Universidad Sergio Arboleda
Sistemas Embebidos
Laboratorio #3 – Informe
Integrantes:

- **Cristian Mora**
- **Yeison Lara**
- **Daniel Avila**

---

1. Introducción

En este laboratorio se implementó un sistema embebido basado en el laboratorio #2, realizando modificaciones para mejorar su funcionamiento. El objetivo principal fue calcular un TOKEN utilizando la hora actual como referencia en milisegundos, además de sincronizar la comunicación entre un PC y un microcontrolador STM32. La sincronización debía realizarse mediante la transmisión de la hora codificada con parpadeos o cambios en la señal.

2. Objetivos

El objetivo de este laboratorio es desarrollar un sistema embebido que permita la transmisión y sincronización de la hora entre un PC y un microcontrolador STM32, implementando un desfase de 5 segundos. Se busca utilizar el módulo TIM para el cálculo del TOKEN, descartando funciones como GetTick, y garantizando una mayor precisión mediante la conversión de la hora actual a milisegundos. Además, se pretende codificar la hora a través de parpadeos, optimizando la comunicación y asegurando una sincronización eficiente entre dispositivos. Otro propósito clave es la generación y actualización automática del TOKEN cada 10 segundos, empleando todos los bytes del número calculado para evitar pérdidas de información. Finalmente, se busca integrar una señal de sincronismo que permita iniciar el proceso de conteo de manera automática, sin necesidad de intervención manual, asegurando así la estabilidad y autonomía del sistema.

3. Desarrollo del Laboratorio

3.1 Cálculo del TOKEN

El TOKEN se obtiene a partir de la hora actual del sistema, expresada en milisegundos. Se utiliza la siguiente conversión:

Ejemplo: Hora = 19:53:45

- 19 horas = 19 × 3,600,000 ms = 72,000,000 ms
- 53 minutos = 53 × 60,000 ms = 3,180,000 ms
- 45 segundos = 45 × 1,000 ms = 45,000 ms

- TOTAL = 75,225,000 ms

Este número se emplea junto con una clave para calcular el TOKEN final.

3.2 Sincronización de la Hora

El sistema debe extraer la hora del PC, sumarle 5 segundos y codificarla para ser enviada al STM32 mediante parpadeos o cambios de señal. La transmisión debe respetar el orden de los milisegundos.

Proceso:

1. El PC extrae la hora del sistema y le suma 5 segundos.
2. Codifica la hora en un formato binario (parpadeos o cambios de señal).
3. Envía la hora codificada al STM32.
4. El STM32 recibe y decodifica la hora, quedando en espera de la señal de sincronismo.
5. La señal de sincronismo se genera automáticamente aproximadamente 5 segundos después.
6. Una vez sincronizado, el STM32 muestra el TOKEN en pantalla cada 10 segundos.

3.3 Descripción del Código Implementado

El código desarrollado tiene como objetivo recibir la hora desde el PC a través de una señal óptica codificada mediante parpadeos, decodificar dicha señal y utilizarla para calcular y actualizar el TOKEN cada 10 segundos.

El flujo general del programa está dividido en varios bloques funcionales:

1. Bloqueo Inicial con Señal de Sincronismo: Se utiliza un bucle while que mantiene bloqueado el sistema hasta que la señal en el pin GPIOB_PIN_8 (fotocelda) sea baja. Esta señal representa el inicio del proceso de transmisión y sincronización, asegurando que el sistema comience a capturar datos únicamente cuando la señal de inicio esté presente.
2. Captura del Tiempo de Parpadeos: Se utiliza un bucle for que recorre 24 ciclos para capturar los tiempos de duración de los pulsos de la señal recibida (blancos y negros), correspondientes a los bits de la hora en formato binario. El sistema mide la duración de los estados lógicos utilizando el contador del TIM2, lo que permite decodificar si un bit es '0' o '1' en función del ancho del pulso.
3. Decodificación de Hora, Minutos y Segundos:
    - Los primeros 8 bits capturados se interpretan como las horas.
    - Los siguientes 8 bits, como los minutos.

- Los últimos 8 bits, como los segundos.

Cada uno se construye desplazando los bits e interpretando la duración del pulso como un '0' lógico (si es corto) o un '1' lógico (si es largo).

4. Conversión a Milisegundos y Ajuste de Tiempo: Después de decodificar los valores de hora, minuto y segundo, el sistema los convierte a milisegundos. Además, se aplica un ajuste de 6 segundos (6000 ms), considerando el desfase introducido intencionalmente por el programa de transmisión en el PC.
5. Generación y Actualización del TOKEN: Dentro del bucle infinito while, se controla el tiempo utilizando los valores del contador del TIM2. Cada segundo se incrementa la variable de segundos (segundos1), y cada 10 segundos se calcula el TOKEN mediante la fórmula:
6. TOKEN = TIEMPOTOTAL ^ CLAVE

Donde TIEMPOTOTAL representa el tiempo actual en milisegundos y CLAVE es una constante definida para encriptar el TOKEN. Esta operación XOR garantiza la variabilidad del TOKEN con el paso del tiempo.

7. Visualización en Pantalla OLED: Finalmente, el TOKEN y la hora actual son mostrados en la pantalla OLED. Se hace uso de funciones de la librería SSD1306 para limpiar la pantalla, posicionar el cursor e imprimir la información en dos líneas: una para el TOKEN (en formato hexadecimal) y otra para la hora (en formato HH:MM:SS).

4. Resultados y Conclusiones

Durante el desarrollo del laboratorio se logró implementar exitosamente el sistema de sincronización entre el PC y el microcontrolador STM32. Se consiguió transmitir la hora desde el PC al STM32 con un desfase de 5 segundos, utilizando codificación mediante parpadeos que facilitó una sincronización precisa. Gracias al uso del módulo TIM, el STM32 pudo calcular el TOKEN adecuadamente y actualizarlo de forma automática cada 10 segundos. Además, la señal de sincronismo funcionó como disparador para iniciar el conteo, eliminando la necesidad de intervención manual y permitiendo que el sistema funcionara de forma autónoma y estable. La implementación del módulo TIM permitió un manejo más preciso de los tiempos, mientras que la conversión de la hora a milisegundos resultó esencial para la correcta generación del TOKEN. La señal de sincronismo garantizó la correcta transmisión de la hora y, junto con la actualización periódica del TOKEN, proporcionó un sistema robusto, confiable y preciso para el control temporal en sistemas embebidos.