From blinking an LED to reading silicon

A documented first STM32 Black Pill experiment: wiring, CubeMX configuration, a working HAL program, SWD flashing, debugging, and observing GPIOC bit 13.

Outcome: Onboard PC13 LED blinking; 500 ms and 100 ms delays tested; firmware verified through ST-LINK V2; breakpoint hit in main(); ODR13 observed changing from 0 to 1.

Hardware: Black Pill development board, STM32F401 family as reported by ST-LINK; ST-LINK V2-compatible SWD programmer. Software: STM32CubeMX, STM32CubeIDE and STM32 HAL.

Part identification note: The configuration used STM32F401CCUx, whereas an earlier tool reported 128 KB flash for the connected device. The precise chip marking and usable flash size still require independent verification.

The process, end to end

01. WIRE - ST-LINK to SWDIO, SWCLK, GND and voltage sense as applicable
02. CONFIGURE - STM32CubeMX: PC13 GPIO output; SYS debug Serial Wire
03. GENERATE - STM32CubeIDE HAL project, complete startup and drivers
04. CODE - Toggle PC13 inside main() forever, delay 500/100 ms
05. BUILD - 0 errors, 0 warnings
06. PROGRAM - ST-LINK flash download verified successfully
07. DEBUG - Break at line 97; F6 step; observe GPIOC ODR13

Connection reference

ST-LINK V2	Black Pill	Meaning
SWDIO	PA13 / SWDIO	Bidirectional debug data
SWCLK	PA14 / SWCLK	Debug clock
GND	GND	Common ground
3.3V / VTref	3V3 (as applicable)	Voltage reference; avoid tying independent outputs together

The first program that worked

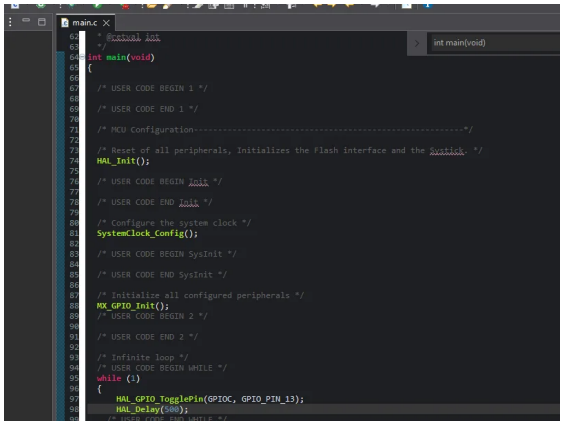
The code belongs in main(), after MX_GPIO_Init(), inside the CubeMX protected USER CODE section. It should not go inside Error_Handler().

```
/* USER CODE BEGIN WHILE */
while (1)
{
    HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_13);
    HAL_Delay(100);
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```

Delay call	One ON/OFF cycle	Approximate blink frequency
HAL_Delay(500)	1,000 ms	1 Hz
HAL_Delay(100)	200 ms	5 Hz

How it works

HAL_GPIO_TogglePin() changes the PC13 output latch via the GPIO peripheral. HAL_Delay() uses the HAL time base to wait before the next toggle. The Black Pill LED is typically active-low: PC13 = 0 lights it, PC13 = 1 turns it off.



Actual STM32CubeIDE screenshot of the working while(1) loop and successful build.

From a breakpoint to a register bit

After a verified flash, the debugger halted the Cortex-M4 at main.c:97, on HAL_GPIO_TogglePin(). F6 (Step Over) executed the function and suspended the core at the next line.

Shortcut	Operation	Observed result
F6	Step Over	Execute one C source line and suspend again
F8	Resume	Run until the next breakpoint
F5	Step Into	Enter a called function where debug information permits
F7	Step Return	Finish current function and return

The decisive measurement

State	GPIOC_ODR	ODR13	Active-low LED
Before toggle / illustrative snapshot	0x00000000	0	ON
After Step Over (halted)	0x00002000	1	OFF

The first screenshot was captured while execution was running, so it is not a synchronized instruction-level snapshot. The second screenshot - CPU suspended at main.c:98 - demonstrates that the register has changed to 0x2000.

Why 0x2000? Bit 13 has weight 2^13 = 8192 decimal = 0x2000. When only this bit is set, the output data register reads 0x00002000.

Memory-mapped peripheral addresses

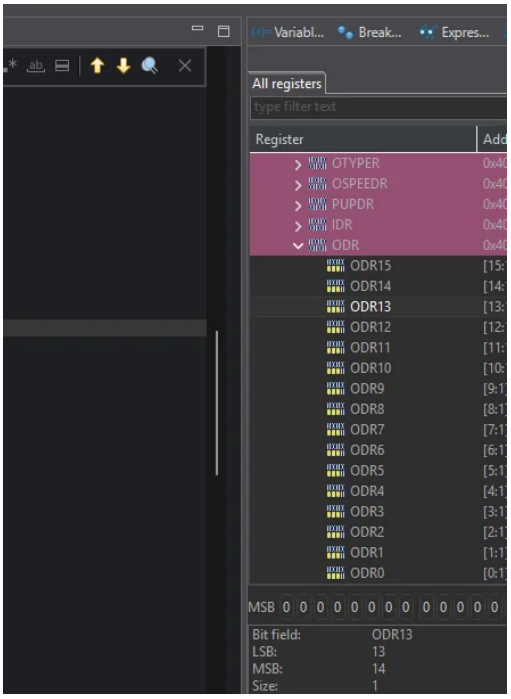
Register	Address	Purpose
GPIOC_MODER	0x40020800	Input/output/alternate/analog selection
GPIOC_IDR	0x40020810	Input data bits
GPIOC_ODR	0x40020814	Output data bits
GPIOC_BSRR	0x40020818	Atomic output set/reset

BSRR can set or reset GPIO bits without a read-modify-write sequence. STM32 HAL GPIO toggling commonly uses BSRR after reading the current ODR.

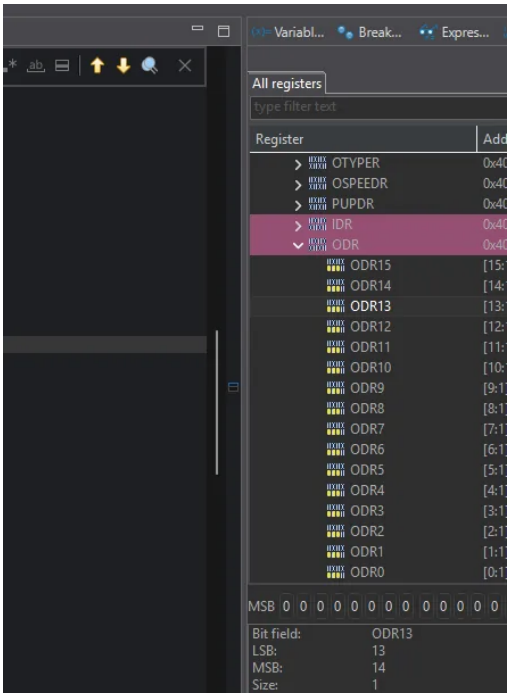
A moment captured in the debugger

The two cropped SFR screenshots below record the register-value change after stepping over the toggle call.

01 - Register before: ODR = 0x0, ODR13 = 0 (running-state view)



02 - Register after: ODR = 0x2000, ODR13 = 1 (CPU suspended)



What this experiment taught us

GPIO

PC13 is a physical output. MODER selects its mode; ODR holds output latch state; IDR samples pin state; BSRR provides set/reset actions.

The firmware lifecycle

Configure peripherals, generate initialization code, implement the application loop, build ELF, flash the MCU, verify, debug.

Debugging versus execution

The LED stops changing when the processor is halted. A breakpoint suspends the core; Step Over executes one source line; Resume restarts execution.

Binary place value

Bit 0 is least significant; bit 13 contributes 2^13 = 0x2000. The 32-bit ODR word has bit 31 as its highest-position bit (though GPIOC ODR bits 16-31 are reserved).

Field notation

The SFR field for ODR13 has size 1. Some debugger views show a boundary range LSB 13, MSB 14; the end boundary is exclusive.

A practical lesson

Do not assume the first while(1) search hit is in main(); Error_Handler() also has a loop. Keep user logic in marked USER CODE sections.

Next practical exercises

01 Direct-register PC13 blink (compare ODR and BSRR)

02 Pushbutton on an input pin with internal pull-up

03 External interrupt (EXTI) instead of polling

04 Hardware timer instead of HAL_Delay()

05 UART output for diagnostics

Published as part of The Thinking Workshop - a growing notebook of hands-on electronics, embedded programming, and experiments. Website: thethinkingworkshop.com