STM32 | PUSH BUTTON

Title: Interfacing Push Button with STM32F103C8T6 (Blue Pill) - CMSIS

Objective:

To interface a push button switch with the STM32F103C8T6 microcontroller and control an LED using the button input.

Learning Outcomes

After completing this experiment, the student will be able to:

1. Understand GPIO pin modes and input configuration in STM32.
2. Configure push button as input with pull-up resistor.
3. Interface LED as output and control it using button logic.
4. Write embedded C code directly using register-level programming.
5. Verify hardware connection and simulate in Proteus.

Concept and Theory

1. What is a Push Button?

A push button is a simple switch used to make or break an electrical connection when pressed.
It has two states:

- Pressed (logic 0) → connected to GND
- Released (logic 1) → pulled up internally to Vcc

In STM32, we can configure the input pin as:

- Floating Input
- Pull-up Input
- Pull-down Input

2. GPIO Input Configuration

Each GPIO pin can be configured using the CRL/CRH (Control Registers):

MODE bits	CNF bits	Function
00	00	Analog Input
00	01	Floating Input
00	10	Input with Pull-up / Pull-down
00	11	Reserved

Pull-up or Pull-down selection is done using the ODR register:

- `GPIOx->ODR |= (1 << pin);` → Pull-up
- `GPIOx->ODR &= ~(1 << pin);` → Pull-down

3. GPIO Output Configuration

MODE bits	CNF bits	Description
00	00	Input Mode
01	00	Output Mode (10 MHz, Push-pull)
10	00	Output Mode (2 MHz, Push-pull)
11	00	Output Mode (50 MHz, Push-pull)

4. STM32F103C8T6 Pin Selection

Function	Port/Pin	Description
Button Input	PA0	Configured as input (pull-up)
LED Output	PA1	Configured as output (push-pull)

Hardware Required

Component	Specification	Quantity
STM32F103C8T6	Blue Pill Board	1
Push Button	Momentary switch	1
LED	Red or any color	1
Resistor	220 Ω for LED	1
Jumper Wires	Male-Female	As required
Breadboard	–	1
USB to Serial/ST-Link	Programmer	1

Circuit Diagram

```
PA0 ----> Push Button ----> GND
(PA0 internally pulled up)

PA1 ----> LED ----> 220Ω ----> GND
```

Pin Configuration

STM32 Pin	Direction	Connection
PA0	Input	Push Button
PA1	Output	LED
GND	–	Common Ground
3.3V	–	Pull-up Voltage

Software Tools Required

Tool	Purpose
STM32CubeIDE	Code development & debugging
ST-Link Utility	Programming the MCU
Proteus / Tinkercad	Simulation (optional)

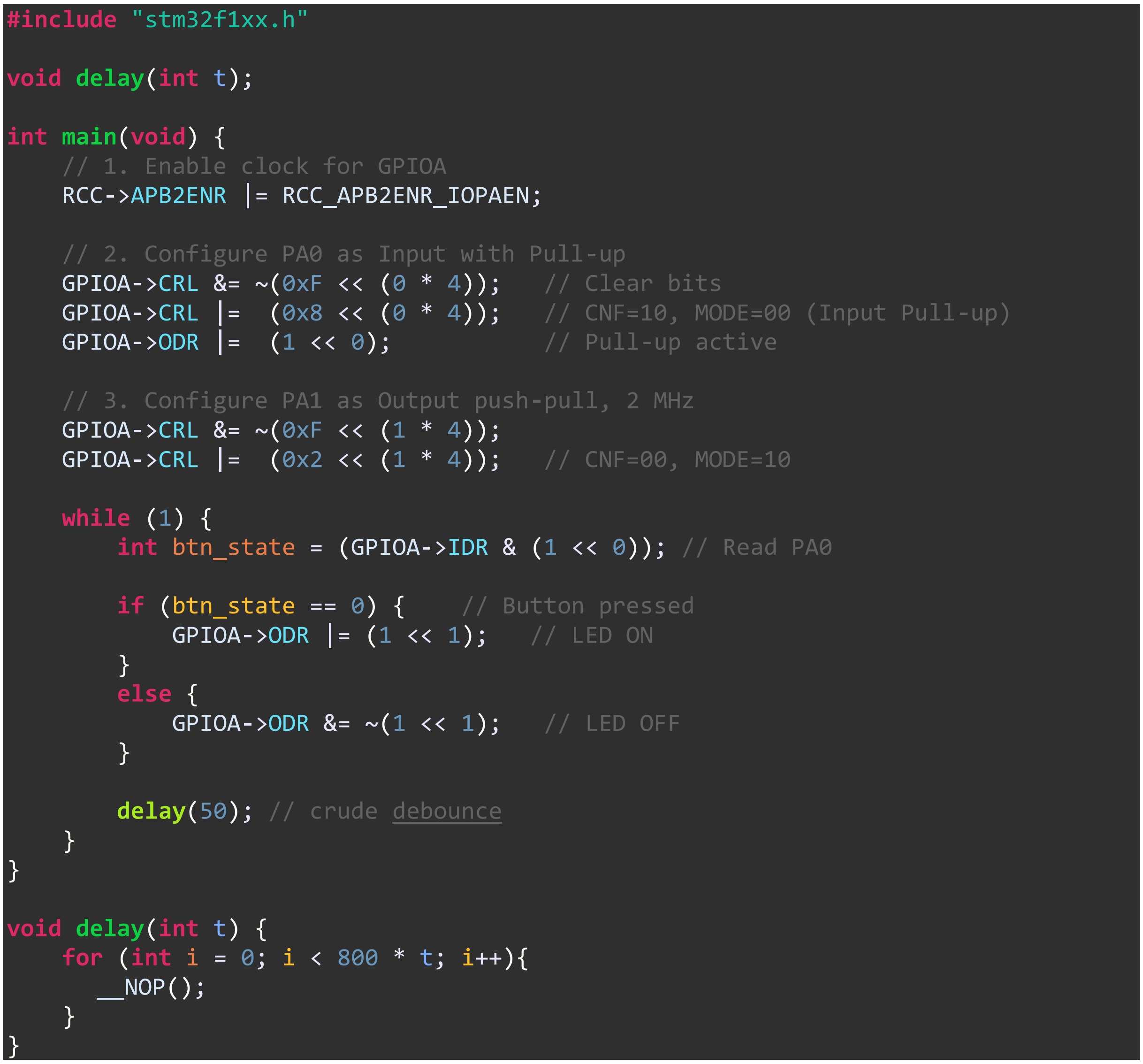
Algorithm / Procedure

Step 1: Create Project

1. Open STM32CubeIDE
2. Create new project → Board/Chip: STM32F103C8Tx
3. Give project name: Button_LED
4. Disable HAL drivers (we'll use registers)

Step 2: Code the Program

File: main.c



```c
#include "stm32f1xx.h"

void delay(int t);

int main(void) {
    // 1. Enable clock for GPIOA
    RCC->APB2ENR |= RCC_APB2ENR_IOPAEN;

    // 2. Configure PA0 as Input with Pull-up
    GPIOA->CRL &= ~(0xF << (0 * 4));   // Clear bits
    GPIOA->CRL |=  (0x8 << (0 * 4));   // CNF=10, MODE=00 (Input Pull-up)
    GPIOA->ODR |=  (1 << 0);           // Pull-up active

    // 3. Configure PA1 as Output push-pull, 2 MHz
    GPIOA->CRL &= ~(0xF << (1 * 4));
    GPIOA->CRL |=  (0x2 << (1 * 4));   // CNF=00, MODE=10

    while (1) {
        int btn_state = (GPIOA->IDR & (1 << 0)); // Read PA0

        if (btn_state == 0) {     // Button pressed
            GPIOA->ODR |= (1 << 1);   // LED ON
        }
        else {
            GPIOA->ODR &= ~(1 << 1);   // LED OFF
        }

        delay(50); // crude debounce
    }
}

void delay(int t) {
    for (int i = 0; i < 800 * t; i++){
        __NOP();
    }
}
```

Explanation of Each Line

Code Line	Description
`RCC->APB2ENR	= …`
GPIOA->CRL	Sets PA0 and PA1 modes
GPIOA->ODR	Used to apply internal pull-up
GPIOA->IDR	Reads input value
GPIOA->ODR |= (1 << 1)	Sets LED pin high
GPIOA->ODR &= ~(1 << 1)	Resets LED pin low
delay()	Adds time delay

Execution Steps

1. Connect hardware as per circuit.
2. Open STM32CubeIDE → build → Run/Debug.
3. If "GDB Server Failed" → ensure:
   - ST-Link driver installed
   - Board powered from USB
   - ST-Link connected correctly (SWCLK → SWDIO, GND, 3.3V)
4. Observe LED behavior:
   - LED OFF when button released
   - LED ON when button pressed

Observations

Button State	PA0 Logic	LED (PA1) State
Released	HIGH (1)	OFF
Pressed	LOW (0)	ON

Result

Successfully interfaced a push button with the STM32F103C8T6 microcontroller and controlled an LED output based on button press.

Viva Questions

1. What is the purpose of pull-up resistor in GPIO input mode?
2. What happens if we configure the button pin as floating input?
3. Explain the difference between ODR, IDR, BSRR, and BRR.
4. Why do we enable clock before configuring GPIO?
5. How can debounce be handled in software and hardware?

Extended Task (For Practice)

1. Modify the code to toggle LED on each button press (edge detection).
2. Interface two buttons — one to turn ON, one to turn OFF.