# MICROCONTROLADORES

## Práctica No. 16. Timers.

### 1. Objetivo

- Uso de los TIMERS.

### 2. Material y Equipo.

- Computador o laptop con el STM32CubeIDE.
- Un led 5mm o 3mm.

### 3. Marco de Referencia.

Los Timers son periféricos muy usados tanto para contar pulsos externos como para hacer temporizaciones. Un periférico se refiere a hardware que puede ser interno o externo al microcontrolador en el caso de los Timers son periféricos comunes a todos los microcontroladores, pero su implementación en hardware varia de cada fabricante ya que cada fabricante agrega las funcionalidades a sus Timers.

Existen tres tipos de Timers en los microcontroladores STM32 son:

- Basic Timers.
- General – Prupose Timers.
- Advanced – Control Timers.

El microcontrolador STM32F103C8T6 tiene implementados un Advanced – Control Timer y tres General – Prupose Timer. Para esta práctica se está usando el Timer 2 que es un timer

de propósito general. El modo que se está usando es el modo temporizador en modo contador de subida. La ecuación usada para calcular la temporización es la siguiente.

$$Update\ event = \frac{High\ speed\ clock}{(Prescaler + 1)(Period + 1)}$$

Donde “High speed clock” es la frecuencia con la que se alimenta el timer que son 72MHz, “Prescaler” es el divisor de frecuencia que nos sirve para bajar la frecuencia de 72MHz para poder lograr la temporización y el “Period” es el valor desde donde el timer va a empezar su conteo para desbordarse al pasar de 0xFFFF a 0x0000 ya que el Timer 2 es un timer de 16 bits.

Por ejemplo, para lograr un retardo de 0.5 segundos.

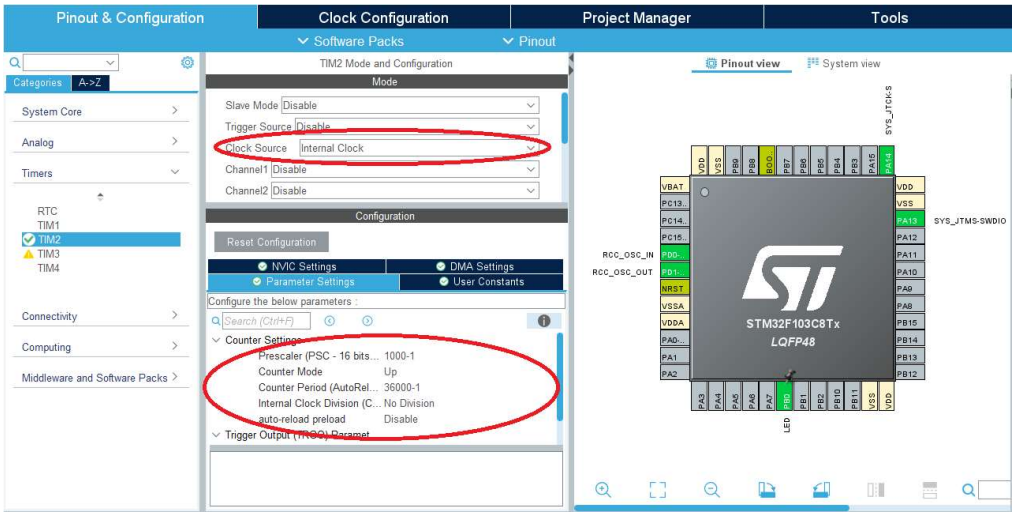
$$2\ Hz = \frac{72000000}{(35999 + 1)(999 + 1)}$$

Este cálculo se usará para realizar esta práctica.

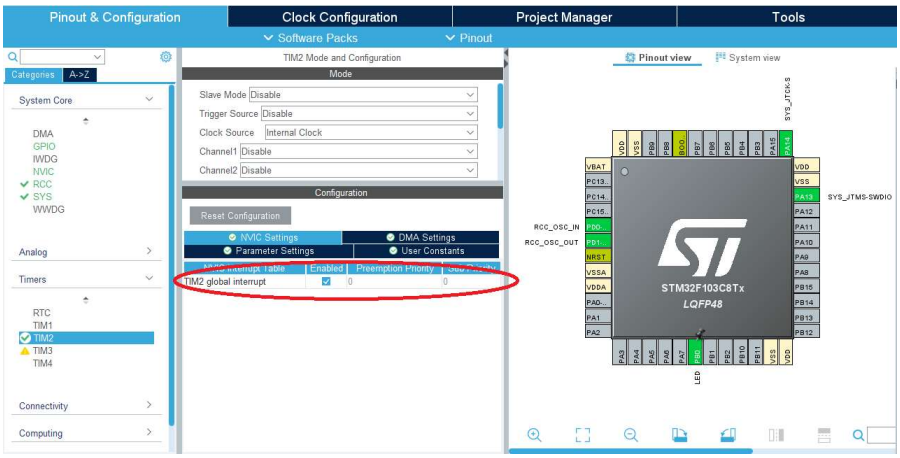
## 4. Desarrollo y Procedimiento.

Se creará un proyecto en el STM32CubeIDE como se indicó anteriormente.

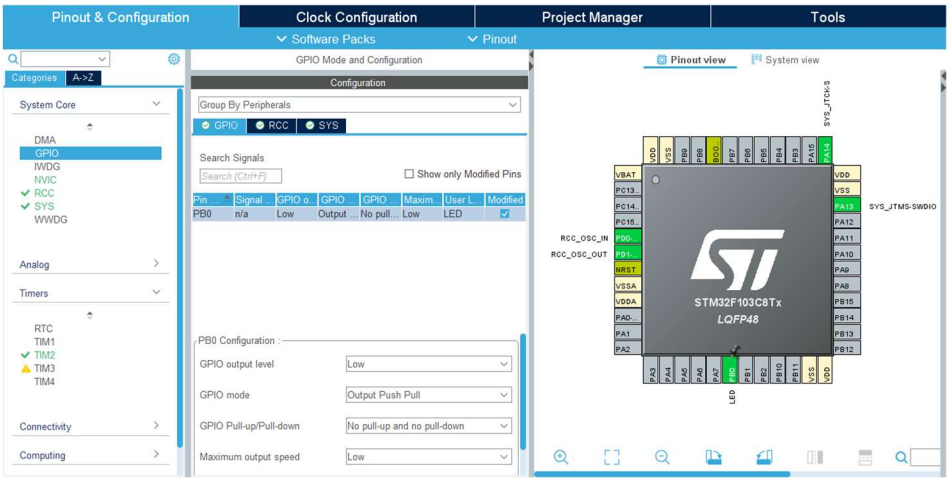
Primero nos vamos a “Categories/Timers” y seleccionamos TIM2 en “Clock Source” seleccionamos “Internal Clock” después en “Parameter Settings” configuramos prescaler y periodo como se muestra en la siguiente figura.



Despues en “NVIC Settings” habilitamos la interrupcion del TIM2.




Por ultimo configuramos el “PB0” como salida para poder medir el retardo de 0.5 segundos.



El código de la función main es el siguiente.

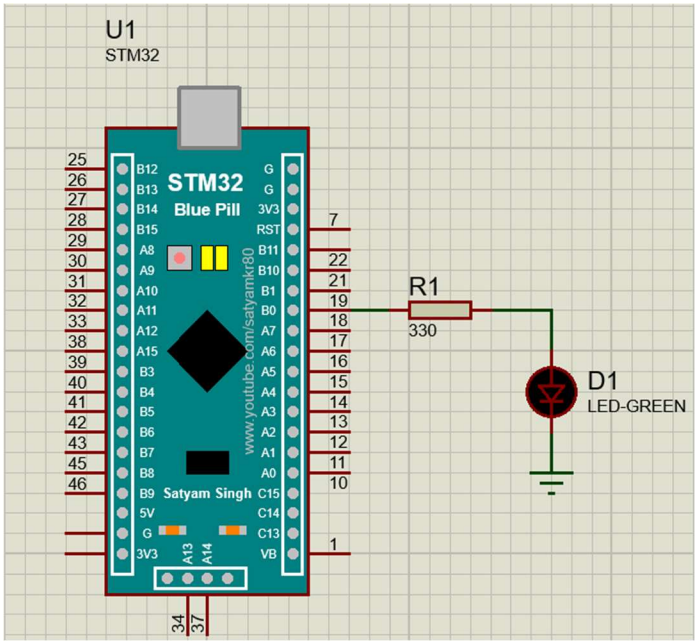
```c
#include "main.h"
#include "tim.h"
#include "gpio.h"

void SystemClock_Config(void);

int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
    MX_TIM2_Init();
    HAL_TIM_Base_Start_IT(&htim2);

    while (1)
    {
    }
}
```

El código de la función de interrupción.

```c
/* USER CODE BEGIN 4 */
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
    HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
}
/* USER CODE END 4 */
```

## 5. Esquemático del circuito.

El esquemático de la práctica se muestra a continuación.



## 6. Mejora.

En esta práctica con la base de tiempo de 0.5 segundos hay que generar dos retardos de 1 segundo y 1.5 segundos.

## 7. Observaciones.

Esta sección es para que el alumno anote sus observaciones.

## 8. Conclusiones.

Esta sección es para que el alumno anote sus conclusiones.

## 9. Importante.

La práctica deberá ser validad en el salón de clases antes de anexar el reporte al manual de prácticas. Una vez validad realizar el reporte de practica como se anteriormente y anexar al manual de prácticas que se entregara a final del curso.