Session 03: Frequency Counter using Timer and Interrupt of ESP 32 Microcontroller

Date of the Session: ___/___/___ **Time of the Session:** 12:45 to 2:20

PREREQUISITE:

- General idea of Timer, ESP32 board
- General idea of basic circuit

PRE-LAB:

1. What is the primary purpose of timers in microcontroller development, such as the ESP32?
   To generate precise time delays, schedule tasks and trigger events at regular intervals.

2. Explain the difference between hardware timers and software timers on the ESP32.
   hardware timers on the esp32 are managed by dedicated hardware peripherals for precise timing while software relies for time-based task scheduling.

OBJECTIVE:

- Generate Timer Interrupt events in Arduino IDE.

COMPONENTS REQUIRED:

- ESP32
- breadboard
- Jumper Wires Pack
- Micro USB Cable

THEORY:

ESP32 Timers & Timer Interrupts (Arduino IDE):

In this tutorial, you'll learn how to use ESP32 internal Timers & generate Timer Interrupt events in Arduino IDE. We'll discuss how ESP32 Timers work, how to configure ESP32's Timers, and how to generate periodic interrupts to synchronize the execution of logic within your project. And also measure the timer between two events whether they're external or internal events.

ESP32 Timers:

CODE:

```
#include <wire.h>
#include <LiquidCrystal_I2C.h>
#define IN_SIGNAL_GPIO 35
LiquidCrystal_I2C I2C_LCD(0x27,16,2);
hw_timer_t* Timer0_Cfg = NULL;
bool Measurement_Inprogress = false;
uint64_t Measured_Time = 0;
uint64_t Measured_Freq = 0;
void IRAM_ATTR Ext_INT1_ISR()
{
    if (Measurement_Inprogress == false)
    {
      Measurement_Inprogress = true;
      timerWrite(Timer0_Cfg, 0);
      timerStart(Timer0_Cfg);
    }
    else
    {
      Measured_Time = 0
      timerRead(Timer0_Cfg)
      timerStop(Timer0_Cfg)
      Measurement_Inprogress = false;
      if (Measured_Time == 0)
      {
        Measured_Freq = 0;
      }
      else
      {
        Measured_Freq = (40000000/Measured_Time);
      }
    }
}
```

POST LAB:

Take the snapshot of Tinker CAD simulation and paste here with your REG NO on it.

INTERFERENCE & ANALYSIS

Ensuring accurate frequency measuring by properly synchronising the timer interrupts and external signal, avoiding potential delays or misreads due to timer ints configuration or signal noise.

RESULT

this shows accurate ~~fr~~ frequency measurement, with the ESP32 sucessfully canting timer interrupts and displaying the frequency on the LCD without delays or errors