Um Altímetro de Baixo Custo

TOPUS Pesquisas Aeroespaciais

Universidade de São Paulo
Escola de Engenharia de São Carlos
Departamento de Engenharia Aeronáutica

Avenida João Dagnone, 1100
Jardim Santa Angelina, CEP 13563-120
São Carlos – SP/Brasil
Tel: (16) 3373-9430

Motivação

Visando uma maior difusão de conhecimento acerca da construção e manutenção de foguetes, o grupo TOPUS de Pesquisas Aeroespaciais produziu esse artigo e espera que o mesmo seja útil não só para outros grupos do gênero como também para qualquer pessoa interessada no assunto, buscando, assim, divulgar o foguetemodelismo no Brasil.

Introdução

O altímetro é um dispositivo destinado a medir altitudes ou alturas por meio de tomadas de pressão atmosférica. Seu uso é abrangente e inclui uma série de esportes, entre eles o alpinismo e o paraquedismo, além de estar presente na aeronáutica. Para fins de foguetemodelismo, o altímetro tem servido para demonstrar a evolução da trajetória vertical de foguetes, o que possibilita checar a exatidão do modelo desenvolvido para o projeto e suas considerações. Há também o uso para validação de missões relacionadas à altura, sendo o fator essencial para a determinação do sucesso, ou fracasso.

Objetivo

O objetivo desse artigo é expor, de forma didática, um método para construir um altímetro utilizando um sistema Arduino-Barômetro. O intuito principal dessa construção é o mesmo já supracitado para os fins de foguetemodelismo.

Itens Necessários

1 Arduino Pro Mini
1 Cabo Mini-USB
1 Conversor Serial-USB (FTDI FT232RL)
1 Sensor barométrico (BMP180)
1 Buzzer 5v
1 Módulo microSD (Catalex)
1 Cartão microSD
1 Bateria (7v -12v)

Metrologia

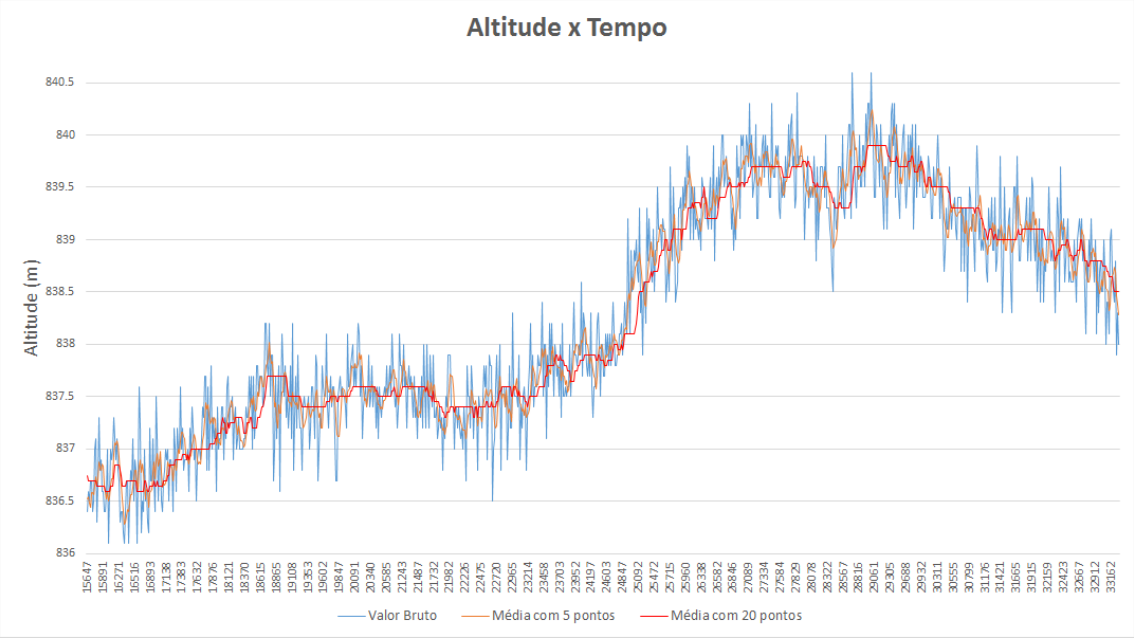
Grandezas físicas podem ser obtidas direta ou indiretamente com o uso de instrumentos. A régua é um exemplo que nos fornece distâncias de forma direta, ou seja, o valor visualmente obtido é a medida procurada. O velocímetro, por sua vez, é um exemplo que nos fornece velocidades de forma indireta, pois o valor visualmente obtido é uma conversão da velocidade angular de um ímã, que por sua vez gira em função de outro movimento: um cabo acoplado à caixa de câmbio do carro.

Dito isso, a determinação da altitude é feita indiretamente por meio de pressão local e um referencial, comumente a pressão ao nível do mar. A expressão matemática que correlaciona a pressão local com a altitude é obtida por meio do ISA (*International Standard Atmosphere*), o modelo atmosférico mais utilizado na navegação aérea.

Outro ponto muito importante sobre medidas é a incerteza de medição: não é possível a obtenção exata de grandezas através de um número finito de medições. Ao usar uma régua, por exemplo, é impossível medir qualquer comprimento com uma certeza superior à menor divisão do instrumento. Há ainda as imprecisões humanas de construção, a influência das

condições ambientais, como temperatura e umidade, entre outros fatores também responsáveis por erros na grandeza obtida.

Para sistemas eletroeletrônicos, ainda existem outras fontes de imprecisão, sendo a mais importante delas o ruído. Tal imprecisão nada mais é que a oscilação em um sinal, geralmente devido à alimentação do circuito, micro-oscilações no ambiente ou campos magnéticos próximos. Ruídos são, em geral, simétricos a uma média, o que possibilita reduzir sua intensidade. O problema surge com a existência de alguma variável dinâmica, tornando mais complexa a filtragem em questão. Ao aplicar algum tipo de média a uma medida dinâmica, há um amortecimento das variações, o que distorce o sinal real. Segue um exemplo sem utilizar média, com 5 pontos e com 20 pontos.



Nesse tutorial será utilizado um algoritmo interno do barômetro utilizando 20 pontos.

O Altímetro

Para baratear o desenvolvimento e reduzir a carga teórica necessários para a construção, opta-se por utilizar uma plataforma de prototipagem rápida muito conhecida, o Arduino, juntamente com um sensor barométrico (BMP-180) de baixo custo. Mais informações, como modelos e especificações, podem ser encontradas em www.arduino.cc ou no site do fabricante do sensor (Bosch). Em geral, as placas de prototipagem são constituídas de um microcontrolador fabricado pela Atmel, um cristal oscilador de quartzo, capacitores e resistores. Algumas placas contém um conversor, que é responsável por comunicar as interfaces Serial-Usb, e um jack de alimentação. Por limitações de espaço interno, a versão do Arduino recomendada é a Pro-Mini, oficialmente vendida pela SparkFun (www.sparkfun.com).

Um dos motivos mais importantes pela popularização da plataforma Arduino é ser OpenSource, ou seja, qualquer pessoa tem acesso aos desenhos da placa e pode reproduzi-los.

Dessa forma, existem diversas alternativas (mais acessíveis) à versão fabricada pela SparkFun. As especificações do microcontrolador presente na versão Pro Mini 328p estão na tabela:

Microcontroller	ATmega328
Operating Voltage	3.3V or 5V (depending on model)
Input Voltage	3.35 -12 V (3.3V model) or 5 - 12 V (5V model)
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
Flash Memory	32 kB (of which 0.5 kB used by bootloader)
SRAM	2 kB
EEPROM	1 kB
Clock Speed	8 MHz (3.3V model) or 16 MHz (5V model)

A versão recomendada para o altímetro é a com o microcontrolador ATmega328 5v. O preço reduzido dessa plataforma, quando comparado aos outros modelos, deve-se em parte à ausência de um conversor Serial-USB na placa, de pinos soldados e de um JACK de alimentação. Para programar a placa é necessário a aquisição de um Conversor Serial-USB. O conversor indicado é a placa da FTDI baseada no chip FT232RL, responsável por emular uma porta Serial no computador.

Outro ponto a ser citado: a comunicação entre o Arduino e o BMP-180 é feita através da interface I2C(I-dois-C). I2C é um duto de comunicação serial criado pela Phillips para interligação de periféricos. Possui apenas dois fios: SDA, o qual é o duto de dados bidirecional, e o SCL, o qual é o duto de clock. O dispositivo que inicia a transação se torna o Mestre (quem enviará o sinal de clock) e os demais dispositivos são rotulados de Escravos. Cada periférico compatível com o I2C é dotado de um endereço pré-definido. Inicialmente o Mestre manda um bit para estabelecimento como Mestre e depois o endereço do Escravo com o qual deseja comunicar. Em seguida um bit especificando operação de escrita (0) ou leitura (1). O Escravo envia um bit ACK =1 (acknowledgement) para confirmar o recebimento, então o transmissor manda um byte de dados e o receptor, portanto, envia um bit de ACK. Se necessário, o padrão se repete. Quando é escrita, o Mestre manda todos os bytes, monitora o último bit de ACK e envia um STOP. Em uma leitura, o Mestre não envia o último ACK, e sim um STOP direto. O tamanho do endereço e velocidade de transmissão podem variar de acordo com a necessidade. Geralmente o endereço tem 7 ou 10 bits e a velocidade fica em torno de 100kbit/s.

Toda essa descrição se faz necessária para entender o funcionamento mais a fundo e compreender as limitações. O algoritmo responsável por esse processo está na biblioteca Wire.h e a Adafruit_BMP085.h. O armazenamento dos dados será feito em um Cartão MicroSD através de um módulo para Arduino via interface SPI.

Como o minifoguete percorre parte da trajetória com velocidade elevada, efeitos causados pela perturbação do escoamento interferem nas medições de pressão. Em versões posteriores do Altímetro serão estudados tais efeitos e a sua devida influência nas medições.

Equações

A relação entre a pressão e a altitude pode ser explicada através da equação diferencial, e é facilmente deduzida. A biblioteca que será utilizada, fornecida pela Adafruit, faz uso da seguinte expressão para cálculo da altitude, sendo que pressão é dada em Pa.

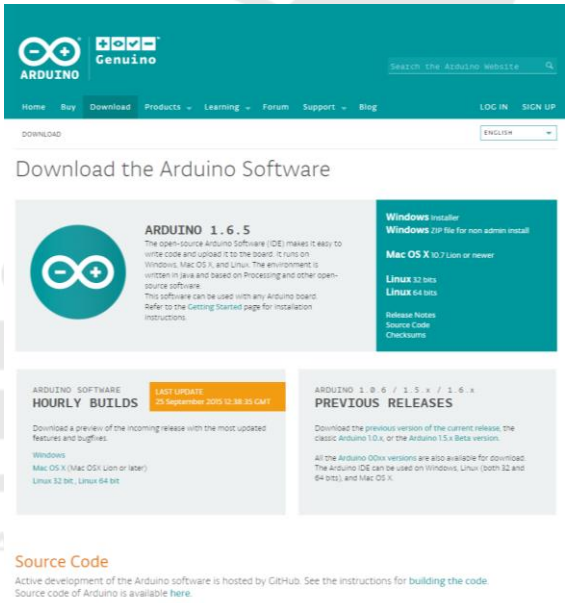
$$\text{altitude} = 44330 * (1.0 - \text{pow}(\text{pressure} / \text{sealevelPressure}, 0.1903))$$

A relação fornecida considera que a temperatura ao nível do mar é de 15ºC, sendo razoável já que o valor não influência consideravelmente na altitude. Para fins de análise, um novo algoritmo foi escrito, será avaliado e, caso comprovada a melhoria, será exposto em novas versões do tutorial.

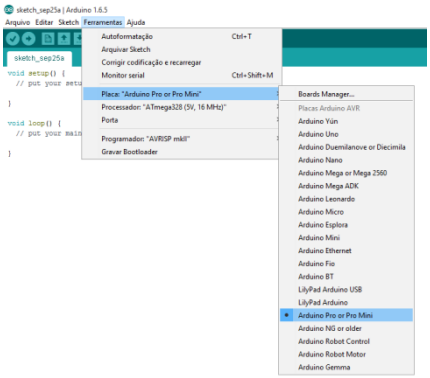
Tutorial

<u>Configurando a IDE</u>

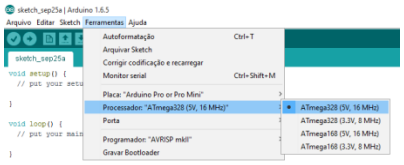
Primeiramente, deve-se selecionar o sistema em www.arduino.cc/en/Main/Software e o download se iniciará em instantes.



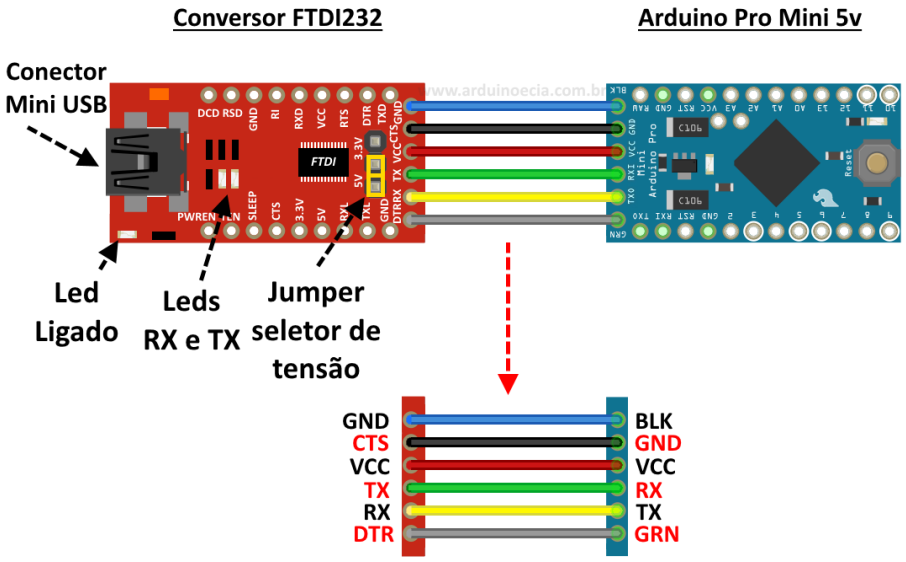
A instalação é simples, sem nenhuma configuração especial a ser feita. O próximo passo é selecionar a versão do Microcontrolador, que no caso é o ATmega328 5v.
Na IDE, deve-se clicar em Ferramentas>Placa:>"Arduino Pro or Arduino Pro Mini".



Em seguida, Ferramentas>Processador:>"ATmega328 (5v, 16Mhz)"



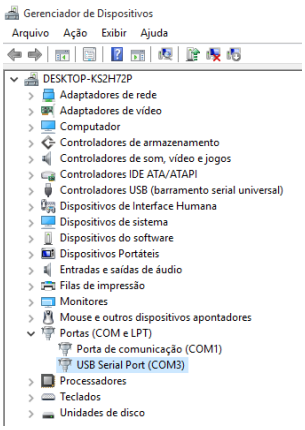
Deve-se baixar o driver do Conversor Serial-USB correspondente, no caso da placa sugerida o driver pode ser obtido pela página http://www.ftdichip.com/Drivers/VCP.htm . Selecione a versão correspondente ao sistema operacional, e clique na versão para iniciar o download. No caso do conversor indicado, o FTDI 232 a conexão é a seguinte:



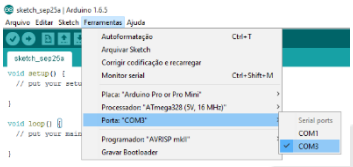
Após o download, deve-se conectar o cabo USB ao computador e ao conversor Serial-USB, abrir o arquivo .rar e selecionar o executável correspondente ao sistema operacional:

- dpinst-amd64.exe se o Windows for a versão de 64 bits
- dpinst-x86.exe se o Windows for a versão de 32 bits

A instalação não necessita de configurações especiais. Após instalado, deve-se verificar qual a porta COM (Porta Serial) que o Arduino está conectado. Basta acessar o Gerenciador de Dispositivos. Deve-se expandir a seção "Portas (COM e LPT)", e verificar qual é o número da porta "USB Serial Port".

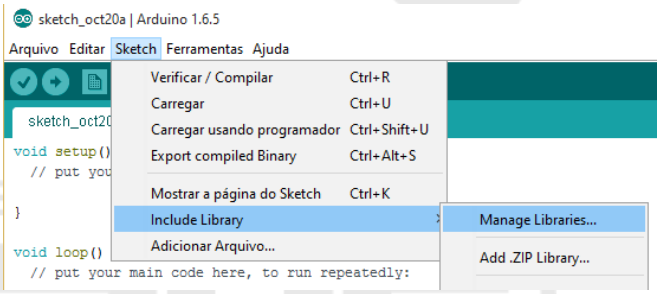


Nesse caso, a porta é a COM3. Deve-se retornar a IDE e acessar a guia Ferramentas>Porta:””>COM3

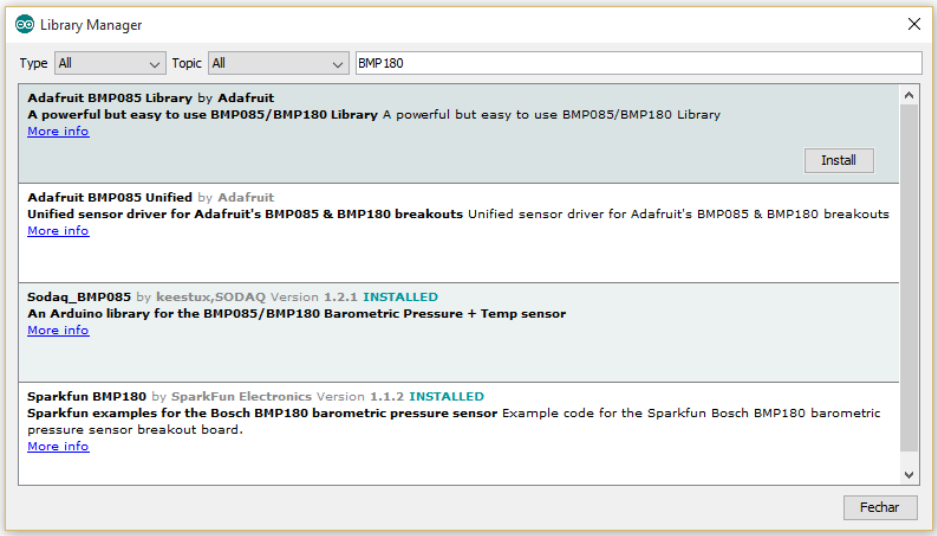


Instalando Bibliotecas

Após a configuração, deve-se instalar a biblioteca fornecida pela Adafruit para o módulo BMP180 e BMP085. Basta clicar em Sketch>Include Library>”Manage libraries...”



Na janela aberta, digite “BMP180” na caixa de pesquisa e clique no botão “Install” no item “Adafruit BMP085 Library by Adafruit”. A biblioteca será baixada e instalada automaticamente.



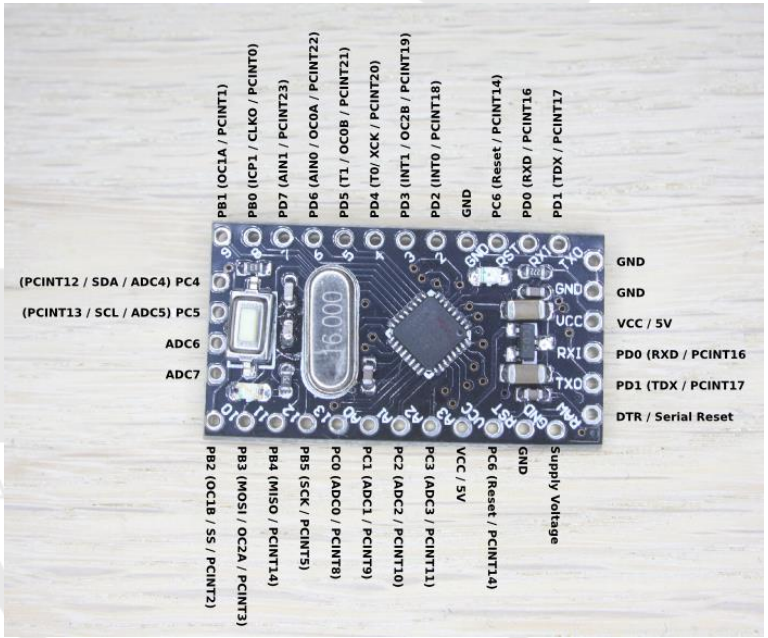
As conexões

As conexões de cada componente dependem da placa escolhida. Nesse caso, para o Arduino Pro Mini seguem os pinos correspondentes:

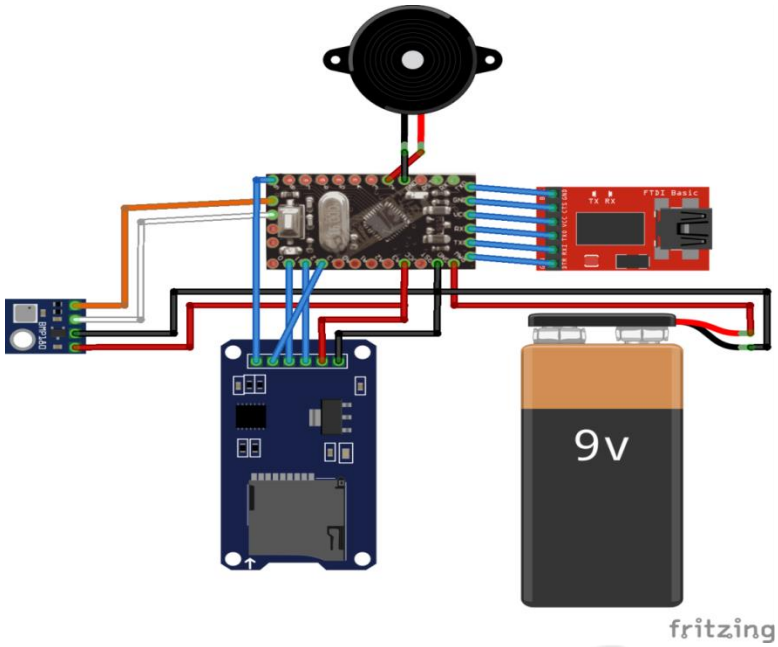
SD:
CS -> D9
SCK -> D13
MOSI -> D11
MISO -> D12
VCC -> VCC
GND -> GND

Buzzer:
Vermelho -> D2
Preto -> GND
BMP180:
SDA -> A4
SCL -> A5
GND -> GND
VCC -> VCC

Sendo os pinos do Arduino fabricados pela Baite descritos por:



O fio positivo da alimentação de 7-12v deve ser ligada no pino RAW e o fio negativo no pino GND. Segue uma ilustração de como deve ficar:



O código

Para evitar gravações desnecessárias, é preciso controlar quando os dados são gravados. Por praticidade, optou-se por colocar um contador que ativa um aviso sonoro, e em seguida inicia a coleta, finalizando após um intervalo predeterminado. A frequência de leitura obtida é em torno de 70 hertz com o oversampling configurado para 1. Oversampling é o uso de várias medidas buscando reduzir o ruído, sendo que quando mais medidas se usar mais tempo demora para obtê-las. O tempo médio para se obter o valor está descrito na tabela abaixo, juntamente com o ruído, e o consumo:

Table 3: Overview of BMP180 hardware accuracy modes, selected by driver software via the variable *oversampling_setting*

Mode	Parameter *oversampling_setting*	Internal number of samples	Conversion time pressure max. [ms]	Avg. current @ 1 sample/s typ. [µA]	RMS noise typ. [hPa]	RMS noise typ. [m]
ultra low power	0	1	4.5	3	0.06	0.5
standard	1	2	7.5	5	0.05	0.4
high resolution	2	4	13.5	7	0.04	0.3
ultra high resolution	3	8	25.5	12	0.03	0.25

Outro fator limitante na taxa de aquisição é o tempo que se leva para gravar os dados. Tal otimização não será feita nessa versão do altímetro. De forma simplificada, o algoritmo do altímetro segue os seguintes passos:

- Define a Pressão atmosférica ao nível do mar
- Define quantas medidas serão feitas para retornar a pressão
- Define os tempos relativos a execução
- Aguarda “wait” milissegundos

- Enquanto o módulo SD e o barômetro não estiverem prontos, aguarde. Do contrário, prossiga.
- Cria um novo arquivo
- Buzzer dispara
- Gravação inicia
- Gravação encerra após decorridos “tmax” milissegundos

O código utilizado é este abaixo:

```
/********************* BMP180 *******************/
#include <Wire.h>
#include <Adafruit_BMP085.h>
Adafruit_BMP085 bmp;
/********************* Configuração *************/
#define P0 101400
#define oversampling 1
#define buzzer 2
#define led 3
#define wait 10000
float tmax = 20000;
double h, t = 0, t0;
int k = 0;
char i = '1';
char file[] = "voo01.csv";
char file2[] = "config.cfg";
/********************* SD **********************/
#include <SPI.h>
#include <SD.h>
#define SS 9
File arquivo;

void setup(){
  pinMode(buzzer, OUTPUT);
  pinMode(led, OUTPUT);
  delay(wait);
  Serial.begin(115200);
  SD.begin(SS);
  while(!SD.open(file2, FILE_WRITE) || !bmp.begin(oversampling)){
      delay(1000);
      digitalWrite(led, HIGH);
      delay(1000);
      digitalWrite(led, LOW);
  }
  arquivo.close();
  while(SD.exists(file)){
    i++;
    file[4] = i;
  }
  for(int i = 0; i < 4; i++){
    tone(buzzer, 2048, 1000);
    delay(1200);
  }
  arquivo = SD.open(file, FILE_WRITE);
  t0 = millis();
  tmax = t0+tmax;
}

void loop(){
  Serial.print(tmax);
  Serial.print(t<tmax);
  if(t<tmax){
    while(k<50){
```

```
      k++;
      h = bmp.readAltitude(P0);t = millis();
      arquivo.print(h,1);arquivo.print(";");arquivo.println(t,0);
    }
    arquivo.close();
    arquivo = SD.open(file, FILE_WRITE);
    k = 0;
  }
  else{
    for(int i = 0; i < 4; i++){
      tone(buzzer, 2048, 600);
      delay(1200);
    }
  }
}
```

Os parâmetros a serem alterados são:

- P0 – pressão ao nível do mar local (pode ser encontrada em sites metereológicos)
- wait – tempo de espera até iniciar as leituras em microssegundos
- tmax – tempo de leitura em microssegundos

O tempo wait recomendado é de 300000 microssegundos (5 minutos), pois é suficiente para preparar o sistema e se afastar do local. Já tmax é recomendado que se coloque uma margem extra, evitando assim a perda de dados. Para implementar, basta copiar o código e colar na IDE. Para compilar e gravar o código na placa, basta clicar em "Carregar". Após a gravação a IDE irá indicar se a operação foi bem-sucedida.

Conclusão

Um modelo de barômetro com datalogging foi descrito nesse tutorial de forma simples, mas sem deixar a base teórica de lado. Futuras otimizações serão feitas e publicadas pelo Grupo TOPUS de Pesquisas Aeroespaciais.

Fontes

SparkFun – Imagem https://www.sparkfun.com/products/11113

Arduino e Cia – Imagem http://4.bp.blogspot.com/-OHGNmIzoF2E/VBrKgcqd6BI/AAAAAAAACsU/HT3zh4N8oC8/s1600/Conexao-FTDI-Arduino-Pro-Mini.png

BOSCH BMP180 Datasheet - https://www.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf

FTDI Chip Driver - http://www.ftdichip.com/Drivers/VCP.htm

Sites de Compra Brasileiros Sugeridos:

Arduino Pro Mini:

http://www.filipeflop.com/pd-88d41-arduino-pro-mini-atmega328p.html?ct=3d60d&p=1&s=1

Conversor SerialUSB:

http://www.filipeflop.com/pd-14690c-placa-ftdi-ft232rl-conversor-usb-serial.html

Obs: O grupo TOPUS de Pesquisas Aeroespaciais não tem nenhuma relação com a lojas indicada, apenas as sugere como opção mais barata (encontrada à época) e já utilizada pelo grupo.