M032 ISR运行在SRAM

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

应用简述	本范例使用 M032 将中断重映射到 SRAM 运行
BSP 版本	M031_Series_BSP_CMSIS_V3.03.000
开发平台	NuMaker-M032SE V1.3

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

本示例使用M032 ISP功能将中断重映射到SRAM运行，实现当FMC操作Flash时仍能正常使用UART功能。

芯唐支持映射中断向量表到SRAM的芯片如下表，具体设置需根据不同情境而修改。

不支持映射中断向量表至SRAM	支持映射中断向量表至SRAM
M051 M058S M0518 M0519 Mini51 Mini55 Mini57 Mini58 Nano1x2 Nano100 NUC100 NUC122 NUC123 NUC131 NUC200 NUC029xAN NUC029xAE NUC029xDE NUC029xEE	Cortex®-M4 全系列 (VTOR 寄存器) Cortex®-M23 全系列 (VTOR 寄存器) M031 (FMC 支援) M0564 (FMC 支援) Nano103 (FMC 支援) NUC121/125 (FMC 支援) NUC126 (FMC 支援) NUC029xGE (FMC 支援)

表 1-1 支持映射中断向量表至 SRAM

1.1 原理

当Flash存储控制器（FMC）进行操作时(运行程序与所操作Flash处于同一个bank)，CPU任何访问Flash的操作都会造成CPU停机直到FMC完成操作。如果FMC执行页擦除操作，可能会造成CPU长时间等待。

为了避免这种情况，用户如需要在FMC页擦除时执行其他程序，可以把程序放在SRAM中执行。如需要响应中断，可先将中断向量空间及中断处理函数拷贝到SRAM中，然后使用ISP功能重映射向量空间到SRAM。

在IAP模式下，ISP功能可将SRAM连续的512字节映射为中断向量空间。如中断向量空间被重映射到SRAM中，CPU不需要访问Flash获取中断向量地址。当程序在SRAM中执行，CPU

不会访问Flash读取指令。故当中断产生时，CPU可以零等待读取SRAM中的中断向量并跳转到SRAM中的中断函数运行。图 1-1展示了向量表映射的流程。

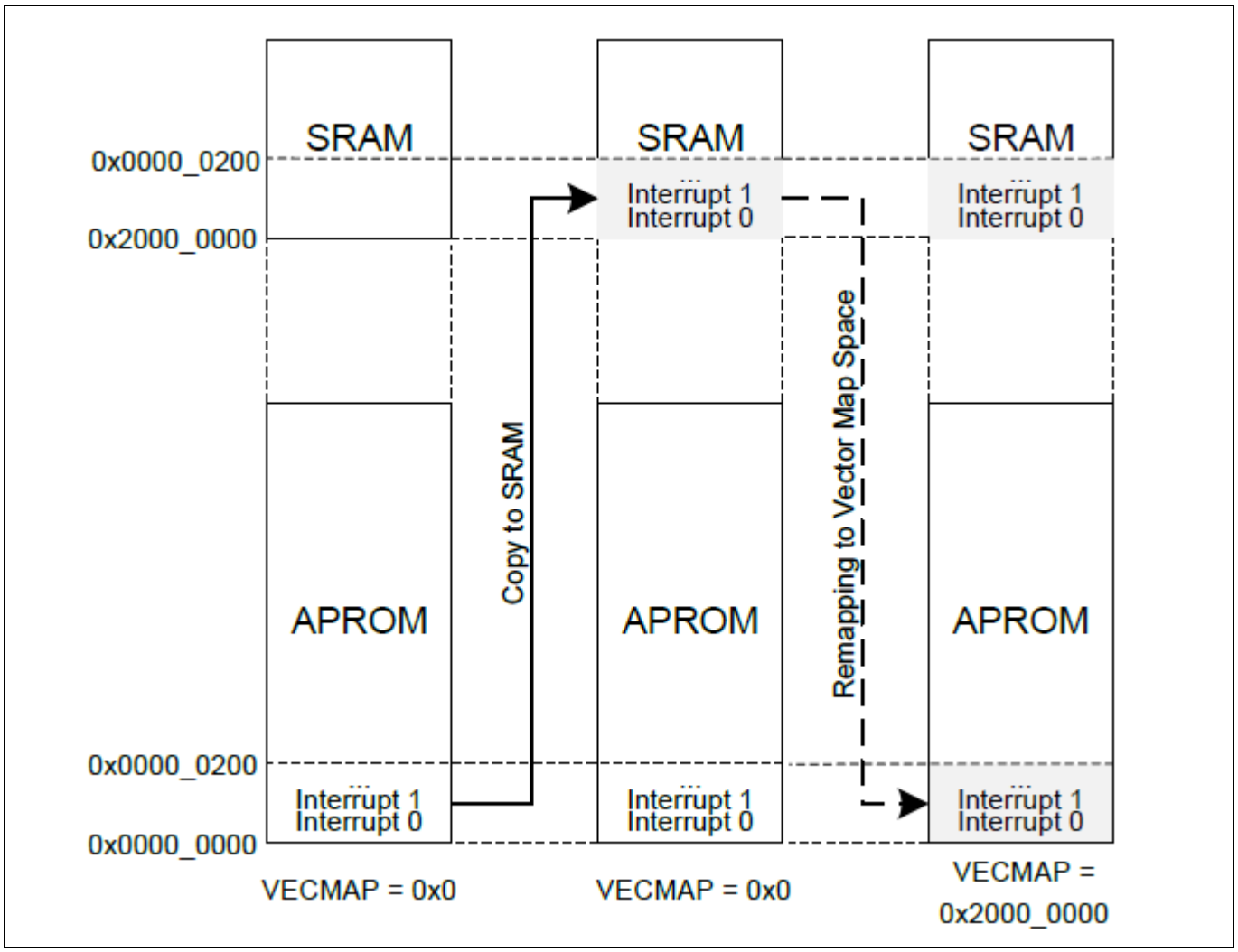

图 1-1 使用 ISP 将中断向量空间重映射到 SRAM 区域

此功能需使能芯片的 IAP 功能，芯片配置如下图 1-2。

图 1-2 芯片配置

1.2 执行结果

1.2.1 中断向量无映射到 SRAM

在 KEIL 目录下选择 **Project->Options for Target**'ISR_In_SRAM'，在“Preprocessor Symbols”去掉定义REMAP_ISR_IN_SRAM，再点击Build编译工程。待编译完成下载到开发板。

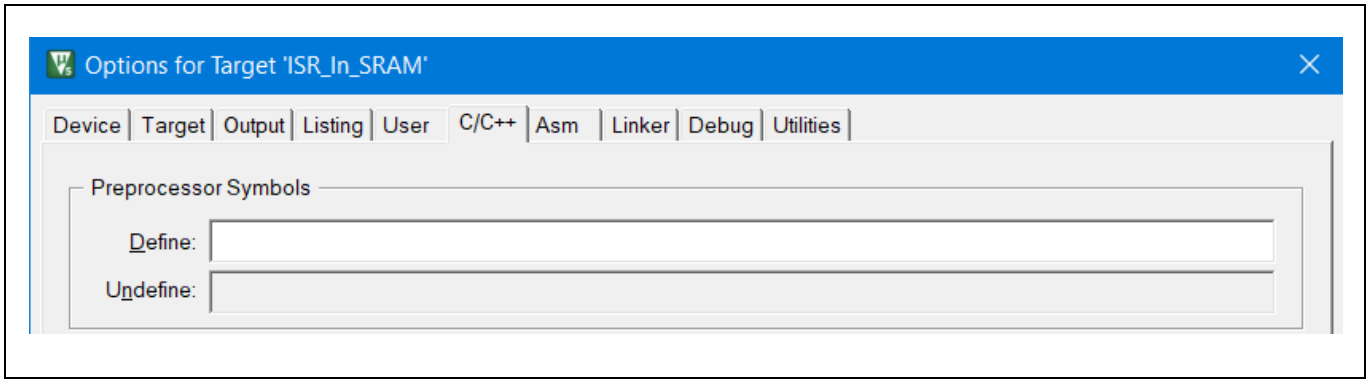

图 1-3 去掉 REMAP_ISR_IN_SRAM 定义

打开串口工具eagleCom，并使用 “周期发送”功能，间隔1ms每次发送400个字符‘1’，一直发送数据给开发板串口。运行结果如图 1-4所示。可以看出，总共发送了95600字节，实际接收到6480字节。

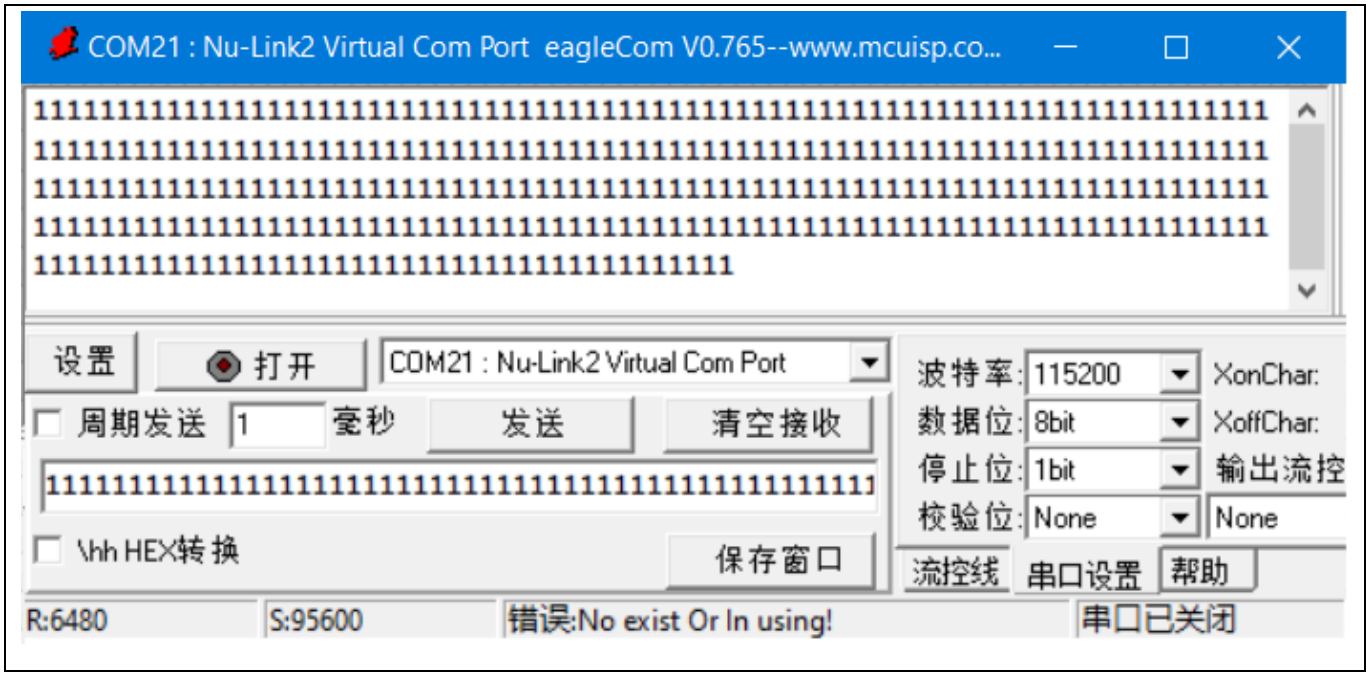

图 1-4 中断向量表在 Flash 上运行结果

使用逻辑分析仪抓取UART的数据波形，可以看到M032在收到UART数据后，存在19.2ms的时间段，没有处理并返回UART数据。这是因为M032 FMC在擦除Flash指定区域时，芯片无法同时访问处于同一bank的程序，导致存放在Flash的程序没办法及时执行。

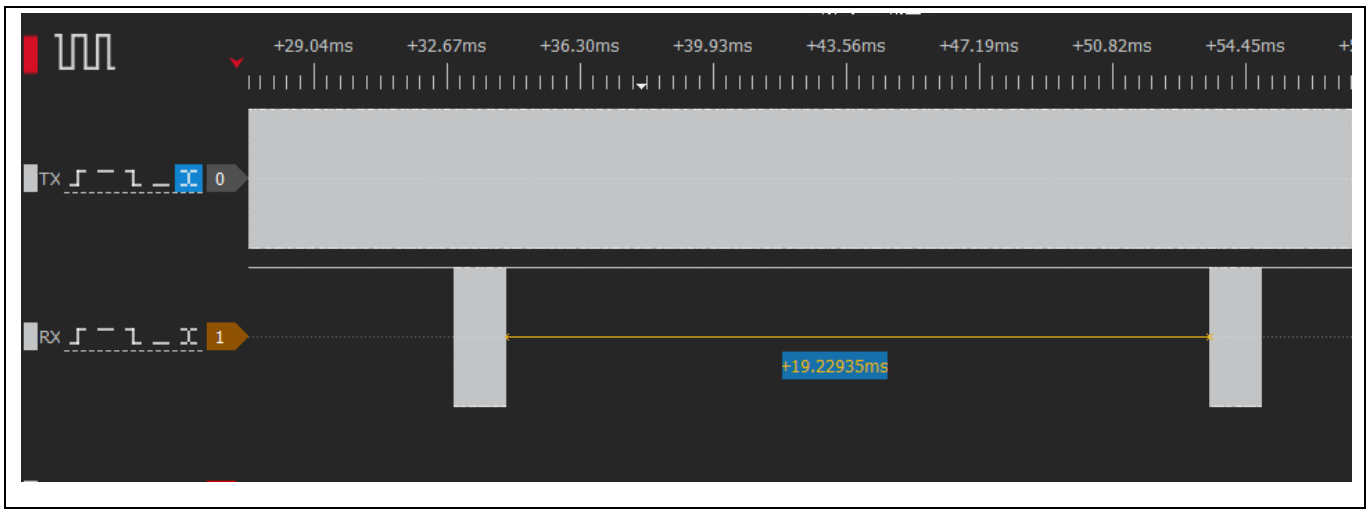

图 1-5 当 FMC 访问 Flash 时 UART 接收发送图 1

1.2.2 中断向量映射到 SRAM

在 KEIL 目录下选择 **Project->Options for Target**'ISR_In_SRAM'，在 "Preprocessor Symbols"中添加定义REMAP_ISR_IN_SRAM。再点击Build编译工程。待编译完成下载到开发板。

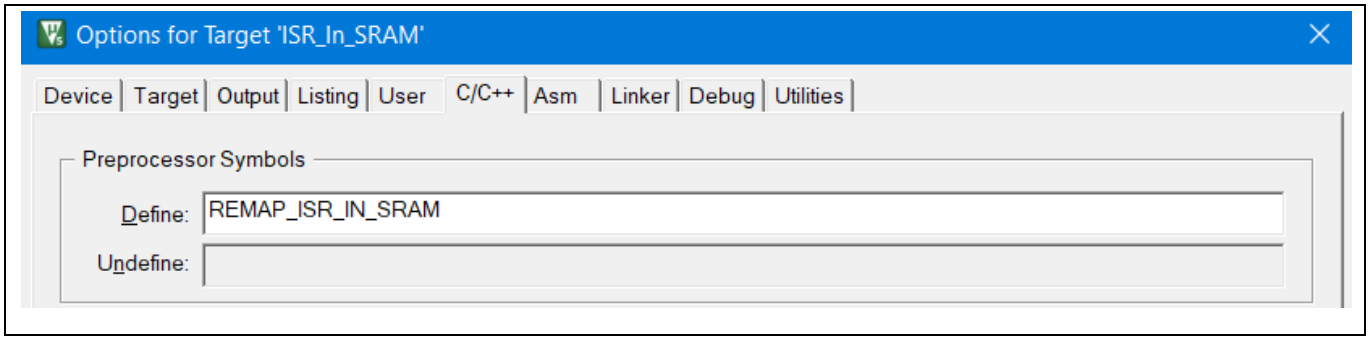

图 1-6 增加 REMAP_ISR_IN_SRAM 定义

打开串口工具eagleCom，并使用“周期发送”功能，间隔1ms每次发送400个字符‘1’，一直发送数据给开发板串口。运行结果如图 1-7所示。可以看出，总共发送了59200字节，实际接收到59200字节。

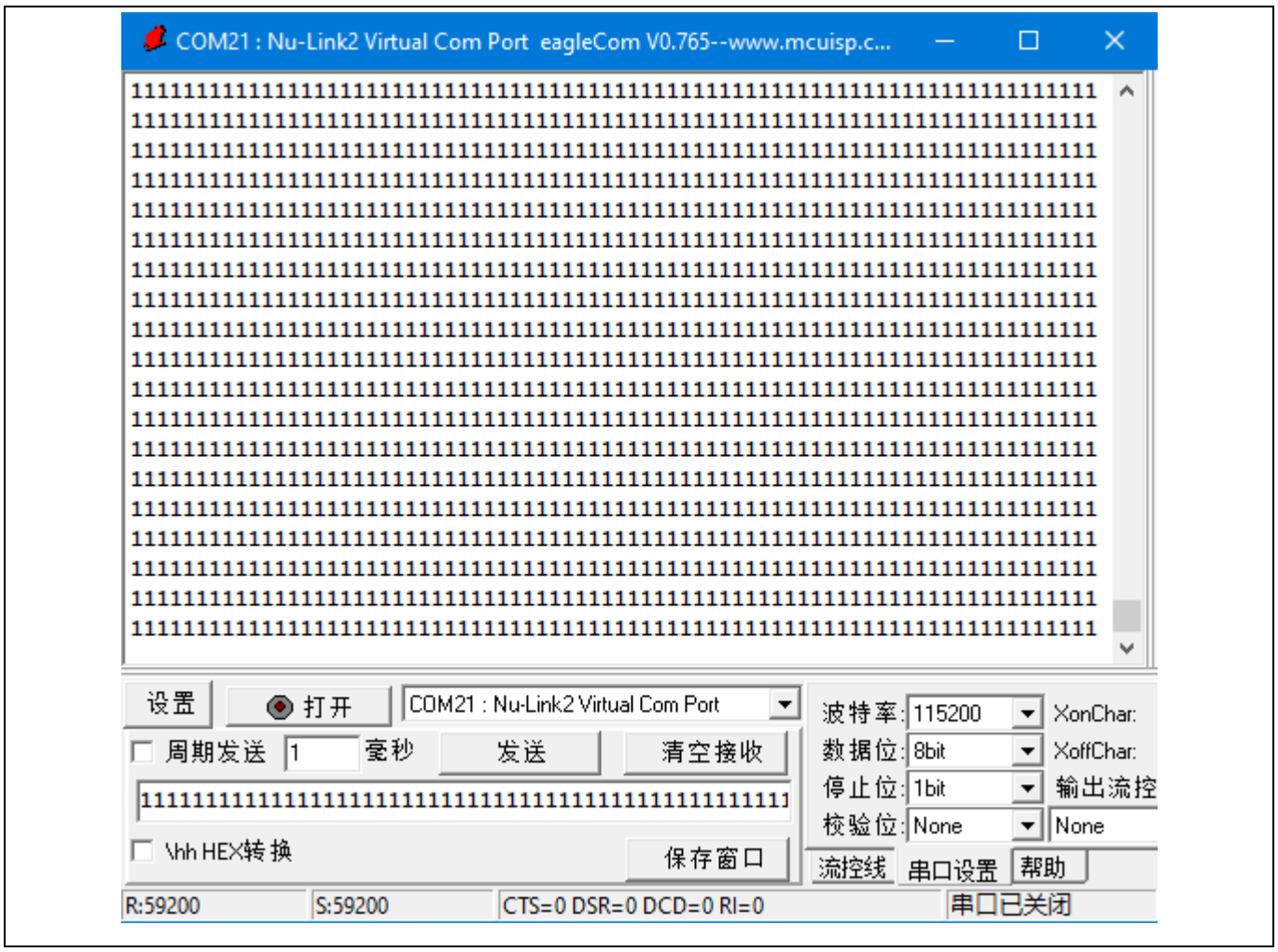

图 1-7 中断向量表在 SRAM 上运行结果

使用逻辑分析仪抓取UART的数据波形，如图 1-8可以看到M032在收到UART数据后，可以正常接收处理。所以将程序放在SRAM中执行并重映射中断向量空间到SRAM中可避免Flash编程时长时间停机等待。

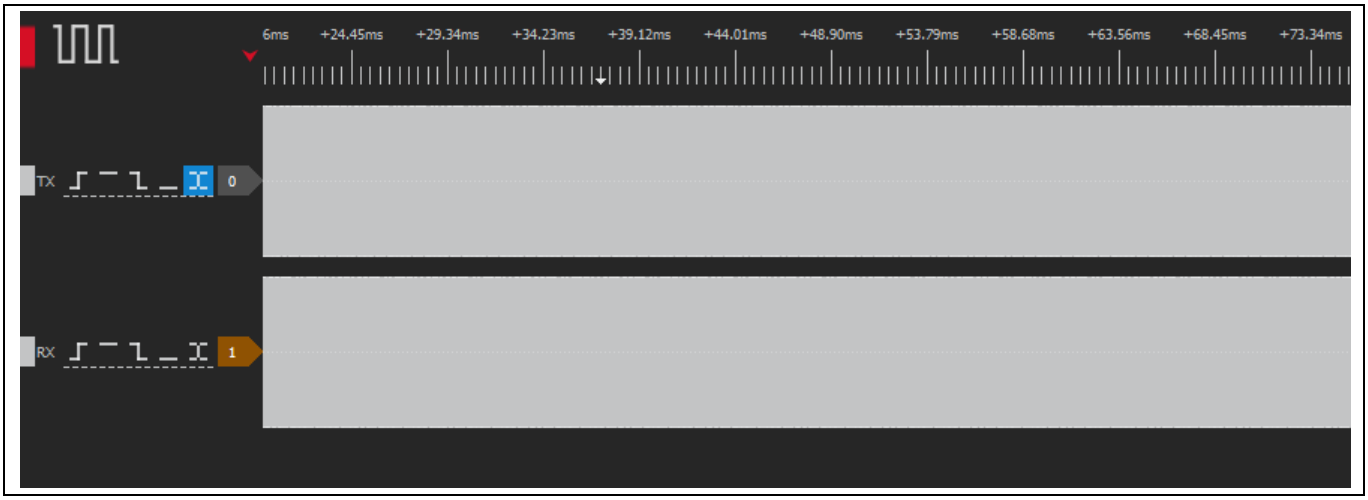

图 1-8 当 FMC 访问 Flash 时 UART 接收发送图 2

2 代码介绍

使用 Scatter File 分配芯片内存存储空间，在 SRAM 中划分出中断向量空间所需要的连续 512 字节地址空间。

```
FLASH 0x00000000
{
    ER_STARTU +0
    {
        startup_M031Series_SRAM.o(RESET, +First)
    }
     ER_RO +0
    {
        *(+RO)
    }

     ;on SRAM region
    VECTORS_SRAM 0x20000000 0x200 {
        *(VectorSection)
    }
    FASTCODE_INIT +0
    {
        uart.o(+RO)
        *(fastcode)
    }
    ER_RW +0
    {
       *(+RW, +ZI)
    }
}
```

在 SRAM 中自定义代码段 FASTCODE_INIT，将处理函数及其需要调用的函数放在此空间。请注意在此例程中 FMC 对 FLASH 的擦写操作也需放在 SRAM 中。

```
#pragma arm section code="fastcode"

void UART02_IRQHandler(void)
{
    uint8_t u8InChar = 0xFF;

    if (UART_GET_INT_FLAG(UART0, UART_INTSTS_RDAINT_Msk))
    {
        /* Get all the input characters */
        while (UART_IS_RX_READY(UART0))
        {
            /* Get the character from UART Buffer */
            u8InChar = UART_READ(UART0);

            UART_WRITE(UART0, u8InChar);
        }
    }

    if (UART0->FIFOSTS & (UART_FIFOSTS_BIF_Msk | UART_FIFOSTS_FEF_Msk |
UART_FIFOSTS_PEF_Msk | UART_FIFOSTS_RXOVIF_Msk))
    {
        UART_ClearIntFlag(UART0, (UART_INTSTS_RLSINT_Msk | UART_INTSTS_BUFERRINT_Msk));
    }
}

void FMC_ISPGO(void)
```

```
{
    FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk;
    while (FMC->ISPTRG & FMC_ISPTRG_ISPGO_Msk) { }
}
#pragma arm section
```

拷贝中断向量空间到 SRAM 指定地址并设置中断向量空间重映射地址。

```
#define VECMAP_SRAM_BASE    0x20000000UL
uint8_t u8Vectors_SRAM[512] __attribute__((section("VectorSection")));
    /* Lock protected registers */
    SYS_LockReg();

    if (FMC_GetVECMAP() != VECMAP_SRAM_BASE)
    {
        printf("Before setting VECMAP=0x%x\n\r", FMC_GetVECMAP());

        /* Mask all interrupt before changing VECMAP to avoid wrong interrupt handler
fetched */
        __set_PRIMASK(1);

        memcpy(u8Vectors_SRAM, &__Vectors, 512);

        /* Unlock protected register */
        SYS_UnlockReg();

        /* Enable FMC ISP function */
        FMC_Open();

        /* set VECMAP to SRAM */
        FMC_SetVectorPageAddr((uint32_t)&u8Vectors_SRAM);

        /* Disable FMC ISP function */
        FMC_Close();

        /* Lock protected Register */
        SYS_LockReg();

        __set_PRIMASK(0);

        printf("After  setting VECMAP=0x%x\n\r", FMC_GetVECMAP());
    }
```

3 软件与硬件需求

- 软件需求
  - BSP 版本
    - M031_Series_BSP_CMSIS_V3.03.000
  - IDE 版本
    - Keil uVersion 5.35
- 硬件需求
  - 电路组件
    - NuMaker-M032SE V1.3

4 目录信息

EC_M032_ISR_In_SRAM_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

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 ExampleCode 路径中的 KEIL 文件夹，双击 *ISR_In_SRAM.uvproj*。
2. 进入编译模式界面
   a. 编译
   b. 下载代码至内存
3. 打开串口工具选择对应串口
   a. 按下复位键后查看串口打印信息
   b. 使用逻辑分析仪抓取串口通信波形

6 修订纪录

Date	Revision	Description
2021.07.16	1.00	1. 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.