AN_1024

Download Firmware Through Mass Storage Interface

Abstract	This document explains the sample code, "Smpl_MassStorage_ISP" in BSP. The MassStorageISP supports to download firmware to APROM through mass storage interface. By the way, there is no any tool or driver installation required before downloading firmware. It is assumed that the reader is familiar with the Universal Serial Bus (USB) Specification 1.1 and the Device Class Definition for Mass Storage.
Apply to	NUC100 Series

Application Note

Table of Contents-

1 INTRODUCTION

This document explains the sample code, “Smpl_MassStorage_ISP” in BSP and demonstrates how to upgrade user’s application code with Mass Storage ISP.

The source code package of Mass Storage ISP could be downloaded from Nuvoton’s web site: http://www.nuvton.com.tw/. After downloading the source code, please unzip the sample code to:

\NUC100SeriesBSP\NuvotonPlatform_Keil\Sample\USB\Smpl_MassStorage_ISP

Please note you may get build error when placing the code into wrong path.

1.1 Scope

Mass Storage ISP is one In-System Programming methodology to support download firmware to APROM of NuMicro™ Family through mass-storage interface. By using mass-storage interface, we don’t need to provide any tool or driver for PC to support firmware download. For a device which supports Mass Storage ISP, user just needs to plug-in the device by USB cable. Then PC will recognized it as a mass-storage disk and user just need to “COPY” an binary firmware image to the mass-storage disk to do “firmware download”.

1.2 Features

- Use GPB15 to select to execute APROM or to perform mass-storage ISP.
- Supports to download firmware through mass-storage disk.

1.3 Limitation

- In order to make NuMicro™ MCU look like a disk on PC, we need to provide a FAT (File Allocation Table) for different APROM size. FAT file system type is FAT12 in Mass Storage ISP sample code.
- The size of upgraded binary file should not large than actual APROM size.
- The Mass Storage ISP only supports firmware download, so user can not read out the data in APROM.

2 CODE SECTION

In this section, we will show the main functions of Mass Storage ISP sample code and give the relative sample code.

The flow charts are also provided to show how Mass Storage ISP works.

2.1 Main function (Smpl_MassStorage_ISP.c)

In the main function, The Mass Storage ISP program will decide to enter Application Upgrade mode or pass the control privilege to APROM by checking the status of the specified GPIO pin. Here, we use GPIO pin, *GPB15*, as an example.

If Application Upgrade mode is entered, the PLL clock should be set to 48MHz to enable USB interface.

After the hardware and USB initialization, it will check and set USB state variable, g_u8UsbState, according to the USB connection status.

As following sample code, udcFlashInit() will be called to check APROM size and update FAT table. The last step is a while loop to check various events about USB.

```
int32_t main(void)
{
    volatile uint32_t u32INTSTS;
    UNLOCKREG();
    /* Check if GPB15 is low */
    if ( (inp32(GPIOB_BASE + 0x10) & BIT15) != 0)
    {
        /* Boot from AP */
        FMC->ISPCON.BS = 0;
        outp32(&SYS->IPRSTC1, 0x02);
        while(1);
    }

    /* Enable 12M Crystal */
    SYSCLK->PWRCON.XTL12M_EN = 1;
    RoughDelay(0x2000);
    /* Enable PLL */
    outp32(&SYSCLK->PLLCON, 0xC22E);
    RoughDelay(0x2000);

    /* Switch HCLK source to PLL */
    SYSCLK->CLKSEL0.HCLK_S = 2;
```

```
/* Initialize USB Device function */

/* Enable PHY to send bus reset event */
_DRVUSB_ENABLE_USB();
outp32(&USBD->DRVSE0, 0x01);
RoughDelay(1000);
outp32(&USBD->DRVSE0, 0x00);
RoughDelay(1000);
/* Disable PHY */
_DRVUSB_DISABLE_USB();
/* Enable USB device clock */
outp32(&SYSCLK->APBCLK, BIT27);
/* Reset IP */
outp32(&SYS->IPRSTC2, BIT27);
outp32(&SYS->IPRSTC2, 0x0);
_DRVUSB_ENABLE_USB();
outp32(&USBD->DRVSE0, 0x01);
RoughDelay(1000);
outp32(&USBD->DRVSE0, 0x00);
g_u8UsbState = USB_STATE_DETACHED;
_DRVUSB_TRIG_EP(1, 0x08);
UsbFdt();

/* Initialize mass storage device */
udcFlashInit();

/* Start USB Mass Storage */
/* Handler the USB ISR by polling */
  while(1)
  {
    u32INTSTS = _DRVUSB_GET_EVENT_FLAG();
    if (u32INTSTS & INTSTS_FLDET)
      {
          /* Handle the USB attached/detached event */
             UsbFdt();
      }
```

```
            else if(u32INTSTS & INTSTS_BUS)
            {
             /* Handle the USB bus event: Reset, Suspend, and Resume */
                   UsbBus();
            }
            else if(u32INTSTS & INTSTS_USB)
            {
                /* Handle the USB Protocol/Clase event */
                   UsbUsb(u32INTSTS);
          }
        }
}
```

2.2 USB function (MassStorage_ISP.c)

Except udcFlashInit() function, other functions are similar to Mass Storage Device implementation described in application note, *AN1020EN MassStorage*.

2.3 FAT Handler and Flash Program function (FlashProg.c)

The FMC_ReadPage() is used to response the FAT table to PC and the DataFlashWrite() is used to program the new firmware into APROM.

.

```
……
void FMC_ReadPage(uint32_t u32startAddr, uint32_t * u32buff)
{
   uint32_t i;

   for (i = 0; i < FLASH_PAGE_SIZE/4; i++)
   {
      u32buff[i] = 0;
   }

   if (u32startAddr == 0x00000000)
   {
      my_memcpy((uint8_t *)u32buff, u8FormatData, 62);

      u32buff[FLASH_PAGE_SIZE/4-1] = 0xAA550000;
```

```
    }
    else
    {
        if ( (u32startAddr == (FAT_SECTORS * 512)) || (u32startAddr ==
((FAT_SECTORS+1) * 512)) )
        {
            u32buff[0] = 0x00FFFFF8;
        }
    }
}
……
void DataFlashWrite(uint32_t addr, uint32_t buffer)
{
    /* This is low level write function of USB Mass Storage */

    if ( (addr >= DATA_SECTOR_ADDRESS) && (addr <
(DATA_SECTOR_ADDRESS + g_u32FlashSize)) )
    {
        addr -= DATA_SECTOR_ADDRESS;
        if (g_u8SecurityLockBit)
        {
            myFMC_Erase(addr);
            FMC_ProgramPage(addr, (uint32_t *) buffer);
        }
        /* For Security Lock */
        else
        {
            if (addr == 0)
            {
                uint32_t i;
                for (i = 0; i < g_u32FlashSize; i += FLASH_PAGE_SIZE)
                {
                    myFMC_Erase(i);
                }
            }
```

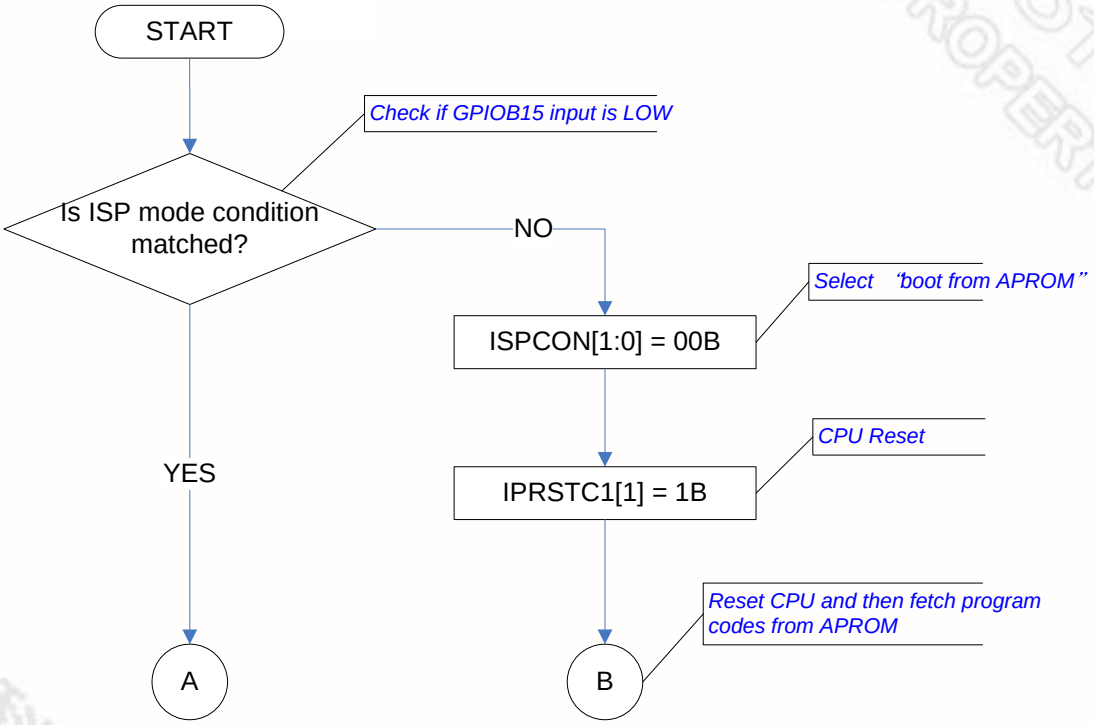
```
        FMC_ProgramPage(addr, (uint32_t *) buffer);
    }
}
```

3 CALLING SEQUENCE

The flow of the Mass Storage ISP program is divided into two stages. One is Mass Storage ISP booting, and the other is main process. Both stages will be described as following:

3.1 Mass Storage ISP Booting Flow

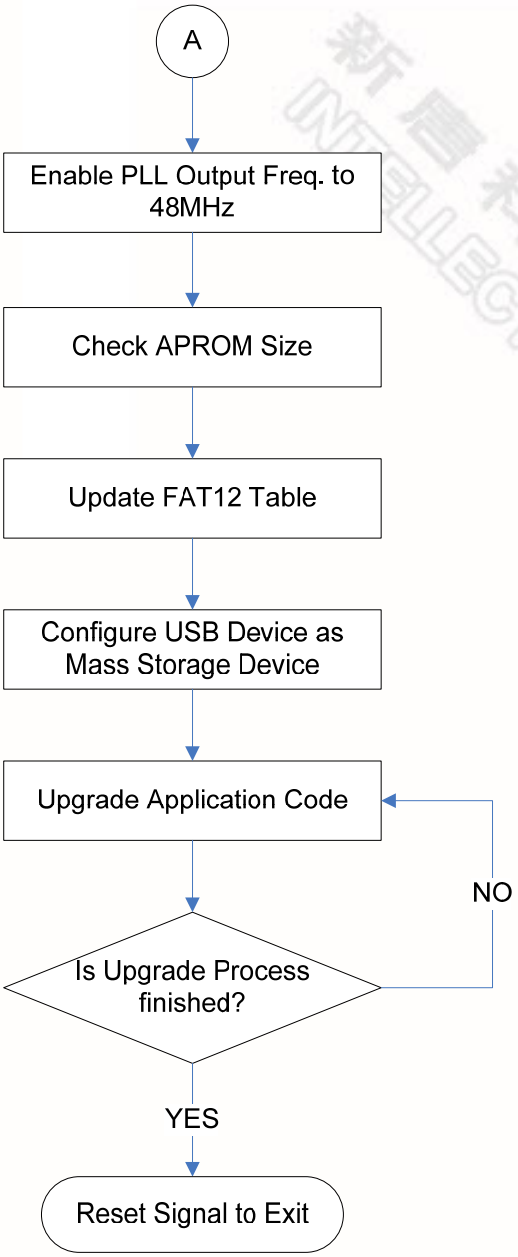
The Mass Storage ISP program will enter application upgrade mode or reset to boot from APROM according to the status of the specified GPIO pin. In the sample code, *GPB15* is used to control Mass Storage ISP to reset to APROM or do mass-storage (application upgrade mode).



3.2 Mass Storage ISP Main Flow

When entering application upgrade mode, the Mass Storage ISP program will check APROM size, update FAT table and configure NuMicro™ MCU as mass storage class device. After all initial steps have been done successfully; it is ready to upgrade new application code transferred from Host.

After upgrade operation has been completed, User needs to set GPB15 as high and reset the NuMicro™ MCU to execute the new firmware in APROM.

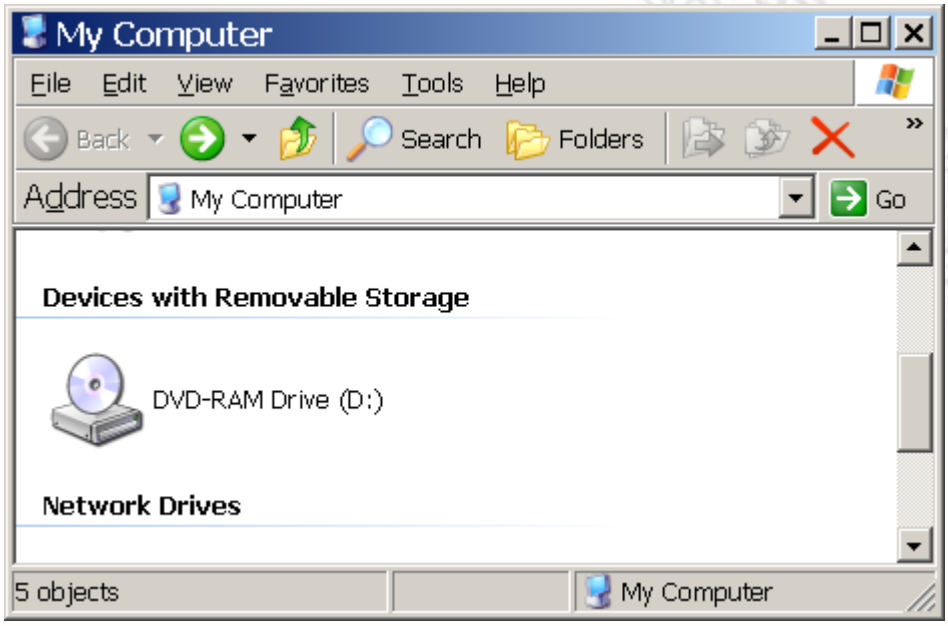


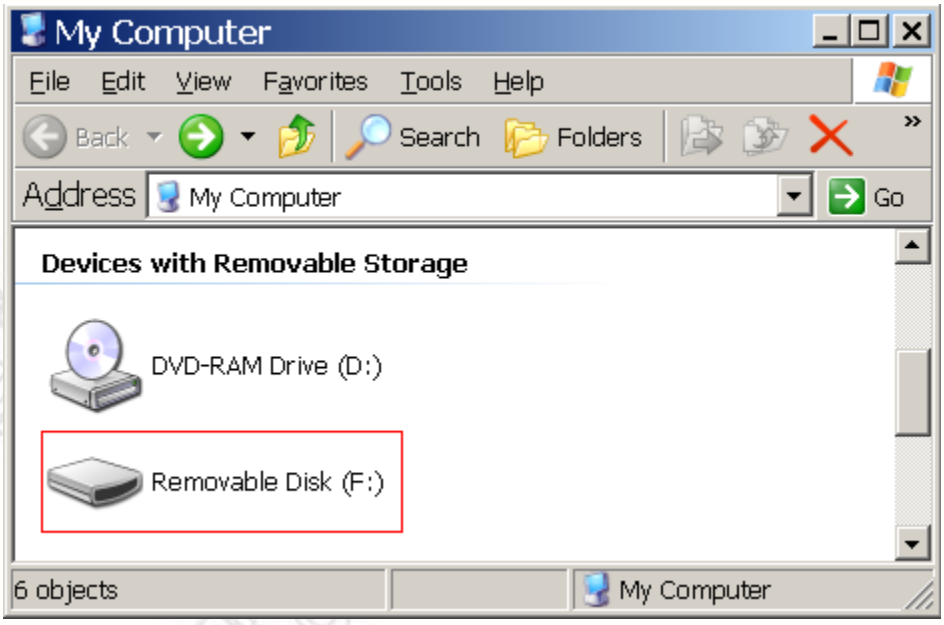
4 EXECUTION ENVIRONMENT SETUP AND RESULT

4.1 Firmware Download by Mass Storage ISP

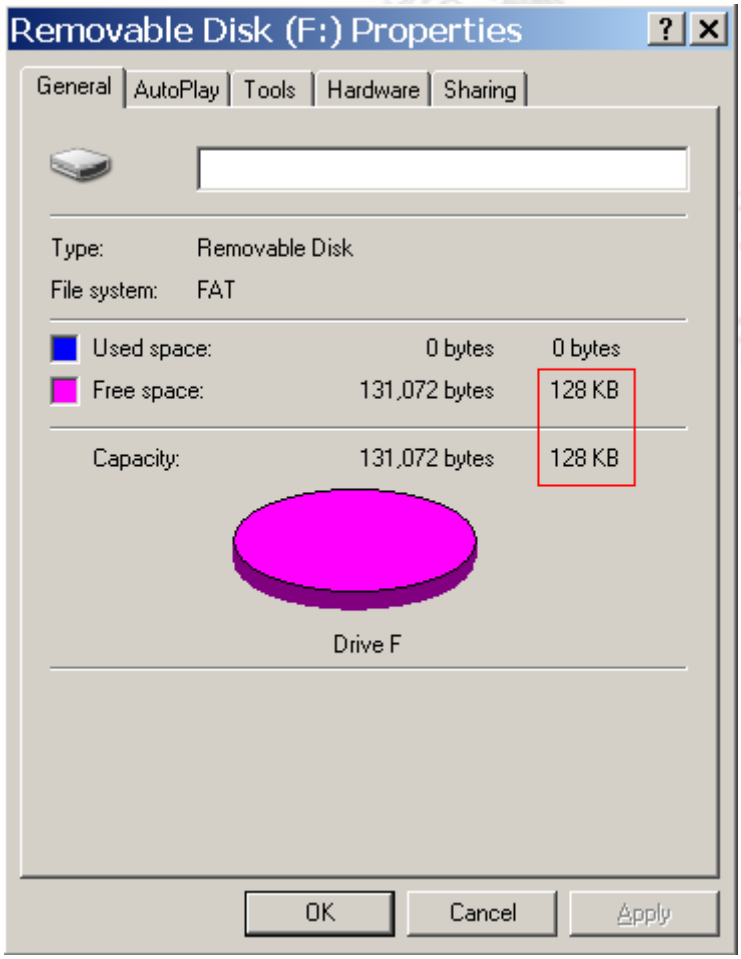
4.1.1 Configure NuMicro™ MCU as mass storage class device

On Power-On, the Mass Storage ISP will enter Application Upgrade mode if the specified GPIO pin is low. Then, PC can detect a mass storage class device when the NuMicro™ MCU is plugged into PC USB port.



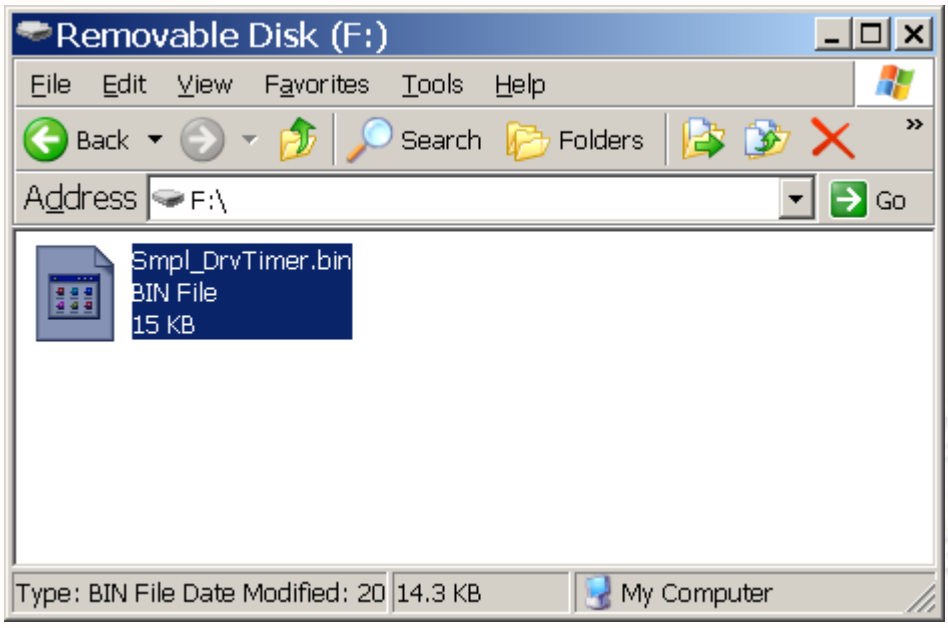


We can also find that the capacity of the mass storage device is equal to APROM size of the NuMicro™ MCU.



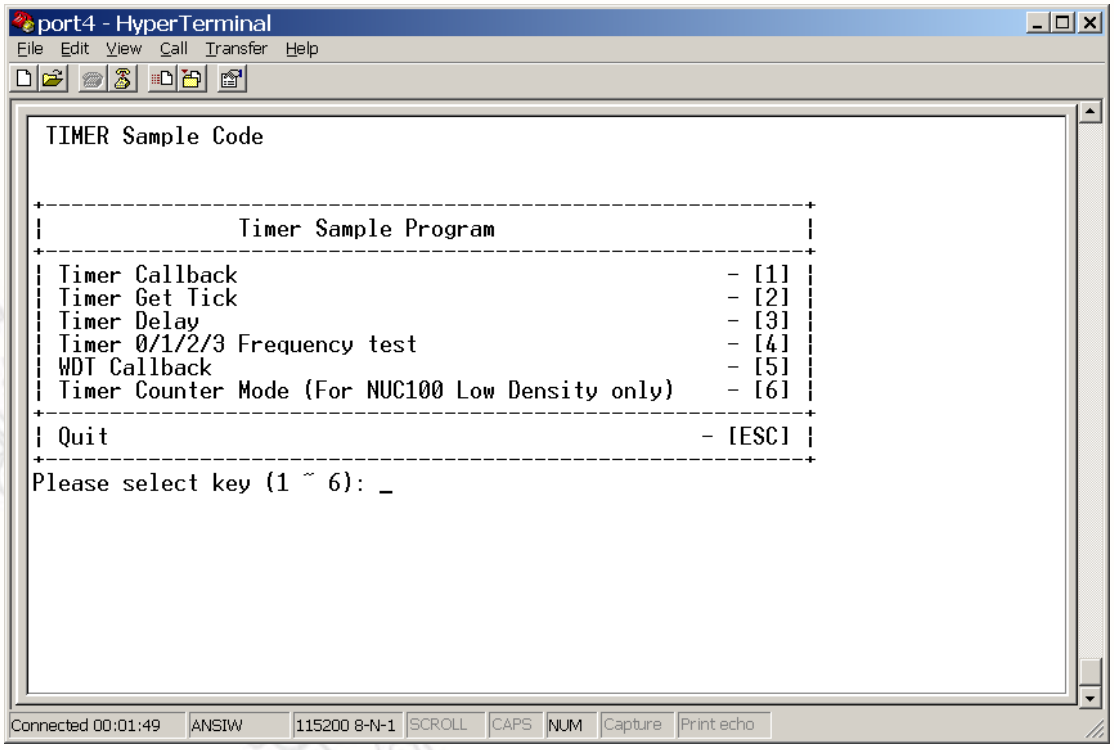
4.1.2 Upgrade Firmware

Then, we drag or copy-and-paste the binary file of upgraded application code into window of the mass storage device. Here, we select the binary file of the sample code of Timer Driver in BSP.



4.1.3 Execution result

After the upgrade process has been completed, the status of the specified GPIO pin is high and press down RESET button or Power-On reset. Then Mass Storage ISP will reset to APROM when it detect the GPB15 is high and execute the firmware in APROM. Here, we can find that the Timer sample code works and sends out message on Hyper Terminal Window.



5 REVISION HISTORY

REV.	DATE	DESCRIPTION
1.00	March 04, 2011	Created.

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.