****

**VILNIUS UNIVERSITY  
SIAULIAI ACADEMY**

PROGRAMŲ SISTEMOS BACHELOR STUDY PROGRAMME

Software engineering

**ANNA KUTOVA**

**Programming of Embedded Systems**

**Laboratory work No.3**

**Pulse Width Modulation (PWM)**

Šiauliai, 2025

**Laboratory Work Report**

Table of Contents

[1. The Aim of the Laboratory Work 2](#_Toc192000534)

[2. Variant No and Data 2](#_Toc192000535)

[3. Program Algorithm 3](#_Toc192000536)

[4. Program Body with Comments 3](#_Toc192000537)

[5. Conclusions 4](#_Toc192000538)

# **The Aim of the Laboratory Work**

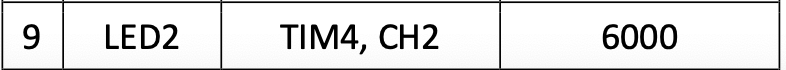
The goal of this laboratory work is to study Pulse Width Modulation (PWM) and its application in controlling LED brightness using an STM32 microcontroller. The experiment focuses on generating a PWM signal with a specified frequency and using a timer interrupt to create a fading effect on an LED.

# **Variant No and Data**

**Variant No:** 9

A black text on a white background

AI-generated content may be incorrect.

****

**Date:** 04/03/2025

# **A diagram of a flowchart AI-generated content may be incorrect.Program Algorithm**

# 

# **Program body**

1. **from** **pyb** **import** Pin, Timer
2. *# LED2 is on TIM4, CH2*
3. p = Pin("LED2")
4. *# Set TIM4 frequency to 6000Hz*
5. tim = Timer(4, freq=6000)
6. *# Configure timer in PWM mode on channel 2*
7. ch = tim.channel(2, Timer.PWM, pin=p)
8. *# Initial duty cycle*
9. brightness = 0
10. step = 5 *# Brightness change step*
11. **def** fade\_led(timer):
12. **global** brightness, step
13. brightness += step
14. **if** brightness >= 100 **or** brightness <= 0:
15. step = -step *# Reverse direction*
16. ch.pulse\_width\_percent(brightness)
17. *# Use timer interrupt to change brightness*
18. fade\_timer = Timer(5) *# Create Timer 5*
19. fade\_timer.init(freq=20, callback=fade\_led) *# 20Hz → period 50ms*
20. **while** True:
21. **pass**

# **Conclusions**

* PWM is a simple and effective way to control LED brightness. By changing the duty cycle, the LED appears brighter or dimmer without extra hardware.
* Timers and interrupts automate brightness control. Instead of manually updating values, the timer interrupt adjusts brightness at regular intervals.
* The fading effect happens by smoothly increasing and decreasing the duty cycle over time, creating a gradual transition.
* MicroPython makes embedded programming easier by providing built-in functions for PWM and timers, reducing code complexity.

This laboratory work demonstrates the practical application of PWM in embedded systems, particularly for LED control.