



**Facultad Regional Tucumán**  
**Departamento Electrónica**

## **Técnicas Digitales II**

### **Actividad de Formación Práctica 3**

**Tema: Funciones no bloqueantes, aplicación con SysTick en STM32CubeIDE, programación de microcontroladores.**

Profesor:

Ing. Rubén Darío Mansilla

ATTP:

Ing. Lucas Abdala

vencimiento:

25 de octubre de 2024

*Año: 2024*

## Índice

<b>1. Objetivos . . . . .</b>	<b>4</b>
<b>2. Generalidades . . . . .</b>	<b>4</b>
<b>3. Consignas para desarrollar . . . . .</b>	<b>5</b>
<b>4. Bibliografía y documentación . . . . .</b>	<b>7</b>

## Registros de cambios

Revisión	Detalles de los cambios realizados	Fecha
0	Creación del documento	25 de setiembre de 2024
1	Ultimas modificaciones	12 de octubre de 2024

## 1. Objetivos

- Que el alumno logre implementar un módulo de software para trabajar con retardos no bloqueantes.
- Que el alumno aplique el concepto de funciones no bloqueantes en aplicaciones prácticas concretas utilizando el *IDE* STM32CubeIDE.
- Que el alumno desarrolle un driver de funciones no bloqueantes.
- Que el alumno aplique este driver en la implementación de aplicaciones con el IDE que estamos utilizando.

## 2. Generalidades

Esta actividad de formación práctica es del tipo **práctica de laboratorio**.

### Cómo desarrollar esta actividad:

- Esta actividad debe desarrollarse siguiendo las consignas del punto 3.
- Cada integrante del grupo debe tomar al menos una de las Apps desarrolladas en la actividad de formación práctica 2, utilizar el driver desarrollado para funciones no bloqueantes y realizar las modificaciones correspondientes en el **main.c** de la App para que utilice las funciones de este driver.
- Los integrantes del grupo deben ponerse de acuerdo para usar el mismo nombre para el driver de funciones no bloqueantes y para las funciones que lo integran.
- Una vez modificadas todas las aplicaciones, se las ubicará en el repositorio grupal para su presentación. Todas las Apps deben presentar una coherencia en cuanto al formato del nombre del mismo y de las funciones desarrolladas para este.
- Se debe utilizar el repositorio grupal creado en GitHub para la presentación de las actividades de formación práctica:
  - Se debe crear una carpeta para cada actividad de formación práctica.
  - Se deben ubicar en la carpeta correspondiente, en este repositorio, las aplicaciones desarrolladas para esta actividad de formación práctica respetando el siguiente formato de nombre: **App\_1\_Y\_Grupo\_X\_2024**.

**Nota sobre el formato de nombre de la carpeta:** X es el número de su grupo y Y es el número de aplicación como figura en el enunciado.

- Se debe insertar el link al repositorio de GitHub al final del informe que se presenta con esta actividad de formación práctica.
- El archivo que contiene el desarrollo del informe de la actividad de formación práctica debe ser de tipo pdf y su nombre debe respetar el siguiente formato:

**AFP\_3\_Grupo\_X\_TDII\_2024.pdf**

- Se debe realizar la presentación de las aplicaciones, en el laboratorio de Sistemas Digitales, funcionando según lo que pide la consigna, con su correspondiente defensa de lo desarrollado. Fecha a acordar. Cada alumno debe presentar y defender la aplicación que ha desarrollado para esta actividad.

**Nota 1:** Para el desarrollo de las aplicaciones de esta actividad de formación práctica se utilizará el entorno de desarrollo integrado STM32CubeIDE para aplicar sobre una placa de desarrollo STM32-NUCLEO-F4XX-YY.

**Nota 2:** Las placas de desarrollo sugeridas para las practicas son: STM32-NUCLEO-F401-RE, STM32-NUCLEO-F412-ZG, STM32-NUCLEO-F413-ZH ó STM32-NUCLEO-F429-ZI.

**Nota 3:** El desarrollo de las actividades de formación práctica se realizará y presentará de manera grupal.

### 3. Consignas para desarrollar

1. Tome como base un proyecto nuevo vacío para implementar las funciones auxiliares necesarias para usar retardos no bloqueantes en un archivo fuente **main.c** con su correspondiente archivo de cabecera **main.h**.

- 1.1. En **main.h** se deben ubicar los prototipos de las siguientes funciones y las declaraciones:

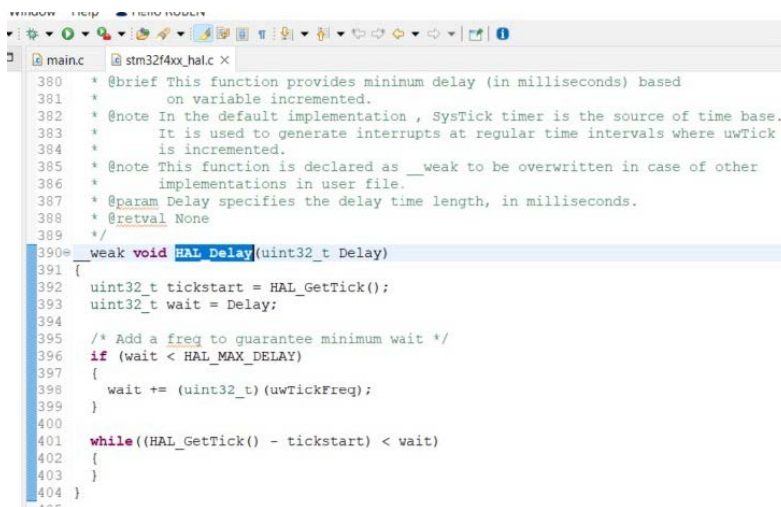
- Definición de tipos de variables personalizadas del driver:
  - **typedef** *uint32\_t* *tick\_t*; Ver qué biblioteca se debe incluir para que esto compile.
  - **typedef** *bool* *bool\_t*; Ver qué biblioteca se debe incluir para que esto compile.
  - **typedef** *struct* {  
  
    *tick\_t* *startTime*;  
    *tick\_t* *duration*;  
    *bool\_t* *running*;  
  
} *delay\_t*;
- Declaración de funciones para el driver de funciones no bloqueantes:
  - *void* **delayInit**( *delay\_t* \* *delay*, *tick\_t* *duration* );
  - *bool\_t* **delayRead**( *delay\_t* \* *delay* );
  - *void* **delayWrite**( *delay\_t* \* *delay*, *tick\_t* *duration* );

- 1.2. En **main.c** se debe ubicar la implementación de todas las funciones. Consideraciones para la implementación:

- **delayInit** debe:
  - cargar el valor de duración del retardo en la estructura, en el campo correspondiente.
  - No debe iniciar el conteo del retardo.
  - Debe inicializar el flag *running* en '*false*'.
- **delayRead** debe verificar el estado del flag *running*:
  - si es *false*, tomar marca de tiempo y cambiar *running* a '*true*'.

- si es *true*, hacer la cuenta para saber si el tiempo del retardo se cumplió o no: ‘**marca de tiempo actual - marca de tiempo inicial** es mayor o igual a **duración del retardo**’? y devolver un valor booleano que indique si el tiempo se cumplió o no.
- Cuando el tiempo se cumple se debe cambiar el flag *running* a *false*.
- **delayWrite** debe permitir cambiar el tiempo de duración de un *delay* existente.

**Nota 4:** para obtener una marca de tiempo se puede usar la función **HAL\_GetTick()** que devuelve un valor que se incrementa cada 1 ms y que se puede usar como base de tiempo.



```

380  * @brief This function provides minimum delay (in milliseconds) based
381  *       on variable incremented.
382  * @note In the default implementation , SysTick timer is the source of time base.
383  *       It is used to generate interrupts at regular time intervals where uwTick
384  *       is incremented.
385  * @note This function is declared as __weak to be overwritten in case of other
386  *       implementations in user file.
387  * @param Delay specifies the delay time length, in milliseconds.
388  * @retval None
389  */
390  __weak void HAL_Delay(uint32_t Delay)
391  {
392      uint32_t tickstart = HAL_GetTick();
393      uint32_t wait = Delay;
394
395      /* Add a freq to guarantee minimum wait */
396      if (wait < HAL_MAX_DELAY)
397      {
398          wait += (uint32_t)(uwTickFreq);
399      }
400
401      while((HAL_GetTick() - tickstart) < wait)
402      {
403      }
404  }

```

Figura 1. Funciones del SysTick.

- Una vez creadas estas funciones, haga las pruebas básicas necesarias en **main.c** para verificar que estas se comportan como se pide en el punto anterior. Compile el proyecto y verifique que funciona como corresponde.
- Cree los archivos **fuentes.c** y **header.h** para armar el driver de funciones no bloqueantes. Mueva las definiciones de tipos de variables y declaraciones de funciones al archivo **header.h** y las implementaciones al archivo **fuentes.c** y haga los ajustes necesarios para que compile sin errores. Ubique estos archivos del nuevo driver en la carpeta API, donde corresponde en la estructura de archivos del proyecto.
- Compile, y realice el *debug* y la programación de la placa de desarrollo para verificar que la aplicación corre y ejecuta las operaciones correctamente como lo hacía originalmente antes de la modificación. En este punto debería notar ciertas mejoras en la performance de las aplicaciones.
- Cada alumno, integrante del grupo, tomará una App de la practica anterior y realizará las modificaciones necesarias para integrar el nuevo driver de funciones no bloqueantes y reemplazar el uso de la función bloqueante **HAL\_delay()** por las nuevas funciones no bloqueantes desarrolladas en esta practica. Compilará y correrá la App en la placa de desarrollo.
- Desarrolle un breve informe que contenga los siguientes ítems:
  - Introducción al tema:** Todo lo que vio acerca de las funciones de retardo no bloqueantes, en que se basan, qué ventajas tiene el uso de este tipo de funciones en una aplicación comercial.

**6.2. Aplicaciones desarrolladas:**

- **Aplicación:** Nombre de la aplicación modificada.
- **Autor de la modificación:** Apellido y nombres del alumno que realizó las modificaciones en cada aplicación.
- **Observaciones:** sobre lo realizado en esa aplicación. En este campo puede compartir su experiencia en el desarrollo, los problemas que pudo haber enfrentado y como los solucionó. Recomendaciones si las hubiera.

**6.3. Link al repositorio grupal:** en el que se encuentran las aplicaciones desarrolladas para esta actividad de formación práctica.

#### 4. Bibliografía y documentación

- Documentación accesible desde el IDE en:  
**Help → Information Center → STM32CubeIDE Manuals**
- Documentación desarrollada por la cátedra: [Funciones no bloqueantes.pdf](#)