Lab 3

Colored LEDs

In [ ]:
```
import analogio
import pwmio
import time
from board import *

red_knob = analogio.AnalogIn(A0)
red_light = pwmio.PWMOut(GP15)
green_knob = analogio.AnalogIn(A1)
green_light = pwmio.PWMOut(GP14)
blue_knob = analogio.AnalogIn(A2)
blue_light = pwmio.PWMOut(GP13)

MAX = 2**16
GAMMA = 3.0

def gamma_correction(value):
    return int(((value / MAX) ** GAMMA) * MAX)

while True:
    red_value = red_knob.value
    red_gamma = gamma_correction(red_value)
    green_value = green_knob.value
    green_gamma = gamma_correction(green_value)
    blue_value = blue_knob.value
    blue_gamma = gamma_correction(blue_value)

    red_light.duty_cycle = red_gamma
    green_light.duty_cycle = green_gamma
    blue_light.duty_cycle = blue_gamma

    time.sleep(0.05)
```

Describe the behavior of the LED - can you replicate any color? How about black?

You can replicate almost any color because all colors are composed of Red, Green, and Blue. Unfortunatly you can't replicate black because that would require a value of zero for all three colors which would just be an off LED.

Random Colors

In [ ]:
```
import analogio
import pwmio
import time
from random import randint
from board import *

red_light = pwmio.PWMOut(GP15)
green_light = pwmio.PWMOut(GP14)
blue_light = pwmio.PWMOut(GP13)

MAX = 2**16

def random_color():
    return randint(0, MAX)

while True:
    red_light.duty_cycle = random_color()
    green_light.duty_cycle = random_color()
    blue_light.duty_cycle = random_color()

    time.sleep(1)
```

Color Palettes

In [ ]:
```
import seaborn as sns
sns.color_palette("hls", 8)
```

In [ ]:
```
colors = sns.color_palette("hls", 8)
print(colors)
```

In [ ]:

```
import analogio
import pwmio
import time
from board import *

red_light = pwmio.PWMOut(GP15)
green_light = pwmio.PWMOut(GP14)
blue_light = pwmio.PWMOut(GP13)

MAX = 2**16
GAMMA = 3.0

color_palette = [(0.86, 0.3712, 0.33999999999999997),
(0.86, 0.7612000000000001, 0.33999999999999997),
(0.5688000000000001, 0.86, 0.33999999999999997),
(0.33999999999999997, 0.86, 0.5012000000000001),
(0.33999999999999997, 0.8287999999999999, 0.86),
(0.33999999999999997, 0.43879999999999986, 0.86),
(0.6311999999999998, 0.33999999999999997, 0.86),
(0.86, 0.33999999999999997, 0.6987999999999996)]

corrected_palette = []

for x in color_palette:
    corrected_color = []
    for i in range(len(x)):
        y = x[i]
        corrected_color.append(int((y ** GAMMA) * MAX))
    corrected_palette.append(corrected_color)

while True:
    for colors in corrected_palette:
        red_light.duty_cycle = colors[0]
        green_light.duty_cycle = colors[1]
        blue_light.duty_cycle = colors[2]

        time.sleep(1)
```

Bonus: Frequency

In [ ]:

```
import analogio
import pwmio
import time
from board import *

knob = analogio.AnalogIn(A0)
red_light = pwmio.PWMOut(GP15)
green_light = pwmio.PWMOut(GP14)
blue_light = pwmio.PWMOut(GP13)

MAX = 2**16
GAMMA = 3.0

def frequency_correction(value):
    return ((value / MAX) ** GAMMA)

color_palette = [(0.86, 0.3712, 0.33999999999999997),
(0.86, 0.7612000000000001, 0.33999999999999997),
(0.5688000000000001, 0.86, 0.33999999999999997),
(0.33999999999999997, 0.86, 0.5012000000000001),
(0.33999999999999997, 0.8287999999999999, 0.86),
(0.33999999999999997, 0.43879999999999986, 0.86),
(0.6311999999999998, 0.33999999999999997, 0.86),
(0.86, 0.33999999999999997, 0.6987999999999996)]

corrected_palette = []

for x in color_palette:
    corrected_color = []
    for i in range(len(x)):
        y = x[i]
        corrected_color.append(int((y ** GAMMA) * MAX))
    corrected_palette.append(corrected_color)

print(corrected_palette)

while True:
    for colors in corrected_palette:
        value = knob.value
        frequency = frequency_correction(value)

        red_light.duty_cycle = colors[0]
        green_light.duty_cycle = colors[1]
        blue_light.duty_cycle = colors[2]

        time.sleep(frequency)
```

Bonus: Fading

In [ ]:

```
import analogio
import pwmio
import time
from board import *

red_light = pwmio.PWMOut(GP15)
green_light = pwmio.PWMOut(GP14)
blue_light = pwmio.PWMOut(GP13)

MAX = 2**16
GAMMA = 3.0

def calc_increment(old, new):
    if old < new:
        return int(abs(old - new)/10)
    else:
        return -int(abs(old - new)/10)

color_palette = [(0.86, 0.3712, 0.33999999999999997),
(0.86, 0.7612000000000001, 0.33999999999999997),
(0.5688000000000001, 0.86, 0.33999999999999997),
(0.33999999999999997, 0.86, 0.5012000000000001),
(0.33999999999999997, 0.8287999999999999, 0.86),
(0.33999999999999997, 0.43879999999999986, 0.86),
(0.6311999999999998, 0.33999999999999997, 0.86),
(0.86, 0.33999999999999997, 0.6987999999999996),
(0.86, 0.3712, 0.33999999999999997)]

corrected_palette = []

for x in color_palette:
    corrected_color = []
    for i in range(len(x)):
        y = x[i]
        corrected_color.append(int((y ** GAMMA) * MAX))
    corrected_palette.append(corrected_color)

while True:
    for i in range(len(corrected_palette)-1):
        red = corrected_palette[i][0]
        red_new = corrected_palette[i+1][0]

        blue = corrected_palette[i][1]
        blue_new = corrected_palette[i+1][1]

        green = corrected_palette[i][2]
        green_new = corrected_palette[i+1][2]

        for x in range(1,11):
            red_light.duty_cycle = red + (x * calc_increment(red, red_new))
            blue_light.duty_cycle = blue + (x * calc_increment(blue, blue_new
            green_light.duty_cycle = green + (x * calc_increment(green, green_
            time.sleep(0.05)
```

```
        time.sleep(0.5)
```