Centro De Control Con Stm32 Nucleo Mediante Uart, Pwm E Interrupciones

Juan Diego Restrepo Hernandez
C.C. 1104695823

Universidad Nacional de Colombia
Facultad de Ingeniería
Programa: Ingenieria Electronica
Asignatura: Estructuras computacionales

Manizales Caldas
21/05/2025

- **Introduccion:**

Este proyecto implementa, en un sistema embebido basado en **STM32 Nucleo**, el control de diferentes periféricos e integrados, incluyendo:

- Parpadeo de un LED integrado como señal de actividad del sistema (heartbeat).
- Control de un LED externo mediante un botón físico, con gestión de rebotes y temporizador de apagado.
- Comunicación UART bidireccional con el PC, con procesamiento de comandos recibidos.
- Generación de una señal PWM para el control de intensidad de un segundo LED.

Todo esto se logra haciendo uso de periféricos como **UART, TIM3, EXTI** y **SysTick**.

- **Descripcion Funcional:**

El programa está diseñado para interactuar tanto con el usuario como con dispositivos externos a través de LEDs, un botón físico y comunicación UART. Su funcionamiento se basa en las siguientes acciones principales:

- Un LED integrado (LD2) parpadea periódicamente para indicar que el sistema está en funcionamiento.
- Al presionar el botón (B1), se enciende un LED externo durante 3 segundos.
- El sistema puede recibir comandos desde el PC por UART para controlar otro LED mediante señal PWM.
- Gestion de eventos de forma no bloqueante (sin delays) utilizando interrupciones y temporizadores

Estas funciones demuestran el uso combinado de periféricos en sistemas embebidos, como GPIO, EXTI, UART, TIM3 y SysTick.

- **Funcionalidad del programa:**
  - **Heartbeat LED integrado**

Un LED integrado (conectado al pin PA5) parpadea cada 500 ms como señal de vida del sistema. Este parpadeo se controla usando el temporizador **SysTick**, configurado para generar una interrupción cada 1 ms.

  - **Control de LED externo por botón (B1)**

El boton (B1) conectado al pin PC13, está configurado para generar una interrupción mediante EXTI cuando se detecta un flanco de bajada(cuando se presiona).

Al presionar el botón sucede

- Se enciende el led externo conectado al pin PA7
- Se activa un temporizador de 3 segundos usando la función systick_get_tick()
- Pasados los 3 segundos, el LED se apaga automáticamente
- Se agrega una protección antirebote de 200ms para evitar ruido eléctrico, fallas mecánicas, o falsas lecturas

- **Comunicación UART:**

El sistema utiliza USART2 (PA2-TX, PA3-RX) a 115200 baudios para enviar mensajes al PC y recibir comandos de control.

Cuando se recibe un carácter, el sistema:

- Hace *eco* del carácter recibido.
- Interpreta los siguientes comandos:
    - 'h' o 'H': el PWM se aumenta al 100% (encendiendo el LED).
    - 'l' o 'L': PWM se disminuye a 0% (apagado el LED).
    - 't': Alterna el estado del LED conectado a PA7.

- **Control de intensidad con PWM:**

El pin PA6 está configurado en modo Alternate Function (AF2) para generar una señal PWM desde el temporizador TIM3_CH1 a una frecuencia de 1000 Hz.
El ciclo de trabajo (duty cycle) es ajustado dinámicamente desde el código, por defecto inicia en 70%, y puede ser modificado por comandos UART.

- **Conecciones físicas de los LEDs:**

LED Heartbeat (LD2): PA5 = D13

LED externo (PWM): PA6 = D12

LED externo ON/OFF: PA7 = D11

- **Estructura del código:**

El proyecto esta organizado en múltiples archivos, los cuales separan la lógica de la aplicación de la configuración de los periféricos. A continuación se hace un resumen de los archivos

Archivos principales:

- main.c

Es el punto central del programa. Inicializa los periferics y contiene el bucle principal (while(1)), en el cual se ejecuta el heartbeat y la lógica de la aplicación

- room_control.c y room_control.h

Contiene la lógica de control del sistema: manejo de botón, temporizador de apagado del LED y procesamiento de comandos UART

Control de periféricos:

- gpio.c y gpio.h:

Configura los pines de entrada y salida, ya sea en su modo, estado o alternancia

- rcc.c y rcc.h

Habilita los relojes necesarios para GPIO, USART2 y TIM3

- nvic.c y nvic.h

Configura y habilita las interrupciones externas como EXTI y UART

- systick.c y systick.h

Es el contador en milisegundos con systick además posee las funciones de retardo

- tim.c y tim.h

Inicializa el temporizador TIM3 para generar la señal PWM

- uart.c y uart.h

Configura el USART2 para enviar y recibir datos del PC

Flujo del programa:

1. Inicio del sistema (main.c):

Al iniciar el microcontrolador, el programa ejecuta el main() en el cual:

- Se configura el temporizador Systick para que genere una interrupción cada 1ms

- Se inicializan los pines GPIO:
- PA5 (LED heartbeat)
- PA6 (PWM para el LED)
- PA7 (LED ON/OFF)
- PC13 (botón B1, como entrada)
- Se configura el botón B1 para que genere una interrupción externa mediante EXTI
- Se inicializa USART2 para la comunicación con UART a 115200 baudios y se habilita su interrupción
- Se inicializa el temporizador TIM3 para generar una señal PWM de 1000hz en PA6
- Por ultimo se llama a room_control_app_init() para terminar e iniciar el sistema
- Se imprime “Sistema Inicializado. Esperando eventos...”

2. Bucle principal (while(1)):

El sistema entra en un bucle infinito donde se ejecutan dos funciones clave.

- Heartbeat_led_toggle():
- Usando systick_get_tick() hace que el LED PA5 alterne su estado cada 500ms
- Room_control_process():
- Es la función encargada de apagar el LED externo PA7 despues de 3 segundos si este fue encendido por el botón B1.

El bucle incluye un pequeño retardo apoyado del systick (systick_delay_ms(10)) para evitar un consumo innecesario de CPU

3. Interrupcion por botón (EXTI13 -> PC13)

Cuando el usuario persona el botón B1:

- Se inicial la interrupción EXTI15_10_IRQHandler()
- Se llama a la función room_control_on_button_press()

Dentro de la función:

- Se verifica si la pulsación es valida (ósea que no sea un rebote, o falla mecánica, esto con ayuda de una espera de 200ms)
- Se enciende el LED en PA7
- Se guarda el tiempo de encendido
- Se marca que el LED esta activo
- Se envia un mensaje por el UART “Boton B1: Presionado. LED encendido por 3 segundos.”

Con ayuda del Systick y las funciones ya creadas, se cuentan los 3 segundos y se apaga el led

4. Interrupción por UART (USART2_IRQHandler):

Cuando llega un carácter por UART:

- Se lee el carácter y se hace eco (se reenvía al monitor)
- Se llama a la función room_control_on_uart_receive()

Dentro de esta función se realiza:

se analiza el carácter recibido

- “h” o “H” = Se aumenta el PWM al 100%, haciendo brillar el LED al máximo
- “l” o “L” = Se disminuye el PWM al 0%, haciendo que el LED se apague
- “t” = Hace que el LED PA7 cambie su estado actual (ON -> OFF, OFF -> ON)

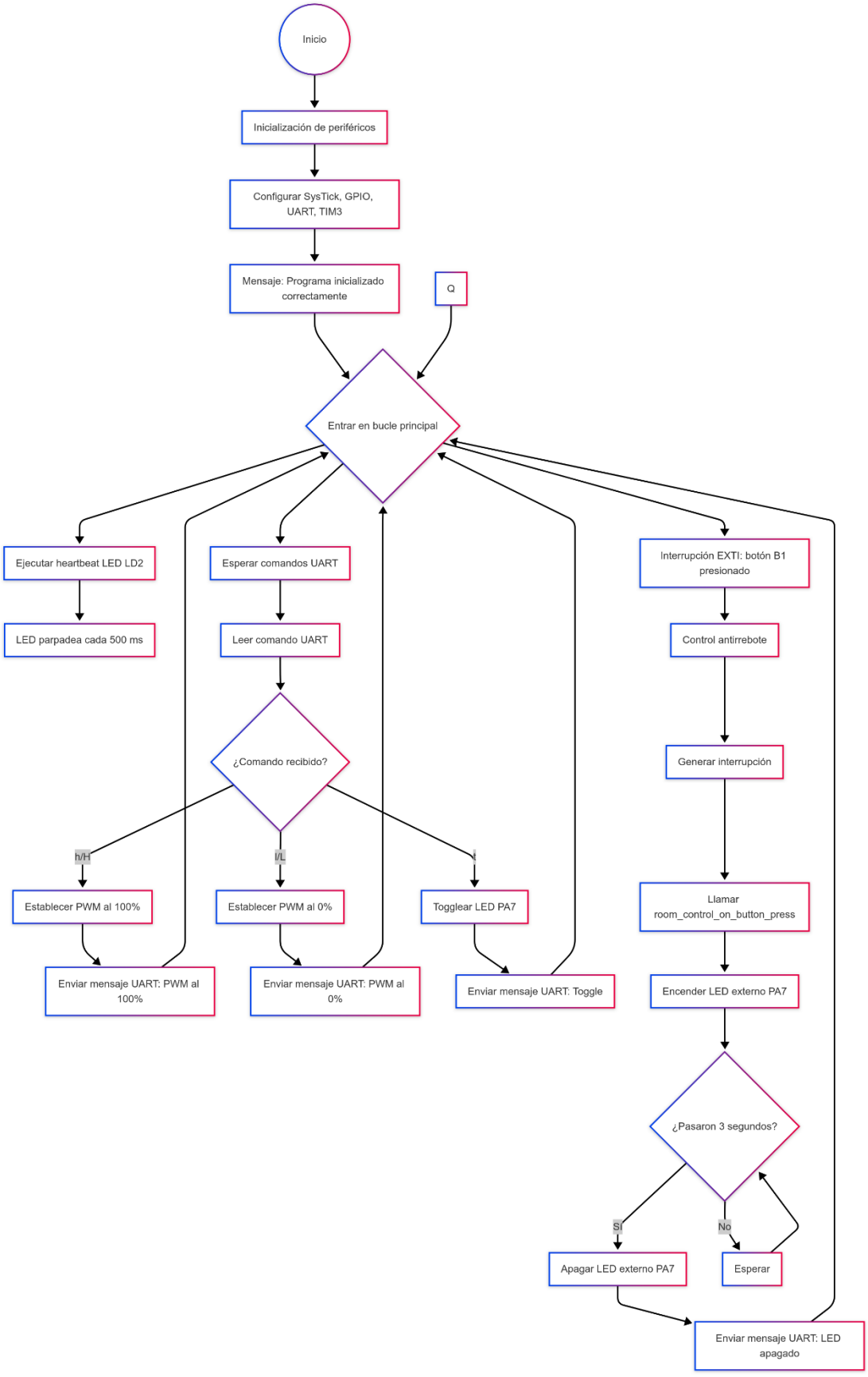
Se envia un mensaje por UART dependiendo del comando utilizado

5. Temporizador SysTick (SysTick_Handler):

Cada 1ms se incremente una variable global llamada g_systick_ms_count, que se usa como base de tiempo para:

- medir intervalos como lo son los 3 segundos de encendido del LED
- Generar los delays no bloqueantes
- Controlar el heartbeat

- 5. **Diagrama de flujo:**


Inicio
Inicialización de periféricos
Configurar SysTick, GPIO, UART, TIM3
Mensaje: Programa inicializado correctamente
Q
Entrar en bucle principal
Ejecutar heartbeat LED LD2
LED parpadea cada 500 ms
Esperar comandos UART
Leer comando UART
¿Comando recibido?
h/H
l/L
Establecer PWM al 100%
Establecer PWM al 0%
Togglear LED PA7
Enviar mensaje UART: PWM al 100%
Enviar mensaje UART: PWM al 0%
Enviar mensaje UART: Toggle
Interrupción EXTI: botón B1 presionado
Control antirrebote
Generar interrupción
Llamar room_control_on_button_press
Encender LED externo PA7
¿Pasaron 3 segundos?
Sí
No
Apagar LED externo PA7
Esperar
Enviar mensaje UART: LED apagado

6. Conclusiones:

Este proyecto permitió integrar múltiples periféricos del microcontrolador STM32 Nucleo para el control de LEDs, el uso de interrupciones, la comunicación UART y la generación de señales PWM. Además, la organización modular del código ayuda a mantener un proyecto más ordenado, con archivos más cortos pero a la vez más comprensibles.