North South University

Department of Electrical & Computer Engineering

Project Report

Course name: Operating Systems Design
Course code: CSE323
Term: Summer’19
Instructor: Dr. Saeed Mahmud Ullah (SMU1)
Date: 29/08/19

Group Information

Group Name: The Binary Bombers
Group Members:

- Md. Masudur Rahman (1631189042)
- Saraf Sumaita Hasan (1631258042)
- Samina Kamal Upama (1620562042)

Heart-pulse monitoring system with Real-time streaming and visualization with Raspberry Pi 3

Objective: The aim of this project is to monitor the heart pulse and then to synchronize it in real-time in a web platform which eventually will increase the flexibility and portability.

Introduction: This project is a simple heart pulse monitoring system with the help of pulse sensors and raspberry pi 3b+, and this combination is done with the help of an analog to digital converter. Then it visually represents the taken data from the user in real-time as well as sync it in a web platform – "Initial State".

Equipment:

- **Hardware:**
  - Raspberry pi 3b+
  - Pulse Sensor
  - Jumper wires
  - Ethernet Cable (for initial setup)
  - Analog to Digital converter (ADS 1015)

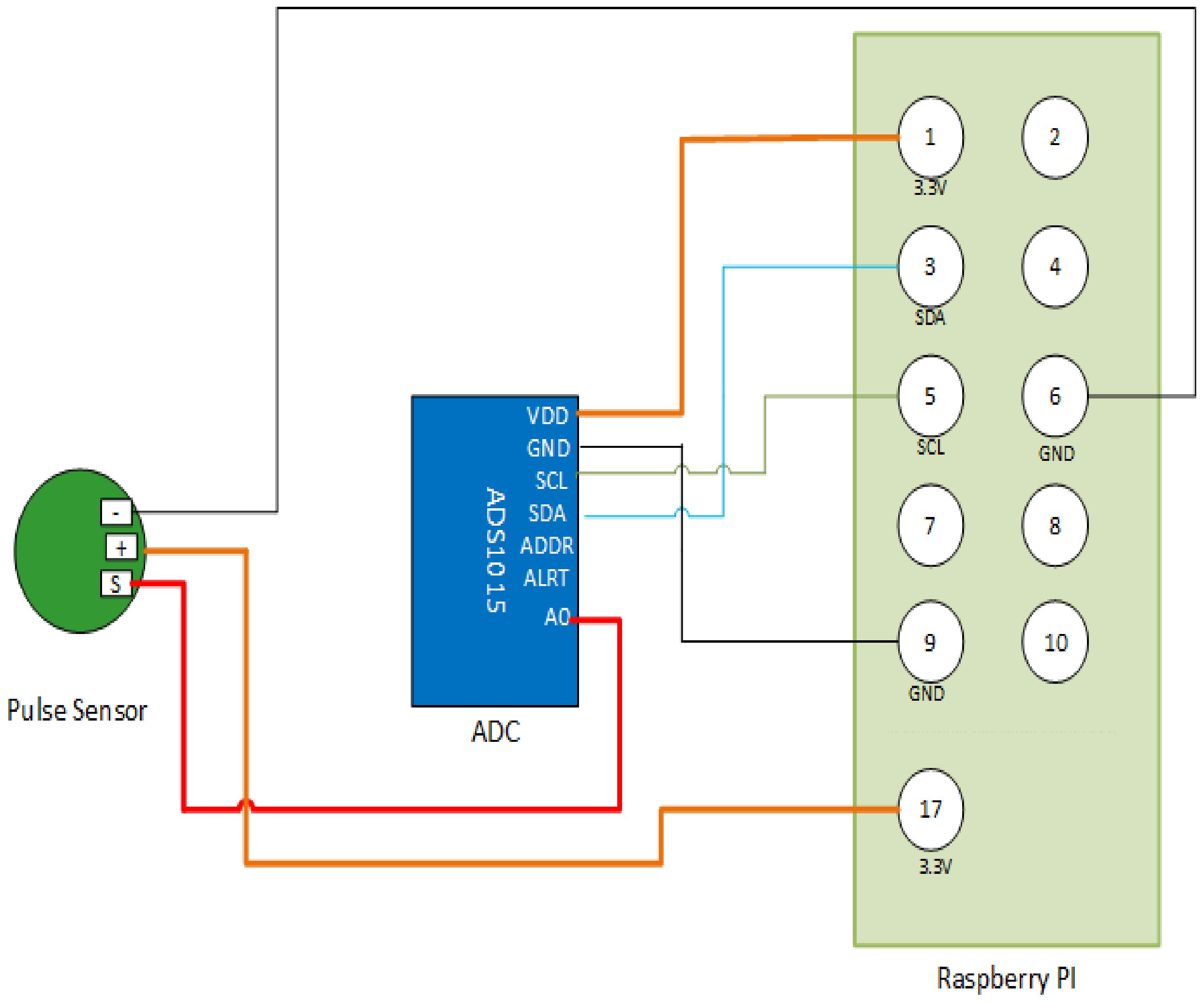
- **Software:**
  - Raspbian OS
  - Etcher
  - Pycharm
  - Putty
  - VNC Viewer
  - Initial State's API

Working Algorithm:

1. Initial setup of Raspbian Pi OS
2. Connections of the pulse sensor and Raspberry pi have to done perfectly using the Analog to Digital converter which will connect these two platforms
3. Testing of the pulse sensor and analog to digital converter by connecting according to the working diagram mentioned below
4. Then, heartbeat.py script from the Raspberry Pi has to be executed
5. Tip of the finger has to be placed on the pulse sensor
6. Step 4 will show the pulse in the Raspberry Pi's terminal

7. In the background, the data collected from step 6, will be uploaded in real-time Initial State’s web interface
8. The script of step 3 will use the bucket key and the access key to connect our code to the Initial State’s API
9. By using step 6 and 7, the data collected from step 3 will be shown on the Initial State’s web interface and also will be saved in their database for the future reference

Working diagram:



Code:

```
import time
from ISStreamer.Streamer import Streamer
import Adafruit_ADS1x15

if __name__ == '__main__':

    streamer =
    Streamer(bucket_name="test",bucket_key="Q79MEVU343FN",access_key="ist_inDy3o8P3
    AZh-fdra8P0TG04IaXR_-yo")

    adc = Adafruit_ADS1x15.ADS1015()
    GAIN = 2/3
    curState = 0
    thresh = 525
    P = 512
    T = 512
    stateChanged = 0
    sampleCounter = 0
    lastBeatTime = 0
    firstBeat = True
    secondBeat = False
    Pulse = False
    IBI = 600
    rate = [0]*10
    amp = 100
    z = 0

    lastTime = int(time.time()*1000)


    while True:

        Signal = adc.read_adc(0, gain=GAIN)
        curTime = int(time.time()*1000)

        sampleCounter += curTime - lastTime;
        lastTime = curTime
        N = sampleCounter - lastBeatTime;


        if Signal < thresh and N > (IBI/5.0)*3.0 :
            if Signal < T :
              T = Signal;
```

```
if Signal > thresh and  Signal > P:
    P = Signal;

if N > 250 :
    if  (Signal > thresh) and  (Pulse == False) and  (N > (IBI/5.0)*3.0)  :
      Pulse = True;                               # set the Pulse flag when we think there is a pulse
      IBI = sampleCounter - lastBeatTime;
      lastBeatTime = sampleCounter;

      if secondBeat :                                   # if this is the second beat, if secondBeat == TRUE
        secondBeat = False;                         # clear secondBeat flag
        for i in range(0,10):                # seed the running total to get a realisitic BPM at startup
          rate[i] = IBI;

      if firstBeat :                          # if it's the first time we found a beat, if firstBeat == TRUE
        firstBeat = False;                  # clear firstBeat flag
        secondBeat = True;              # set the second beat flag
        continue



      runningTotal = 0;

      for i in range(0,9):
        rate[i] = rate[i+1];
        runningTotal += rate[i];

      rate[9] = IBI;
      runningTotal += rate[9];
      runningTotal /= 10;
      BPM = 60000/runningTotal;
      streamer.log("Heart-Beat Rate",BPM)
      print ('BPM: {}'.format(BPM))

if Signal < thresh and Pulse == True :
    Pulse = False;
    amp = P - T;
    thresh = amp/2 + T;
    P = thresh;
    T = thresh;

if N > 2500 :
    thresh = 512;
    P = 512;
    T = 512;
    lastBeatTime = sampleCounter;
```

```
            firstBeat = True;
            secondBeat = False;
            print (z)

        time.sleep(0.005)

    streamer.close()
```

Result and discussion: The main target of this project was to collect the heart rate in bpm unit and to visualize and sync it by using a web interface in real-time. This project is able to include all of these. The objective was to build a cheap way of monitoring the heart rate and to sync it for further reference, which is also wholly achieved. But still, for the future and desired purpose, there are rooms for improvement. If we can implement it more cheaply, improve the accuracy and also if we can implement a faster way for data uploading, then this project can be a medium for diagnosing the patients' of the remote area, as the doctor can see the real-time report by using his own mobile or computer. As well as this project can be used for collecting data from the mass population for the data science purpose.

Conclusion: This project is built by taking portability and flexibility issue in mind so that it can be used on the go. The hardware setup is minimal as well as the system automatically connects to the nearest network for the data transfer, which eventually compliments the portability and flexibility.

Reference:

- https://www.raspberrypi.org/documentation/
- https://microcontrollerslab.com/analog-to-digital-adc-converter-working/
- https://www.te.com/usa-en/products/sensors/pressure-sensors.html
- https://initialstateeventsapi.docs.apiary.io/
- https://www.initialstate.com/python/
- https://io.adafruit.com/api/docs/