Switching Code between M480 LDROM and SPI Flash

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This example code is used for the M480 series microcontroller (MCU) to switch code between LDROM and SPI Flash.
BSP Version	M480 BSP CMSIS V3.05.003
Hardware	NuMaker-PFM-M487 Ver3.0

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 Overview

This example code is used for the M480 series microcontroller (MCU) to jump from LDROM to SPI Flash, and then jump from SPI Flash to LDROM. Bootloader is a project in LDROM used to jump to the application in SPI Flash. It writes to the VTOR to relocate the vector table start address to SPI Flash's start address 0x8000000, and then the program pointer points to 0x8000000 and the stack pointer points to 0x8000004 to jump to the application. APP is a project in SPI Flash used to jump to the Bootloader in LDROM. It sets BS bit to 1 to boot from LDROM, and then sets SYSENSETREQ bit to 1 to assert the MCU Reset to jump to the Bootloader. This document introduces the example code of SPI Flash initialization, modifying VTOR to jump from LDROM to SPI Flash, and modifying BS to jump from SPI Flash to LDROM.

1.1 Demo Result

If debug UART is connected to PC when executing the program, the UART tool can display information as shown below.

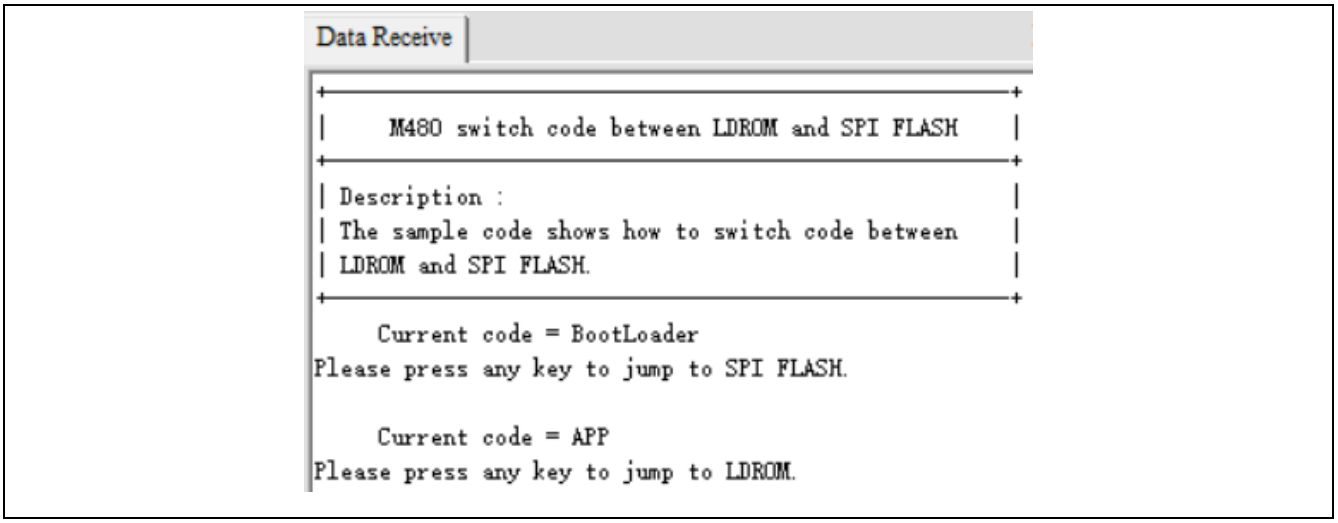

Figure 1-1 Demo Result

2 Code Description

Define SPI Flash address:

```
#define SPIFLASH_ADDR 0x8000000   /* SPI FLASH */
```

Define SPI Flash opcodes:

```
#define OPCODE_WREN             0x06U   /* Write enable */
#define OPCODE_RDSR             0x05U   /* Read status register #1*/
#define OPCODE_WRSR             0x01U   /* Write status register #1 */
#define OPCODE_READ_ID          0x90U   /* Read ID */
#define OPCODE_RDID             0x9fU   /* Read JEDEC ID */

/* Used for SST flashes only. */
#define OPCODE_WRDI             0x04U   /* Write disable */

#define OPCODE_RSTEN            0x66U
#define OPCODE_RST              0x99U
```

SPI Flash initialization:

```
void SpiFlash_Init(void)
{
    /* Set SPIM clock as HCLK divided by 2 */
    SPIM->CTL1 = (SPIM->CTL1 & (~SPIM_CTL1_DIVIDER_Msk)) |
                 ((1) << SPIM_CTL1_DIVIDER_Pos);
    /* Insert 0 delay cycle. Adjust the sampling clock of received data to latch the
correct data. */
    SPIM->RXCLKDLY = (SPIM->RXCLKDLY & (~SPIM_RXCLKDLY_RDDLYSEL_Msk)) |
                     ((0) << SPIM_RXCLKDLY_RDDLYSEL_Pos);
    /* Use SPI input clock rising edge to sample received data. */
    SPIM->RXCLKDLY |= SPIM_RXCLKDLY_RDEDGE_Msk;
    /* 8 is the default value. */
    SPIM->CTL2 = (SPIM->CTL2 & ~SPIM_CTL2_DCNUM_Msk) |
                 (((8) & 0x1FUL) << SPIM_CTL2_DCNUM_Pos) | SPIM_CTL2_USETEN_Msk;
    if (SPIM_InitFlash(1) != 0)        /* Initialized SPI flash */
    {
        while (1);
    }
    SPIM->CTL1 &= ~SPIM_CTL1_CCMEN_Msk; /* Disable CCM */
    SPIM->CTL1 &= ~SPIM_CTL1_CACHEOFF_Msk; /* Enable cache */
    SPIM->CTL0 |= SPIM_CTL0_CIPHOFF_Msk; /* Disable cipher */

    SPIM->CTL1 |= SPIM_CTL1_CDINVAL_Msk; /* invalid cache */
```

```
    SPIM_EnterDirectMapMode(USE_4_BYTES_MODE, CMD_DMA_FAST_READ, 0);  /* Enter Direct Map mode */
}
```

Jump from LDROM to SPI Flash:

```
int main(void)
{
    FUNC_PTR    *func; /* function pointer */
    /* Unlock protected registers */
    SYS_UnlockReg();
    /* Init System, peripheral clock and multi-function I/O */
    SYS_Init();
    /* Init UART to 115200-8n1 */
    UART_Init();
    CLK->AHBCLK |= CLK_AHBCLK_ISPCKEN_Msk;
    FMC->ISPCTL |= FMC_ISPCTL_ISPEN_Msk;
    /* init spi flash */
    SpiFlash_Init();
    /*---------------------------------------------------------------------------
     This sample code shows how to switch code between LDROM and SPI FLASH.
     In bootloader, VTOR is used to implement the jump from LDROM to SPI FLASH.
     In APP, BS is used to implement the jump from SPI FLASH to LDROM.
     To use this sample code, please:
     1. Build all targets and download to device individually. The targets are:
       BootLoader, RO=0x10000
       APP,        RO=0x8000000
     2. Set boot from LDROM.
     3. Set SPIM pins in config.
     4. Reset MCU to execute BootLoader.
---------------------------------------------------------------------------*/
     printf("+-----------------------------------------------------+\n");
     printf("|             M480 LDROM jump to SPI FLASH            |\n");
     printf("+-----------------------------------------------------+\n");
     printf("| Description :                                       |\n");
     printf("| The sample code shows how to switch code between    |\n");
     printf("| LDROM and SPI FLASH.                                |\n");
     printf("+-----------------------------------------------------+\n");
     printf("     Current code = BootLoader                \n");
     printf("Please press any key to jump to SPI FLASH. \n\n");
     getchar();

     __disable_irq(); //disable all interrupt
```

```
    SCB->VTOR = SPIFLASH_ADDR; //VTOR
    func = (FUNC_PTR *)*(uint32_t *)(SPIFLASH_ADDR + 4); //PC
    __set_SP(*(uint32_t *)SPIFLASH_ADDR); //SP
    func();
    while(1);
}
```

Jump from SPI Flash to LDROM:

```
int32_t main(void)
{
    /* Unlock protected registers */
    SYS_UnlockReg();
    /* Init System, IP clock and multi-function I/O. */
    SYS_Init();
    /* Configure UART0: 115200, 8-bit word, no parity bit, 1 stop bit. */
    UART_Open(UART0, 115200);
    /* Unlock protected registers */
    SYS_UnlockReg();
    __enable_irq();
    /* Enable FMC ISP function */
    FMC_Open();
    printf("     Current code = APP                \n");
    printf("Please press any key to jump to LDROM. \n\n");
    getchar();
    __disable_irq(); //disable all interrupt
    SCB->VTOR = 0;
    SYS->RSTSTS = (SYS_RSTSTS_PORF_Msk | SYS_RSTSTS_PINRF_Msk);
    FMC->ISPCTL &= ~(FMC_ISPCTL_ISPEN_Msk | FMC_ISPCTL_BS_Msk);
    FMC->ISPCTL |= FMC_ISPCTL_BS_Msk; //BS = LDROM
    SCB->AIRCR = (V6M_AIRCR_VECTKEY_DATA | V6M_AIRCR_SYSRESETREQ); //systerm reset
    while(1);
}
```

3 Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M480 Series BSP CMSIS v3.05.003
- IDE version
  - Keil uVersion 5.26

3.2 Hardware Requirements

- Circuit components
  - NuMaker-PFM-M487 Ver3.0
  - PC

- Pin Connect

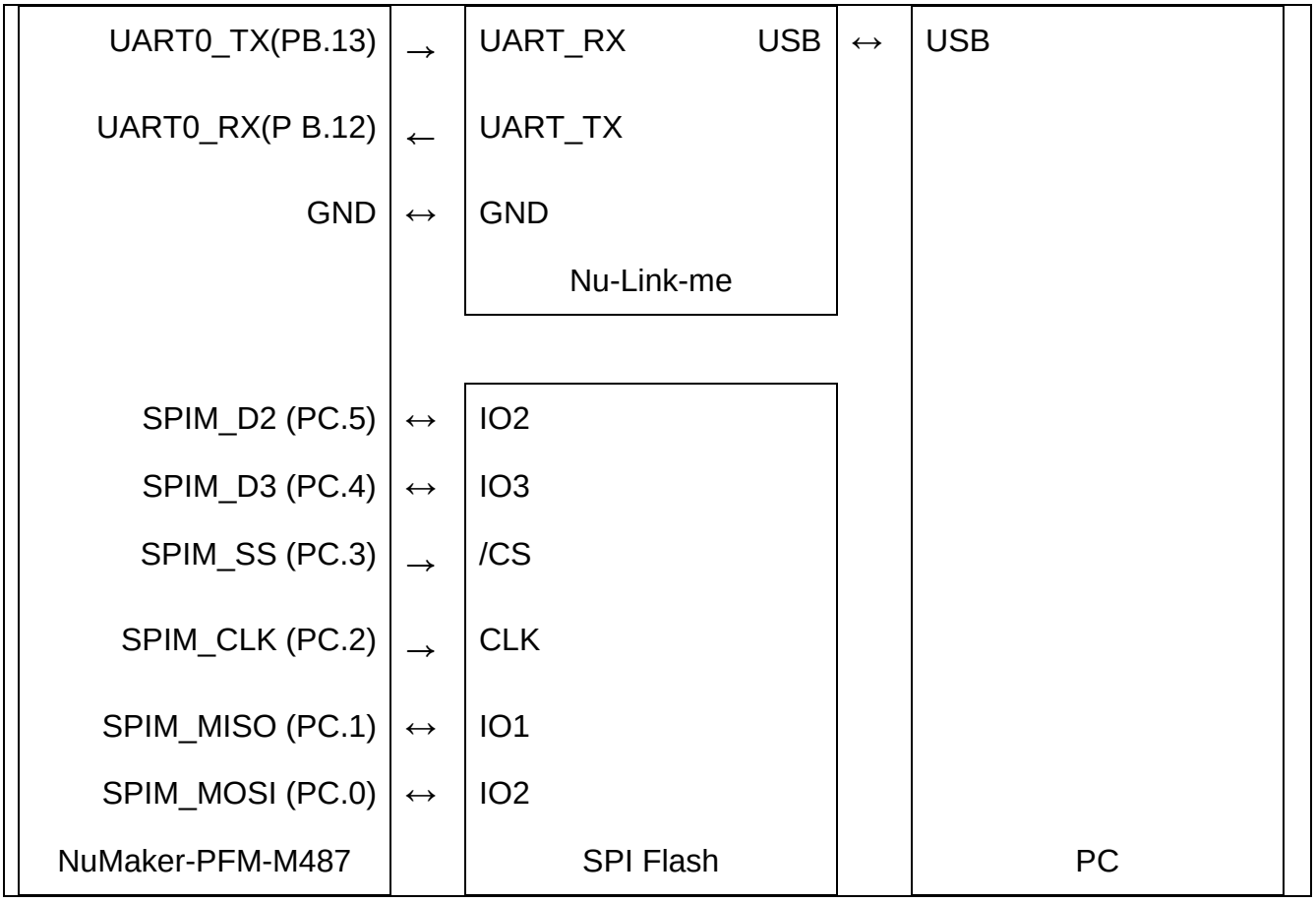

Figure 3-1 Pin Connect

4 Directory Information

The directory structure is shown below.

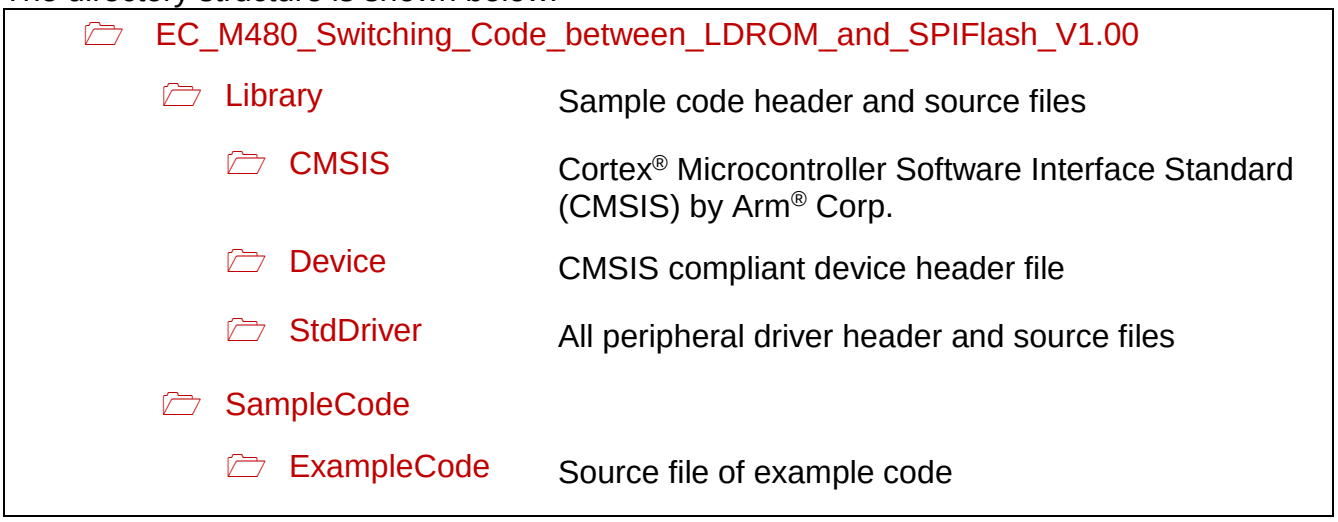

Figure 4-1 Directory Structure

5 Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *M480_Switching_Code_between_DROM_and_SPIFlash.uvmpw*.
2. Modify config in Keil.
   - Options for Target -> Utilities -> Settings -> Configure
   - Boot Select = LDROM
   - Boot Loader SPIM CLK/SS/MISO/MOSI Multi-function Pin Select = PC.2/PC.3/PC.1/PC.0
   - Click OK
3. Set bootloader as active project, and enter Keil compile mode.
   - Build
   - Download
4. Set APP as active project, and enter Keil compile mode.
   - Build
   - Download
5. Run.
   - Press reset key on NuMaker-PFM-M487.
6. Print debug information.
   - Open *UartAssist.exe*
   - Baud rate: 115200bps

6 Revision History

Date	Revision	Description
2022.05.06	1.00	1. Initially issued.

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.