STM32 course

WOW, DMA!

Going back to ADC for a while

- Main features
- Set of registers
- Single, continuous and discontinuous modes
- ADC clock sources
- Sharp difference between sampling and converting
- Overrun
- Low power mode
- Temperature sensor

Outline

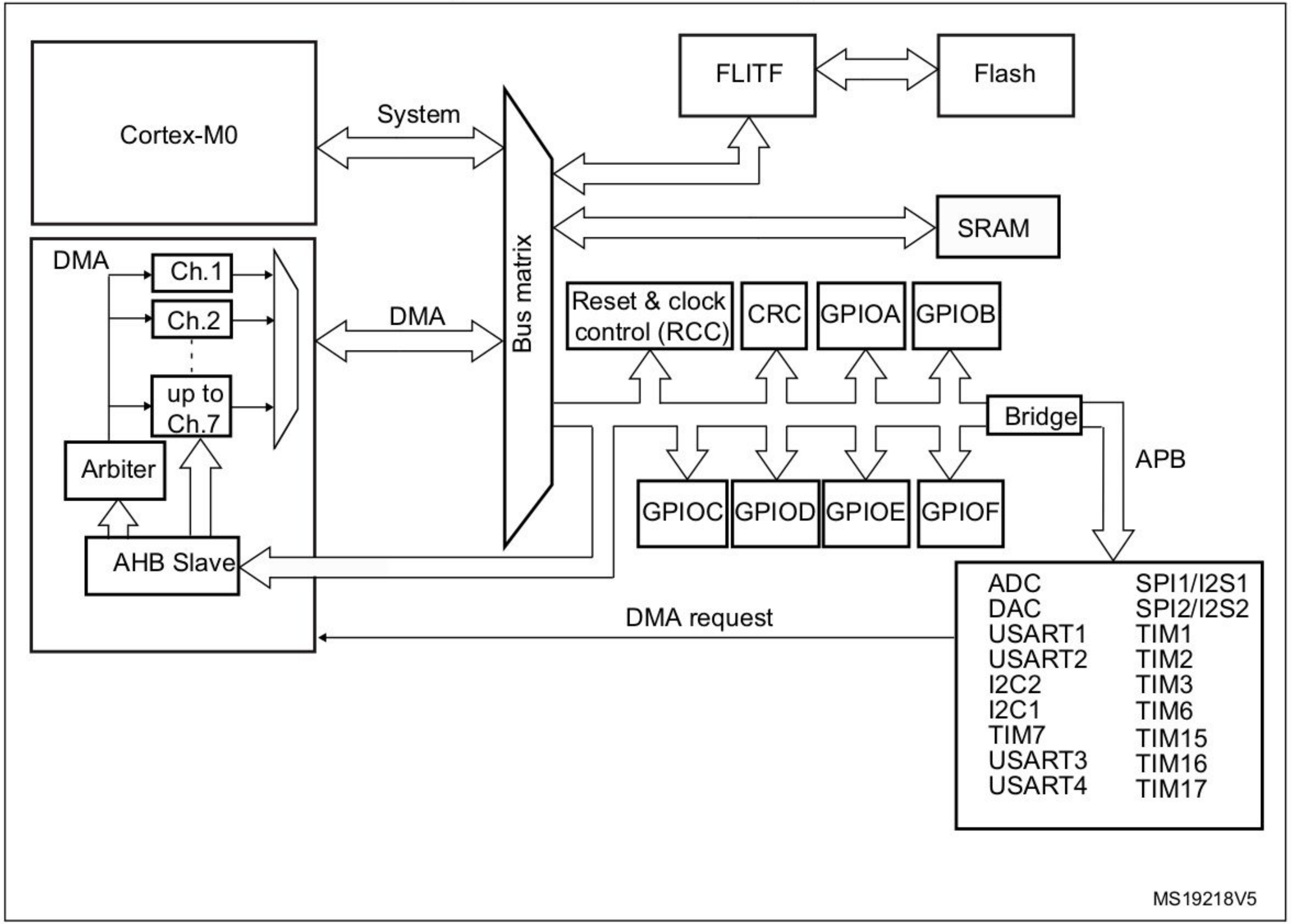
~~ADC~~

DMA

DMA. Main features

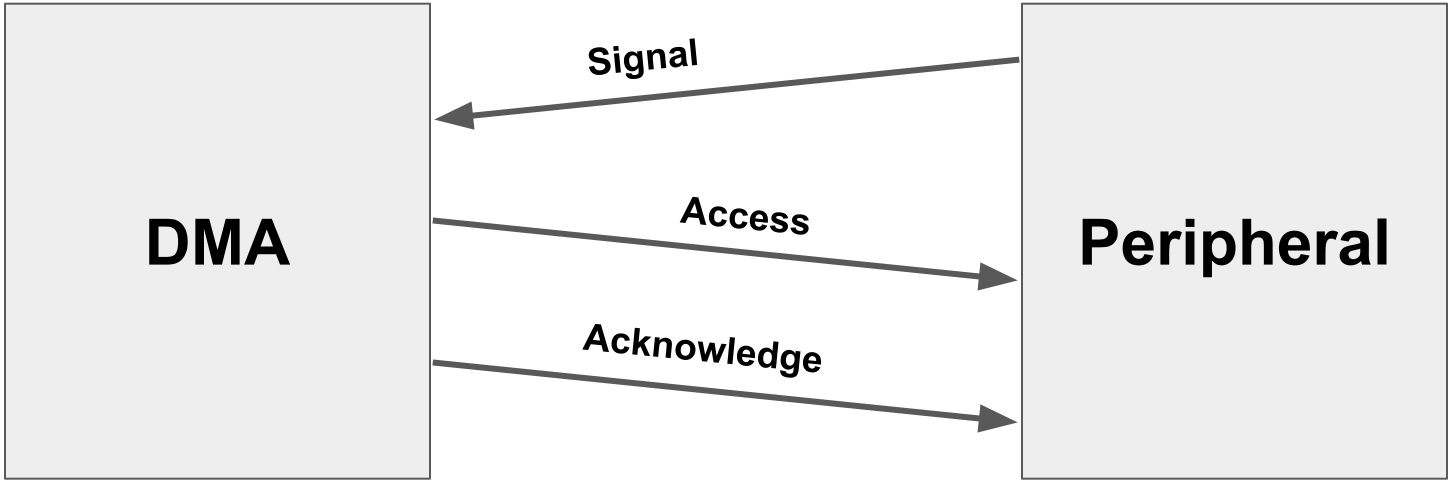
- Up to 7 independently configurable channels
- Priorities between requests from the DMA channels are software programmable
- Independent source and destination transfer size (byte, halfword, word)
- Support for circular buffer management
- 3 event flags (DMA Half Transfer, DMA Transfer complete and DMA Transfer Error)
- Memory-to-memory transfer
- Peripheral-to-memory and memory-to-peripheral, and peripheral-to-peripheral transfers
- Access to Flash, SRAM, APB and AHB peripherals as source and destination
- Programmable number of data to be transferred: up to 65535

DMA. Diagram



Cortex-M0
System
FLITF
Flash
SRAM
DMA
Ch.1
Ch.2
up to Ch.7
Arbiter
AHB Slave
DMA
Bus matrix
Reset & clock control (RCC)
CRC
GPIOA
GPIOB
Bridge
APB
GPIOC
GPIOD
GPIOE
GPIOF
ADC
DAC
USART1
USART2
I2C2
I2C1
TIM7
USART3
USART4
SPI1/I2S1
SPI2/I2S2
TIM1
TIM2
TIM3
TIM6
TIM15
TIM16
TIM17
DMA request
MS19218V5

DMA transactions



DMA_CPARx, DMA_CMARx and DMA_CNDTRx

Arbiter

- The arbiter manages the channel requests based on their priority and launches the peripheral/memory access sequences
- Software: each channel priority can be configured in the DMA_CCRx register. There are four levels:
  - Very high priority
  - High priority
  - Medium priority
  - Low priority
- Hardware: if 2 requests have the same software priority level, the channel with the lowest number will get priority versus the channel with the highest number.

DMA. Channel configuration procedure

- Set the peripheral register address in the DMA_CPARx register
- Set the memory address in the DMA_CMARx register
- Configure the total number of data to be transferred in the DMA_CNDTRx register
- Configure the channel priority using the PL[1:0] bits in the DMA_CCRx register
- Configure data transfer direction, circular mode, peripheral & memory incremented mode, peripheral & memory data size, and interrupt after half and/or full transfer in the DMA_CCRx register
- Activate the channel by setting the ENABLE bit in the DMA_CCRx register

```c
RCC->AHBENR |= RCC_AHBENR_DMA1EN;
ADC1->CFGR1 |= ADC_CFGR1_DMAEN;
DMA1_Channel1->CPAR = (uint32_t) (&(ADC1->DR));
DMA1_Channel1->CMAR = (uint32_t)(ADC_array);
DMA1_Channel1->CNDTR = 3;
/* MINC - Memory increment mode; MSIZE_0 - 8-bits memory;
   PSIZE_0 - 8-bits peripheral; TEIE - Transfer error interrupt;
   TCIE - Transfer complete interrupt */
DMA1_Channel1->CCR |= DMA_CCR_MINC | DMA_CCR_MSIZE_0 |
                      DMA_CCR_PSIZE_0 | DMA_CCR_TEIE |
                      DMA_CCR_TCIE ;
DMA1_Channel1->CCR |= DMA_CCR_EN;
NVIC_EnableIRQ(DMA1_Channel1_IRQn);
NVIC_SetPriority(DMA1_Channel1_IRQn,0);
```

DMA. Reminder of sizes

- When the DMA initiates an AHB byte or halfword write operation, the data are duplicated on the unused lanes of the HWDATA[31:0] bus
- To write the halfword “0xABCD”, the DMA sets the HWDATA bus to “0xABCDABCD” with HSIZE = HalfWord
- To write the byte “0xAB”, the DMA sets the HWDATA bus to “0xABABABAB” with HSIZE = Byte
- An AHB byte write operation of the data “0xB0” to 0x0 (or to 0x1, 0x2 or 0x3) will be converted to an APB word write operation of the data “0xB0B0B0B0” to 0x0
- An AHB halfword write operation of the data “0xB1B0” to 0x0 (or to 0x2) will be converted to an APB word write operation of the data “0xB1B0B1B0” to 0x0

DMA. Error management

- Error can be generated by reading from or writing to a reserved address space
- Only it happens, the faulty channel is automatically disabled through a hardware clear of its EN bit in the corresponding Channel configuration register (DMA_CCRx)
- The channel transfer error interrupt flag (TEIF) in the DMA_IFR register is set and an interrupt is generated if the transfer error interrupt enable bit (TEIE) in the DMA_CCRx register is set

DMA. Interrupts

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

DMA. Be aware of limitations

The hardware requests from the peripherals (TIMx, ADC, DAC, SPI, I2C, and USARTx) are simply logically ORed before entering the DMA. This means that on one channel, only one request must be enabled at a time.

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. Registers

Prepare for switching to the reference manual

ADC with DMA

Switch to example!