```
# GUI to control RPI GPIO
# 3.2.17
# by Julian Rogers
# gpio_3

from tkinter import *   #GUI

import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(27, GPIO.OUT)
GPIO.setup(22, GPIO.OUT)

GPIO.output(4, False)
GPIO.output(17, False)
GPIO.output(27, False)
GPIO.output(22, False)

import datetime
import time
import socket           #UDP
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
import smbus
bus = smbus.SMBus(1)

# create the root window
root = Tk()

# modify the window
root.title("Julian's IOT CONTROLLER")
#root.geometry("700x430")
root.configure(bg = "gray")
#root.attributes('-fullscreen', True)   #eliminates the title bar

w, h = root.winfo_screenwidth(), root.winfo_screenheight()
root.geometry("%dx%d+0+0" % (w, h))

# create a frame
app = Frame(root)
app.configure(bg = "gray")
app.grid()

title_lab = Label(app, text = "GPIO CONTROL- Broadcom Numbering", font = ("Arial Bold", 20), fg = "maroon", bg = "gray")
title_lab.grid(row = 1, column = 1, columnspan = 8)

blank_lab = Label(app, bg = "gray")
blank_lab.grid(row = 2, column = 1)

sys_1_but = Button(app, text = "Sys 1(04)", font = ("Arial", 16), fg = "maroon", bg = "light blue")
sys_1_but.grid(row = 3, column = 1)

def on_off_1():
    but_col = sys_1_but.config('bg')[-1]
    if but_col == "light blue":
        but_col = "yellow"
        GPIO.output(4, True)
    else:
        but_col = "light blue"
        GPIO.output(4, False)

    sys_1_but.config(bg = but_col)


sys_1_but.config(command = on_off_1)


sys_2_but = Button(app, text = "Sys 2(17)", font = ("Arial", 16), fg = "maroon", bg = "light blue")
sys_2_but.grid(row = 3, column = 2)

def on_off_2():
    but_col = sys_2_but.config('bg')[-1]
    if but_col == "light blue":
        but_col = "yellow"
        GPIO.output(17, True)
    else:
        but_col = "light blue"
        GPIO.output(17, False)

    sys_2_but.config(bg = but_col)


sys_2_but.config(command = on_off_2)

sys_3_but = Button(app, text = "Sys 3(27)", font = ("Arial", 16), fg = "maroon", bg = "light blue")
sys_3_but.grid(row = 3, column = 3)

def on_off_3():
    but_col = sys_3_but.config('bg')[-1]
    if but_col == "light blue":
        but_col = "yellow"
        GPIO.output(27, True)
    else:
        but_col = "light blue"
        GPIO.output(27, False)

    sys_3_but.config(bg = but_col)


sys_3_but.config(command = on_off_3)

sys_4_but = Button(app, text = "Sys 4(22)", font = ("Arial", 16), fg = "maroon", bg = "light blue")
sys_4_but.grid(row = 3, column = 4)

blank_lab2 = Label(app, bg = "gray")
blank_lab2.grid(row = 4, column = 1)
```

```
tmp_lab = Label(app, bg = "gray", fg = "maroon", text = "temp ", font = ("Arial", 16) )
tmp_lab.grid(row = 5, column = 1)
temp_lab = Label(app, bg = "gray", fg = "maroon", text = "reading...", font = ("Arial", 16) )
temp_lab.grid(row = 5, column = 2)

def on_off_4():
    but_col = sys_4_but.config('bg')[-1]
    if but_col == "light blue":
        but_col = "yellow"
        GPIO.output(22, True)
    else:
        but_col = "light blue"
        GPIO.output(22, False)

    sys_4_but.config(bg = but_col)


sys_4_but.config(command = on_off_4)

def read_temperature():
    #some error handling needed here
    data = bus.read_i2c_block_data(0x48,0)
    msb = data[0]
    lsb = data[1]
    temperature = (((msb << 8) | lsb) >> 4) * 0.0625
    temp_lab.config(text = round(temperature,1))

# calls a function after given time
def after(self, ms, func = None, *args):
    """call function after a given time"""

# updates screen every 10 seconds
def task():

    read_temperature()
    root.after(10000, task)

# calls the screen update every 10 seconds
root.after(10000, task)

#---------------------------------------------------------------------------------------
# kick off the window's event-loop
root.mainloop()
```