Vector Table Remap

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	Use vector table remapping to implement execution switch between two programs
BSP Version	NUC123 Series BSP CMSIS V3.01.001
Hardware	NuTiny-EVB-123-LQFP64_V1.0

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 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 Function Description

1.1 Introduction

The Nuvoton NUC123 series of microcontrollers provides the ability to remap vector table, and developers can download two programs to different locations in APROM. By remapping the vector table and the function pointer, two programs can switch for execution.

1.2 Principle

Since the Cortex-M0 specification vector table must be placed at address 0, if there are two programs that need to be switched in APROM, there will be problems with addresses overwriting each other. At this point, you can properly arrange the addresses of the two programs in the APROM, and then remap the vector table address to avoid this problem caused by address coverage.
This example is divided into APP1 and APP2. The address of APP1 is 0x0, and the address of APP2 is 0x3000, as shown in Figure 1-1. When APP1 is executed, first reset the peripheral IPs, then use the ISP command to remap the address of 0x3000 to the location of 0x0, and set the stack pointer to the value stored in 0x3000. After completing the above steps, you can switch to APP2 execution, that is, use the function point to jump to APP2's Reset function (0x3000+0x4). When APP2 is executed, the procedure for switching to APP1 is the same as above, but it is necessary to restore the settings of mapping 0x3000 to 0x0, and set the stack pointer to the value stored in 0x0.

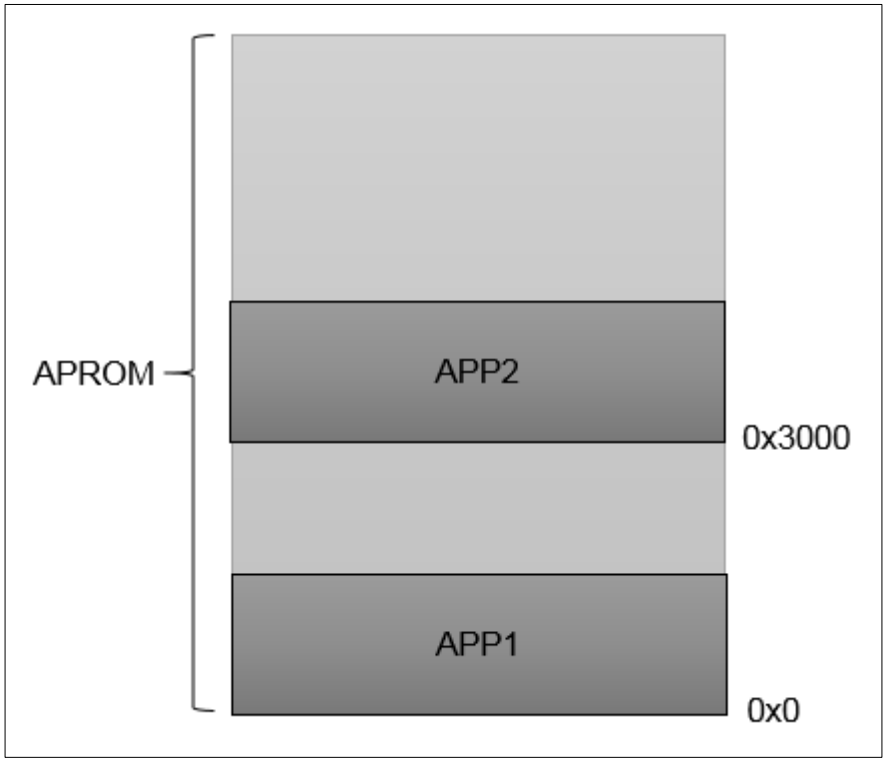

Figure 1-1 Memory map

1.3 Demo Result

APP1 and APP2 print different strings respectively. And toggle PB4 pin to make the LED light flash, APP1 toggle frequency is faster, APP2 is slower, as shown in Figure 1-2. Through the printed string and the frequency at which the LED light flashes, it can be observed that APP1 and APP2 are continuously switched and executed.

```
/*****************************************************************/
/***   NUC123 Change Vector Table Address Sample Code(App1)  ***/
/*****************************************************************/
/*****************************************************************/
/***   NUC123 Change Vector Table Address Sample Code(App2)  ***/
/*****************************************************************/
/*****************************************************************/
/***   NUC123 Change Vector Table Address Sample Code(App1)  ***/
/*****************************************************************/
/*****************************************************************/
/***   NUC123 Change Vector Table Address Sample Code(App2)  ***/
/*****************************************************************/
```

Figure 1-2 Massage on the terminal

2 Code Description

2.1 APP1

Print message and flashing LED

```
    printf("/*****************************************************************/\n");
    printf("/***   NUC123 Change Vector Table Address Sample Code(App1)  ***/\n");
    printf("/*****************************************************************/\n");

    /* Set GPIO to output mode */
    GPIO_SetMode(PB, BIT4, GPIO_PMD_OUTPUT);

    /* Flashing LED */
    while(u8LEDFlashNum--)
    {
        PB4 = 0;
        for(i = 0; i < LED_TIME_LENGTH; i++);
        PB4 = 1;
        for(i = 0; i < LED_TIME_LENGTH; i++);
    }
```

Initialize peripheral IPs and remapping addresses

```
    /* Reset IPs */
    SYS->IPRSTC1 = SYS_IPRSTC1_PDMA_RST_Msk;
    SYS->IPRSTC2 = (SYS_IPRSTC2_I2S_RST_Msk|SYS_IPRSTC2_ADC_RST_Msk
                   |SYS_IPRSTC2_USBD_RST_Msk|SYS_IPRSTC2_PS2_RST_Msk
                   |SYS_IPRSTC2_PWM03_RST_Msk|SYS_IPRSTC2_UART1_RST_Msk
                   |SYS_IPRSTC2_UART0_RST_Msk|SYS_IPRSTC2_SPI2_RST_Msk
                   |SYS_IPRSTC2_SPI1_RST_Msk|SYS_IPRSTC2_SPI0_RST_Msk
                   |SYS_IPRSTC2_I2C1_RST_Msk|SYS_IPRSTC2_I2C0_RST_Msk
                   |SYS_IPRSTC2_TMR3_RST_Msk|SYS_IPRSTC2_TMR2_RST_Msk
                   |SYS_IPRSTC2_TMR1_RST_Msk|SYS_IPRSTC2_TMR0_RST_Msk
                   |SYS_IPRSTC2_GPIO_RST_Msk);
    SYS->IPRSTC1 &= ~SYS_IPRSTC1_PDMA_RST_Msk;
    SYS->IPRSTC2 &= ~(SYS_IPRSTC2_I2S_RST_Msk|SYS_IPRSTC2_ADC_RST_Msk
                   |SYS_IPRSTC2_USBD_RST_Msk|SYS_IPRSTC2_PS2_RST_Msk
                   |SYS_IPRSTC2_PWM03_RST_Msk|SYS_IPRSTC2_UART1_RST_Msk
                   |SYS_IPRSTC2_UART0_RST_Msk|SYS_IPRSTC2_SPI2_RST_Msk
                   |SYS_IPRSTC2_SPI1_RST_Msk|SYS_IPRSTC2_SPI0_RST_Msk
                   |SYS_IPRSTC2_I2C1_RST_Msk|SYS_IPRSTC2_I2C0_RST_Msk
                   |SYS_IPRSTC2_TMR3_RST_Msk|SYS_IPRSTC2_TMR2_RST_Msk
                   |SYS_IPRSTC2_TMR1_RST_Msk|SYS_IPRSTC2_TMR0_RST_Msk
                   |SYS_IPRSTC2_GPIO_RST_Msk);
```

```
    /* Remap vector table address to APP2 address */
    CLK->APBCLK |= CLK_AHBCLK_ISP_EN_Msk;
    FMC->ISPCON |= FMC_ISPCON_ISPEN_Msk;
    FMC->ISPCMD = FMC_ISPCMD_VECMAP;
    FMC->ISPADR = APP2_ADDR;
    FMC->ISPTRG |= FMC_ISPTRG_ISPGO_Msk;
```

Reset stack point and execute APP2

```
    /* Set stack pointer to the value stored in APP2 address */
    pu32SP = (uint32_t *)APP2_ADDR;
    spChange(pu32SP);

    /* Execute another application */
    func = (FUNC_PTR *)(*(uint32_t *)(APP2_ADDR+ 4));
    func();
```

2.2 APP2

Print message and flashing LED

```
    printf("/*****************************************************************/\n");
    printf("/***   NUC123 Change Vector Table Address Sample Code(App2)  ***/\n");
    printf("/*****************************************************************/\n");

    /* Set GPIO to output mode */
    GPIO_SetMode(PB, BIT4, GPIO_PMD_OUTPUT);

    /* Flashing LED */
    while(u8LEDFlashNum--)
    {
        PB4 = 0;
        for(i = 0; i < LED_TIME_LENGTH; i++);
        PB4 = 1;
        for(i = 0; i < LED_TIME_LENGTH; i++);
    }
```

Initialize peripheral IPs and remapping addresses

```
    /* Reset IPs */
    SYS->IPRSTC1 = SYS_IPRSTC1_PDMA_RST_Msk;
    SYS->IPRSTC2 = (SYS_IPRSTC2_I2S_RST_Msk|SYS_IPRSTC2_ADC_RST_Msk
                   |SYS_IPRSTC2_USBD_RST_Msk|SYS_IPRSTC2_PS2_RST_Msk
                   |SYS_IPRSTC2_PWM03_RST_Msk|SYS_IPRSTC2_UART1_RST_Msk
                   |SYS_IPRSTC2_UART0_RST_Msk|SYS_IPRSTC2_SPI2_RST_Msk
                   |SYS_IPRSTC2_SPI1_RST_Msk|SYS_IPRSTC2_SPI0_RST_Msk
                   |SYS_IPRSTC2_I2C1_RST_Msk|SYS_IPRSTC2_I2C0_RST_Msk
                   |SYS_IPRSTC2_TMR3_RST_Msk|SYS_IPRSTC2_TMR2_RST_Msk
                   |SYS_IPRSTC2_TMR1_RST_Msk|SYS_IPRSTC2_TMR0_RST_Msk
                   |SYS_IPRSTC2_GPIO_RST_Msk);
    SYS->IPRSTC1 &= ~SYS_IPRSTC1_PDMA_RST_Msk;
    SYS->IPRSTC2 &= ~(SYS_IPRSTC2_I2S_RST_Msk|SYS_IPRSTC2_ADC_RST_Msk
                   |SYS_IPRSTC2_USBD_RST_Msk|SYS_IPRSTC2_PS2_RST_Msk
                   |SYS_IPRSTC2_PWM03_RST_Msk|SYS_IPRSTC2_UART1_RST_Msk
                   |SYS_IPRSTC2_UART0_RST_Msk|SYS_IPRSTC2_SPI2_RST_Msk
                   |SYS_IPRSTC2_SPI1_RST_Msk|SYS_IPRSTC2_SPI0_RST_Msk
                   |SYS_IPRSTC2_I2C1_RST_Msk|SYS_IPRSTC2_I2C0_RST_Msk
                   |SYS_IPRSTC2_TMR3_RST_Msk|SYS_IPRSTC2_TMR2_RST_Msk
                   |SYS_IPRSTC2_TMR1_RST_Msk|SYS_IPRSTC2_TMR0_RST_Msk
                   |SYS_IPRSTC2_GPIO_RST_Msk);


    /* Remap vector table address to 0x0 */
    CLK->APBCLK |= CLK_AHBCLK_ISP_EN_Msk;
    FMC->ISPCON |= FMC_ISPCON_ISPEN_Msk;
    FMC->ISPCMD = FMC_ISPCMD_VECMAP;
    FMC->ISPADR = APP1_ADDR;
    FMC->ISPTRG |= FMC_ISPTRG_ISPGO_Msk;
```

Reset stack point and execute APP1

```
    /* Set stack pointer to the value stored in APP1 address */
    pu32SP = (uint32_t *)APP1_ADDR;
    spChange(pu32SP);


    /* Execute another application */
    func = (FUNC_PTR *)(*(uint32_t *)(APP1_ADDR+ 4));
    func();
```

3 Software and Hardware Environment

Software Environment

- BSP version
  - NUC123 Series BSP CMSIS V3.01.001
- IDE version
  - Keil uVersion 5.26
- IDE environment setting
  - APP1 linker (R/O Base: 0x00000000, R/W Base: 0x20000000), As shown in Figure 3-1.

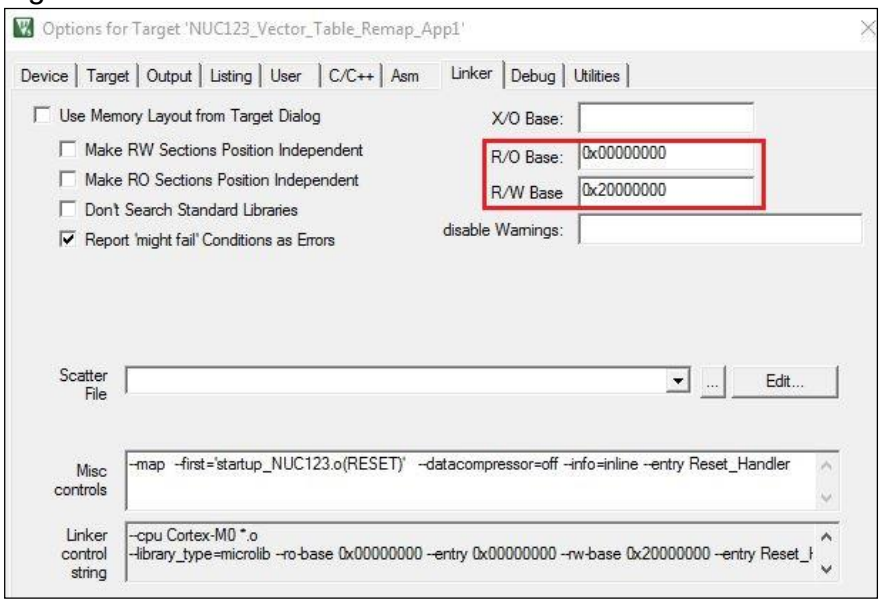

Figure 3-1 APP1 linker setting

- APP2 linker (R/O Base: 0x00003000, R/W Base: 0x20000000), as shown in Figure 3-2.

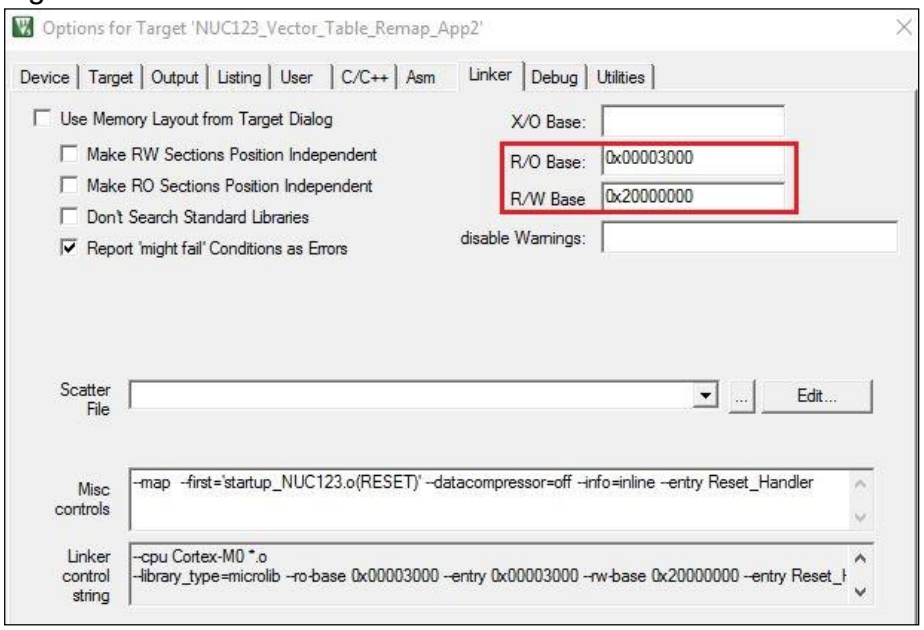

Figure 3-2 APP2 linker setting

Hardware Environment

- Circuit components
  - NuTiny-EVB-123-LQFP64_V1.0

4 Directory Information

- EC_NUC123_Vector_Table_Remap_V1.00
  - 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 — Source file of example code

5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click NUC123_Vector_Table_Remap_App1.uvprojx.
2. Enter Keil compile mode
   a. Build
   b. Download
3. Double click NUC123_Vector_Table_Remap_App2.uvprojx.
4. Enter Keil compile mode
   c. Build
   d. Download

6 Revision History

Date	Revision	Description
Aug. 7, 2019	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.