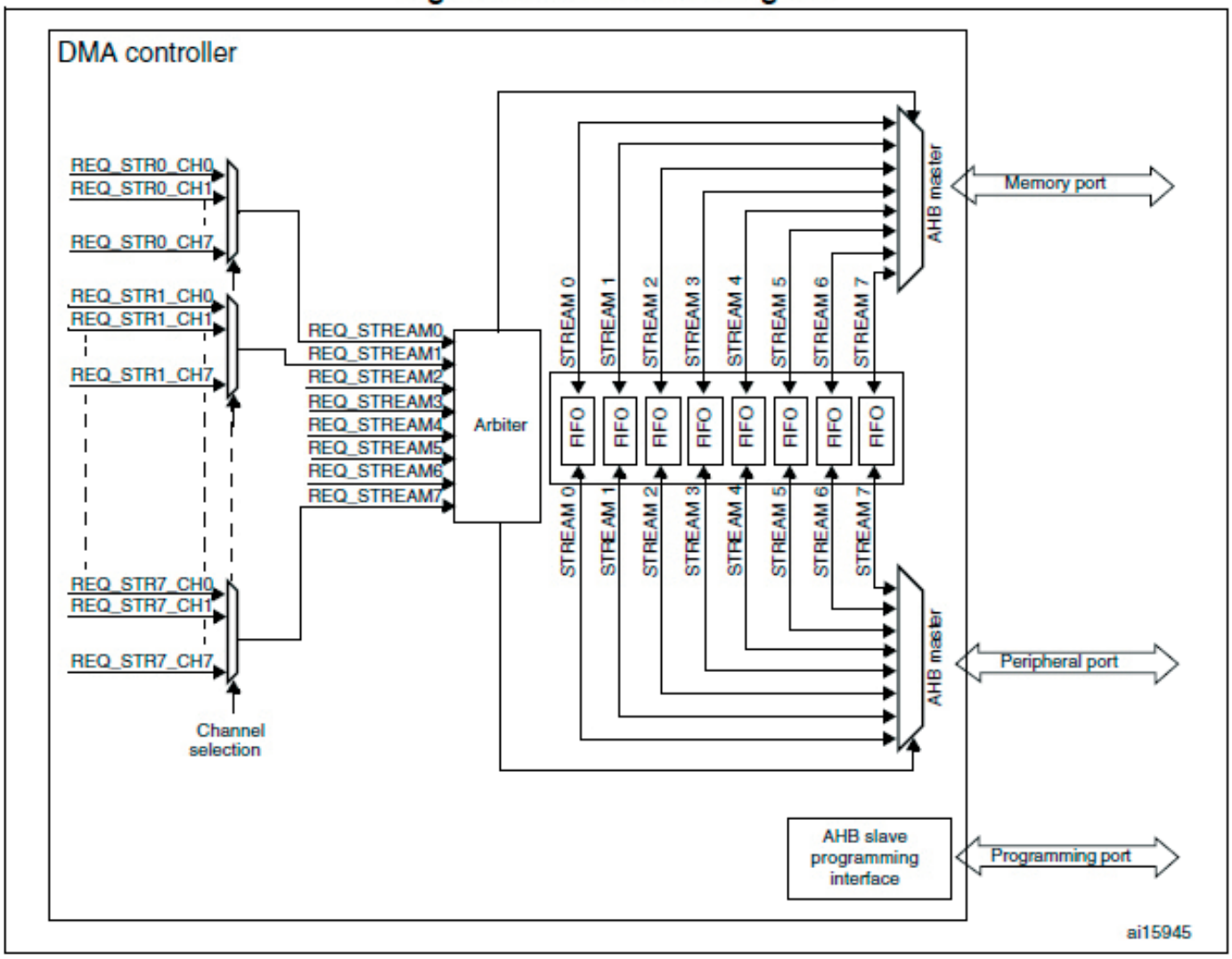
10.3 Функциональное описание DMA

10.3.1 Общее описание

На рисунке 32 показана блок-схема DMA.

Figure 32. DMA block diagram



Контроллер DMA выполняет прямую передачу памяти: как ведущий AHB, он может взять под контроль матрицу шины AHB для инициирования транзакций AHB.

Он может выполнять следующие транзакции:

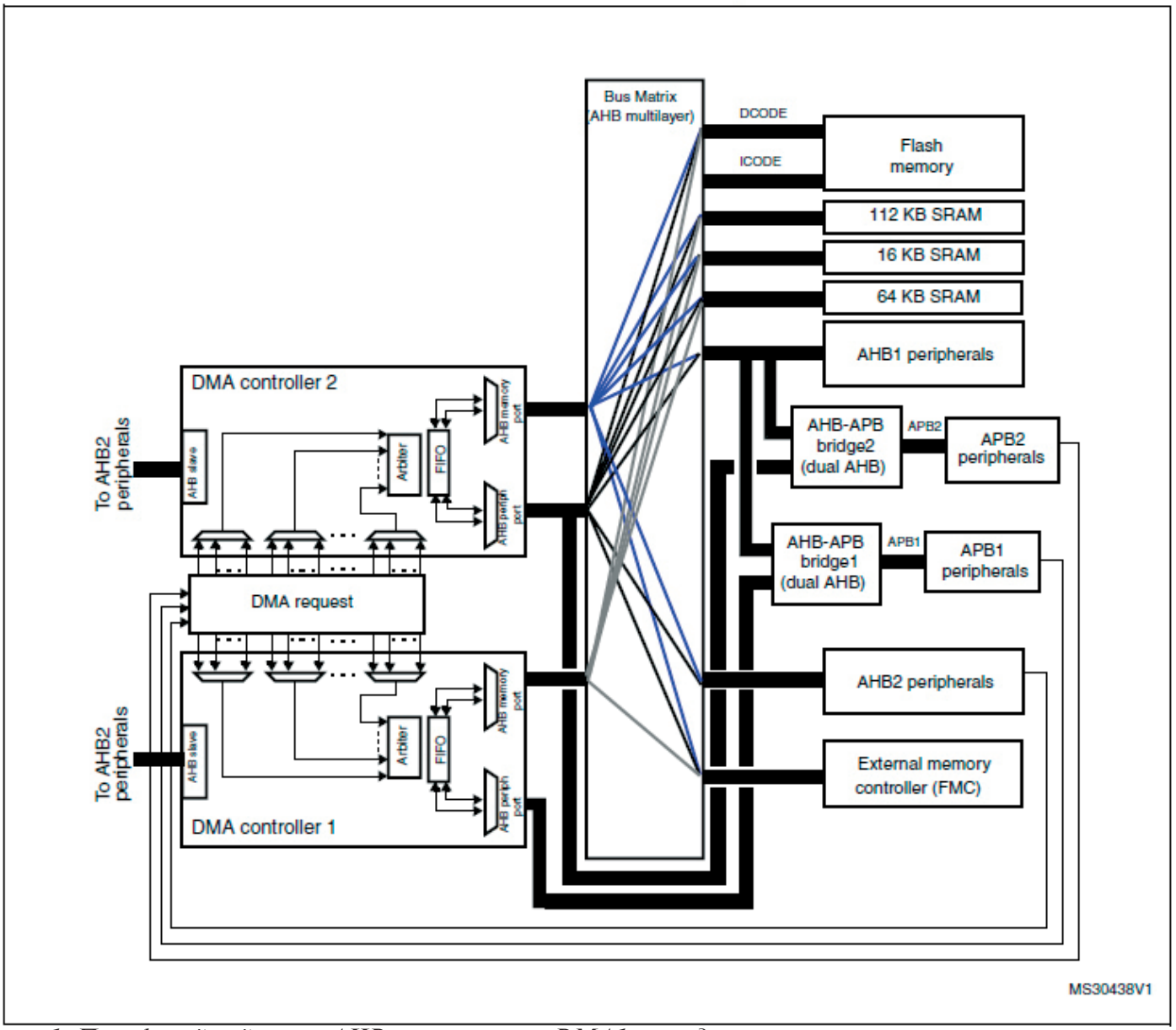
- периферия-память
- память-периферия
- память-в-память

Контроллер DMA предоставляет два основных порта AHB: порт памяти AHB, предназначенный для подключения к памяти и периферийному порту AHB, предназначенный для подключения к периферийным устройствам. Однако для передачи данных между памятью и периферийным портом AHB также должен быть доступ к памяти.

Ведомый порт AHB используется для программирования контроллера DMA (он поддерживает только 32-разрядные обращения).

См. Рис. 33 и рис. 34 для реализации системы из двух контроллеров DMA.

Figure 34. System implementation of the two DMA controllers (STM32F42xxx and STM32F43xxx)



1. Периферийный порт AHB контроллера DMA1 не подключен к матрице шины, в отличие от контроллера DMA2. В результате только потоки DMA2 могут выполнять передачу данных между памятью.

10.3.2. Операции DMA

Операция DMA состоит из последовательности заданного количества передач данных. Количество передаваемых элементов данных и их ширина (8 бит, 16 бит или 32 бит) программируются.

Каждая передача DMA состоит из трех операций:

• Загрузка из периферийного регистра данных или места в памяти, адресованного через регистр DMA_SxPAR или DMA_SxM0AR

• Хранение данных, загружаемых в регистр периферийных данных, или местоположение в памяти, адресованное через регистр DMA_SxPAR или DMA_SxM0AR

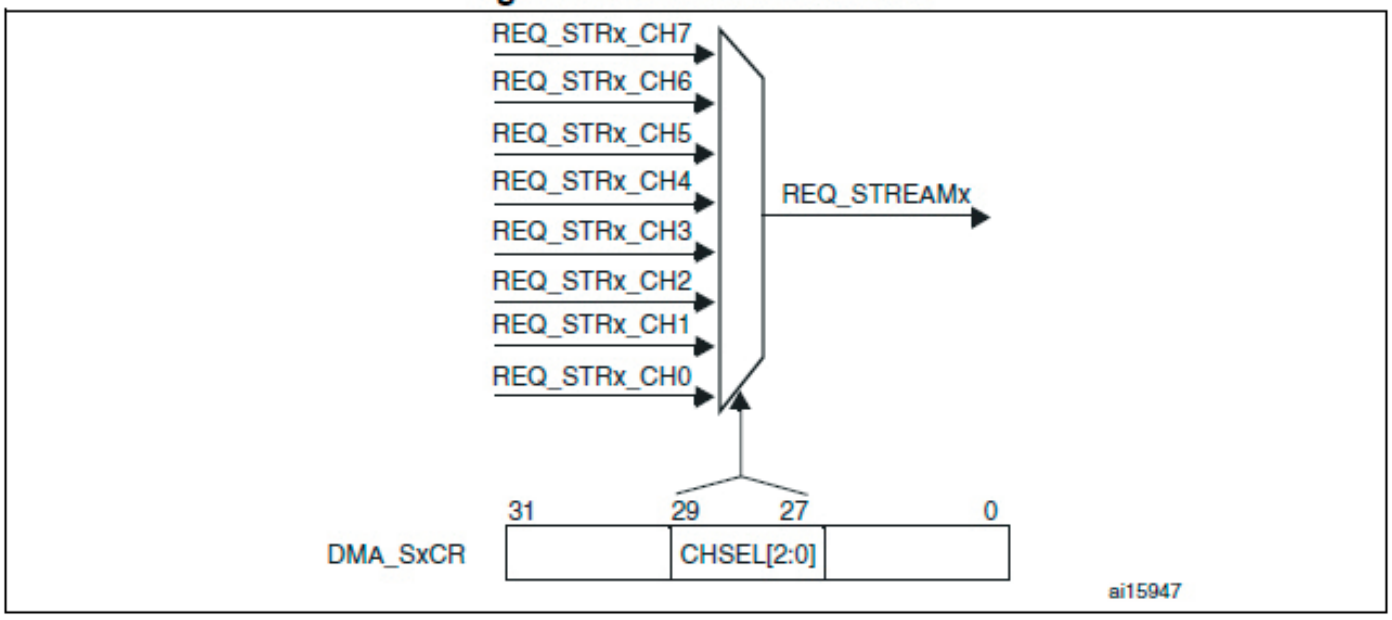
• Пост декремент регистра DMA_SxNDTR, который содержит количество транзакций, которые еще нужно выполнить

После события периферийное устройство отправляет сигнал запроса на контроллер DMA. Контроллер DMA выполняет запрос в зависимости от приоритетов канала. Как только контроллер DMA обращается к периферийному устройству, сигнал подтверждения подтверждается контроллером DMA. Периферийное устройство освобождает свой запрос, как только он получает сигнал подтверждения от контроллера DMA. После того как запрос был отменен периферийным устройством, контроллер DMA освобождает сигнал подтверждения. Если запросов больше, периферийное устройство может инициировать следующую транзакцию.

10.3.3 Выбор канала

Каждый поток связан с запросом DMA, который может быть выбран из 8 возможных запросов канала. Выбор контролируется битами CHSEL [2: 0] в регистре DMA_SxCR.

Figure 35. Channel selection



8 запросов от периферийных устройств (TIM, ADC, SPI, I2C и т. Д.) Независимо подключаются к каждому каналу, и их соединение зависит от реализации продукта.

См. Следующие таблицы (примеры) для сопоставлений запросов DMA.

Table 42. DMA1 request mapping

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	SPI3_RX		SPI3_RX	SPI2_RX	SPI2_TX	SPI3_TX		SPI3_TX
Channel 1	I2C1_RX		TIM7_UP		TIM7_UP	I2C1_RX	I2C1_TX	I2C1_TX
Channel 2	TIM4_CH1		I2S3_EXT_RX	TIM4_CH2	I2S2_EXT_TX	I2S3_EXT_TX	TIM4_UP	TIM4_CH3
Channel 3	I2S3_EXT_RX	TIM2_UP TIM2_CH3	I2C3_RX	I2S2_EXT_RX	I2C3_TX	TIM2_CH1	TIM2_CH2 TIM2_CH4	TIM2_UP TIM2_CH4
Channel 4	UART5_RX	USART3_RX	UART4_RX	USART3_TX	UART4_TX	USART2_RX	USART2_TX	UART5_TX
Channel 5	UART8_TX(1)	UART7_TX(1)	TIM3_CH4 TIM3_UP	UART7_RX(1)	TIM3_CH1 TIM3_TRIG	TIM3_CH2	UART8_RX(1)	TIM3_CH3

Table 42. DMA1 request mapping (continued)

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 6	TIM5_CH3 TIM5_UP	TIM5_CH4 TIM5_TRIG	TIM5_CH1	TIM5_CH4 TIM5_TRIG	TIM5_CH2		TIM5_UP	
Channel 7		TIM6_UP	I2C2_RX	I2C2_RX	USART3_TX	DAC1	DAC2	I2C2_TX

1. These requests are available on STM32F42xxx and STM32F43xxx only.

Table 43. DMA2 request mapping

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	ADC1	SAI1_A(1)	TIM8_CH1 TIM8_CH2 TIM8_CH3	SAI1_A(1)	ADC1	SAI1_B(1)	TIM1_CH1 TIM1_CH2 TIM1_CH3	
Channel 1		DCMI	ADC2	ADC2	SAI1_B(1)	SPI6_TX(1)	SPI6_RX(1)	DCMI
Channel 2	ADC3	ADC3		SPI5_RX(1)	SPI5_TX(1)	CRYP_OUT	CRYP_IN	HASH_IN
Channel 3	SPI1_RX		SPI1_RX	SPI1_TX		SPI1_TX		
Channel 4	SPI4_RX(1)	SPI4_TX(1)	USART1_RX	SDIO		USART1_RX	SDIO	USART1_TX
Channel 5		USART6_RX	USART6_RX	SPI4_RX(1)	SPI4_TX(1)		USART6_TX	USART6_TX
Channel 6	TIM1_TRIG	TIM1_CH1	TIM1_CH2	TIM1_CH1	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	
Channel 7		TIM8_UP	TIM8_CH1	TIM8_CH2	TIM8_CH3	SPI5_RX(1)	SPI5_TX(1)	TIM8_CH4 TIM8_TRIG TIM8_COM

1. Эти запросы доступны на STM32F42xxx и STM32F43xxx.

10.3.4 Арбитр

Арбитр управляет 8 потоковыми запросами DMA на основе их приоритета для каждого из двух главных портов AHB (памяти и периферийных портов) и запускает последовательности доступа к периферии / памяти.

Приоритеты управляются в два этапа:

• Программное обеспечение: каждый приоритет потока может быть настроен в регистре DMA_SxCR. Существует четыре уровня:

- Очень высокий приоритет
- Высокий приоритет
- Средний приоритет
- Низкий приоритет

• Аппаратное обеспечение. Если два запроса имеют одинаковый уровень приоритета программного обеспечения, поток с более низким номером имеет приоритет над потоком с более высоким номером. Например, Stream 2 имеет приоритет над Stream 4.

10.3.5 Потоки DMA

Каждый из 8 потоков контроллера DMA обеспечивает однонаправленную линию передачи между источником и пунктом назначения.

Каждый поток может быть настроен для выполнения:

• Операции с обычным типом: пересылка между памятью и периферией, периферией и памятью или памятью и памятью

• Операции с двойным буфером: передача двух буферов с использованием двух указателей памяти для памяти (в то время как DMA считывает / записывает из / в буфер, приложение может писать / читать в / из другого буфера).

Объем передаваемых данных (до 65535) программируется и связан с шириной источника периферии, которая запрашивает передачу DMA, подключенную к периферийному порту AHB. Регистр, который содержит количество передаваемых элементов данных, уменьшается после каждой транзакции.

10.3.6 Режимы источника, адресата и передачи

Передача как источника, так и адресата может адресовать периферийные устройства и память во всей области 4 ГБ, по адресам, составляющим от 0x00000000 до 0xFFFF FFFF.

Направление сконфигурировано с использованием бит DIR [1: 0] в регистре DMA_SxCR и предлагает три возможности: передачу между периферией и памятью, памятью и периферией или памятью и памятью. В таблице 44 описаны соответствующие адреса источника и получателя.

Table 44. Source and destination address

Bits DIR[1:0] of the DMA_SxCR register	Direction	Source address	Destination address
00	Peripheral-to-memory	DMA_SxPAR	DMA_SxM0AR
01	Memory-to-peripheral	DMA_SxM0AR	DMA_SxPAR
10	Memory-to-memory	DMA_SxPAR	DMA_SxM0AR
11	reserved	-	-

Когда ширина данных (запрограммированная в битах PSIZE или MSIZE в регистре DMA_SxCR) является полусловом или словом, соответственно, адрес периферии или памяти, записанный в регистры DMA_SxPAR или DMA_SxM0AR / M1AR, должен быть выровнен по слову или границы слова полуслова, соответственно.

Режим от периферии к памяти

На рисунке 36 описан этот режим.

Когда этот режим включен (путем установки бит EN в регистре DMA_SxCR), каждый раз, когда возникает запрос периферии, поток инициирует передачу из источника для заполнения FIFO.

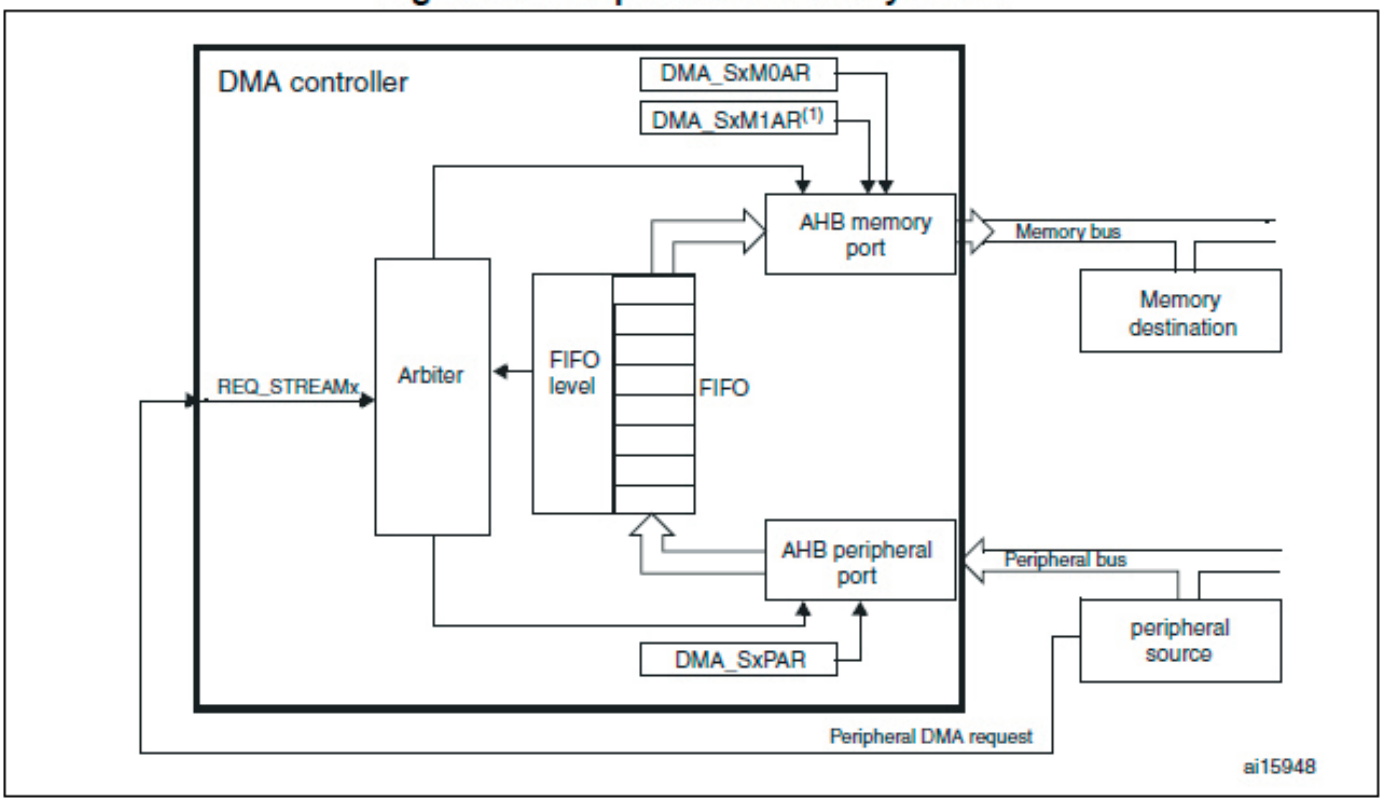
Когда достигается пороговый уровень FIFO, содержимое FIFO сливается и сохраняется в месте назначения.

Передача останавливается, когда регистр DMA_SxNDTR достигает нуля, когда периферийное устройство запрашивает конец передачи (в случае контроллера периферийного потока) или когда бит EN в регистре DMA_SxCR очищается программным обеспечением.

В прямом режиме (когда значение DMDIS в регистре DMA_SxFCR равно «0»), пороговый уровень FIFO не используется: после каждой отдельной передачи данных от периферии к FIFO соответствующие данные немедленно сливаются и сохраняются в место назначения.

Поток имеет доступ к исходному или целевому порту AHB только в случае победы арбитража соответствующего потока. Этот арбитраж выполняется с использованием приоритета, определенного для каждого потока, используя бит PL [1: 0] в регистре DMA_SxCR.

Figure 36. Peripheral-to-memory mode



1. Для режима с двумя буферами.

Режим памяти-периферия

Рисунок 37 описывает этот режим.

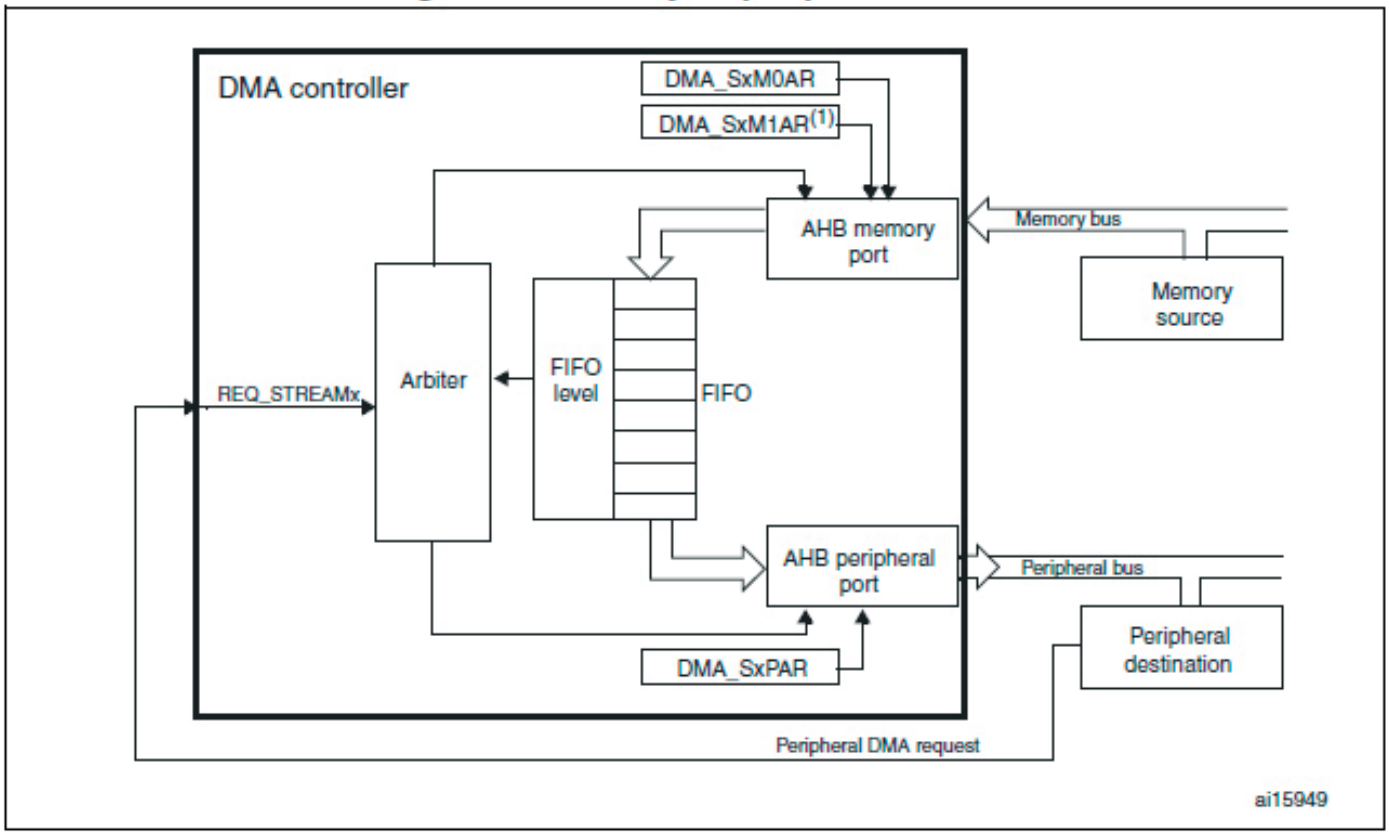
Когда этот режим включен (путем установки бит EN в регистре DMA_SxCR), поток немедленно инициирует передачу из источника, чтобы полностью заполнить FIFO.

Каждый раз, когда возникает периферийный запрос, содержимое FIFO сливается и сохраняется в месте назначения. Когда уровень FIFO ниже или равен предопределенному пороговому уровню, FIFO полностью перезагружается данными из памяти.

Передача останавливается, когда регистр DMA_SxNDTR достигает нуля, когда периферийное устройство запрашивает конец передачи (в случае контроллера периферийного потока) или когда бит EN в регистре DMA_SxCR очищается программным обеспечением.

В прямом режиме (когда значение DMDIS в регистре DMA_SxFCR равно «0»), пороговый уровень FIFO не используется. Как только поток включен, DMA предварительно загружает первые данные для передачи во внутренний FIFO. Как только периферийное устройство запрашивает передачу данных, DMA передает предварительно загруженное значение в сконфигурированное место назначения. Затем он перезагружает снова пустой внутренний FIFO со следующими данными для передачи. Предварительно загруженный размер данных соответствует значению битового поля PSIZE в регистре DMA_SxCR.

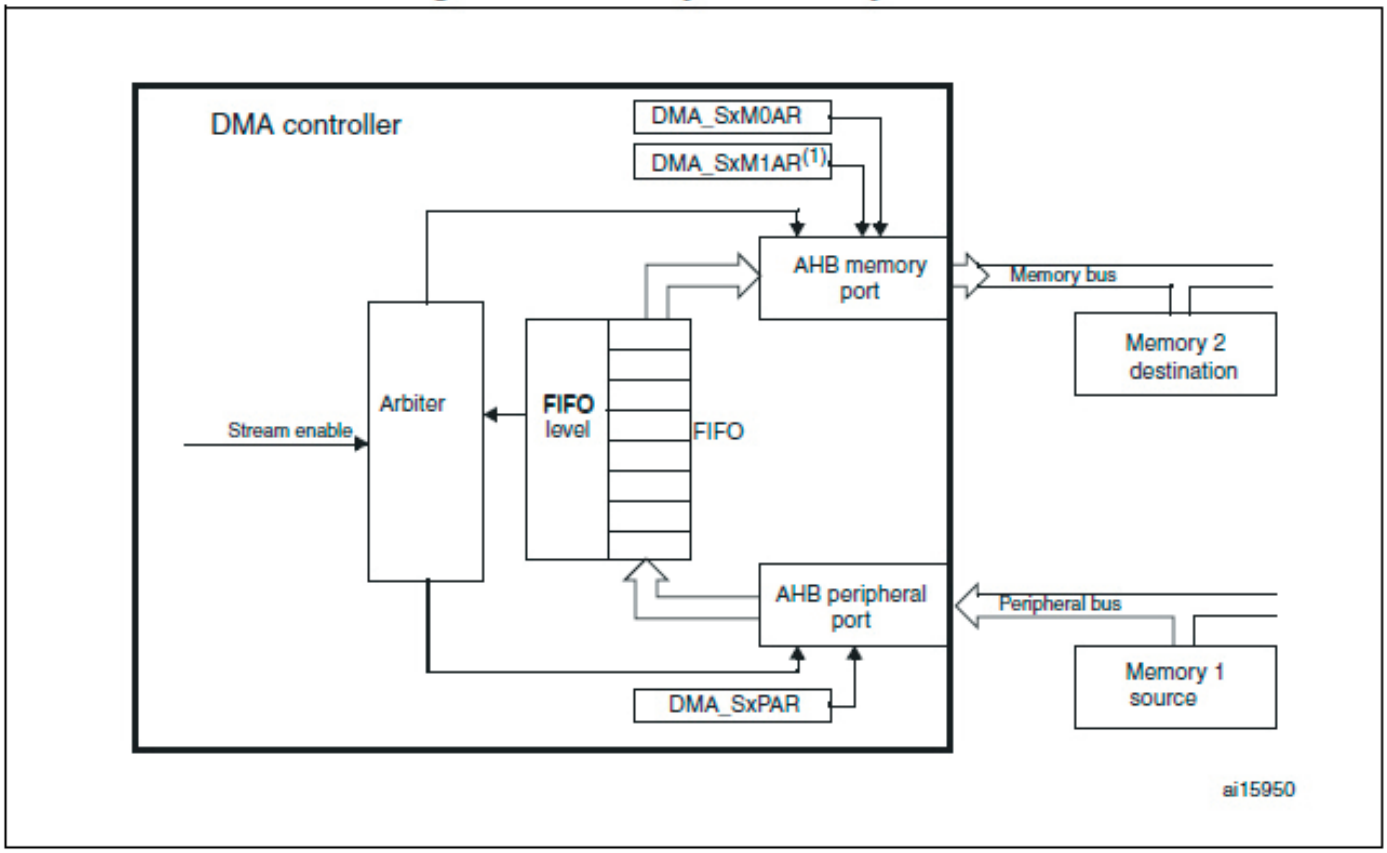
Figure 37. Memory-to-peripheral mode



1. Для режима с двумя буферами.

Поток имеет доступ к исходному или целевому порту AHB только в случае победы арбитража соответствующего потока. Этот арбитраж выполняется с использованием приоритета, определенного для каждого потока, используя бит PL [1: 0] в регистре DMA_SxCR.

Figure 38. Memory-to-memory mode



Режим памяти в память

Каналы DMA также могут работать без запроса по периферии.

Это режим памяти в память, описанный на рисунке 38.

Когда поток активируется установкой бит Enable (EN) в регистр DMA_SxCR, поток немедленно начинает заполнять FIFO до порогового уровня. Когда достигнут пороговый уровень, содержимое FIFO сливается и сохраняется в месте назначения.

Передача останавливается, когда регистр DMA_SxNDTR достигает нуля или когда бит EN в регистре DMA_SxCR очищается программным обеспечением.

Поток имеет доступ к исходному или целевому порту AHB только в случае победы арбитража соответствующего потока. Этот арбитраж выполняется с использованием приоритета, определенного для каждого потока, используя бит PL [1: 0] в регистре DMA_SxCR.

Примечание. *Когда используется режим памяти в память, режимы Circular и direct не допускаются.*

Только контроллер DMA2 способен выполнять передачу данных между памятью.

10.3.7 Инкремент указателя

Указатели периферии и мамяти могут автоматически увеличиваться или сохраняться постоянными после каждой передачи в зависимости от бит PINC и MINC в регистре DMA_SxCR.

Отключение режима инкремента полезно, когда доступ к периферийным источникам или данным назначения осуществляется через единый регистр.

Если режим Приращения включен, адрес следующей передачи будет адресом предыдущего, увеличенным на 1 (для байтов), 2 (для полуслов) или 4 (для слов) в зависимости от ширины данных, запрограммированной в PSIZE или MSIZE в регистре DMA_SxCR.

Чтобы оптимизировать операцию упаковки, можно исправить размер смещения приращения для периферийного адреса независимо от размера данных, переданных на периферийном порту AHB. Бит PINCOS в регистре DMA_SxCR используется для выравнивания размера смещения приращения с размером данных на периферийном AHB-порту или на 32-битном адресе (затем адрес увеличивается на 4). Бит PINCOS влияет только на периферийный порт AHB.

Если бит PINCOS установлен, адрес следующей передачи - это адрес предыдущего, который увеличивается на 4 (автоматически выравнивается по 32-разрядному адресу), независимо от значения PSIZE. Однако на порт AHB не влияет эта операция.

10.3.8 Циклический режим

Режим Circular доступен для обработки циклических буферов и непрерывных потоков данных (например, режим сканирования ADC). Эта функция может быть включена с использованием бита CIRC в регистре DMA_SxCR.

Когда циклический режим активирован, количество передаваемых элементов данных автоматически перезагружается с начальным значением, запрограммированным во время фазы конфигурации потока, и запросы DMA продолжают обслуживаться.

Примечание. В циклическом режиме в случае режима пакетной передачи, настроенного для памяти, необходимо соблюдать следующее правило:

DMA_SxNDTR = Множество из ((Mburst beat) x (Msize) / (Psize)), где:

- (Mburst beat) = 4, 8 или 16 (в зависимости от бит MBURST в регистре DMA_SxCR)
- ((Msize) / (Psize)) = 1, 2, 4, 1/2 или 1/4 (Msize и Psize представляют бит MSIZE и PSIZE в регистре DMA_SxCR, которые зависят от байта)
- DMA_SxNDTR = Количество элементов данных для передачи на периферийном порту AHB

Например: Mburst beat = 8 (INCR8), MSIZE = '00' (байт) и PSIZE = '01' (half-word), в этом случае: DMA_SxNDTR должен быть кратным (8 x 1/2 = 4) ,

Если эта формула не соблюдается, поведение DMA и целостность данных не гарантируются.

NDTR также должен быть кратным размеру периферийного пакета, умноженному на размер периферийных данных, в противном случае это может привести к ухудшению поведения DMA.

10.3.9 Режим двойного буфера

Этот режим доступен для всех потоков DMA1 и DMA2.

Режим Double buffer включен, установив бит DBM в регистре DMA_SxCR.

Двухбуферный поток работает как обычный (один буфер) поток с разницей, которая имеет два указателя памяти. Когда режим Double buffer включен, режим Circular автоматически активируется (бит CIRC в DMA_SxCR не заботится), и на каждом конце транзакции указатели памяти меняются местами.

В этом режиме контроллер DMA меняет местами с одного целевого объекта памяти на каждый конец транзакции. Это позволяет программному обеспечению обрабатывать одну область памяти, а вторая область памяти заполняется / используется передачей DMA. Двухбуферный поток может работать в обоих направлениях (память может быть либо источником, либо пунктом назначения), как описано в таблице 45: регистры адреса источника и получателя в режиме двойного буфера (DBM = 1).

Примечание. В режиме двойного буфера можно обновлять базовый адрес для порта памяти AHB «на лету» (DMA_SxM0AR или DMA_SxM1AR), когда поток включен, соблюдая следующие условия:

- Когда бит CT равен «0» в регистре DMA_SxCR, регистр DMA_SxM1AR может быть записан. Попытка записать в этот регистр, в то время как CT = '1' устанавливает флаг ошибки (TEIF), и поток автоматически отключается.
- Когда бит CT равен «1» в регистре DMA_SxCR, регистр DMA_SxM0AR может быть записан. Попытка записать в этот регистр при CT = '0', устанавливает флаг ошибки (TEIF), и поток автоматически отключается.

Чтобы избежать каких-либо условий ошибки, рекомендуется изменить базовый адрес, как только будет установлен флаг TCIF, поскольку в этот момент целевая память должна быть изменена с 0 до 1 (или от 1 до 0) в зависимости от значения CT в регистре DMA_SxCR в соответствии с одним из двух вышеуказанных условий.

Для всех других режимов (кроме режима Double buffer) регистры адресов памяти защищены от записи, как только поток включен.

Table 45. Source and destination address registers in Double buffer mode (DBM=1)

Bits DIR[1:0] of the DMA_SxCR register	Direction	Source address	Destination address
00	Peripheral-to-memory	DMA_SxPAR	DMA_SxM0AR / DMA_SxM1AR
01	Memory-to-peripheral	DMA_SxM0AR / DMA_SxM1AR	DMA_SxPAR
10	Not allowed(1)		
11	Reserved	-	-

1. Когда включен режим Double buffer, режим Circular автоматически включается. Поскольку режим памяти в память не совместим с режимом Circular, когда включен режим Double buffer, не разрешается конфигурировать режим памяти в память.

10.3.10. Программируемая ширина данных, упаковка / распаковка, endianess

*Порядок следования байт (***endianness***). Соответственно бывает 2 варианта порядка байт - big endian и little endian. Big endian это такой порядок байт, когда самый значимый по значению байт числа (most significant byte) сохранен в памяти первым по порядку (т. е. у него самый маленький абсолютный адрес байта, в сравнении с остальными байтами числа). Соответственно little endian это противоположный порядок байт, когда наименее значимый байт сохраняется в памяти первым.*

Количество передаваемых элементов данных должно быть запрограммировано в DMA_SxNDTR (количество элементов данных для передачи бит, NDT) перед включением потока (кроме случаев, когда контроллер потока является периферийным, установлен бит PFCTRL в DMA_SxCR).

При использовании внутреннего FIFO ширина данных источника и данных назначения программируется через биты PSIZE и MSIZE в регистре DMA_SxCR (может быть 8-, 16- или 32-разрядная).

Когда PSIZE и MSIZE не равны:

• Ширина данных количества передаваемых элементов данных, настроенных в регистре DMA_SxNDTR, равна ширине периферийной шины (с помощью битов PSIZE в регистре DMA_SxCR). Например, в случае пересылки от периферии к памяти, из памяти в периферию или из памяти в память, и если биты PSIZE [1: 0] настроены для полуслова, количество передаваемых байтов равно 2 x NDT.

• Контроллер DMA управляется только с минимальной адресацией для источника и адресата. Это описано в таблице 46: Упаковка / распаковка и поведение endian (бит PINC = MINC = 1).

Эта процедура упаковки / распаковки может представлять риск повреждения данных, когда операция прерывается, прежде чем данные будут полностью упакованы / распакованы. Таким образом, для обеспечения согласованности данных поток может быть сконфигурирован для генерации пакетных передач: в этом случае каждая группа передач, принадлежащих пакету, является неделимой (см. Раздел 10.3.11: Одиночная и пакетная передача).

В прямом режиме (DMDIS = 0 в регистре DMA_SxFCR) упаковка / распаковка данных невозможна. В этом случае не разрешается иметь разные ширины передачи данных источника и назначения: оба равны и определяются битами PSIZE. В битах DMS_SxCR MSIZE об этом не заботятся).

Table 46. Packing/unpacking & endian behavior (bit PINC = MINC = 1)

AHB memory port width	AHB peripheral port width	Number of data items to transfer (NDT)	Memory transfer number	Memory port address / byte lane	Peripheral transfer number	Peripheral port address / byte lane	
						PINCOS = 1	PINCOS = 0
8	8	4	1 2 3 4	0x0 / B0[7:0] 0x1 / B1[7:0] 0x2 / B2[7:0] 0x3 / B3[7:0]	1 2 3 4	0x0 / B0[7:0] 0x4 / B1[7:0] 0x8 / B2[7:0] 0xC / B3[7:0]	0x0 / B0[7:0] 0x1 / B1[7:0] 0x2 / B2[7:0] 0x3 / B3[7:0]
8	16	2	1 2 3 4	0x0 / B0[7:0] 0x1 / B1[7:0] 0x2 / B2[7:0] 0x3 / B3[7:0]	1 2	0x0 / B1|B0[15:0] 0x4 / B3|B2[15:0]	0x0 / B1|B0[15:0] 0x2 / B3|B2[15:0]
8	32	1	1 2 3 4	0x0 / B0[7:0] 0x1 / B1[7:0] 0x2 / B2[7:0] 0x3 / B3[7:0]	1	0x0 / B3|B2|B1|B0[31:0]	0x0 / B3|B2|B1|B0[31:0]
16	8	4	1 2	0x0 / B1|B0[15:0] 0x2 / B3|B2[15:0]	1 2 3 4	0x0 / B0[7:0] 0x4 / B1[7:0] 0x8 / B2[7:0] 0xC / B3[7:0]	0x0 / B0[7:0] 0x1 / B1[7:0] 0x2 / B2[7:0] 0x3 / B3[7:0]
16	16	2	1 2	0x0 / B1|B0[15:0] 0x2 / B1|B0[15:0]	1 2	0x0 / B1|B0[15:0] 0x4 / B3|B2[15:0]	0x0 / B1|B0[15:0] 0x2 / B3|B2[15:0]
16	32	1	1 2	0x0 / B1|B0[15:0] 0x2 / B3|B2[15:0]	1	0x0 / B3|B2|B1|B0[31:0]	0x0 / B3|B2|B1|B0[31:0]
32	8	4	1	0x0 / B3|B2|B1|B0[31:0]	1 2 3 4	0x0 / B0[7:0] 0x4 / B1[7:0] 0x8 / B2[7:0] 0xC / B3[7:0]	0x0 / B0[7:0] 0x1 / B1[7:0] 0x2 / B2[7:0] 0x3 / B3[7:0]
32	16	2	1	0x0 /B3|B2|B1|B0[31:0]	1 2	0x0 / B1|B0[15:0] 0x4 / B3|B2[15:0]	0x0 / B1|B0[15:0] 0x2 / B3|B2[15:0]
32	32	1	1	0x0 /B3|B2|B1|B0 [31:0]	1	0x0 /B3|B2|B1|B0 [31:0]	0x0 / B3|B2|B1|B0[31:0]

Примечание. *Периферийный порт может быть источником или пунктом назначения (он также может быть источником памяти в случае передачи памяти в память).*

PSIZE, MSIZE и NDT [15: 0] должны быть настроены таким образом, чтобы последняя передача не была неполной. Это может произойти, когда ширина данных периферийного порта (биты PSIZE) ниже, чем ширина данных порта памяти (бит MSIZE). Это ограничение суммировано в таблице 47.

10.3.11 Одиночные и пакетные передачи

Контроллер DMA может генерировать одиночные передачи или инкрементную передачу пакетов 4, 8 или 16 бит.

Table 47. Restriction on NDT versus PSIZE and MSIZE

PSIZE[1:0] of DMA_SxCR	MSIZE[1:0] of DMA_SxCR	NDT[15:0] of DMA_SxNDTR
00 (8-bit)	01 (16-bit)	must be a multiple of 2
00 (8-bit)	10 (32-bit)	must be a multiple of 4
01 (16-bit)	10 (32-bit)	must be a multiple of 2

Размер пакета настраивается программным обеспечением независимо для двух портов AHB с использованием бит MBURST [1: 0] и PBURST [1: 0] в регистре DMA_SxCR.

Размер пакета указывает количество ударов в пакете, а не количество переданных байтов.

Для обеспечения согласованности данных каждая группа передач, образующих пакет, неделима: передача AHB заблокирована, и арбитр матрицы шины AHB не ухудшает мастер DMA во время последовательности пакетной передачи.

В зависимости от одиночной или пакетной конфигурации каждый запрос DMA инициирует различное количество передач на периферийном порту AHB:

• Когда периферийный порт AHB настроен для одиночной передачи, каждый запрос DMA генерирует передачу данных байта, полуслова или слова в зависимости от битов PSIZE [1: 0] в регистре DMA_SxCR

• Если периферийный порт AHB настроен для пакетных передач, каждый запрос DMA генерирует 4,8 или 16 бит байта, половины слова или передачи слов в зависимости от бит PBURST [1: 0] и PSIZE [1: 0] в DMA_SxCR регистр.

То же, что и выше, необходимо учитывать для порта памяти AHB с учетом бит MBURST и MSIZE.

В прямом режиме поток может генерировать только отдельные передачи, а биты MBURST [1: 0] и PBURST [1: 0] принудительно используются аппаратными средствами.

Указатели адресов (регистры DMA_SxPAR или DMA_SxM0AR) должны быть выбраны так, чтобы обеспечить, чтобы все передачи в пакетном блоке были выровнены по границе адреса, равным размеру передачи.

Конфигурация пакета должна выбираться для того, чтобы соблюдать протокол AHB, где пакеты не должны пересекать границу адреса 1 КБ, поскольку минимальное адресное пространство, которое может быть выделено одному подчиненному устройству, составляет 1 КБ. Это означает, что граница адреса 1 КБ не должна пересекаться посредством пакетной передачи блока, в противном случае будет генерироваться ошибка AHB, которая не регистрируется регистрами DMA.

10.3.12 FIFO

Структура FIFO

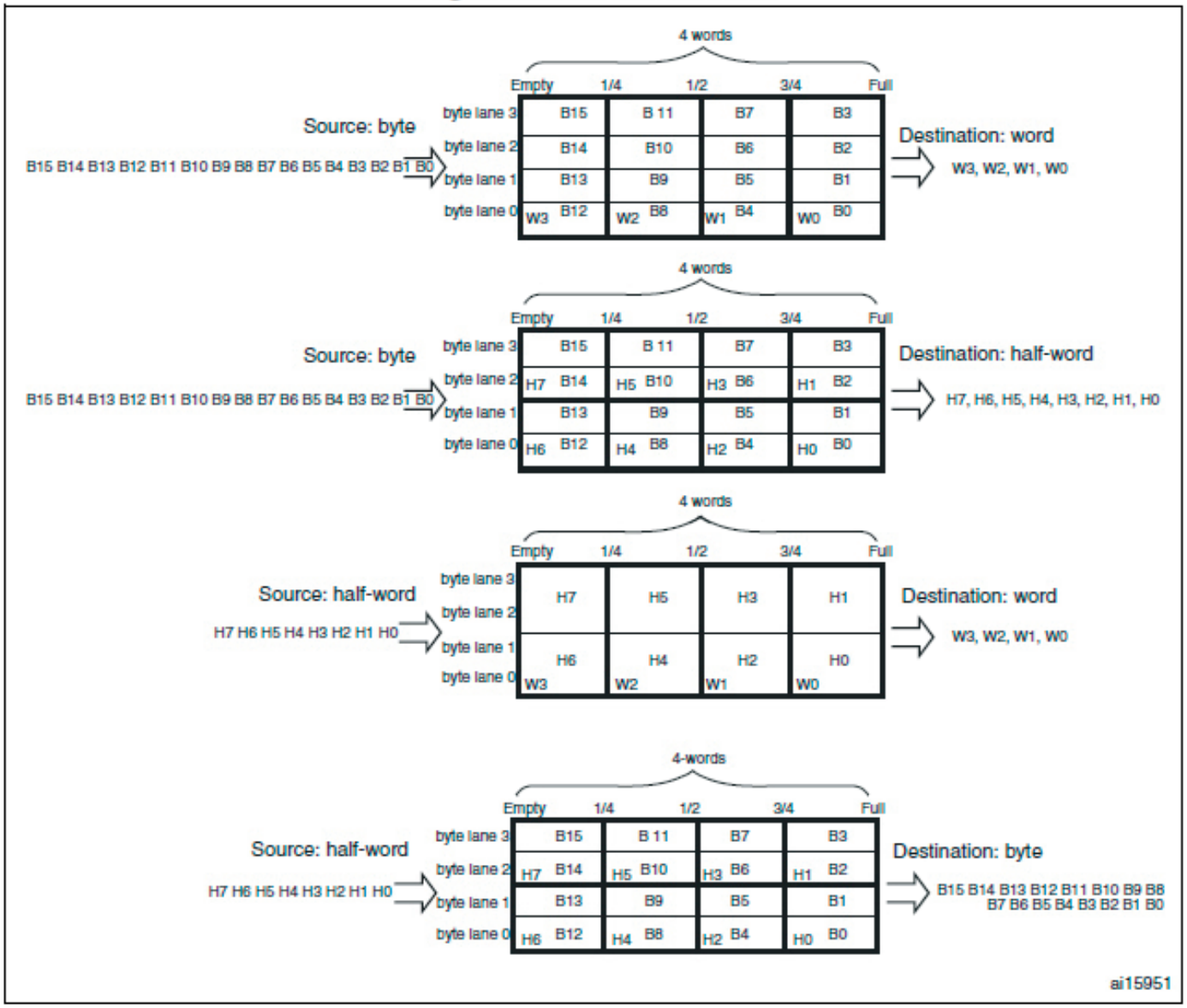
FIFO используется для временного хранения данных, поступающих от источника, прежде чем передавать их в пункт назначения.

Каждый поток имеет независимый 4-словный FIFO и пороговый уровень, настраиваемый программным обеспечением, конфигурируемое между 1/4, 1/2, 3/4 или полным.

Чтобы включить использование порогового уровня FIFO, прямой режим необходимо отключить, установив бит DMDIS в регистр DMA_SxFCR.

Структура FIFO отличается в зависимости от ширины данных источника и назначения и описана на рисунке 39: Структура FIFO.

Figure 39. FIFO structure



Порог FIFO и burst configuration (пакетная конфигурация)

Необходимо соблюдать осторожность при выборе порога FIFO (бит FTH [1: 0] регистра DMA_SxFCR) и размера пакета памяти (MBURST [1: 0] регистра DMA_SxCR): содержимое, указанное порогом FIFO, должно быть точно совпадают с целым числом пакетов пакетов памяти. Если этого не происходит, при включении потока будет генерироваться ошибка FIFO (флаг FEIFx регистра DMA_HISR или DMA_LISR), после чего поток будет автоматически отключен. Разрешенные и запрещенные конфигурации описаны в таблице 48 (Конфигурации порога FIFO)

Во всех случаях размер пакета, умноженный на размер данных, не должен превышать размер FIFO (размер данных может быть: 1 (байт), 2 (полуслово) или 4 (слово)).

Передача неполного пакета в конце передачи DMA может произойти, если происходит одно из следующих условий:

Table 48. FIFO threshold configurations

MSIZE	FIFO level	MBURST = INCR4	MBURST = INCR8	MBURST = INCR16
Byte	1/4	1 burst of 4 beats	forbidden	forbidden
	1/2	2 bursts of 4 beats	1 burst of 8 beats	
	3/4	3 bursts of 4 beats	forbidden	
	Full	4 bursts of 4 beats	2 bursts of 8 beats	1 burst of 16 beats
Half-word	1/4	forbidden	forbidden	forbidden
	1/2	1 burst of 4 beats		
	3/4	forbidden		
	Full	2 bursts of 4 beats	1 burst of 8 beats	
Word	1/4	forbidden	forbidden	
	1/2			
	3/4			
	Full	1 burst of 4 beats		

• Для конфигурации периферийного порта AHB общее количество элементов данных (установлено в регистре DMA_SxNDTR) не кратно размеру пакета, умноженному на размер данных

• Для конфигурации порта памяти AHB: количество оставшихся элементов данных в FIFO, которые должны быть переданы в память, не кратно размеру пакета, умноженному на размер данных

В таких случаях оставшиеся данные, подлежащие передаче, будут управляться в одном режиме с помощью DMA, даже если во время конфигурации потока DMA была запрошена пакетная транзакция.

Примечание. *Если на периферийном AHB-порту запрашиваются пакетные передачи и используется FIFO (DMDIS = 1 в регистре DMA_SxCR), необходимо соблюдать следующее правило, чтобы избежать постоянных условий переполнения или перерасхода, в зависимости от направления потока DMA:*

Если (PBURST x PSIZE) = FIFO_SIZE (4 слова), FIFO_Threshold = 3/4 запрещается с PSIZE = 1, 2 или 4 и PBURST = 4, 8 или 16.

Это правило гарантирует, что достаточное пространство FIFO в то же время будет свободно обслуживать запрос от периферии.

FIFO flush (промывка)

FIFO может быть сброшен, когда поток отключен путем сброса бит EN в регистре DMA_SxCR и когда поток настроен для управления пересылкой между периферией и памятью или памятью в память: если некоторые данные все еще присутствуют в FIFO, когда поток отключен, контроллер DMA продолжает передачу оставшихся данных в пункт назначения (даже если поток эффективно отключен). По завершении этого сброса устанавливается бит состояния завершения передачи (TCIFx) в регистре DMA_LISR или DMA_HISR.

Оставшийся счетчик данных DMA_SxNDTR сохраняет значение в этом случае, чтобы указать, сколько элементов данных доступно в настоящее время в целевой памяти.

Обратите внимание, что во время операции сброса FIFO, если количество оставшихся элементов данных в FIFO, которое должно быть передано в память (в байтах), меньше ширины данных памяти (например, 2 байта в FIFO, в то время как MSIZE настроено на слово), данные будет отправляться с шириной данных, установленной в бит MSIZE в регистре DMA_SxCR. Это означает, что память будет записана с нежелательным значением. Программное обеспечение может считывать регистр DMA_SxNDTR для определения области памяти, содержащей хорошие данные (начальный адрес и последний адрес).

Если количество оставшихся элементов данных в FIFO ниже размера пакета (если биты MBURST в регистре DMA_SxCR настроены на настройку потока для управления пакетом на порте памяти AHB), для завершения FIFO-флеша будут созданы отдельные транзакции.

Прямой режим

По умолчанию FIFO работает в прямом режиме (бит DMDIS в DMA_SxFCR сбрасывается), а пороговый уровень FIFO не используется. Этот режим полезен, когда система требует немедленной и одиночной передачи в память или из нее после каждого запроса DMA.

Когда DMA сконфигурирован в прямом режиме (FIFO disabled), чтобы передавать данные в режиме памяти вперёд, DMA предварительно загружает одну информацию из памяти во внутренний FIFO, чтобы обеспечить немедленную передачу данных, как только запрос DMA будет вызван периферийное устройство.

Чтобы избежать насыщения FIFO, рекомендуется настроить соответствующий поток с высоким приоритетом. Этот режим ограничивается передачами, где:

• Ширины передачи источника и назначения равны, и оба бита PSIZE [1: 0] в битах DMA_SxCR (MSIZE [1: 0] не заботятся)

• Передача пакета невозможна (PBURST [1: 0] и MBURST [1: 0] бит в DMA_SxCR не заботятся)

Прямой режим не должен использоваться при передаче памяти в память.

10.3.13 Завершение передачи DMA

Различные события могут генерировать конец передачи, установив бит TCIFx в регистр состояния DMA_LISR или DMA_HISR:

• В режиме контроллера потока DMA:

- Счетчик DMA_SxNDTR достиг нулевого значения в режиме памяти-к-периферии

- Поток отключен до окончания передачи (путем сброса бит EN в регистре DMA_SxCR) и (когда передачи между периферией и памяти или из памяти в память) все остальные данные были сброшены из FIFO в Память

• В режиме контроллера периферийного потока:

- Последний внешний пакет или один запрос был сгенерирован из периферийного устройства (и когда DMA работает в режиме «от периферии к памяти») оставшиеся данные были перенесены из FIFO в память

- поток отключен программным обеспечением, и (когда DMA работает в режиме от периферии к памяти) оставшиеся данные были перенесены из FIFO в память

Примечание. *Завершение передачи зависит от оставшихся данных в FIFO, которые должны быть переданы в память только в случае режима от периферии к памяти. Это условие не применимо в режиме от памяти-к-периферии.*

Если поток сконфигурирован в нециркулярном режиме, то после окончания передачи (то есть, когда количество передаваемых данных достигает нуля), DMA останавливается (бит EN в регистре DMA_SxCR очищается аппаратным обеспечением), и никакой запрос DMA не появляется пока программное обеспечение не перепрограммирует поток и не включит его (установив бит EN в регистре DMA_SxCR).

10.3.14 Подвеска передачи DMA

В любой момент передача DMA может быть приостановлена, чтобы перезапустить ее позже или окончательно отключиться до окончания передачи DMA.

Есть два случая:

• Поток отключает передачу без повторного перезапуска с того места, где он был остановлен. Нет особого действия, кроме как очистить бит EN в регистре DMA_SxCR, чтобы отключить поток. Поток может занять время, которое необходимо отключить (текущая передача завершается первой). Флаг завершения передачи (TCIF в регистре DMA_LISR или DMA_HISR) установлен для указания конца передачи. Значение бит EN в DMA_SxCR теперь «0», чтобы подтвердить прерывание потока. Регистр DMA_SxNDTR содержит количество оставшихся элементов данных в момент остановки потока, чтобы программное обеспечение могло определить, сколько элементов данных было передано до того, как поток был прерван.

• Поток приостанавливает передачу до того, как количество оставшихся элементов данных, которые будут переданы в регистре DMA_SxNDTR, достигнет 0. Цель состоит в том, чтобы перезапустить передачу позже путем повторного включения потока. Чтобы возобновить работу с момента остановки передачи, программное обеспечение должно прочитать регистр DMA_SxNDTR после отключения потока, записав бит EN в регистре DMA_SxCR (а затем проверив, что он равен «0»), чтобы узнать количество элементы данных уже собраны. Затем:

- Периферийные и / или адреса памяти должны быть обновлены, чтобы настроить адресные указатели

- Регистр SxNDTR должен быть обновлен с оставшимся количеством передаваемых элементов данных (значение, считанное при отключении потока)

- Затем поток может быть снова включен для перезапуска передачи с точки, в которой он был остановлен

Примечание. *Обратите внимание, что флаг полного завершения передачи (TCIF в DMA_LISR или DMA_HISR) установлен для указания конца передачи из-за прерывания потока.*

10.3.15 Контроллер потока (Регулятор расхода))))

Объект, который контролирует количество передаваемых данных, известен как контроллер потока. Этот контроллер потока настраивается независимо для каждого потока, используя бит PFCTRL в регистре DMA_SxCR.

Контроллер потока может быть:

• Контроллер DMA: в этом случае количество передаваемых элементов данных программируется программным обеспечением в регистр DMA_SxNDTR до того, как поток DMA будет включен.

• Периферийный источник или получатель: это тот случай, когда количество передаваемых элементов данных неизвестно. При передаче последних данных периферийное устройство указывает аппаратными средствами контроллер DMA. Эта функция поддерживается только для периферийных устройств, которые могут сигнализировать о завершении передачи, то есть:

- SDIO

Когда периферийный контроллер потока используется для данного потока, значение, записанное в DMA_SxNDTR, не влияет на передачу DMA. Фактически, независимо от того, какое значение записано, оно будет принудительно с помощью аппаратного обеспечения до 0xFFFF, как только поток будет включен, чтобы соблюдать следующие схемы:

• Ожидаемое прерывание потока: бит EN в регистре DMA_SxCR сбрасывается на 0 программным обеспечением, чтобы остановить поток до того, как периферийное устройство отправит последний аппаратный сигнал данных (одиночный или пакетный). В таком случае поток отключается, и сбой FIFO запускается в случае передачи DMA с периферийной памяти. Флаг TCIFx соответствующего потока устанавливается в регистре состояния для указания завершения DMA. Чтобы узнать количество элементов данных, переданных во время передачи DMA, прочитайте регистр DMA_SxNDTR и примените следующую формулу:

- Number_of_data_transferred = 0xFFFF - DMA_SxNDTR

• Обычное прерывание потока из-за приема последнего аппаратного сигнала данных: поток автоматически прерывается, когда периферийное устройство запрашивает последнюю передачу (одиночную или пакетную) и когда эта передача завершена. флаг TCIFx соответствующего потока устанавливается в регистре состояния для указания завершения передачи DMA. Чтобы узнать количество переданных элементов данных, прочитайте регистр DMA_SxNDTR и примените ту же формулу, что и выше.

• Регистр DMA_SxNDTR достигает 0: флаг TCIFx соответствующего потока устанавливается в регистре состояния для указания завершения принудительной передачи DMA. Поток автоматически отключается, даже если последний аппаратный сигнал данных (одиночный или пакетный) еще не был подтвержден. Уже перенесенные данные не будут потеряны. Это означает, что максимум 65535 элементов данных может управляться DMA в одной транзакции, даже в режиме управления периферийным потоком.

Примечание. *При настройке в режиме памяти в память DMA всегда является контроллером потока, а бит PFCTRL принудительно равен 0 аппаратным обеспечением.*

Режим Circular запрещен в режиме контроллера периферийного потока.

10.3.16 Краткое описание возможных конфигураций DMA

Table 49. Possible DMA configurations

DMA transfer mode	Source	Destination	Flow controller	Circular mode	Transfer type	Direct mode	Double buffer mode
Peripheral-to-memory	AHB peripheral port	AHB memory port	DMA	possible	single	possible	possible
					burst	forbidden	
			Peripheral	forbidden	single	possible	forbidden
					burst	forbidden	
Memory-to-peripheral	AHB memory port	AHB peripheral port	DMA	possible	single	possible	possible
					burst	forbidden	
			Peripheral	forbidden	single	possible	forbidden
					burst	forbidden	
Memory-to-memory	AHB peripheral port	AHB memory port	DMA only	forbidden	single	forbidden	forbidden
					burst		

В таблице 49 приведены различные возможные конфигурации DMA.

10.3.17 Процедура настройки потока

Следующая последовательность должна следовать для настройки потока DMA x (где x - номер потока):

1. Если поток включен, отключите его, сбросив бит EN в регистр DMA_SxCR, затем прочитайте этот бит, чтобы подтвердить, что нет текущей операции потока. Запись этого бита в 0 не сразу эффективна, так как она фактически записывается в 0 после завершения всех текущих передач. Когда бит EN считывается как 0, это означает, что поток готов к настройке. Поэтому необходимо дождаться, когда бит EN будет очищен до начала любой конфигурации потока. Все потоковые выделенные биты, установленные в регистре состояния (DMA_LISR и DMA_HISR) из предыдущего переноса DMA блока данных, должны быть очищены до повторного включения потока.

2. Задайте адрес регистра периферийного порта в регистре DMA_SxPAR. Данные будут перемещены с / на этот адрес в / из периферийного порта после периферийного события.

3. Задайте адрес памяти в регистре DMA_SxMA0R (и в регистре DMA_SxMA1R в случае режима двойного буфера). Данные будут записаны или прочитаны из этой памяти после периферийного события.

4. Настройте общее количество элементов данных, которые будут переданы в регистре DMA_SxNDTR. После каждого периферийного события или каждого бита пакета это значение уменьшается.

5. Выберите канал DMA (запрос) с помощью CHSEL [2: 0] в регистре DMA_SxCR.

6. Если периферийное устройство предназначено для управления потоком, и если оно поддерживает эту функцию, установите бит PFCTRL в регистр DMA_SxCR.

7. Настройте приоритет потока, используя бит PL [1: 0] в регистре DMA_SxCR.

8. Настройте использование FIFO (включение или отключение, порог при передаче и приеме)

9. Настройте направление передачи, режим периферии и памяти (инкрементный / фиксированный), одиночные или пакетные транзакции, ширину данных периферийных устройств и памяти, циклический режим, режим двойного буфера и прерывания после половины и / или полной передачи и / или ошибки в DMA_SxCR регистр.

10. Активируйте поток, установив бит EN в регистр DMA_SxCR.

Как только поток включен, он может обслуживать любой запрос DMA от периферийного устройства, подключенного к потоку.

После того, как половина данных была перенесена на порт назначения AHB, установлен флаг передачи передачи (HTIF), и прерывание генерируется, если установлен бит разрешения прерывания передачи (HTIE). В конце передачи устанавливается флаг завершения передачи (TCIF), и прерывание генерируется, если установлен бит разрешения завершения передачи (TCIE).

****

Предупреждение. Чтобы отключить периферийное устройство, подключенное к запросу потока DMA, необходимо, во-первых, отключить поток DMA, к которому подключено периферийное устройство, а затем дождаться бит EN = 0.

Только тогда периферийное устройство может быть безопасно отключено.

****

10.3.18 Управление ошибками

Контроллер DMA может обнаруживать следующие ошибки:

• Ошибка передачи: флаг прерывания ошибки передачи (TEIFx) устанавливается, когда:

- Ошибка шины возникает во время чтения DMA или доступа к записи

- Доступ к записи запрашивается программным обеспечением в регистре адресов памяти в режиме двойного буфера, тогда как поток включен, и текущая целевая память - это та, на которую влияет запись в регистр адресов памяти (см. Раздел 10.3.9: Режим двойного буфера)

• Ошибка FIFO: флаг прерывания FIFO (FEIFx) устанавливается, если:

- Обнаружено условие переполнения FIFO

- Обнаружено условие переполнения FIFO (нет обнаружения в режиме памяти в память, поскольку запросы и передачи внутренне управляются DMA)

- Поток активирован, пока пороговый уровень FIFO не совместим с размером пакета памяти (см. Таблицу 48: Конфигурации порога FIFO)

• Ошибка прямого режима: флаг прерывания ошибки прямого режима (DMEIFx) может быть установлен только в режиме «периферийно-оперативная память» при работе в прямом режиме и когда бит MINC в регистре DMA_SxCR очищается. Этот флаг устанавливается, когда выполняется запрос DMA, пока предыдущие данные еще не были полностью перенесены в память (поскольку шина памяти не была предоставлена). В этом случае флаг указывает, что два элемента данных были последовательно перенесены на один и тот же адрес назначения, что может быть проблемой, если адресат не может управлять этой ситуацией. В прямом режиме флаг ошибки FIFO также можно установить под следующие условия:

• В режиме периферийной памяти FIFO может быть насыщенным (переполнением), если шина памяти не предоставляется для нескольких периферийных запросов

• В режиме памяти-к-периферии может возникнуть недопустимое условие, если шина памяти не была предоставлена до того, как возникнет запрос периферии

Если флаг TEIFx или FEIFx установлен из-за несовместимости между размером пакета и пороговым уровнем FIFO, неисправный поток автоматически отключается через аппаратное обеспечение, очищающее его бит EN в соответствующем регистре конфигурации потока (DMA_SxCR).

Если флаг DMEIFx или FEIFx установлен из-за переполнения или недогрузки, неисправный поток автоматически не отключается, и программное обеспечение отключает или не передает поток, перезагружая бит EN в регистре DMA_SxCR. Это связано с тем, что при возникновении такого рода ошибок нет потери данных.

Когда установлен флаг прерывания ошибки потока (TEIF, FEIF, DMEIF) в регистре DMA_LISR или DMA_HISR, прерывание генерируется, если установлен соответствующий бит разрешения прерывания (TEIE FEIE, DMIE) в регистре DMA_SxCR или DMA_SxFCR.

Примечание. *При возникновении переполнения или недогрузки FIFO данные не теряются, так как запрос периферии не подтверждается потоком до тех пор, пока не будет сброшено условие перерасхода или недогрузки. Если это подтверждение занимает слишком много времени, периферийное устройство может обнаружить переполнение или недоработку своего внутреннего буфера, и данные могут быть потеряны.*

10.4 прерывания DMA

Для каждого потока DMA прерывание может быть создано на следующих событиях:

• Достигнута половина буфера
• Передача завершена
• Ошибка передачи
• Ошибка Fifo (ошибка переполнения, недогрузки или FIFO)
• Ошибка прямого режима

Для гибкости доступны отдельные биты управления разрешения прерывания, как показано в таблице 50.

Table 50. DMA interrupt requests

Interrupt event	Event flag	Enable control bit
Half-transfer	HTIF	HTIE
Transfer complete	TCIF	TCIE
Transfer error	TEIF	TEIE
FIFO overrun/underrun	FEIF	FEIE
Direct mode error	DMEIF	DMEIE

Примечание: *перед тем, как установить бит «Включить» в «1», соответствующий флаг события должен быть очищен, иначе прерывание будет немедленно сгенерировано.*

20 Универсальный драйвер HAL DMA

20.1 Драйвер прошивки DMA регистрирует структуры

20.1.1 DMA_InitTypeDef

Поля данных

- uint32_t Channel
- uint32_t Направление
- uint32_t PeriphInc
- uint32_t MemInc
- uint32_t PeriphDataAlignment
- uint32_t MemDataAlignment
- uint32_t Mode
- uint32_t Priority
- uint32_t FIFOMode
- uint32_t FIFOThreshold
- uint32_t MemBurst
- uint32_t PeriphBurst

Назначение полей

- **uint32_t DMA_InitTypeDef :: Channel**

Указывает канал, используемый для указанного потока. Этот параметр может быть значением DMA_Channel_selection

- **uint32_t DMA_InitTypeDef :: Direction**

Указывает, будут ли данные передаваться из памяти в периферию, из памяти в память или из периферии в память. Этот параметр может быть значением DMA_Data_transfer_direction

- **uint32_t DMA_InitTypeDef :: PeriphInc**

Указывает, должен ли регистр периферийного адреса увеличиваться или нет. Этот параметр может быть значением DMA_Peripheral_incremented_mode

- **uint32_t DMA_InitTypeDef :: MemInc**

Указывает, должен ли регистр адреса памяти увеличиваться или нет. Этот параметр может быть значением DMA_Memory_incremented_mode

- **uint32_t DMA_InitTypeDef :: PeriphDataAlignment**

Определяет ширину периферийных данных. Этот параметр может быть значением DMA_Peripheral_data_size

- **uint32_t DMA_InitTypeDef :: MemDataAlignment**

Задает ширину данных памяти. Этот параметр может быть значением DMA_Memory_data_size

- **uint32_t DMA_InitTypeDef :: Mode**

Задает режим работы DMAy Streamx. Этот параметр может быть значением DMA_mode. Примечание. Режим циклического буфера не может использоваться, если передача данных между памятью и памятью настроена на выбранном потоке

- **uint32_t DMA_InitTypeDef :: Priority**

Указывает приоритет программного обеспечения для DMAy Streamx. Этот параметр может быть значением DMA_Priority_level

- **uint32_t DMA_InitTypeDef :: FIFOMode**

Указывает, будет ли использоваться режим FIFO или прямой режим для указанного потока. Этот параметр может быть значением DMA_FIFO_direct_mode

Примечание. *Прямой режим (режим FIFO отключен) не может использоваться, если передача данных между памятью и памятью настроена на выбранном потоке*

- **uint32_t DMA_InitTypeDef :: FIFOThreshold**

Определяет пороговый уровень FIFO. Этот параметр может быть значением DMA_FIFO_threshold_level

- **uint32_t DMA_InitTypeDef :: MemBurst**

Определяет конфигурацию передачи Burst для передачи памяти. Он определяет объем данных, которые должны быть переданы в одной не прерываемой транзакции. Этот параметр может быть значением DMA_Memory_burst. Примечание. Режим пакетной передачи возможен только в том случае, если включен режим увеличения адреса.

- **uint32_t DMA_InitTypeDef :: PeriphBurst**

Указывает конфигурацию передачи Burst для периферийных передач. Он определяет объем данных, которые должны быть переданы в одной не прерываемой транзакции. Этот параметр может быть значением DMA_Peripheral_burst. Примечание. Режим пакетной передачи возможен только в том случае, если включен режим увеличения адреса.

20.1.2 __DMA_HandleTypeDef

Поля данных

- DMA_Stream_TypeDef * Instance
- DMA_InitTypeDef Init
- HAL_LockTypeDef Lock
- __IO HAL_DMA_StateTypeDef State
- void * Parent
- void (* XferCpltCallback
- void (* XferHalfCpltCallback
- void (* XferM1CpltCallback
- void (* XferM1HalfCpltCallback
- void (* XferErrorCallback
- void (* XferAbortCallback
- __IO uint32_t ErrorCode
- uint32_t StreamBaseAddress
- uint32_t StreamIndex

Назначение полей

- **DMA_Stream_TypeDef * __DMA_HandleTypeDef :: Init**

Базовый адрес регистра экземпляра

- **DMA_InitTypeDef __DMA_HandleTypeDef::Init**

Параметры связи DMA

- **HAL_LockTypeDef __DMA_HandleTypeDef :: Lock**

Залоченный DMA объект

- **__IO HAL_DMA_StateTypeDef __DMA_HandleTypeDef :: State**

Состояние передачи DMA

• **void* __DMA_HandleTypeDef::Parent**

Состояние родительского объекта

• **void(* __DMA_HandleTypeDef::XferCpltCallback)(struct __DMA_HandleTypeDef *hdma)**

DMA-передача полного обратного вызова

• **void(* __DMA_HandleTypeDef::XferHalfCpltCallback)(struct __DMA_HandleTypeDef *hdma)**

DMA-передача половинного обратного вызова

• **void(* __DMA_HandleTypeDef::XferM1CpltCallback)(struct __DMA_HandleTypeDef *hdma)**

DMA-передача полного обратного вызова Memory1

• **void(* __DMA_HandleTypeDef::XferM1HalfCpltCallback)(struct __DMA_HandleTypeDef *hdma)**

DMA-передача половинного обратного вызова Memory1

• **void(* __DMA_HandleTypeDef::XferErrorCallback)(struct __DMA_HandleTypeDef *hdma)**

DMA-передача половинного обратного вызова ошибки

• **void(* __DMA_HandleTypeDef::XferAbortCallback)(struct __DMA_HandleTypeDef *hdma)**

DMA-передача отменить обратный вызов

• **__IO uint32_t __DMA_HandleTypeDef::ErrorCode**

Код ошибки DMA

• **uint32_t __DMA_HandleTypeDef::StreamBaseAddress**

Базовый адрес потока DMA

• **uint32_t __DMA_HandleTypeDef::StreamIndex**

Индекс потока DMA

20.2 Описание API драйвера прошивки DMA

20.2.1 Как использовать этот драйвер

1. Включите и настройте периферийное устройство, которое должно быть подключено к потоку DMA (за исключением внутренних блоков памяти SRAM / FLASH: не требуется инициализация), обратитесь к справочному руководству для подключения периферийных устройств и запросов DMA.

2. Для данного потока запрограммируйте требуемую конфигурацию с помощью следующих параметров: Форматы передачи, Форматы данных источника и назначения, Режим управления круговым, нормальным или периферийным потоком, Уровень приоритета потока, Режим увеличения источника и назначения, режим FIFO и его порог (если необходимо), режим Burst для источника и / или назначения (если необходимо) с использованием функции HAL_DMA_Init (). До HAL_DMA_Init () часы должны быть включены для DMA с помощью следующих макросов: __HAL_RCC_DMA1_CLK_ENABLE () или __HAL_RCC_DMA2_CLK_ENABLE ().

Операция ввода-вывода в режиме опроса

• Используйте HAL_DMA_Start (), чтобы начать передачу DMA после настройки адреса источника и адреса получателя и длины передаваемых данных.

• Используйте HAL_DMA_PollForTransfer () для опроса для завершения текущей передачи, в этом случае фиксированный тайм-аут может быть настроен пользователем в зависимости от его приложения.

• Используйте функцию HAL_DMA_Abort (), чтобы прервать текущую передачу.

Режим прерывания операции ввода-вывода

• Настройте приоритет прерывания DMA, используя HAL_NVIC_SetPriority ()

• Включить обработчик IRQ DMA с помощью HAL_NVIC_EnableIRQ ()

• Используйте HAL_DMA_Start_IT (), чтобы начать передачу DMA после конфигурации адреса источника и адреса получателя и длины передаваемых данных. В этом случае прерывание DMA настраивается

• Используйте HAL_DMA_IRQHandler (), вызванный под DMA_IRQHandler () Подпрограмма прерывания

• В конце передачи данных выполняется функция HAL_DMA_IRQHandler (), и пользователь может добавить свою собственную функцию путем настройки указателя функции XferCpltCallback и XferErrorCallback (то есть члена структуры дескриптора DMA).

1. Используйте функцию HAL_DMA_GetState (), чтобы вернуть состояние DMA и HAL_DMA_GetError () в случае обнаружения ошибок.

2. Используйте функцию HAL_DMA_Abort_IT (), чтобы прервать текущую передачу. В режиме передачи данных с памятью режим Циркуляр не разрешен. FIFO используется в основном для уменьшения использования шины и для упаковки / распаковки данных: можно установить разные

Размеры данных для периферийного устройства и памяти (т. е. Вы можете установить размер данных Half-Word для периферийного устройства, чтобы получить доступ к его регистру данных и установить размер данных Word для памяти, чтобы получить время доступа. Каждое два слова будут упакованы и записаны в одном доступе к Word в памяти). Когда FIFO отключен, не разрешается настраивать различные размеры данных для источника и места назначения. В этом случае размер периферийных данных будет применяться как к источнику, так и к месту назначения.

Список макросов драйвера DMA HAL

Ниже список наиболее используемых макросов в драйвере DMA HAL.

• __HAL_DMA_ENABLE: Включить указанный поток DMA.

• __HAL_DMA_DISABLE: отключить указанный поток DMA.

• __HAL_DMA_GET_IT_SOURCE: проверьте, произошло ли указанное прерывание DMA Stream или нет.

Вы можете обратиться к файлу заголовка драйвера DMA HAL для получения более полезных макросов

20.2.2 Функции инициализации и деинициализации

В этом разделе представлены функции, позволяющие инициализировать источники и адреса отправителя DMA, размеры и размеры данных, направление передачи, выбор кругового / нормального режима, выбор режима памяти в память и значение приоритета потока.

Функция HAL_DMA_Init () следует процедурам конфигурации DMA, как описано в справочном руководстве.

Этот раздел содержит следующие API:

- HAL_DMA_Init ()
- HAL_DMA_DeInit ()

20.2.3 Функции ввода-вывода

В этом разделе представлены функции, позволяющие:

- Настройте источник, адрес назначения и длину данных и начните передачу DMA
- Настроить источник, адрес назначения и длину данных и начать передачу DMA с прерыванием
- Abort DMA transfer (Отмена передачи DMA)
- Poll for transfer complete (Опрос для передачи завершен)
- Handle DMA interrupt request (Обработать запрос прерывания DMA)

Этот раздел содержит следующие API:

- HAL_DMA_Start ()
- HAL_DMA_Start_IT ()
- HAL_DMA_Abort ()
- HAL_DMA_Abort_IT ()
- HAL_DMA_PollForTransfer ()
- HAL_DMA_IRQHandler ()
- HAL_DMA_RegisterCallback ()
- HAL_DMA_UnRegisterCallback ()
- HAL_DMA_CleanCallbacks ()

20.2.4. Функции состояния и ошибок

В этом подразделе представлены функции, позволяющие

- Check the DMA state (Проверьте состояние DMA)
- Get error code (Получить код ошибки)

Этот раздел содержит следующие API:

- HAL_DMA_GetState ()
- HAL_DMA_GetError ()

20.2.5 Подробное описание функций

21 Расширенный драйвер HAL DMA

21.1 Описание API драйвера прошивки DMAEx

21.1.1. Как использовать этот драйвер

Драйвер HAL Extension HAL можно использовать следующим образом:

1. Запустите передачу нескольких буферов, используя функцию HAL_DMA_ MultiBufferStart () для режима опроса или HAL_DMA_MultiBufferStart_IT () для режима прерывания. В режиме передачи данных с памятью режим Multi (Double) Buffer не допускается. Когда включен режим Multi (Double) Buffer, передача по умолчанию является циклической. В режиме Multi (Double) buffer можно обновлять базовый адрес для порта AHB на лету (DMA_SxM0AR или DMA_SxM1AR), когда поток включен.

21.1.2 Функции расширенного функционала

В этом разделе представлены функции, позволяющие:

- Настройте источник, адрес получателя и длину данных и запустите перенос DMA MultiBuffer
- Настроить источник, адрес назначения и длину данных и начать передачу DMA с помощью MultiBuffer с прерыванием
- Изменить на лету адрес памяти0 или memory1.

Этот раздел содержит следующие API:

- HAL_DMAEx_MultiBufferStart ()
- HAL_DMAEx_MultiBufferStart_IT ()
- HAL_DMAEx_ChangeMemory ()

21.1.3 Подробное описание функций