Ain Shams University Faculty of Engineering CSE345/CSE347:
Embedded Systems

Lab 8 - Mutex in FreeRTOS

Lab Objective:

- In this Lab we will cover the following points.
  - Mutex and how they are implemented
  - Difference between mutex and semaphores
  - Uses of mutex and semaphores
  - Use case 1 (Main mission)
  - Software interrupts Vs Hardware interrupts
  - Use case 2 (VERY LARGE BONUS !!)

Lab Mission:

- In this lab we aim to know the difference between Mutex and Semaphores and how to use Mutex. Mutex

1) Create an Init Task to Initizalize the UART0 , 1 push button and led.

```
void InitTask(void *)
{
        ....
}
```

P.S : Use the following to unlock PORTF using the Tivaware (if needed)

```
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "inc/hw_gpio.h"

HWREG(GPIO_PORTF_BASE+GPIO_O_LOCK) = GPIO_LOCK_KEY;
HWREG(GPIO_PORTF_BASE+GPIO_O_CR) |= 0x01;
```

2) Configure the push button to generate interrupt when pressed
3) Create a Mutex using the FreeRTOS API " xSemaphoreCreateMutex "
4) Create a Binary Semaphore
5) Create a continuous Task called " CounterTask " that prints "This is the CounterTask " then counts from 0 – 10 and prints each count on the console.

6) Create a periodic Task with higher priority called " LedTogglerTask " which is unlocked by Binary Semaphore Given from ISR of the push button.
   The Task should Write on UART " This is LedToggler Task " when given the semaphore then toggles the Led then sleeps for 500 ms.
7) The ISR should give the binary semaphore.
8) Expected behavior:
   This is the Counter task
   0
   1
   2
   3
   4
   5
   6
   7
   8
   9
   10
   This is the LedToggler task
   0
   1
   2
   ..

Super Bonus:
Configure a Software Interrupt to produce the above behavior without Mutex with all the same givens.