Ismael Elsharkawi 900173030

Embedded Systems Lab
Lab 2 Report

Experiment 1:

```
#include "TM4C123GH6PM.h"
void delayMs(int n);
unsigned int LED = 0;
unsigned int masks[] = {0x08, 0x06};
int main(void)
{
 SYSCTL->RCGCGPIO |= 0x20; /* enable clock to PORTF */
 /* PORTF0 has special function, need to unlock to modify */
 GPIOF->LOCK = 0x4C4F434B; /* unlock commit register */
 GPIOF->CR = 0x01; /* make PORTF0 configurable */
 GPIOF->LOCK = 0; /* lock commit register */
 /* configure PORTF for switch input and LED output */
 GPIOF->DIR &= ~0x11; /* make PORTF4,0 input for switch */
 GPIOF->DIR |= 0x0E; /* make PORTF3, 2, 1 output for LEDs */
 GPIOF->DEN |= 0x1F; /* make PORTF4-0 digital pins */
 GPIOF->PUR |= 0x11; /* enable pull up for PORTF4,0 */

 /* configure PORTF4, 0 for falling edge trigger interrupt */
 GPIOF->IS &= ~0x11; /* make bit 4, 0 edge sensitive */
 GPIOF->IBE &= ~0x11; /* trigger is controlled by IEV */
 GPIOF->IEV &= ~0x11; /* falling edge trigger */
 GPIOF->ICR |= 0x11; /* clear any prior interrupt */
 GPIOF->IM |= 0x11; /* unmask interrupt for PF4,PF0 */
/* enable interrupt in NVIC and set priority to 3 */
 NVIC->IP[30] = 3 << 5; /* set interrupt priority to 3 */
 NVIC->ISER[0] |= 0x40000000; /* enable IRQ30 (D30 of ISER[0]) */
 __enable_irq(); /* global enable IRQs */

 /* toggle the green LED (PF3) continuously */
 while(1)
 {
 GPIOF->DATA |= masks[LED];
 delayMs(500);
 GPIOF->DATA &= ~ 0x0e; /* turn all LEDs off */
 delayMs(500);
 }
```

```
}
/* SW1 is connected to PF4 pin, SW2 is connected to PF0. */
/* Both of them trigger PORTF interrupt */
void GPIOF_Handler(void)
{
 volatile int readback;
 LED = ! LED;
 GPIOF->ICR |= 0x11; /* clear the interrupt flag before return */
 readback = GPIOF->ICR; /* a read to force clearing of interrupt flag */
}
/* delay n milliseconds (16 MHz CPU clock) */
void delayMs(int n)
{
SysTick->LOAD = 15999*n;
SysTick->CTRL = 0x5; /*Enable the timer and choose sysclk */
while((SysTick->CTRL & 0x10000) == 0) /*wait till Count flag is set */
{ }
SysTick->CTRL = 0; /*Stop the timer (Enable = 0) */
}
```

Experiment 2:

```
#include "TM4C123GH6PM.h"
int main (void)
{
 /* enable clock to GPIOF at clock gating control register */
 SYSCTL->RCGCGPIO |= 0x20;
 /* enable the GPIO pins for the LED (PF3, 2, 1) as output */
 GPIOF->DIR = 0x0e;
 /* enable the GPIO pins for digital function */
 GPIOF->DEN = 0x0e;

 /* Configure SysTick */
 SysTick->LOAD = 16000000-1; /* reload with number of clocks per second */
 SysTick->CTRL = 7; /* enable SysTick interrupt, use system clock */
 __enable_irq(); /* global enable interrupt */

 while (1)
 {
 }
}
void SysTick_Handler(void)
```

```
{
 GPIOF->DATA ^= 2; /* toggle the red LED */
}
```

Experiment 3:

```
#include "TM4C123GH6PM.h"
void delayMs(int n);
unsigned int LED = 0;
unsigned int masks[] = {0x08, 0x06};
int enable =0;
int main(void)
{
 SYSCTL->RCGCGPIO |= 0x20; /* enable clock to PORTF */
 /* PORTF0 has special function, need to unlock to modify */
 GPIOF->LOCK = 0x4C4F434B; /* unlock commit register */
 GPIOF->CR = 0x01; /* make PORTF0 configurable */
 GPIOF->LOCK = 0; /* lock commit register */
 /* configure PORTF for switch input and LED output */
 GPIOF->DIR &= ~0x11; /* make PORTF4,0 input for switch */
 GPIOF->DIR |= 0x0E; /* make PORTF3, 2, 1 output for LEDs */
 GPIOF->DEN |= 0x1F; /* make PORTF4-0 digital pins */
 GPIOF->PUR |= 0x11; /* enable pull up for PORTF4,0 */

 /* configure PORTF4, 0 for falling edge trigger interrupt */
 GPIOF->IS &= ~0x11; /* make bit 4, 0 edge sensitive */
 GPIOF->IBE &= ~0x11; /* trigger is controlled by IEV */
 GPIOF->IEV &= ~0x11; /* falling edge trigger */
 GPIOF->ICR |= 0x11; /* clear any prior interrupt */
 GPIOF->IM |= 0x11; /* unmask interrupt for PF4,PF0 */
        /* enable interrupt in NVIC and set priority to 3 */
 NVIC->IP[30] = 3 << 5; /* set interrupt priority to 3 */
 NVIC->ISER[0] |= 0x40000000; /* enable IRQ30 (D30 of ISER[0]) */

 SysTick->LOAD = 16000000-1; /* reload with number of clocks per second */
 SysTick->CTRL = 7; /* enable SysTick interrupt, use system clock */

 __enable_irq(); /* global enable IRQs */
```

```
 /* toggle the green LED (PF3) continuously */
 while(1)
 {
 //GPIOF->DATA |= masks[LED];
 //delayMs(500);
 //GPIOF->DATA &= ~ 0x0e; /* turn all LEDs off */
 //delayMs(500);
 }
}
/* SW1 is connected to PF4 pin, SW2 is connected to PF0. */
/* Both of them trigger PORTF interrupt */
void GPIOF_Handler(void)
{
 volatile int readback;
 LED = ! LED;
 GPIOF->ICR |= 0x11; /* clear the interrupt flag before return */
 readback = GPIOF->ICR; /* a read to force clearing of interrupt flag */
}
/* delay n milliseconds (16 MHz CPU clock) */
void SysTick_Handler(void)
{
 if(enable ==0) GPIOF->DATA |= masks[LED];
          else GPIOF->DATA &= ~ 0x0e; /* turn all LEDs off */
          enable = !enable;

}
```

Question 1:

https://www.youtube.com/watch?v=w5sqydlEG4M

```
#include "TM4C123GH6PM.h"
void delayMs(int n);
unsigned int LED = 1;
unsigned int masks[] = {0x0e, 0x06};
int enable =0;
int main(void)
{
 SYSCTL->RCGCGPIO |= 0x20; /* enable clock to PORTF */
 /* PORTF0 has special function, need to unlock to modify */
 GPIOF->LOCK = 0x4C4F434B; /* unlock commit register */
 GPIOF->CR = 0x01; /* make PORTF0 configurable */
```

```
 GPIOF->LOCK = 0; /* lock commit register */
 /* configure PORTF for switch input and LED output */
 GPIOF->DIR &= ~0x11; /* make PORTF4,0 input for switch */
 GPIOF->DIR |= 0x0E; /* make PORTF3, 2, 1 output for LEDs */
 GPIOF->DEN |= 0x1F; /* make PORTF4-0 digital pins */
 GPIOF->PUR |= 0x11; /* enable pull up for PORTF4,0 */

 /* configure PORTF4, 0 for falling edge trigger interrupt */
 GPIOF->IS &= ~0x11; /* make bit 4, 0 edge sensitive */
 GPIOF->IBE |= ~0x11; /* trigger is controlled by IEV */

 //GPIOF->IEV &= ~0x11; /* falling edge trigger */



 GPIOF->ICR |= 0x11; /* clear any prior interrupt */
 GPIOF->IM |= 0x11; /* unmask interrupt for PF4,PF0 */
         /* enable interrupt in NVIC and set priority to 3 */
 NVIC->IP[30] = 3 << 5; /* set interrupt priority to 3 */
 NVIC->ISER[0] |= 0x40000000; /* enable IRQ30 (D30 of ISER[0]) */

 SysTick->LOAD = 16000000/4-1; /* reload with number of clocks per second */
 SysTick->CTRL = 7; /* enable SysTick interrupt, use system clock */

 __enable_irq(); /* global enable IRQs */

 /* toggle the green LED (PF3) continuously */
 while(1)
 {

 }
}
/* SW1 is connected to PF4 pin, SW2 is connected to PF0. */
/* Both of them trigger PORTF interrupt */
void GPIOF_Handler(void)
{
 volatile int readback;
 LED = ! LED;
 GPIOF->ICR |= 0x11; /* clear the interrupt flag before return */
 readback = GPIOF->ICR; /* a read to force clearing of interrupt flag */
}
```

```
/* delay n milliseconds (16 MHz CPU clock) */
void SysTick_Handler(void)
{
 if(enable ==0) GPIOF->DATA |= masks[LED];
        else GPIOF->DATA &= ~ 0x0e; /* turn all LEDs off */
        enable = !enable;

}
```

Question 2:

https://youtu.be/F7mu8OsPTTc

```
#include "TM4C123GH6PM.h"
void delayMs(int n);
unsigned int LED = 1;
unsigned int masks[] = { 0x128};
int enable =0;
double freq=1.5;
int main(void)
{

 SYSCTL->RCGCGPIO |= 0x20; /* enable clock to PORTF */
 /* PORTF0 has special function, need to unlock to modify */
 GPIOF->LOCK = 0x4C4F434B; /* unlock commit register */
 GPIOF->CR = 0x01; /* make PORTF0 configurable */
 GPIOF->LOCK = 0; /* lock commit register */
 /* configure PORTF for switch input and LED output */
 GPIOF->DIR &= ~0x11; /* make PORTF4,0 input for switch */
 GPIOF->DIR |= 0x0E; /* make PORTF3, 2, 1 output for LEDs */
 GPIOF->DEN |= 0x1F; /* make PORTF4-0 digital pins */
 GPIOF->PUR |= 0x11; /* enable pull up for PORTF4,0 */

 /* configure PORTF4, 0 for falling edge trigger interrupt */
 GPIOF->IS &= ~0x11; /* make bit 4, 0 edge sensitive */
 GPIOF->IBE &= ~0x11; /* trigger is controlled by IEV */
 GPIOF->IEV &= ~0x11; /* falling edge trigger */
        SYSCTL->RCGCGPIO |= 0x04;
        GPIOC->DIR = 0xff;
        GPIOC->DEN = 0xff;
```

```
GPIOF->ICR |= 0x11; /* clear any prior interrupt */
GPIOF->IM |= 0x11; /* unmask interrupt for PF4,PF0 */
        /* enable interrupt in NVIC and set priority to 3 */
NVIC->IP[30] = 3 << 5; /* set interrupt priority to 3 */
NVIC->ISER[0] |= 0x40000000; /* enable IRQ30 (D30 of ISER[0]) */

SysTick->LOAD = (int)(8000000/freq-1); /* reload with number of clocks per second */
SysTick->CTRL = 7; /* enable SysTick interrupt, use system clock */

__enable_irq(); /* global enable IRQs */

/* toggle the green LED (PF3) continuously */
while(1)
{

}
}
/* SW1 is connected to PF4 pin, SW2 is connected to PF0. */
/* Both of them trigger PORTF interrupt */
void GPIOF_Handler(void)
{
         volatile int readback;
        if((GPIOF->DATA & 0x10)==0){
                freq +=0.3;
                SysTick->LOAD = (int)(8000000/freq-1);
                SysTick->CTRL = 7;
        }

        if((GPIOF->DATA & 0x01)==0){
                freq -=0.3;
                if(freq<=0){
                        freq=0;
                        SysTick->CTRL = 0;
                }else{
                        SysTick->LOAD = (int)(8000000/freq-1);
                        SysTick->CTRL = 7;              }

        }

GPIOF->ICR |= 0x11; /* clear the interrupt flag before return */
```

```
 readback = GPIOF->ICR; /* a read to force clearing of interrupt flag */
}
/* delay n milliseconds (16 MHz CPU clock) */
void SysTick_Handler(void)
{
 if(enable ==0) GPIOC->DATA |= 0x128;
          else GPIOC->DATA =  0x0; /* turn all LEDs off */
          enable = !enable;

}
```