
- Components

Introducing the SMART Home Real-Time IOS Device

SMART Home Real-Time IOT Device is a sensor platform intended for smart home applications or home monitoring. The main data processing unit is a STM32f411 Nucleo board, which is assigned to communicate with the sensors and process its raw data.

The main sensor component is the BME280 Environmental sensor, which is capable of measuring temperature, humidity and pressure. The sensor can communicate with the microcontroller using I2C communication protocol

The device is capable of Real-Time environmental monitoring using FreeRTOS. The device's main features are temperature, humidity and pressure measuring. The raw data coming from the sensor is converted to SI metrics.

After connecting the device with USB, the user can see the measured data sent with USART communication protocol.

Components

SMART Home Real-Time IOT Device contains the following:

- ARM Cortex-M4 microprocessor on the STM32f411 Nucleo board.
- Bosch BME280 Environmental sensor
- FreeRTOS for real-time operation software

About the STM32f411 Nucleo board

The STM32F411XC/XE devices are based on the high-performance Arm® Cortex® -M4 32- bit RISC core operating at a frequency of up to 100 MHz. The Cortex®-M4 core features a Floating-point unit (FPU) single precision which supports all Arm single-precision dataprocessing instructions and data types. It also implements a full set of DSP instructions and a memory protection unit (MPU) which enhances application security.

Figure 1. STM32f411 Nucleo board

The STM32F411xC/xE incorporate high-speed embedded memories (up to 512 Kbytes of Flash memory, 128 Kbytes of SRAM), and an extensive range of enhanced I/Os and peripherals connected to two APB buses, two AHB bus and a 32-bit multi-AHB bus matrix.

All devices offer one 12-bit ADC, a low-power RTC, six general-purpose 16-bit timers including one PWM timer for motor control, two general-purpose 32-bit timers. They also feature standard and advanced communication interfaces.

Source: www.st.com

About the Bosch BME280 Environmental sensor

General Description

The BME280 is a combined digital humidity, pressure and temperature sensor based on proven sensing principles. The sensor module is housed in an extremely compact metal-lid LGA package with a footprint of only 2.5 x 2.5 mm2 with a height of 0.93mm. Its small dimensions and its low power consumption allows the implementation in battery driven devices.

The BME280 achieves high performance in all applications requiring humidity and pressure measurement. The humidity sensor provides an extremely fast response time for fast context awareness applications and high overall accuracy over a wide temperature range.The pressure sensor is an absolute barometric pressure sensor with extremely high accuracy and resolution.

The integrated temperature sensor has been optimized for lowest noise and highest resolution. Its output is used for temperature compensation of the pressure and humidity sensors and can also be used for estimation of the ambient temperature.

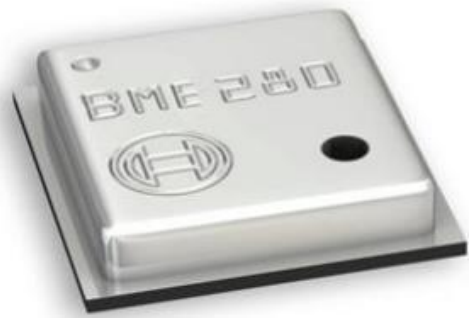

Figure 2. Bosch BME280 Environmental sensor

Operating range:

- Temperature: -40…..+85 °C
- Relative humidity: 0….100%
- Pressure: 300…..1100 hPa

Register Name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Reset state
hum_lsb	0xFE	hum_lsb<7:0>								0x00
hum_msb	0xFD	hum_msb<7:0>								0x80
temp_xlsb	0xFC	temp_xlsb<7:4>				0	0	0	0	0x00
temp_lsb	0xFB	temp_lsb<7:0>								0x00
temp_msb	0xFA	temp_msb<7:0>								0x80
press_xlsb	0xF9	press_xlsb<7:4>				0	0	0	0	0x00
press_lsb	0xF8	press_lsb<7:0>								0x00
press_msb	0xF7	press_msb<7:0>								0x80
config	0xF5	t_sb[2:0]			filter[2:0]				spi3w_en[0]	0x00
ctrl_meas	0xF4	osrs_t[2:0]			osrs_p[2:0]			mode[1:0]		0x00
status	0xF3					measuring[0]			im_update[0]	0x00
ctrl_hum	0xF2						osrs_h[2:0]			0x00
calib26..calib41	0xE1…0xF0	calibration data								individual
reset	0xE0	reset[7:0]								0x00
id	0xD0	chip_id[7:0]								0x60
calib00..calib25	0x88…0xA1	calibration data								individual

Registers:	Reserved registers	Calibration data	Control registers	Data registers	Status registers	Chip ID	Reset
Type:	do not change	read only	read / write	read only	read only	read only	write only

Figure 3. Bosch BME280 Memory map

Source: Bosch Sensortec

About the FreeRTOS Real-Time Operating System

FreeRTOS is a class of RTOS that is designed to be small enough to run on a microcontroller - although its use is not limited to microcontroller applications.

Microcontrollers are used in deeply embedded applications (those applications where you never actually see the processors themselves, or the software they are running) that normally have a very specific and dedicated job to do. The size constraints, and dedicated end application nature, rarely warrant the use of a full RTOS implementation - or indeed make the use of a full RTOS implementation possible. FreeRTOS therefore provides the core real time scheduling functionality, inter-task communication, timing and synchronization primitives only. This means it is more accurately described as a real time kernel, or real time executive. Additional functionality, such as a command console interface, or networking stacks, can then be included with add-on components. Source: www.freertos.org

Figure 4. FreeRTOS real-time operating system

Connecting the sensor

The BME280 Environmental sensor has the following pins connected:

Power pins:

- Vin is the power pin. The sensor chip uses 3V DC, with an integrated voltage regulator that will take 3-5V DC. Connected to the microcontroller 3.3V pin.
- 3Vo is the 3.3V output from the voltage regulator, not used.
- GND is the common ground for power and logic, connected to GND

I2C logic pins:

- SCK is the I2C clock pin connected to the microcontroller SCL pin.
- SDI is the I2C data pin, connected to the microcontroller SDA pin.

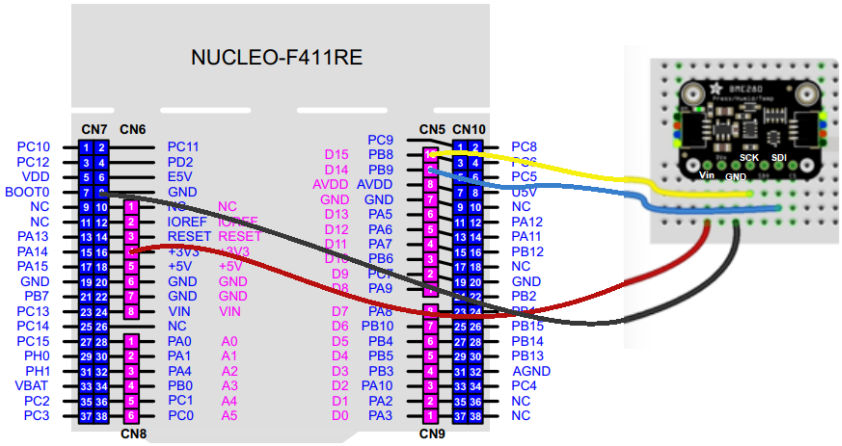

Figure 4. Sensor wiring

Peripherals

The main peripherals used in the project are the following:

GPIO

The GPIO or the General-Purpose Input Output pins are necessary for the device. The Peripherals (I2C, USART, LED) are using input and outputs as a connection to the microcontroller.

USART

The USART or Universal Synchronous Asynchronous Receiver Transmitter is used for the communication between the user PC and the microcontroller.

I2C

The I2C or Inter-Integrated Circuit protocol is used for the communication between the BME280 Environmental sensor and the microcontroller

LED

The STM32F411 has USER LED built in, used for showing process status.

Tasks

The device has four independent tasks, executed every x millisecond. Every task has a different priority depending on the importance of it.

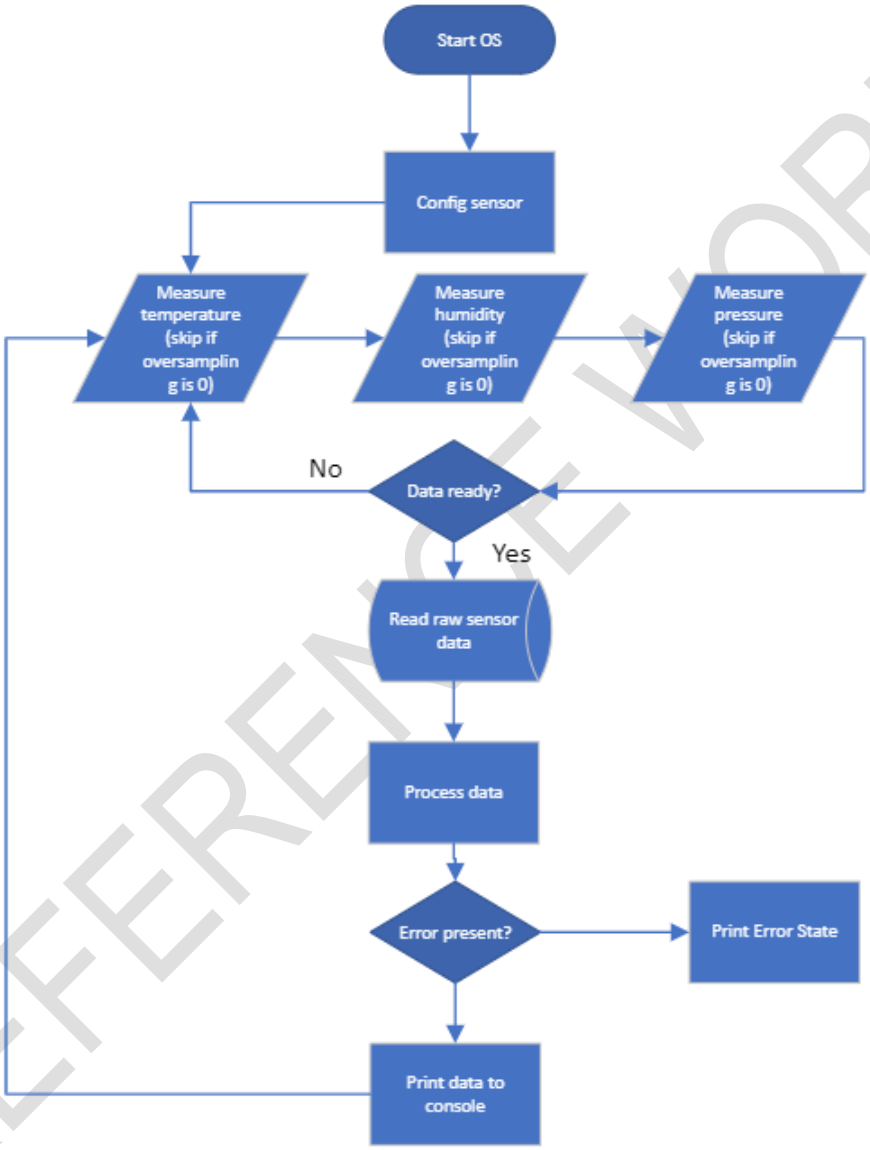

Figure 5. Sensor wiring

1. Config task

Before using the BME280, the user needs to set:

- Sampling time
- IIR filter
- Oversampling for temperature, pressure and humidity
- Sensor Mode

Highest priority task gets executed first. Configures the sensor. After the config is done, the task will read the config and control registers to make sure it is set up properly. Later in the data processing task, the calculation needs calibration values provided by the sensor manufacturer. These are stored in 8-bit registers and also read and parsed into 16-bit constants in this task.

This task gets deleted after executing the configuration.

2. Read task

The BME280 Sensor has eight 8bit registers to store raw temperature, pressure and humidity data. The temperature and pressure data is 20bit, the humidity is a 16bit value. According to the datasheet it is best to read these values in a single burst. The read task is uses the bme280_I2C_read function, which reads the 8 addresses consecutively.

When the reading is in progress the USER LED2 is blinking to indicate the process.

3. Process task

After measuring the environment and reading the raw sensor data, the device can start processing the data. To calculate Celsius, Pascal and relative humidity % from raw data, the program is using calculations provided by BOSCH Sensortec.

4. Print task

After the data processing is done, the device is sending the complete values to the terminal via USART communication.

```
Apr  5 2022 13:54:17
Configuring sensor....
Sensor configuratin was successful!
Temperature:  24,34 °C
Pressure: 997 hPa
Humidity: 32,82%
```

Figure 6. Terminal

The values are only printed out to the terminal if there is no error present during the process. In case an error occurs, the print task is going to print the error code instead.

Driver

The user_BME280 driver consist of two files:

- User_bme280.h Header file
- User_bme280.c Source file

1. user_bme280.h

The header file contains the definitions of:

- Error table
- Register address
- Config options
- Function prototypes

2. user_bme280.c

In the source file the reader can find the main functions required to operate the SMART Home device. The functions are the following:

- **bme280_I2C_read**: use to read the BME280 sensor registers
- **bme280_I2C_write**: use to write the BME280 sensor registers
- **bme280_measure_temperature_int32**: measures temperature, returns value in °C*100
- **bme280_measure_pressure_int32**: measures pressure, returns value in Pa
- **bme280_measure_humidity_int32**: measures humidity, returns relative humidity %
- **parse_compensate**: parse 8-bit calibration values into 16-bit constants.

Error handling

To ensure proper functionality the device can handle various errors throughout the whole process. The following errors are stored in an Enum array: BME280_ErrorStatus. The returned error codes are the following:

Error name	Value
No_Error	0x00
Device_ID_Error	0x04
NullPointer_Error	0x08
Config_Error	0x16
CTRL_Hum_Error	0x20
CTRL_Meas_Error	0x22
I2C_Error	0x32

Table 1. Error values

Source code

The source code of this project can be found in the project repository on GitHub:

https://github.com/horvathadamb/SMART_Home