Experiment 4: SysTick & PWM

Time: **3** hrs **Weight:** 5%
Marks Available: 15 out of 15 (Prelim=1, Experiment=14)

Aim

To use PWM to vary the flashing rate on the D1 LED and then to vary the intensity of Blue LED output on F2. The PWM concept will be demonstrated using a SysTick interrupt to manually create the PWM signal and then the Timer subsystem is used to generate the PWM.

Introduction and Background

Timers can be used to not only periodically trigger sampling or to read GPIO inputs, but can be used to generate PWM outputs. These PWM signals can be used to drive DC motors at different speeds, create sound, or vary the intensity of lights such as the LEDs on-board the Tiva. This can be achieved by varying the match value of the subsystem, which changes the duty cycle of the PWM output. In this experiment, the duty cycle will be varied between 0 and 100% of the reload value, according to the value entered from the terminal (Serial).

On board LEDs will be used as shown here:

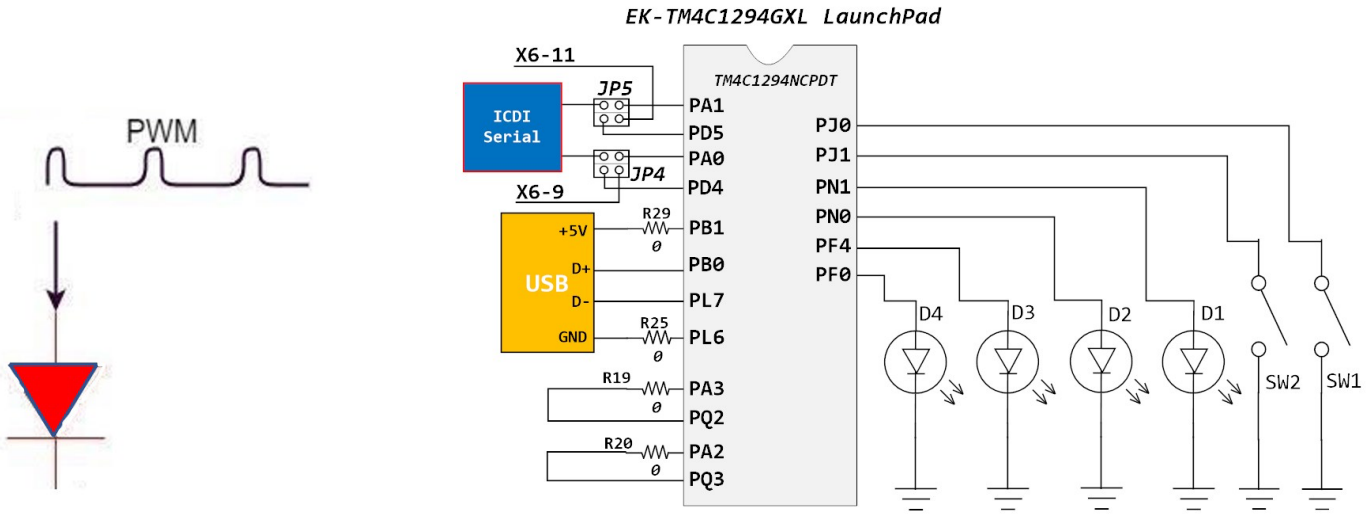


Figure 1: Use of LEDs controlled with PWM and the TM4C1294XL Development Board

PWM can be achieved using counters or timers as shown in Figure 1. The waveform repeats at a given frequency which is set by the *period*. Within the waveform, the *duty cycle* is determined by setting the output High until the *Match* is met and then the output is set Low.

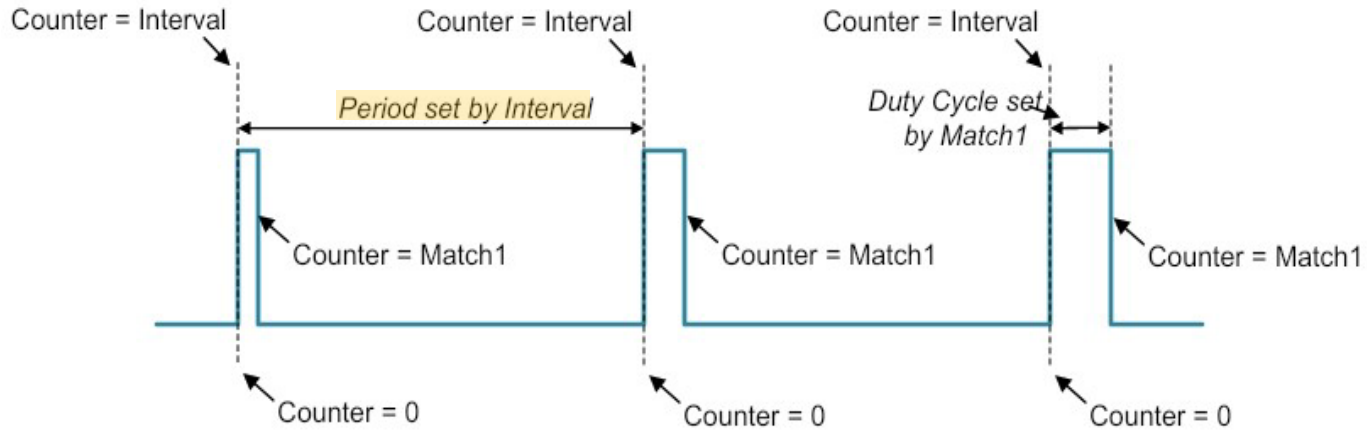


Figure 1: Use of LEDs controlled with PWM

The equation for calculating the Duty Cycle is :

$$Duty\ Cycle\ \% = \frac{t_{HIGH}}{T_{Waveform}} \times\ 100$$

Where

t_{HIGH} is the duration of the HIGH Pulse

$T_{Waveform}$ is the duration of the repeating waveform (Period)

Example PWM Calculations

- A 50 % duty cycle for a 1 HZ waveform

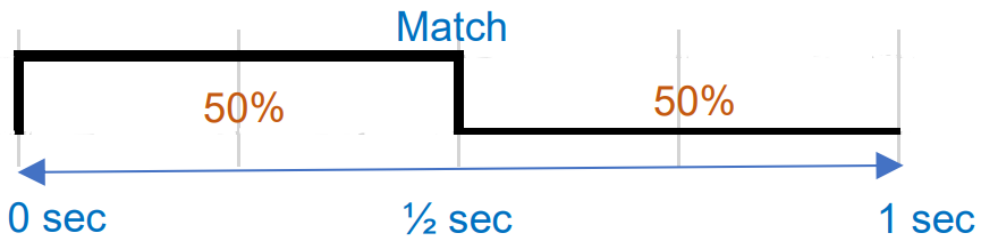
  1 Hz has a period of 1 second

  50% is 50/100 = ½ = T_{HIGH}/1sec

  Therefore T_{HIGH} is ½ seconds

  Assuming a ¼ second interrupt then ½ second is 2 interrupts.

  (Gray lines on graph below indicate the ¼ second interrupt)



- A 25 % duty cycle for a 1 HZ waveform

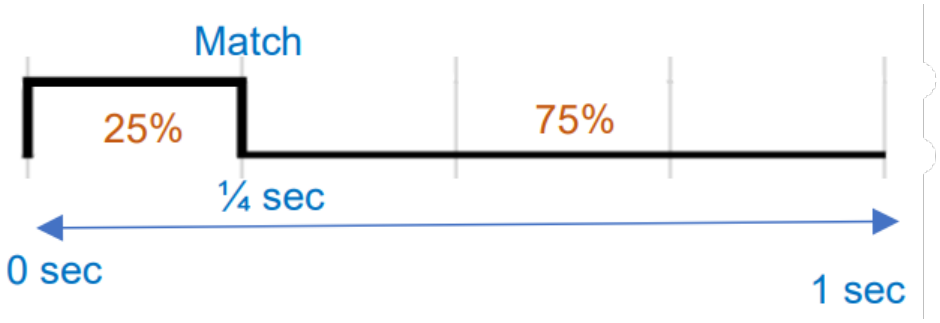
  1 Hz has a period of 1 second

  25% is 25/100 = ¼ = T_{HIGH}/1sec

  Therefore T_{HIGH} is ¼ seconds

  Assuming a ¼ second interrupt then ¼ seconds is 1 interrupt only.

  (Gray lines on graph below indicate the ¼ second interrupt)



Preliminary — 1 mark

Prior to this laboratory, complete the following:

- Tabulate the registers and values required to setup the SysTick for a counter that repeats at 1KHz using the format given below. Note that the system clock is 16MHz.
- Use the same notation that can be found in #include <TM4C129.h> and the Register Reference from the Course Website
  Hint: What is the period of 1KHz signal and how many counts does it represent?

Register	Operator	Mask Value	Code line

Experiment

14 marks

The template provided on L@G is setup for the Tiva microcontroller and Keil uVision 5.
For this experiment, you will again make use of Tera Term (or Putty), and use the following communication parameters:
Baud rate = 115200, 8 bit data, no parity, 1 stop, no flow control. (115200,8,N,1)

You will need to submit your code at the end of the session.

Before starting, do the following:

- Download the template from learning@griffith.
- Open the project, and tera term.
- A test message will be displayed using the

In this experiment, you will use a PWM signal to flash the D2 LED (PN0) at a given duty cycle. The duty cycle of the timer PWM output on the D2 LED will result in different intensity of the LED being visible.

You MUST have a separate project for each part. You will be marked on programs that you can demonstrate.

- Once you have finished with a Part ,zip up the folder containing the project (see the Submission section near the end of this document for instructions on how to zip your files.
- Next copy your folder to a new folder with a new name eg copy folder L4PA to L4PB. You can then work on the project in the L4PB folder.
- You can rename the **.uvprojx** and **.uvoptx** file to whatever you want to call the new project.
- You can then open your new project and edit it. You will need these skills for when you do your project.

A. Using SysTick to Flash the PN0 (D2) LED — 3 marks

Write SysTick code to produce an interrupt every millisecond. Note that the system clock is **16MHz**.

Use this interrupt to produce a 1Hz 50% duty cycle waveform on Port N Pin 0 (D2 LED). The LED should flash on for ½ second and off for ½ second. This is equivalent to manually creating a PWM waveform. See Figure 1 to help you understand the concept.

Basic Structure:

Turn Led On initially
Repeat:
    At ½ second worth of 1ms interrupts turn LED OFF
    At 1 second worth of 1ms interrupts turn LED ON

***Hint*:** Think about counting interrupts. i.e. how many milliseconds corresponds to 1 second

Do NOT use the supplied delay function to produce the delay – you **must** use the SysTick interrupt to give the 1 second

Expected Output: The LED should flash ON for ½ second and OFF for ½ second.

B. When Duty Calls: 2 marks

Modify Part A so that the duty cycle can be changed to a value between 0 and 100% duty cycle using a function. Refer to Figure 1 and the accompanying text to see how to set the Duty Cycle.

- You should **create** a function called **Set_Duty_Cycle (percent)**. Which will take a parameter that is a percentage between 0 and 100 and set the duty cycle of the toggle accordingly. The function should not accept a value outside of these ranges.
- Use the supplied function called **readDec()** which reads a decimal number typed in from the terminal and converts it to an integer. Eg typing "12<enter>" returns an integer which holds the value 12
- Test your function by watching how long the LED stays on as the duty cycle changes.

Hint: Think about the relationship between your counter value in Part A and the required duty cycle (Fig 2).

Expected Output: The LED should flash ON for the Duty cycle given eg 50% should blink the LED with ½ second ON and ½ second OFF, 25% should blink the LED with ¼ second ON and 3/4 second OFF, 75% should blink the LED with ¾ second ON and ¼ second off, 0% OFF the whole time, and 100% ON the whole time. Basicially the LED should be ON for the DUTY Cycle fraction of 1 second.

C. Blinded by the Light 3 marks

Set the SysTick interrupt frequency to be **10KHz** and set the period of the waveform as shown in Figure 1 to be 100Hz. Modify Part B so that the PN0 LED (D2) sweeps in different intensities from fully OFF to fully ON. The PN0 LED will appear to change brightness.

Hint: You may find a delay function to be helpful.
Hint: Think about how many interrupts occur per period of a 100Hz waveform. How does this relate to your percentage duty cycle?

Expected Output: The brightness of the D2 LED will change from OFF to fully ON with different levels of brightness in between. The process of the brightness changing from OFF to ON will take 4 seconds.

D. All together now 3 marks

Modify Part C so that the intensities of the D1, D2, D3, and D4 LEDs all sweep intensity at the same rate at the same time.

Hint: You may find a delay function to be helpful.

Expected Output: Sweeping the Duty Cycle from 0% to 100% will cause all the LEDs to change their brightness in unison.

E. Pressing on 3 marks

Modify Part D so that pressing SW1 increases the intensity of the LEDs by 10% and pressing SW2 decreases the intensity of the LEDs by 10%. The value of the intensity in precent (5) should be written to the Terminal using the supplied serial functions. At all times the intensity must remain within the 0 to 100% range where 0% is OFF and 100% is ON. You should start the program with the LEDs OFF ie at 0%.

Note 1: The Switches will require a Pull Up resistor. Think carefully about the sense of the Switches are they High when the switch is pressed or are they LOW when the switch is pressed. It will help you to think about the purpose of the Pull Up resistor.

Note 2: The switches are not debounced.

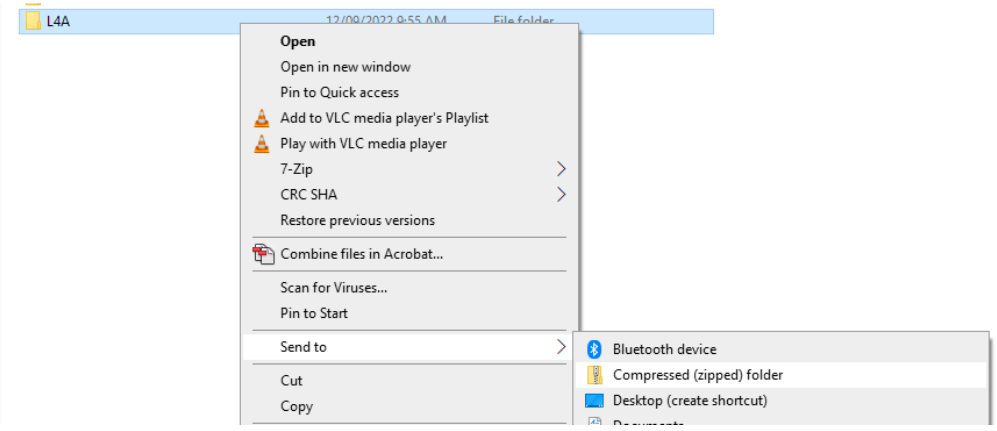
***Hint*:** You may find a delay function useful to help with the debouncing of the switches.

Expected Output: When the SW1 button is pressed the brightness of all the LEDs should increase and when the SW2 button is pressed the brightness of all the LEDs should decrease. The value of the intensity of the LEDs should be written to the Terminal.
Eg if the intensity was 50

- SW1 pressed the brightness will be higher and the Terminal will display 60%
- SW2 pressed the brightness will be lower and the Terminal will display 40%
- The range of the intensity is locked between 0% and 100% which will be evident from the terminal display.

Experiment Submission

Before leaving the lab you need to submit all code you have written for each part of the experiment. This is done by uploading the text file/s containing your code to L@G. The projects can be zipped up by selecting the folder containing the project. Right Clicking on the folder, Selecting “Send To” and then selecting :Compressed zip folder”



Ensure you have shown your demonstrator the correct operation of each completed part of this lab, and have completed the online submission before leaving the laboratory.

Appendix

Appendix 1: Functions provided in the library file

The functions provide in the library file can be found in the header file Lib4.h. Please read the header file to determine the names and the arguments for the required functions.

```
#include "Lib4.h"
```

Lib4.h can be found under “Source Code 1” in the Project Pane of the Template. Double click on Lib4.h to get its contents to appear in the editor.

Note: This is a common approach to find out the functions and the defines for a Library.

Appendix 2: Tiva TM4C1294XL Development Board Connected Components