Lesson 7 Breathing LED

Overview

In this lesson, we will learn how to program the Raspberry Pi to generate PWM signal. And use the PWM square-wave signal control an LED gradually becomes brighter and then gradually becomes dark like the animal's breath.

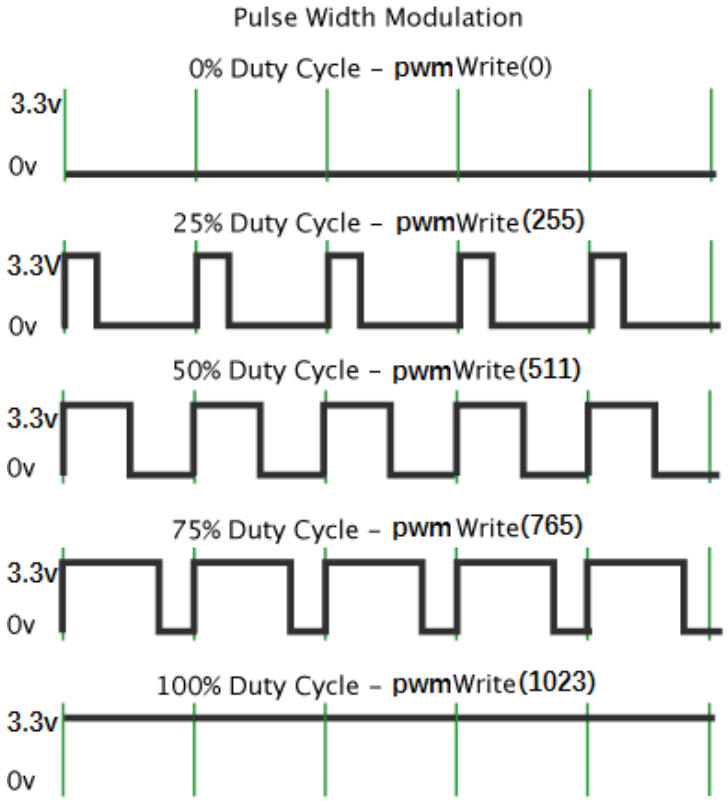
Requirement

- 1* Raspberry Pi
- 1* LED
- 1* 220Ω Resistor
- 1* Breadboard
- Several Jumper wires

Principle

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and 3.3v controlling the brightness of the LED.

In the graphic below, the green lines represent a regular time period. This duration or period is the inverse of the PWM frequency. In other words, with Raspberry Pi's PWM frequency at about 500Hz, the green lines would measure 2 milliseconds each. A call to pwmWrite() is on a scale of 0-1023, such that pwmWrite(1023) requests a 100% duty cycle (always on), and pwmWrite(511) is a 50% duty cycle (on half the time) for example.



Key function:

C user:

- pwmWrite(int pin, int value)

Writes the value to the PWM register for the given pin. The Raspberry Pi has one on-board PWM pin, pin 1 (BMC_GPIO 18, Phys 12) and the range is 0-1024. Other PWM devices may have other PWM ranges.

This function is not able to control the Pi's on-board PWM when in Sys mode.

Python user:

- p = GPIO.PWM(channel, frequency)

This is used for creating a PWM.

- p.start(dc)

Start the pwm you have created.

- p.ChangeFrequency(freq)

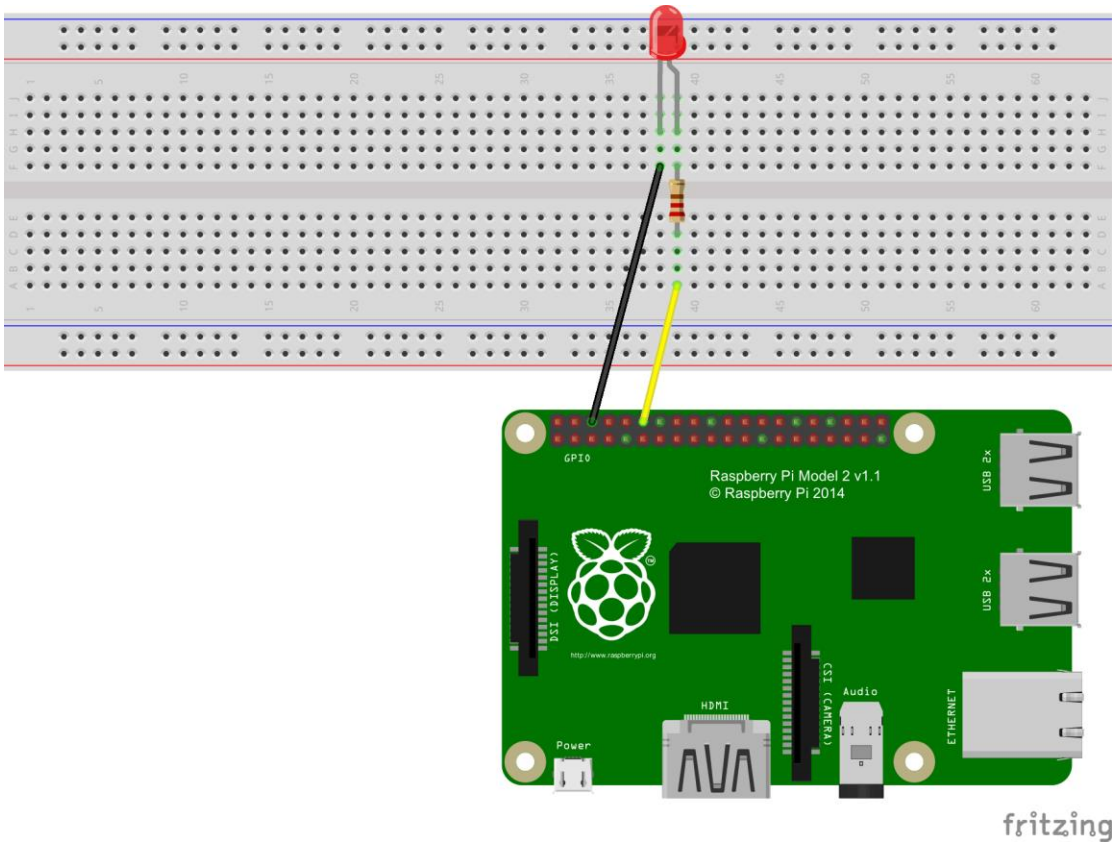
Change the frequency of pwm.

- p.stop( )

Stop the pwm.

Procedures

1. Build the circuit



2. Program

C user:

2.1 Edit and save the code with vim or nano.

(Code path: /home/Adeept_Ultimate_Starter_Kit_C_Code_for_RPi/07_breathingLed/breathingLed.c)

2.2 Compile the program

```
$ gcc breathingLed.c -o breathingLed -lwiringPi
```

2.3 Run the program

```
$ sudo ./breathingLed
```

Python user:

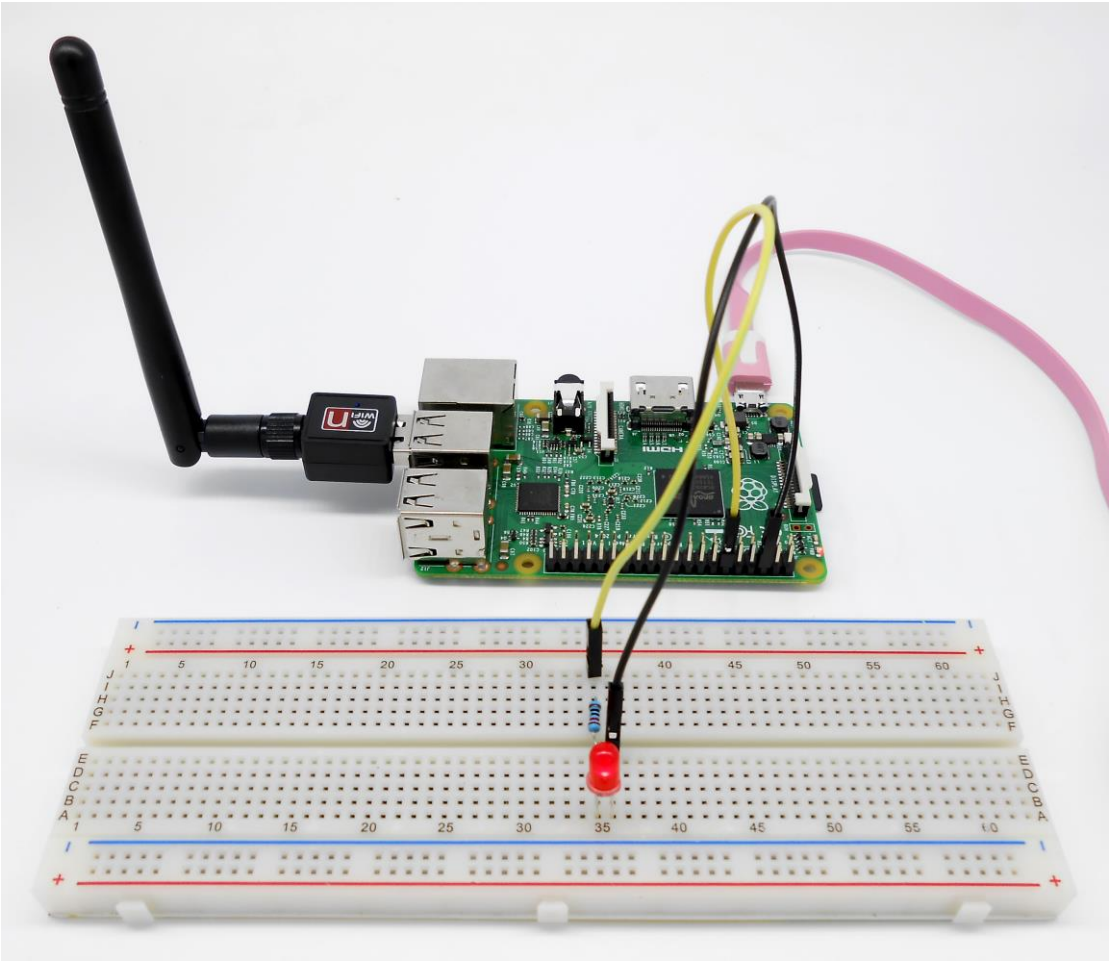
2.1 Edit and save the code with vim or nano.

(Code path: /home/Adeept_Ultimate_Starter_Kit_Python_Code_for_RPi/07_breathingLed.py)

2.2 Run the program

```
$ sudo python 07_breathingLed.py
```

Now, you should see the LED gradually from dark to brighter, and then from brighter to dark, continuing to repeat the process, its rhythm like the animal's breathing.



Summary

By learning this lesson, I believe that you have understood the basic principles of the PWM, and mastered the PWM programming on the Raspberry Pi platform.