SIOC嵌入式軟體實驗
實驗七 ：DMA

WU-YANG
Technology Co., Ltd.

support.wuyang@gmail.com

實驗目的

- 控制DMA自動搬一記憶體或周邊之資料，並透過VCP傳送到超級終端機顯示操作過程。

實作重點

- DMA的控制
  - Polling
  - Interrupt

SIOC DMA 簡介

- DMA(Direct Memory Access)，無須CPU可直接控制記憶體，透過DMA，能使CPU的效率大為提高。

- 最多有兩個DMA控制器，DMA1有7個通道，每個通道專門用來管理來自一個或多個外部的請求，還有一個仲裁器來協調各個DMA請求的優先權。

DMA Registers

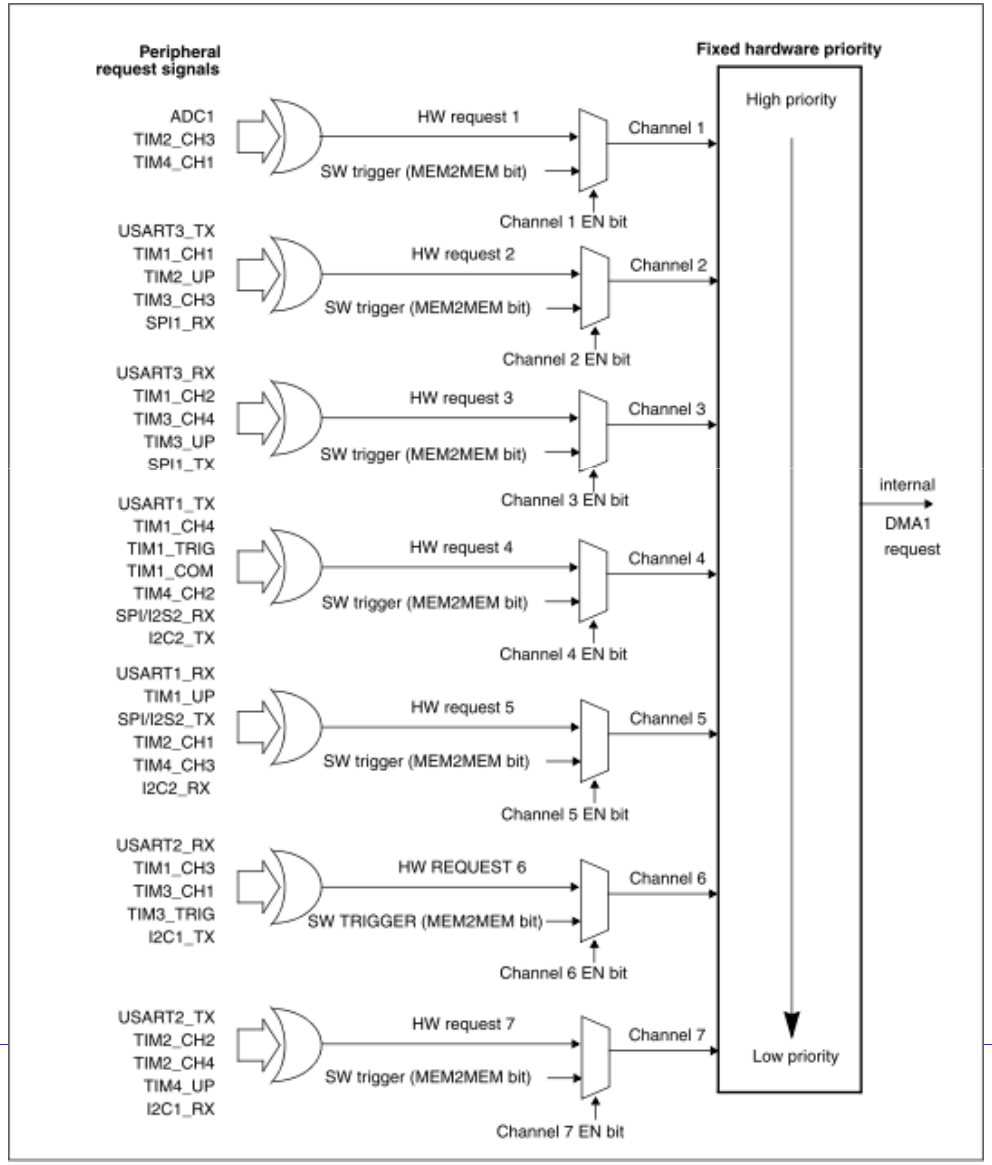
WU-YANG
Technology Co., Ltd.

DMA Channel Mapping

Table 78. Summary of DMA1 requests for each channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
ADC1	ADC1						
SPI/I^2S		SPI1_RX	SPI1_TX	SPI2/I2S2_RX	SPI2/I2S2_TX		
USART		USART3_TX	USART3_RX	USART1_TX	USART1_RX	USART2_RX	USART2_TX
I^2C				I2C2_TX	I2C2_RX	I2C1_TX	I2C1_RX
TIM1		TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	
TIM2	TIM2_CH3	TIM2_UP			TIM2_CH1		TIM2_CH2 TIM2_CH4
TIM3		TIM3_CH3	TIM3_CH4 TIM3_UP			TIM3_CH1 TIM3_TRIG	
TIM4	TIM4_CH1			TIM4_CH2	TIM4_CH3		TIM4_UP

DMA Request Mapping



DMA Interrupt Status Register

10.4.1 DMA interrupt status register (DMA_ISR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				TEIF7	HTIF7	TCIF7	GIF7	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5
				r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:28 Reserved, always read as 0.

Bits 27, 23, 19, 15, 11, 7, 3 **TEIFx:** Channel x transfer error flag (x = 1 ..7)

This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register.

0: No transfer error (TE) on channel x

1: A transfer error (TE) occurred on channel x

Bits 26, 22, 18, 14, 10, 6, 2 **HTIFx:** Channel x half transfer flag (x = 1 ..7)

This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register.

0: No half transfer (HT) event on channel x

1: A half transfer (HT) event occurred on channel x

DMA Interrupt Status Register Cont.,

Bits 25, 21, 17, 13, 9, 5, 1 **TCIFx:** Channel x transfer complete flag (x = 1 ..7)

This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register.

0: No transfer complete (TC) event on channel x

1: A transfer complete (TC) event occurred on channel x

Bits 24, 20, 16, 12, 8, 4, 0 **GIFx:** Channel x global interrupt flag (x = 1 ..7)

This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register.

0: No TE, HT or TC event on channel x

1: A TE, HT or TC event occurred on channel x

DMA Interrupt Flag Clear Register

10.4.2 DMA interrupt flag clear register (DMA_IFCR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				CTEIF 7	CHTIF 7	CTCIF 7	CGIF 7	CTEIF 6	CHTIF 6	CTCIF 6	CGIF 6	CTEIF 5	CHTIF 5	CTCIF 5	CGIF 5
				w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTEIF 4	CHTIF 4	CTCIF 4	CGIF 4	CTEIF 3	CHTIF 3	CTCIF 3	CGIF 3	CTEIF 2	CHTIF 2	CTCIF 2	CGIF 2	CTEIF 1	CHTIF 1	CTCIF 1	CGIF 1
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits 31:28 Reserved, always read as 0.

Bits 27, 23, 19, 15, 11, 7, 3 **CTEIFx:** Channel x transfer error clear (x = 1 ..7)
This bit is set and cleared by software.
0: No effect
1: Clears the corresponding TEIF flag in the DMA_ISR register

Bits 26, 22, 18, 14, 10, 6, 2 **CHTIFx:** Channel x half transfer clear (x = 1 ..7)
This bit is set and cleared by software.
0: No effect
1: Clears the corresponding HTIF flag in the DMA_ISR register

DMA Channel x Configuration Register

10.4.3 DMA channel x configuration register (DMA_CCRx) (x = 1 ..7)

Address offset: 0x08 + 20d × Channel number

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MEM2 MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:15 Reserved, always read as 0.

Bit 14 **MEM2MEM:** Memory to memory mode
This bit is set and cleared by software.
0: Memory to memory mode disabled
1: Memory to memory mode enabled

Bits 13:12 **PL[1:0]:** Channel priority level
These bits are set and cleared by software.
00: Low
01: Medium
10: High
11: Very high

DMA Channel x Configuration Register Cont.,

Bits 11:10 **MSIZE[1:0]:** Memory size
These bits are set and cleared by software.
00: 8-bits
01: 16-bits
10: 32-bits
11: Reserved

Bits 9:8 **PSIZE[1:0]:** Peripheral size
These bits are set and cleared by software.
00: 8-bits
01: 16-bits
10: 32-bits
11: Reserved

Bit 7 **MINC:** Memory increment mode
This bit is set and cleared by software.
0: Memory increment mode disabled
1: Memory increment mode enabled

Bit 6 **PINC:** Peripheral increment mode
This bit is set and cleared by software.
0: Peripheral increment mode disabled
1: Peripheral increment mode enabled

DMA Channel x Configuration Register Cont.,

Bit 5 **CIRC:** Circular mode
This bit is set and cleared by software.
0: Circular mode disabled
1: Circular mode enabled

Bit 4 **DIR:** Data transfer direction
This bit is set and cleared by software.
0: Read from peripheral
1: Read from memory

Bit 3 **TEIE:** Transfer error interrupt enable
This bit is set and cleared by software.
0: TE interrupt disabled
1: TE interrupt enabled

Bit 2 **HTIE:** Half transfer interrupt enable
This bit is set and cleared by software.
0: HT interrupt disabled
1: HT interrupt enabled

Bit 1 **TCIE:** Transfer complete interrupt enable
This bit is set and cleared by software.
0: TC interrupt disabled
1: TC interrupt enabled

Bit 0 **EN:** Channel enable
This bit is set and cleared by software.
0: Channel disabled
1: Channel enabled

DMA Channel x Number of Data Register

10.4.4 DMA channel x number of data register (DMA_CNDTRx) (x = 1 ..7)

Address offset: 0x0C + 20d × Channel number

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDT															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:16 Reserved, always read as 0.

Bits 15:0 **NDT[15:0]:** Number of data to transfer

Number of data to be transferred (0 up to 65535). This register can only be written when the channel is disabled. Once the channel is enabled, this register is read-only, indicating the remaining bytes to be transmitted. This register decrements after each DMA transfer.

Once the transfer is completed, this register can either stay at zero or be reloaded automatically by the value previously programmed if the channel is configured in auto-reload mode.

If this register is zero, no transaction can be served whether the channel is enabled or not.

DMA Channel x Peripheral Address Register

10.4.5 DMA channel x peripheral address register (DMA_CPARx) (x = 1 ..7)

Address offset: 0x10 + dx20 × Channel number

Reset value: 0x0000 0000

This register must *not* be written when the channel is enabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 **PA[31:0]:** Peripheral address

Base address of the peripheral data register from/to which the data will be read/written.

When PSIZE is 01 (16-bit), the PA[0] bit is ignored. Access is automatically aligned to a half-word address.
When PSIZE is 10 (32-bit), PA[1:0] are ignored. Access is automatically aligned to a word address.

DMA Channel x Memory Address Register

10.4.6 DMA channel x memory address register (DMA_CMARx) (x = 1 ..7)

Address offset: 0x14 + dx20 × Channel number

Reset value: 0x0000 0000

This register must *not* be written when the channel is enabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MA																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 **MA[31:0]:** Memory address

Base address of the memory area from/to which the data will be read/written.

When MSIZE is 01 (16-bit), the MA[0] bit is ignored. Access is automatically aligned to a half-word address.
When MSIZE is 10 (32-bit), MA[1:0] are ignored. Access is automatically aligned to a word address.

DMA Register Mapping

10.4.7 DMA register map

The following table gives the DMA register map and the reset values.

Table 59. DMA register map and reset values

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	**DMA_ISR**	Reserved				TEIF7	HTIF7	TCIF7	GIF7	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x004	**DMA_IFCR**	Reserved				CTEIF7	CHTIF7	CTCIF7	CGIF7	CTEIF6	CHTIF6	CTCIF6	CGIF6	CTEIF5	CHTIF5	CTCIF5	CGIF5	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x008	**DMA_CCR1**	Reserved																	MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZ E [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00C	**DMA_CNDTR1**	Reserved																	NDT[15:0]														
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	**DMA_CPAR1**	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x014	**DMA_CMAR1**	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x018	Reserved																																
0x01C	**DMA_CCR2**	Reserved																	MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZ E [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA Register Mapping Cont.,

0x020	**DMA_CNDTR2**	Reserved															NDT[15:0]														
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x024	**DMA_CPAR2**	PA[31:0]																													
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x028	**DMA_CMAR2**	MA[31:0]																													
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x02C	Reserved																														
0x030	**DMA_CCR3**	Reserved															MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZ E [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x034	**DMA_CNDTR3**	Reserved															NDT[15:0]														
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x038	**DMA_CPAR3**	PA[31:0]																													
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x03C	**DMA_CMAR3**	MA[31:0]																													
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x040	Reserved																														
0x044	**DMA_CCR4**	Reserved															MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZ E [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x048	**DMA_CNDTR4**	Reserved															NDT[15:0]														
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA Register Mapping Cont.,

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x04C	**DMA_CPAR4**	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x050	**DMA_CMAR4**	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x054	Reserved																																
0x058	**DMA_CCR5**	Reserved																	MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZ E [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x05C	**DMA_CNDTR5**	Reserved																	NDT[15:0]														
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x060	**DMA_CPAR5**	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x064	**DMA_CMAR5**	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x068	Reserved																																
0x06C	**DMA_CCR6**	Reserved																	MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZ E [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x070	**DMA_CNDTR6**	Reserved																	NDT[15:0]														
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x074	**DMA_CPAR6**	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x078	**DMA_CMAR6**	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x07C	Reserved																																

DMA Register Mapping Cont.,

0x080	DMA_CCR7	Reserved																	MEM2MEM	PL [1:0]		M SIZE [1:0]		PSIZE [1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x084	DMA_CNDTR7	Reserved																NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x088	DMA_CPAR7	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08C	DMA_CMAR7	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x090	Reserved																																

DMA Standard Driver Library

WU-YANG
Technology Co., Ltd.

DMA Standard Driver Library List

Function name	Description
DMA_DeInit	Resets the DMAy Channelx registers to their default reset values.
DMA_Init	Initializes the DMAy Channelx according to the specified parameters in the DMA_InitStruct.
DMA_StructInit	Fills each DMA_InitStruct member with its default value.
DMA_Cmd	Enables or disables the specified DMAy Channelx.
DMA_ITConfig	Enables or disables the specified DMAy Channelx interrupts.
DMA_GetCurrDataCounter	Returns the number of remaining data units in the current DMAy Channelx transfer.
DMA_GetFlagStatus	Checks whether the specified DMAy Channelx flag is set or not.
DMA_ClearFlag	Clears the DMAy Channelx pending flags.
DMA_GetITStatus	Checks whether the specified DMAy Channelx interrupt has occurred or not.
DMA_ClearITPendingBit	Clears the DMAy Channelx interrupt pending bits.

DMA DeInit Function

DMA_DeInit function

Table 106 describes the DMA_DeInit function.

Table 106. DMA_DeInit function

Function name	DMA_DeInit
Function prototype	void DMA_DeInit(DMA_Channel_TypeDef* DMAy_Channelx)
Behavior description	Resets the DMAy Channelx registers to their default reset values.
Input parameter	DMAy_Channelx: where y selects the DMA (y = 1 for DMA1, y = 2 for DMA2) and x selects the DMA Channel (x = 1 to 7 for DMA1 or x = 1 to 5 for DMA2).
Output parameter	None
Return parameter	None
Required preconditions	None
Called functions	RCC_AHBPeriphClockCmd().

Example:

```
/* Deinitialize the DMA1 Channel2 */
DMA_DeInit(DMA1_Channel2);
```

DMA Init Function

DMA_Init function

Table 107 describes the DMA_Init function.

Table 107. DMA_Init function

Function name	DMA_Init
Function prototype	void DMA_Init(DMA_Channel_TypeDef* DMAy_Channelx, DMA_InitTypeDef* DMA_InitStruct)
Behavior description	Initializes the DMAy Channelx according to the parameters specified in the DMA_InitStruct.
Input parameter1	DMAy_Channelx: where y selects the DMA (y = 1 for DMA1, y = 2 for DMA2) and x selects the DMA Channel (x = 1 to 7 for DMA1 or x = 1 to 5 for DMA2).
Input parameter2	DMA_InitStruct: pointer to a DMA_InitTypeDef structure that contains the configuration information for the specified DMAy Channelx. Refer to *DMA_InitTypeDef structure* for more details on the allowed values for this parameter.
Output parameter	None
Return parameter	None
Required preconditions	None
Called functions	None

DMA Init Function Cont.,

DMA_InitTypeDef structure

The DMA_InitTypeDef structure is defined in the *stm32f10x_dma.h* file:

```
typedef struct
{
  u32 DMA_PeripheralBaseAddr;
  u32 DMA_MemoryBaseAddr;
  u32 DMA_DIR;
  u32 DMA_BufferSize;
  u32 DMA_PeripheralInc;
  u32 DMA_MemoryInc;
  u32 DMA_PeripheralDataSize;
  u32 DMA_MemoryDataSize;
  u32 DMA_Mode;
  u32 DMA_Priority;
  u32 DMA_M2M;
} DMA_InitTypeDef;
```

DMA_PeripheralBaseAddr

This member is used to define the peripheral base address for DMAy Channelx.

DMA_MemoryBaseAddr

This member is used to define the memory base address for DMAy Channelx.

DMA_DIR

DMA_DIR specifies if the peripheral is the source or destination. The values taken by this member are given in *Table 108*.

DMA Init Function Cont.,

Table 108. DMA_DIR definition

DMA_DIR	Description
DMA_DIR_PeripheralDST	Peripheral is the destination
DMA_DIR_PeripheralSRC	Peripheral is the source

DMA_BufferSize

DMA_BufferSize is used to define the buffer size, in data unit, of the specified Channel. The data unit is equal to the configuration set in DMA_PeripheralDataSize or DMA_MemoryDataSize members depending in the transfer direction.

DMA_PeripheralInc

DMA_PeripheralInc specifies whether the Peripheral address register is incremented or not. The values taken by this member are given in *Table 109*.

Table 109. DMA_PeripheralInc definition

DMA_PeripheralInc	Description
DMA_PeripheralInc_Enable	Current peripheral register incremented
DMA_PeripheralInc_Disable	Current peripheral register unchanged

DMA Init Function Cont.,

DMA_MemoryInc

DMA_MemoryInc specifies whether the memory address register is incremented or not. The values taken by this member are given in *Table 110*.

Table 110. DMA_MemoryInc definition

DMA_MemoryInc	Description
DMA_MemoryInc_Enable	Current memory register incremented
DMA_MemoryInc_Disable	Current memory register unchanged

DMA_PeripheralDataSize

DMA_PeripheralDataSize configures the Peripheral data width. The values taken by this member are given in *Table 111*.

Table 111. DMA_PeripheralDataSize definition

DMA_PeripheralDataSize	Description
DMA_PeripheralDataSize_Byte	Data width = 8 bits
DMA_PeripheralDataSize_HalfWord	Data width = 16 bits
DMA_PeripheralDataSize_Word	Data width = 32 bits

DMA Init Function Cont.,

DMA_MemoryDataSize

DMA_MemoryDataSize defines the Memory data width. The values taken by this member are given in *Table 112*.

Table 112. DMA_MemoryDataSize definition

DMA_MemoryDataSize	Description
DMA_MemoryDataSize_Byte	Data width = 8 bits
DMA_MemoryDataSize_HalfWord	Data width = 16 bits
DMA_MemoryDataSize_Word	Data width = 32 bits

DMA_Mode

DMA_Mode configures the operation mode of the DMAy Channelx. The values taken by this member are given in *Table 113*.

Table 113. DMA_Mode definition

DMA_Mode	Description
DMA_Mode_Circular	Circular buffer mode is used
DMA_Mode_Normal	Normal buffer mode is used

The circular buffer mode cannot be used if the memory-to-memory data transfer is configured on the selected Channel (see DMA_M2M).

DMA Init Function Cont.,

DMA_Priority

DMA_Priority configures the software priority for the DMAy Channelx. The values taken by this member are given in *Table 114*.

Table 114. DMA_Priority definition

DMA_Priority	Description
DMA_Priority_VeryHigh	DMAy Channelx has a very high priority
DMA_Priority_High	DMAy Channelx has a high priority
DMA_Priority_Medium	DMAy Channelx has a medium priority
DMA_Priority_Low	DMAy Channelx has a low priority

DMA_M2M

DMA_M2M enables the DMAy Channelx memory- to-memory transfer. The values taken by this member are given in *Table 115*.

Table 115. DMA_M2M definition

DMA_M2M	Description
DMA_M2M_Enable	DMAy Channelx configured for memory-to-memory transfer
DMA_M2M_Disable	DMAy Channelx not configured for memory-to-memory transfer

DMA Init Function Cont.,

Example:

```
/* Initialize the DMA1 Channel1 according to the DMA_InitStructure
members */
  DMA_InitTypeDef  DMA_InitStructure;

  DMA_InitStructure.DMA_PeripheralBaseAddr = 0x40005400;
  DMA_InitStructure.DMA_MemoryBaseAddr = 0x20000100;
  DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralSRC;
  DMA_InitStructure.DMA_BufferSize = 256;
  DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
  DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
  DMA_InitStructure.DMA_PeripheralDataSize =
  DMA_PeripheralDataSize_HalfWord;
  DMA_InitStructure.DMA_MemoryDataSize =
  DMA_MemoryDataSize_HalfWord;
  DMA_InitStructure.DMA_Mode = DMA_Mode_Normal;
  DMA_InitStructure.DMA_Priority = DMA_Priority_Medium;
  DMA_InitStructure.DMA_M2M = DMA_M2M_Disable;
  DMA_Init(DMA1_Channel1, &DMA_InitStructure);
```

DMA Command Function

DMA_Cmd function

Table 118 describes DMA_Cmd function.

Table 118. DMA_Cmd function

Function name	DMA_Cmd
Function prototype	void DMA_Cmd(DMA_Channel_TypeDef* DMAy_Channelx, FunctionalState NewState)
Behavior description	Enables or disables the specified DMAy Channelx.
Input parameter1	DMAy_Channel*x*: where y selects the DMA (y = 1 for DMA1, y = 2 for DMA2) and x selects the DMA Channel (x = 1 to 7 for DMA1 or x = 1 to 5 for DMA2).
Input parameter2	NewState: new state of the DMAy Channelx. This parameter can be: ENABLE or DISABLE.
Output parameter	None
Return parameter	None
Required preconditions	None
Called functions	None

Example:

```
/* Enable DMA1 Channel7 */
DMA_Cmd(DMA1_Channel7, ENABLE);
```

DMA Interrupt Function

DMA_ITConfig function

Table 119 describes DMA_ITConfig function.

Table 119. DMA_ITConfig function

Function name	DMA_ITConfig
Function prototype	void DMA_ITConfig(DMA_Channel_TypeDef* DMAy_Channelx, u32 DMA_IT, FunctionalState NewState)
Behavior description	Enables or disables the specified DMAy Channelx interrupts.
Input parameter1	DMAy_Channelx: where y selects the DMA (y = 1 for DMA1, y = 2 for DMA2) and x selects the DMA Channel (x = 1 to 7 for DMA1 or x = 1 to 5 for DMA2).
Input parameter2	DMA_IT: specifies the DMAy Channelx interrupt sources to be enabled or disabled. More than one interrupt can be selected using the "|" operator. Refer to *DMA_IT* for more details on the allowed values for this parameter.
Input parameter3	NewState: new state of the specified DMAy Channelx interrupts. This parameter can be: ENABLE or DISABLE.
Output parameter	None
Return parameter	None
Required preconditions	None
Called functions	None

DMA Interrupt Function Cont.,

DMA_IT

The DMA_IT input parameter enables or disables DMAy Channelx interrupts. One or a combination of the following values can be used.

Table 120. DMA_IT values

DMA_IT	Description
DMA_IT_TC	Transfer complete interrupt mask
DMA_IT_HT	Half transfer interrupt mask
DMA_IT_TE	Transfer error interrupt mask

Example:

```
/* Enable DMA1 Channel5 complete transfer interrupt */
DMA_ITConfig(DMA1_Channel5, DMA_IT_TC, ENABLE);
```

DMA Get CurrDataCounter Function

DMA_GetCurrDataCounter function

Table 121 describes DMA_GetCurrDataCounter function.

Table 121. DMA_GetCurrDataCounter function

Function name	DMA_GetCurrDataCounter
Function prototype	u16 DMA_GetCurrDataCounter(DMA_Channel_TypeDef* DMAy_Channelx)
Behavior description	Returns the number of remaining data units in the current DMAy Channelx transfer.
Input parameter	DMAy_Channel*x*: where y selects the DMA (y = 1 for DMA1, y = 2 for DMA2) and x selects the DMA Channel (x = 1 to 7 for DMA1 or x = 1 to 5 for DMA2).
Output parameter	None
Return parameter	The number of remaining data units in the current DMAy Channelx transfer.
Required preconditions	None
Called functions	None

Example:

```
/* Get  the number of remaining data units in the current DMA1
Channel2 transfer */
u16 CurrDataCount;
CurrDataCount = DMA_GetCurrDataCounter(DMA1_Channel2);
```

DMA Get Flag Status Function

DMA_GetFlagStatus function

Table 122 describes DMA_GetFlagStatus function.

Table 122. DMA_GetFlagStatus function

Function name	DMA_GetFlagStatus
Function prototype	FlagStatus DMA_GetFlagStatus(u32 DMA_FLAG)
Behavior description	Checks whether the specified DMAy Channelx flag is set or not.
Input parameter	DMA_FLAG: specifies the flag to check. Refer to *DMA_FLAG* for more details on the allowed values for this parameter.
Output parameter	None
Return parameter	New state of DMA_FLAG (SET or RESET).
Required preconditions	None
Called functions	None

DMA_FLAG

The DMA_FLAG is used to define the type of flag that will be checked. See *Table 123* for a description of this input parameter.

DMA Get Flag Status Function Cont.,

Table 123. DMA_FLAG definition

DMA_FLAG	Description
DMA1_FLAG_GL1	DMA1 Channel1 global flag
DMA1_FLAG_TC1	DMA1 Channel1 transfer complete flag
DMA1_FLAG_HT1	DMA1 Channel1 half transfer flag
DMA1_FLAG_TE1	DMA1 Channel1 transfer error flag
DMA1_FLAG_GL2	DMA1 Channel2 global flag
DMA1_FLAG_TC2	DMA1 Channel2 transfer complete flag
DMA1_FLAG_HT2	DMA1 Channel2 half transfer flag
DMA1_FLAG_TE2	DMA1 Channel2 transfer error flag
DMA1_FLAG_GL3	DMA1 Channel3 global flag
DMA1_FLAG_TC3	DMA1 Channel3 transfer complete flag
DMA1_FLAG_HT3	DMA1 Channel3 half transfer flag
DMA1_FLAG_TE3	DMA1 Channel3 transfer error flag
DMA1_FLAG_GL4	DMA1 Channel4 global flag
DMA1_FLAG_TC4	DMA1 Channel4 transfer complete flag
DMA1_FLAG_HT4	DMA1 Channel4 half transfer flag
DMA1_FLAG_TE4	DMA1 Channel4 transfer error flag
DMA1_FLAG_GL5	DMA1 Channel5 global flag
DMA1_FLAG_TC5	DMA1 Channel5 transfer complete flag

DMA Get Flag Status Function Cont.,

Table 123. DMA_FLAG definition (continued)

DMA_FLAG	Description
DMA1_FLAG_HT5	DMA1 Channel5 half transfer flag
DMA1_FLAG_TE5	DMA1 Channel5 transfer error flag
DMA1_FLAG_GL6	DMA1 Channel6 global flag
DMA1_FLAG_TC6	DMA1 Channel6 transfer complete flag
DMA1_FLAG_HT6	DMA1 Channel6 half transfer flag
DMA1_FLAG_TE6	DMA1 Channel6 transfer error flag
DMA1_FLAG_GL7	DMA1 Channel7 global flag
DMA1_FLAG_TC7	DMA1 Channel7 transfer complete flag
DMA1_FLAG_HT7	DMA1 Channel7 half transfer flag
DMA1_FLAG_TE7	DMA1 Channel7 transfer error flag

DMA Get Flag Status Function Cont.,

DMA2_FLAG_GL1	DMA2 Channel1 global flag
DMA2_FLAG_TC1	DMA2 Channel1 transfer complete flag
DMA2_FLAG_HT1	DMA2 Channel1 half transfer flag
DMA2_FLAG_TE1	DMA2 Channel1 transfer error flag
DMA2_FLAG_GL2	DMA2 Channel2 global flag
DMA2_FLAG_TC2	DMA2 Channel2 transfer complete flag
DMA2_FLAG_HT2	DMA2 Channel2 half transfer flag
DMA2_FLAG_TE2	DMA2 Channel2 transfer error flag
DMA2_FLAG_GL3	DMA2 Channel3 global flag
DMA2_FLAG_TC3	DMA2 Channel3 transfer complete flag
DMA2_FLAG_HT3	DMA2 Channel3 half transfer flag
DMA2_FLAG_TE3	DMA2 Channel3 transfer error flag
DMA2_FLAG_GL4	DMA2 Channel4 global flag
DMA2_FLAG_TC4	DMA2 Channel4 transfer complete flag
DMA2_FLAG_HT4	DMA2 Channel4 half transfer flag
DMA2_FLAG_TE4	DMA2 Channel4 transfer error flag
DMA2_FLAG_GL5	DMA2 Channel5 global flag
DMA2_FLAG_TC5	DMA2 Channel5 transfer complete flag
DMA2_FLAG_HT5	DMA2 Channel5 half transfer flag
DMA2_FLAG_TE5	DMA2 Channel5 transfer error flag

DMA Get Flag Status Function Cont.,

Example:

```
/* Test if the DMA1 Channel6 half transfer interrupt flag is set or
not */
FlagStatus Status;
Status = DMA_GetFlagStatus(DMA1_FLAG_HT6);
```

DMA Clear Flag Function

DMA_ClearFlag function

Table 124 describes DMA_ClearFlag function.

Table 124. DMA_ClearFlag function

Function name	DMA_ClearFlag
Function prototype	void DMA_ClearFlag(u32 DMA_FLAG)
Behavior description	Clears the DMAy Channelx's pending flags.
Input parameter	DMA_FLAG: flag to be cleared. More than one flag can be cleared using the "|" operator. Refer to *DMA_FLAG* for more details on the allowed values for this parameter. The user can select more than one flag, by 'ORing' them.
Output parameter	None
Return parameter	None
Required preconditions	None
Called functions	None

Example:

```
/* Clear the DMA1 Channel3 transfer error interrupt pending bit */
DMA_ClearFlag(DMA1_FLAG_TE3);
```

DMA Get Interrupt Status Function

DMA_GetITStatus function

Table 125 describes DMA_GetITStatus function.

Table 125. DMA_GetITStatus function

Function name	DMA_GetITStatus
Function prototype	ITStatus DMA_GetITStatus(u32 DMA_IT)
Behavior description	Checks whether the specified DMAy Channelx interrupt has occurred or not.
Input parameter	DMA_IT: DMAy Channelx interrupt source to check. Refer to *DMA_IT* for more details on the allowed values for this parameter.
Output parameter	None
Return parameter	The new state of DMA_IT (SET or RESET).
Required preconditions	None
Called functions	None

DMA_IT

The DMA_IT selects the interrupt that will be checked. See *Table 126* for a description of this input parameter.

DMA Get Interrupt Status Function Cont.,

Table 126. DMA_IT values

DMA_IT	Description
DMA1_IT_GL1	DMA1 Channel1 global interrupt
DMA1_IT_TC1	DMA1 Channel1 transfer complete interrupt
DMA1_IT_HT1	DMA1 Channel1 half transfer interrupt
DMA1_IT_TE1	DMA1 Channel1 transfer error interrupt
DMA1_IT_GL2	DMA1 Channel2 global interrupt
DMA1_IT_TC2	DMA1 Channel2 transfer complete interrupt
DMA1_IT_HT2	DMA1 Channel2 half transfer interrupt
DMA1_IT_TE2	DMA1 Channel2 transfer error interrupt
DMA1_IT_GL3	DMA1 Channel3 global interrupt
DMA1_IT_TC3	DMA1 Channel3 transfer complete interrupt
DMA1_IT_HT3	DMA1 Channel3 half transfer interrupt
DMA1_IT_TE3	DMA1 Channel3 transfer error interrupt
DMA1_IT_GL4	DMA1 Channel4 global interrupt
DMA1_IT_TC4	DMA1 Channel4 transfer complete interrupt
DMA1_IT_HT4	DMA1 Channel4 half transfer interrupt
DMA1_IT_TE4	DMA1 Channel4 transfer error interrupt
DMA1_IT_GL5	DMA1 Channel5 global interrupt
DMA1_IT_TC5	DMA1 Channel5 transfer complete interrupt

DMA Get Interrupt Status Function Cont.,

Table 126. DMA_IT values (continued)

DMA_IT	Description
DMA1_IT_HT5	DMA1 Channel5 half transfer interrupt
DMA1_IT_TE5	DMA1 Channel5 transfer error interrupt
DMA1_IT_GL6	DMA1 Channel6 global interrupt
DMA1_IT_TC6	DMA1 Channel6 transfer complete interrupt
DMA1_IT_HT6	DMA1 Channel6 half transfer interrupt
DMA1_IT_TE6	DMA1 Channel6 transfer error interrupt
DMA1_IT_GL7	DMA1 Channel7 global interrupt
DMA1_IT_TC7	DMA1 Channel7 transfer complete interrupt
DMA1_IT_HT7	DMA1 Channel7 half transfer interrupt
DMA1_IT_TE7	DMA1 Channel7 transfer error interrupt
DMA2_IT_GL1	DMA2 Channel1 global interrupt
DMA2_IT_TC1	DMA2 Channel1 transfer complete interrupt
DMA2_IT_HT1	DMA2 Channel1 half transfer interrupt
DMA2_IT_TE1	DMA2 Channel1 transfer error interrupt
DMA2_IT_GL2	DMA2 Channel2 global interrupt
DMA2_IT_TC2	DMA2 Channel2 transfer complete interrupt
DMA2_IT_HT2	DMA2 Channel2 half transfer interrupt
DMA2_IT_TE2	DMA2 Channel2 transfer error interrupt

DMA Get Interrupt Status Function Cont.,

DMA2_IT_GL2	DMA2 Channel2 global interrupt
DMA2_IT_TC2	DMA2 Channel2 transfer complete interrupt
DMA2_IT_HT2	DMA2 Channel2 half transfer interrupt
DMA2_IT_TE2	DMA2 Channel2 transfer error interrupt
DMA2_IT_GL3	DMA2 Channel3 global interrupt
DMA2_IT_TC3	DMA2 Channel3 transfer complete interrupt
DMA2_IT_HT3	DMA2 Channel3 half transfer interrupt
DMA2_IT_TE3	DMA2 Channel3 transfer error interrupt
DMA2_IT_GL4	DMA2 Channel4 global interrupt
DMA2_IT_TC4	DMA2 Channel4 transfer complete interrupt
DMA2_IT_HT4	DMA2 Channel4 half transfer interrupt
DMA2_IT_TE4	DMA2 Channel4 transfer error interrupt
DMA2_IT_GL5	DMA2 Channel5 global interrupt
DMA2_IT_TC5	DMA2 Channel5 transfer complete interrupt
DMA2_IT_HT5	DMA2 Channel5 half transfer interrupt
DMA2_IT_TE5	DMA2 Channel5 transfer error interrupt

Example:

```
/* Test if the DMA1 Channel7 transfer complete interrupt has
occurred or not */
ITStatus Status;
Status = DMA_GetITStatus(DMA1_IT_TC7);
```

DMA Clear Interrupt Pending Bit Function

DMA_ClearITPendingBit function

Table 127 describes DMA_ClearITPendingBit function.

Table 127. DMA_ClearITPendingBit function

Function name	DMA_ClearITPending Bit
Function prototype	void DMA_ClearITPendingBit(u32 DMA_IT)
Behavior description	Clears the DMAy Channelx's interrupt pending bits.
Input parameter	DMA_IT: DMAy Channelx interrupt pending bit to clear.More than one interrupt can be cleared using the "|" operator. Refer to *DMA_IT* for more details on the allowed values for this parameter.
Output parameter	None
Return parameter	None
Required preconditions	None
Called functions	None

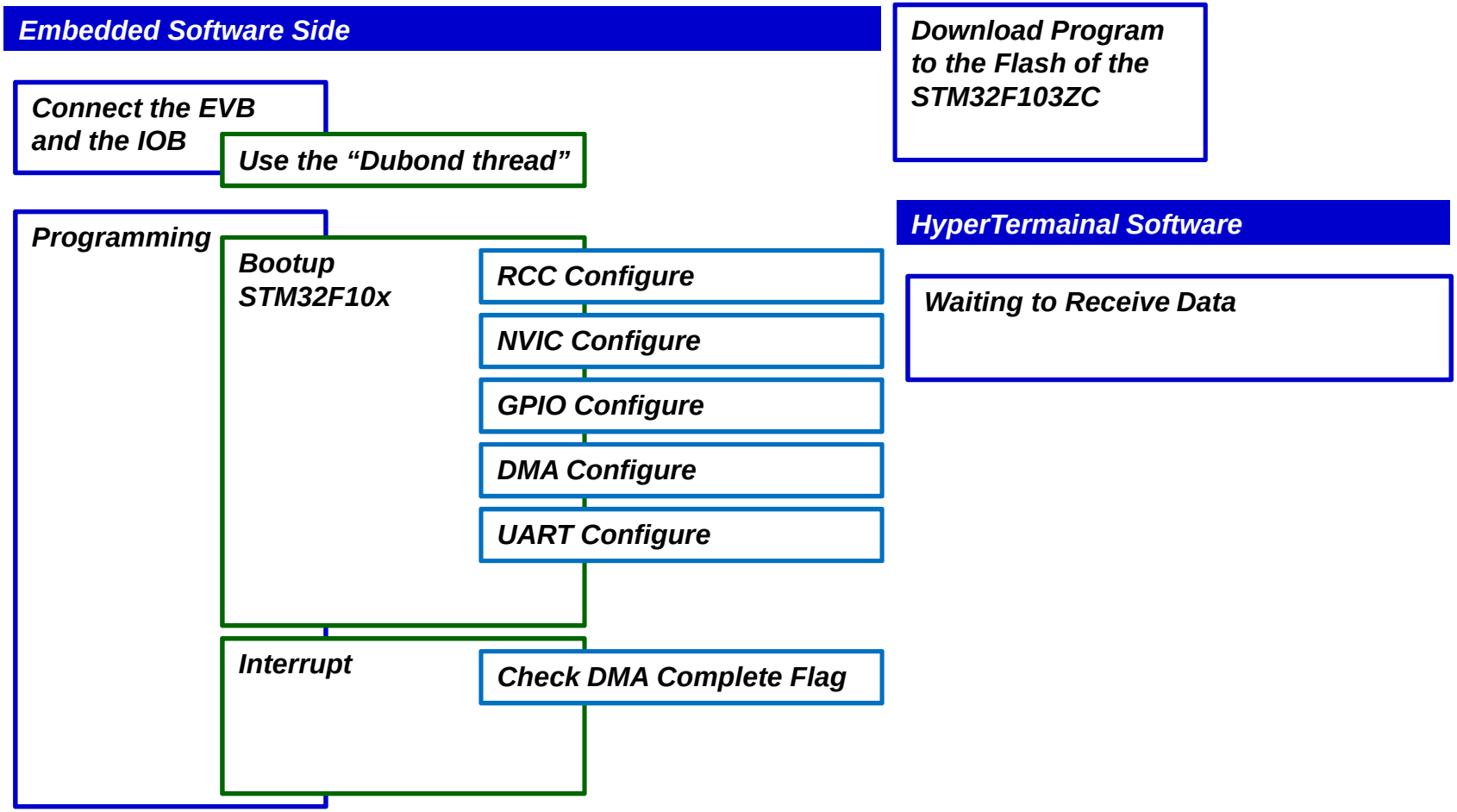
Example:

```
/* Clear the DMA1 Channel5 global interrupt pending bit */
DMA_ClearITPendingBit(DMA1_IT_GL5);
```

實驗1 – Polling DMA

- Goal
  - Use DMA to copy data from memory to memory
- Principle
  - Check the DMA flag

Development Flow



RCC Configure

/* RCC Configure */

```
/* DMA1 clock enable */
    RCC_AHBPeriphClockCmd(RCC_AHBPeriph_DMA1, ENABLE);
```

Turn On DMA1 Clock

DMA Configure

```
/* DMA Configure */
void DMA_Configuration(void)
{
        DMA_InitTypeDef         DMA_InitStructure;

        /* DMA1 channel6 configuration */
        DMA_DeInit(DMA1_Channel6);
        DMA_InitStructure.DMA_PeripheralBaseAddr = (uint32_t)SRC_Const_Buffer;
        DMA_InitStructure.DMA_MemoryBaseAddr = (uint32_t)DST_Buffer;
        DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralSRC;
        DMA_InitStructure.DMA_BufferSize = BufferSize;
        DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Enable;
        DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
        DMA_InitStructure.DMA_PeripheralDataSize = DMA_PeripheralDataSize_Word;
        DMA_InitStructure.DMA_MemoryDataSize = DMA_MemoryDataSize_Word;
        DMA_InitStructure.DMA_Mode = DMA_Mode_Normal;
        DMA_InitStructure.DMA_Priority = DMA_Priority_High;
        DMA_InitStructure.DMA_M2M = DMA_M2M_Enable;
        DMA_Init(DMA1_Channel6, &DMA_InitStructure);
}
```

User Program

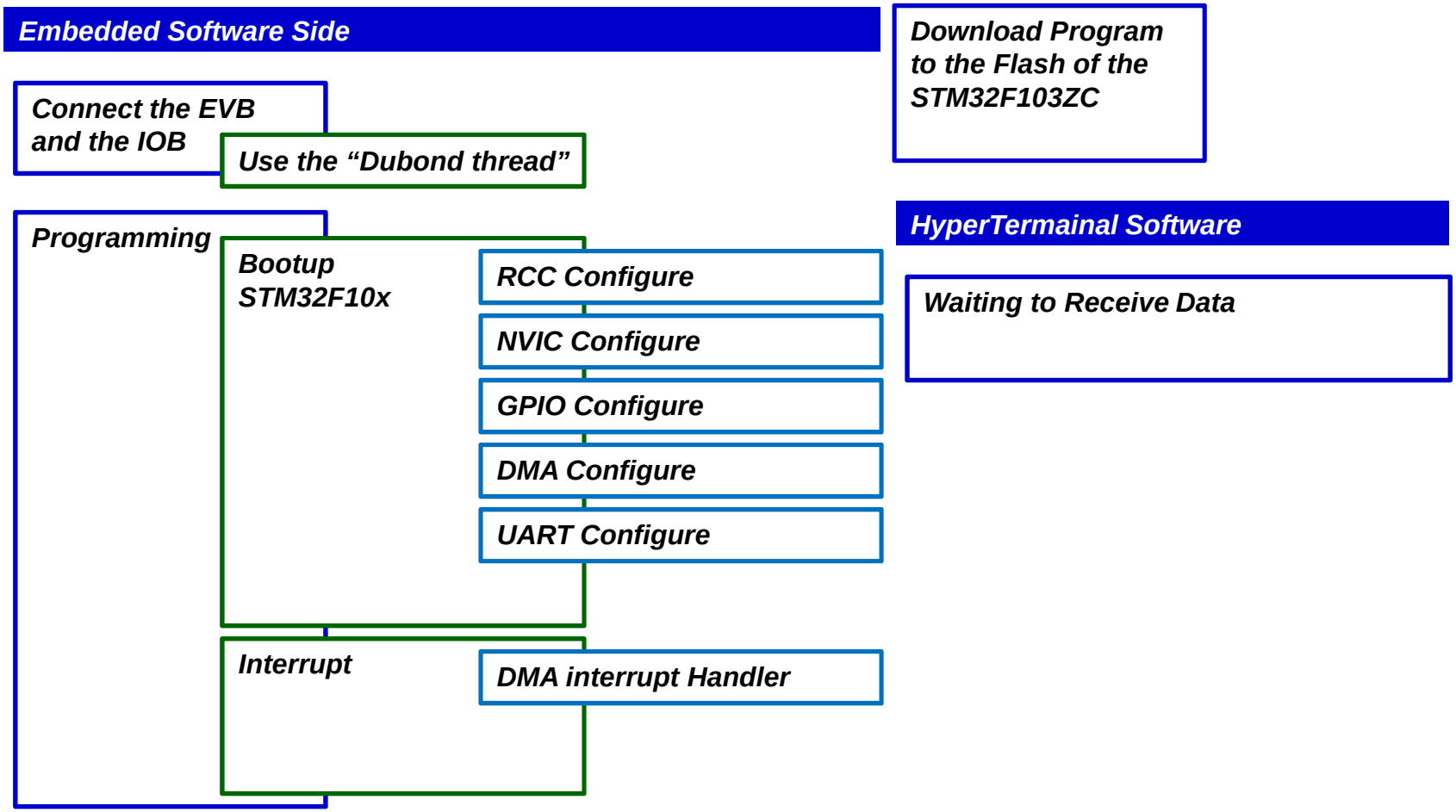
```
/* User Program */
    while (DMA_GetFlagStatus(DMA1_FLAG_TC5) == RESET);

    DMA_ClearFlag(DMA1_FLAG_TC5);
    DMA_Configuration();
    //DMA_Cmd(DMA1_Channel5, ENABLE);
```

實驗2 – Interrupt DMA

- Goal
  - Use DMA to copy data from memory to memory
- Principle
  - Use the DMA Interrupt

Development Flow



Q & A