Interface por I2C com um sensor de pressão, temperatura e humidade: BME280

Irene Garcia do Amaral, up201506644

Luís Pedro Silva Ermida Martins de Freitas , up201605641

Janeiro 2019

Conteúdo

1 Introdução

O presente relatório foi realizado no âmbito da unidade curricular Computadores, do curso Mestrado Integrado em Engenharia Eletrotécnica e de Computadores, da Faculdade de Engenharia do Porto e tem como objetivo a apresentação e descrição do trabalho realizado sobre o sensor BME280.

Com o desenvolvimento deste projeto tencionamos obter um melhor conhecimento em relação à forma como os dados dos sensores são tratados e como são obtidos através do protocolo I2C.
O sensor BME280 é uma criação da Bosch, desenvolvido especificamente para ser utilizado em aplicações de smartphones e cuja finalidade é medir a temperatura, humidade e pressão atmosférica do meio que o rodeia. Tem, por isso, um tamanho reduzido e consumo necessário mínimo.

2 Comunicação

O protocolo utilizado para realizar a interligação do microcontrolador ChipKIT Uno32 com o sensor BME280 foi o I2C (Inter-Integrated Circuit), um protocolo de comunicação série com múltiplas vantagens:

- é síncrono: um dos sinais envolvidos é um sinal de relógio, o que permite transferências rápidas.
- utiliza uma estratégia master-slave: o microcontrolador toma o papel de master que, ao gerar o sinal de relógio, comanda a troca de dados com os outros dispositivos (os slaves)
- possui comunicação full-duplex: a realização em simultâneo do envio e receção de dados, devido à implementação de duas linhas: SDA (Serial Data) e SCL (Serial CLock).

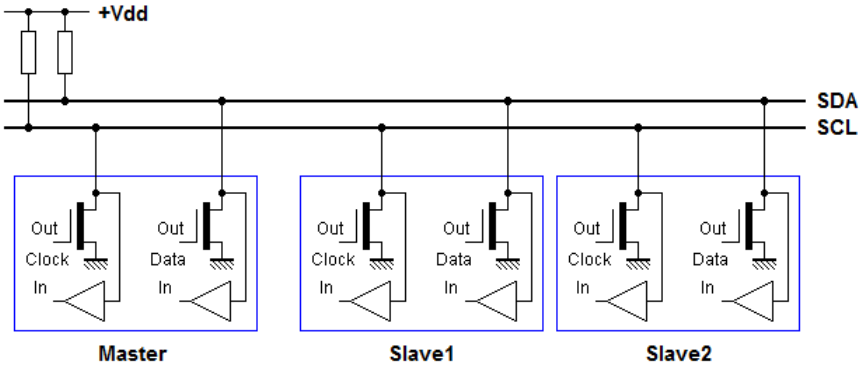


Figura 1: Protocolo I2C

O protocolo I2C exige que haja um mecanismo de escolha de slave de modo a que seja possível enviar e ler dados para o destino correto sem causar conflitos (e. g. curto-circuitos, perda de informação), logo cada slave tem um endereço de identificação de 7 bits. No caso no sensor BME280, o seu endereço é 1110110 (0x76).

A escrita é efetuada enviando primeiramente um sinal S (Start) seguido do endereço do slave a escrever (0x76) no modo de escrita (RW=0), resultando no endereço 11101100. Depois são enviados aos pares o byte de controlo e o byte de dados, intercalados por ACKS (Acknowledge by slave). A transmissão termina com um sinal P (Stop).

A leitura tem a particularidade de que é necessário realizar primeiro uma escrita com a finalidade de escolher o registo a ler e só depois é efetuada a leitura propriamente dita. Portanto começamos por enviar um sinal S, seguindo do endereço do slave no modo de escrita e de um byte de controlo, terminando com um sinal P. Posteriormente começa a sua leitura, enviando de novo um sinal S e o endereço do slave no modo de leitura (RW=1), resultando no endereço 11101101. É também enviado o byte de dados intercalado por ACKS. A transmissão termina com um NACKM (Not acknowledge by master) e um sinal P.

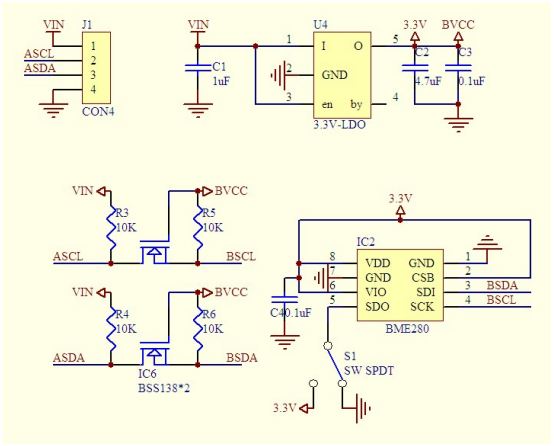
3 Montagem

Para a realização deste projeto foi necessário o seguinte material: uma breadboard, um sensor BME280, uma placa ChipKIT UNO32, o terminal PuTTY, a plataforma PlatformIO e o IDE Eclipse.

Primeiramente estudamos o esquemático do sensor, percebendo que o sensor possui 4 pinos:

- GND: é ligado ao GND da ChipKIT UNO32
- VCC: é ligado aos 3.3V da ChipKIT UNO32
- SDA: é ligado ao pino A4 da ChipKIT UNO32; JP6 na posição RG3
- SCL: é ligado ao pino A5 da ChipKIT UNO32; JP8 na posição RG2

A partir do esquemático concluímos também que o sensor possui internamente as 2 resistências pull-up essenciais para evitar curto-circuitos nas linhas SDA e SCL, tal como referido anteriormente na Comunicação, portanto não será necessário a sua implementação exterior.
As resistências pull-up evitam curto-circuitos ao puxar para o GND (0V) a respetiva linha aquando um dos transístor conduz.



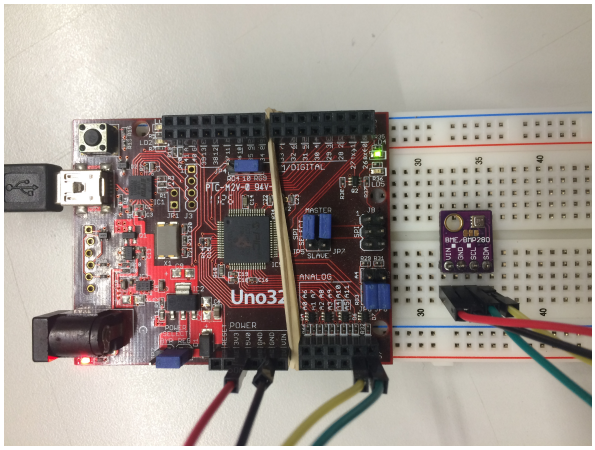


Figura 2: Esquema do módulo BME280

Figura 3: Placa CHIPKit Uno32 e o sensor BME280

4 Sensor BME280

4.1 Modo de operação recomendado

O sensor pode ser adaptado consoante o cenário que o envolve, para tal, possui múltiplas formas de operação: monitoramento meteorológico, detetor de humidade, climatologia de interiores e modo desportivo.
Consideramos que o modo de operação mais apropriado ao nosso trabalho é a climatologia de interiores. Desta escolha são-nos dadas recomendações relativamente às opções de oversampling, de filtro e de modo do sensor.

Suggested settings for indoor navigation	
Sensor mode	normal mode, $t_{standby}$ = 0.5 ms
Oversampling settings	pressure ×16, temperature ×2, humidity ×1
IIR filter settings	filter coefficient 16
Performance for suggested settings	
Current consumption	633 µA
RMS Noise	0.2 Pa / 1.7 cm
Data output rate	25Hz
Filter bandwidth	0.53 Hz
Response time (75%)	0.9 s

Figura 4: Configurações e desempenho para climatologia de interiores

4.2 Sensor Mode

O sensor possui três modos distintos de atuar: sleep mode, forced mode e normal mode. Por defeito, quando ligamos o sensor este inicia em sleep mode, cujo modo não permite a execução de medições, logo é necessário optar por um dos outros dois modos. Conforme nos foi aconselhado, para climatologia de interiores, devemos colocar o dispositivo em normal mode. Esta mudança é efetuada através de uma escrita no registo *crtl_meas* (address= 0xF4), especificamente nos seus dois LSB, mode[1:0]=11.

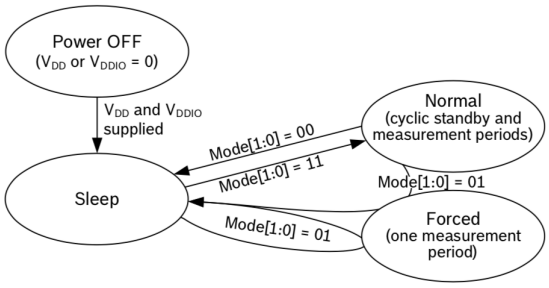


Figura 5: Diagrama de estados dos modos do sensor BME280

4.3 Descrição dos Registos

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

Figura 6: Mapa da memória do sensor BME280

4.3.1 Registo de Identificação

O sensor BME280 é compatível a nível dos registos com o sensor BMP280, logo é necessário identificar com qual dos sensores estamos a trabalhar. Para isso efetuamos uma leitura do registo *id* (address= 0xD0) o qual poderia retornar 0x56/0x57/0x58 caso se tratasse do sensor BMP280 ou retornar 0x60 caso se tratasse do sensor BME280. Efetivamente o valor retornado foi 0x60 confirmando que estamos a trabalhar com o sensor BME280.

4.3.2 Registos de Configuração

De forma a obter uma correta medição dos valores de humidade, temperatura e pressão, os respetivos registos *ctrl_hum* (address= 0xF2) e *ctrl_meas* (address= 0xF4) necessitam de ajustes. Estes registos possuem bits específicos para controlar o nível de oversampling pretendido, nos quais executamos uma escrita com os valores correspondentes ao modo de climatologia de interiores, nomeadamente osrs_h[2:0]=001 (x1), osrs_t[2:0]=010 (x2), osrs_p[2:0]=101 (x16).

*osrs_*[2:0]	oversampling
000	Skipped (output set to 0x8000)
001	oversampling ×1
010	oversampling ×2
011	oversampling ×4
100	oversampling ×8
101, others	oversampling ×16

Figura 7: Tabela de configuração: oversampling

t_sb[2:0]	$t_{standby}$ [ms]
000	0.5
001	62.5
010	125
011	250
100	500
101	1000
110	10
111	20

Figura 8: Tabela de configuração: tempo de standby

filter[2:0]	Filter coefficient
000	Filter off
001	2
010	4
011	8
100, others	16

Figura 9: Tabela de configuração: coeficiente do filtro IIR

Como é referido também no modo de climatologia de interiores, devemos implementar um periodo de standby e efetuar um ajuste do coeficiente do filtro IIR. Esta execução é feita através de uma escrita no registo *config* (address= 0xF5), especificamente nos bits 7, 6, 5 para o período de standby: t_sb[2:0]=000 (t=0.5ms); e nos bits 4, 3, 2 para o filtro IIR: filter[2:0]=100 (coeficiente=16).
Alertamos que o registo *config* apenas sofre alterações quando a escrita é executada durante o sleep mode.

4.3.3 Registos da humidade

A humidade é obtida ao aceder aos registos *hum_lsb* (address= 0xFE) e *hum_msb* (address= 0xFD), os quais contêm respetivamente os 8 LSB e os 8 MSB, sendo portanto constituída por um total de 16 bits.

4.3.4 Registos da temperatura

A temperatura é obtida ao aceder aos registos *temp_xlsb* (address= 0xFC), *temp_lsb* (address= 0xFB) e *temp_msb* (address= 0xFA), os quais contêm respetivamente os 4 LSB, os 8 bits intermédios e os 8 MSB, sendo portanto constituída por um total de 20 bits.

4.3.5 Registos da pressão

A pressão é obtida ao aceder aos registos *press_xlsb* (address= 0xF9), *press_lsb* (address= 0xF8) e *press_msb* (address= 0xF7), os quais contêm respetivamente os 4 LSB, os 8 bits intermédios e os 8 MSB, sendo portanto constituída por um total de 20 bits.

4.4 Compensação de resultados

Os valores do sensor BME280 resultam a partir dos valores ADC. Contudo, cada elemento a medir comporta-se de forma distinta, por isso, a pressão, humidade e temperatura retirada dos registos deve ser recalculada usando parâmetros de calibração. Estes parâmetros estão definidos nos registos *calib00..calib25* (address= 0x88..0xA1) e *calib26..calib41* (address= 0xE1..0xF0). O recalculo das grandezas meteorológicas é efetuado com base em fórmulas apresentadas na data sheet do sensor (pág.23).
Por último, o resultado final deve ser apresentado consoante a resolução de cada grandeza: temperatura: 0.01°C; humidade: 0.008%RH; pressão: 0.18Pa.

5 Resultados

5.1 Aquecimento da placa

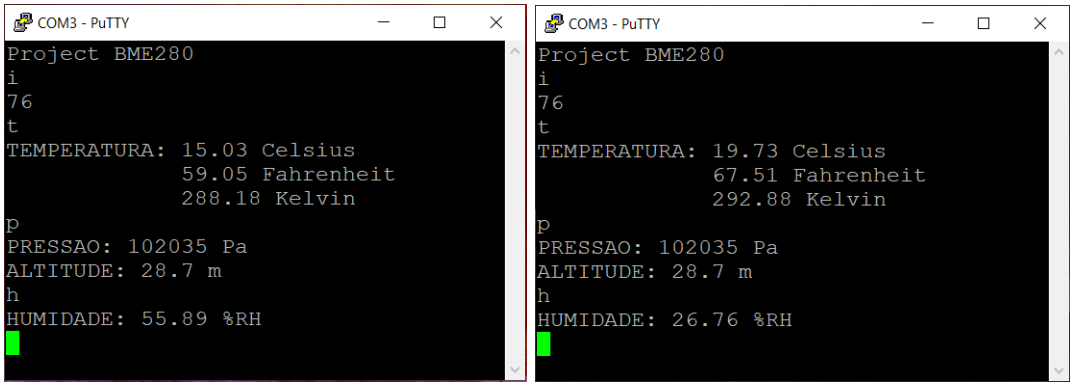


Figura 10: Resultados ao aquecer a placa

É através do terminal PuTTy que observamos os resultados obtidos das grandezas meteorológicas e também o endereço I2C do sensor.
Para verificar a veracidade e coerência dos resultados alcançados:

- **Temperatura:** Expusemos o sensor a um secador de cabelo provocando um aumento da temperatura de alguns graus. Um método mais simples será pressionar ligeiramente o sensor com o dedo, o que causa igualmente um aumento da temperatura, no entanto, mais lento e não tão acentuado.
- **Humidade:** Expusemos o sensor a um secador de cabelo provocando uma diminuição acentuada da humidade.
- **Pressão atmosférica:** Não foi possível realizar um ensaio que alterasse o valor da pressão, no entanto sabemos que o valor da pressão padrão é de 100 000 Pa, logo podemos afirmar que o resultado é razoável.
- **Altitude:** A altitude foi calculada através de uma fórmula que a relaciona com a pressão. Como não foi possível variar a pressão, a altitude também não sofreu alterações.

Consideramos relevante citar a seguinte informação que consta na data sheet do sensor:
"Please note that it is strongly advised to use the API available from Bosch Sensortec to perform readout and compensation. If this is not wanted, the code below can be applied at the user's risk."
A finalidade desta citação serve para alertar de um provável motivo de que os valores não são absolutamente exatos.

6 Conclusão

O trabalho realizado teve um papel importante em relação à compreensão e dominação prática do protocolo I2C, da biblioteca Arduino e do tratamento de dados em sensores. Estes objetivos didáticos foram alcançados.
Um dos objetivos iniciais, para além dos relatados, era implementar um display LCD que apresentasse os resultados tanto de forma escrita como gráfica. Embora não tendo sido cumprido, é um futuro progresso deste trabalho.

7 Bibliografia

- https://cdn-shop.adafruit.com/datasheets/BST-BME280_DS001-10.pdf
- arduino.cc/en/Reference/Wire
- https://paginas.fe.up.pt/~hsm/docencia/comp/spi-e-i2c/
- https://github.com/Seeed-Studio/Grove_BME280?fbclid=IwAR3iY88DXzNIqpCo_gGhpun1BPbAllnhH__2K8OqmqIAZ_kqZ1fC4AyHP-o