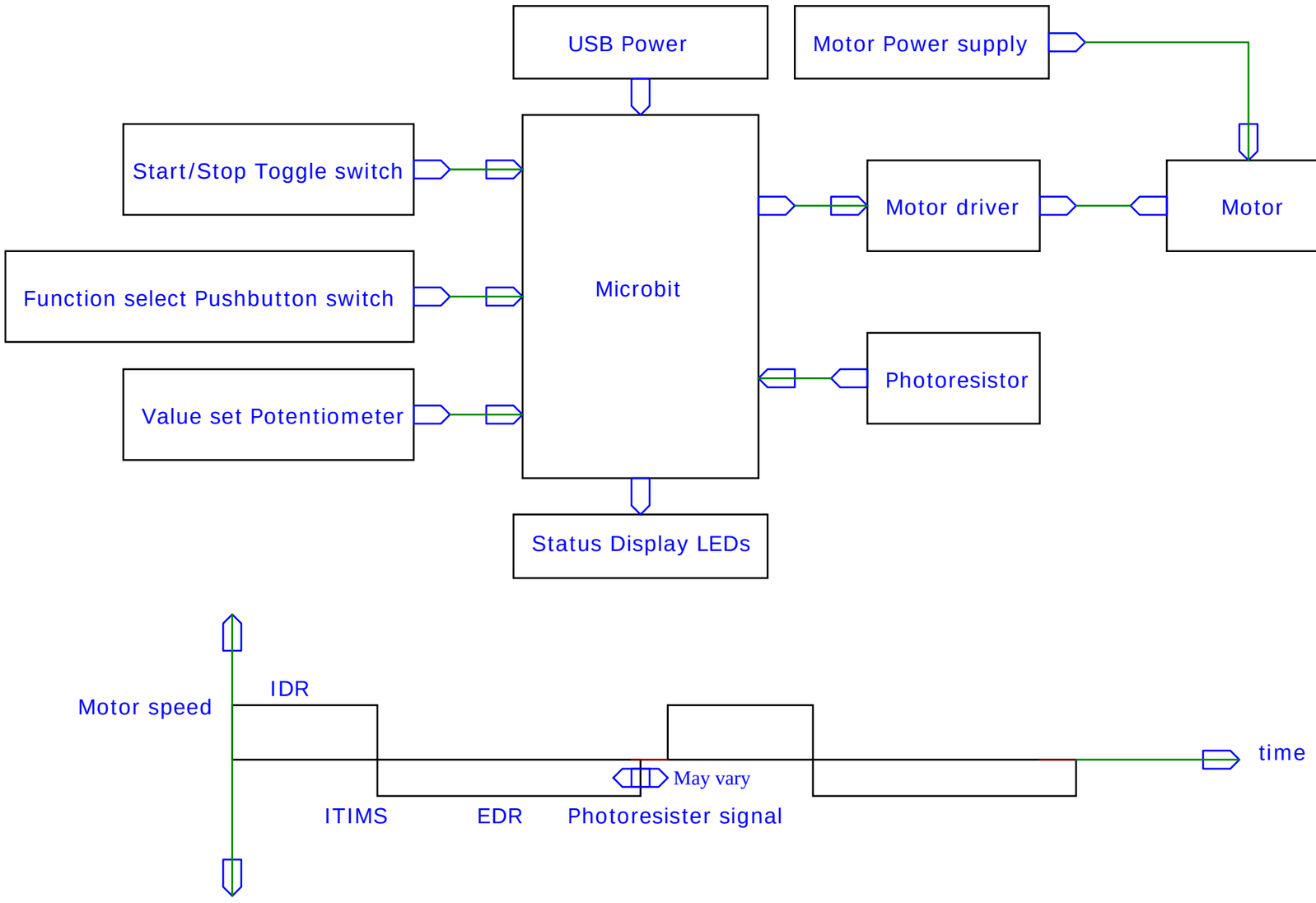
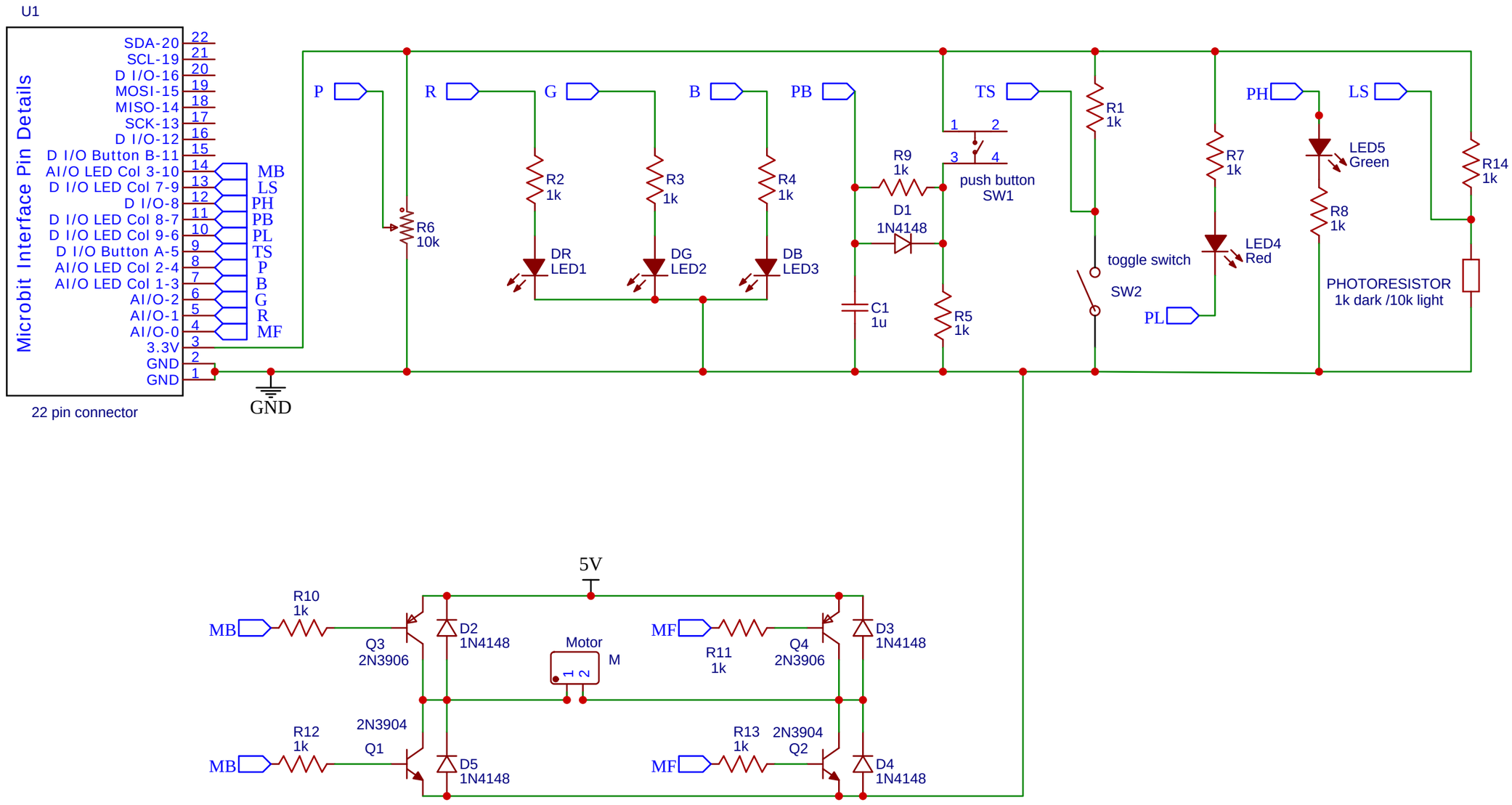
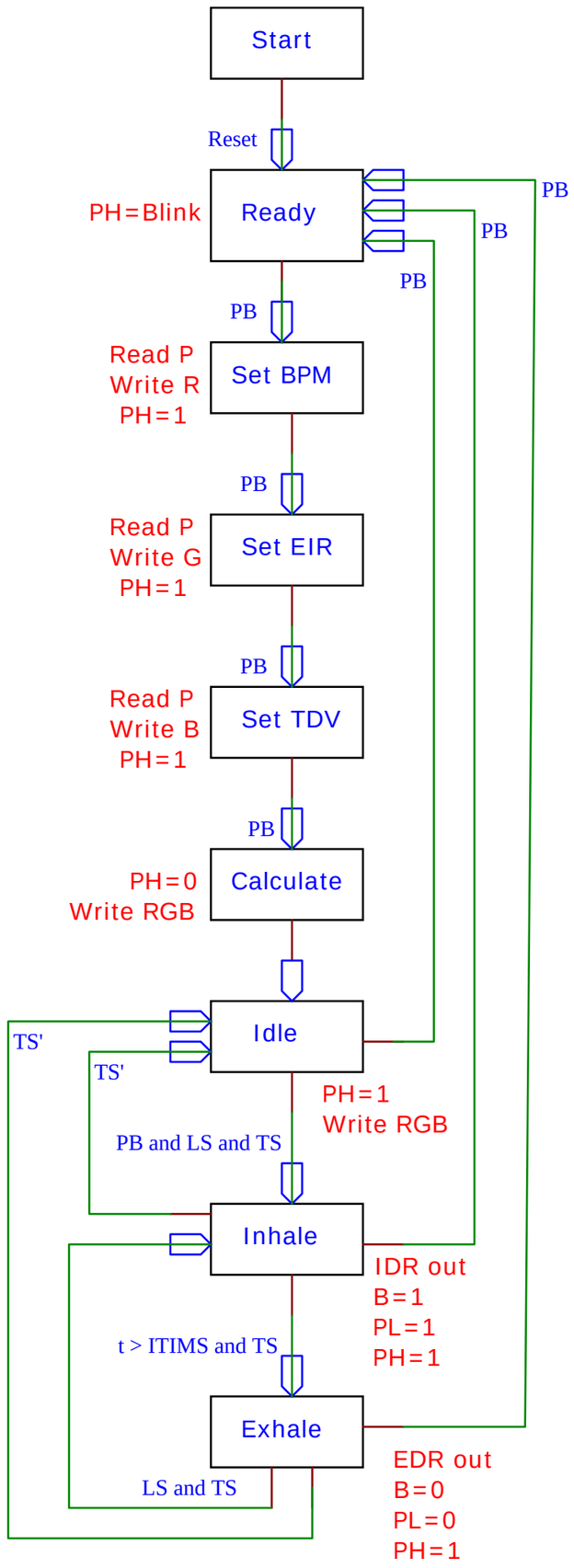


TITLE: Ventilator Block Diagram		REV: 2.0
EasyEDA	Company: UW-Madison	Sheet: 1/1
	Date: 2020-Apr-21 Drawn By: Surya Santhan Thenarasu	



TITLE: ECE 210 Ventilator controller circuit diagram		REV: 2.0
EasyEDA	Company: UW-Madison	Sheet: 1/1
	Date: 2020-Apr-21	Drawn By: Surya Santhan Thenarasu



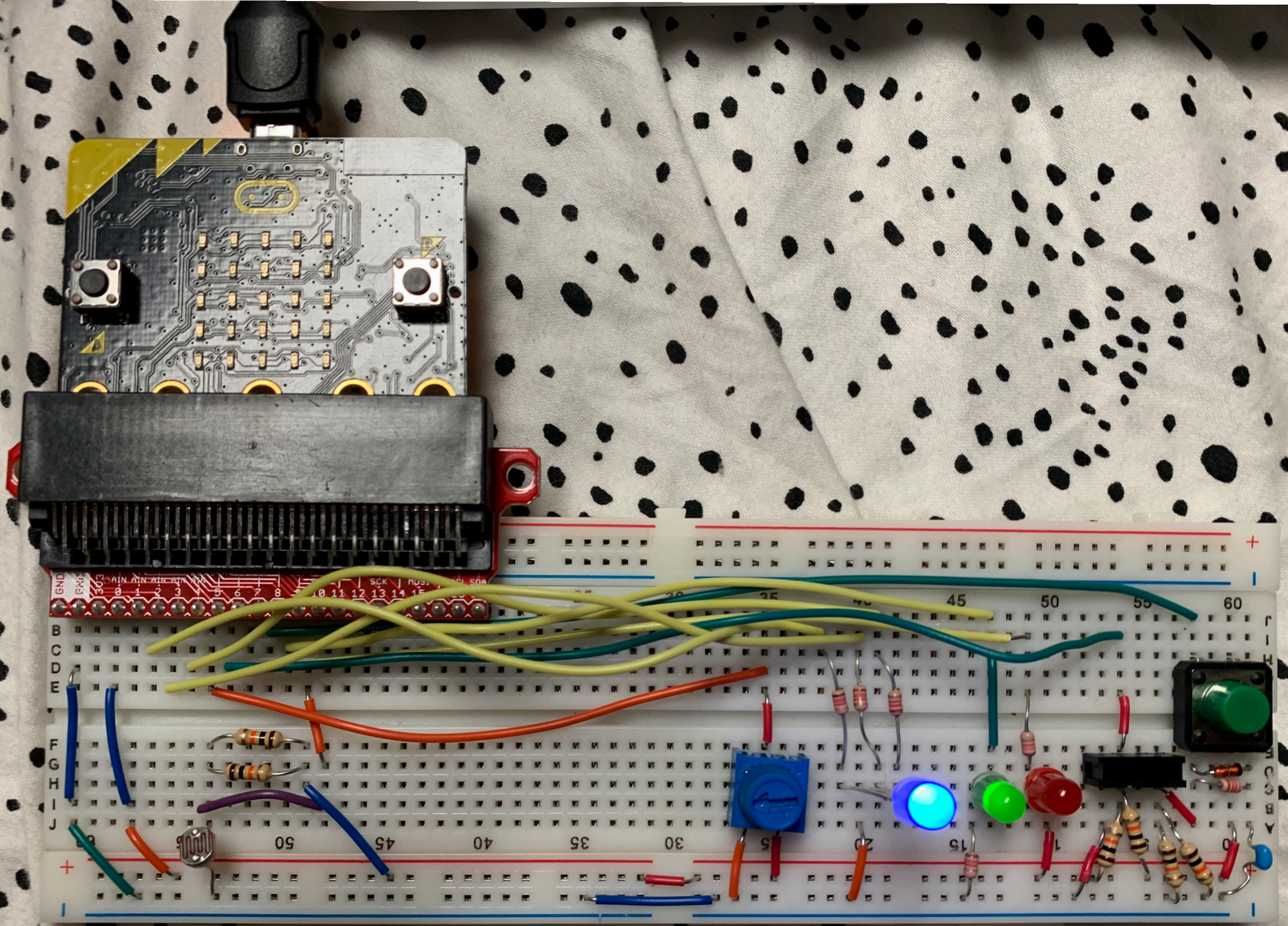
TITLE: Ventilator Flow Chart		REV: 2
EasyEDA	Company: UW Madison	Sheet: 1/1
	Date: 2020-Apr-21 Drawn By: Surya Santhan Thenarasu	

```
# Write your code here :-)
# This program was written by Surya Santhan Thenarasu
# ECE 210
# 16 Apr 2020
# Ventilator control project
# When you press the PB, the RGB LED sequences between
# Ready (Blinking Green), SetBPM (Red),
# SetEIR(Green), SetTDV(Blue), Calculate (RGB) , Idle (R)
# Inhale(B), Exhale (R,B), Ready states
# During the SetXXX states, the POT can be used to set the
# values for the corresponding items through the brightness of each color
# Pin 0 to Motor Forward
# Pins 1-2-3 are to RGB of common cathode LED drive
# Pin 4 to Pot
# Pin 5 to TS
# Pin 6 to PL (Red LED)
# Pin 7 to reading the push-button switch with debounced hardware
# Pin 8 to PH (Green LED)
# Pin 9 to reading the LS photoresistor output to indicate end of exhale cycle
# Pin 10 to Motor Backward

from microbit import *

display.off()

# Defining the state identiification
READY, SetBPA, SetEIR, SetTDV, Calculate, Idle, Inhale, Exhale = range(8)
# Initializing the states
state = READY

# Defining the ready state function
def handle_ready_state():
    global state
    pin8.write_digital(1)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin1.write_digital(0)  # Keep R of RGB off
    pin2.write_digital(0)  # Keep G of RGB off
    pin3.write_digital(0)  # Keep B of RGB off
    pin6.write_digital(1)  # Keep Red LED off
    pin10.write_digital(0)  # Keep motor backward off
    sleep(350)
    if pin7.read_digital():  # If PB next state
        state = SetBPA
    pin8.write_digital(0)
    sleep(350)

# Defining the SetBPA state function
```

```
def handle_SetBPA_state():
    global state, R_BPA
    R_BPA = pin4.read_analog()
    pin8.write_digital(1)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin1.write_analog(R_BPA)  # Write R_BPA into R
    pin2.write_digital(0)  # Keep G of RGB off
    pin3.write_digital(0)  # Keep B of RGB off
    pin6.write_digital(1)  # Keep Red LED off
    pin10.write_digital(0)  # Keep motor backward off
    sleep(50)
    if pin7.read_digital():  # If PB next state
        state = SetEIR
    sleep(350)

# Defining the SetEIR state function
def handle_SetEIR_state():
    global state, R_EIR
    R_EIR = pin4.read_analog()
    pin8.write_digital(1)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin2.write_analog(R_EIR)  # Write R_EIR into G
    pin1.write_digital(0)  # Keep R of RGB off
    pin3.write_digital(0)  # Keep B of RGB off
    pin6.write_digital(1)  # Keep Red LED off
    pin10.write_digital(0)  # Keep motor backward off
    sleep(50)
    if pin7.read_digital():  # If PB next state
        state = SetTDV
    sleep(350)

# Defining the SetTDV state function
def handle_SetTDV_state():
    global state, R_TDV
    R_TDV = pin4.read_analog()
    pin8.write_digital(1)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin3.write_analog(R_TDV)  # Write R_EIR into B
    pin1.write_digital(0)  # Keep R of RGB off
    pin2.write_digital(0)  # Keep G of RGB off
    pin6.write_digital(1)  # Keep Red LED off
    pin10.write_digital(0)  # Keep motor backward off
    sleep(50)
    if pin7.read_digital():  # If PB next state
        state = Calculate
    sleep(350)
```

```
# Defining the Calculate state function
def handle_Calculate_state():
    global state, R_BPA, R_EIR, R_TDV, ITIMS, IDRB, EDRB
    sleep(50)
    pin8.write_digital(0)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin2.write_digital(1)  # Keep G of RGB on
    pin1.write_digital(1)  # Keep R of RGB on
    pin3.write_digital(1)  # Keep B of RGB on
    pin6.write_digital(1)  # Keep Red LED off
    pin10.write_digital(0)  # Keep motor backward off
#    ITIMS = 60000/(4+R_BPA/28)*1/(2+R_EIR/1024)  # caclulate ITIMS
#    IDRB = (200+R_TDV/600)*1024/900*1000/ITIMS   # caclulate IDRB
#    EDRB = IDRB*1/(2+R_EIR/1024)  # caclulate EDRB
    ITIMS = 600 # Assign ITIMS
    IDRB = 1000  # Assign EDRB
    EDRB = 800  # Assign EDRB
    sleep(50)
    state = Idle

# Defining the Idle state function
def handle_Idle_state():
    global state
    pin8.write_digital(1)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin2.write_digital(1)  # Keep G of RGB on
    pin1.write_digital(1)  # Keep R of RGB off
    pin3.write_digital(1)  # Keep B of RGB off
    pin6.write_digital(1)  # Keep Red LED off
    pin10.write_digital(0)  # Keep motor backward off
    sleep(50)
#    if pin7.read_digital() and pin5.read_digital() and pin9.read_digital():
    if pin7.read_digital():  # If PB Ready
        state = READY
    elif pin5.read_digital() and pin9.read_digital():  # If TS and LS Inhale
        state = Inhale
    sleep(350)

# Defining the Inhale state function
def handle_Inhale_state():
    global state, IDRB, ITIMS
    pin8.write_digital(1)  # turn on green LED
    pin0.write_digital(0)  # Keep motor forward off
    pin2.write_digital(0)  # Keep G of RGB off
    pin1.write_digital(0)  # Keep R of RGB off
    pin3.write_digital(0)  # Keep B of RGB off
    pin6.write_digital(0)  # Keep Red LED on
```

```
    pin10.write_analog(IDRB)  # Keep motor backward on
    if pin7.read_digital():  # If PB Ready
        state = READY
    elif not pin5.read_digital():  # If not TS Idle
        state = Idle
    else:
        sleep(ITIMS)  # Wait for ITIMS to exhale
        state = Exhale

# Defining the Exhale state function
def handle_Exhale_state():
    global state, EDRB
    pin8.write_digital(1)  # turn on green LED
    pin0.write_analog(EDRB)  # Keep motor forward on
    pin2.write_digital(0)  # Keep G of RGB off
    pin1.write_digital(0)  # Keep R of RGB off
    pin3.write_digital(1)  # Keep B of RGB on
    pin6.write_digital(1)  # Keep Red LED on
    pin10.write_digital(0)  # Keep motor backward off
    if pin7.read_digital():  # If PB Ready
        state = READY
    elif not pin5.read_digital():  # If not TS Idle
        state = Idle
    elif pin9.read_digital():  # Wait for LS to Inhale
        state = Inhale


# Main state machine
while True:
    if state == READY:
        handle_ready_state()
    elif state == SetBPA:
        handle_SetBPA_state()
    elif state == SetEIR:
        handle_SetEIR_state()
    elif state == SetTDV:
        handle_SetTDV_state()
    elif state == Calculate:
        handle_Calculate_state()
    elif state == Idle:
        handle_Idle_state()
    elif state == Exhale:
        handle_Exhale_state()
    elif state == Inhale:
        handle_Inhale_state()
```


GND
3V3
0 1 2 3 4 5 6 7 8 10 11 12 13 14
SCK
B
C
D
E
F
G
H
I
J
50
45
40
35
30
25
20
15
10
5
0