UTN Universidad Tecnológica Nacional

Facultad Regional Tucumán

Departamento Electrónica

Técnicas Digitales II

Actividad de Formación Práctica 5

Tema: Programación de microcontroladores: Anti-rebote desarrollado por MEF para garantizar detección positiva de pulsadores mecánicos, aplicación en STM32CubeIDE.

Profesor:
Ing. Rubén Darío Mansilla

ATTP:
Ing. Lucas Abdala

Alumnos: Angelillo Alessandro

Plaate Iván

Torres Juan

Hualampa Leonel

Año: 2025

1. Introducción al tema

En los sistemas digitales, los pulsadores mecánicos presentan un fenómeno conocido como **rebote** (*bounce*), que consiste en múltiples transiciones eléctricas indeseadas al momento de presionar o soltar el botón. Esto ocurre porque los contactos metálicos del pulsador no se estabilizan instantáneamente, generando varios flancos de subida y bajada en pocos milisegundos. Si el microcontrolador lee directamente ese pin sin tratamiento, puede interpretar una sola pulsación como varias, provocando errores en la aplicación (por ejemplo, incrementos múltiples o activaciones no deseadas).
Para evitar este problema, se implementan técnicas de **anti-rebote (debounce)**, que filtran las transiciones falsas y garantizan que cada pulsación sea detectada una sola vez.

En este trabajo se desarrolló una solución de **anti-rebote por software**, implementada mediante una **Máquina de Estados Finitos (MEF)**.
La MEF se encarga de controlar la secuencia de estados del pulsador: BUTTON_UP, BUTTON_FALLING, BUTTON_DOWN y BUTTON_RISING.
En cada ciclo, el programa evalúa el estado actual y las entradas, aplicando un retardo **no bloqueante** de 40 ms para validar el cambio de estado, utilizando la API de temporización API_delay.

El uso de una MEF para debounce ofrece varias ventajas:

- Permite un **código estructurado, modular y fácil de mantener**.
- Evita el uso de **delays bloqueantes**, mejorando la eficiencia del microcontrolador.
- Facilita la **reutilización del driver** en múltiples proyectos.
- Permite una **detección robusta y estable** de pulsaciones en aplicaciones industriales o comerciales.

Esta metodología es ampliamente utilizada en el desarrollo de **firmware profesional**, especialmente en sistemas embebidos que requieren estabilidad y respuesta precisa, como interfaces de usuario, teclados matriciales y paneles de control.

2. Aplicaciones desarrolladas

Aplicación: App_5_1_Grupo_5_2025

Autor de la modificación: Angelillo Alessandro

Descripción: La App 1.1 implementa el parpadeo de un único LED onboard con una frecuencia fija. Se integró el driver de funciones anti-rebote basado en MEF para detectar pulsaciones válidas del botón externo. Cada pulsación activa o desactiva el parpadeo del LED. El sistema utiliza retardos no bloqueantes y garantiza que el LED solo responda a pulsaciones reales.

Observaciones: La integración del driver fue directa. Se verificó que el LED responde correctamente a cada pulsación, sin falsos disparos. El uso de readKey() simplificó la

lógica en main.c. Se recomienda mantener la modularidad para futuras ampliaciones con múltiples LEDs.

Aplicación: App_5_2_Grupo_5_2025

Autor de la modificación: Hualampa Leonel

Descripción: La App 1.2 controla el encendido secuencial de tres LEDs. Cada pulsación del botón avanza al siguiente LED, encendiéndolo y apagando el anterior. Se utilizó el driver de anti-rebote para garantizar que cada pulsación sea única y válida. La lógica de transición se basa en una variable de estado y se actualiza con readKey().

Observaciones: Se detectó un problema inicial con rebotes múltiples que fue resuelto al integrar correctamente el driver. La MEF permitió una transición limpia entre LEDs. Se recomienda documentar bien los estados para facilitar el mantenimiento.

Aplicación: App_5_3_Grupo_5_2025

Autor de la modificación: Torres Juan

Descripción: La App 1.3 permite alternar entre dos modos de parpadeo: todos los LEDs encendidos o todos apagados. Cada pulsación del botón cambia el modo. Se utilizó el driver de anti-rebote para evitar cambios erráticos. La lógica se basa en una variable booleana que se invierte con cada pulsación detectada por readKey().

Observaciones: La aplicación mostró mejoras notables en estabilidad tras integrar el driver. Se probó con pulsaciones rápidas y prolongadas sin errores. Se recomienda agregar una indicación visual del modo actual para mejorar la experiencia del usuario.

Aplicación: App_5_4_Grupo_5_2025

Autor de la modificación: Plaate Ivan

Descripción: La App 1.4 implementa el parpadeo simultáneo de los tres LEDs con cambio de frecuencia mediante pulsador. Las frecuencias son: 100 ms, 250 ms, 700 ms y 2000 ms. Cada pulsación válida avanza a la siguiente frecuencia en forma cíclica. Se utilizó el driver de anti-rebote y retardos no bloqueantes con HAL_GetTick().

Observaciones: Inicialmente, el sistema solo detectaba las dos primeras frecuencias. El problema se resolvió al configurar correctamente el pin del pulsador con GPIO_PULLUP. La MEF permitió una detección precisa de cada pulsación. Se recomienda validar siempre la configuración de hardware antes de depurar el software.

3. Link al repositorio grupal

Haz click aquí