

```

1  import os
2  import glob
3  import time
4  import RPi.GPIO as GPIO
5  import sys
6  import shutil
7  GPIO.setwarnings(False)
8
9  #sets up IO and temp readings
10 os.system('modprobe w1-gpio')
11 os.system('modprobe w1-therm')
12
13 base_dir = '/sys/bus/w1/devices/'
14 #assigns variable to each temperature sensor address
15 device_file1 = glob.glob(base_dir + '28-0000074b7f2d')[0] + '/w1_slave'
16 device_file2 = glob.glob(base_dir + '28-0000074b8662')[0] + '/w1_slave'
17 #variable definitions
18 oldtime = time.time() - 60
19 oldtime2 = time.time()
20 oldday = time.localtime()
21 otime = time.time()
22 ostate = 0
23 startup = 1
24 global flow
25 flow = 47
26 energy = 0
27 totalen = 0
28 power = 0
29 dayen = 0
30
31 #flashes led on call
32 def flash(pin, state, wait):
33     boo = not state
34     GPIO.output(pin, state)
35     time.sleep(wait)
36     GPIO.output(pin, boo)
37
38 #opens temperature sensor when passed address
39 def read_temp_raw(d_file):
40     f = open(d_file, 'r')
41     lines = f.readlines()
42     f.close()
43     return lines
44
45 #reads and returns the temperature
46 def read_temp(d_file):
47     lines = read_temp_raw(d_file)
48     while lines[0].strip()[-3:] != 'YES':
49         time.sleep(0.2)
50         lines = read_temp_raw()
51     equals_pos = lines[1].find('t=')
52     if equals_pos != -1:
53         temp_string = lines[1][equals_pos+2:]
54         temp_c = float(temp_string) / 1000.0
55         return temp_c
56
57 #updates the vent LED and returns the vent state
58 def IRup():
59     if GPIO.input(24):
60         GPIO.output(17, 0)
61         state = False
62     else:
63         GPIO.output(17, 1)
64         state = True
65     return state
66
67 #delays the program until specified time or until the measurement button has been

```

```

pushed or the vent opens
68 def wait(itime, oldtime):
69     while(((time.time() - oldtime) < itime) and (IRup() == False) and (GPIO.input(23)
    == True)):
70         time.sleep(.1)
71         usave()
72
73 #tries to save data to a usb if there is one inserted
74 def usave():
75     folders = (os.popen("ls -l /media/pi/").readlines())
76     i = 0
77     flow = 0
78     while (i < len(folders)):
79         stringf = folders[i].split(" ")
80         if (len(folders[i]) > 15):
81             if (stringf[2] == "pi"):
82                 stringf = folders[i].split(":")
83                 if (len(stringf) == 1):
84                     stringf = stringf[0].split("196")
85                     stringf = stringf[1].split(" ")
86                     h = len(stringf)
87                     g = 1
88                     folders[i] = stringf[g]
89                     g = g + 1
90                     while (g < h):
91                         folders[i] = folders[i] + " " + stringf[g]
92                         g = g + 1
93                     folders[i] = folders[i].rstrip()
94                     print (folders[i] + " device has been inserted.\n")
95                     GPIO.output(6,1)
96                     usbfolders = os.listdir("/media/pi/" + folders[i])
97                     j = 0
98                     while (j < len(usbfolders)):
99                         if(usbfolders[j] == 'Data'):
100                             k = 0
101                             dirfiles = os.listdir("/media/pi/" + folders[i] + '/' +
                                usbfolders[j])
102                             while (k < len(dirfiles)):
103                                 if(dirfiles[k] == '1-AIRSPEED.txt'):
104                                     if (lps('/media/pi/' + folders[i] +
                                        '/Data/1-AIRSPEED.txt')):
105                                         shutil.copyfile('/media/pi/' + folders[i] +
                                            '/Data/1-AIRSPEED.txt',
                                            '/home/pi/Desktop/Programs/Data/1-AIRSPEED.txt')
106                                     k = k + 1
107                                     shutil.rmtree('/media/pi/' + folders[i] + '/Data')
108                                     time.sleep(0.1)
109                                     j = j + 1
110                                     shutil.copypath('/home/pi/Desktop/Programs/Data', '/media/pi/' +
                                        folders[i] + '/Data')
111                                     time.sleep(0.1)
112                                     os.system('sudo udisks --unmount /dev/sda1')
113                                     GPIO.output(6, 0)
114                                     time.sleep(0.4)
115                                     flash(6,1,0.1)
116                                     flash(6,0,0.1)
117                                     flash(6,1,0.1)
118                                     print (folders[i] + " device has been removed.\n")
119                                     i = i + 1
120                                     GPIO.output(6, 0)
121
122 #reads the LPS file drive and assigns a new LPS amount
123 def lps(directory):
124     global flow
125     with open(directory, "r") as ff:
126         fstring = ff.read()
127         if(len(fstring) > 3):

```

```

128         if (fstring[1] == '1'):
129             start = 1;
130         elif (fstring[2] == '1'):
131             start = 2
132         elif (fstring[3] == '1'):
133             start = 3
134         elif (fstring[4] == '1'):
135             start = 4
136         if ((fstring[start] == '1') and (fstring[start + 1] == 'p') and
137             (fstring[start + 2] == 's')):
138             print (fstring)
139             k = 0
140             lstring = ''
141             while (k < start):
142                 lstring = lstring + fstring[k]
143                 k = k + 1
144             flow = int(lstring)
145             return True
146         return False
147
148 #START OF MAIN PROGRAM
149
150 lps("/home/pi/Desktop/Programs/Data/1-AIRSPEED.txt")
151                                     #sets LPS
152 while 1:                             #loops indefinitely
153     GPIO.setmode(GPIO.BCM)           #set up GPIO using BCM numbering
154     GPIO.setup(5, GPIO.OUT)
155     GPIO.setup(17, GPIO.OUT)
156     GPIO.setup(6, GPIO.OUT)
157     GPIO.setup(24, GPIO.IN)
158     GPIO.setup(23, GPIO.IN, pull_up_down = GPIO.PUD_UP)
159     usave()                          #tries to copy data to usb
160     if IRup():                       #checks the vent position and waits a different
161         amount                       #of time depending on the last time the vent
162         otime = time.time()          #was open
163         wait(1, oldtime)              #if the vent is open now
164     elif ((time.time() - otime) <= 300):
165         wait(10, oldtime)             #if the vent was last open under 300 secs ago
166     elif ((time.time() - otime) <= 900):
167         wait(30, oldtime)             #if the vent was last open under 900 secs ago
168     else:
169         wait(60, oldtime)             #else
170         oldtime = time.time()         #saves the measurement start time
171         GPIO.output(5, 1)            #turns on the measurement light
172         temp1 = read_temp(device_file1) #reads the first temperature
173         IRup()
174         flash(5, 0, 0.1)             #flashes LED and updates vent
175         IRup()
176         temp2 = read_temp(device_file2) #reads the second temperature
177         IRup()
178         flash(5, 0, 0.1)             #flashes LED and updates vent
179         IRup()                       #creates date string
180         date2 = '/home/pi/Desktop/Programs/Data/' + str((time.localtime()).tm_year) + "-" +
181         str((time.localtime()).tm_mon) + "-" + str((time.localtime()).tm_mday) + '.csv'
182         with open(date2, 'a') as f:   #opens file to save data
183             if (f.tell() == 0):      #adds headers if it's a new file
184                 f.write("Day of month,Date,Vent Temperature,Ambient Temperature,Difference
185                 in Temperature,Vent Open/Closed,Power (W),Energy(kWh),Total Energy(kWh)\n")
186             timedata = str(time.ctime()).split(" ")
187             if (timedata[2] == ''):   #splits up date and time to reformat it
188                 timedata[2] = timedata[3]
189                 timedata[3] = timedata[4]
190                 timedata[4] = timedata[5] #reformats date string
191             savetime = "\"" + timedata[1] + " " + timedata[2] + ", " + timedata[4] + " " +
192             timedata[3] + "\""
193             tempdata = str((time.localtime()).tm_mday) + "," + savetime + ',' +

```

[illegible]

```

246     1.7")
    print("
    |
    |
    |
    |
    |
    |
    |
    |
    |
    |
    |")
247     powera = (str(format(power / 1000, '.4f')) + " - |"
248     power = power - 25
249     while (power >= 0):
250         powera = powera + "="
251         power = power - 12.5
252     powera = powera + "|"
253     print(powera)
254     print("\nEnergy")
255     print("Energy kWh 0.0      0.1      0.2      0.3      0.4      0.5      0.6
    0.7      0.8      0.9      1.0      1.1      1.2      1.3      1.4      1.5      1.6
    1.7")
256     print("
    |
    |
    |
    |
    |
    |
    |
    |
    |
    |
    |")
    print("\n")
257     totalena = (str(format(totalen, '.4f')) + " - |"
258     totalenb = totalen - 0.025
259     while (totalenb >= 0):
260         totalena = totalena + "="
261         totalenb = totalen - 0.0125
262     totalena = totalena + "|"
263     print(totalena)
264
265     print("\n\n\n")
266
267
268     IRup()                                #updates vent
269     GPIO.output(5, 0)                     #flashes LED to show end of measurement
270     flash(5, 1, 0.1)
271     flash(5, 0, 0.1)
272     flash(5, 1, 0.1)
273     time.sleep(0.5)                       #delay to next read
274     IRup()                                #updates vent
275     startup = 0
276
277
278     #END OF MAIN PROGRAM
279
280
281
282
283
284
285

```