

03 DMA

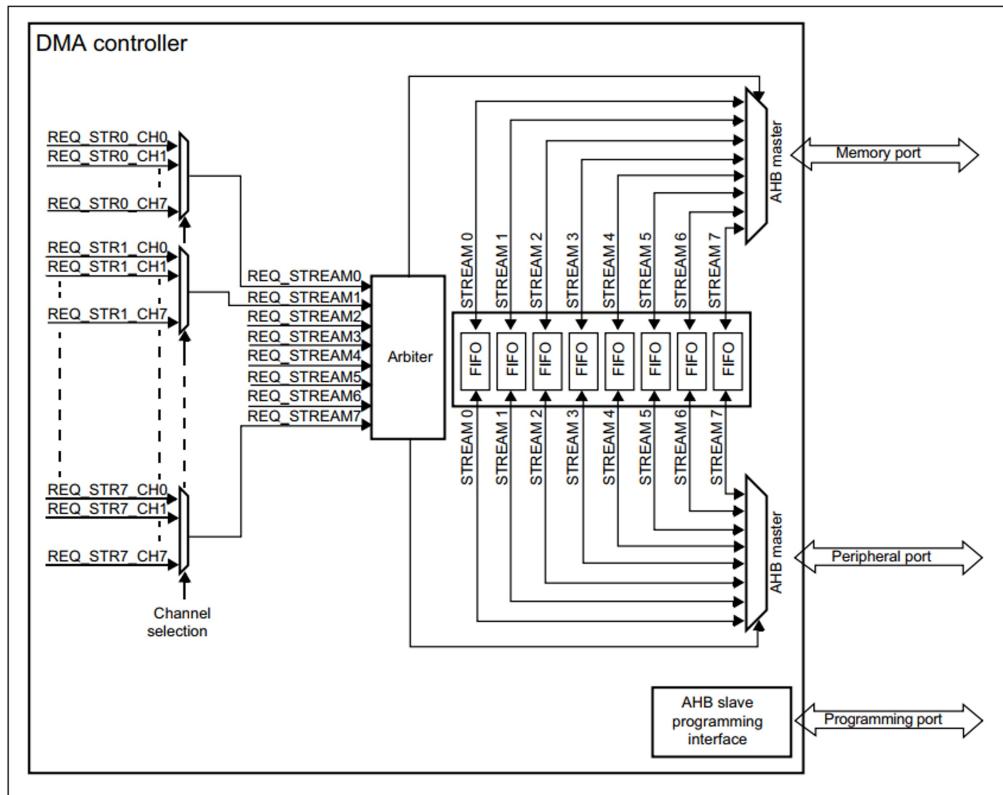
5 Mayıs 2021 Çarşamba 08:03

03 DMA

Giriş

- <https://mikrodunya.wordpress.com/2016/06/23/dma-direct-memory-access-dogrudan-bellek-erisimi/>
 - DMA (Direct Memory Adres), veri transferlerini hiç bir CPU işlemini kullanmadan sağlaması amacıyla oluşturulmuştur.
 - Çeşitli çevre birimlerinden okuduğumuz verileri bir değişken'e atarız. Bu değişkenler RAM'de depolanır. Bu işlem normalde çevre birimlerinde okunan verinin CPU'ya alınıp ardından RAM'e yazılır. Ancak CPU kullanımı hem işlemciyi yorar hemde kayıplara yol açar. DMA sayesinde verileri direkt olarak **RAM'e** yazma imkanı buluruz.
 - DMA donanımı CPU'dan bağımsız olarak verilerimizi hızlı bir şekilde kaynak adresinden hedef adres'e aktarır.
 - **peripheral -> memory**, bir çevre biriminden (peripheral) gelen veri doğrudan belleğe (memory) aktarılır.
 - **memory -> peripheral**, bellekten (memory) alınan veri doğrudan bir çevre birime (peripheral) gönderilir.
 - **memory -> memory**, bir bellek bölgesindeki (memory) veri başka bir bellek bölge'ne aktarılır.
 - Bu sayede CPU'nun yükünü hafifletmiş oluruz. Sistem sanki 2 CPU ile çalışıyordu gibi düşünebiliriz. Örneğin bilgisayarlarımıza bulunan 4 gerçek 4 sanal çekirdekteki sanal, aslında DMA diyebiliriz. DMA isteği için çevresel birim tarafından (ADC, DAC, I2C vs) DMA kontrolcüsüne istek gönderilir, kontrolcüde bu isteğin sırası gelince ilgili çevresel birime geri bildirimde bulunur ve işlem kaynak adresinden hedef adres'e doğru gerçekleşir.
 - DMA işlemi, özellikle **çok fazla veri** alışverişesi yapıldığı durumlarda kullanılmalıdır. CPU Yükünü Azaltma, Verimlilik, Hız, Kesintisizlik gibi nedenlerden dolayı sistem performansını önemli ölçüde artırır.
 - STM32F4'te iki adet DMA vardır. DMA1'in DMA 2'den kanal 1'in kanal 2'den yüksek olduğu bilinmektedir. Öncelik sırası belirtmek için dört seviye vardır. Low, Medium, High, Very High.
- Aynı anda birçok kanal kullanıldığında hangi kanalın öncelik değeri fazla ise ilk o kanal alınır. DMA'lar paralel olarak çalışmazlar, seri olarak çalışırlar. Bu nedenle hangisinin sırası geldi ise o anda o çalışır.

Birim Yapısı



- AHB Master Portları:** Şemanın sağ tarafında görülen AHB Master portları, DMA'nın hem bellek portuna hem de peripheral portlara veri aktarımı yapabilmesini sağlar. Yüksek performanslı bir veri yolu olan AHB'yi kullanarak, verilerin hızlı bir şekilde taşınmasını sağlar.
 - Programming Port**, DMA'nın programlanması ve yapılandırılmasını sağlayan Mikrodenetleyicinin çekirdeği (CPU), DMA'yı programlamak için bu portu kullanır. Bu port üzerinden DMA akışları, kanallar, öncelikler ve diğer parametreler ayarlanır.
 - Memory Port**, DMA'nın belleğe erişmesini sağlar. Bellek portu üzerinden DMA, veriyi bellekten alabilir veya belleğe yazabilir.
 - Peripheral Port**, DMA'nın peripheral cihazlarla veri alışverişini gerçekleştirir. Örneğin, bir UART cihazından veri almak veya bir ADC'den ölçülen veriyi bellek adresine aktarmak için bu port kullanılabilir.
- Channel:** Şemanın sol tarafında, her bir kanal için (CH0-CH7) bir seçim süreci olduğunu görebiliriz. Bu kanallar, belirli bir donanım isteği (örneğin, ADC, SPI, UART gibi) karşılık gelir. Her bir kanal belirli bir DMA akışına (Stream) atanabilir.
- Streams:** Her bir DMA denetleyicisi 8 ayrı akışa sahiptir (Stream 0 - Stream 7). Bu akışlar, aynı anda birden fazla DMA işleminin yürütülmesine izin verir. Bir akış, belirli bir kaynaktan (örneğin, bir peripheral) veriyi alıp belirli bir hedefe (örneğin, bellek) iletebilir. Her akış, belirli bir kanal tarafından tetiklenir ve bu kanal üzerinden veri taşırlar.

Her bir stream, 8 olası kanal isteği arasından seçilebilecek bir DMA isteği ile ilişkilendirilmiştir. Çevresel birimlerden (TIM, ADC, SPI, I2C, vb.) gelen 8 request, her bir kanala bağımsız olarak bağlanır ve bu bağlantı ürünün uygulanmasına bağlıdır. DMA istek eşlemeleri için aşağıdaki tabloları inceleyebilirsiniz.

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
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

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

- Arbiter:** DMA denetleyicisinde, aynı anda birden fazla akışın (Stream) çalışabileceğini söylemek. Ancak aynı anda birden fazla akışın aynı hedefe (örneğin, belleğe) erişmeye çalışması durumunda, bir çakışma

olabilir. Bu çakışmayı yönetmek için bir arbiter bulunur. Arbiter, hangi akışın öncelikli olacağına ve hangi akışın belleğe veya peripheral porta erişim sağlayacağına karar verir.

- **FIFO:** Her akışın (Stream) bir FIFO tampon belleği (buffer) vardır. Bu FIFO tamponları, veri akışını geçici olarak depolamak için kullanılır. FIFO kullanımı, DMA'nın veriyi daha etkin bir şekilde yönetmesine yardımcı olur, özellikle de veri hızları arasında bir uyuşmazlık olduğunda (örneğin, veri kaynağı veriyi çok hızlı gönderiyorsa veya hedef veriyi daha yavaş alıyorsa).

Register

Offset	Register	31	30	29	28	
0x0000	DMA_LISR	Reserved	TCIF3	27		
			0	TCIF3	27	
0x0004	DMA_HISR	Reserved	HTIF3	26		
			0	HTIF3	26	
0x0008	DMA_LIFCR	Reserved	TEIF3	25		
			0	TEIF3	25	
0x000C	DMA_HIFCR	Reserved	CDMEIF3	24		
			0	CDMEIF3	24	
0x0010	DMA_S0CR	Reserved	CFEIF7	23		
			0	CFEIF7	23	
0x0014	DMA_S0NDTR	Reserved	CTCIF7	22		
			0	CTCIF7	22	
0x0018	DMA_S0PAR	Reserved	CHSEL[2:0]	21		
			0	CHSEL[2:0]	21	
0x001C	DMA_S0M0AR	Reserved	CTEIF6	20		
			0	CTEIF6	20	
0x0020	DMA_S0M1AR	Reserved	CDMEIF6	19		
			0	CDMEIF6	19	
0x0024	DMA_S0FCR	Reserved	PL[1:0]	18		
			-	PL[1:0]	18	
0x0028	DMA_S1CR	Reserved	CFEIF6	17		
			0	CFEIF6	17	
0x002C	DMA_S1NDTR	Reserved	CTCIF6	16		
			0	CTCIF6	16	
0x0030	DMA_S1PAR	Reserved	CHSEL[2:0]	15		
			0	CHSEL[2:0]	15	
0x0034	DMA_S1M0AR	Reserved	CTEIF6	14		
			0	CTEIF6	14	
0x0038	DMA_S1M1AR	Reserved	CDMEIF6	13		
			0	CDMEIF6	13	
0x003C	DMA_S1FCR	Reserved	PL[1:0]	12		
			-	PL[1:0]	12	
0x0040	DMA_S2CR	Reserved	CFEIF5	11		
			0	CFEIF5	11	
0x0044	DMA_S2NDTR	Reserved	CTCIF5	10		
			0	CTCIF5	10	
0x0048	DMA_S2PAR	Reserved	CHSEL[2:0]	9		
			0	CHSEL[2:0]	9	
0x004C	DMA_S2M0AR	Reserved	CTEIF5	8		
			0	CTEIF5	8	
0x0050	DMA_S2M1AR	Reserved	CDMEIF5	7		
			0	CDMEIF5	7	
0x0054	DMA_S2FCR	Reserved	PL[1:0]	6		
			-	PL[1:0]	6	
0x0058	DMA_S3CR	Reserved	PFCTRL	5		
			0	PFCTRL	5	
0x005C	DMA_S3NDTR	Reserved	TCIE	4		
			0	TCIE	4	
0x0060	DMA_S3PAR	Reserved	HTIE	3		
			0	HTIE	3	
0x0064	DMA_S3M0AR	Reserved	TEIE	2		
			0	TEIE	2	
0x0068	DMA_S3M1AR	Reserved	DMEIE	1		
			0	DMEIE	1	
0x006C	DMA_S3FCR	Reserved	EN	0		
			0	EN	0	
0x0070	DMA_S4CR	Reserved	CFEIF4	0		
			0	CFEIF4	0	
0x0074	DMA_S4NDTR	Reserved	CTCIF4	0		
			0	CTCIF4	0	
0x0078	DMA_S4PAR	Reserved	CHSEL[2:0]	1		
			0	CHSEL[2:0]	1	
0x007C	DMA_S4M0AR	Reserved	CTEIF4	0		
			0	CTEIF4	0	
0x0080	DMA_S4M1AR	Reserved	CDMEIF4	0		
			0	CDMEIF4	0	
0x0084	DMA_S4FCR	Reserved	PL[1:0]	0		
			-	PL[1:0]	0	
0x0088	DMA_S5CR	Reserved	FEIE	0		
			0	FEIE	0	
0x0092	DMA_S5NDTR	Reserved	FS[2:0]	1		
			1	FS[2:0]	1	
0x0096	DMA_S5PAR	Reserved	DMDIS	0		
			0	DMDIS	0	
0x009A	DMA_S5M0AR	Reserved	FTH[1:0]	1		
			1	FTH[1:0]	1	

- **DMA_LISR** ve **DMA_HISR** (Low/High Interrupt Status Register), DMA'nın düşük ve yüksek öncelikli kesmelerin durumunu izleyen register'lardır. Her bir bit, ilgili DMA kanalındaki bir kesmeyi temsil eder.
- **DMA_LIFCR** ve **DMA_HIFCR** (Low/High Interrupt Flag Clear Register), DMA'nın düşük ve yüksek öncelikli kesme bayraklarını temizlemek için kullanılır. Her bir bit, ilgili DMA kanalındaki bir kesme bayrağını temsil eder.
- **DMA_SxCR** (Stream x Configuration Register), DMA'nın belirli bir kanalının yapılandırma register'ıdır. Kanalın çalışma modu, transfer yönü, veri genişliği, bellek ve perifer adresi inkrement modu gibi özelliklerini içerir.
- **DMA_SxNDTR** (Stream x Number of Data Register), ilgili DMA kanalında aktarılacak veri miktarını belirten register'dır.
- **DMA_SxPAR** (Stream x Peripheral Address Register), DMA'nın belirli bir kanalındaki perifer başlangıç adresini belirten register'dır.
- **DMA_SxMxAR** ve **DMA_SxMxAR** (Stream x Memory 0/1 Address Register), DMA'nın belirli bir kanalındaki bellek başlangıç adreslerini belirten register'lardır. Bazı STM32 modellerinde birden fazla bellek adresi kullanılabilir.

- **DMA_SxFIFO (Stream x FIFO Control Register)**, DMA FIFO (First In, First Out) kontrolünü sağlayan register'dır. FIFO'nun kullanılması, DMA transfer performansını artırabilir.

Süreç

- DMA kullanarak herhangi bir işlemci üzerinde veri transferi gerçekleştirmek için genel adım sıralaması aşağıdaki gibidir.
 - **Kullanılacak DMA Denetleyicisinin Belirlenmesi**
 - Uygulamanız için uygun olan DMAx denetleyicisini belirleyin. İşlemcinizin hangi DMA kanallarının kullanılabilir olduğunu öğrenin.
 - **Yapıllandırma Ayarlarının Yapılması**
 - Verilerin **nereden nereye** transfer edileceğini belirleyin.
 - Transfer edilecek **veri miktarını** (byte, half-word, word) belirleyin
 - Veri transferinin **yönü** (kaynak->hedef) ve **modunu** (normal, döngüsel, vb.) ayarlayın.
 - Transfer tamamlandığında veya hata durumunda kesme kullanmak istiyorsanız, ilgili kesme yapılandırmalarını yapın ve kesme hizmet rutini (ISR) tanımlayın.
 - **Transferin Tetiklenmesi**
 - Tüm yapılandırmalar tamamlandıktan sonra, DMA kanalını başlatın. Bu işlem, yazılım veya donanım tetiklemesi ile gerçekleştirilebilir.
 - **Transferin İzlenmesi ve Tamamlanmasının Beklenmesi**
 - Transfer sırasında DMA durumunu izleyin ve transferin tamamlanmasını bekleyin. Eğer kesme kullanıyorsanız, kesme servis rutininde gerekli işlemleri yapın.
 - Transfer tamamlandıktan sonra DMA kanalını devre dışı bırakın ve gerekli temizleme işlemlerini yapın.