ARM Microcontroller Internship – Assignment

Topic: External Interrupt (EXTI) using Bare-Metal Programming

Board: STM32F407G-DISC1
Programming Style: Bare-metal (Register Level)
Demo Covered: EXTI0 on USER Switch (PA0)

Question 1: Modify Interrupt Trigger (Edge Selection)

Problem Statement

The given program configures EXTI0 to generate an interrupt when the USER switch is **pressed**.

Modify the program so that the interrupt is generated when the USER switch is **released** instead of pressed.

Tasks

1. Identify which **EXTI trigger selection register** must be modified.
2. Write the **required register-level code** to implement this change.
3. Explain the **logic behind the modification** in terms of signal transition.

Hint

Think in terms of **edge transitions**:

- LOW → HIGH
- HIGH → LOW

Objective

To understand **EXTI edge selection** using:

- Rising Trigger Selection Register (RTSR)
- Falling Trigger Selection Register (FTSR)

Question 2: EXTI with Software Flag (volatile Concept)

Problem Statement

Modify the existing EXTI program with the following constraints:

- The **ISR must NOT toggle the LED**
- The ISR should **only set a software flag**
- The LED toggle operation should be performed inside the `while(1)` loop in `main()`

Requirements

1. Declare a **global flag variable**.
2. Modify the `EXTI0_IRQHandler()` so that it:
    - Clears the EXTI pending bit
    - Sets the flag
3. In the `main()` loop:
    - Check the flag
    - Toggle the LED
    - Clear the flag
4. The flag variable **must be declared as `volatile`**.

Explain in comments

- Why the flag variable must be declared as `volatile`
- What problem may occur if `volatile` is not used

Compiler Optimization Note (Important)

At higher optimization levels (e.g., `-O2`, `-O3`), the compiler may:

- Cache non-volatile variables in CPU registers
- Assume the variable does not change unexpectedly

Since an ISR is a **separate execution context**, any variable shared between:

- `main()` and
- an ISR

must be declared `volatile`, otherwise:

- Updates done inside the ISR may not be visible to `main()`
- This can result in **infinite loops or incorrect behavior**

Objective

To understand:

- ISR best practices
- Compiler optimization effects
- Correct usage of the `volatile` keyword in interrupt-based programs

---

Question 3: Extend EXTI Functionality (Design Thinking)

Problem Statement

Design and implement the following behavior using **EXTI interrupt only**:

- **First button press** → LED ON
- **Second button press** → LED OFF
- Subsequent button presses should continue toggling the LED state

---

Conditions

- Use **EXTI interrupt only**
- Use **bare-metal register-level programming**
- **No delay functions** inside the ISR
- ISR must be **short and non-blocking**

Requirements

1. Design a **state-based logic** to track button presses.
2. Declare required **global variables**.
3. Implement the logic using:
    - EXTI interrupt
    - Software state variables
4. Explain the logic in simple steps.

Objective

To develop:

- Interrupt-driven state machines
- Proper separation of ISR and application logic
- Real-world embedded design thinking
