VILNIUS UNIVERSITY
ŠIAULIAI ACADEMY

BACHELOR PROGRAMME SOFTWARE ENGINEERING

Programming of Embedded Systems

Laboratory work No. 3

Student : Hanna Asiadouskaya

Lecturer: dr. Nerijus Ramanauskas

Šiauliai, 2025

Aim of the laboratory work:

The aim of the practical work is to understand PWM (Pulse Width Modulation), configuration of the timer outputs and interrupts and as an output implement dimming effect, using STM32 Nucleo F756ZG board.

Variant No 2.

No	pin	TIMx, CHx	PWM frequency, Hz
2	LED1	TIM8, CH2	2500

Code and Comments:

Importing modules to control pins and to generate PWM signal.

Then create an output pin for LED1, then create time instance to generate PWM, and configure Channel 2, so we can control LED brightness by changing the pulse width.

Also set fade (brightness level) and indicator that show is it decreasing (true) or increasing (false).

```
1 from pyb import Pin, Timer as pTimer
2 from machine import Timer as mTimer
3
4 led = Pin("LED1", Pin.OUT)
5 tim_ch = pTimer(8, freq=2500)
6 ch = tim_ch.channel(2, pTimer.PWM, pin=led)
7 fade = 0
8 invert = False
```

Here we increase fade by chosen step, and when in reaches 100 it’s invert, and in case with decreasing is vice versa.

```
10 def increase():
11     global fade, invert
12     fade = round((fade + 0.05), 3)
13     if (fade == 100):
14         invert = True
15 
16 def decrease():
17     global fade, invert
18     fade = round((fade - 0.05), 3)
19     if (fade == 0):
20         invert = False
```

Function heartbeat, that use invert indicator, to chose function increase or decrease. And after one of the functions, it's checks is this whole number or not and then it updates the LED brightness. Setting PWM duty cycle from 0 to 100 makes LED fade in and out smoothly.

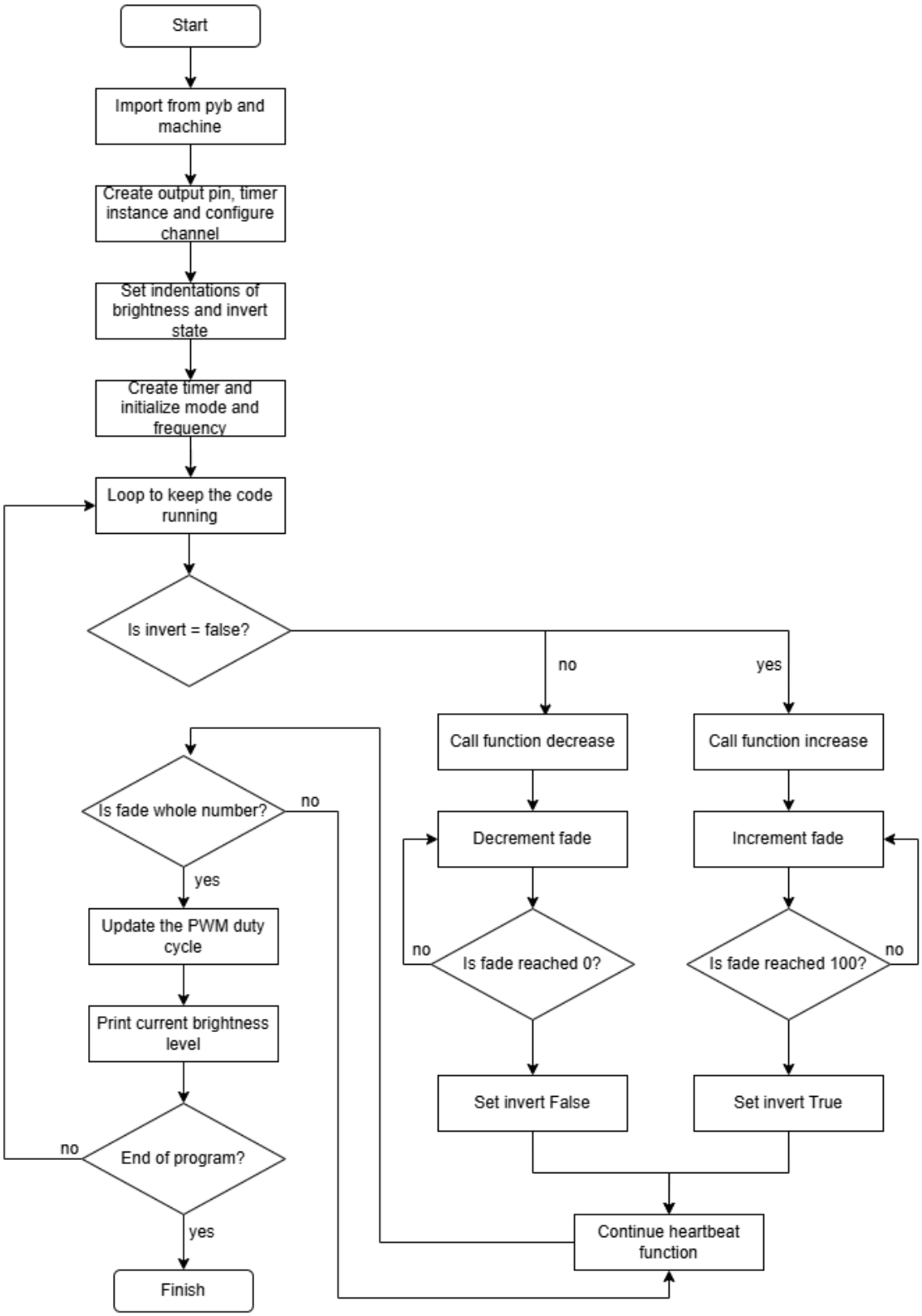
```
22 def heartbeat(t):
23     global fade, invert
24 
25     if invert == False:
26         increase()
27     else:
28         decrease()
29 
30     if (fade % 1 == 0):
31         ch.pulse_width_percent(int(fade))
32         print(fade)
33 
```

Here I create time instance, initialize periodic mode and setting frequency.

And then loop, so the function can work endlessly.

```
35 tim = mTimer()
36 tim.init(mode=mTimer.PERIODIC, freq=2500, callback=heartbeat)
37 
38 
39 while True:
40     pass
```

Algorithm:


Start
Import from pyb and machine
Create output pin, timer instance and configure channel
Set indentations of brightness and invert state
Create timer and initialize mode and frequency
Loop to keep the code running
Is invert = false?
no
yes
Call function decrease
Call function increase
Is fade whole number?
no
Decrement fade
Increment fade
yes
Update the PWM duty cycle
no
Is fade reached 0?
Is fade reached 100?
no
Print current brightness level
Set invert False
Set invert True
no
End of program?
Continue heartbeat function
yes
Finish

Conclusion:

In this study, we successfully practiced programming on the STM32 Nucleo F756ZG microcontroller, explored PWM configuration, configuration of the timer outputs and interrupt and create dimming effect on the board with desired frequency.

Code as a text:

```
from pyb import Pin, Timer as pTimer

from machine import Timer as mTimer


led = Pin("LED1", Pin.OUT)

tim_ch = pTimer(8, freq=2500)

ch = tim_ch.channel(2, pTimer.PWM, pin=led)

fade = 0

invert = False


def increase():

    global fade, invert

    fade = round((fade + 0.05), 3)

    if (fade == 100):

        invert = True


def decrease():

    global fade, invert

    fade = round((fade - 0.05), 3)

    if (fade == 0):

        invert = False


def heartbeat(t):

    global fade, invert


    if invert == False:

        increase()

    else:
```

```
        decrease()


    if (fade % 1 == 0):
        ch.pulse_width_percent(int(fade))
        print(fade)



tim = mTimer()
tim.init(mode=mTimer.PERIODIC, freq=2500, callback=heartbeat)



while True:
    pass
```