STM32H7 DMA 传输异常案例分析

前言

利用 ST 提供的辅助开发工具 STM32CubeMX，可以快速开发 STM32 应用。在本文中讨论的 ADC 应用，同样是建立在 STM32CubeMX 生成工程的基础上。具体为在 STM32H743ZI 上，利用 DMA 自动实现 ADC 数据周期采集和转移，在采集指定数量的采样值后产生中断，对数据进行处理。

文中仅对上述实现过程中出现的一种异常，进行介绍和分析。不涉及具体的 ADC 采集和处理实现的介绍。

一 实验环境

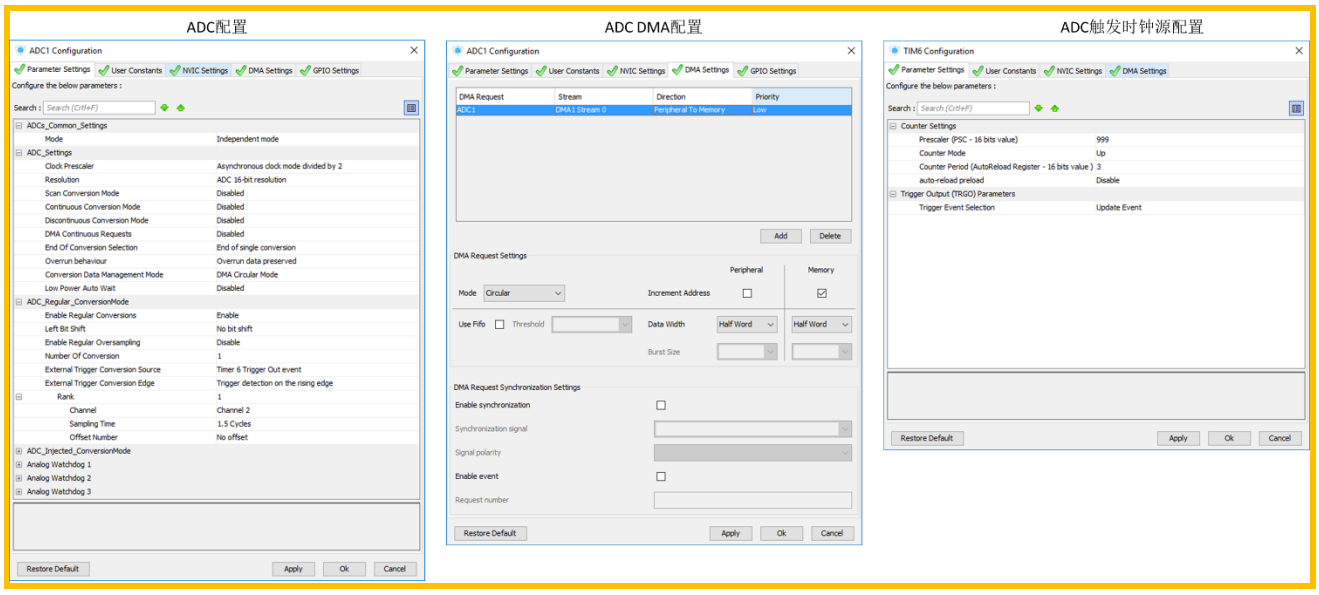
硬件平台：Nucleo-H743ZI

STM32CubeMX 版本：v4.28.0

STM32CubeH7 版本：v1.2.0

IDE： MDK-ARM v5.25.2.0 (优化级别 Level3)

首先利用 STM32CubeMX 生成 ADC 应用的初始化工程，涉及的配置如下所示：



然后在工程中增加对 ADC 周期采集和 DMA 传输实现的命令，如下所示。并且增加回调函数内容，本文中实现仅添加了空指令。在调试过程中，在空指令处增加断点，判断 ADC DMA 传输半完成和传输完成中断是否正常进入。

文件名	函数名	操作
main.c		/* USER CODE BEGIN PV */ #define BUFFER_SIZE ((uint32_t) 32) /* Variable containing ADC conversions data */ uint16_t aADCxConvertedData[BUFFER_SIZE]; /* USER CODE END PV */

		/* USER CODE BEGIN 4 */
		void HAL_ADC_ConvHalfCpltCallback(ADC_HandleTypeDef* hadc)
		{
		__nop;
		}
		void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
		{
		__nop;
		}
		/* USER CODE END 4 */

	main	/* USER CODE BEGIN 2 */ HAL_ADC_Start_DMA(&hadc1,(uint32_t *) aADCxConvertedData,BUFFER_SIZE); HAL_TIM_Base_Start(&htim6); /* USER CODE END 2 */

注： 如果在编译链接的过程中，出现没有定义 ECC_IRQn 提示，需将 stm32h7xx_hal_msp.c 中对 HAL_NVIC_SetPriority(ECC_IRQn, 0, 0);的调用移除。

二 异常现象

在运行过程中，无法进入 HAL_ADC_ConvHalfCpltCallback 和 HAL_ADC_ConvCpltCallback 回调函数（分别对应 DMA 传输半完成和传输完成回调函数）。

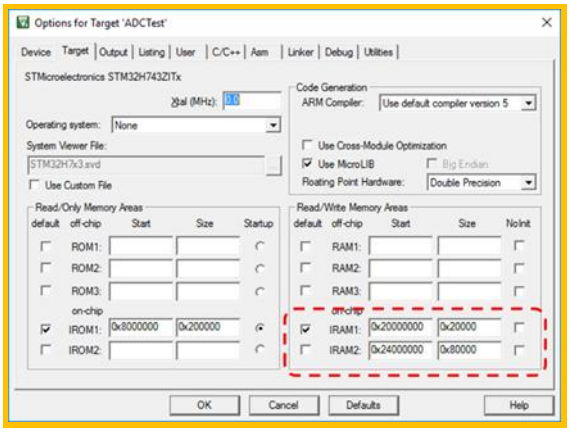
跟踪中断服务函数，发现在首次 DMA1 传输后，DMA_LISR 寄存器值为 0x8。即 TEIF0 置位，Stream 0 出现传输错误。

三 原因分析

根据错误信息，问题可能与 DMA 1 传输错误有关。重点围绕 DMA1 传输配置， 进行检查，并没有发现异常。

不过，同样的初始化配置和执行命令，放置在 ADC 例程（路径： STM32Cube_FW_H7_V1.2.0\Projects\STM32H743ZI-Nucleo\Examples\ADC\ADC_DMA_Transfer），进行编译链接后，能够正常执行。

比对两者间的工程配置，发现在 RAM 分配上存在差异。下图为 STM32CubeMX 生成工程中，对应的 RAM 分配情况。与此对应，例程中 RAM 分配至 IRAM2 (0x24000000)。



打开.map 文件可以看到：

a. 配置至 IRAM1 时：

aADCxConvertedData 0x200000b4 Data 64 main.o(.bss)

b. 配置至 IRAM2 时：

aADCxConvertedData 0x240000b4 Data 64 main.o(.bss)

而 aADCxConvertedData 数组设置为 DMA 目标地址。问题定位为 DMA 目标地址引起的异常。

aADCxConvertedData 数组都分配在 RAM 中，RAM 区域情况如下表所示。

区域	范围	空间大小(KB)	ARM Cortex-M7
RAM	0x38801000 -0x3FFFFFFF		预留
	0x38800000 -0x38800FFF	4	后备 SRAM
	0x38010000 -0x387FFFFF		预留
	0x38000000 -0x3800FFFF	64	SRAM4
	0x30048000 -0x37FFFFFF		预留
	0x30040000 -0x30047FFF	32	SRAM3
	0x30020000 - 0x3003FFFF	128	SRAM2
	0x30000000 -0x3001FFFF	128	SRAM1
	0x24080000 -0x2FFFFFFF		预留
	0x24000000 -0x2407FFFF	512	AXI SRAM
	0x20020000 -0x23FFFFFF		预留
	0x20000000 -0x2001FFFF	128	DTCM

STM32H7 内部包含三个域，每个区域中含有总线矩阵，具有不同的 DMA 主设备支持。不同区域 DMA 主设备能够访问的空间不同，如下表所示（下表摘自 RM0433，更多详细介绍请参考 RM0433）。其中，红色框中表示 DMA1 不支持对 DTCM 区域的访问。绿色框中，表示 DMA1 支持对 SRAM1/2/3/4，后备 SRAM 和 AXI SRAM 的访问。

上述不同的工程配置，分别将 aADCxConvertedData 分配至 DMA1 不能访问的 DTCM 区域和 DMA1 能够访问的 AXI SRAM 区域。 从而出现描述的现象。

Table 2. Bus-master-to-bus-slave interconnect

Bus slave / type[1]	Bus master / type[1] → Interconnect path and type[2]																
	Cortex-M7 - AXIM	Cortex-M7 - AHBP	Cortex-M7 - ITCM	Cortex-M7 -DTCM	SDMMC1	MDMA - AXI	MDMA - AHBS	DMA2D	LTDC	DMA1 - MEM	DMA1 - PERIPH	DMA2 - MEM	DMA2 - PERIPH	Eth. MAC - AHB	SDMMC2 - AHB	USBHS1 - AHB	USBHS2 - AHB	BDMA - AHB
ITCM	-	-	X	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-
DTCM	-	-	-	X	-	-	X	-	-	-	-	-	-	-	-	-	-	-
AHB3 peripherals	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	-
APB3 peripherals	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	-
Flash bank 1	X	-	-	-	X	X	-	X	X	X	X	X	X	X	X	X	X	-
Flash bank 2[3]	X	-	-	-	X	X	-	X	X	X	X	X	X	X	X	X	X	-
AXI SRAM	X	-	-	-	X	X	-	X	X	X	X	X	X	X	X	X	X	-
QUADSPI	X	-	-	-	X	X	-	X	X	X	X	X	X	X	X	X	X	-
FMC	X	-	-	-	X	X	-	X	X	X	X	X	X	X	X	X	X	-
SRAM 1	X	-	-	-	-	X	-	X	-	X	X	X	X	X	X	X	X	-
SRAM 2	X	-	-	-	-	X	-	X	-	X	X	X	X	X	X	X	X	-
SRAM 3	X	-	-	-	-	X	-	X	-	X	X	X	X	X	X	X	X	-
AHB1 peripherals	-	X	-	-	-	X	-	X	-	X	X	X	X	-	-	-	-	-
APB1 peripherals	-	X	-	-	-	X	-	X	-	X	X	X	X	-	-	-	-	-
AHB2 peripherals	-	X	-	-	-	-	-	-	-	X	X	X	X	-	-	-	-	-
APB2 peripherals	-	X	-	-	-	X	-	X	-	X	X	X	X	-	-	-	-	-
AHB4 peripherals	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X
APB4 peripherals	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X
SRAM4	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X
Backup RAM	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X

1. **Bold** font type denotes 64-bit bus, plain type denotes 32-bit bus.
2. "X" = access possible, "-" = access not possible, shading = access useful/usable.
3. Bank 2 is not available on STM32H750xB devices.

四 解决方法

通过上述分析，问题产生的原因在于 DMA1 访问了无法访问到的地址。

介绍两种解决方法，方法一，通过原因分析中描述的方法，通过在工程中，指定 RAM 使用空间实现目标地址（访问的数组）在支持的 RAM 区域。方法二，通过在数组定义时，强制分配至支持的 RAM 空间区域，如下所示。

```
uint16_t  aADCxConvertedData[BUFFER_SIZE] __attribute__((section(".ARM.__at_0x24000000")));
```

其中 __attribute__((section(".ARM.__at_address")))是被 MDK-ARM 支持的，指定对应空间的方式。

小结

STM32H7 相对于之前 STM32 系列，具有更灵活，更多样的内部区域，以及总线矩阵。带来应用灵活性提升的同时，也一定程度上增加了使用的复杂度。用户在进行 STM32H7 开发时，要注意各区域和多区域之间应用实现的实际情况。在遇到问题时，可与 ST 提供的大量例程进行比对，并结合数据手册和参考手册，进行问题的定位和解决。

参考文档

RM0433 STM32H743/753 and STM32H750 advanced ARM®-based 32-bit MCUs

DS12110 32-bit Arm® Cortex®-M7 400MHz MCUs, up to 2MB Flash,1MB RAM, 46 com. and analog interfaces

重要通知 – 请仔细阅读

意法半导体公司及其子公司（“ST”）保留随时对ST 产品和/ 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于ST 产品的最新信息。ST 产品的销售依照订单确认时的相关ST 销售条款。

买方自行负责对ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的ST 产品如有不同于此处提供的信息的规定，将导致ST 针对该产品授予的任何保证失效。

ST 和ST 徽标是ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。