Sistemas Electrónicos Digitales

Tema #3

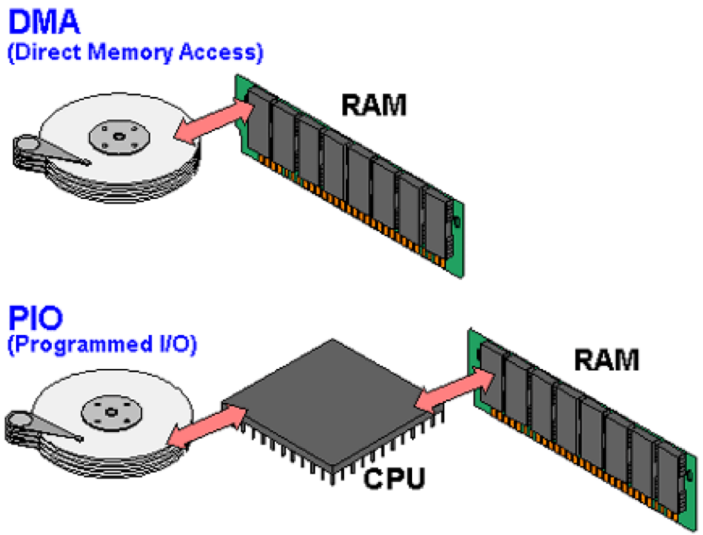
7. Acceso Directo a Memoria

Temario

1. Introducción
2. GPIO: General Purpose Input/Output
3. Arquitectura Arm Cortex-M4
4. Interrupciones
5. C en ensamblador
6. Temporizadores (Timers)
7. Acceso Directo a Memoria (DMA)
8. Comunicaciones Serie
9. Conversores A/D y D/A

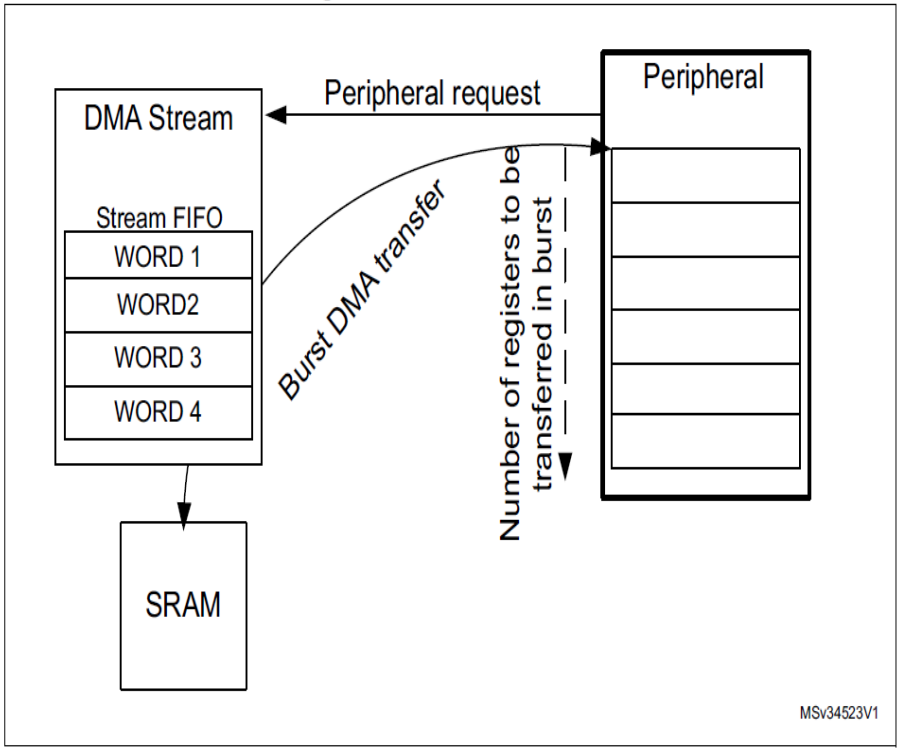
DMA: concepto

- El **Acceso Directo a Memoria** (**DMA**) es una prestación presente en muchos sistemas basados en ordenador que permite que componentes del sistema accedan a la memoria principal sin intervención de la CPU.



DMA: concepto (II)

Figure 5. DMA burst transfer



¡¡¡ NO INTERVIENE LA CPU !!!

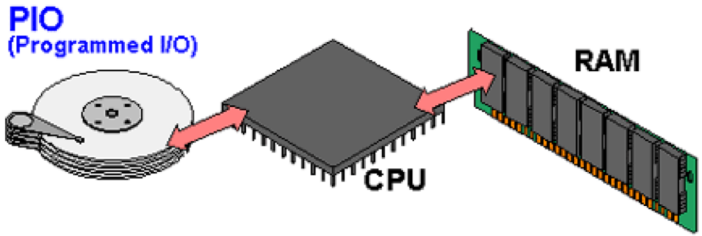
Ventajas del uso de DMA

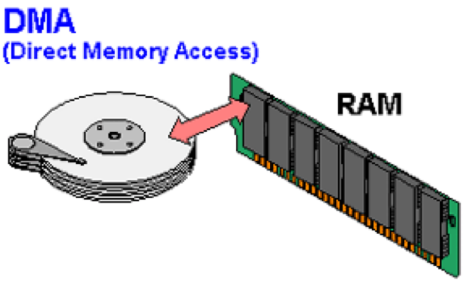
Sin DMA (PIO)

Si la CPU usa E/S programada (PIO), está ocupada por completo durante toda la lectura y escritura. No puede realizar otra tarea.

Con DMA

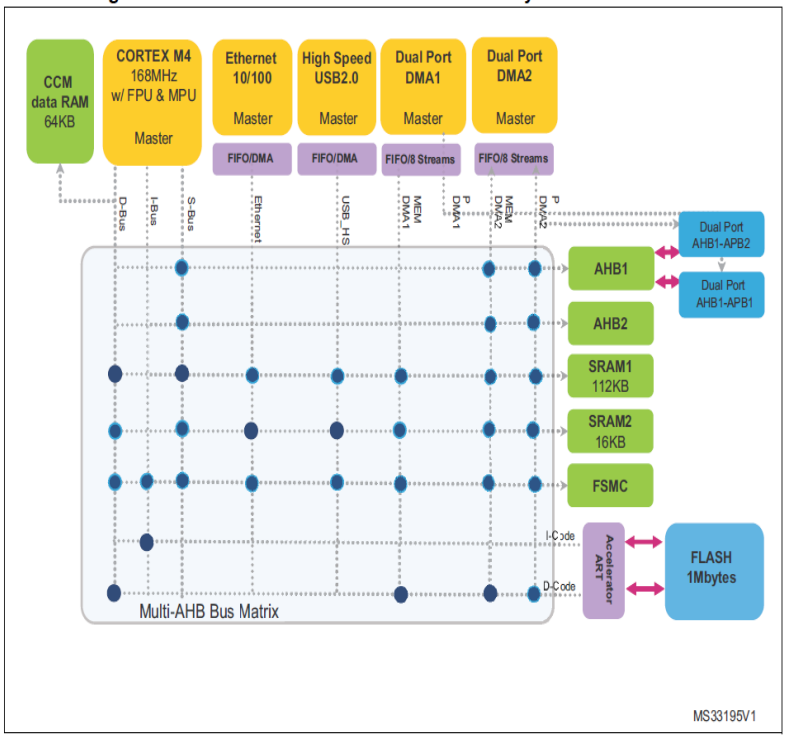
La CPU inicializa la transferencia, realiza otras tareas mientras la transferencia está en curso y recibe una interrupción desde el controlador de DMA cuando la operación acaba.



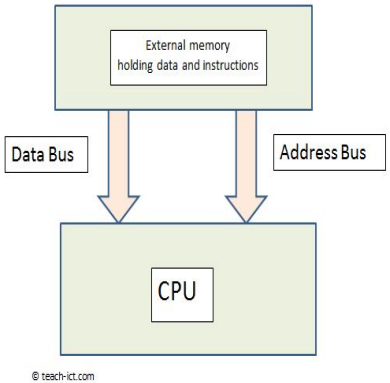


DMA en el STM32F4

Figure 7. STM32F405/415 and STM32F407/417 system architecture



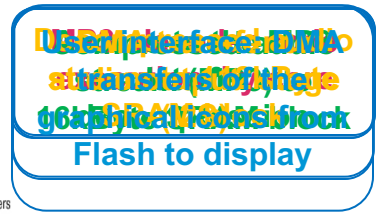
VS

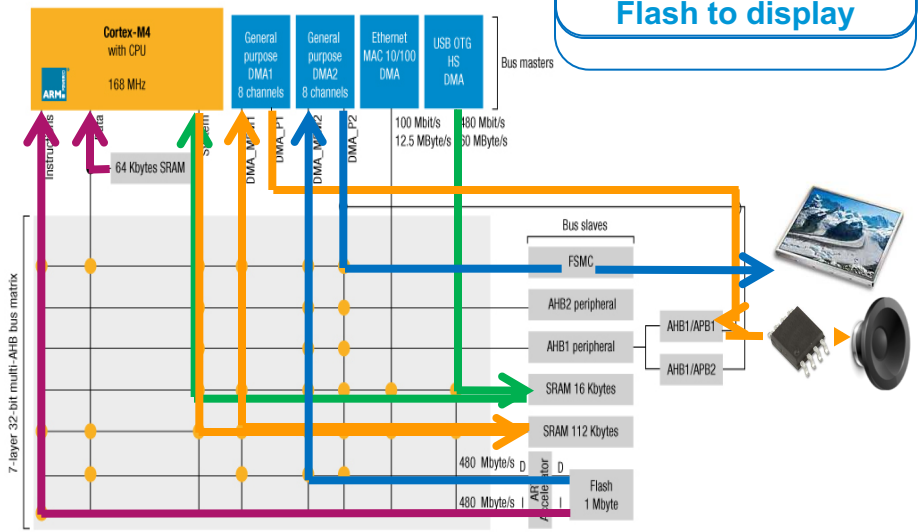


Multiple Data paths!!!!

Ejemplo de uso de DMA

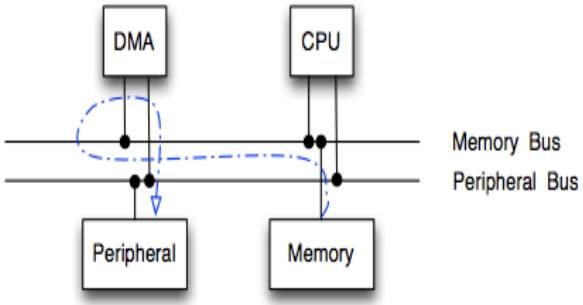
32-bit multi-AHB bus matrix





Módulos DMA

- STM32 tiene dos controladores DMA. Cada uno tiene varios “canales” configurables (7 para DMA1 y 5 para DMA2). Un canal es como la realización hardware de una transacción.
- Para inicializar una transacción DMA entre un periférico y memoria, es necesario configurar el canal apropiado. por ejemplo, los canales 2 (3) del DMA1 pueden usarse para recibir (transmitir) datos de (a) SPI1.



Configurar una transferencia DMA

- Antes de utilizar DMA, hay que habilitar el reloj del periférico.

```
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_DMA1, ENABLE);
```

- Configurar un canal consiste en fijar los parámetros apropiados en una estructura DMA_InitTypeDef
  - Los parámetros incluyen la dirección base del periférico (ej. ADC1-> DR), la dirección de memoria del buffer, la dirección de la transferencia, el tamaño del buffer, etc.

Configurar una transferencia DMA (II)

- Estructura DMA_InitTypeDef

```
typedef struct
{
  uint32_t DMA_PeripheralBaseAddr
  uint32_t DMA_MemoryBaseAddr;
  uint32_t DMA_DIR;
  uint32_t DMA_BufferSize;
  uint32_t DMA_PeripheralInc;
  uint32_t DMA_MemoryInc;
  uint32_t DMA_PeripheralDataSize;
  uint32_t DMA_MemoryDataSize;
  uint32_t DMA_Mode;
  uint32_t DMA_Priority;
  uint32_t DMA_M2M;
}DMA_InitTypeDef;
```

Configurar una transferencia DMA (III)

- Una vez inicializado, hay que habilitarlo. STM32 proporciona comandos específicos.

```
DMA_Init(txChan, &DMA_InitStructure);
DMA_Cmd(txChan, ENABLE);
```

- Hay que activar el DMA en el periférico:

ADC_DMACmd(ADC1, ENABLE);
ADC_DMARequestAfterLastTransferCmd(ADC1,ENABLE);
ó

```
SPI_I2S_DMACmd(SPIx, SPI_I2S_DMAReq_Tx, ENABLE);
```

- Los canales DMA en STM32 están asociados a periféricos específicos. Por ejemplo, el canal 1 del DMA1 está asociado a ADC1, TIM2_CH3 y TIM4_CH1.

Fin