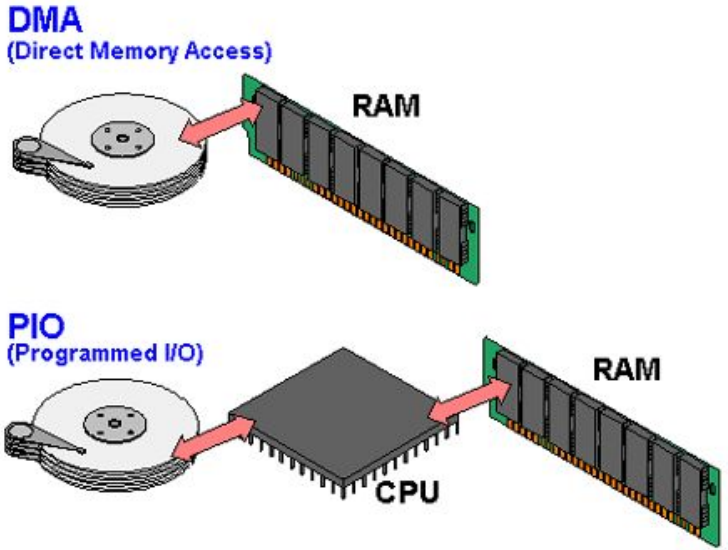
Программирование микроконтроллеров STM32

Прямой доступ к памяти

МФТИ, 2019

Зачем нужен модуль прямого доступа к памяти? (DMA - direct memory access)

- Разгрузить основной процессор
  - Ядро инициирует работу DMA
  - Выполняет свои задачи
  - Получает прерывание по окончанию работы DMA

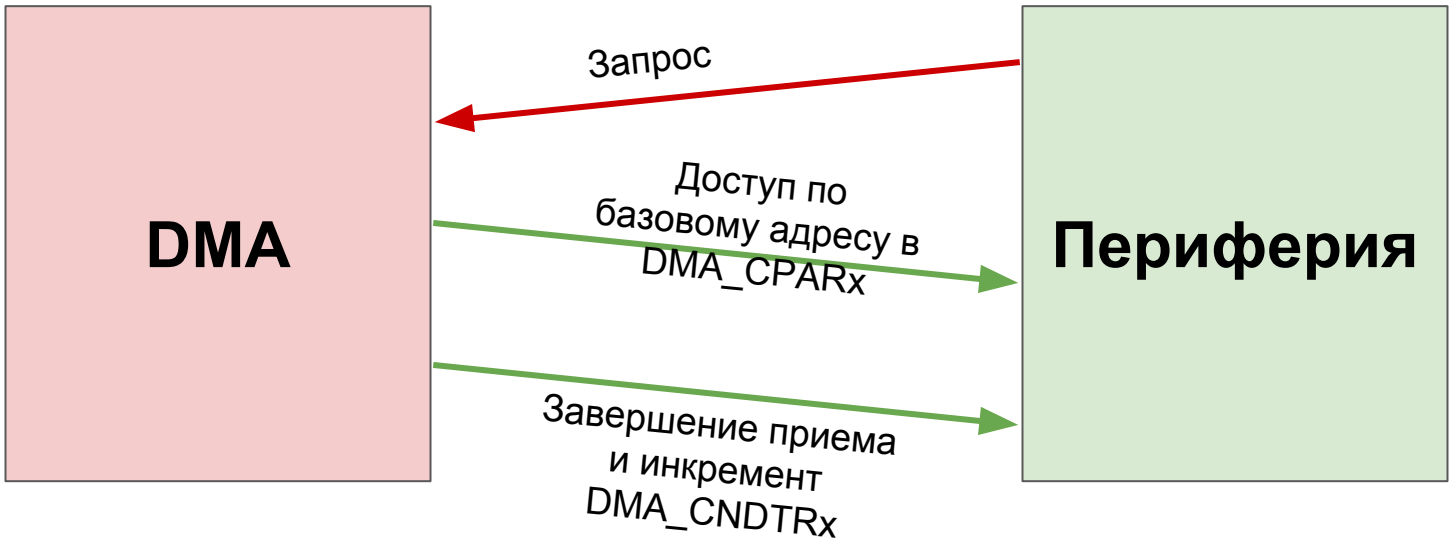


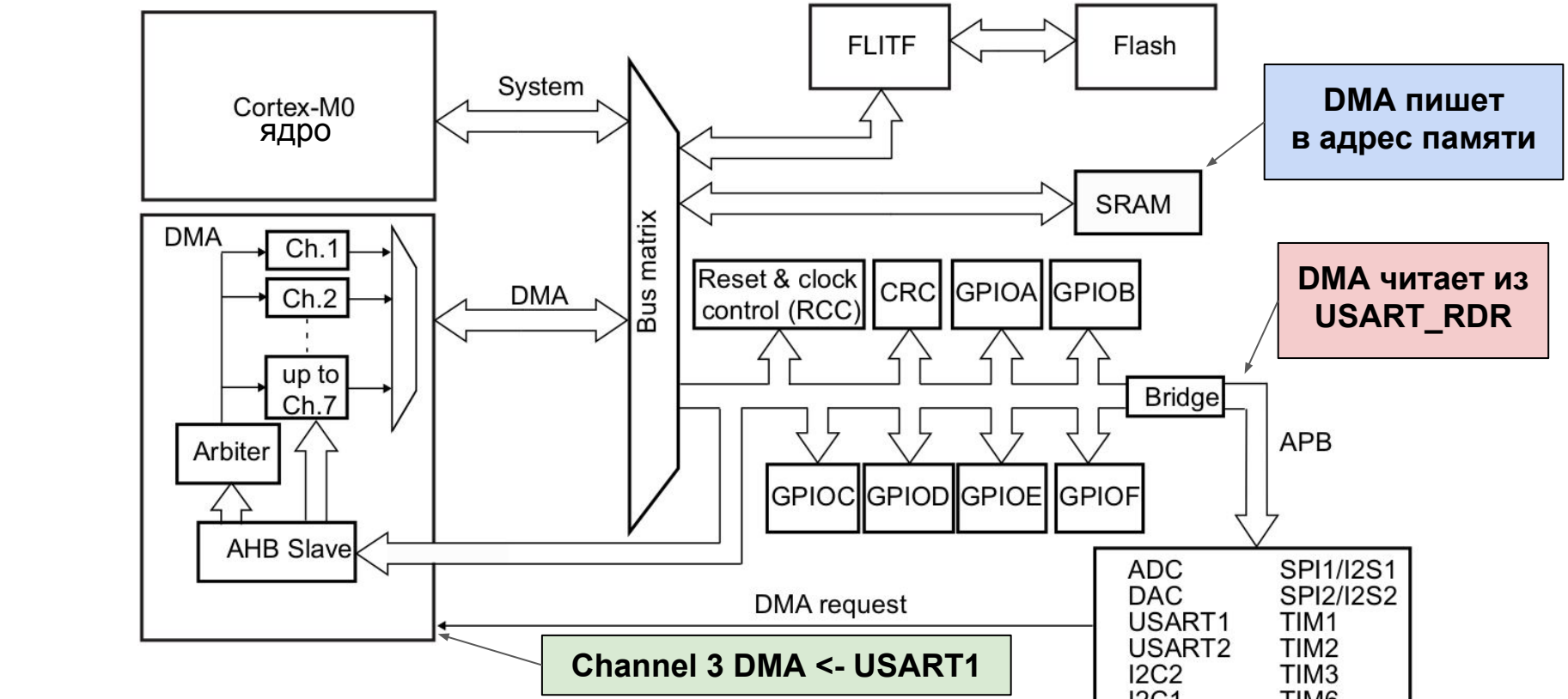
credits: pcmag.com

DMA в STM32

- 5 каналов
- 4 уровня приоритетов между каналами
- Конфигурируемый мин. размер сообщения (слово, полуслово, байт)
- Поддержка кольцевого режима
- 3 флага для всех каналов с прерыванием **(через лог. “ИЛИ”)**
- Передача данных между периферияими, между памятью, между памятью и периферией
- Максимальная длина пакета - 65536 сообщений

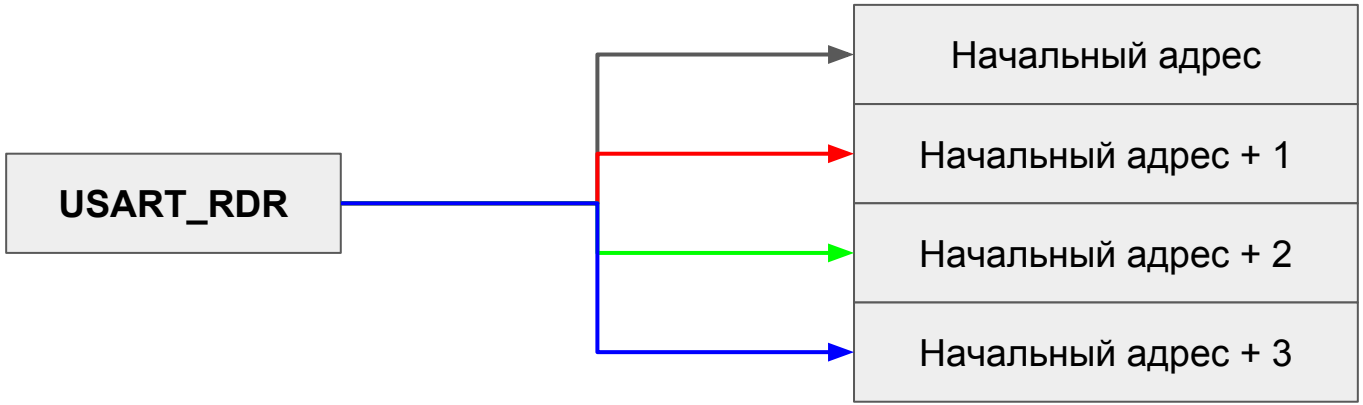
Пример транзакции в DMA





DMA и ядро используют одну и ту же системную шину, поэтому запросы со стороны ядра и DMA к одной и той же периферии будут предоставлены по очереди

DMA. Параметры передачи



Размер сообщения конфигурируется как для источника, так и для получателя:
В данном случае: размер сообщения для источника - 1 байт (USART_RDR), размер сообщения для получателя - 1 байт (SRAM)

DMA. Параметры передачи

Source port width	Destination port width	Number of data items to transfer (NDT)	Source content: address / data	Transfer operations	Destination content: address / data
8	8	4	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3	1: READ B0[7:0] @0x0 then WRITE B0[7:0] @0x0 2: READ B1[7:0] @0x1 then WRITE B1[7:0] @0x1 3: READ B2[7:0] @0x2 then WRITE B2[7:0] @0x2 4: READ B3[7:0] @0x3 then WRITE B3[7:0] @0x3	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3
8	32	4	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3	1: READ B0[7:0] @0x0 then WRITE 000000B0[31:0] @0x0 2: READ B1[7:0] @0x1 then WRITE 000000B1[31:0] @0x4 3: READ B2[7:0] @0x2 then WRITE 000000B2[31:0] @0x8 4: READ B3[7:0] @0x3 then WRITE 000000B3[31:0] @0xC	@0x0 / 000000B0 @0x4 / 000000B1 @0x8 / 000000B2 @0xC / 000000B3
32	8	4	@0x0 / B3B2B1B0 @0x4 / B7B6B5B4 @0x8 / BBBAB9B8 @0xC / BFBEBDBC	1: READ B3B2B1B0[31:0] @0x0 then WRITE B0[7:0] @0x0 2: READ B7B6B5B4[31:0] @0x4 then WRITE B4[7:0] @0x1 3: READ BBBAB9B8[31:0] @0x8 then WRITE B8[7:0] @0x2 4: READ BFBEBDBC[31:0] @0xC then WRITE BC[7:0] @0x3	@0x0 / B0 @0x1 / B4 @0x2 / B8 @0x3 / BC

DMA. Инициализация

- Подача тактирования на модуль DMA
- Указание адреса памяти *LL_DMA_SetMemoryAddress* [CMAR]
- Указание адреса периферии *LL_DMA_SetPeriphAddress* [CPAR]
- Настройка направления передачи данных
  - *LL_DMA_SetDataTransferDirection* [CCR]
- Настройка длины пакета *LL_DMA_SetDataLength* [CNDTR]
- Параметры работы с сообщениями
  - *LL_DMA_SetMemoryIncMode* [CCR]
  - *LL_DMA_SetPeriphIncMode* [CCR]
  - *LL_DMA_SetPeriphSize* [CCR]
  - *LL_DMA_SetMemorySize* [CCR]
- Настройка режима DMA *LL_DMA_SetMode* [CCR] (кольцевой или нет)
- Включение модуля DMA *LL_DMA_EnableChannel* [CCR]

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
ADC	ADC(1)	ADC(2)	-	-	-
SPI	-	SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX
USART	-	USART1_TX(1)	USART1_RX(1)	USART1_TX(2) USART2_TX	USART1_RX(2) USART2_RX
I2C	-	I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX
TIM1	-	TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_CH3 TIM1_UP
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_Channel1	-	-
TIM15	-	-	-	-	TIM15_CH1 TIM15_UP TIM15_TRIG TIM15_COM
TIM16	-	-	TIM16_CH1(1) TIM16_UP(1)	TIM16_CH1(2) TIM16_UP(2)	-
TIM17	TIM17_CH1(1) TIM17_UP(1)	TIM17_CH1(2) TIM17_UP(2)	-	-	-

DMA. Прерывания

Interrupt event	Event flag	Enable control bit
Half-transfer	HTIF	HTIE
Transfer complete	TCIF	TCIE
Transfer error	TEIF	TEIE

- Включение прерывания *LL_DMA_EnableIT_TC* [CCR]
- Настройка NVIC

  DMA1_Channel1_IRQHandler
  DMA1_Channel2_3_IRQHandler
  DMA1_Channel4_5_IRQHandler
- Проверка флага *LL_DMA_IsActiveFlag_TCx* [ISR]
- Сброс флага *LL_DMA_ClearFlag_TC5* [IFCR]

Репозиторий

https://github.com/edosedgar/stm32f0_ARM