Electronic Basics #2: Dimming all kinds of LEDs!?

LED brightness can be controlled using both **analog** and **digital** methods. The main goal is to regulate the current flowing through the LED, as brightness is directly proportional to the current.

1. Importance of Resistor and Bandgap Voltage

- **Resistor:**
    - A resistor is essential in LED circuits to limit the current and prevent LED damage.
    - Ohm's Law (V=IRV = IR) determines the required resistor value.
- **Bandgap Voltage:**
    - The bandgap voltage of an LED is the minimum voltage required for it to emit light.
    - For example, red LEDs have a bandgap voltage of ~1.8V, while blue LEDs need ~3.0V.
    - Exceeding this voltage without current limiting may burn out the LED.

2. Analog LED Brightness Control (Using a Potentiometer)

A **potentiometer (variable resistor)** can be used to adjust LED brightness by varying resistance, which controls the current flowing through the LED.

Circuit Components

- LED
- Resistor (~330Ω to 1kΩ)
- Potentiometer (10kΩ)
- Power supply (e.g., 5V or 9V battery)

Working Principle

- The potentiometer acts as a voltage divider, adjusting the voltage supplied to the LED.
- As resistance changes, the LED brightness varies accordingly.

Basic Circuit Connection

1. Connect one terminal of the potentiometer to **VCC** (positive voltage).
2. Connect the other terminal to **GND** (ground).
3. Connect the wiper (middle pin) of the potentiometer to one end of the LED (through a series resistor).
4. Connect the other end of the LED to **GND**.

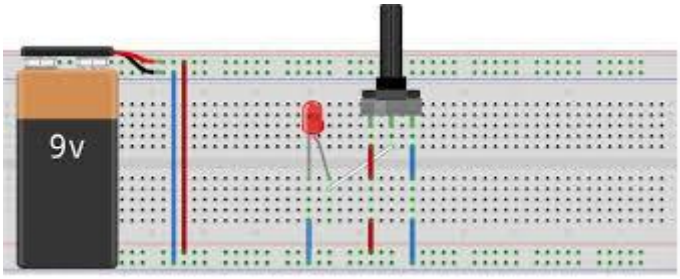

Fig2.1: Controlling and LED brightness with a Potentiometer

3. Digital LED Brightness Control (Using PWM - Arduino & 555 Timer IC)

Method 1: Using PWM with Arduino

Pulse Width Modulation (PWM) is a technique used to control the power delivered to electrical devices by rapidly switching the signal between ON and OFF states. The **duty cycle** (percentage of time the signal is ON) determines the average power supplied.

Key Features of PWM:

- **Efficient Power Control:** Unlike analog dimming (which wastes energy as heat), PWM reduces power consumption.
- **Duty Cycle:** The ratio of ON time to the total cycle time, expressed as a percentage. For example:
    - **0% duty cycle → LED OFF**
    - **50% duty cycle → LED at half brightness**
    - **100% duty cycle → LED fully ON**
- **Frequency:** PWM signals operate at high frequencies (e.g., 500Hz–20kHz) to prevent flickering.

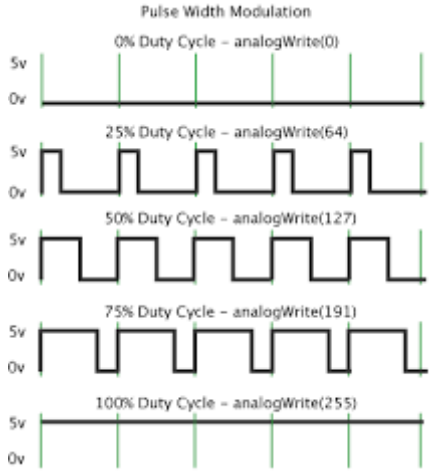

Fig2.2: PWM Duty Cycle

Circuit Components

- LED
- Resistor (220Ω - 1kΩ)
- Arduino board (e.g., Arduino Uno)

Arduino Code for PWM LED Control

```
int ledPin = 9;  // PWM pin

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  for (int brightness = 0; brightness <= 255; brightness += 5) {
    analogWrite(ledPin, brightness);  // Set LED brightness
    delay(30);
  }
  for (int brightness = 255; brightness >= 0; brightness -= 5) {
    analogWrite(ledPin, brightness);
    delay(30);
  }
}
```

Explanation:

- The LED is connected to **PWM pin 9** of Arduino.
- analogWrite(pin, value) sets brightness (0 = OFF, 255 = fully ON).
- The brightness gradually increases and decreases, creating a fading effect.

Method 2: Using 555 Timer IC as a PWM Generator

A **555 Timer IC** can generate a **variable PWM signal** for controlling LED brightness.

Circuit Components

- 555 Timer IC
- LED

- Resistor (1kΩ, 10kΩ)
- Potentiometer (50kΩ)
- Capacitor (0.01µF, 1µF)
- Power supply (5V or 9V)

Circuit Working

- The **555 Timer is configured in astable mode** to generate a PWM signal.
- The **duty cycle is controlled by the potentiometer**, adjusting LED brightness.

Basic Circuit Connections

1. **Pin 1 (GND)** → Connect to ground.
2. **Pin 2 (Trigger)** → Connect to Pin 6.
3. **Pin 3 (Output)** → Connect to LED (via a resistor).
4. **Pin 4 (Reset)** → Connect to VCC.
5. **Pin 5 (Control Voltage)** → Connect to GND via a capacitor (0.01µF).
6. **Pin 6 (Threshold)** → Connect to Pin 2.
7. **Pin 7 (Discharge)** → Connect to resistor & potentiometer combination.
8. **Pin 8 (VCC)** → Connect to 5V/9V power source.

Working Principle

- The **potentiometer adjusts the duty cycle**, varying LED brightness.
- The 555 Timer rapidly turns the LED ON/OFF, and the human eye perceives it as dimming or brightening.

Application

- **Analog control** (using a potentiometer) is simple but wastes power as heat.
- **Digital control** (using PWM with Arduino or 555 Timer) is more efficient, as it rapidly switches the LED ON/OFF instead of dissipating excess energy.
- **For precise brightness control**, **PWM-based methods** are preferred in modern applications.