DFU Dual Interface ISP

NuMicro® 32 位元系列微控制器範例代碼介紹

文件資訊

應用簡述	本範例程式使用 M252 USBD 實作具雙介面的 DFU 裝置，以更新 APROM 與 SPI Flash。
BSP 版本	M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.008
開發平台	NuMaker-M252KG V1.4 和外部 SPI Flash

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. 概述

此範例程式展示在 NuMicro M252 系列微控制器的內部 LDROM 執行的 USB DFU（Device Firmware Upgrade）ISP 實作。透過標準 USB DFU 介面，支援更新內部 APROM 與外部 SPI Flash 韌體。此範例程式需要以下額外硬體與軟體模組：

- 標準的 SPI Flash 包含四個訊號：/SS、CLK、MOSI、MISO。

  例如：Winbond W25Q32 SPI Flash。

- DFU 工具，用於韌體下載/上傳作業。

  https://dfu-util.sourceforge.net/

- Windows 驅動程式，提供 DFU 裝置識別所需的 USB 驅動。

1.1 原理

USB DFU 是一項 USB 標準，允許使用者透過 USB 連線更新裝置韌體。此範例程式支援雙介面的儲存後端（內部 APROM 與外部 SPI Flash），並與標準 DFU 工具（如 dfu-util）相容。

本範例程式自內部 LDROM 開機，之後可搭配 dfu-util 對 APROM 與 SPI Flash 進行下載（寫入）或上傳（讀出）。為符合 LDROM 容量（4 KB）的限制，本範例僅實現作為 DFU 相容裝置所需的最低限度 USB DFU 功能，並非完整實作整個協定。

圖 1-1 是本範例的簡易方塊圖；詳細的腳位連接說明請參見第 3.2 節。

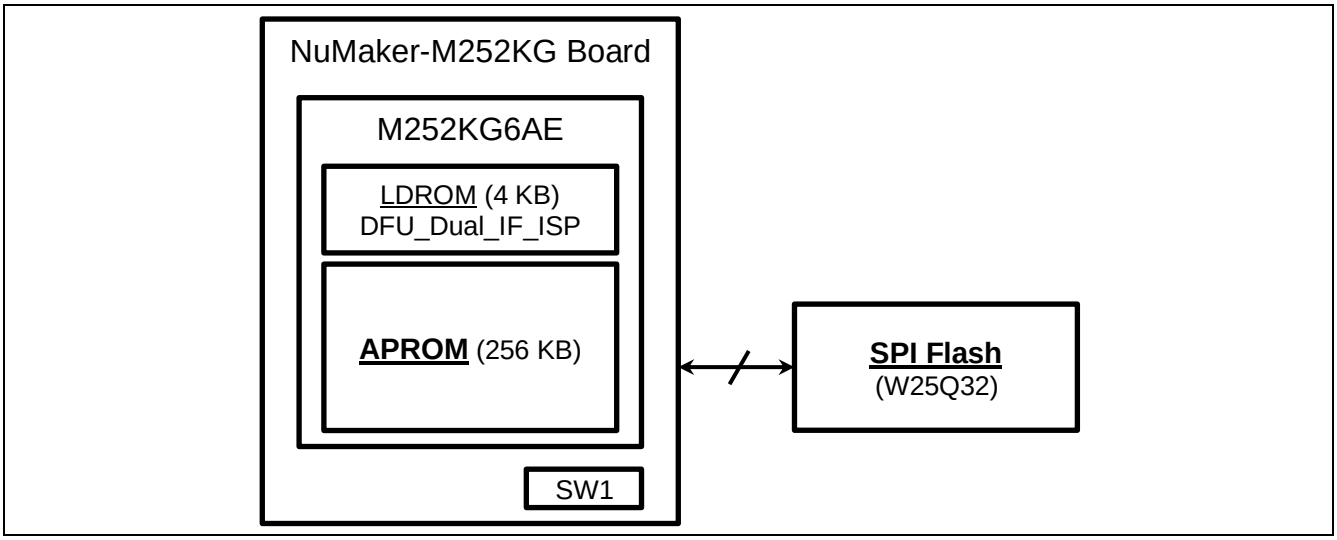


圖 1-1 硬體配置

1.2 執行結果

請參考圖 3-1，將 **SW1** 接至 VCC 並上電以進入 DFU ISP 模式。然後將 NuMaker-M252KG 的 USB 裝置埠連接到 Windows PC 的 USB 主機埠，以建立 DFU 連線。

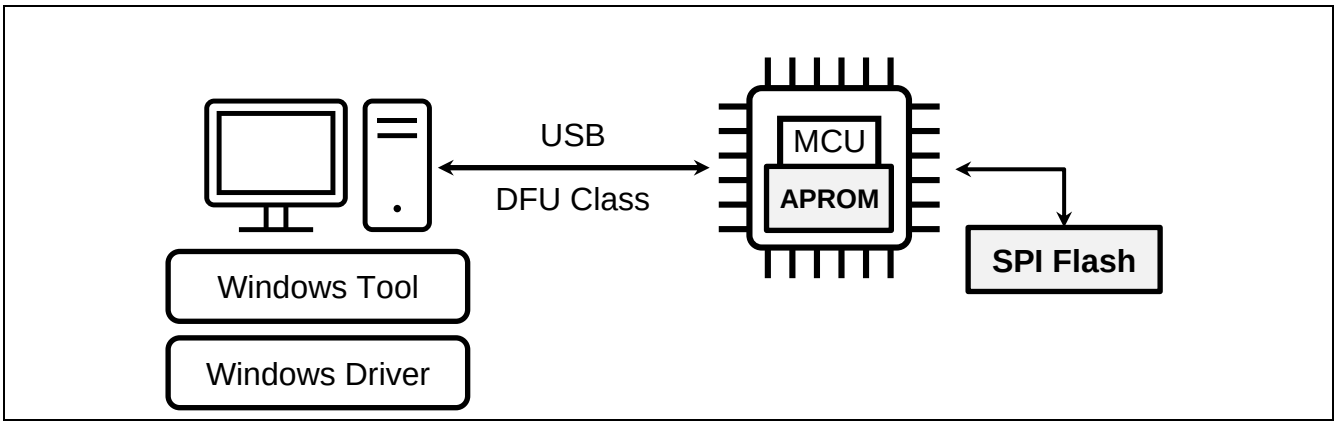


圖 1-2 USB DFU 連線

若首次在此電腦連接 Nuvoton DFU 裝置，Windows 可能無法辨識並將其標示為未知裝置。請依照 1.2.1 安裝 Nuvoton DFU 驅動程式後再使用該 DFU 裝置。

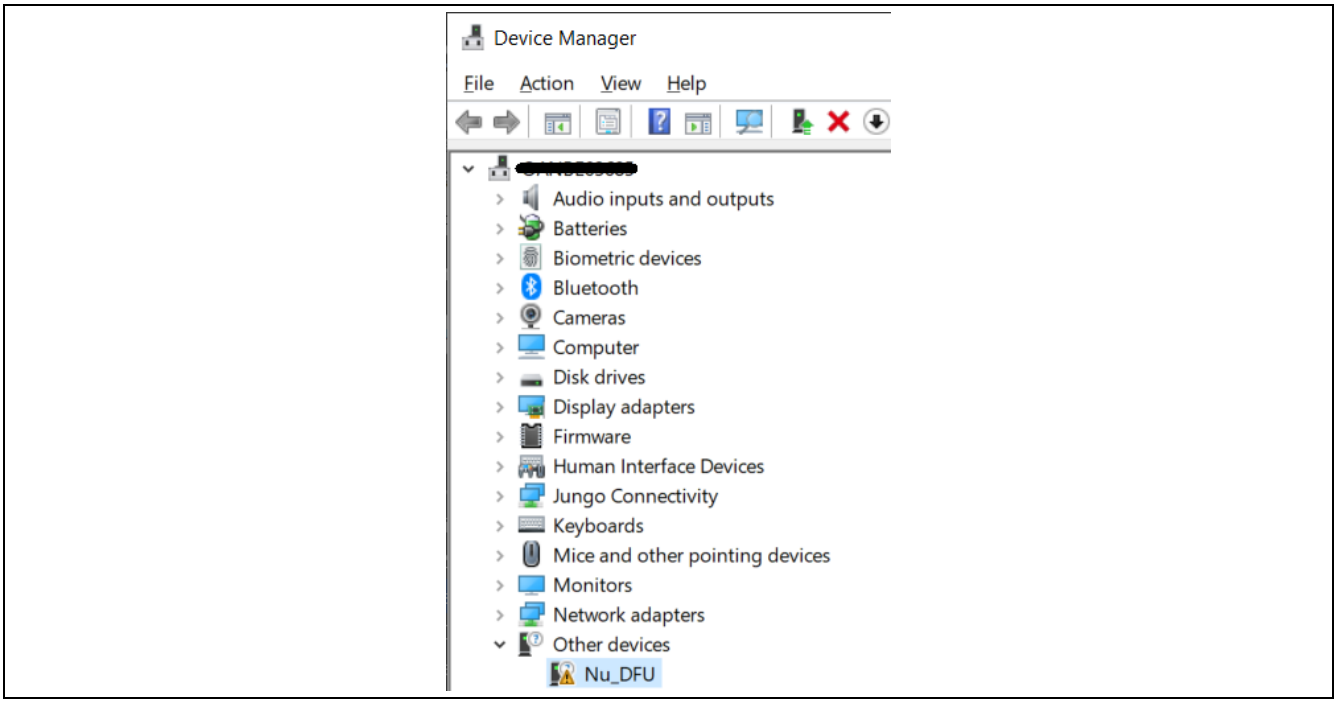
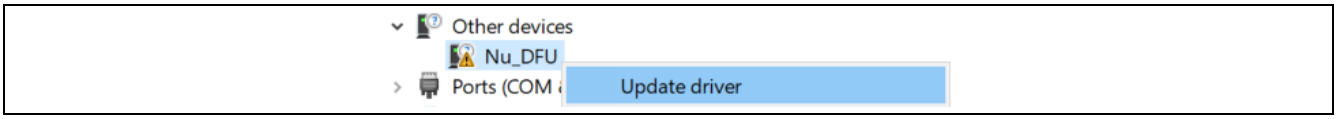


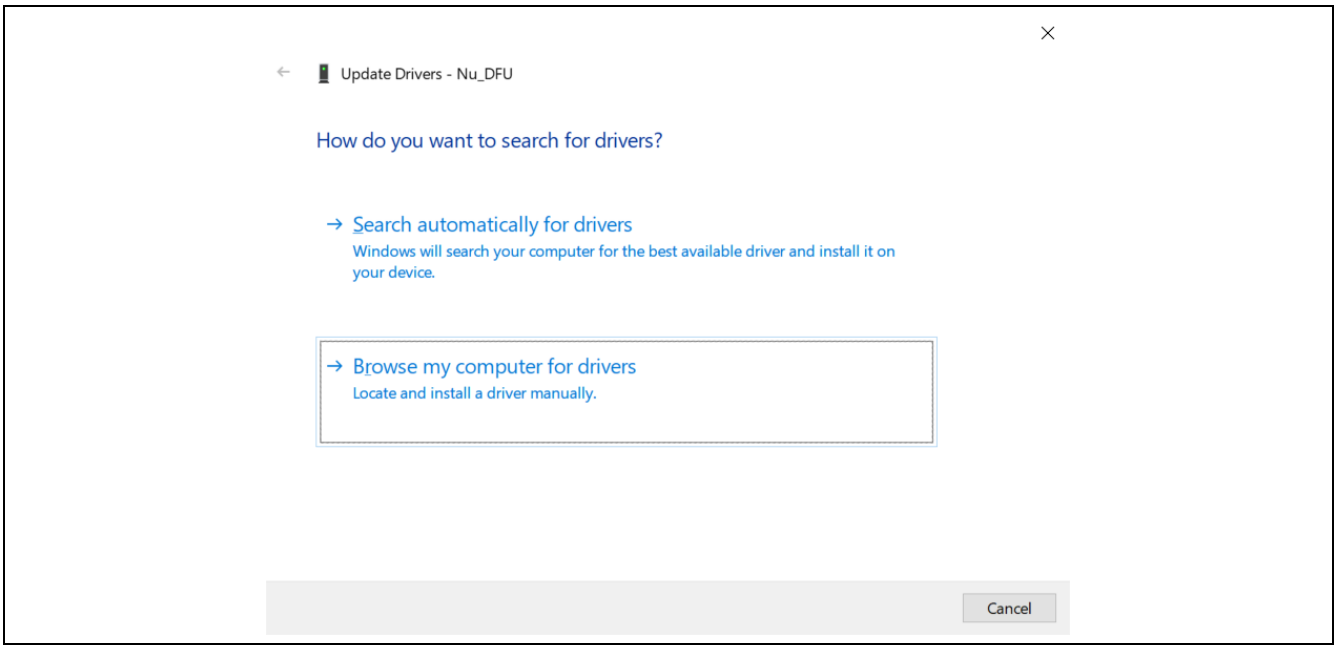
圖 1-3 裝置管理員中的未知裝置

1.2.1 安裝 Nuvoton DFU 驅動程式

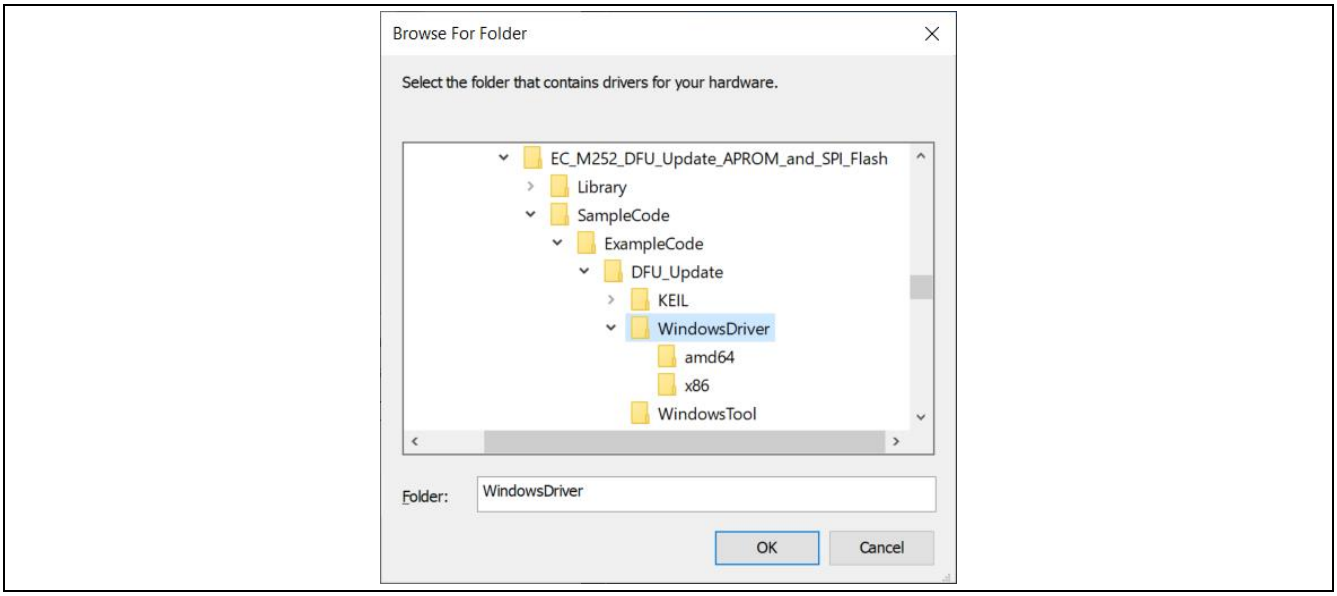
1. 在 Nu_DFU 裝置按右鍵選擇「**Update driver**」。



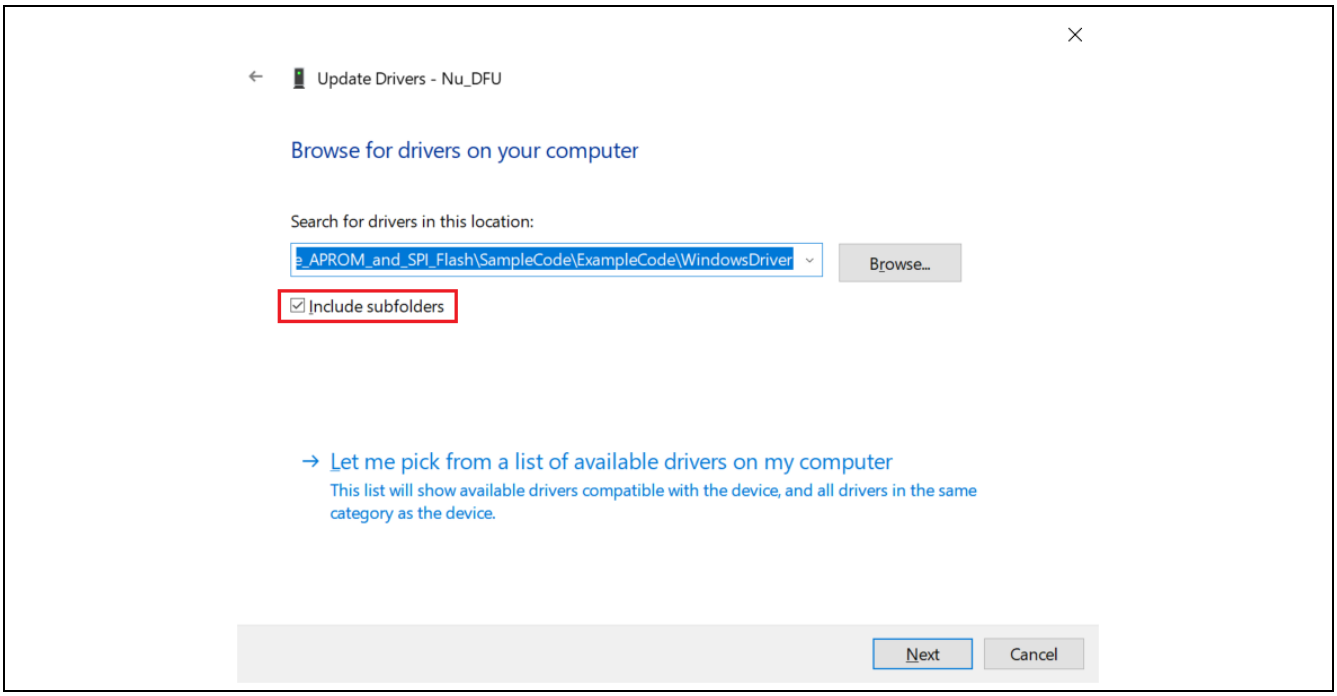
2. 選擇「Browse my computer for drivers」。



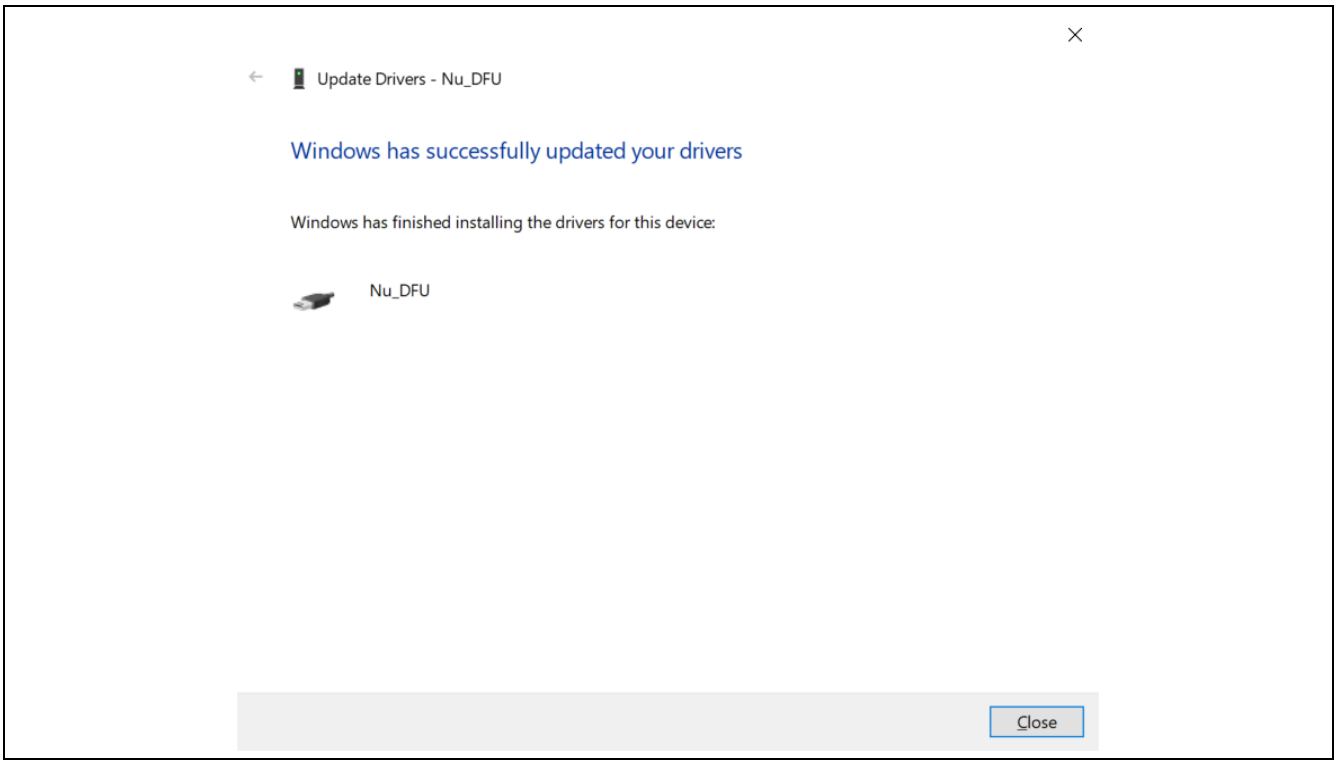
3. 查看目錄資訊章節選擇 **WindowsDriver** 目錄，點擊「**OK**」。



4. 勾選「Include subfolders」，點擊「**Next**」。



5. 確認安裝完成後點擊「**Close**」結束驅動程式安裝。



6. 從裝置管理員確認 **Nu_DFU** 已成功安裝。

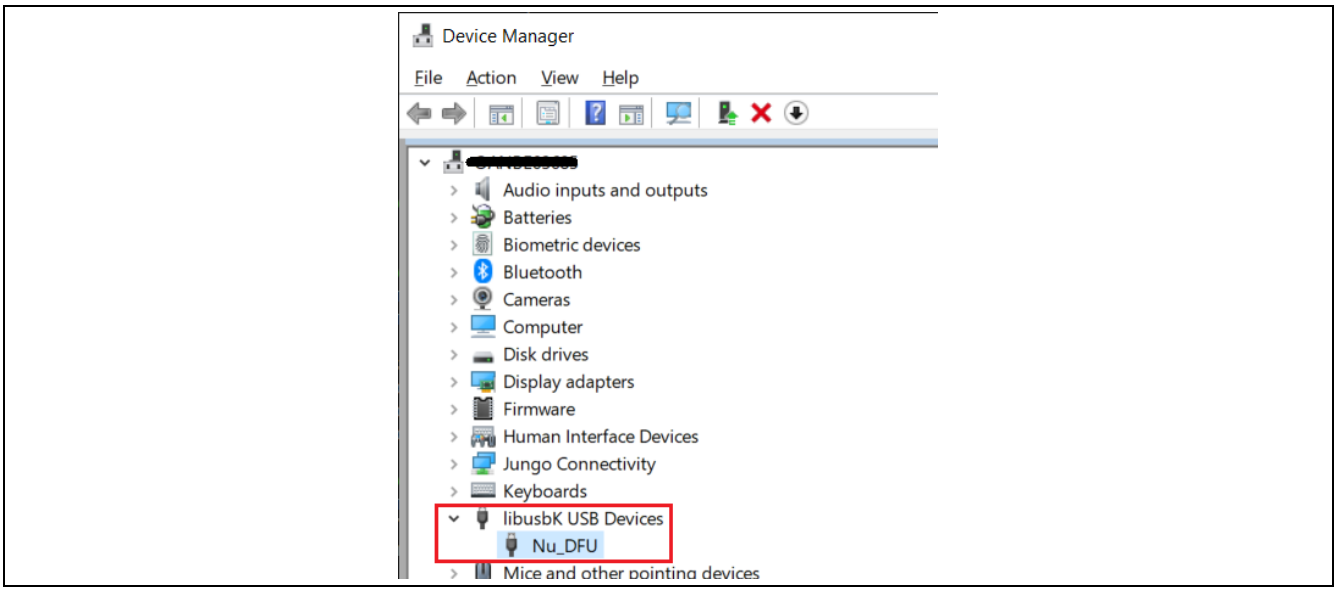
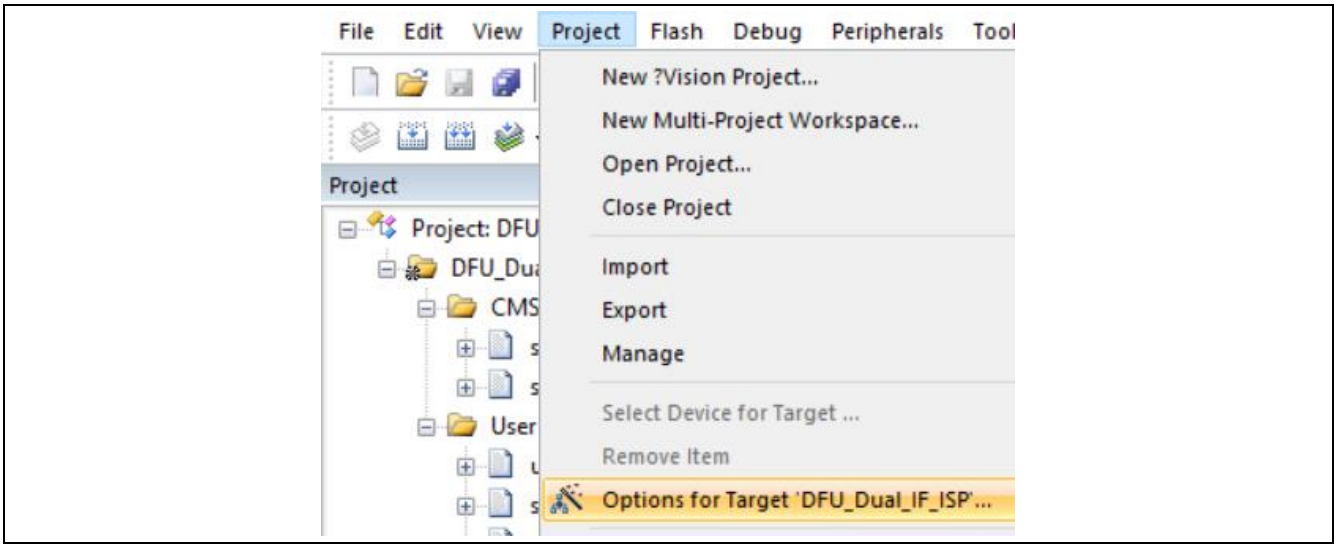


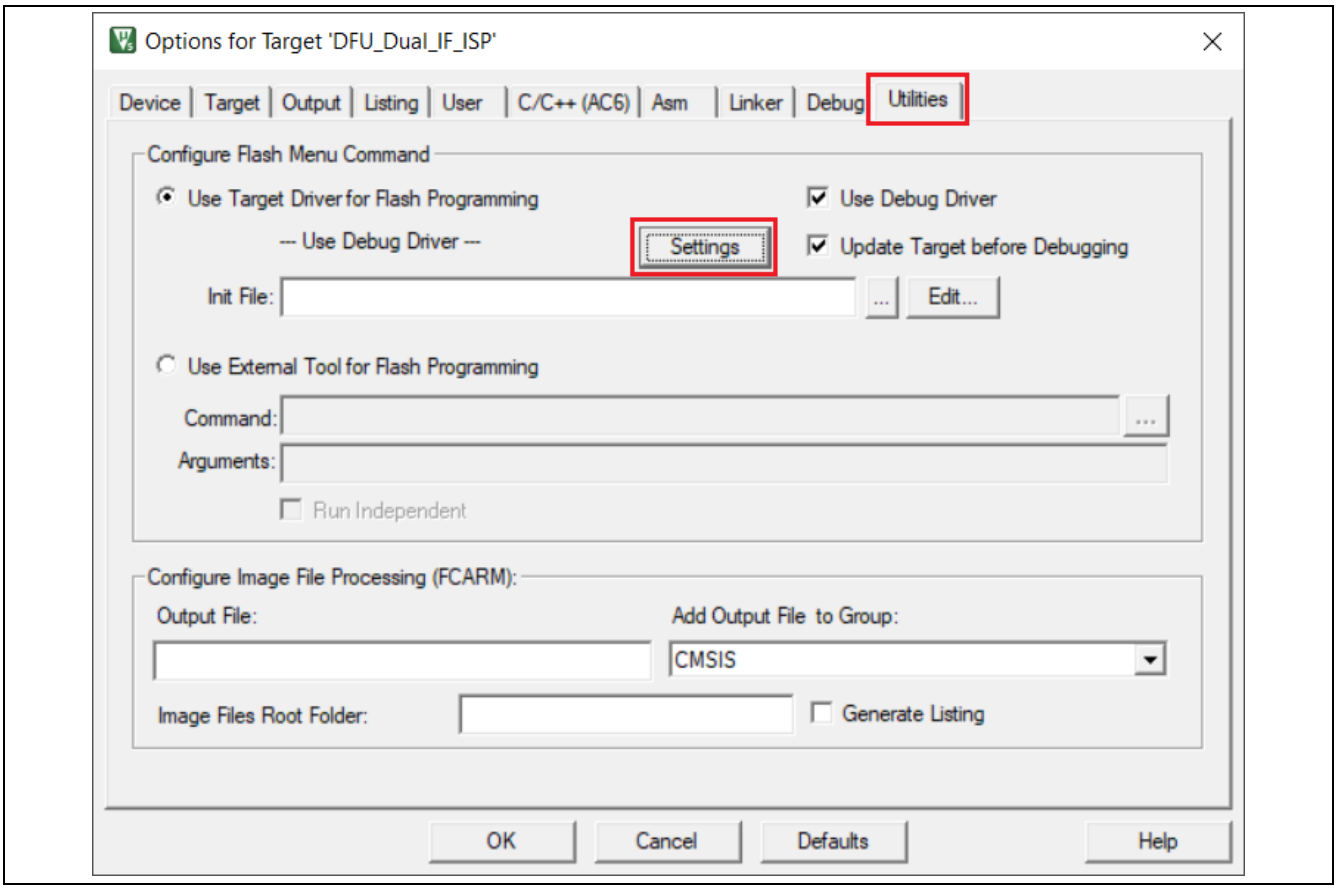
圖 1-4 裝置管理員中的 DFU 裝置

1.2.2 設定由 LDROM 開機

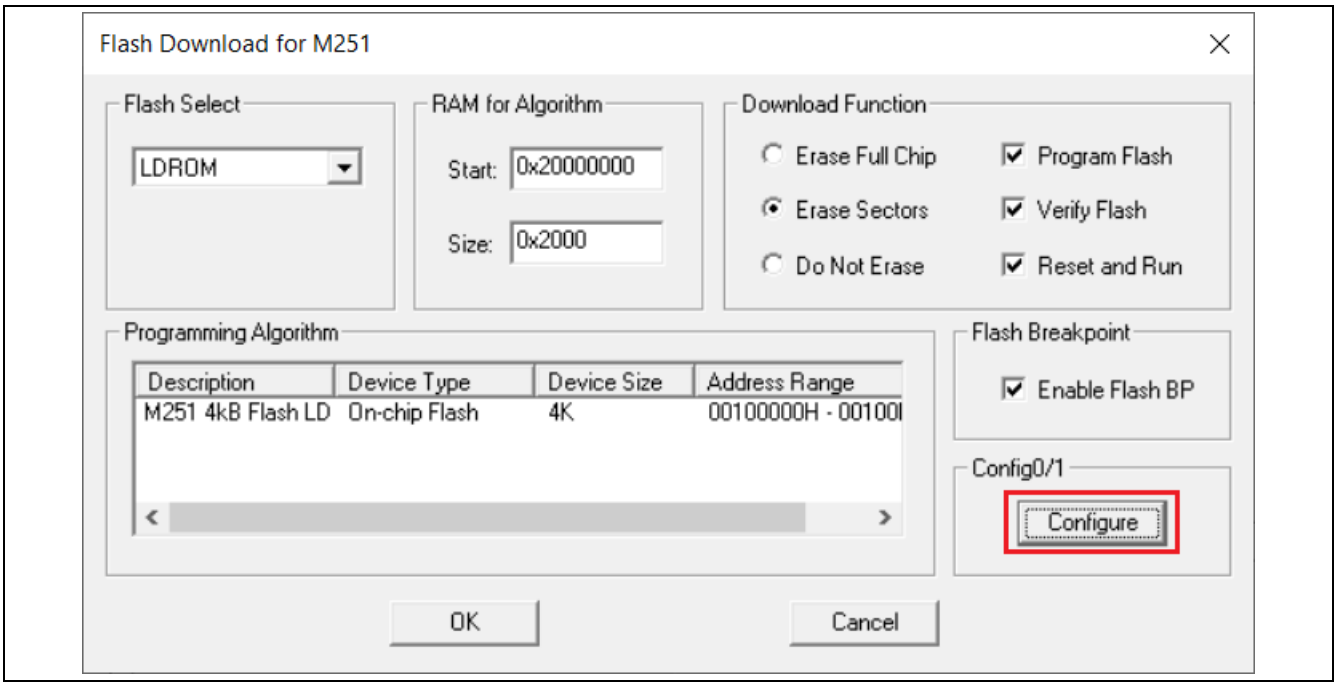
1. 前往「**Project**」選單，選擇「**Options for Target**」。



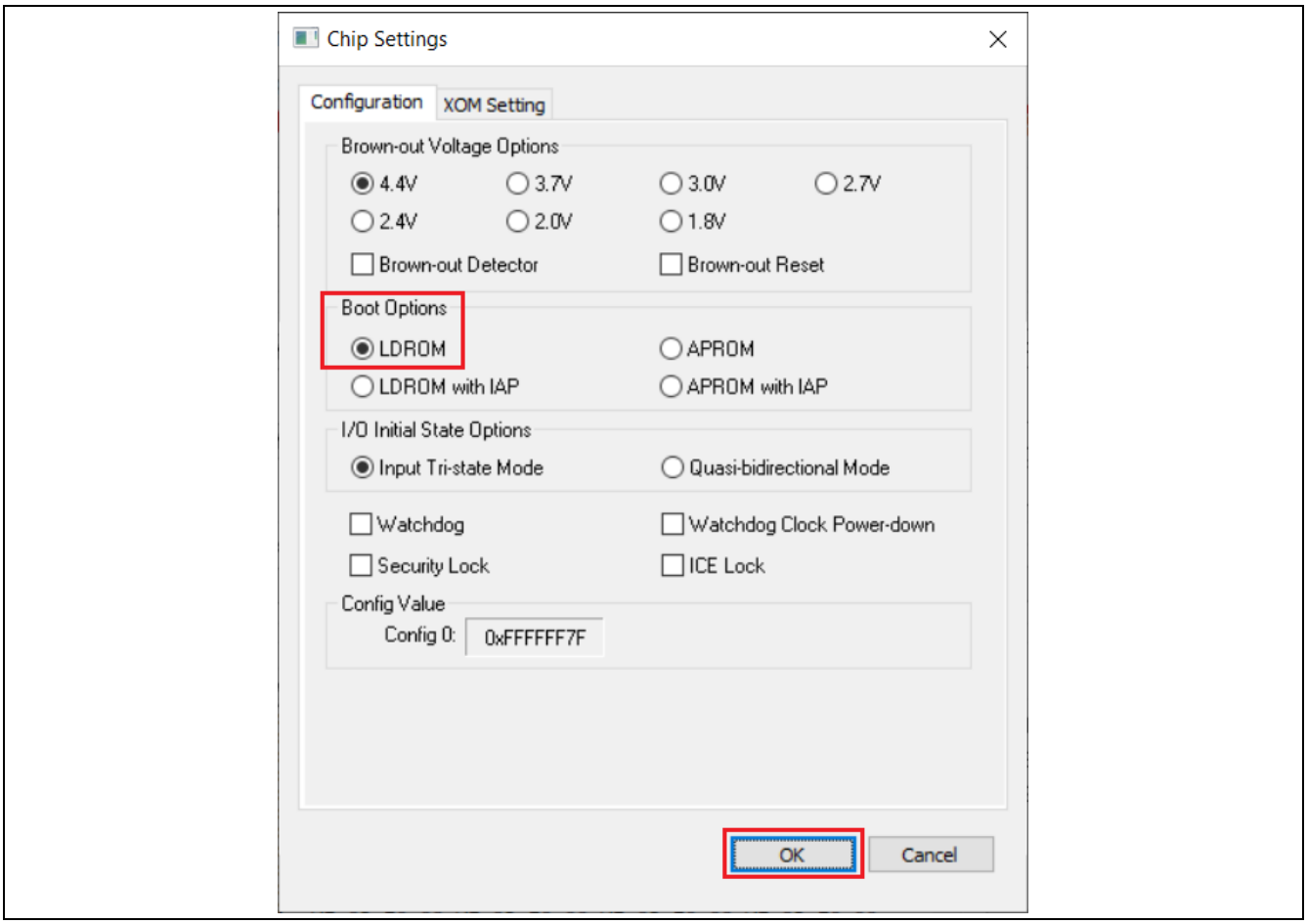
2. 移至「**Utilities**」分頁，點選「**Settings**」按鈕。



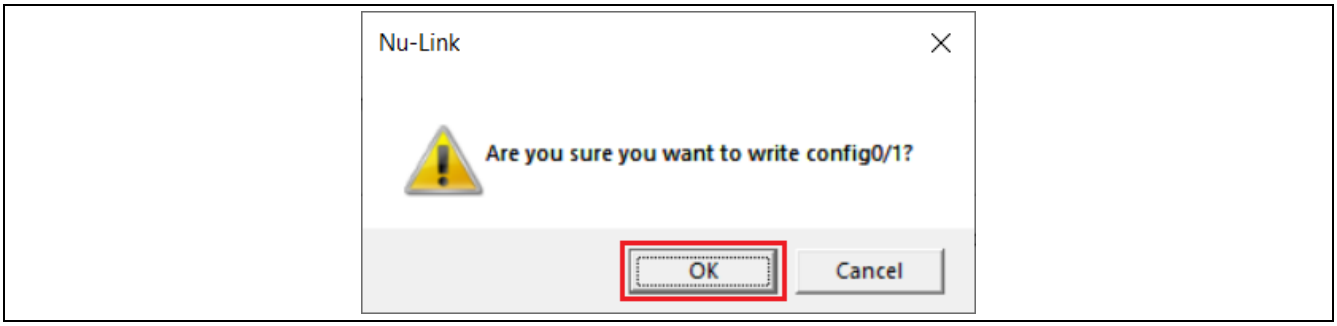
3. 點選「**Configure**」按鈕以開啟「**Chip Settings**」對話框。



4. 在「Boot Options」中選擇「**LDROM**」，並點選「**OK**」。



5. 點選「**OK**」以寫入設定。



6. 建置並將程式下載至 LDROM。

1.2.3 DFU 工具

dfu-util 是 USB DFU 通訊協定的開源主機端工具。它可以列出支援 DFU 的裝置、選擇目標（替代設定），並將韌體映像檔下載 / 上傳到嵌入式裝置的各種儲存區域。

請查看目錄資訊章節，在 WindowsTool 資料夾下取得 *dfu-util.exe*。

1.2.3.1 列出 DFU 裝置與目標

使用「dfu-util.exe -l」列出所有可用裝置。在下載或上傳韌體映像前，請先用此命令確認 DFU 裝置已就緒。

```
.\dfu-util.exe -l
dfu-util 0.11

Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.
Copyright 2010-2021 Tormod Volden and Stefan Schmidt
This program is Free Software and has ABSOLUTELY NO WARRANTY
Please report bugs to http://sourceforge.net/p/dfu-util/tickets/

Found DFU: [0416:bdf0] ver=0100, devnum=11, cfg=1, intf=0, path="1-1.1.1.3", alt=1,
name="@SPI Flash/0x0/1024*4Ke", serial="NVT_DFU"
Found DFU: [0416:bdf0] ver=0100, devnum=11, cfg=1, intf=0, path="1-1.1.1.3", alt=0,
name="@APROM/0x0/512*512Be", serial="NVT_DFU"
```

裝置透過替代設定（alt=0 與 alt=1）提供兩個 DFU 目標介面。

	APROM	SPI Flash
	name="@APROM/ 0x0/512*512Be"	name="@SPI Flash/ 0x0/1024*4Ke"
Alternate Setting	0	1
Base Address	0x0	0x0
Layout	512 x 512 B	1024 x 4 KB
Program/Erase Unit	512 B	4 KB
Capacity	256 KB	4 MB

1.2.3.2 APROM 下載和上傳

使用「dfu-util.exe -D」將二進位資料下載到指定的儲存區。

以下指令使用「-a 0」選擇 APROM，加上「-D .\aprom.bin」將 *aprom.bin* 透過 DFU 下載到 APROM。測試映像檔 *aprom.bin* 位於 WindowsTool 資料夾中，並可依需要替換為您自己的映像檔。

「-d 0x0416:0xBDF0」用於指定目標 DFU 裝置的廠商 / 產品 ID，若電腦僅連接一個 DFU 裝置，則可省略 -d 選項。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 0 -D .\aprom.bin
dfu-util 0.11
```

```
...

Opening DFU capable USB device...
Device ID 0416:bdf0
Device DFU version 0110
Claiming USB DFU Interface...
Setting Alternate Interface #0 ...
Determining device status...
DFU state(2) = dfuIDLE, status(0) = No error condition is present
DFU mode device DFU version 0110
Device returned transfer size 1024
Copying data from PC to DFU device
Download        [=========================] 100%       262144 bytes
Download done.
DFU state(2) = dfuIDLE, status(0) = No error condition is present
Done!
```

使用「dfu-util.exe -U」從指定的儲存區上傳二進位資料。

以下指令使用「-a 0」選擇 APROM，加上「-U .\read_aprom.bin」透過 DFU 從裝置上傳完整的 APROM 內容儲存至 *read_aprom.bin*。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 0 -U .\read_aprom.bin
dfu-util 0.11

...

Opening DFU capable USB device...
Device ID 0416:bdf0
Device DFU version 0110
Claiming USB DFU Interface...
Setting Alternate Interface #0 ...
Determining device status...
DFU state(2) = dfuIDLE, status(0) = No error condition is present
DFU mode device DFU version 0110
Device returned transfer size 1024
Copying data from DFU device to PC
Upload  [=========================] 100%       262144 bytes
Upload done.
Received a total of 262144 bytes
```

1.2.3.3 SPI Flash 下載和上傳

以下指令使用「-a 1」選擇 SPI Flash，加上「-D .\spi.bin」透過 DFU 將 *spi.bin* 下載到 SPI Flash。測試映像檔 *spi.bin* 位於 WindowsTool 資料夾中，並可依需要替換為您自己的映像檔。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 1 -D .\spi.bin
dfu-util 0.11

...
```

```
Opening DFU capable USB device...
Device ID 0416:bdf0
Device DFU version 0110
Claiming USB DFU Interface...
Setting Alternate Interface #1 ...
Determining device status...
DFU state(2) = dfuIDLE, status(0) = No error condition is present
DFU mode device DFU version 0110
Device returned transfer size 1024
Copying data from PC to DFU device
Download        [=========================] 100%      4194304 bytes
Download done.
DFU state(2) = dfuIDLE, status(0) = No error condition is present
Done!
```

以下指令使用「-a 1」選擇 SPI Flash，加上「-U .\read_spi.bin」透過 DFU 從裝置上傳完整的 SPI Flash 內容儲存至 *read_spi.bin*。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 1 -U .\read_spi.bin
dfu-util 0.11

...

Opening DFU capable USB device...
Device ID 0416:bdf0
Device DFU version 0110
Claiming USB DFU Interface...
Setting Alternate Interface #1 ...
Determining device status...
DFU state(2) = dfuIDLE, status(0) = No error condition is present
DFU mode device DFU version 0110
Device returned transfer size 1024
Copying data from DFU device to PC
Upload  [=========================] 100%      4194304 bytes
Upload done.
Received a total of 4194304 bytes
```

1.2.3.4 部分下載與上傳

使用「dfu-util.exe -s 偏移:大小」以位元組為單位指定起始偏移與資料大小。

以下指令將 *spi_1024KB.bin* 下載到 SPI Flash 的位移 0x10000；若省略大小參數，則會下載整個二進位檔。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 1 -s 0x10000 -D .\spi_1024KB.bin
dfu-util 0.11

...

Opening DFU capable USB device...
Device ID 0416:bdf0
Device DFU version 0110
```

```
Claiming USB DFU Interface...
Setting Alternate Interface #1 ...
Determining device status...
DFU state(2) = dfuIDLE, status(0) = No error condition is present
DFU mode device DFU version 0110
Warning: DfuSe option used on non-DfuSe device
Device returned transfer size 1024
DfuSe interface name: "SPI Flash"
Downloading element to address = 0x00010000, size = 1048576
Erase           [=========================] 100%      1048576 bytes
Erase    done.
Download        [=========================] 100%      1048576 bytes
Download done.
File downloaded successfully
```

以下指令從 SPI Flash 上傳部分資料區塊；自位移 0x10000 起上傳 0x100000 位元組，並將回傳的資料儲存至 *read_spi_1024KB.bin*。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 1 -s 0x10000:0x100000 -U .\read_spi_1024KB.bin
dfu-util 0.11

...

Opening DFU capable USB device...
Device ID 0416:bdf0
Device DFU version 0110
Claiming USB DFU Interface...
Setting Alternate Interface #1 ...
Determining device status...
DFU state(2) = dfuIDLE, status(0) = No error condition is present
DFU mode device DFU version 0110
Warning: DfuSe option used on non-DfuSe device
Device returned transfer size 1024
DfuSe interface name: "SPI Flash"
Upload  [=========================] 100%      1048576 bytes
Upload done.
```

1.2.3.5 下載程式到 APROM 並執行

要跳至 APROM 執行程式碼，請依照第 1.2.2 節設定由 LDROM 開機。

本範例在 WindowsTool 資料夾中包含測試映像檔 *Dump_SPI_Flash.bin*，用於印出 SPI Flash 中的資料。Dump_SPI_Flash 專案可在 ExampleCode 資料夾中找到。

請使用以下指令將 *Dump_SPI_Flash.bin* 下載到 APROM。

```
.\dfu-util.exe -d 0x0416:0xBDF0 -a 0 -D .\Dump_SPI_Flash.bin
```

下載完成後，將 **SW1** 接至 GND 以退出 DFU 模式。系統即會跳轉去執行 APROM 上的程式，並透過 UART 印出 SPI Flash 中的原始資料。

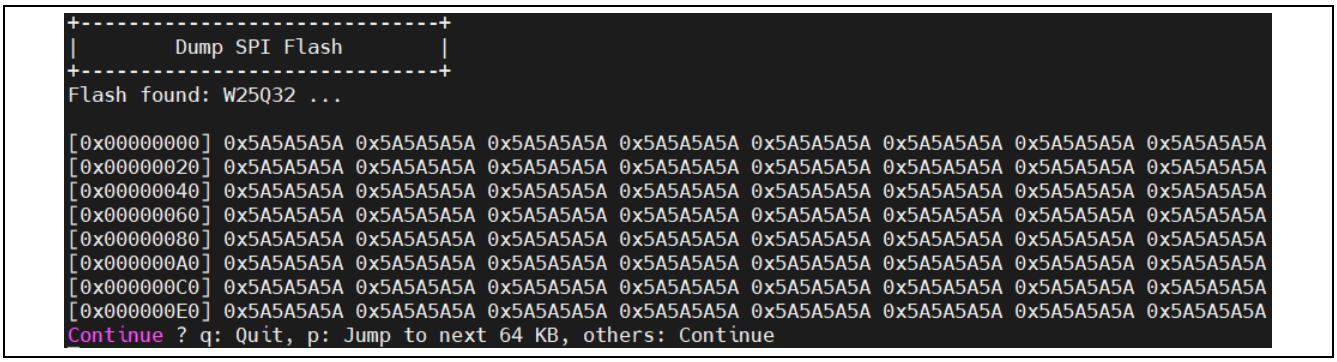


圖 1-5 SPI Flash 資料內容

2. 代碼介紹

在 *main.c* 的 SYS_Init() 中配置 USBD 和 SPI0 模組的時鐘以及多功能引腳設定。

```
void SYS_Init(void)
{
    /*---------------------------------------------------------------------------------------*/
    /* Init System Clock                                                                     */
    /*---------------------------------------------------------------------------------------*/
    /* Enable Internal HIRC 48 MHz clock */
    CLK->PWRCTL |= CLK_PWRCTL_HIRCEN_Msk;

    /* Waiting for Internal RC clock ready */
    while (!(CLK->STATUS & CLK_STATUS_HIRCSTB_Msk));

    /* Switch HCLK clock source to HIRC and HCLK source divided by (HCLK_DIV + 1),
       and USB clock source is from HIRC and divided by (USBD_DIV + 1) */
    CLK->CLKSEL0 = (CLK->CLKSEL0 & ~CLK_CLKSEL0_HCLKSEL_Msk) | CLK_CLKSEL0_HCLKSEL_HIRC;
    CLK->CLKDIV0 = (CLK->CLKDIV0 & ~CLK_CLKDIV0_HCLKDIV_Msk) | CLK_CLKDIV0_HCLK(HCLK_DIV);
    CLK->CLKDIV0 = (CLK->CLKDIV0 & ~CLK_CLKDIV0_USBDIV_Msk)  | CLK_CLKDIV0_USB(USBD_DIV);
```

此範例使用 PCLK1 作為 SPI 的時鐘來源，並以 PA0 作為 MOSI、PA1 作為 MISO、PA2 作為 CLK、PA3 作為 /SS 連接 SPI Flash。

```
    /* Select PCLK1 as the clock source of SPI0 */
    CLK->CLKSEL2 = (CLK->CLKSEL2 & ~CLK_CLKSEL2_SPI0SEL_Msk) | CLK_CLKSEL2_SPI0SEL_PCLK1;

    /* Enable module clock */
    CLK->APBCLK0 |= CLK_APBCLK0_USBDCKEN_Msk | CLK_APBCLK0_SPI0CKEN_Msk;
    CLK->AHBCLK  |= CLK_AHBCLK_ISPCKEN_Msk;

    /* Setup SPI0 multi-function pins */
    SYS->GPA_MFPL &= ~(SYS_GPA_MFPL_PA0MFP_Msk | SYS_GPA_MFPL_PA1MFP_Msk |
                       SYS_GPA_MFPL_PA2MFP_Msk | SYS_GPA_MFPL_PA3MFP_Msk);
    SYS->GPA_MFPL  = SYS_GPA_MFPL_PA0MFP_SPI0_MOSI | SYS_GPA_MFPL_PA1MFP_SPI0_MISO |
                     SYS_GPA_MFPL_PA2MFP_SPI0_CLK  | SYS_GPA_MFPL_PA3MFP_SPI0_SS;

    /* Enable SPI0 clock pin (PA2) schmitt trigger */
    PA->SMTEN |= GPIO_SMTEN_SMTEN2_Msk;
}
```

使用 PB4 作為偵測腳位，以控制系統停留在 DFU ISP 模式或跳轉至 APROM 開機。

```
#define DetectPin           PB4

...

int32_t main(void)
{
    /* The system stays in DFU ISP mode when DetectPin is high and
       boots to APROM when DetectPin is low. */
    while (DetectPin != 0)
    {
        ...
```

```
    }
}
```

請查閱 MCU 與 SPI Flash 的資料手冊，並在 *descriptors.c* 中的 g_au8Alt0StringDesc 與 g_au8Alt1StringDesc 描述子中指定 APROM 與 SPI Flash 的大小與擦除單位。

注意：USB 字串描述子的字串資料使用 UTF-16LE (小端序) 編碼，對於標準 ASCII 字元，其高位元組通常為 0。

```
/* APROM
 *   Base address: 0x0
 *   Erase unit: 512B
 *   Size: 512 * 512B = 256KB
 */
const uint8_t g_au8Alt0StringDesc[] = {
    42, DESC_STRING,
    '@', 0, 'A', 0, 'P', 0, 'R', 0, 'O', 0, 'M', 0, '/', 0,
    '0', 0, 'x', 0, '0', 0, '/', 0,
    '5', 0, '1', 0, '2', 0, '*', 0, '5', 0, '1', 0, '2', 0, 'B', 0,
    // e: Readable and Writeable
    'e', 0
};

/* SPI Flash
 *   Base address: 0x0
 *   Erase unit: 4KB
 *   Size: 1024 * 4KB = 4MB
 */
const uint8_t g_au8Alt1StringDesc[] = {
    48, DESC_STRING,
    '@', 0, 'S', 0, 'P', 0, 'I', 0, ' ', 0, 'F', 0, 'l', 0, 'a', 0, 's', 0, 'h', 0, '/',
0,
    '0', 0, 'x', 0, '0', 0, '/', 0,
    '1', 0, '0', 0, '2', 0, '4', 0, '*', 0, '4', 0, 'K', 0,
    // e: Readable and Writeable
    'e', 0
};
```

請依照 SPI Flash 的資料手冊，在 *cmd_handle.h* 中定義 SPI 相關資訊。

```
#define SPI_FLASH_PORT          SPI0
// SPI information depends on target SPI Flash.
#define SPI_FLASH_ID            0xEF15
#define SPI_FLASH_SECTOR_SIZE   4096
#define SPI_FLASH_PAGE_SIZE     256
#define SPI_FLASH_SIZE          0x400000
```

DFU 的下載與上傳請求由 *cmd_handle.c* 中的 WriteData 和 ReadData 函式處理。USBD_GetAltSetting() 用於判定資料傳輸的目標儲存區。

```
// Read data from APROM or SPI Flash
int ReadData(uint32_t u32StartAddr, uint32_t u32EndAddr, uint32_t *pu32Data)
{
    // Get the target storage
    uint32_t u32AltSetting = USBD_GetAltSetting();
```

```
    if (u32AltSetting == DFU_INTF_APROM)
    {
        return FMC_Proc(FMC_ISPCMD_READ, u32StartAddr, u32EndAddr, pu32Data);
    }
    else if (u32AltSetting == DFU_INTF_SPI_FLASH)
    {
        return SPI_Proc(FMC_ISPCMD_READ, u32StartAddr, u32EndAddr, pu32Data);
    }

    return -1;
}

// Write data into APROM or SPI Flash
int WriteData(uint32_t u32StartAddr, uint32_t u32EndAddr, uint32_t *pu32Data)
{
    // Get the target storage
    uint32_t u32AltSetting = USBD_GetAltSetting();

    if (u32AltSetting == DFU_INTF_APROM)
    {
        return FMC_Proc(FMC_ISPCMD_PROGRAM, u32StartAddr, u32EndAddr, pu32Data);
    }
    else if (u32AltSetting == DFU_INTF_SPI_FLASH)
    {
        return SPI_Proc(FMC_ISPCMD_PROGRAM, u32StartAddr, u32EndAddr, pu32Data);
    }

    return -1;
}
```

SPI_Proc() 負責處理 SPI Flash 的讀取、寫入與擦除操作。

```
int SPI_Proc(uint32_t u32Cmd, uint32_t u32StartAddr, uint32_t u32EndAddr,
             uint32_t *pu32Data)
{
    uint32_t u32Addr;

    if (u32StartAddr >= SPI_FLASH_SIZE)
        return -1;
```

呼叫 SPI_Open() 時會指定 SPI 通訊參數與時鐘頻率。

```
    /* Configure SPI_FLASH_PORT as a master, MSB first, 8-bit transaction,
       SPI Mode-0 timing, clock is 4 MHz */
    SPI_Open(SPI_FLASH_PORT, SPI_MASTER, SPI_MODE_0, 8, FREQ_4MHZ);
```

檢查 SPI Flash 的裝置 ID 是否符合預期；若不符合，則回報錯誤。

```
    /* Check SPI Flash ID */
    if (SpiFlash_ReadMidDid() != SPI_FLASH_ID)
    {
        return -1;
    }
```

根據 u32Cmd 處理 SPI Flash 操作：

- 從起始位址到結束位址按頁讀取資料。
- 按頁寫入資料，並自動處理扇區擦除。

```
    if (u32Cmd == FMC_ISPCMD_READ)
    {
        for (u32Addr = u32StartAddr; u32Addr < u32EndAddr; u32Addr += SPI_FLASH_PAGE_SIZE)
        {
            SpiFlash_ReadPage(u32Addr, (uint8_t *)pu32Data);
            pu32Data = (uint32_t *)((uint32_t)pu32Data + SPI_FLASH_PAGE_SIZE);
        }
    }
    else
    {
        for (u32Addr = u32StartAddr; u32Addr < u32EndAddr; u32Addr += SPI_FLASH_PAGE_SIZE)
        {
            /* Erase flash by sector before write data */
            if ((u32Addr % SPI_FLASH_SECTOR_SIZE) == 0)
            {
                SpiFlash_EraseSector(u32Addr);
                SpiFlash_WaitReady();
            }

            SpiFlash_WritePage(u32Addr, (uint8_t *)pu32Data);
            SpiFlash_WaitReady();
            pu32Data = (uint32_t *)((uint32_t)pu32Data + SPI_FLASH_PAGE_SIZE);
        }
    }
}
```

spi_user.h 定義支援基本的 SPI Flash 讀取 / 寫入 / 擦除指令。

```
#define SPI_FLASH_WRITE_STATUS  0x01
#define SPI_FLASH_PROG_PAGE     0x02
#define SPI_FLASH_READ          0x03
#define SPI_FLASH_READ_STATUS   0x05
#define SPI_FLASH_WRITE_ENABLE  0x06
#define SPI_FLASH_ERASE_SECTOR  0x20
#define SPI_FLASH_READ_ID       0x90
#define SPI_FLASH_ERASE_CHIP    0xC7
```

dfu_transfer.c 中的 DFU_ClassRequest() 處理 DFU 狀態轉換的最低需求。

當主機發出資料傳輸請求時會呼叫 WriteData() 和 ReadData()，若傳輸失敗則回報錯誤。

```
void DFU_ClassRequest(void)
{
    ...

    if (buf[0] & 0x80)
    {
        /* Device-to-Host */
```

```
        switch (buf[1])
        {
            case DFU_GETSTATUS:
            {
                if (dfu_status.bState == STATE_dfuDNLOAD_SYNC)
                {
                    dfu_status.bState = STATE_dfuDNLOAD_IDLE;

                    if ((prog_struct.data_len > 0) &&
                        /* Write data to request offset of target storage */
                        WriteData(
                            prog_struct.u32Offset + prog_struct.block_num * TRANSFER_SIZE,
                            prog_struct.data_len,
                            (uint32_t *)prog_struct.buf) != 0
                    )
                    {
                        dfu_status.bStatus = STATUS_errWRITE;
                        dfu_status.bState  = STATE_dfuERROR;
                    }
                }
        ...
    }
    else
    {
        // Host-to-Device
        switch (buf[1])
        {
            ...

            case DFU_UPLOAD:
            {
                if (dfu_status.bState == STATE_dfuIDLE || dfu_status.bState ==
STATE_dfuUPLOAD_IDLE)
                {
                    if (wLength == 0)
                    {
                        dfu_status.bState = STATE_dfuIDLE;
                        return;
                    }
                    else
                    {
                        if (dfu_status.bState == STATE_dfuIDLE)
                        {
                            dfu_status.bState = STATE_dfuUPLOAD_IDLE;
                        }

                        /* Read data from request offset of target storage */
                        if (ReadData(
                                prog_struct.u32Offset + wValue * TRANSFER_SIZE,
                                wLength,
                                (uint32_t *)prog_struct.buf) < 0
                        )
                        {
                            dfu_status.bState = STATE_dfuIDLE;
                            USBD_PrepareCtrlIn(0, 0);
                        }
```

```
        ...
    }
}
```

3. 軟體與硬體需求

3.1 軟體需求

- BSP 版本
  - M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.008
- IDE 版本
  - Keil uVision 5.43.0
- DFU 工具
  - dfu-util.exe
- Windows 驅動程式

3.2 硬體需求

- 電路元件
  - NuMaker-M252KG V1.4
  - Winbond W25Q32 SPI Flash (Or compatible SPI Flash)
- 線路示意圖
  - 將外部 SPI Flash 依圖 3-1 所示連接至 NuMaker-M252KG 的 SPI0。
  - 將 SW1（PB.4）接至 VCC 以進入 DFU ISP 模式，或接至 VSS 以從 APROM 開機。
  - 將 NuMaker-M252KG 的 USB device 埠連接至 PC 的 USB host，以建立 DFU 連線。
  - 將 UART0 TX（PB.13）腳位連接到 PC 的 UART RX 以顯示 SPI Flash 的資料傾印結果，並將 UART0 RX（PB.12）腳位連接到 PC 的 UART TX 以選擇選單項目。

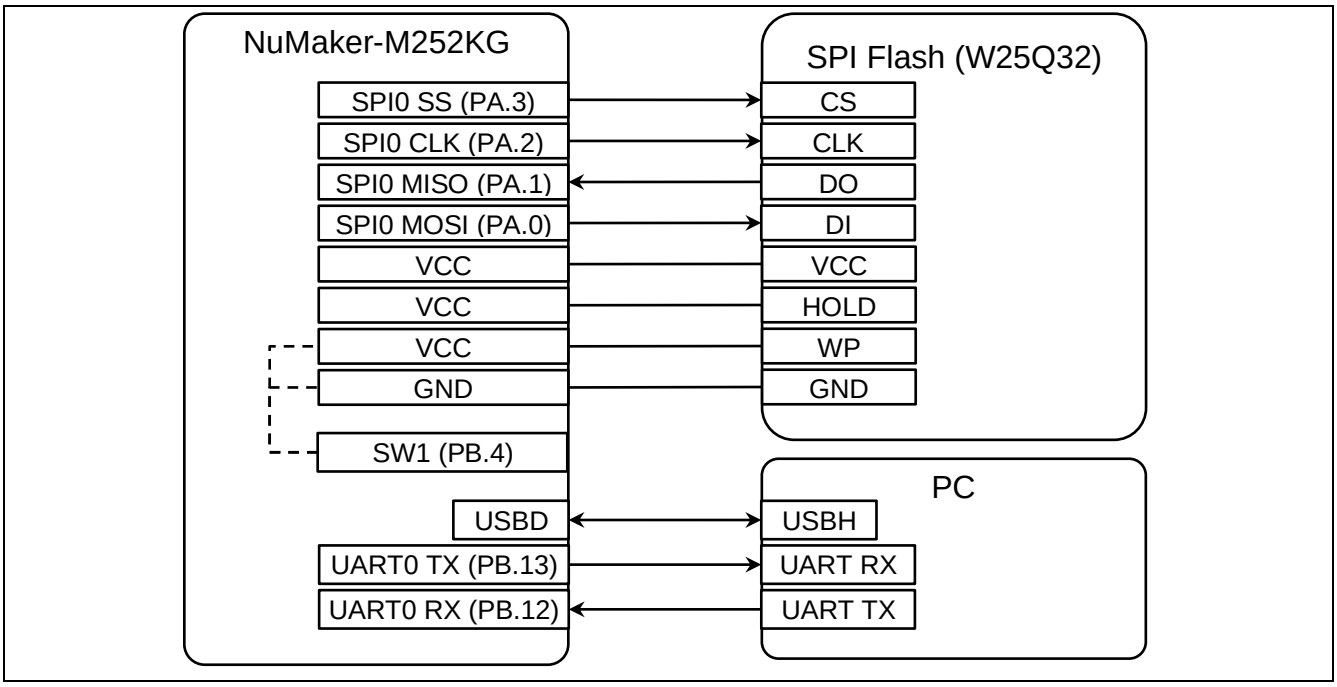

NuMaker-M252KG
SPI0 SS (PA.3)
SPI0 CLK (PA.2)
SPI0 MISO (PA.1)
SPI0 MOSI (PA.0)
VCC
VCC
VCC
GND
SW1 (PB.4)
USBD
UART0 TX (PB.13)
UART0 RX (PB.12)
SPI Flash (W25Q32)
CS
CLK
DO
DI
VCC
HOLD
WP
GND
PC
USBH
UART RX
UART TX

圖 3-1 線路示意圖

4. 目錄資訊

目錄結構如下圖所示。

EC_M252_DFU_Dual_Interface_ISP	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	
DFU_Dual_IF_ISP	Source files of example code
WindowsDriver	USB drivers for Nuvoton DFU device
WindowsTool	dfu-util.exe for firmware download/upload operations
Dump_SPI_Flash	Example code to dump the SPI Flash

圖 4-1 目錄資訊

5. 範例程式執行

1. 根據目錄資訊章節進入 DFU_Dual_IF_ISP 路徑中的 KEIL 資料夾，雙擊 *DFU_Dual_IF_ISP.uvprojx*。
2. 進入編譯模式介面
   - 編譯
   - 下載代碼至記憶體
   - 進入 / 離開除錯模式
3. 進入除錯模式介面
   - 執行代碼

6. 修訂紀錄

Date	Revision	Description
2025.10.15	1.00	初始發布。

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.