life.augmented

AN4104
应用笔记

使用 STM32F0xx DMA 控制器

前言

本应用笔记介绍了如何使用 STM32F0xx 直接存储器访问 (DMA) 控制器。STM32F0xx 的 DMA 控制器、Cortex™-M0 内核、高级微控制器总线架构 (AMBA) 的总线和存储器系统为用户开发高数据带宽低延迟响应的软件提供了便利条件。

本应用笔记还介绍了如何充分利用这些特性以及确保不同的外设和子系统具有正确的响应时间。

注： *为确保用户快速入门，本文档介绍的应用示例均用 C 语言编程，这些示例位于 STM32F0xx_StdPeriph_Lib 软件包内的 Project\STM32F0xx_StdPeriph_Examples 下。*

目录

1 DMA 控制器说明

直接存储器访问 (DMA) 用于在外设与存储器之间以及存储器与存储器之间提供高速数据传输。可以在无需任何 CPU 操作的情况下通过 DMA 快速传输传输。这样节省的 CPU 资源可供其它操作使用。

DMA 允许在后台执行数据传输，无需 Cortex-M0 处理器干预。在此操作过程中，主处理器可以执行其它任务，仅当整个数据块需要处理时，才会中断主处理器。这样即使传输大量数据也不会对系统性能造成太大影响。

DMA 主要用于为不同的外设模块实现集中数据缓冲存储（通常在系统 SRAM 中）。与分布式解决方案（其中每个外设都需要实现自己的本地数据存储）相比，DMA 解决方案在硅片成本和功耗方面的成本较低。

根据使用的产品型号的不同，有一个或两个 DMA 模块。

STM32F0xx DMA 控制器总共有 5 个通道用于 DMA1，每个通道都专门管理来自一个或多个外设的存储器访问请求。它具有一个仲裁器，用于处理不同的 DMA 请求的优先级。

1.1 DMA 概述

DMA 主要特性：

- 可单独配置的通道（请求）
- 每个通道都与专用的硬件 DMA 请求相连，同样每个通道上都支持软件触发
- 来自一个 DMA 的通道的请求间的优先级可用软件编程（4 个级别：很高、高、中等、低），在软件优先级相同的情况下可以通过硬件决定优先级（例如，请求 1 的优先级高于请求 2）
- 独立的源和目标传输大小（字节、半字、字），模拟打包和解包。源和目标必须具有相同的数据大小（以通过数据大小对齐）
- 支持循环缓冲区管理
- 3 个事件标志（DMA 半传输、DMA 传输完成和 DMA 传输错误），这 3 个事件标志逻辑或成为一个单独的中断请求
- 存储器到存储器的传输
- 外设到存储器和存储器到外设以及外设到外设的传输
- FLASH、SRAM、APB 和 AHB 外设均可作为访问的源和目标
- 可编程的数据传输数目：最大 65536

DMA 旨在为所有外设提供相对较大的数据缓冲区。此缓冲区通常位于系统 SRAM 中。

每个通道都会在给定时间分配到一个唯一外设（数据通道）。连接到同一 DMA 通道（对于 STM32F0xx 器件，为*表 1* 中的 CH1 到 CH5）的外设不能与使能的 DMA（DMA 功能在外设寄存器中处于使能状态）同时使用。

表 1 中显示了 STM32F0xx 器件中支持 DMA 传输的各种不同外设。

表 1. 支持 DMA1 的外设和通道分配

外设		CH1	CH2	CH3	CH4	CH5
ADC	**ADC1**	ADC1	ADC1			
SPI	**SPI1**		SPI1_RX	SPI1_TX		
	SPI2				SPI2_RX	SPI2_TX
USART	**USART1**		USART1_TX	USART1_RX	USART1_TX	USART1_RX
	USART2				USART2_TX	USART2_RX
I^2C	**I^2C1**		I2C1_TX	I2C1_RX		
	I^2C2				I2C2_TX	I2C2_RX
TIM	**TIM1**		TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP TIM1_CH3
	TIM2	TIM2_CH3	TIM2_UP	TIM2_CH2	TIM2_CH4	TIM2_CH1
	TIM3		TIM3_CH3	TIM3_CH4 TIM3_UP	TIM3_CH1 TIM3_TRIG	
	TIM6/DAC			TIM6_UP DAC		
	TIM15					TIM15_CH1 TIM15_UP TIM15_TRIG TIM15_COM
	TIM16			TIM16_CH1 TIM16_UP	TIM16_CH1 TIM16_UP	
	TIM17	TIM17_CH1 TIM17_UP	TIM17_CH1 TIM17_UP			

注： *有关详细信息，请参见 STM32F0xx 器件参考手册 RM0091 DMA 一节。*

1.2 DMA 数据管理

DMA 控制器通过与 Cortex-M0 内核共用系统总线来执行直接存储器访问。当 CPU 与 DMA 具有相同的目标（存储器或外设）时，DMA 请求会暂停 CPU 访问系统总线达若干个周期，总线仲裁器执行循环调度，以确保 CPU 至少可以获得一半的系统总线带宽（存储器和外设）。

1.2.1 循环优先级方案

NVIC 和 Cortex-M0 处理器实施了延迟非常低的高性能中断方案。所有 Cortex-M0 指令均可在一个周期内执行，也可以按周期进行中断。为了在系统级保持这一优势，DMA 和总线矩阵可确保 DMA 不会长时间阻塞总线。循环优先级方案确保 CPU 可以根据需要每隔一个周期访问任意从总线一次。

1.2.2 外设到存储器、存储器到外设以及外设到外设的 DMA 传输

在发生一个事件后，外设向 DMA 控制器发送一个请求信号。DMA 控制器根据通道的优先权处理请求。当 DMA 控制器开始访问发出请求的外设时，DMA 控制器立即发送给它一个应答信号。当从 DMA 控制器得到应答信号时，外设立即释放它的请求。一旦外设释放了这个请求，DMA 控制器同时撤销应答信号。如果有更多的请求时，外设可以启动下一个周期。

每次 DMA 传输包含三个操作：

- 从外设数据寄存器或者从当前外设/存储器地址寄存器指示的存储器地址取数据，第一次传输时的开始地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设基地址或存储器单元。
- 向外设数据寄存器或者从当前外设/存储器地址寄存器指示的存储器地址存数据，第一次传输时的开始地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设基地址或存储器单元。
- 对 DMA_CNDTRx 寄存器执行一次递减操作，DMA_CNDTRx 寄存器存放着未完成的 DMA 操作的计数。

1.2.3 存储器到存储器的 DMA 传输

DMA 通道的操作可以在没有外设请求的情况下进行。此模式称为“存储器到存储器”模式。如果 MEM2MEM 位置 1，则只要软件通过将使能位置 1 来使能通道，通道就会启动传输。

DMA 计数器达到零后，传输结束。存储器到存储器模式不能与循环模式同时使用。

1.2.4 选择通道优先级

为了实现与外设之间的连续数据传输，相应的 DMA 通道必须能够维持足够的外设数据速率，并确保传输延迟的时间小于两个连续数据的间隔时间。

高速/高带宽外设必须具有最高的 DMA 优先级。符合外设最大数据延迟并且避免发生过载/欠载情况。

在带宽要求相同的情况下，建议为工作在从模式的外设（对数据传输速度没有控制权）分配的优先级高于工作在主模式的外设（可以控制数据流）。

默认情况下，对通道分配和硬件优先级（对于 STM32F0xx 器件，为 1 到 5）进行设置，以将最快的外设分配给优先级最高的通道。然而，对于某些应用来说，并不一定适合。在这种情况下，用户可以为每个通道配置一个软件优先级（4 个级别 —— 从“最高”到“低”），它将优先于硬件优先级。

1.3 DMA 中断管理

对于每个 DMA 通道，在发生“半传输”、“传输完成”或“传输错误”时都可以产生中断。可以使用单独的中断使能位以实现灵活的配置。

表 2. DMA 中断请求

中断事件	事件标志	控制位使能
半传输	HTIF	HTIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

当在 DMA 读写操作时发生 DMA 传输，错误时，硬件会自动地清除发生错误的通道所对应的通道配置寄存器 (DMA_CCRx) 的 EN 位，该通道操作被停止。此时，在 DMA_IFR 寄存器中对应该通道的传输错误中断标志位 (TEIF) 将被置位，如果在 DMA_CCRx 寄存器中设置了传输错误中断允许位，则将产生中断。

2 DMA 固件驱动程序 API

此驱动程序提供一系列固件函数来管理 DMA 外设的下列功能：

- 初始化和配置函数
- 数据计数器函数
- 中断和标志管理函数

对于 STM32F0xx 系列，DMA 驱动程序 stm32f0xx_dma.c/.h 位于以下目录：STM32F0xx_StdPeriph_Lib_vX.Y.Z\Libraries\STM32F0xx_StdPeriph_Driver。

此驱动程序提供完全兼容的 API，从而能轻松实现产品之间的迁移。

表 3. DMA 函数说明

组	函数名称	说明
初始化和配置函数	DMA_DeInit	将 DMAy Channelx 寄存器取消初始化为默认复位值。
	DMA_Init	根据 DMA_InitStruct 中指定的参数初始化 DMAy Channelx。
	DMA_StructInit	为每个 DMA_InitStruct 成员填充缺省值。
	DMA_Cmd	使能或禁止相应的 DMAy Channelx。
数据计数器函数	DMA_SetCurrDataCounter	设置当前 DMAy Channelx 传输中的数据单元数。
	DMA_GetCurrDataCounter	返回当前 DMAy Channelx 传输中的剩余数据单元数。
中断和标志管理函数	DMA_ITConfig	使能或禁止相应的 DMAy Channelx 中断。
	DMA_GetFlagStatus	获取相应的 DMAy Channelx 标志是否置 1 或清 0。
	DMA_ClearFlag	将 DMAy Channelx 的挂起标志清零。
	DMA_GetITStatus	检查是否发生了 DMAy Channelx 中断。
	DMA_ClearITPendingBit	将 DMAy Channelx 的中断挂起位清零。

为了使 DMA 传输能够进行，必须指定多个参数，如源/目标地址（将要读取或传输数据的位置）以及传输长度。

DMA 域配置存储在如下所述的结构体中：

- **DMA_PeripheralBaseAddr**：指定 DMAy Channelx 的外设基址。
- **DMA_MemoryBaseAddr**：指定 DMAy Channelx 的存储器基址。
- **DMA_DIR**：指定外设是源还是目标。
- **DMA_BufferSize**：指定所用通道的缓冲区大小（以数据单元为单位）。数据单元等于 DMA_PeripheralDataSize 或 DMA_MemoryDataSize 部分中根据传输方向设置的配置。
- **DMA_PeripheralInc**：指定在传输每个单元之后外设地址寄存器是否递增。
- **DMA_MemoryInc**：指定在传输每个单元之后存储器地址寄存器是否递增。
- **DMA_PeripheralDataSize**：指定外设数据宽度。传输单元的大小可以是字节、半字或字。
- **DMA_MemoryDataSize**：指定存储器数据宽度。传输单元的大小可以是字节、半字或字。
- **DMA_Mode**：指定 DMAy Channelx 的操作模式（正常模式或循环模式）。
- **DMA_Priority**：指定 DMAy Channelx 的软件优先级。
- **DMA_M2M**：配置 DMAy-Channelx 是否用来做为存贮器到存贮器访问用。

注：　有关更多详细信息，请参见 STM32F0x 器件的参考手册 RM0091 中的 DMA 章节。

2.1 如何使用 DMA 驱动程序

1. 在使用 DMA 驱动程序之前，请使用 RCC_AHBPeriphClockCmd (RCC_AHBPeriph_DMAX, ENABLE) 函数使能 DMA 控制器时钟。
2. 使能并配置要连接到 DMA 通道的外设（内部 SRAM/Flash 除外：不需要进行初始化）。
3. 对于给定通道，使用 DMA_Init() 函数设定源地址和目标地址、传输方向、缓冲区大小、外设和存储器递增模式和数据大小、循环或正常模式、通道传输优先级以及存储器到存储器传输模式（如果需要）。
4. 如果需要使用 DMA 中断，则使用函数 DMA_ITConfig() 使能 NVIC 和相应中断。
5. 使用 DMA_Cmd() 函数使能 DMA 通道。
6. 使用 PPP_DMACmd() 函数为除了内部 SRAM 和 Flash 外的合适 PPP 外设（即 SPI、USART...）激活所需通道请求。每个 PPP 外设驱动程序（即 SPI 外设的 SPI_DMACmd）都提供了允许此操作的函数。
7. 可以使用 DMA_SetCurrDataCounter() 函数来配置在禁止通道时（即，每个传输完成事件之后或发生传输错误时）传输的数据数目。还可以在运行时（DMA 通道已使能且正在运行时）使用 DMA_GetCurrDataCounter() 函数获取将要传输的剩余数据数目。
8. 要控制 DMA 事件，可以使用下列两种方法之一：
   - 使用 DMA_GetFlagStatus() 函数检查 DMA 通道标志。
   - 在初始化阶段通过 DMA_ITConfig() 函数、在通信阶段通过 DMA_GetITStatus() 函数配置 DMA 来使用中断程序。在检查到标志后，使用 DMA_ClearFlag() 函数将其清除。在检查到中断事件后，使用 DMA_ClearITPendingBit() 函数将其清除。

3 DMA 编程示例

DMA 固件驱动程序提供了一组示例，以帮助用户快速熟悉 DMA 外设。它演示了如何在不同模式下使用 DMA。

程序包和应用笔记均可以从意法半导体网站 http://www.st.com 下载。

3.1 ADC DMA 传输至 TIM 示例

此示例介绍了如何使用 DMA 通道从一个外设 (ADC) 向另一个支持 DMA 传输的外设 (TIM) 连续传输数据。将 ADC 配置为以连续转换模式工作。将 TIM 配置为在其输出中生成 PWM 信号。

将专用 DMA 通道配置为在循环模式下将最后一个 ADC 通道转换的值传输到 TIMER 捕捉/比较寄存器。DMA 通道请求由 TIM 更新事件驱动。然后，每次修改 ADC 通道引脚上的输入电压值时，TIMER 通道输出信号的占空比都会发生变化。

使用电位计更改 ADC 通道上的模拟输入时，示波器上可以显示占空比变化。

3.2 DMA Flash 至 RAM 示例

此示例说明了如何使用 DMA 在两个存储器单元之间传输数据。

其中介绍了如何使用 DMA 通道将 Flash 中的字数据缓冲区传输到嵌入式 SRAM 存储器。

DMA 通道配置为将 Flash 中存储的 32 位 WORD 字数据缓冲区的内容传输到 RAM 中声明的接收缓冲区。

传输的启动由软件触发。使能 DMA 通道存储器到存储器传输。配置源地址和目标地址为递增模式。

通过将 DMA 通道的通道使能位置 1 来启动传输。在传输结束时，将生成传输完成中断，因为已使能该中断。产生中断后，将读取要传输的剩余数据，该数据数目必须等于 0（如果已传输所有数据，DMA 计数器会达到 0）。然后将传输完成中断挂起位清零。

还将比较源缓冲区与目标缓冲区，以检查是否所有数据都已正确传输。

3.3 DMA RAM 至 DAC 示例

此示例介绍了如何使用 DMA 通道将数据缓冲区从存储器（RAM 存储器）传输到外设 DAC。

DMA 通道配置为以连续方式将半字缓冲区从 RAM 存储器逐字传输到 DAC 寄存器 DAC_DHR12R。DAC 通道转换配置为通过 TIM2 TRGO 触发信号触发，并且不生成噪声/三角波。由于选择了访问 DAC_DHR12R 寄存器，因此会选择 12 位数据右对齐方式。

3.4 SPI DMA 示例：使用 DMA 在两个 SPI 之间通信

此示例说明了如何使用 DMA 进行 SPI 通信。

在作为主机模式的电路板中，SPI 外设配置为具有 DMA 和 NSS 硬件模式的全双工主器件。

TIM2 配置为在 TIM2_CH2 引脚 (PA.01) 上生成占空比为 50% 的 4 KHz PWM 信号，此信号用作 DMA 触发信号以及用于锁存 SPI 数据传输的 NSS 信号输入。而在作为从机的电路板中，SPI 外设配置为具有 DMA 和 NSS 硬件模式的全双工从器件。

- 主器件使用 TIM2_CH2 DMA 请求 (DMA1_Channel3) 向从器件发送特定命令（此命令包含传送代码（CMD_RIGHT、CMD_LEFT、CMD_UP、CMD_DOWN 或 CMD_SEL）并使用 SPI_Rx DMA 请求 (DMA1_Channel2) 接收从器件的 ACK 命令）。
- 从器件使用 SPI_Rx DMA 请求 (DMA1_Channel2) 接收命令并使用 SPI_Tx DMA 请求 (DMA1_Channel3) 发送 ACK 命令。

3.5 使用 DMA 的 USART 通讯板间数据交换示例

此示例提供了使用 DMA 进行 USART 通信的小型应用。

在两个电路板中，均使用 USART Tx/Rx 通道 DMA 请求来管理数据传输。

4 版本历史

表 4. 文档版本历史

日期	版本	变更
2012 年 05 月 02 日	1	初始版本

请仔细阅读：

中文翻译仅为方便阅读之目的。该翻译也许不是对本文档最新版本的翻译，如有任何不同，以最新版本的英文原版文档为准。

本文档中信息的提供仅与ST产品有关。意法半导体公司及其子公司（“ST”）保留随时对本文档及本文所述产品与服务进行变更、更正、修改或改进的权利，恕不另行通知。

所有ST产品均根据ST的销售条款出售。

买方自行负责对本文所述ST产品和服务的选择和使用，ST概不承担与选择或使用本文所述ST产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为 ST 授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在ST的销售条款中另有说明，否则，ST对ST产品的使用和/或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

意法半导体的产品不得应用于武器。此外，意法半导体产品也不是为下列用途而设计并不得应用于下列用途：（A）对安全性有特别要求的应用，例如，生命支持、主动植入设备或对产品功能安全有要求的系统；（B）航空应用；（C）汽车应用或汽车环境，且/或（D）航天应用或航天环境。如果意法半导体产品不是为前述应用设计的，而采购商擅自将其用于前述应用，即使采购商向意法半导体发出了书面通知，采购商仍将独自承担因此而导致的任何风险，意法半导体的产品设计规格明确指定的汽车、汽车安全或医疗工业领域专用产品除外。根据相关政府主管部门的规定，ESCC、QML或JAN正式认证产品适用于航天应用。

经销的ST产品如有不同于本文档中提出的声明和/或技术特点的规定，将立即导致ST针对本文所述ST产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大ST的任何责任。

意法半导体集团公司

澳大利亚 - 比利时 - 巴西 - 加拿大 - 中国 - 捷克共和国 - 芬兰 - 法国 - 德国 - 中国香港 - 印度 - 以色列 - 意大利 - 日本 - 马来西亚 - 马耳他 - 摩洛哥 – 菲律宾 – 新加坡 - 西班牙 - 瑞典 - 瑞士 - 英国 - 美国

www.st.com