**FreeRTOS Exercises**

**LAB\_ONE:**

This lab covers the point of Scheduling based on the priority. The main points is what happened when change the task priority?

**LAB\_TWO:**

This lab covers the point of using interrupt with RTOS and using the semaphore to synchronize.

**LAB\_THREE:**

This lab covers how to synchronize between the tasks using the message queue. The main point is how to create, send and receive from the message queue.

**Ex.1:**

Using FreeRTOS, write a program using two tasks:

Task1: blinks a LED every 200 msec

Task2: blinks a LED every 1000 msec

The two tasks run in parallel.

hint: You will need those three APIs:

xTaskCreate()

vTaskStartScheduler()

vTaskDelay()

**Ex.2:**

Using FreeRTOS, repeat Ex.1 but using the same task function body. Try to use the task argument to change the behavior of each task.

**Ex.3:**

Using FreeRTOS, write a program using two tasks to communicate with each other using a binary semaphore:

First task: Scans a push button every 200msec and gives the semaphore when it's pressed.

Second Task: Waits for that semaphore and toggle a LED once when getting it.

**Ex.4:**

Using FreeRTOS, write a program using one task and one ISR (UART).

The (UART Rec ISR) receives a byte then send it via a message queue to a task that is blocked waiting for this byte. This task verifies if this byte is a number (0 to 9), it displays it on seven segment. If not, it displays (-) sign on seven segment.

Hint: you may need to use the (virtual Terminal) to send this byte.