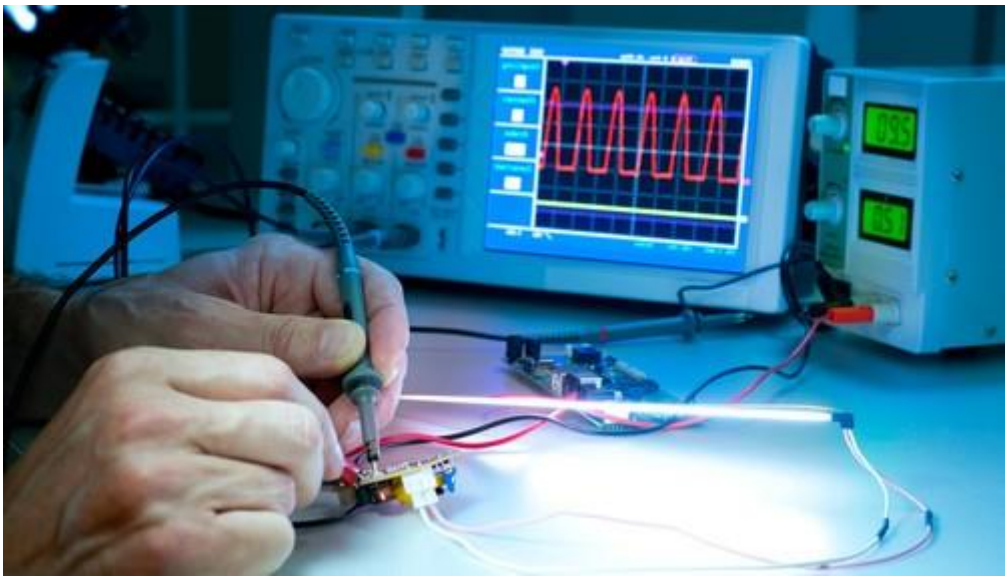
Bioinstrumentação

Plataforma de Instrumentação Médica



Autores: Luís Pereira 14868;
Daniel Mendes 14871;
Marcelo Barbosa 14163;
Ricardo Carvas 14812;

Orientador Científico: Prof. Nuno Dias

Data: 14/06/2019

Índice

Introdução

Este trabalho segue no contexto de avaliação a respeito da componente prática da UC Bioinstrumentação, e tem como objetivo implementar um circuito e respetivo algoritmo, onde será possível adquirir dados de temperatura e, posteriormente, o desenvolvimento de uma aplicação capaz de detetar picos e calcular batimentos cardíacos por minuto (BPM) e intervalos entre batimentos(IBI).

O trabalho encontra-se assim dividido em duas partes.

1ª Parte corresponde á montagem do circuito e aquisição da temperatura;

2ª Parte corresponde ao desenvolvimento de uma aplicação em Python para aquisição do batimento cardíaco através do *BITalino* em junção com o sensor PulseSensor.

Este projeto tem como principal meta a consolidação de conhecimentos lecionados na unidade curricular de Bioinstrumentação e o melhor entendimento do funcionamento dos sistemas e circuitos de deteção de sinais biomédicos.

Parte I – Leituras de Temperatura

I.1. Montagem em Ponte de Wheatstone

1

a)

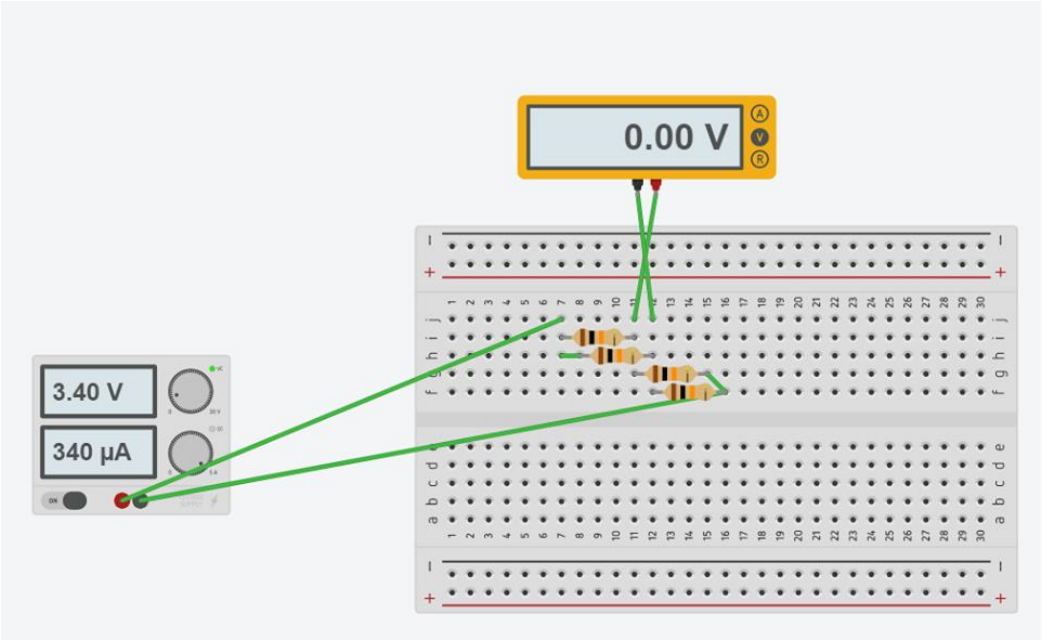


Figura 1-Simulação do circuito

V0 = 0 pois a ponte de Wheatstone está balanceada, ou seja, as resistências irão ter o mesmo valor.

b)

Figura 2-Circuito

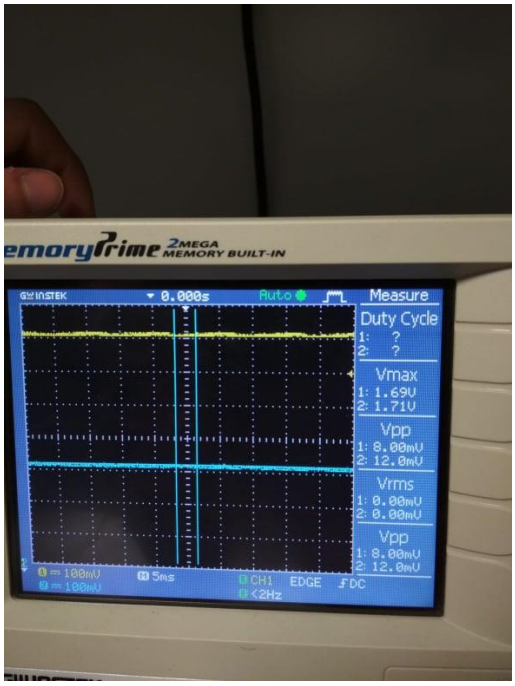


Figura 3-Osciloscópio

Nesta alínea, tivemos de montar o circuito como foi pedido, estando presente uma Ponte Wheatstone balanceada, isto é, o valor obtido irá ser aproximadamente nulo.
Comparando com o valor encontrado na simulação do circuito, afirmamos que os valores encontrados são iguais, o que seria de esperar porque V0=VA-VB=0.

c)

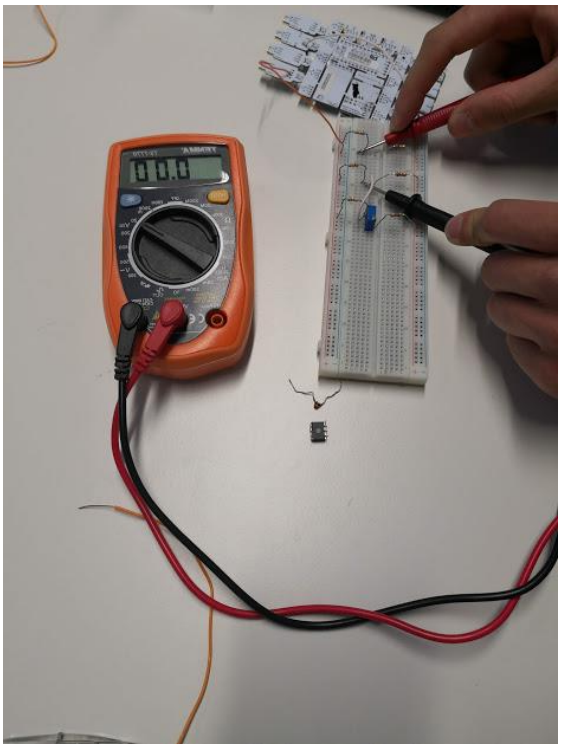


Figura 4- Confirmação do balanceamento da ponte de Wheatstone

Depois de realizar as alterações no circuito, procedemos então ao ajuste do potenciómetro de modo que a tensão V0 seja nula. Para que fosse possível realizar esta tarefa tivemos de rodar o parafuso do potenciómetro que ajusta o seu ponto medio de modo a que o valor de V0 seja nulo e confirmamos utilizando um multímetro.

d)

Valores obtidos:

VA=1,82 V

VB=1,70 V

V0=VB – VB= 0,12 V

Cálculos:

$$VA = 3{,}3 \times \frac{10 \times 10^3 + \Delta R}{20 \times 10^3 + \Delta R}$$

$$VB = 3{,}3 \times \frac{10 \times 10^3}{20 \times 10^3}$$

$$VB = 1{,}65\ V$$

$$V0 = 3{,}3 \times \frac{10 \times 10^3 + \Delta R}{20k \times 10^3 + \Delta R} - 1{,}65$$

$$0{,}12 = 3{,}3 \times \frac{10k + \Delta R}{20k + \Delta R} - 1{,}65$$

$$\Leftrightarrow 1{,}77 = 3{,}3 \times \frac{10k + \Delta R}{20k + \Delta R}$$

$$\Leftrightarrow 0{,}5364 = \frac{10k + \Delta R}{20k + \Delta R}$$

$$\Leftrightarrow 10728 + 0{,}5354\Delta R = 10k + \Delta R$$

$$\Leftrightarrow 0{,}4636\Delta R = 728$$

$$\Leftrightarrow \Delta R = 1570{,}3192$$

$$T°C \rightarrow \Delta R + R$$

$$\Delta R + R = 1570{,}3192 + 10k = 11570{,}32$$

$$T°C = 22°C$$

20,0	12487,74
21,0	11938,78
22,0	11416,96
23,0	10920,81

Figura 5-RT curve

Segundo a curva característica R(Ω) vs T(ºC), a temperatura vai ser aproximadamente 22ºC.

e)

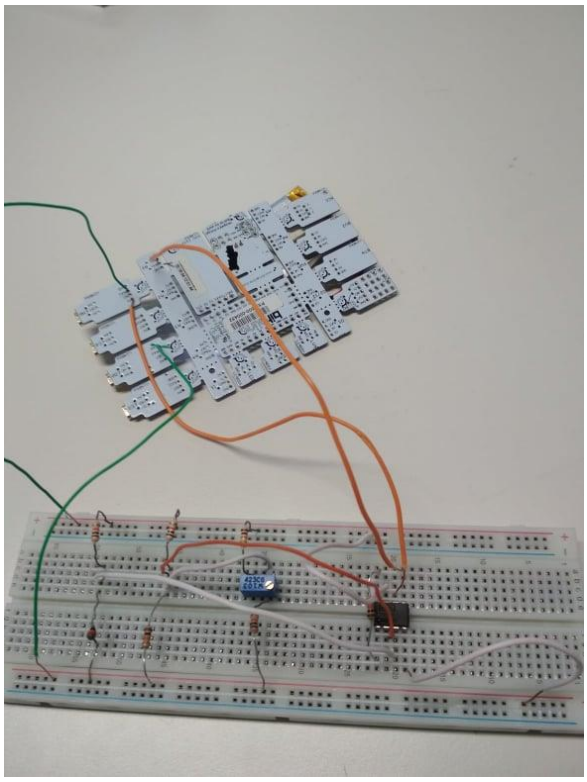


Figura 6-Circuito ligado ao BITtalino

Para calcular o ganho, usaremos a fórmula $A = 1 + \frac{50\ k\Omega}{R}$

O Vin vai ser aproximadamente igual ao Vout logo o ganho vai ser igual a 1

$$A = 1 + \frac{50k\Omega}{\infty} = 1$$

Como tivemos problemas com a leitura da temperatura com o *BITtalino*, achamos que o ganho não era suficiente, então acrescentamos ao circuito uma resistência de 1kΩ e calculamos o ganho.

$$A = 1 + \frac{50k\Omega}{5.7k\Omega} = 9.77$$

Concluindo o ganho vai ser 9.77.

f)

Valores:

B= 3932

R0=10kΩ

T= 25ºC = 298,15K

Vin = 1,65 V

$$R\infty = R0 \times e^{-B/T}$$

$$\Leftrightarrow R\infty = 10000 \times e^{-3932/298,15}$$

$$\Leftrightarrow R\infty = 0,0187$$

$$V0 = Vin(VA - VB)$$

$$VA = Vin \times \frac{R3}{R3 + R1}$$

$$\Leftrightarrow VA = Vin \times \frac{R3}{10000 + R3}$$

$$VB = Vin \times \frac{R4}{R2 - R4}$$

$$\Leftrightarrow VB = Vin \times \frac{10000}{20000}$$

$$\Leftrightarrow VB = Vin \times \frac{1}{2}$$

$$V0 = 1,65 \times \left(\frac{R3}{10000 + R3}\right) - \frac{1}{2}$$

$$\Leftrightarrow V0 = 1,65 \times \left(\frac{2R3 - R3 - 10000}{20000 + 2R3}\right)$$

$$\Leftrightarrow V0 = 1,65 \times \frac{R3 - 10000}{20000 + 2R3}$$

$$\Leftrightarrow \frac{V0}{1,65} = \frac{R3 - 10000}{20000 + 2R3}$$

$$\Leftrightarrow \frac{V0}{1,65} \times (20000 + 2R3) = R3 - 10000$$

$$\Leftrightarrow \frac{V0 \times 20000 + 2R3 \times V0}{1,65} = R3 - 10000$$

$$\Leftrightarrow \frac{V0 \times 20000}{1,65} + \frac{2R3 \times V0}{1,65} = R3 - 10000$$

$$\Leftrightarrow \frac{2R3 \times V0}{1,65} - R3 = -10000 - \frac{V0 \times 20000}{1,65}$$

$$\Leftrightarrow R3 \times \frac{2V0 - 1,64}{1,65} = \frac{-16400 - 20000 \times V0}{1,65}$$

$$\Leftrightarrow R3 = \frac{\frac{-16500 - 20000 \times V0}{1,65}}{\frac{2V0 - 1,65}{1,65}}$$

$$\Leftrightarrow R3 = \frac{-16500 - 20000 \times V0}{2V0 - 1,65}$$

Expressão para calcular a tensão de saída do circuito (V0) em função da temperatura (T):

$$T = \frac{B}{\ln(R3/R\infty)}$$

$$T = \frac{3932}{\ln(\frac{-16400 - 20000 \times V0}{2V0 - 1{,}64}/0{,}0187)}$$

g)

Expressão v0 em função de temperatura

$$T = \frac{3932}{\ln(\frac{-16500 - 20000 \times V0}{2V0 - 1{,}65}/0{,}0187)}$$

$$\Leftrightarrow \ln(\frac{-16500 - 20000 \times V0}{2V0 - 1{,}65}/0{,}0187) = \frac{3932}{T}$$

$$\Leftrightarrow \frac{-16500 - 20000 \times V0}{2V0 - 1{,}65} = 0{,}0187 \times e^{3932/T}$$

$$\Leftrightarrow -16500 - 20000 \times V0 = \left(0{,}0374 \times e^{\frac{3932}{T}}\right) \times V0 - 0{,}0309 \times e^{3932/T}$$

$$\Leftrightarrow \left(-20000 - 0{,}0374 \times e^{\frac{3932}{T}}\right) \times V0 = 16500 - 0{,}0307 \times e^{3932/T}$$

$$\Leftrightarrow V0 = \frac{16500 - 0{,}0309 \times e^{3932/T}}{-20000 - 0{,}0374 \times e^{3932/T}}$$

Fórmula para converter os valores do *BITalino* para volts

$$Volts = \frac{(\frac{Raw}{1023} - 0{,}5) \times 3{,}3}{Ganho}$$

$$Volts = \frac{(\frac{Raw}{1023} - 0{,}5) \times 3{,}3}{9{,}77}$$

I.2. Aplicação de Leitura de Temperatura

Após a montagem e teste do circuito e tendo os cálculos necessários realizados, passamos ao desenvolvimento de uma aplicação capaz de:

- Ler os valores do BITalino,
- Aplicar funções de transferência para temperatura aos valores lidos
- Representar temperatura em gráfico em tempo real

Para o desenvolvimento desta aplicação foi utilizanda a linguagem *Python* em acréscimo com o *ServerBit* e o BioinstrumentationClient.py disponibilizado pelo docente.

O *BioinstrumentationClient.py* foi utilizado para guardar valores em ficheiro *txt* à medida com que estes eram lidos

```
def run(self):
    while True:
        msg = yield self.ws.read_message()
        if msg is None:
            print ("connection closed")
            self.ws = None
            break
        else:
            global ts
            # print(str(msg))
            try:
                msgDict = js.loads(msg)
            except:
                print("Erro de json")
            else:

                print(msgDict["A1"][:])

                data_value = msgDict["A1"]
                data = np.array(data_value)

                data.tofile(file, sep='\n', format='%s')
                file.write("\n")
```

Figura 7-BioinstrumentationClient.py

Assim sendo, foi desenvolvido um novo script "*RealTimeTempMesure.py*" capaz de abrir e ler o ficheiro *txt* enquanto ele é gerado pela leitura do *BioinstrumentationClient.py.*

À medida que os valores vão sendo lidos, são transferidos para temperatura e dispostos em gráfico. O script também gera um novo ficheiro *txt* com os valores temperatura dispostos no gráfico.

Os scripts *BioinstrumentationClient.py e RealTimeTempMesure.py* devem ser corridos em simultâneo

```
for val in valores:
    x=x+1

    #Valores Bitalino para volt
    v0 = (((float(val) / float(1023)) - 0.5) * 3.3) / 10.09

    #Valores termistor
    R3 = (float(-16500) - float(20000 * v0)) / (float(2 * v0) - 1.65)

    #Calculo Temperatura
    t = (b / (np.log(R3 / rinf))) - 273.15
    print(t)

    t.tofile(file, sep='\n', format='%f')
    file.write("\n")

    xs.append(float(x))
    ys.append(float(t))


ax1.clear()
ax1.set_ylim(15, 40)
ax1.plot(xs, ys)
```

Figura 8- Conversão dos valores para temperatura e disposição em gráfico

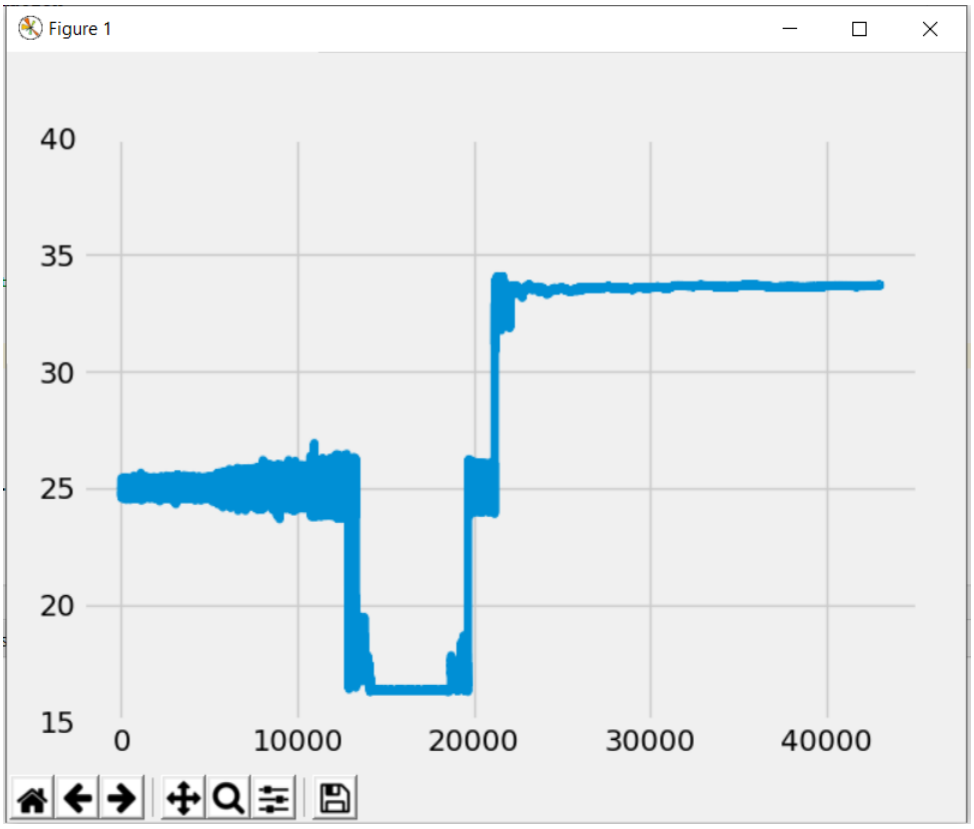


Figura 9-Gráfico dos valores lidos

Devido a algum problema com o amplificador do circuito, não foi possível ler as temperaturas do termístor, contudo conseguimos observar no gráfico acima as 3 fases onde era possível manipular os valores lidos pelo BIT*talino* através da ligação A1.

- A1 no v0 do amplificador → Tº ≈25ºC;
- A1 no AVCC → Tº ≈15ºC;
- A1 na GND, → Tº ≈33ºC

Parte II – Batimento Cardíaco

II.1. Aplicação de Leitura de Pulso e Deteção de Picos

Nesta parte 2, o objetivo era realizar uma aplicação capaz de recolher os valores lidos pelo *Bitalino* ligado ao pulse sensor e dispor esses valores em gráfico.

Após este passo, foi também proposto a realização de um algoritmo capaz de detetar picos e calcular os bpm's.

```
import matplotlib.pyplot as plt

graph_data = open("ondaPulso.txt", "r")
values = graph_data.read().split(', ')
del values[-1]
pulse = map(float, values)
time = range(0, len(pulse))

fig = plt.figure()
ax = fig.add_subplot(111)

plt.plot(time, pulse)

up = None
down = None
npeaks = 0
for x in range(0, len(values)):
    if x >= len(values)-1:
        break
    else:
        if float(values[x]) > 530:
            up = True

        if float(values[x]) < 530:
            down = True
        if up == down:

            if(values[x]>values[x-1]):
                plt.plot(x, float(values[x]), '-ko')
                npeaks = npeaks + 1
            up = None
            down = None

ax.text(500, 450, 'Picos Encontrados:{}'.format(npeaks) , size=10, ha='center', va='center')
print("Picos:", npeaks)
plt.show(block=True)
```

Figura 10-PeakCounter.py

Este algoritmo, serve-se do desenho do sinal, pelo que não é fiável em todo o tipo de amostrar, pois utiliza como referência valores específicos (530) para verificar se existe uma subida ou descida do sinal. Quando encontramos um momento em que se deu uma subida e uma descida, encontramos um pico e é colocado um ponto no gráfico.

No fim é também disposto o número de picos encontrados na amostra.

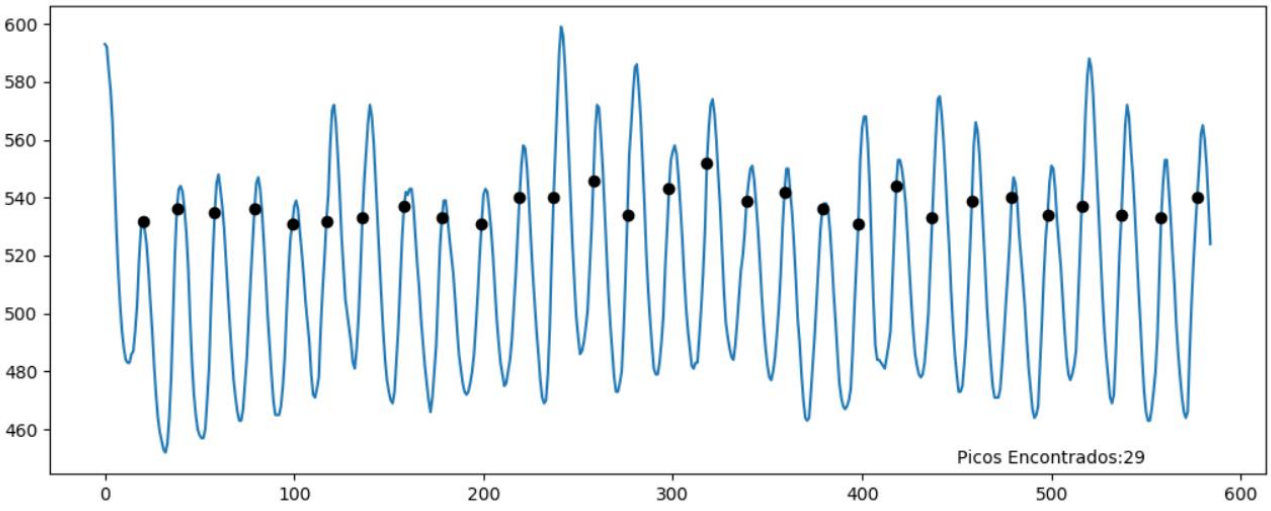


Figura 11-Picos encontrados

Conclusão

Ao longo dos quase dois meses da realização deste trabalho foram postos em prático diversos conhecimentos, alguns desses já algo esquecidos, outros que foram adquiridos no decorrer do desenvolvimento do mesmo, nomeadamente eletrónica e programação em python, respetivamente.

Consideramos este trabalho como fundamental para o aprofundamento do conteúdo programático da UC, bem como o evoluir das capacidades autodidatas dos alunos e a persistência para se chegar a um bom resultado.

As dificuldades foram várias, desde problemas com o circuito e o pulse sensor aos problemas com os ambientes de desenvolvimento software, acrescentando ainda a carga extra da utilização duma linguagem "desconhecida".

Apesar de tudo isso, achamos que os objetivos foram conseguidos com sucesso e certamente que os conhecimentos adquiridos com a realização deste trabalho serão com certeza úteis num futuro próximo.

Fontes Bibliográficas:

Documentação Bitalino:
http://bitalino.com/index.php/board-kit-bt
http://bitalino.com/datasheets/REVOLUTION_BITalino_Board_Kit_Datasheet.pdf
http://bitalino.com/index.php/software
http://bitalino.com/downloads/int-releases/OpenSignals_(r)evolution_Manual.pdf
http://bitalino.com/index.php/development/apis
http://bitalino.com/downloads/bitalino_manual/manual.html

Documentação API Python Bitalino:
http://bitalino.com/pyAPI/

Datasheet termístor NTC:
http://cdn.sparkfun.com/datasheets/Sensors/Temp/ntcle100.pdf

Datasheet INA128:
http://www.ti.com/product/INA128/datasheet/abstract#SBOS0515844

Datasheet Pulsesensor:
https://pulsesensor.com /
https://pulsesensor.com/pages/pulsesensor-playground-toolbox

WebSockets:
https://pt.wikipedia.org/wiki/WebSocket

Documentação Numpy
https://docs.scipy.org/doc/numpy/reference/generated/numpy.array.html

Documentação Matplolib
https://matplotlib.org/