EE 14 Lab 5: More blinky blink!

Lab report due a week after your lab session (24-28 February 2025)

1 Introduction

Back in lab 2, you wrote code to control an LED. In this lab, you're going to take that to the next level by controlling a red-green-blue (RGB) LED and varying the brightness of each color to achieve any color you want.

To do this, we'll use the STM32's built-in timer hardware to generate PWM signals. Our chip includes several timers which can be configured to count pulses, time intervals, and generate square waves. The CPU can configure one of the timers, and then go off to do other things while the timer hardware takes care of generating the signal with very precise timing.

We're also going to start building a library of useful functions to control the STM32 hardware, so that in the future you don't have to write ten lines of register-level code every time you want to do something simple.

After successfully completing this lab, you should be able to:

- Calculate the parameters to configure PWM frequency and duty cycle
- Mix RGB colors to produce other colors
- Write code as part of a "hardware abstraction layer" (i.e., a driver for a microcontroller peripheral which has a simpler interface than the individual registers)

Documentation focus: Code documentation — you will write detailed comments explaining the hardware abstraction layer functions that you write. See the links on the course website about writing good code comments!

2 Prelab

Look at the Nucleo L432KC pin diagram (link on the course website), and pick three pins that you'd like to use for the red, green, and blue channels of your LED. The pins must be PWM-capable, and ideally on the same timer (probably `TIM1` or `TIM2`; `TIM15` has two channels, and `TIM16` has just one).

Note that the channels with an `N` at the end are inverted versions and cannot be controlled independently. For example `TIM1_1N` is the inverted (logical NOT) of `TIM1_1`.

P1: Which pins did you choose? What timer channels are they on?

P2: Look at the datasheet for the RGB LED posted on the course website. Sketch a diagram showing how you could connect it to your Nucleo board in order to drive the LEDs. What value would you need to set the pin to in order to turn the LED on?

3 In lab

Read the documenation in the textbook (section 16, especially 16.3) about what PWM is, and/or look at the resources on the course website, and talk to the TAs!

L1: By default, the STM32L432KC clock runs at 4 MHz. What should the prescaler and auto-reload register be set to in order to generate a 1 kHz signal?

L2: What should the prescaler and auto-reload register be set to in order to generate a 10 Hz signal?

L3: Generalize your answers to the previous questions, and write a formula / algorithm to set the prescaler and auto-reload register values to achieve a target frequency F.

L4: Suppose the duty cycle is specified using an integer in the range 0-1023 (full off - full on). If the desired duty cycle value is 767 (75%), and the reload register is currently set to 640, what should the compare register be set to?

L5: Generalize your answers to the previous question, and write a formula / algorithm to set the compare register based on a desired duty cycle, given in the range 0-1023.

L6: Create an empty PlatformIO project with the same settings as usual:

- **Board**: ST Nucleo L432KC
- **Framework**: CMSIS

L7: Download the framework starter files from the course website (`gpio.c`, `timer.c`, `ee14lib.h`) and add them to your project.

L8: Complete the code in `timer.c` to implement the `timer_config_pwm()` and `timer_config_channel_pwm()` functions, using the formulas you created. Write comments documenting the functions!

L9: Connect an RGB LED to your Nucleo, and write code in `main()` to use the timer functions to drive one channel of the LED. You should be able to make it fade by adjusting the duty cycle. Can you make it do that "breathing" fade pattern like when a computer is hibernating?

L10: Wire up the other two channels, and add code to vary the duty cycle of all three channels at different rates or in sequence. If you do it well, you should be able to achieve an entire rainbow of colors!

4 What to turn in

Your code is your lab report for this week. Submit all of your code files (.c and .h) to Gradescope, along with a file named `README.txt` with your name and the answers to the usual reflection questions:

- What do you understand now, that you didn't before the lab?
- What are you confused or curious about? (And no, you're still not allowed to say "nothing".)
- How long did it take you to complete this lab?

You must comment the `timers.c` file extensively, including detailed comments at the top of each function explaining what the function does, what each parameter is, and what the return value is. See `gpio.c` and the links on the course website for examples.