### Southern New Hampshire University

CS 350: Emerging System Architecture & Technology

### 2-4 Milestone One: Pulse Width Modulation Lab

Christian J. Busca

### Instructor Roland Morales

May 18, 2025

**Abstract**

This report explores the use of pulse width modulation (PWM) for controlling an LED with a Raspberry Pi 4B as part of Milestone One in CS 350. Building upon foundational skills developed in Module One (Southern New Hampshire University, 2025), the project transitioned from basic digital GPIO control to analog-like PWM manipulation using Python (Python Software Foundation, 2023). The objective was to use PWM to vary the LED's brightness and visually demonstrate smooth fade-in and fade-out effects. The report outlines key phases of development, from circuit assembly and software adaptation to experimental observation and final code implementation. It also reflects on perceptual thresholds for frequency and duty cycle changes and evaluates how increment and delay parameters influence transition smoothness. This milestone serves as a practical step toward more advanced embedded system functions involving motor control and actuator responses (Raspberry Pi Foundation, 2024).

**Introduction**

This lab builds upon the foundation laid in Module One, transitioning from basic GPIO control to pulse width modulation (PWM) on a Raspberry Pi 4B. PWM introduces a more nuanced level of control, allowing a digital signal to simulate varying levels of voltage through the modulation of pulse duration (Python Software Foundation, 2023). In embedded systems, such capabilities are essential for applications ranging from LED brightness control to managing motor speeds and actuator responses.

**Developmental Progress**

In Module One, the groundwork was established by setting up the Raspberry Pi hardware and configuring a basic circuit involving an LED and resistor on a solderless breadboard (Southern New Hampshire University, 2025). A simple script using the RPi.GPIO library enabled binary on/off control of the LED. This task familiarized the Raspberry Pi’s GPIO interface, Python scripting, and the embedded Linux environment (Raspberry Pi Foundation, 2024).

The progression into Module Two and Milestone One marked the transition from digital toggling to analog-style output via PWM. I revised the script to utilize the RPi.GPIO.PWM class, enabling modulation of LED brightness by adjusting duty cycles and frequencies (Python Software Foundation, 2023). This involved testing thresholds for perceptible changes in blink rate and brightness, constructing a fade-in and fade-out loop, and ensuring clean program termination with a try/except structure. A demonstration video was recorded as practical evidence of successful implementation.

### Observations and Analysis

At lower frequencies, typically below 30 Hz, the human eye begins to detect the LED blinking rather than perceiving a constant glow. Through experimentation, it was determined that around 10 Hz, blinking becomes clearly visible, and at frequencies near 5 Hz or lower, the blinking is pronounced (Southern New Hampshire University, 2025).

Adjusting the duty cycle independently, it was observed that intensity begins to perceptibly decrease below a 30% duty cycle. The LED appeared notably dim at around 10% to 20% duty cycle levels. These observations support the understanding that duty cycle directly influences perceived brightness, a concept fundamental to PWM (Python Software Foundation, 2023).

The original loop implementation used increments of 5 with a 0.1-second delay between updates. While functional, this configuration produced a somewhat stepped fade effect. A smoother result would be achievable by using smaller increments and shorter sleep intervals, such as increments of 1 or delays of 0.05 seconds, yielding a more continuous transition in brightness (Southern New Hampshire University, 2025).

The PWM frequency for a GPIO line is configured using the function GPIO.PWM(pin, frequency), while the duty cycle can be adjusted dynamically using pwm.ChangeDutyCycle(dutyCycle) during program execution (Python Software Foundation, 2023). These two functions together provide precise control over timing behavior in GPIO outputs.

**Conclusion**

This milestone expanded embedded systems proficiency by advancing from digital switching to PWM. Understanding how duty cycles and frequencies influence output enables more sophisticated interactions with electronic components. The PWM technique serves as a bridge to future milestones involving motor control, sensor feedback, and energy regulation.

**References**

Python Software Foundation. (2023). *RPi.GPIO documentation*. <https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/>Raspberry Pi Foundation. (2024). *Getting started with Raspberry Pi*. <https://www.raspberrypi.com/documentation/computers/getting-started.html>Southern New Hampshire University. (2025). *CS 350 Milestone One PWM Lab Guide*.

**Index**

[**Figure 1.**](https://youtube.com/shorts/xtpExrv8Nug?si=5Ax1zY25SvxTP7_2)