DFU Dual Interface ISP

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

应用简述	本范例程序使用 M252 USBD 实作具双接口的 DFU 装置，以更新 APROM 和 SPI Flash。
BSP 版本	M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.008
开发平台	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

此范例程序展示在 NuMicro M252 系列微控制器的内部 LDROM 执行的 USB DFU（Device Firmware Upgrade）ISP 实作。透过标准 USB DFU 接口，支持更新内部 APROM 与外部 SPI Flash 固件。此范例程序需要以下额外硬件与软件模块：

- 标准的 SPI Flash 包含四个信号：/SS、CLK、MOSI、MISO。

  例如：Winbond W25Q32 SPI Flash。

- DFU 工具，用于固件下载/上传作业。

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

- Windows 驱动程序，提供 DFU 装置识别所需的 USB 驱动。

1.1 原理

USB DFU 是一项 USB 标准，允许用户透过 USB 联机更新装置固件。此范例程序支持双接口的储存后端（内部 APROM 与外部 SPI Flash），并与标准 DFU 工具（如 dfu-util）兼容。

本范例程序自内部 LDROM 开机，之后可搭配 dfu-util 对 APROM 与 SPI Flash 进行下载（写入）或上传（读出）。为符合 LDROM 容量（4 KB）的限制，本范例仅实现作为 DFU 兼容装置所需的最低限度 USB DFU 功能，并非完整实作整个协议。

图 1-1 是本范例的简易方框图；详细的脚位连接说明请参见第 3.2 节。

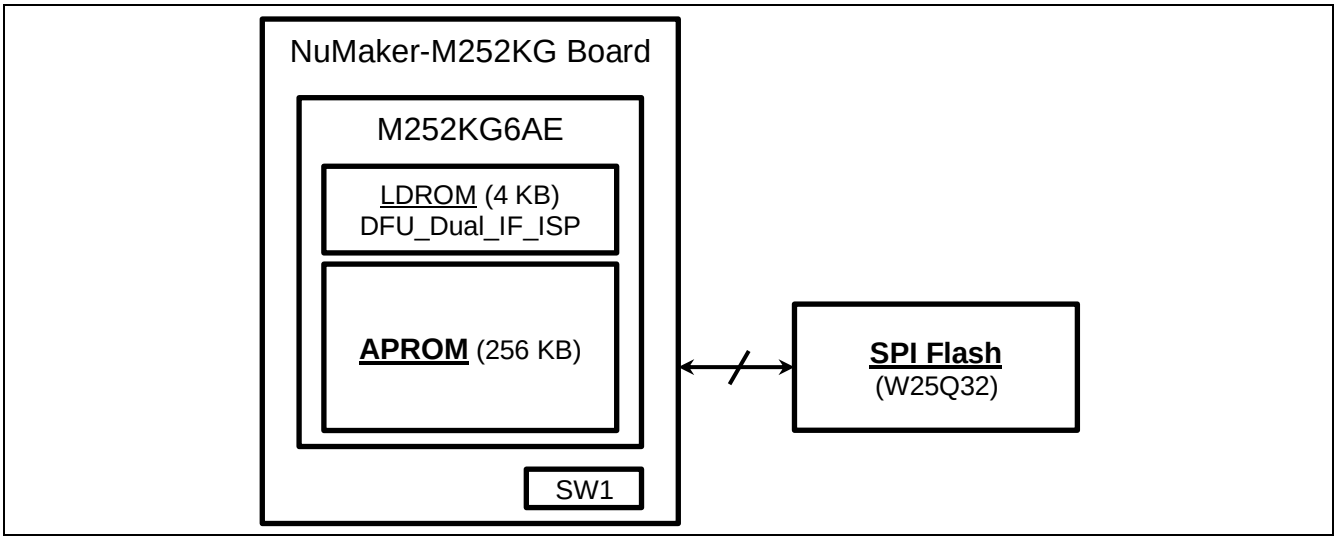


图 1-1 硬件配置

1.2 执行结果

请参考图 3-1，将 **SW1** 接至 VCC 并上电以进入 DFU ISP 模式。然后将 NuMaker-M252KG 的 USB 装置端口连接到 Windows PC 的 USB 主机端口，以建立 DFU 联机。

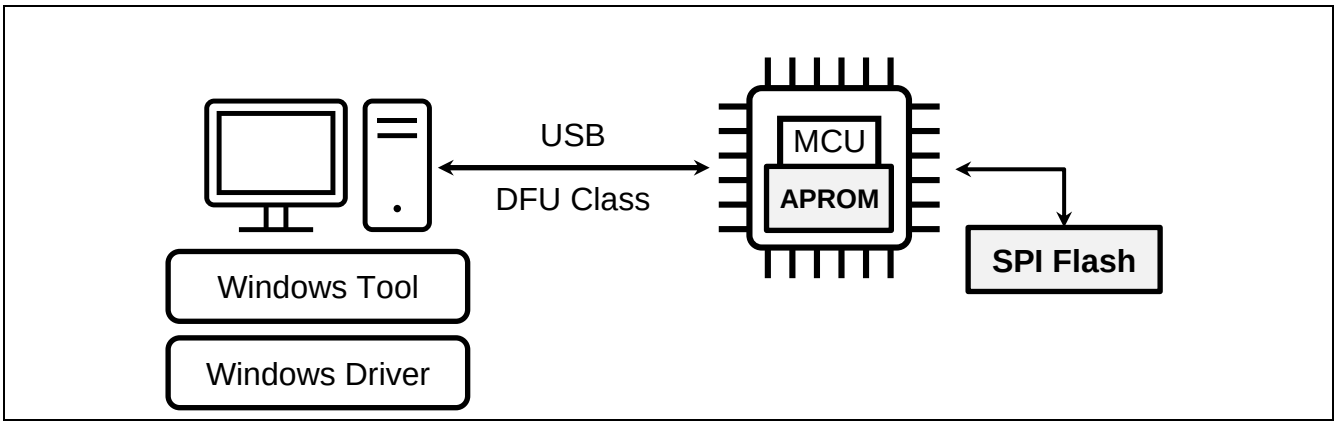


图 1-2 USB DFU 联机

若首次在此计算机连接 Nuvoton DFU 装置，Windows 可能无法辨识并将其标示为未知装置。请依照 1.2.1 安装 Nuvoton DFU 驱动程序后再使用该 DFU 装置。

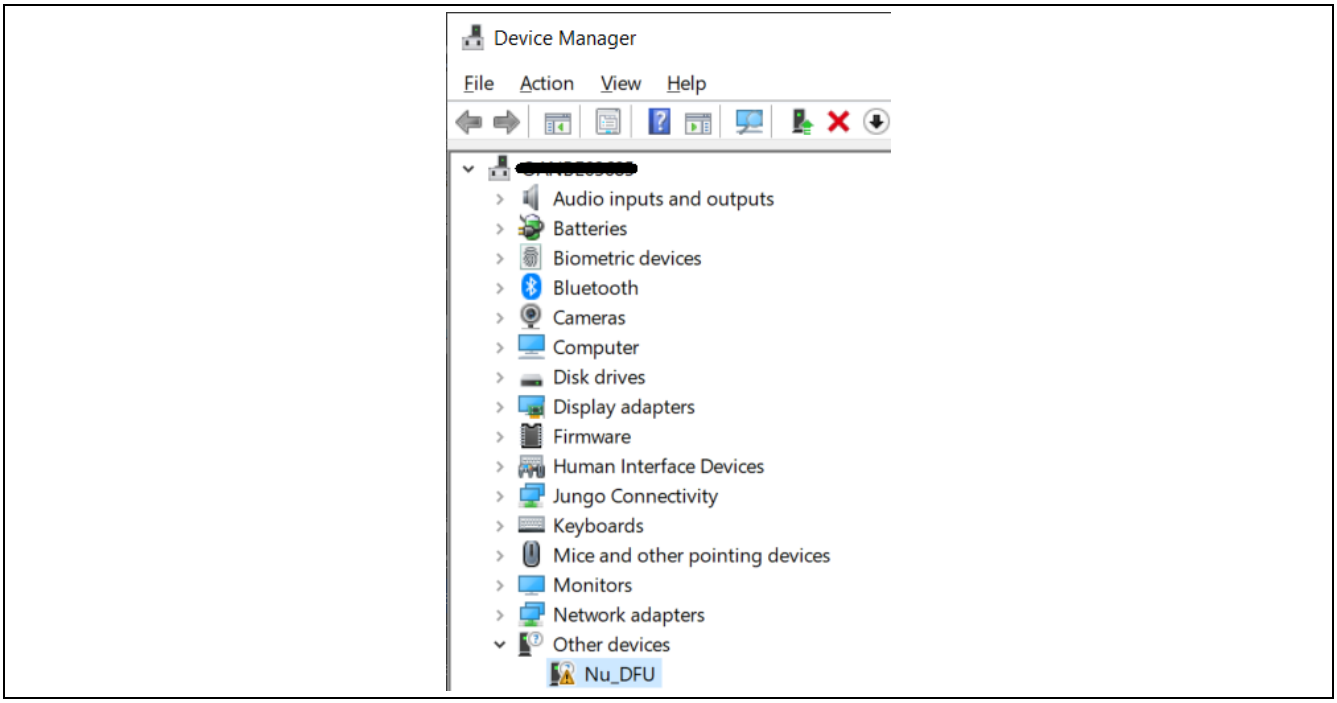
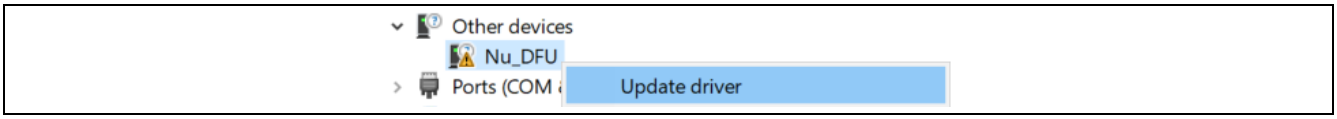


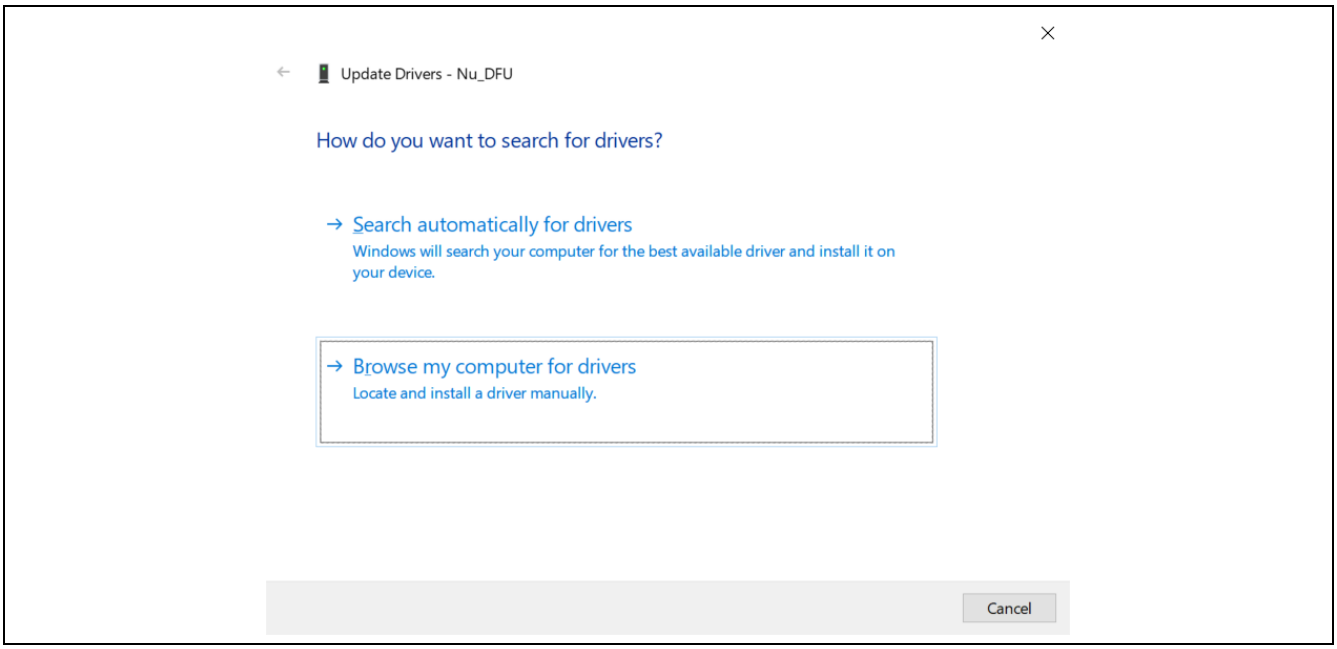
图 1-3 设备管理器中的未知装置

1.2.1 安装 Nuvoton DFU 驱动程序

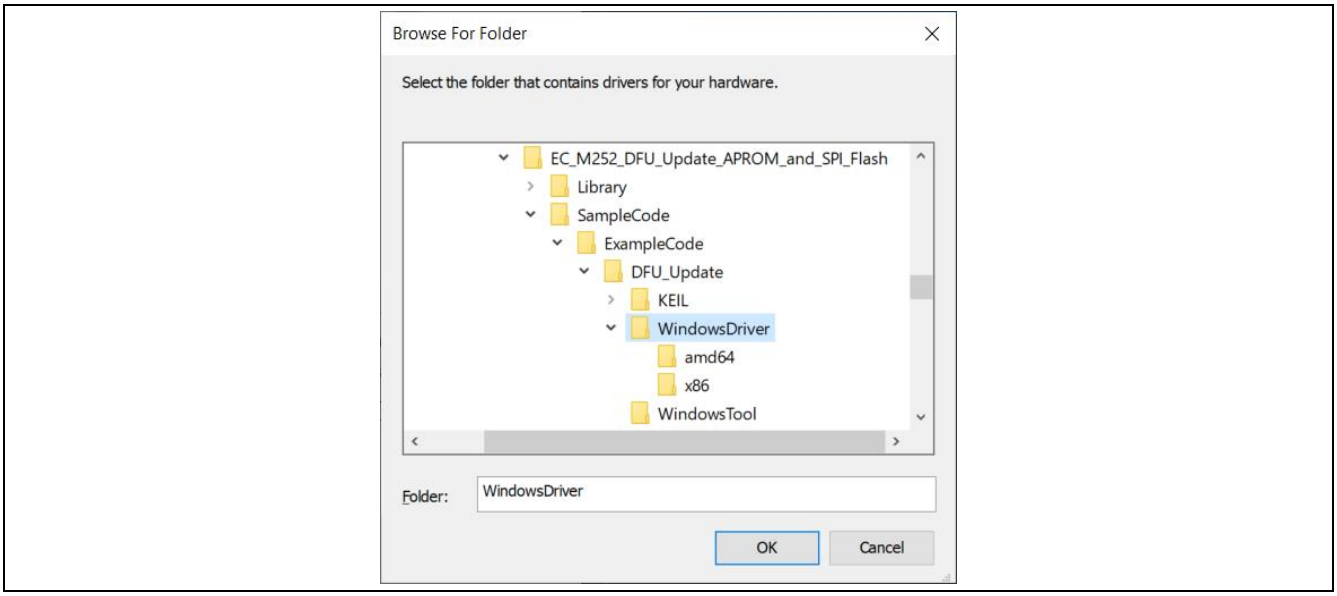
1. 在 Nu_DFU 装置右键单击选择「**Update driver**」。



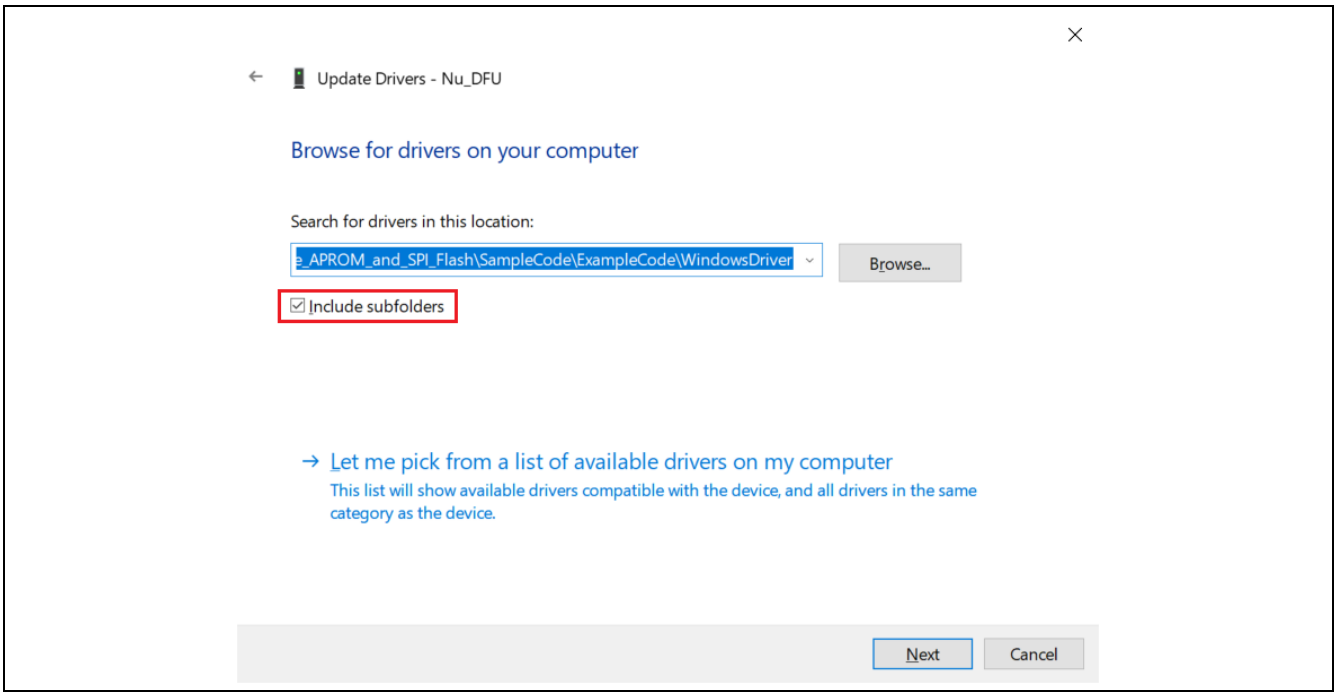
2. 选择「Browse my computer for drivers」。



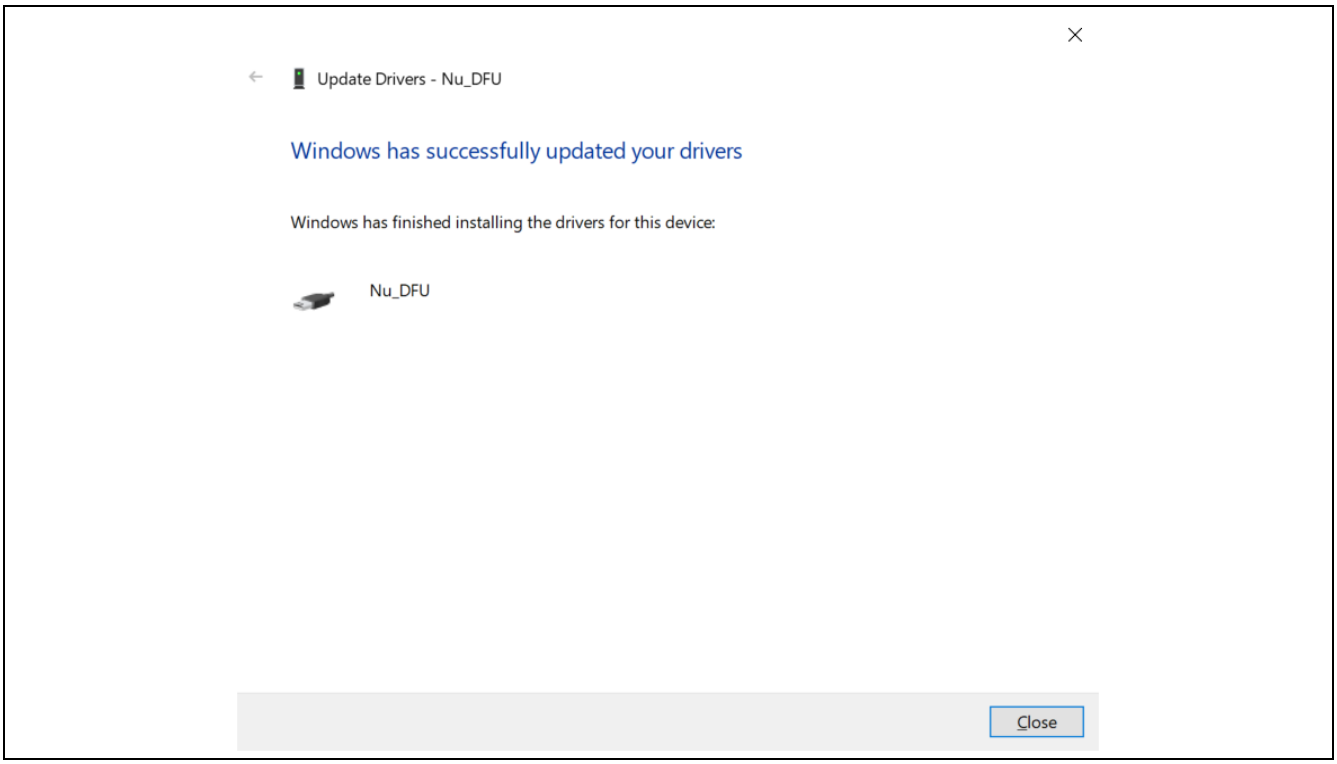
3. 查看目录信息章节选择 **WindowsDriver** 目录，点击「**OK**」。



4. 勾选「Include subfolders」，点击「**Next**」。



5. 确认安装完成后点击「**Close**」结束驱动程序安装。



6. 从设备管理器确认 **Nu_DFU** 已成功安装。

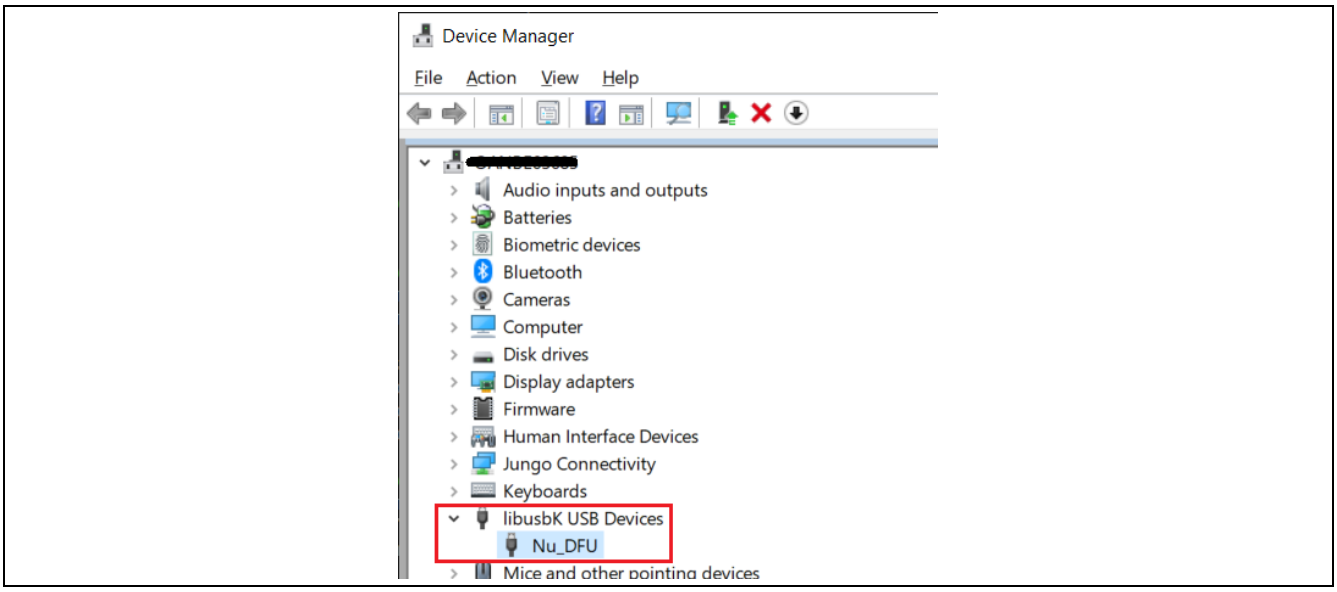
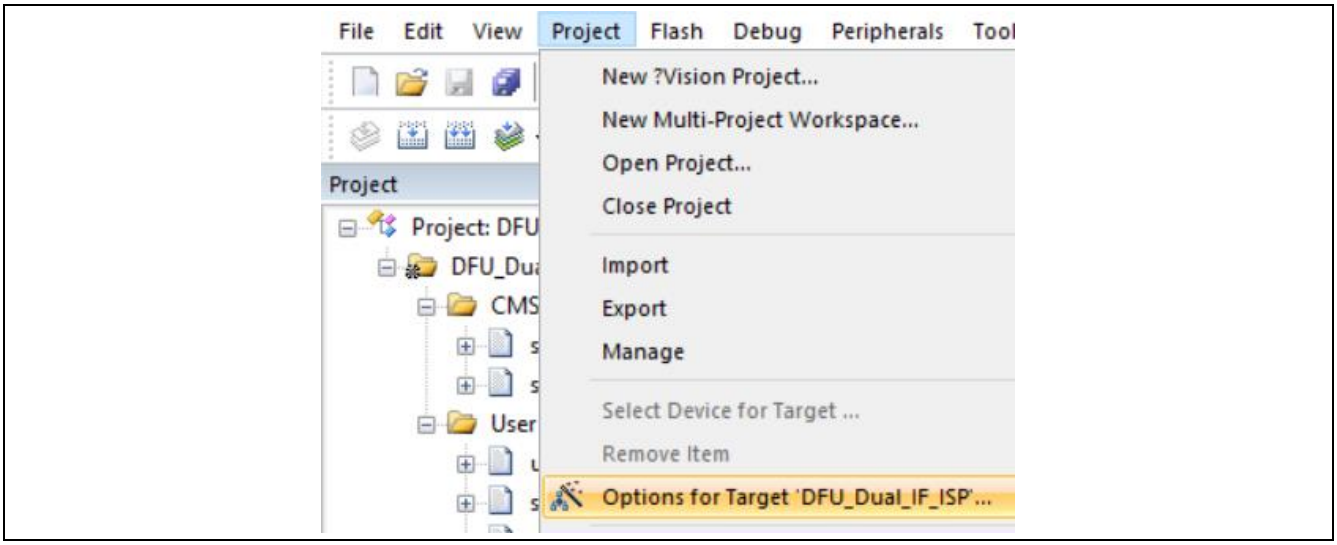


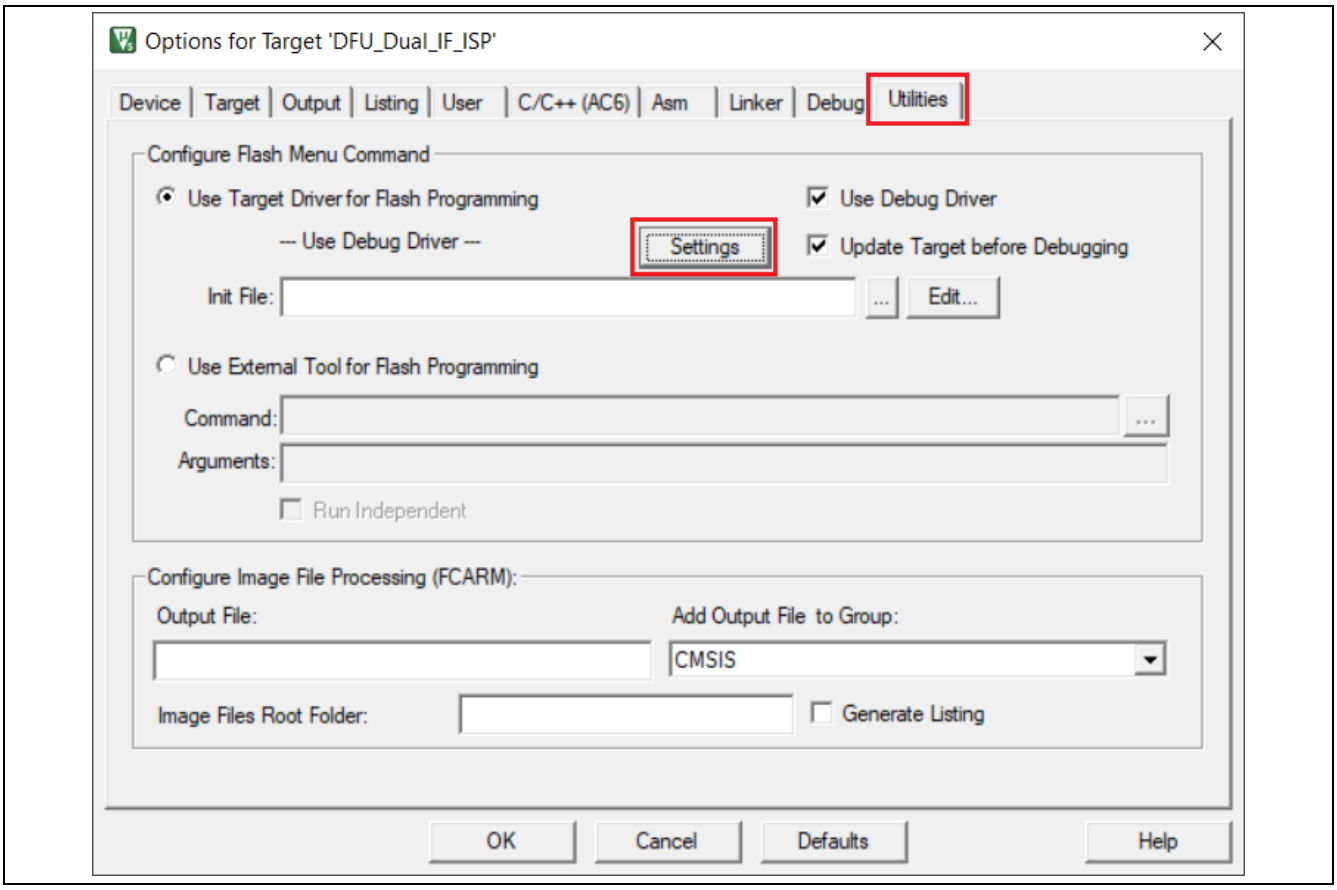
图 1-4 设备管理器中的 DFU 装置

1.2.2 设定由 LDROM 开机

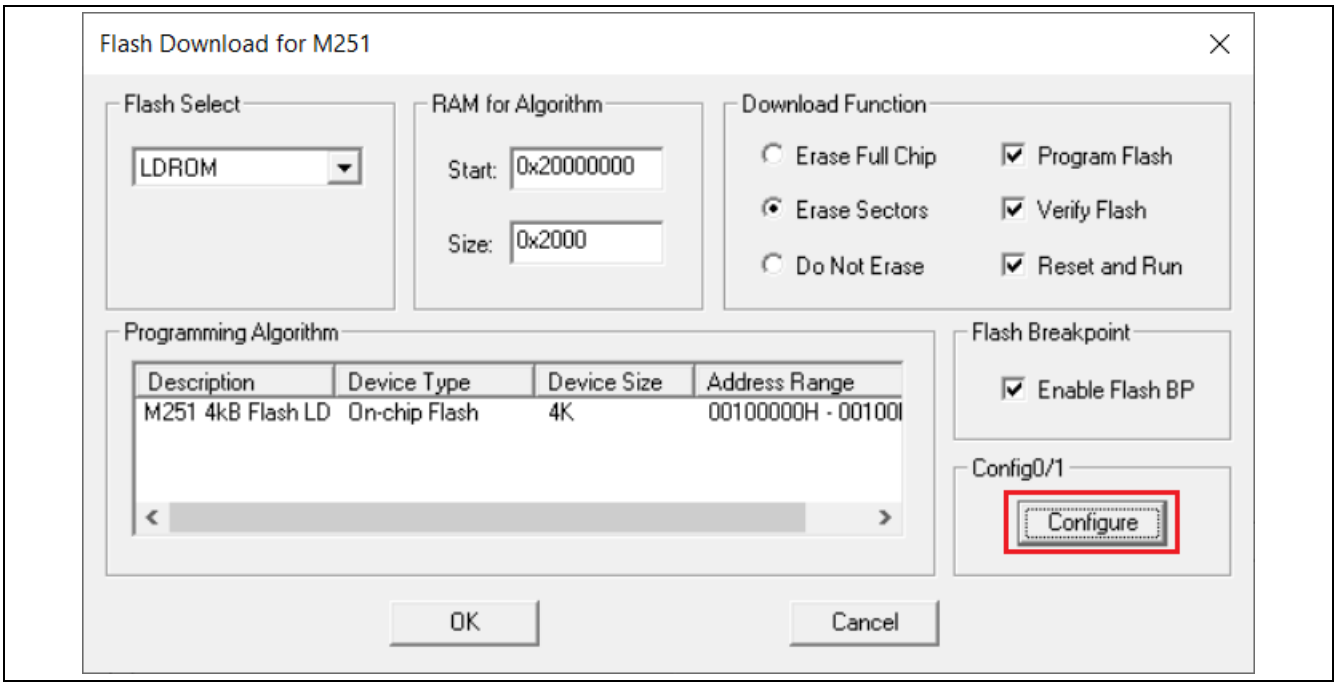
1. 前往「**Project**」选单，选择「**Options for Target**」。



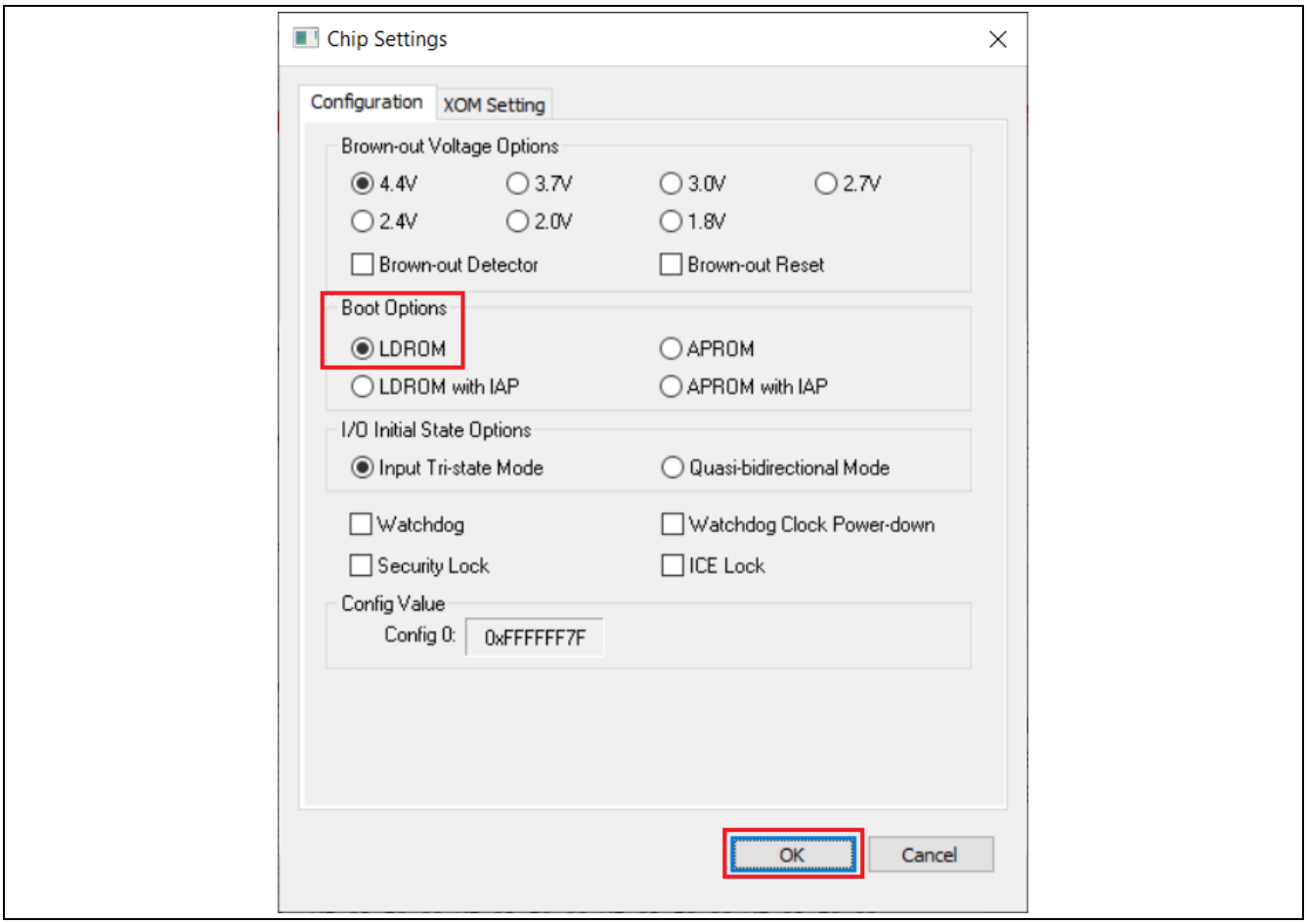
2. 移至「**Utilities**」分页，点选「**Settings**」按钮。



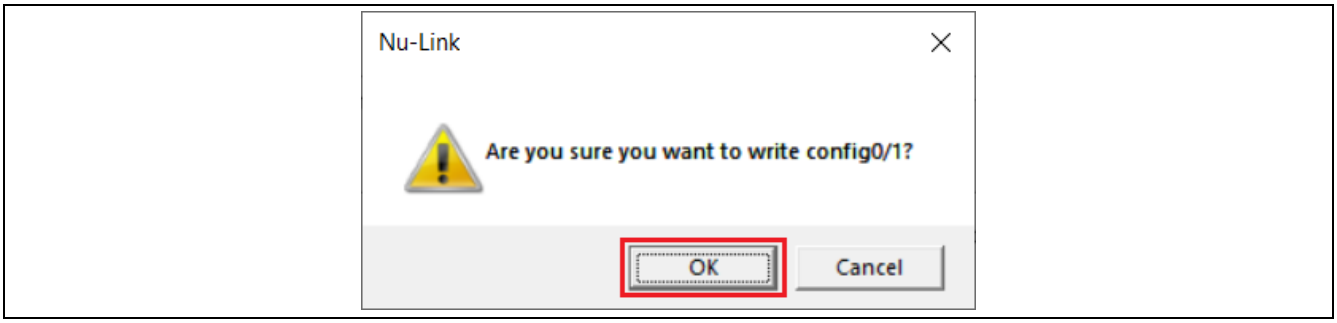
3. 点选「**Configure**」按钮以开启「**Chip Settings**」对话框。



4. 在「Boot Options」中选择「**LDROM**」，并点选「**OK**」。



5. 点选「**OK**」以写入设定。



6. 建置并将程序下载至 LDROM。

1.2.3 DFU 工具

dfu-util 是 USB DFU 通讯协议的开源主机端工具。它可以列出支持 DFU 的装置、选择目标（替代设定），并将固件下载 / 上传到嵌入式装置的各种储存区域。

请查看目录信息章节，在 WindowsTool 文件夹下取得 *dfu-util.exe*。

1.2.3.1 列出 DFU 装置与目标

使用「dfu-util.exe -l」列出所有可用装置。在下载或上传固件前，请先用此命令确认 DFU 装置已就绪。

```
.\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"
```

装置透过替代设定（alt=0 与 alt=1）提供两个 DFU 目标接口。

	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 下载和上传

使用「dfu-util.exe -D」将二进制数据下载到指定的储存区。

以下指令使用「-a 0」选择 APROM，加上「-D .\aprom.bin」将 *aprom.bin* 透过 DFU 下载到 APROM。测试固件 *aprom.bin* 位于 WindowsTool 文件夹中，并可依需要替换为您自己的固件。

「-d 0x0416:0xBDF0」用于指定目标 DFU 装置的厂商 / 产品 ID，若计算机仅连接一个 DFU 装置，则可省略 -d 选项。

```
.\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」从指定的储存区上传二进制数据。

以下指令使用「-a 0」选择 APROM，加上「-U .\read_aprom.bin」透过 DFU 从装置上传完整的 APROM 内容储存至 *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 下载和上传

以下指令使用「-a 1」选择 SPI Flash，加上「-D .\spi.bin」透过 DFU 将 *spi.bin* 下载到 SPI Flash。测试固件 *spi.bin* 位于 WindowsTool 文件夹中，并可依需要替换为您自己的固件。

```
.\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」选择 SPI Flash，加上「-U .\read_spi.bin」透过 DFU 从装置上传完整的 SPI Flash 内容储存至 *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 部分下载与上传

使用「dfu-util.exe -s 偏移:大小」以字节为单位指定起始偏移与数据大小。

以下指令将 *spi_1024KB.bin* 下载到 SPI Flash 的位移 0x10000；若省略大小参数，则会下载整个二进制文件。

```
.\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
```

以下指令从 SPI Flash 上传部分数据区块；自位移 0x10000 起上传 0x100000 字节，并将回传的数据储存至 *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 下载程序到 APROM 并执行

要跳至 APROM 执行程序代码，请依照第 1.2.2 节设定由 LDROM 开机。

本范例在 WindowsTool 文件夹中包含测试固件 *Dump_SPI_Flash.bin*，用于印出 SPI Flash 中的资料。Dump_SPI_Flash 项目可在 ExampleCode 文件夹中找到。

请使用以下指令将 *Dump_SPI_Flash.bin* 下载到 APROM。

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

下载完成后，将 **SW1** 接至 GND 以退出 DFU 模式。系统即会跳转去执行 APROM 上的程序，并透过 UART 打印出 SPI Flash 中的原始数据。

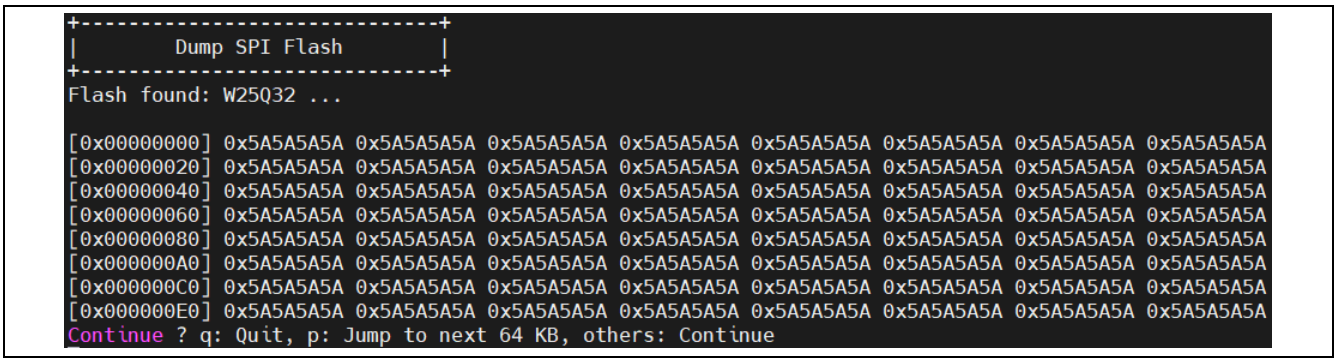


图 1-5 SPI Flash 数据内容

2. 代码介绍

在 *main.c* 的 SYS_Init() 中配置 USBD 和 SPI0 模块的时钟以及多功能引脚设定。

```
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);
```

此范例使用 PCLK1 作为 SPI 的时钟来源，并以 PA0 作为 MOSI、PA1 作为 MISO、PA2 作为 CLK、PA3 作为 /SS 连接 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 作为侦测脚位，以控制系统停留在 DFU ISP 模式或跳转至 APROM 开机。

```
#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)
    {
        ...
```

```
    }
}
```

请查阅 MCU 与 SPI Flash 的资料手册，并在 *descriptors.c* 中的 g_au8Alt0StringDesc 与 g_au8Alt1StringDesc 描述中指定 APROM 与 SPI Flash 的大小与擦除单位。

注意：USB 字符串描述的字符串数据使用 UTF-16LE（小端序）编码，对于标准 ASCII 字符，其高字节通常为 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
};
```

请依照 SPI Flash 的资料手册，在 *cmd_handle.h* 中定义 SPI 相关信息。

```
#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 的下载与上传请求由 *cmd_handle.c* 中的 WriteData 和 ReadData 函数处理。USBD_GetAltSetting() 用于判定数据传输的目标储存区。

```
// 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() 负责处理 SPI Flash 的读取、写入与擦除操作。

```
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;
```

调用 SPI_Open() 时会指定 SPI 通讯参数与时钟频率。

```
    /* 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);
```

检查 SPI Flash 的装置 ID 是否符合预期；若不符合，则回报错误。

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

根据 u32Cmd 处理 SPI Flash 操作：

- 从起始地址到结束地址按页读取数据。
- 按页写入数据，并自动处理扇区擦除。

```
    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 定义支持基本的 SPI Flash 读取 / 写入 / 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() 处理 DFU 状态转换的最低需求。

当主机发出数据传输请求时会调用 WriteData() 和 ReadData()，若传输失败则回报错误。

```
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. 软件与硬件需求

3.1 软件需求

- 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 驱动程序

3.2 硬件需求

- 电路组件
  - NuMaker-M252KG V1.4
  - Winbond W25Q32 SPI Flash (Or compatible SPI Flash)
- 线路示意图
  - 将外部 SPI Flash 依图 3-1 所示连接至 NuMaker-M252KG 的 SPI0。
  - 将 SW1（PB.4）接至 VCC 以进入 DFU ISP 模式，或接至 VSS 以从 APROM 开机。
  - 将 NuMaker-M252KG USB device 接口连接至 PC USB host 接口，以建立 DFU 联机。
  - 将 UART0 TX（PB.13）脚位连接到 PC 的 UART RX 以显示 SPI Flash 的数据导出结果，并将 UART0 RX（PB.12）脚位连接到 PC 的 UART TX 以选择菜单项目。

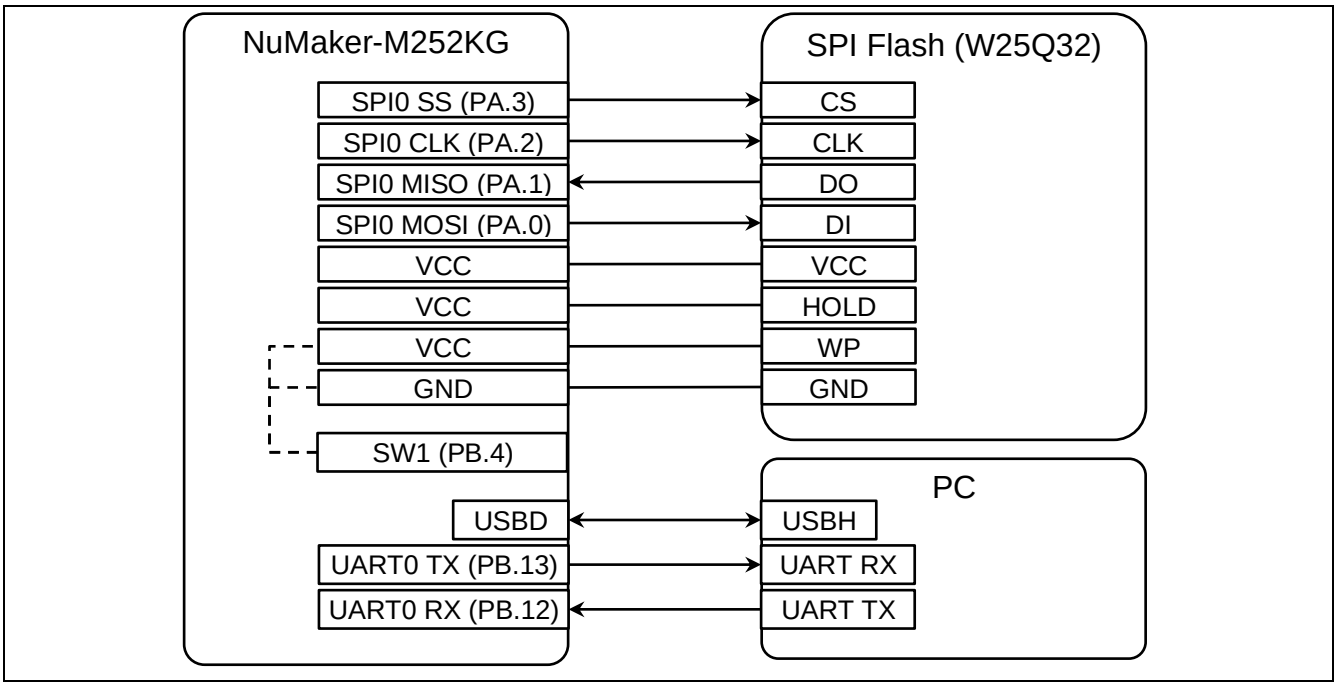

NuMaker-M252KG
SPI0 SS (PA.3)
SPI0 CLK (PA.2)
SPI0 MISO (PA.1)
SPI0 MOSI (PA.0)
VCC
VCC
VCC
GND
SW1 (PB.4)
USBD
UART0 TX (PB.13)
UART0 RX (PB.12)
SPI Flash (W25Q32)
CS
CLK
DO
DI
VCC
HOLD
WP
GND
PC
USBH
UART RX
UART TX

图 3-1 线路示意图

4. 目录信息

目录结构如下图所示。

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

图 4-1 目录信息

5. 范例程序执行

1. 根据目录信息章节进入 DFU_Dual_IF_ISP 路径中的 KEIL 文件夹，双击 *DFU_Dual_IF_ISP.uvprojx*。
2. 进入编译模式接口
   - 编译
   - 下载代码至内存
   - 进入 / 离开除错模式
3. 进入除错模式接口
   - 执行代码

6. 修订纪录

Date	Revision	Description
2025.10.15	1.00	初始发布。

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.