

Trabalho Prático 3

**Computação Física**

Curso de Licenciatura Informática e Multimédia (LEIM)

Ano Letivo 2017/2018

**Data: 04/06/2018**

**Turma: LEIM 23D**

**Docente:**

Eng. Jorge Pais

Eng.Carlos Carvalho

**Grupo:** 5

**Alunos:**

Luis Fonseca (A45125)

Gabriel Diaz(A45133)

Philipp Al-Badavi(A45138)

**Índice**

1.Introdução………………………………………………………………...3

2.Sensor BMP180………………………….……………………………….4

3.Protocolo I2C……………………………………………………………..6

4.Display I2C……………………………………………………………….7

5.Diagrama de Ligações…………………………………………………….8

6.Instruções do Display…………………………………………………….8

7.Automato………..………………………………………………………..9

8.Código Arduíno………………………………………………………....10

9.Código Python.……………………………………………………….....21

10.Conclusões...…………………………………………………………...27

11.Bibliografia…………………………………………………………….27

1. **Introdução**

A realização deste projeto tinha por objetivo o estudo de um sensor de temperatura e pressão atmosférica (BMP180) e um display, ambos com protocolo de comunicação I2C. Para o estudo do sensor foi disponibilizado o “*data sheet”* que contem todas as informações necessárias para utilizar o mesmo de forma correta. Por sua vez a informação do display encontrava-se disponibilizada nas folhas do Engenheiro.

Retirada a informação necessária, foi feita uma implementação no Arduíno, no qual foi calculada o valor da pressão, da altitude e da temperatura. Com estes passos concluídos, procedeu-se ao envio destes valores para o display.

Por último foi necessária a realização de uma estação meteorológica usando o programa *Python*. Para isso foi feita a ligação do Arduíno ao *python*, e através do *pygame*, representamos então os valores que eram pretendidos.

1. **Sensor BMP180**

O sensor BMP180 a estudar, criado pela bosch, tem uma interface de saída I2C. Por sua vez, este sensor tem como funcionalidade cálculo da temperatura (ºC) e a pressão (Pa). Este sensor também é adequado para calcular a altitude acima do nível do mar com uma precisão de 1 metro pois este calcula a pressão atmosférica compensada pela temperatura ambiente.

Para a obtenção dos mesmos é necessário seguir uma série de procedimentos disponibilizados no “*data sheet”* fornecido pelo fabricante.

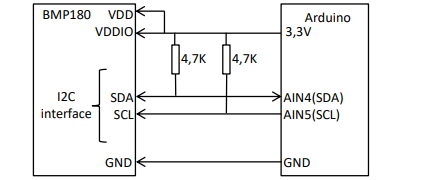
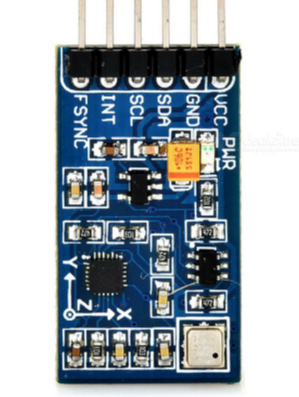
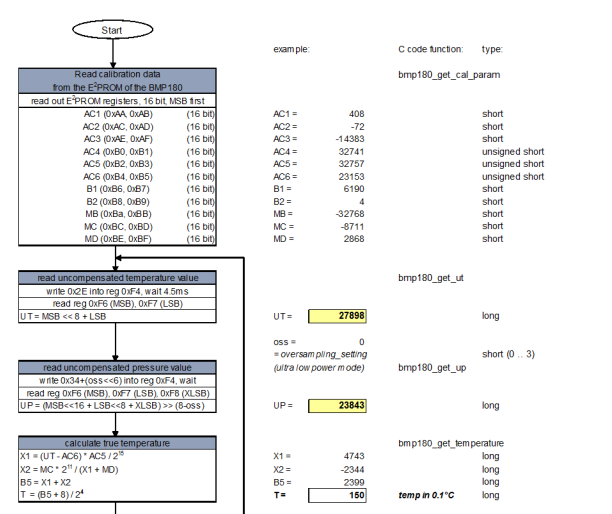


Figura 2-Sensor BMP180

Figura 1-Diagrama ligações do sensor ao Arduíno

A figura a seguir mostra o algoritmo detelhado para medição de pressão e temperatura.



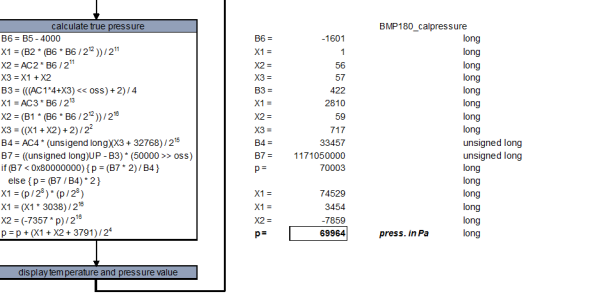


Figura 3-Cálculo da pressão e temperatura

Com uma pressão medida p e a pressão ao nível do mar p0= 1013.25hPa, a altitude em metros pode ser calculada com seguinte formula barométrica:

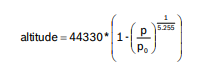


Figura 4-Cálculo da altitude

1. **Protocolo I2C**

O procolo I2C é baseado em dois sinais digitais, o SDA (Serial Data) e o SCL (Serial Clock) e tem como função a comunicação entre dispositivos. SDA é o pino que efetua a transferencia de dados, e SCL tem como função a temporização entre os despositivos. O protocolo tem uma condicão de start, na qual há uma transição descendente em SDA enquanto o SCL tem o valor de 1, também este possui uma codição de stop que ocorre quando há uma transição ascendente do sinal SDA estando o sinal SCL com o valor de 1.

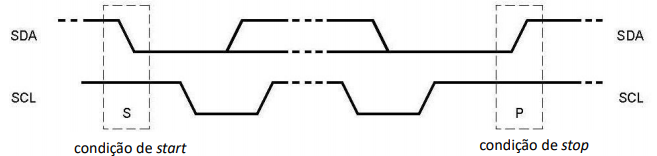


Figura 5-Condições de star e stop do protocolo I2C

1. **Display I2C**

O display I2C é um de display de 2 linhas com 16 carateres em cada linha. O display disponibiliza uma interface I2C para comunicação com um microcontrolador. Por cada byte enviado via I2C, a interface I2C-paralelo define 4 sinais de dados para o display e 4 sinais de controlo de acordo com a figura 5. O endereço I2C do display é o 27h.

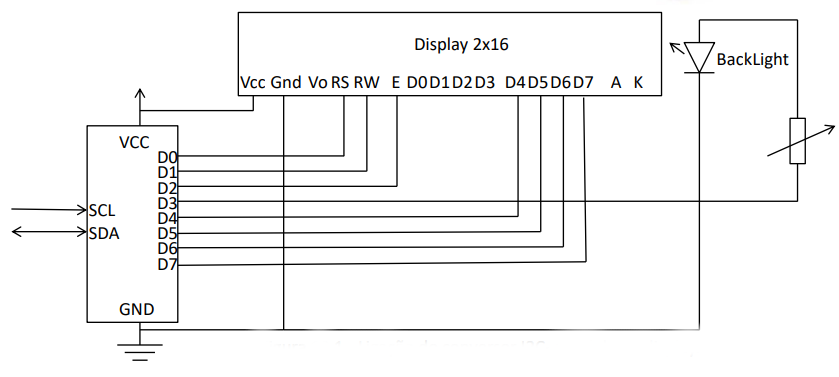
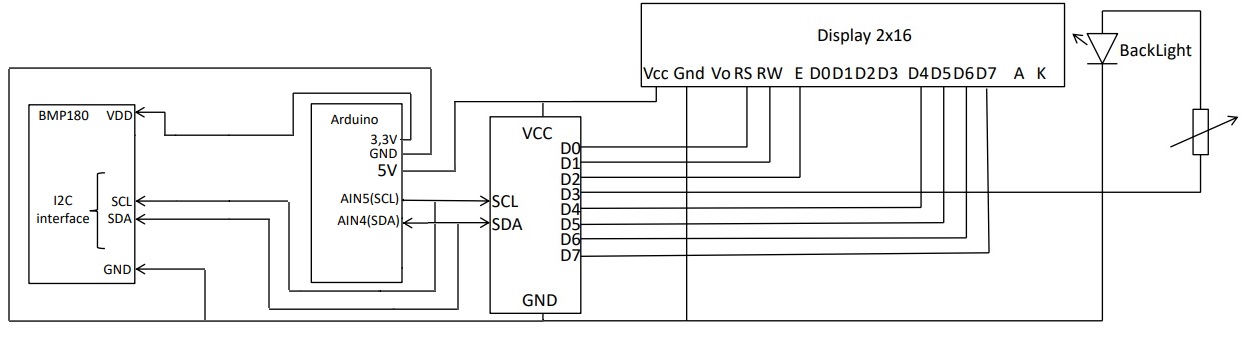


Figura 6-Ligação do conversor I2C



Figura 7-Display I2C

1. **Diagrama de ligações**
2. **Instruções do Display**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Função | Tipo | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Tempo |
| Clear Display | Comando | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 5ms |
| Cursor Home | Comando | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5ms |
| Entry Mode | Comando | 0 | 0 | 0 | 0 | 0 | 1 | I/D | S | 5us |
| Display On/Off | Comando | 0 | 0 | 0 | 0 | 1 | D | C | B | 120us |
| Cursor & Display shift | Comando | 0 | 0 | 0 | 1 | S/C | R/L | x | x | 120us |
| Function set | Comando | 0 | 0 | 1 | DL | N | F | x | x | 120us |
| Set Cursor Adress | Comando | 1 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | 120us |
| Write Data to cursor Position | Dados | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | 120us |

* **Legenda:**

I/D = 1 - incrementa a posição do cursor.

I/D= 0 - decrementa a posição do cursor.

S= 1 - permite display shift.

D= display, C= cursor, B= blink (1= ON, 0= Off).

S/C= 1 - display shift, S/C= 0 - cursor move.

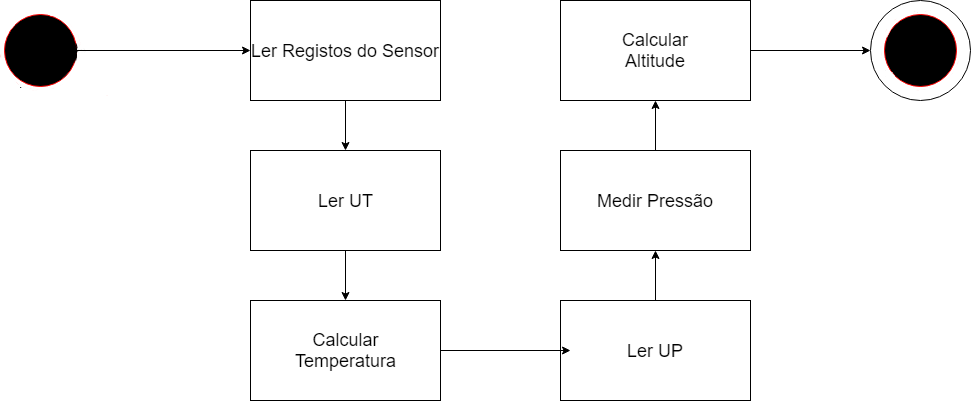
R/L= 1 - shift right, R/L= 0 - shift left.

DL= 1 - 8 bits de data, DL= 0 - 4 bits de data.

N= 1 - display 2 linhas, N= 0 - display de 1 linha.

F=1 - (5x10 dots só em 1 linha), F= 0 (5x7 dots em 1 ou 2 linhas).

1. **Autómato**



1. **Código Arduíno**

#include <Wire.h>

#define BMP180\_ADDRESS 0x77 // I2C adress BMP180

int OSS = 0; // Oversampling Setting

int AC1;int AC2;int AC3;

unsigned int AC4;unsigned int AC5;unsigned int AC6;

int b1;int B2;int MB;int MC;int MD;

long UT;long UP;long X1, X2,X3;long T;

long B3,B5,B6;

unsigned long B4,B7;

long p;

const float p0 = 1013.25; // Pressao ao nivel do mar

float altitude;

void setup()

{

Serial.begin(9600);

Wire.begin();

}

void loop(){

LerCalibracao();

LerUT();

LerUP();

CalcularTemperatura();

CalcularPressao();

altitude = (float)44330 \* (1 - pow(((float) (p\*0.01)/p0), 0.1902949572));

Serial.println(" ");

Serial.print(T\*0.1);Serial.println(" C");

Serial.print(p\*0.01);Serial.println(" hPa");

Serial.print(altitude);Serial.println(" m");

Serial.println();

iniciarDisplay();

setcursor(0,0);

limpaLinha();

setcursor(1,0);

limpaLinha();

setcursor(0,0);

printString("T:" + (String) (T\*0.1) + " C");

setcursor(1,0);

printString("Pres:" + (String) (p\*0.01) + " hPa");

delay(2000);

setcursor(0,0);

limpaLinha();

setcursor(0,0);

printString("Alt:" + (String) (abs (altitude)) + " m");

delay(2000);

}

void LerCalibracao(){

AC1 = LerRegisto16Bits(0xAA);

AC2 = LerRegisto16Bits(0xAC);

AC3 = LerRegisto16Bits(0xAE);

AC4 = LerRegisto16Bits(0xB0);

AC5 = LerRegisto16Bits(0xB2);

AC6 = LerRegisto16Bits(0xB4);

b1 = LerRegisto16Bits(0xB6);

B2 = LerRegisto16Bits(0xB8);

MB = LerRegisto16Bits(0xBA);

MC = LerRegisto16Bits(0xBC);

MD = LerRegisto16Bits(0xBE);

}

byte LerRegisto8Bits(unsigned char address){

Wire.beginTransmission(BMP180\_ADDRESS);

Wire.write(address);

Wire.endTransmission();

Wire.requestFrom(BMP180\_ADDRESS, 1);

while(Wire.available() < 0 );

return Wire.read();

}

int LerRegisto16Bits(unsigned char address){

byte MSB, LSB;

Wire.beginTransmission(BMP180\_ADDRESS);

Wire.write(address);

Wire.endTransmission();

Wire.requestFrom(BMP180\_ADDRESS, 2);

while(Wire.available()<2);

MSB = Wire.read();

LSB = Wire.read();

return (int) MSB<<8 | LSB;

}

void EscreverRegisto16Bits(int address, int value){

Wire.beginTransmission(BMP180\_ADDRESS);

Wire.write(address);

Wire.write(value);

Wire.endTransmission();

}

void LerUT(){

// Escrever 0x2E no registo 0xF4

EscreverRegisto16Bits(0xF4,0x2E);

// Esperar 4.5s pelo menos

delay(5);

UT = LerRegisto16Bits(0xF6);

}

// Read the uncompensated pressure value

void LerUP(){

byte MSB, LSB,XLSB;

EscreverRegisto16Bits(0xF4,0x34 + (OSS<<6));

delay(2 + (3<<OSS));

// Ler registos 0xF6 (MSB), 0xF7 (LSB), and 0xF8 (XLSB)

MSB = LerRegisto8Bits(0xF6);

LSB = LerRegisto8Bits(0xF7);

XLSB = LerRegisto8Bits(0xF8);

UP = (((unsigned long) MSB << 16) | ((unsigned long) LSB << 8) | (unsigned long) XLSB) >> (8-OSS);

}

long CalcularTemperatura(){

X1 =((UT - AC6)\*AC5)>>15;

X2 = ((long)MC<<11)/(X1 + MD);

B5 = X1 + X2;

T = (B5 + 8)>>4;

}

long CalcularPressao(){

B6 = B5 - 4000;

X1 =(B2 \*((long)(B6\*B6>>12)))>>11;

X2 = (AC2\*B6)>>11;

X3 = X1+X2;

B3 = (((AC1\*4+X3)<<OSS)+2)/4;

X1 = (AC3\*B6)>>13;

X2 = ((long)b1 \* ((long)(B6\*B6)>>12))>>16;

X3 =((X1+X2)+2)>>2;

B4 = AC4\*(unsigned long)(X3+32768)>>15;

B7 = ((unsigned long)UP-B3)\*(50000>>OSS);

if (B7 < 0x8000000) {

p = (B7\*2)/B4;

}

else {

p = (B7/B4)\*2;

}

X1 = (p>>8)\*(p>>8);

X1 = (X1 \* 3038)>>16;

X2 = (-7357 \* p)>>16;

p = p + ((X1 + X2 + 3791)>>4);

}

* **Código Display**

#define RS 0x01

#define RW 0x02

#define EN 0x04

#define LUZ 0x08

#define ENDERECO 0x3F

#include <Wire.h>

void escreverDados4(byte quatrobits) {

Wire.beginTransmission(ENDERECO);

Wire.write((quatrobits << 4) | LUZ| RS);

Wire.endTransmission();

Wire.beginTransmission(ENDERECO);

Wire.write((quatrobits << 4) | LUZ | RS | EN );

Wire.endTransmission();

delayMicroseconds(1); // enable ativo >450ns

Wire.beginTransmission(ENDERECO);

Wire.write((quatrobits << 4) | LUZ | RS);

Wire.endTransmission();

delayMicroseconds(40); // tempo > 37us para comando fazer efeito

}

void escreverComandos4(byte quatrobits){

Wire.beginTransmission(ENDERECO);

Wire.write((quatrobits << 4) | LUZ);

Wire.endTransmission();

Wire.beginTransmission(ENDERECO);

Wire.write((quatrobits << 4) | LUZ | EN );

Wire.endTransmission();

delayMicroseconds(1); // enable ativo >450ns

Wire.beginTransmission(ENDERECO);

Wire.write((quatrobits << 4) | LUZ );

Wire.endTransmission();

delayMicroseconds(40); // tempo > 37us para comando fazer efeito

}

void iniciarDisplay(){

//Init display

delay(50);

escreverComandos4(0x3);

delay(5);

escreverComandos4(0x3);

delayMicroseconds(200);

escreverComandos4(0x3);

escreverComandos4(0x2);

//Function set

escreverComandos4(0x2);

escreverComandos4(0x8);

delayMicroseconds(120);

//Blink ON curson ON display ON

escreverComandos4(0x0);

escreverComandos4(0xF);

delayMicroseconds(120);

//Set cursor

escreverComandos4(0x8);

escreverComandos4(0x0);

delayMicroseconds(120);

}

void setcursor(int linha, int coluna){

if( 0 <= linha <= 1 and 0 <= coluna <= 15){

byte MSB = (byte)((linha \* 4) | 0x8);

byte LSB = (byte) coluna;

escreverComandos4(MSB);

escreverComandos4(LSB);

delayMicroseconds(120);

}

}

void printChar(char c){

escreverDados4(c >> 4); // 4 primeiros

escreverDados4(c); //4 ultimos

delayMicroseconds(120);

}

void printString(String s){

for(int i=0; i < s.length(); i++){ //Print de uma string com o printChar

printChar(s[i]);

}

}

void clearDisplay(){

escreverComandos4(0x0);

escreverComandos4(0x1);

}

void limpaLinha(){

printString(" ");

}

void limpaPosicao(){

printChar(" ");

}

1. **Código Python**

import serial

import time

import pygame

import sys

import datetime

pygame.init()

screen = pygame.display.set\_mode([800,800])

timer = pygame.time.Clock()

t0 = pygame.time.get\_ticks()

green =(0,180,0)

cor\_fundo = (255,255,255)

fundo = pygame.image.load("fundo.jpg")

hora = pygame.image.load("hora.jpg")

def Windowhour():

screen.blit(hora, [0,0])

def Window():

screen.fill(cor\_fundo)

screen.blit(fundo, [0,0])

com = 'COM3'

baudrate = 9600

def comInit(com, baudrate):

try:

Serie = serial.Serial(com, baudrate)

print("Sucesso na ligacao ao Arduino")

print("Ligado ao " + Serie.portstr)

return Serie

except Exception as e:

print("Insucesso na ligacao ao Arduino")

print (e)

return None

def stringReceive(Serie):

try:

return Serie.readline().strip()

except Exception as e:

print("Erro na comunicacao (stringReceive)")

print(e)

Serie.close()

def temp(txt, size):

Fonte = pygame.font.Font("DS-DIGI.ttf", size)

Texto = Fonte.render(txt, True, green)

screen.blit(Texto, [580, 443])

def press(txt, size):

Fonte = pygame.font.Font("DS-DIGI.ttf", size)

Texto = Fonte.render(txt, True, green)

screen.blit(Texto, [200, 443])

def altitude(txt, size):

Fonte = pygame.font.Font("DS-DIGI.ttf", size)

Texto = Fonte.render(txt, True, green)

screen.blit(Texto, [235, 625])

def tick(txt,size):

Fonte = pygame.font.Font("DS-DIGI.ttf", size)

Texto = Fonte.render(txt, True, green)

screen.blit(Texto, [15, 100])

s = comInit(com, baudrate)

print(s)

a = [None]\*4

imprimir\_1\_vez = True

imprimir\_cada\_30sec = True

Window()

while (s != None):

Windowhour()

now = datetime.datetime.now()

if( now.hour < 10):

b = ("0" + str(now.hour))

else:

b = str(now.hour)

if( now.minute < 10):

c = ("0" + str(now.minute))

else:

c = str(now.minute)

if( now.second < 10):

d = ("0" + str(now.second))

else:

d = str(now.second)

tick( b + ":" + c + ":" + d,70)

for indice in range(4):

a[indice] = stringReceive(s)

if (imprimir\_1\_vez == True):

if(a[0] == b' '):

temp(a[1], 60)

print(a[1])

press(a[2] , 60)

print(a[2])

altitude(a[3], 60)

print(a[3])

imprimir\_1\_vez = False

t1 = pygame.time.get\_ticks()

if(t1 - t0 > 30000):

if (imprimir\_cada\_30sec == True):

if(a[0] == b' '):

Window()

Windowhour()

temp(a[1], 60)

print(a[1])

press(a[2] , 60)

print(a[2])

altitude(a[3], 60)

print(a[3])

a[4] = None

print(a[4])

imprimir\_cada\_30sec = False

for c in a:

if c == None:

a = [None] \* 4

t0 = t1

imprimir\_cada\_30sec = True

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

pygame.display.update()

timer.tick(60)

pygame.quit()

1. **Conclusões**

Com a realização deste trabalho laboratorial conseguimos entender melhor o funcionamento dos sensores, neste caso do sensor BMP180, uma vez que este foi utilizado durante o projeto e foi implementado o código em Arduino para o correto funcionamento do mesmo.

O grupo concretizou todos os passos do enunciado o que possibilitou uma fácil compreensão do trabalho pedido. Inicialmente foi feito o código do sensor em Arduino, código que fazia os calculos da pressão , temperatura e altura , através do sensor fornecido. De seguida foi feita uma implentção do código do Display I2C para que se pudesse visualizar os valores lidos pelo sensor. Por fim, foi feita uma janela em Python utilizando a biblioteca “pygame” de forma a termos o “placar” da nossa estação metereológica caso a quisessemos afixar nalgum lado.

Resumindo, os objetivos foram todos alcançados, o que por sua vez possibilitou adquirir conhecimentos acerca de toda a nova matéria lecionada.

1. **Bibliografia**

Pais, Eng. Jorge, PowerPoint Computação Física, pp.88-106

Carvalho, Eng. Carlos, PowerPoint Computação Física, pp. 200-207

ManualBMP180, pp. 9-22