# Lab-3 Group -4 report

(210020050,210020007)

## Aim:

- Use the system timer (Systick) in Polled Mode to generate a waveform with f = 1kHz and duty cycle = 20%

## Approach:

To achieve a 1kHz waveform:

- The period T is the inverse of the frequency f, which is:
  - The high time for the 20% duty cycle is 0.2ms.
  - The low time for the 80% duty cycle is 0.8ms.
- Using the SysTick timer, we can generate these delays to toggle a GPIO pin and create the desired waveform.
- Procedure:
  - **SysTick Timer Configuration:**
    - The SysTick timer is configured to run with the internal system clock, which is 16MHz on the Tiva C board.
    - The reload value for the SysTick timer is calculated based on the number of clock cycles needed to achieve the desired delay.
      - For 0.2ms high time: Reload Value=0.2×16000=3200 = 0.2 x 16000 = 3200=0.2×16000=3200
      - For 0.8ms low time: Reload Value=0.8×16000=12800 = 0.8 x 16000 = 12800=0.8×16000=12800

Main loop:

```
while (1)

{

  GPIO_PORTF_DATA_R = 0x02;

  Delay(200);   // In micro sceonds

  GPIO_PORTF_DATA_R = 0x00;

  Delay(800);  // In micro sceonds

}
```

- **Delay Function:** The Delay function uses the SysTick timer to generate a microsecond delay. The reload value is calculated based on the desired delay time and the system clock frequency (16 MHz). The SysTick timer is polled to wait until the timer reaches zero, at which point the delay is complete.

```
void Delay(int us)
{
   STRELOAD = us * 16;            // reload value for 'ms' milliseconds
   STCTRL |= (CLKINT | ENABLE);      // set internal clock, enable the timer
   while ((STCTRL & COUNT_FLAG) == 0) // wait until flag is set
   {
      ;
   }
   STCTRL = 0;
   return;
}
```

## Observation:

- We observed that the light got less intense.
- The output showed a frequency of 1 kHz with a duty cycle of 20%. The GPIO pin toggled correctly between high and low states at the expected intervals, indicating that the SysTick timer was functioning as expected in polled mode.

## Conclusion:

- The experiment successfully demonstrated using the SysTick timer in polled mode to generate a 1kHz waveform with a 20% duty cycle. The timer provided accurate delays, allowing precise control of the output waveform. This method can be extended to generate different waveforms or timing signals by adjusting the reload values and duty cycle percentages.