

# LAB 03 – TASK 01: WAVEFORM WITH TIMER

*Authors: KUSHAL MOHTA (210020023)*

*SINGHI AYUSH (210020048)*

**Aim:** To create a waveform of 1kHz frequency with 20% duty cycle using polled timer (Systick).

**Procedure:** Timer is initialised to act in line with CPU clock and counts till condition is met following which LED is toggled at appropriate times.

**Theory:** Systick register is initialised and bits are set to represent initialisation active and mode set to CPU clock. The count occurs backward and on zero, it breaks condition.

## **Code:**

```
#include <stdint.h>

#include <stdbool.h>

#include "tm4c123gh6pm.h"

#define STCTRL *((volatile long *) 0xE000E010)
#define STRELOAD *((volatile long*) 0xE000E014)
#define STCURRENT *((volatile long *) 0xE000E018)

// #define ENABLE (1 << 0) // bit 0 set .... ENABLES SYSTICK
// #define CLKINT (1 << 2) // bit 2 set .... USE SYSTEM CLOCK FOR SYSTICK
//
// #define COUNT_FLAG (1 << 16)
#define COUNT_FLAG (1 << 16) // bit 16 of CSR automatically set to 1
// when timer expires
#define ENABLE (1 << 0) // bit 0 of CSR to enable the timer
#define CLKINT (1 << 2) // bit 2 of CSR to specify CPU clock
```

```
#define CLOCK_MHZ 16
```

```
void delay(int count){  
    STRELOAD = count*CLOCK_MHZ*100;  
    STCTRL |= (CLKINT | ENABLE);  
  
    while((STCTRL & COUNT_FLAG) == 0){  
        ;// nuffin do  
    }  
  
    STCTRL = 0;  
    return;  
}
```

```
int main(void)  
{  
    SYSCCTL_RCGCGPIO_R = 0x20; // This enables clock for the GPIO Port F Register  
    GPIO_PORTF_LOCK_R = 0x4C4F434B;  
    GPIO_PORTF_CR_R = 0xFF;  
    GPIO_PORTF_DIR_R = 0x0E;  
    GPIO_PORTF_PUR_R = 0x11;  
    GPIO_PORTF_DEN_R = 0x1F;  
  
    while(1){  
        delay(2);  
        GPIO_PORTF_DATA_R = 0x02;  
        delay(8);  
        GPIO_PORTF_DATA_R = 0x00;  
    }
```

```
    return 0;
}
```

\*\*\*\*\*-----\*\*\*\*\*-----\*\*\*\*\*

**Output:** Successfully created a RED LED that switches with a duty cycle of 20% on a 1kHz waveform.

### **Results/Observations:**

The SysTick register is 32 bits and for each mode reference and toggle, it is registered of 4 bits. SysTick timers count in reverse and the condition on count can be equated to a JNB condition. The STRELOAD command repeats the assigned timer value. As always, all calculations are done with the clock frequency at 16MHz.