SOFTWARE GUIDE

INTRACRANIAL PRESSURE WAVEFORM GENERATOR

April 16, 2019

Senior Design Team 3
Wright State University
Department of Biomedical, Industrial, and Human Factors Engineering

Contents

List of Figures

Listings

RASPBERRY PI SETUP

1. Hardware

Raspberry Pi 3
Pi Cobbler
Adafruit Digital Analog Converter (DAC)
32 GB or more microSD Card
Keyboard
Computer Mouse
Wireless HDMI

2. Software Installation

NOOBS Install

We installed NOOBS as the operating system installer. A microSD card with at least 32 GB of storage is required. The set up was as follows:
1. Insert microSD into computer
2. Go to `https://www.raspberrypi.org/downloads/noobs/` and download the NOOBS zip file
3. Extract the zip flie onto the microSD
4. Eject the microSD and insert into the Raspberry Pi 5. Power the Raspberry Pi on. The first screen that will appear select Raspbian and install
6. The Pi will now reboot and after it has power back on the Raspberry Pi is now ready for use

Set up: `https://www.youtube.com/watch?v=wvxCNQ5AYPg`

3. Login Credentials

The username is "pi" – all lowercase
The password is "ICP" – all capitalized letters

4. Package Installation

The process for installing any python packages should be done as follows from inside the terminal:

1. Run "sudo apt update" through the terminal before installing new software packages. This will ensure that all packages are the lastest version and avoid any versioning issues with newer installed packages.
2. Type "sudo pip3 install xxxxxx" where xxxxxx is the name of the package with which you wish to install.
3. This will not include all necessary packages so if one of them doesn't work using "sudo pip3 install" then it is acceptable to use "sudo apt-get install python3-xxxxxx".

Viewing installed packages

To check if a package is installed type into the terminal "dpkg -l | grep <keyword>", where <keyword> is (part of) the name of the package you are interested in finding. For example, searching "dpkg -l | grep nvidia" will list all installed packages with "nvidia" in the name or description. This method will not work for meta-packages or repositories, though it will work for most cases. You can check where a package is installed by typing "sudo dpkg -S packagename".

The packages that are required for our system are as follows:

1. Adafruit MCP4275
Used for the DAC chip. Install was set up using:
`https://learn.adafruit.com/mcp4725-12-bit-dac-with-raspberry-pi`

2. Guizero
Used for the creation of the Graphical User Interface (GUI).

3. ScipPy
contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks.

4. Numpy
Used for multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions.

5.I2C
A bus that allows a connection between the pi and wiring through the GPIO pins. Setup was

done through:
`https://learn.sparkfun.com/tutorials/raspberry-pi-spi-and-i2c-tutorial/all`

6. GPIO
These are the pins that the connects the code to the DAC chip. This connection is done through the Pi Cobbler. This software allows you to pick the pin the signal will come out of. Setup for this was done through:
`https://www.raspberrypi-spy.co.uk/2012/05/install-rpi-gpio-python-library/`

TROUBLESHOOTING

White Bar Across Left Half of Screen

If a white bar appears across the left side of the screen and cannot click on the menu then follow these steps:
Control + Alt + F1
username: pi, password: ICP
sudo rm -r /.config/lxpanel
startx

HDMI Will Not Display

The Raspberry Pi must first be plugged in before the HDMI can be plugged in.

Code Will Not Execute Because Pi Cannot Discover I2C Bus Location

Chances are that the DAC chip has fried and needs to be replaced. If this is not the case, check to see which I2C location the Pi is trying to output the code by typing into the terminal "i2cdetect -y 1".

Pi Will Not Turn On

Check to see what color is being displayed by the LED. If only red is showing, the SD card is either inserted incorrectly or corrupted. It should have a red LED with a green LED that intermittently blinks.

Board Goes Off Intermittently

This is due to the power. Make sure that the Pi is plugged into a power supply that outputs 5V and 2.5A.

DIGITAL ANALOG CONVERTING SOFTWARE

The MCP4725 is a 12-bit DAC chip that converts the digital signal into an analog output to be properly displayed on the ProPAQ patient monitor. Determining what output voltage is given is. The voltage that is output is determined by the following equation:

$$V_{out} = \frac{(V_{cc} * Bit \quad Value)}{4096}$$

The V_{cc} is the output voltage from the Raspberry Pi. The system has a V_{cc} of 5V. There are two sources from the Raspberry Pi, 3.3V and 5V. The 5V source output higher resolution waveform and that is why that source was chosen.

FREQUENCY CALCULATION

In order for the ICP waveform to have the same time synchronization as the EKG waveform, they must have the same frequency. This is calculated using the following equation:

$$Length \quad of \quad Waveform = e^{\frac{-(BPM+193.74)}{54.91}}$$

This equation can be seen in each of the waveform model files (ex. modelone.py).

COMPUTER APPLICATION

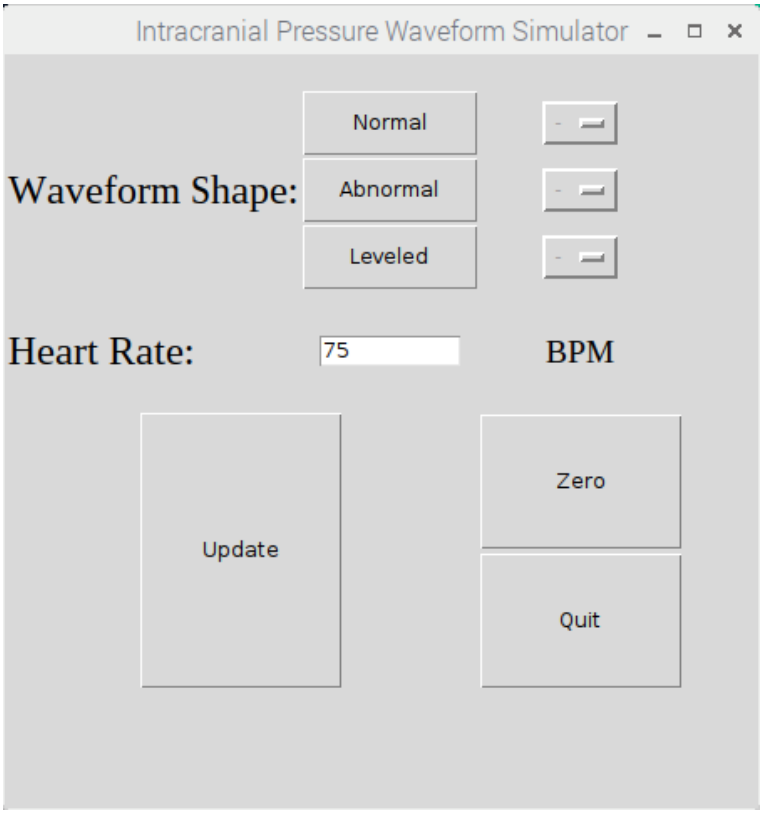

Figure 1: GUI

Listing 1: Gui Code

```
from guizero import Combo, App, Box, Text, Slider, TextBox, ButtonGroup, PushButton,
    CheckBox, MenuBar
import subprocess
import signal
import os
import sys

#Pulled in the necessary tools to create the GUI

class MyGUI:
    def __init__(self, master):
        self.master = master
       #Creates the application window

        self.space2 = Text(master, text=" ", grid=[0,0], width="fill")


        self.Shapemessage1 = Text(master, size=18, text="Waveform Shape:", grid
    =[0,2], align="left", font="Times")

        self.Waveform1 = PushButton(master, text="Normal", grid=[1,1], width = "10",
    command = self.show_choices1)
        self.Waveform2 = PushButton(master, text="Abnormal", grid=[1,2], width="10",
    command = self.show_choices2)
        self.Waveform3 = PushButton(master, text="Leveled", grid=[1,3], width="10",
    command = self.show_choices3)

        #self.Waveform1.toggle()
        #self.Waveform2.toggle()
        #self.Waveform3.toggle()

        self.combo1 = Combo(master, options=["-","10","11","12","13","14","15","16","
    17"], grid=[2,1], selected = "-")
        self.combo2 = Combo(master, options=["-","21","22","23","24","25","26"], grid
    =[2,2])
        self.combo3 = Combo(master, options=["-","0"], grid=[2,3])

        self.combo1.disable()
        self.combo2.disable()
        self.combo3.disable()
       #Creates the three waveform buttons as well as the selection for ICP value
```

```
35          #When the waveform selected the user is then able to select an ICP value
36
37          self.space2 = Text(master, text=" ", grid=[0,6])
38          #Creates space between Waveform Shape and Heart Rate
39
40          self.HRmessage = Text(master, size=18, text="Heart Rate:", grid=[0,10], align
    ="left", font="Times")
41          #Creates the HR subtitle
42
43          self.rate = TextBox(master, grid=[1,10])
44          self.HRunit = Text(master, size=15, text="BPM", grid=[2,10], font="Times")
45          #Creates the HR textbox to allow for typed input
46
47          self.space3 = Text(master, text=" ", grid=[0,11], font="Times")
48          #Creates space between Heart Rate and the Update Button
49
50          self.button = PushButton(master, command=self.update_value, width="12",
    height="10", text="Update", grid=[0,13,2,2], align="bottom")
51          #Creates an Update Button that updates the inputed information and exports
    the user-inputs to the next part of the program
52          self.when_clicked = self.update_value
53          #self.when_mouse_enters = self.update_value
54          #Creates the update button, no new code will be executed until update button
    is pressed
55
56          self.button = PushButton(master, command=self.calibration, width="12", height
    ="4", text="Zero", grid=[2,13], align="bottom")
57          #Creates zero button which is used to calibrate
58
59          self.button = PushButton(master, command=self.quit, width="12", height="4",
    text="Quit", grid=[2,14], align="bottom")
60          #Creates the quit button which will exit out of app and stop the process from
     running
61
62          self.pid = -1 #Initialize process id
63
64
65      #Launches the function for modelone , Normal ICP, in a seperate python process
    and obtains the process ID
66      def modelone(self):
67          self.done()
68
```

```
69      #Launches the function for modeltwo , Leveled ICP, in a seperate python process
        and obtains the process ID
70      def modeltwo(self):
71          self.done()
72
73      #Launches the function for modelthree, Abnormal ICP, in a seperate python process
         and obtains the process ID
74      def modelthree(self):
75          self.done()
76
77      #Launches the calibration
78      def calibration(self):
79          self.done()
80          self.process = subprocess.Popen('python calibration.py', shell=True,
        preexec_fn=os.setsid)
81          self.pid = self.process.pid
82
83      #Used to show the ICP value choices for the normal waveform
84      def show_choices1(self):
85          self.done()
86          if self.Waveform1.value == 0:
87              self.combo1.enable()
88              self.combo2.disable()
89              self.combo3.disable()
90          if self.Waveform1.value == 1:
91              self.combo1.disable()
92              self.combo1.value = 0
93
94      #Used to show the ICP value choices for the abnormal waveform
95      def show_choices2(self):
96          self.done()
97          if self.Waveform2.value == 0:
98              self.combo2.enable()
99              self.combo1.disable()
100             self.combo3.disable()
101         if self.Waveform2.value == 1:
102             self.combo2.disable()
103             self.combo2.value = 0
104
105     #Used to show the ICP value choices for the leveled waveform
106     def show_choices3(self):
107         self.done()
```

```
108         if self.Waveform3.value == 0:
109             self.combo3.enable()
110             self.combo1.disable()
111             self.combo2.disable()
112         if self.Waveform3.value == 1:
113             self.combo3.disable()
114             self.combo3.value = 0
115 
116     #Command for updating the heart rate value
117     def update_value(self):
118         self.done()
119         if self.rate.value == "":
120             self.rate.value=65
121         if self.combo1.value == "10":
122             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
123             self.pid = self.process.pid
124         if self.combo1.value == "11":
125             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
126             self.pid = self.process.pid
127         if self.combo1.value == "12":
128             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
129             self.pid = self.process.pid
130         if self.combo1.value == "13":
131             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
132             self.pid = self.process.pid
133         if self.combo1.value == "14":
134             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
135             self.pid = self.process.pid
136         if self.combo1.value == "15":
137             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
138             self.pid = self.process.pid
139         if self.combo1.value == "16":
140             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
141             self.pid = self.process.pid
142         if self.combo1.value == "17":
```

```
143             self.process = subprocess.Popen('python modelone.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
144             self.pid = self.process.pid
145
146
147
148
149
150         if self.combo2.value == "21":
151             self.process = subprocess.Popen('python modelthree.py {}'.format(self.
    rate.value), shell=True, preexec_fn=os.setsid)
152             self.pid = self.process.pid
153         if self.combo2.value == "22":
154             self.process = subprocess.Popen('python modelthree.py {}'.format(self.
    rate.value), shell=True, preexec_fn=os.setsid)
155             self.pid = self.process.pid
156         if self.combo2.value == "23":
157             self.process = subprocess.Popen('python modelthree.py {}'.format(self.
    rate.value), shell=True, preexec_fn=os.setsid)
158             self.pid = self.process.pid
159         if self.combo2.value == "24":
160             self.process = subprocess.Popen('python modelthree.py {}'.format(self.
    rate.value), shell=True, preexec_fn=os.setsid)
161             self.pid = self.process.pid
162         if self.combo2.value == "25":
163             self.process = subprocess.Popen('python modelthree.py {}'.format(self.
    rate.value), shell=True, preexec_fn=os.setsid)
164             self.pid = self.process.pid
165         if self.combo2.value == "26":
166             self.process = subprocess.Popen('python modelthree.py {}'.format(self.
    rate.value), shell=True, preexec_fn=os.setsid)
167             self.pid = self.process.pid
168
169
170         if self.combo3.value == "0":
171             self.process = subprocess.Popen('python modeltwo.py {}'.format(self.rate.
    value), shell=True, preexec_fn=os.setsid)
172             self.pid = self.process.pid
173
174
175         self.combo1.disable()
176         self.combo2.disable()
```

```
177         self.combo3.disable()
178
179
180         self.combo1.value = "-"
181         self.combo2.value = "-"
182         self.combo3.value= "-"
183
184
185     #Kills the process that is currently running
186     def done(self):
187         if self.pid > 0:
188             try:
189                 os.killpg(os.getpgid(self.pid), signal.SIGTERM)
190                 self.pid = -1
191             except ProcessLookupError:
192                 pass
193     def quit(self):
194         self.done()
195         self.master.destroy()
196 root = App(title="Intracranial Pressure Waveform Simulator", width=450, height=450,
    layout="grid")
197 #Pulls up the GUI
198 my_gui = MyGUI(root)
199 root.display()
200 #Creates then application window
```

WAVEFORM CODES

The waveform codes each rely on a different MATLAB file which contains a graph of the specified waveform. It then converts each point on the MATLAB plot into bit values that the DAC will recognize and be able to output the specified voltage.

The Calibration Code is used to zero the ProPAQ. This should be done before running the simulation just to initialize the system. This should be run before zeroing the probe on the ProPAQ. The process is as follows:
1. Start the GUI and press the zero button
2. Zero the probe on the ProPAQ and change the scaling from 30 to -30
execute the desired waveform, ICP value, and heart rate

Listing 2: Calibration

```
#Imports to Recognize the Dac on the Raspberry Pi
import Adafruit_MCP4725

dac = Adafruit_MCP4725.MCP4725()


zero = 300;
zero = int(zero)
dac.set_voltage(zero) #Set the output Voltage to Zero
```

Figure 2: Normal Waveform

Listing 3: Normal Waveform

```
from scipy import io
import numpy as np
import time
import sys
import math
import Adafruit_MCP4725

dac = Adafruit_MCP4725.MCP4725()
first_arg = sys.argv[1]

def modelone(intvalue = first_arg): #first_arg
    intvalue = int(intvalue)
    mat = io.loadmat('modelonescale.mat');
    y_value = mat['interpoalteddatay'];
    y_value = np.transpose(y_value);
    new_value = np.zeros(y_value.shape)

    for jj in range(len(y_value)):
        if y_value[jj] == 28:
            new_value[jj] = 208
        elif y_value[jj] == 27:
            new_value[jj] = 206
        elif y_value[jj] == 26:
            new_value[jj] = 204
        elif y_value[jj] == 25:
            new_value[jj] = 202
        elif y_value[jj] == 24:
            new_value[jj] = 200
        elif y_value[jj] == 23:
            new_value[jj] = 198
        elif y_value[jj] == 22:
            new_value[jj] = 196
        elif y_value[jj] == 21:
            new_value[jj] = 194
        elif y_value[jj] == 20:
            new_value[jj] = 192
        elif y_value[jj] == 19:
            new_value[jj] = 189
        elif y_value[jj] == 18:
            new_value[jj] = 187
        elif y_value[jj] == 17:
```

```
            new_value[jj] = 184
        elif y_value[jj] == 16:
            new_value[jj] = 182
        elif y_value[jj] == 15:
            new_value[jj] = 179
        elif y_value[jj] == 14:
            new_value[jj] = 177
        elif y_value[jj] == 13:
            new_value[jj] = 174
        elif y_value[jj] == 12:
            new_value[jj] = 172
        elif y_value[jj] == 11:
            new_value[jj] = 169
        elif y_value[jj] == 10:
            new_value[jj] = 167
        elif y_value[jj] == 9:
            new_value[jj] = 164
        elif y_value[jj] == 8:
            new_value[jj] = 162
        elif y_value[jj] == 7:
            new_value[jj] = 158
        elif y_value[jj] == 6:
            new_value[jj] = 156
        elif y_value[jj] == 5:
            new_value[jj] = 154
        elif y_value[jj] == 4:
            new_value[jj] = 152
        elif y_value[jj] == 3:
            new_value[jj] = 149
        elif y_value[jj] == 2:
            new_value[jj] = 147
        elif y_value[jj] == 1:
            new_value[jj] = 144

    var = 1;

    X = math.exp(-(intvalue+193.74)/54.91)
    new_value = new_value.flatten()
    while var == 1:
        for val in new_value:
            val = int(val)
            dac.set_voltage(val)
```

```
                time.sleep(X)
if __name__ == "__main__":
    modelone()
```

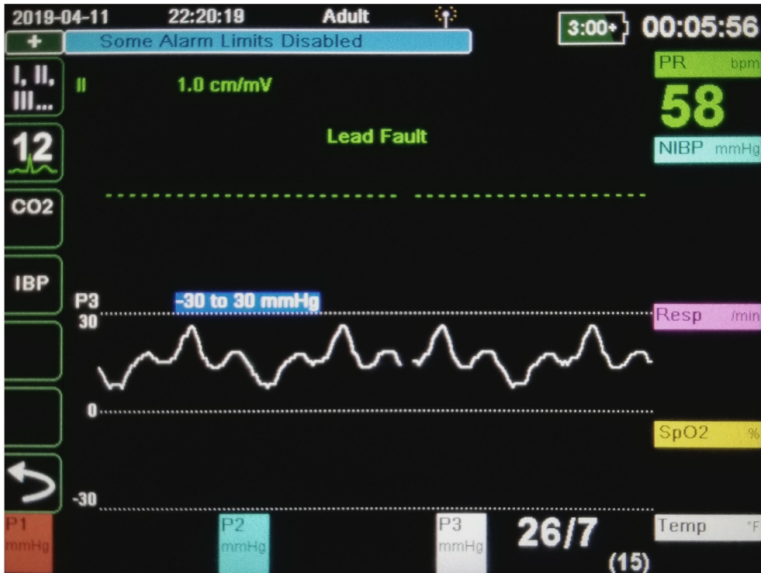

Figure 3: Abnormal Waveform

Listing 4: Abnormal Waveform

```
from scipy import io
import numpy as np
import time
import sys
import math
import Adafruit_MCP4725

dac = Adafruit_MCP4725.MCP4725()
first_arg = sys.argv[1]

def modelthree(intvalue = first_arg):
    intvalue = int(intvalue)
    mat = io.loadmat('ICPL.mat');
    y_value = mat['ICPL'];
    y_value = np.transpose(y_value);
    new_value = np.zeros(y_value.shape)

    for jj in range(len(y_value)):
        if y_value[jj] == 31:
            new_value[jj] = 219
        elif y_value[jj] == 30:
            new_value[jj] = 217
        elif y_value[jj] == 29:
            new_value[jj] = 214
        elif y_value[jj] == 28:
            new_value[jj] = 212
        elif y_value[jj] == 27:
            new_value[jj] = 202
        elif y_value[jj] == 26:
            new_value[jj] = 199
        elif y_value[jj] == 25:
            new_value[jj] = 196
        elif y_value[jj] == 24:
            new_value[jj] = 184
        elif y_value[jj] == 23:
            new_value[jj] = 182
        elif y_value[jj] == 22:
            new_value[jj] = 179
        elif y_value[jj] == 21:
            new_value[jj] = 177
        elif y_value[jj] == 20:
```

```
            new_value[jj] = 167
        elif y_value[jj] == 19:
            new_value[jj] = 164
        elif y_value[jj] == 18:
            new_value[jj] = 162
        elif y_value[jj] == 17:
            new_value[jj] = 160
        elif y_value[jj] == 16:
            new_value[jj] = 150
        elif y_value[jj] == 15:
            new_value[jj] = 148
        elif y_value[jj] == 14:
            new_value[jj] = 146
        elif y_value[jj] == 13:
            new_value[jj] = 144
        elif y_value[jj] == 12:
            new_value[jj] = 142
        elif y_value[jj] == 11:
            new_value[jj] = 139
        elif y_value[jj] == 10:
            new_value[jj] = 138
        elif y_value[jj] == 9:
            new_value[jj] = 136
        elif y_value[jj] == 8:
            new_value[jj] = 134
        elif y_value[jj] == 7:
            new_value[jj] = 132
        elif y_value[jj] == 6:
            new_value[jj] = 130
        elif y_value[jj] == 5:
            new_value[jj] = 128
        elif y_value[jj] == 4:
            new_value[jj] = 126
        elif y_value[jj] == 3:
            new_value[jj] = 120
        elif y_value[jj] == 2:
            new_value[jj] = 116
        elif y_value[jj] == 1:
            new_value[jj] = 114

    new_value = new_value.flatten()
    var = 3;
```

```
    X = math.exp(-(intvalue+193.74)/54.91)
    while var == 3:
        for val in new_value:
            val = int(val)
            dac.set_voltage(val)
            time.sleep(X)

if __name__ == "__main__":
    modelthree()
```

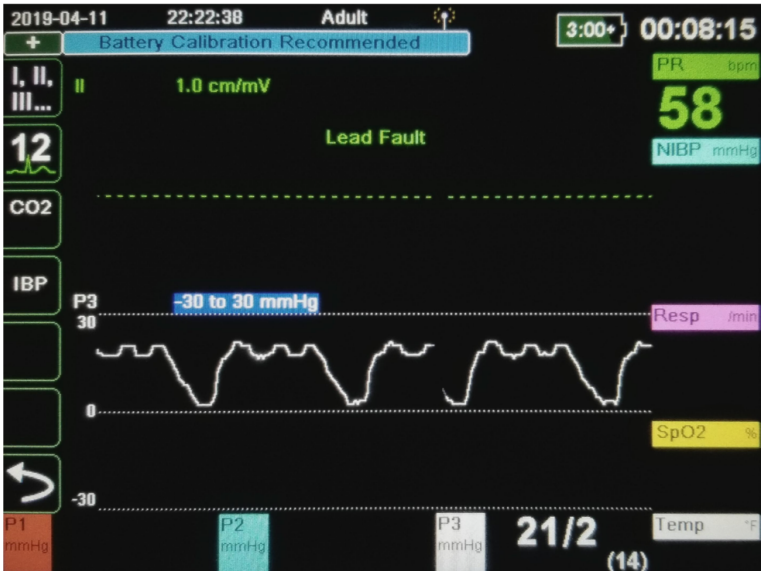

Figure 4: Leveled Waveform

Listing 5: Leveled Waveform

```
from scipy import io
import numpy as np
import time
import sys
import math
import Adafruit_MCP4725

dac = Adafruit_MCP4725.MCP4725()
first_arg = sys.argv[1]

def modeltwo(intvalue = first_arg):
    intvalue = int(intvalue)
    mat = io.loadmat('modeltwoscale.mat');
    y_value = mat['modeltwo'];
    y_value = np.transpose(y_value);
    new_value = np.zeros(y_value.shape)

    for jj in range(len(y_value)):
        if y_value[jj] == 31:
            new_value[jj] = 219
        elif y_value[jj] == 30:
            new_value[jj] = 217
        elif y_value[jj] == 29:
            new_value[jj] = 214
        elif y_value[jj] == 28:
            new_value[jj] = 212
        elif y_value[jj] == 27:
            new_value[jj] = 202
        elif y_value[jj] == 26:
            new_value[jj] = 199
        elif y_value[jj] == 25:
            new_value[jj] = 196
        elif y_value[jj] == 24:
            new_value[jj] = 184
        elif y_value[jj] == 23:
            new_value[jj] = 182
        elif y_value[jj] == 22:
            new_value[jj] = 179
        elif y_value[jj] == 21:
            new_value[jj] = 177
        elif y_value[jj] == 20:
```

```
            new_value[jj] = 167
        elif y_value[jj] == 19:
            new_value[jj] = 164
        elif y_value[jj] == 18:
            new_value[jj] = 162
        elif y_value[jj] == 17:
            new_value[jj] = 160
        elif y_value[jj] == 16:
            new_value[jj] = 150
        elif y_value[jj] == 15:
            new_value[jj] = 148
        elif y_value[jj] == 14:
            new_value[jj] = 146
        elif y_value[jj] == 13:
            new_value[jj] = 144
        elif y_value[jj] == 12:
            new_value[jj] = 142
        elif y_value[jj] == 11:
            new_value[jj] = 139
        elif y_value[jj] == 10:
            new_value[jj] = 138
        elif y_value[jj] == 9:
            new_value[jj] = 136
        elif y_value[jj] == 8:
            new_value[jj] = 134
        elif y_value[jj] == 7:
            new_value[jj] = 132
        elif y_value[jj] == 6:
            new_value[jj] = 130
        elif y_value[jj] == 5:
            new_value[jj] = 128
        elif y_value[jj] == 4:
            new_value[jj] = 126
        elif y_value[jj] == 3:
            new_value[jj] = 120
        elif y_value[jj] == 2:
            new_value[jj] = 116
        elif y_value[jj] == 1:
            new_value[jj] = 114

    new_value = new_value.flatten()
    var = 2;
```

```
84      X = math.exp(-(intvalue+193.74)/54.91)
85      while var == 2:
86          for val in new_value:
87              val = int(val)
88              dac.set_voltage(val)
89              time.sleep(X)
90  if __name__ == "__main__":
91      modeltwo()
```