STM32 ???? ? 1 ?????? + Blink LED

STM32 ???? ? 1 ?????? + Blink LED

? ????

1. 完成 **STM32CubeIDE** 安裝與配置 - 建立開發環境，認識 IDE介面
2. 理解 **Nucleo-F446RC** 硬體 - 認識開發板的主要元件與接腳定義
3. 實現第一個程式 **- Blink LED** - 透過 GPIO 驅動 LED，驗證開發環境正確性

?? ????

????

- **STM32 Nucleo-F446RC** 開發板 × 1
- **USB Type-B** 傳輸線 × 1（隨開發板附贈）
- 麵包板 × 1（選配，後續課程使用）
- **LED + 220Ω** 電阻 × 若干（選配）
- 跳線 × 若干

Nucleo-F446RC ?????

Nucleo-F446RC 是 STMicroelectronics 推出的高性能開發板，搭載 **ARM Cortex-M4** 處理器。開發板已內建:

- **User LED**（綠色 **LED**，接在 **PA5**） - 這是我們第一個實驗的對象
- **User Button**（藍色按鈕，接在 **PC13**） - 後續課程會用到
- **ST-Link v2** 調試器 - 支援程式燒錄和實時調試
- 多個 **GPIO** 接腳 - 可外接各種感測器和執行器

????Blink LED?

組件	接腳	說明
User LED	PA5	開發板內建，直接使用
Ground	GND	參考地

> □□ 重要提示 - 開發板上的 User LED 已直接連接到 PA5 ，無需額外接線！這是最簡單的測試方案。

? ??????

?? 1???????

進入官方網站下載 STM32CubeIDE:

1. 訪問: https://www.st.com/en/development-tools/stm32cubeide.html
2. 點擊 **Download** 按鈕
3. 根據您的作業系統選擇（ Windows / Linux / macOS）
4. 註冊 ST 帳戶（如無則建立）
5. 下載最新版本（本教學基於 v1.13+）

?? 2??? STM32CubeIDE

Windows 安裝流程:

1. 雙擊下載的 `.exe` 安裝檔
2. 選擇安裝位置（建議 `C:\ST\STM32CubeIDE`）
3. 勾選 **Add STM32CubeIDE to PATH**（方便後續命令列操作）
4. 點擊 **Install** 並等待完成（約 5-10 分鐘）

Linux 安裝流程:

```
# 解壓縮下載的 tar.gz 檔案
tar -xzf STM32CubeIDE-*.tar.gz -C ~/opt/

# 進入安裝目錄執行安裝腳本
cd ~/opt/STM32CubeIDE-*/
```

```
./install.sh
```

?? 3?????? Workspace ??

1. 啟動 STM32CubeIDE
2. 選擇 Workspace 位置（例如：`D:\STM32_Workspace`）
3. 點擊 **Launch** 進入 IDE
4. 等待首次初始化（約 1-2 分鐘）
5. 關閉歡迎頁面

?? 4??????????

1. 用 USB 線連接 Nucleo-F446RC 到電腦
2. Windows 會自動下載並安裝 ST-Link 驅動
3. 打開 裝置管理員 檢查：
   - 應能看到 **STMicroelectronics STLink** 設備
   - 若標記 □ □□□□□□□□□ https://www.st.com/en/development-tools/stsw-link009.html
4. 在 STM32CubeIDE 中驗證連接：
   - 點擊 **Window → Preferences**
   - 選擇 **MCU → STMicroelectronics → STM32Cube**
   - 檢查 **ST-Link GDB server path** 是否正確識別

?? CubeMX ????

?? 1??????

1. 在 STM32CubeIDE 中點擊 **File → New → STM32 Project**
2. 搜尋裝置型號：輸入 **STM32F446RC**
3. 選擇 **STM32F446RCTx**
4. 點擊 **Next →** 設定專案名稱（例如：`STM32_Lesson01_BlinkLED`）
5. 選擇 **STM32CubeMX** 作為 Toolchain
6. 點擊 **Finish**

?? 2?CubeMX ??

STM32CubeIDE 會自動開啟 CubeMX 配置介面，您將看到晶片的 Pinout 圖。

GPIO 配置：

1. 在 Pinout 圖上找到 **PA5** (已標記為 User LED)
2. 確認其模式為 **GPIO_Output** (應已預設)
3. 若未設定, 右鍵點擊 PA5 → 選擇 **GPIO_Output**
4. 在左側 **System Core** 中選擇 **GPIO**
5. 展開 **GPIOA** 確認 PA5 的設定:
   - **GPIO output level**: High
   - **GPIO mode**: Output Push-Pull
   - **Pull**: No pull
   - **Maximum output speed**: High

時脈配置 (**Clock Tree**) :

1. 切換到 **Clock Configuration** 標籤
2. 確認以下設定:
   - **HSE (High Speed External)**: 8 MHz (開發板晶振頻率)
   - **System Clock Multiplier**: 配置為 **168 MHz** (F446RC 最大頻率)
   - **AHB Prescaler**: 1 (不分頻)
3. 查看底部確認無警告訊息

SysTick 配置:

1. 在左側選擇 **SysTick**
2. 確認 **Timebase** 設為 **SysTick**
3. 此設定用於系統計時, 後續課程會用到

?? 3??????

1. 點擊 **Project → Generate Code**
2. CubeMX 會根據配置產生初始化程式碼
3. 點擊 **Open Project** 返回 IDE

? ?????

main.c - Blink LED ??

```
/* STM32 Lesson 01 - Blink LED
 * 功能: 使用 GPIO 驅動 PA5 (User LED), 實現 1 秒間隔的閃爍
 * 難度: 初級
 */

#include "main.h"
```

```
#include "gpio.h"

/* 私有函數宣告 */
void SystemClock_Config(void);
static void MX_GPIO_Init(void);

int main(void)
{
  /* 重置所有外設並初始化時脈 */
  HAL_Init();

  /* 配置系統時脈為 168 MHz */
  SystemClock_Config();

  /* 初始化 GPIO */
  MX_GPIO_Init();

  /* 主迴圈 */
  while (1)
  {
    /* 設置 PA5 為高電位（LED 點亮）*/
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);

    /* 延遲 500 ms */
    HAL_Delay(500);

    /* 設置 PA5 為低電位（LED 熄滅）*/
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);

    /* 延遲 500 ms */
    HAL_Delay(500);
  }
}

/**
  * @brief 系統時脈配置函數
  * 配置主振盪器 (HSE) 倍頻到 168 MHz
  */
void SystemClock_Config(void)
{
```

```
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** 配置主內部調節器輸出電壓 */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);

  /** 初始化 RCC 振盪器 */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 8;
  RCC_OscInitStruct.PLL.PLLN = 336;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 7;

  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** 初始化 CPU、AHB 和 APB 匯流排時脈 */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK | RCC_CLOCKTYPE_SYSCLK
                              | RCC_CLOCKTYPE_PCLK1 | RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief GPIO 初始化函數
  * 配置 PA5 為輸出模式
  */
```

```
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO 埠時脈使能 */
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /* 配置 GPIO 腳位 PA5 */
  GPIO_InitStruct.Pin = GPIO_PIN_5;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
}

/**
  * @brief 錯誤處理函數
  * 當系統配置失敗時調用
  */
void Error_Handler(void)
{
  while (1)
  {
    /* 無限迴圈，系統掛起 */
  }
}
```

? ?????

????

 □□□□□□

- 開發板上的綠色 LED 以 **1** 秒間隔（0.5秒亮 + 0.5秒暗）穩定閃爍
- IDE 的 Console 顯示「**Build successful**」
- 編譯無警告（warnings）

?????????

問題	原因	解決方案
編譯失敗 「 undefined reference to HAL_xxx」	HAL 函式庫未正確連結	確保 CubeMX 已生成程式碼，檢查 Project → Properties → C/C++ Build → Libraries
燒錄失敗 「 Failed to connect to target 」	ST-Link 驅動未安裝或硬體連接不良	重新插拔 USB 線，檢查裝置管理員中的 ST-Link 設備
LED 不亮	GPIO 設定錯誤或接腳配置反向	檢查 CubeMX 中 PA5 是否設為 GPIO_Output，確認 GPIO_PIN_SET 為高電位
LED 常亮不閃爍	HAL_Delay 函數未工作	確認 SysTick 定時器已在 CubeMX 中配置

???????

1. 連接 Nucleo-F446RC 到電腦（ USB Type-B）
2. 在 IDE 中點擊 **Project** 選擇您的專案
3. 點擊工具列上的 **Build** 按鈕（錘子圖示）編譯
4. 等待編譯完成，確認無錯誤
5. 點擊 **Run** 按鈕（播放圖示）或按 **Ctrl+F11** 燒錄程式
6. 觀察開發板上的綠色 LED 是否開始閃爍

??/??????

□□ 如何驗證 ：

- 使用 邏輯分析儀 或 示波器 測量 PA5 腳位的信號
- 預期波形: 方波， 0V（ LED 暗） ↔ 3.3V（ LED 亮），週期 1 秒

? ????

?? Debug ??

1. 點擊工具列上的 **Debug** 按鈕（蟲子圖示）進入調試模式
2. 程式會自動暫停在 `main()` 函數開始處
3. 按 **F6** 或點擊 **Step Over** 逐行執行
4. 觀察 **Variables** 視窗中變數的值變化
5. 按 **F8** 或點擊 **Resume** 繼續執行

??????

在調試模式中:

1. 將滑鼠懸停在程式碼中的變數上，會顯示當前值
2. 在 **Breakpoints** 中設定中斷點（點擊程式碼行號）
3. 程式執行到中斷點時自動暫停

? ????

???? - ? 2 ????????

第 2 節將深入探討:

- 振盪器與倍頻原理 - 為何 STM32F446 能達到 168 MHz
- **Clock Tree** 詳解 - 各外設的時脈源選擇
- 電源管理模式 - Sleep、Stop、Standby 模式及其應用

????????

1. 修改閃爍頻率 - 將 `HAL_Delay(500)` 改為不同值，觀察 LED 閃爍速度變化
2. 使用按鈕控制 **LED** - 結合 User Button（PC13），按下時 LED 點亮
3. 呼吸燈效果（預告） - 使用 PWM 調變亮度，實現漸亮漸暗效果

??????

- □□ STM32CubeIDE 官方文檔____
- □□ STM32F446 資料表___
- □□ HAL 函式庫參考手冊_____
- □□ Nucleo-F446RC 使用者手冊_____

□ □□□□□□□ **1** 節，擁有完整的 **STM32** 開發環境!

下一步: 進入第 2 節，學習時脈配置的深層原理 □□

Revision #1
Created 2026-04-01 02:06:32 UTC by TaipeiTechRacing
Updated 2026-04-01 02:12:26 UTC by TaipeiTechRacing