ITMO UNIVERSITY

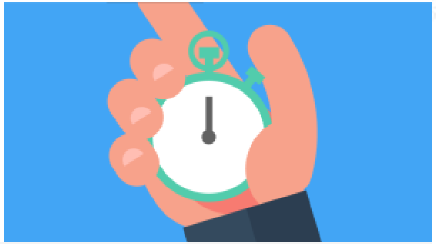
Saint Petersburg, Russia

Аппаратное обеспечение робототехнических систем

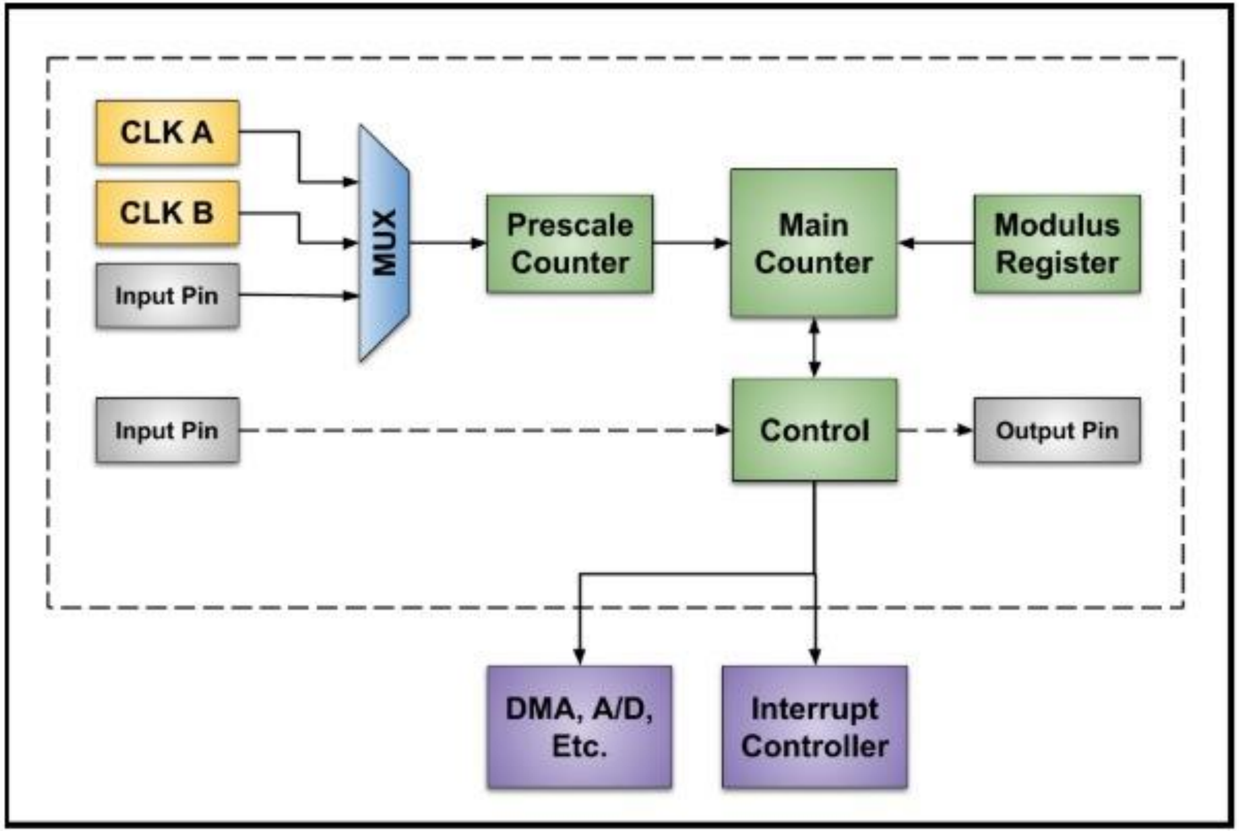
Vlasov Sergey

smvlasov@itmo.ru

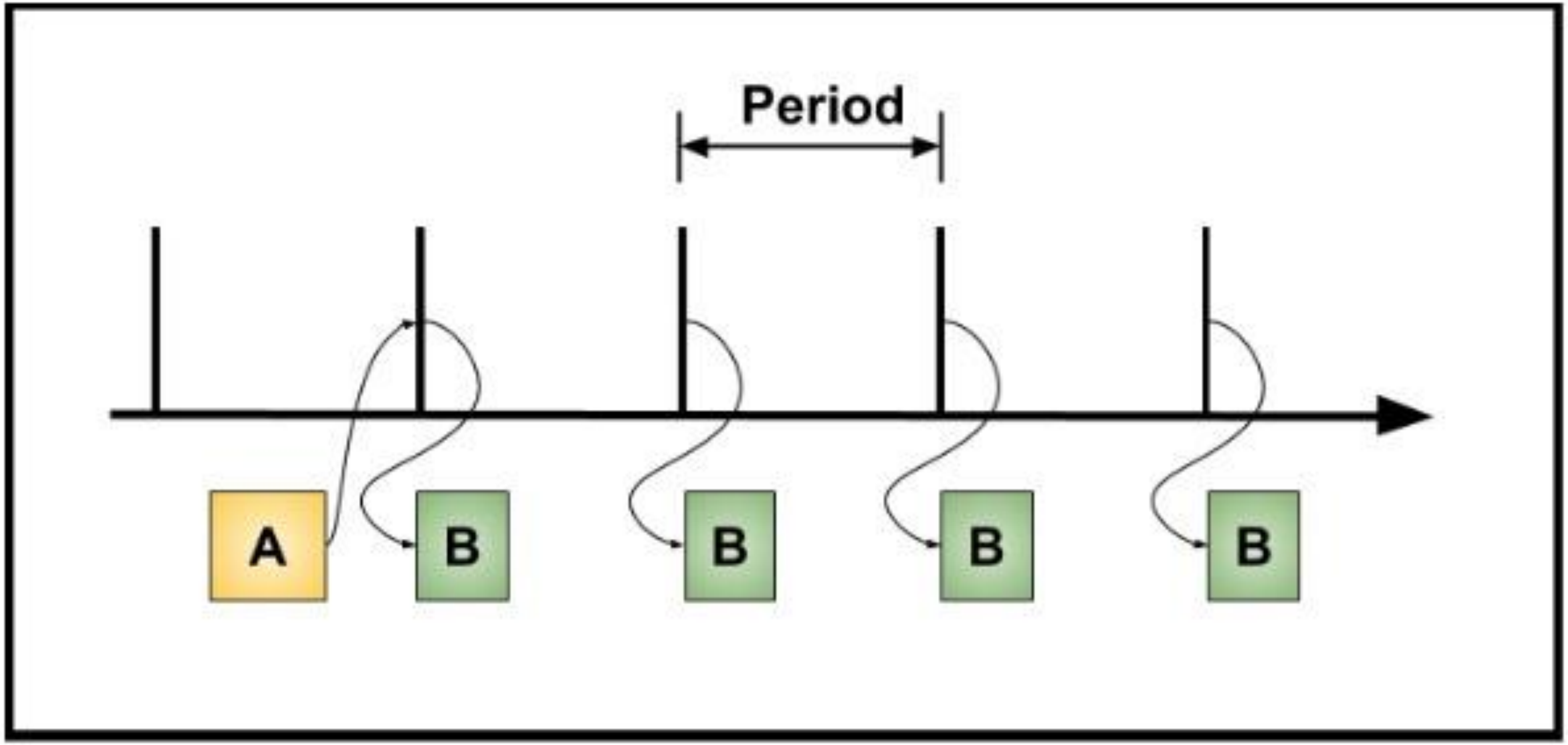
Timer



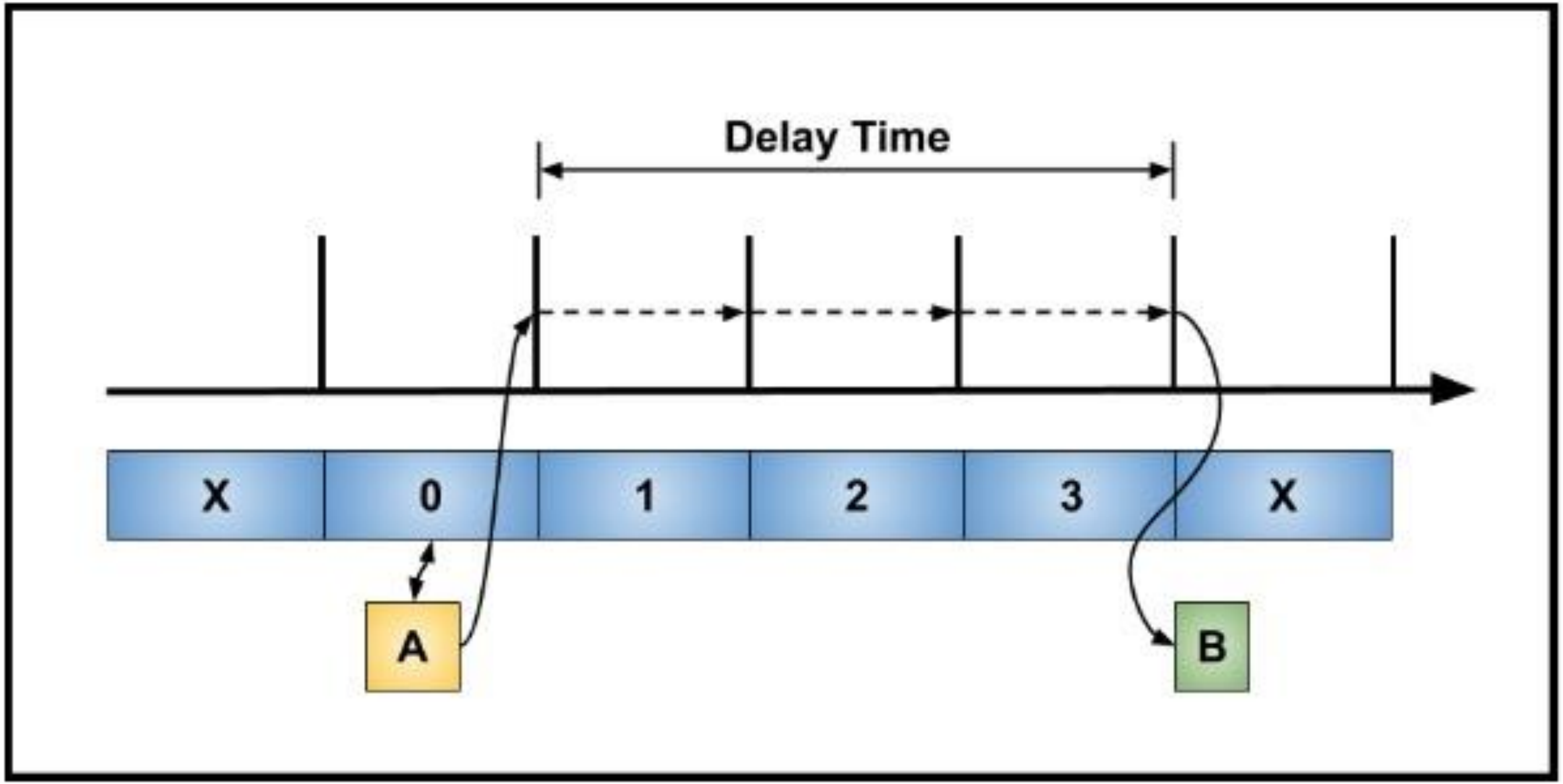
```
999         (M = 999)

998

997

.

.

2

1

0           (Reload M when counter reaches 0)

999

998

.

.
```



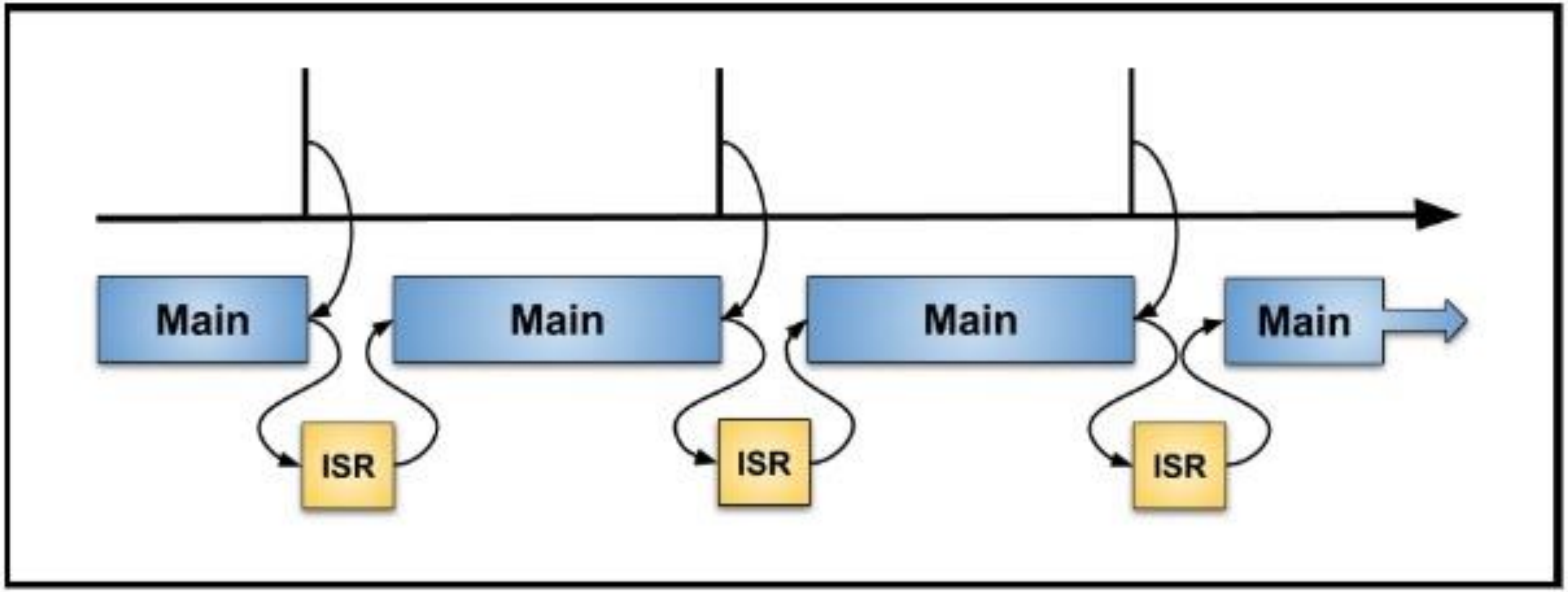
Periodic Timers



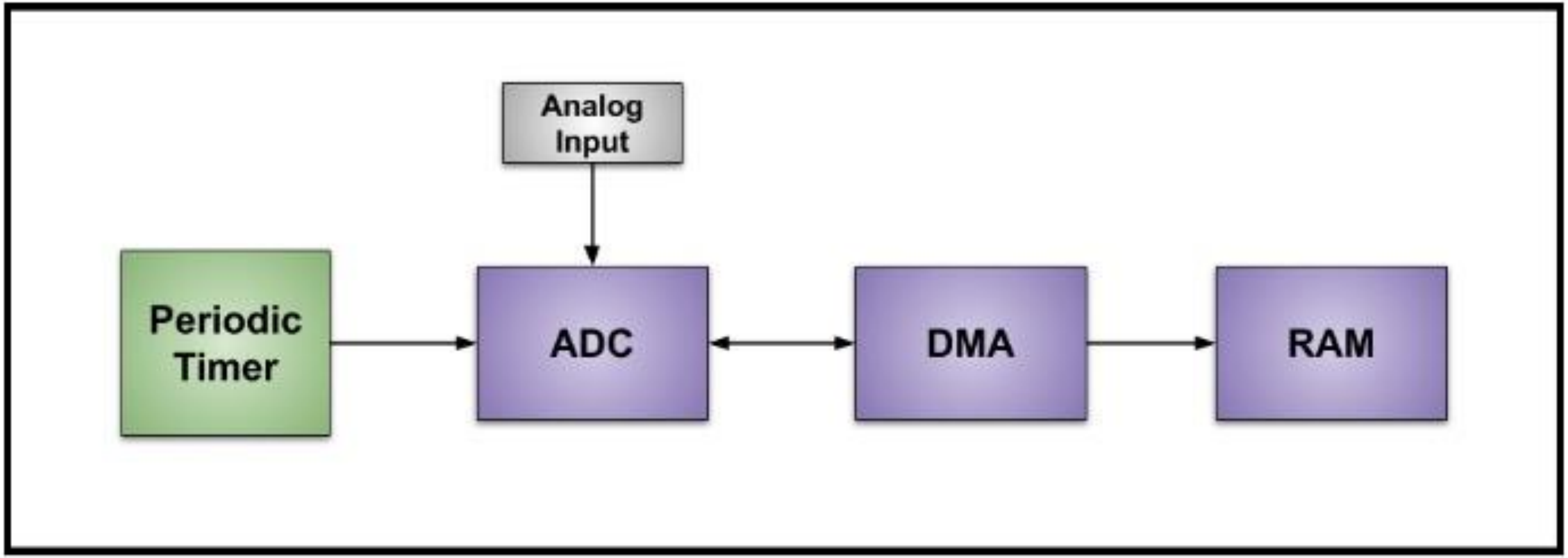
Delay



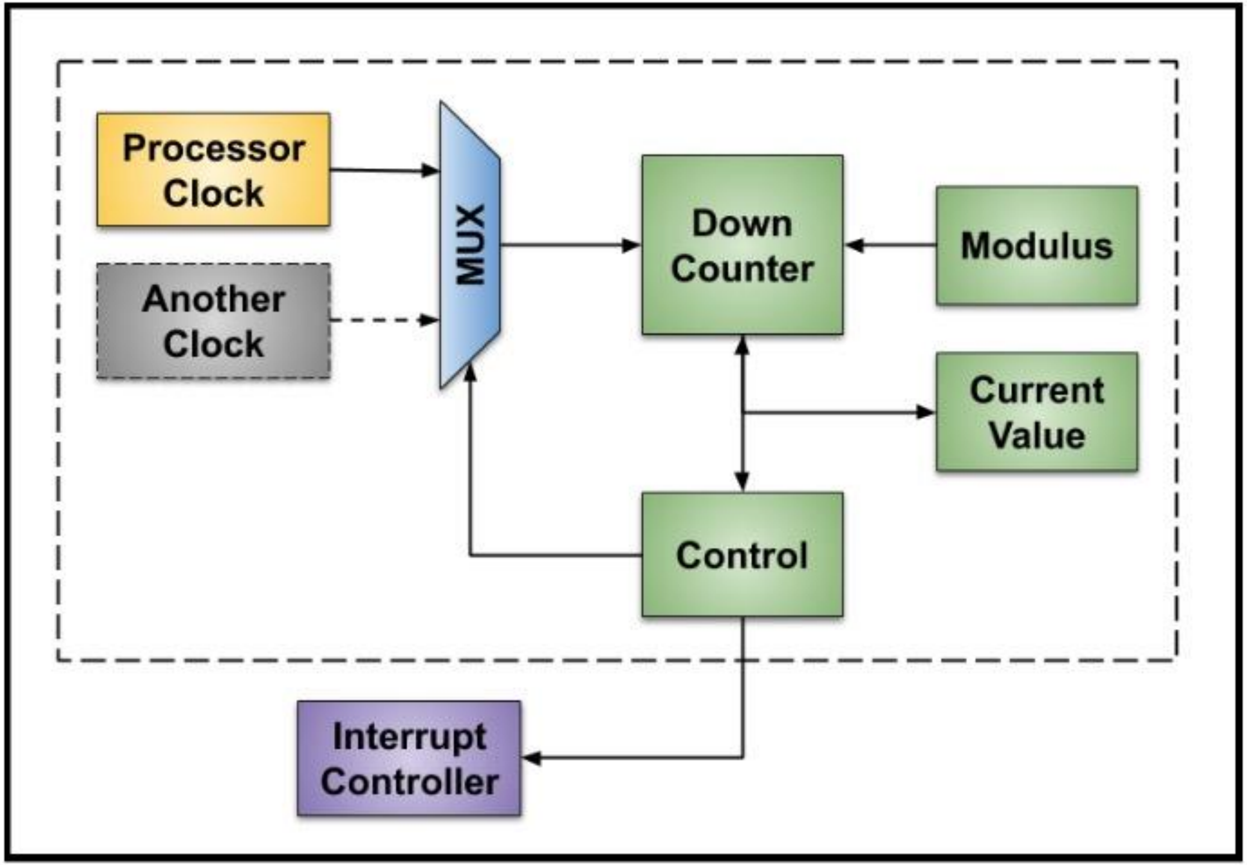
Polling and Interrupts



Triggering

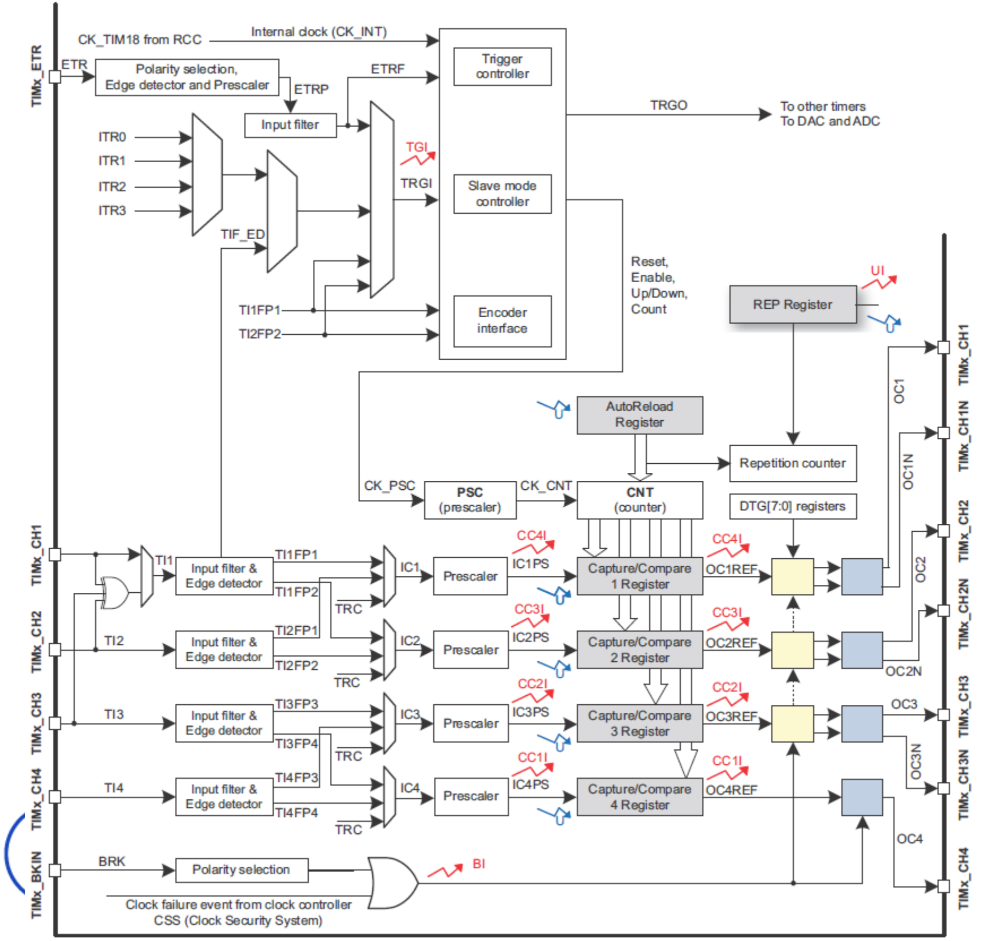


Detailed Description of a Simple, Periodic Timer



General-purpose timers

ITMO UNIVERSITY



Interrupt & DMA output

Event

🟨 DTG

🟦 Output control

Reg Preload registers transferred to active registers on U event according to control bit

Timer’s registers

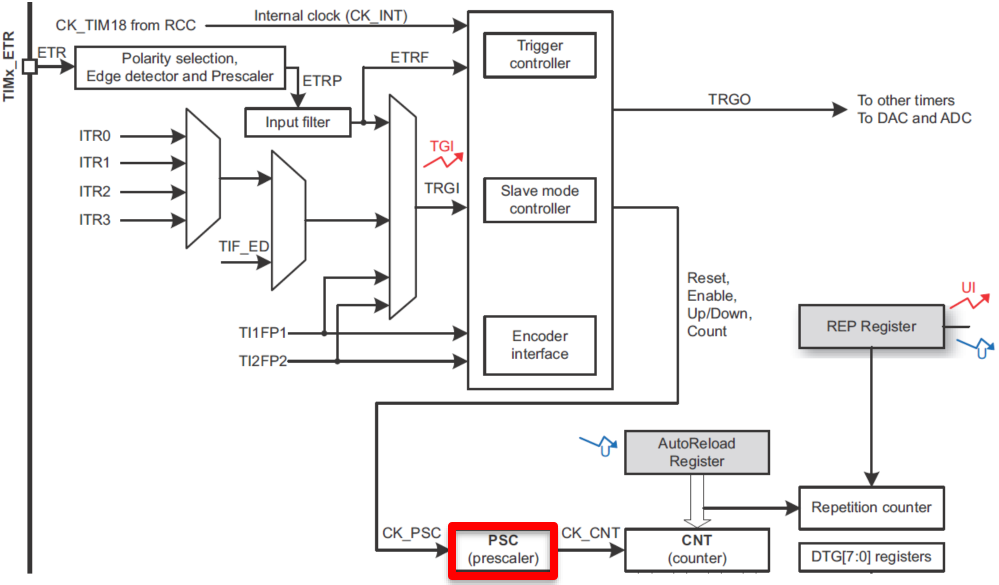
TIM control register 1 (TIMx_CR1)
TIM control register 2 (TIMx_CR2)
TIM slave mode control register (TIMx_SMCR)
TIM DMA/interrupt enable register (TIMx_DIER)
TIM status register (TIMx_SR)
TIM event generation register (TIMx_EGR)
TIM capture/compare mode register 1/2 (TIMx_CCMR1/2)
TIM capture/compare enable register (TIMx_CCER)
TIM counter (TIMx_CNT)
TIM prescaler (TIMx_PSC)
TIM auto-reload register (TIMx_ARR)
TIM repetition counter register (TIMx_RCR)
TIM capture/compare register 1-4 (TIMx_CCR1-4)
TIM break and dead-time register (TIMx_BDTR)
TIM DMA control register (TIMx_DCR)
TIM DMA address for full transfer (TIMx_DMAR)

Timer’s registers

TIM prescaler (TIMx_PSC)

Address offset: 0x28

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw



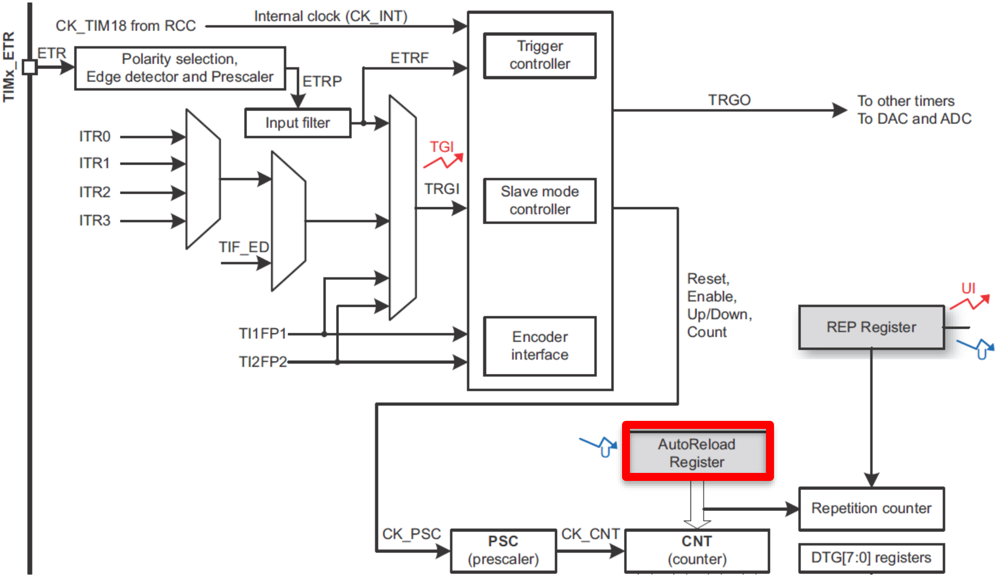
$$f_{CK\_CNT} = \frac{f_{CK\_PSC}}{(PSC[15:0] + 1)}$$

Timer’s registers

TIM auto-reload register (TIMx_ARR)

Address offset: 0x2C

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

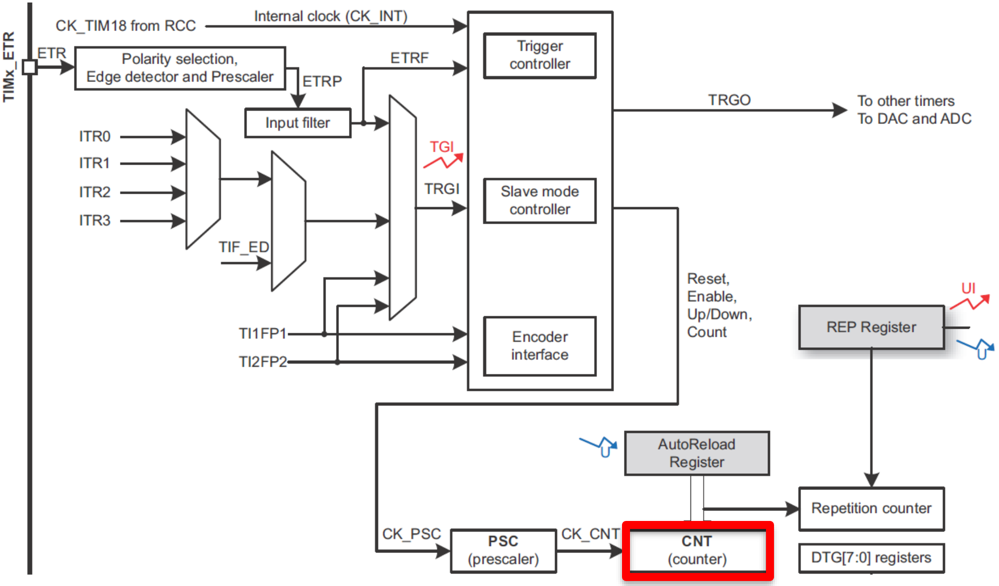


Timer’s registers

TIM counter (TIMx_CNT)

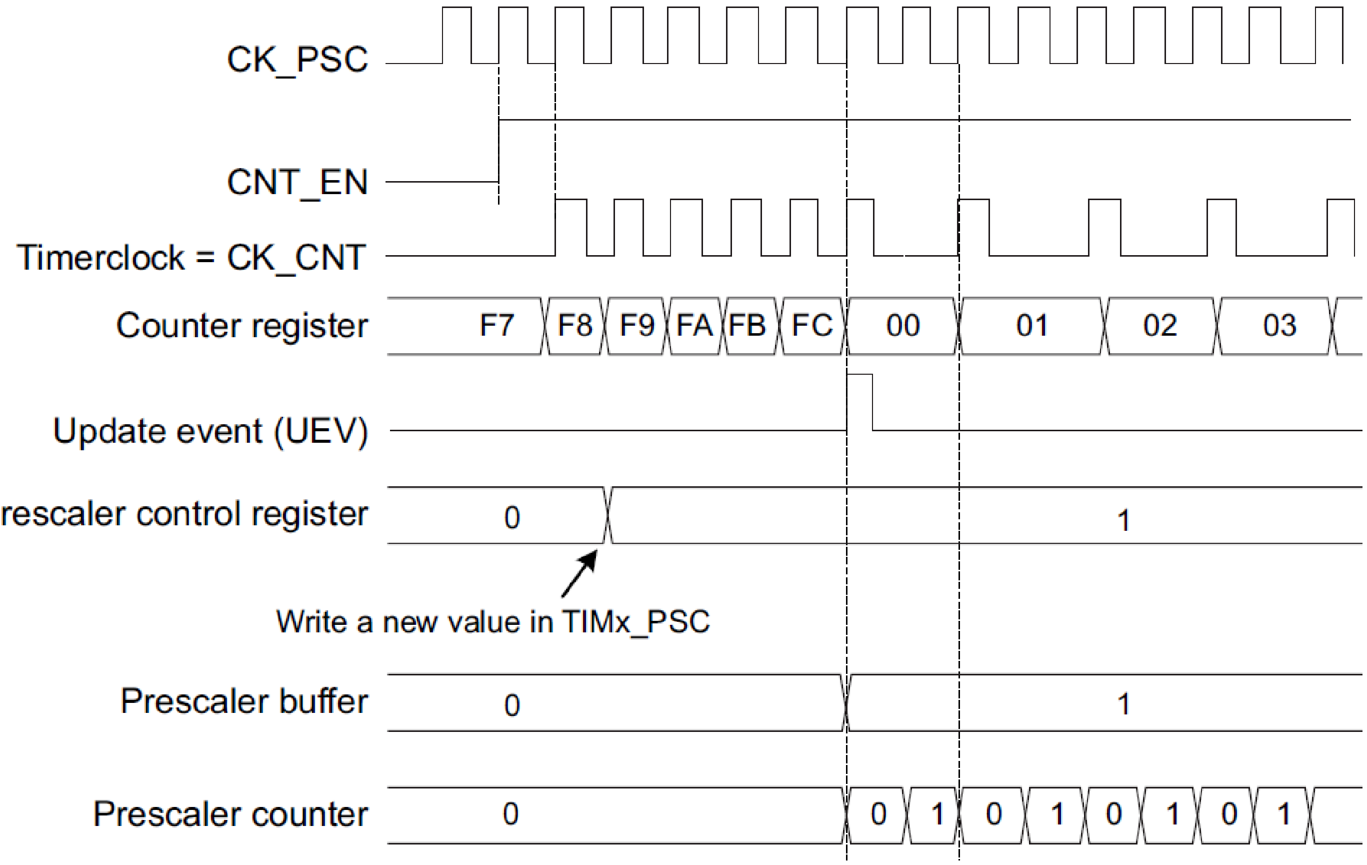
Address offset: 0x24

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw



Prescaler description

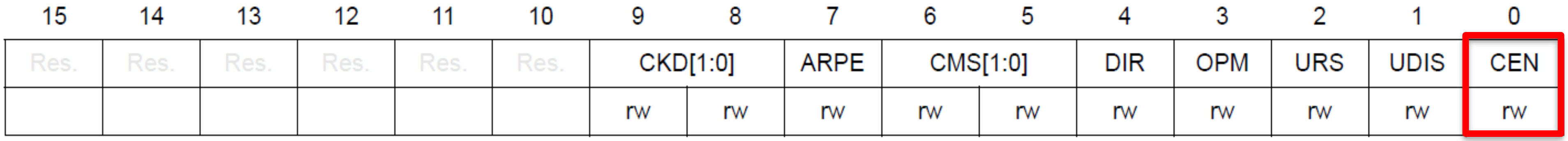
ARR=0xFC



Timer’s registers

TIM control register 1 (TIMx_CR1)

Address offset: 0x00



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

CEN: Counter enable

0: Counter disabled

1: Counter enabled

OPM: One pulse mode

0: Counter is not stopped at update event

1: Counter stops counting at the next update event (clearing the bit CEN)

DIR: Direction

0: Counter used as upcounter

1: Counter used as downcounter

ARPE: Auto-reload preload enable

0: TIMx_ARR register is not buffered

1: TIMx_ARR register is buffered

CMS: Center-aligned mode selection

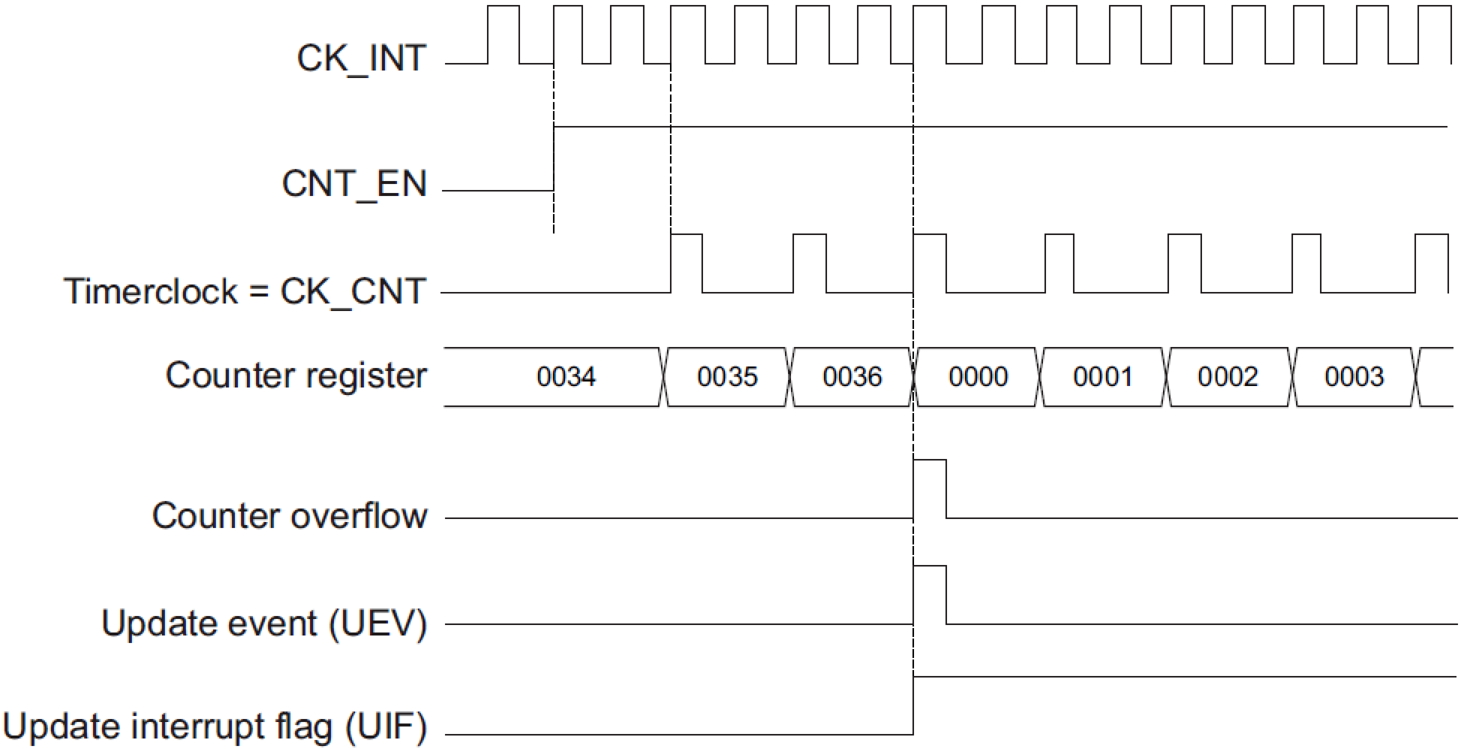
00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR).

01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting down.

10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting up.

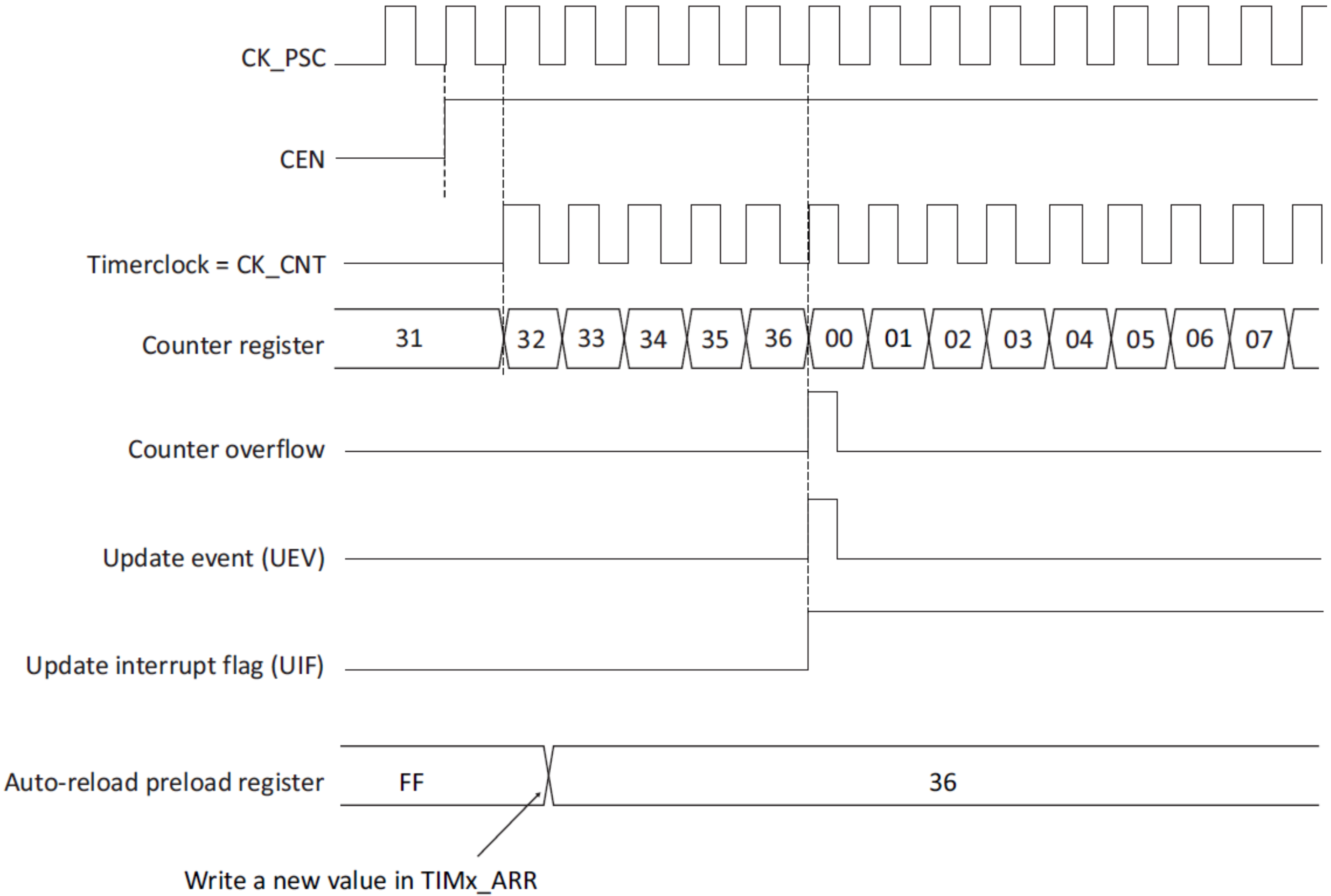
11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set both when the counter is counting up or down.

Upcounting mode



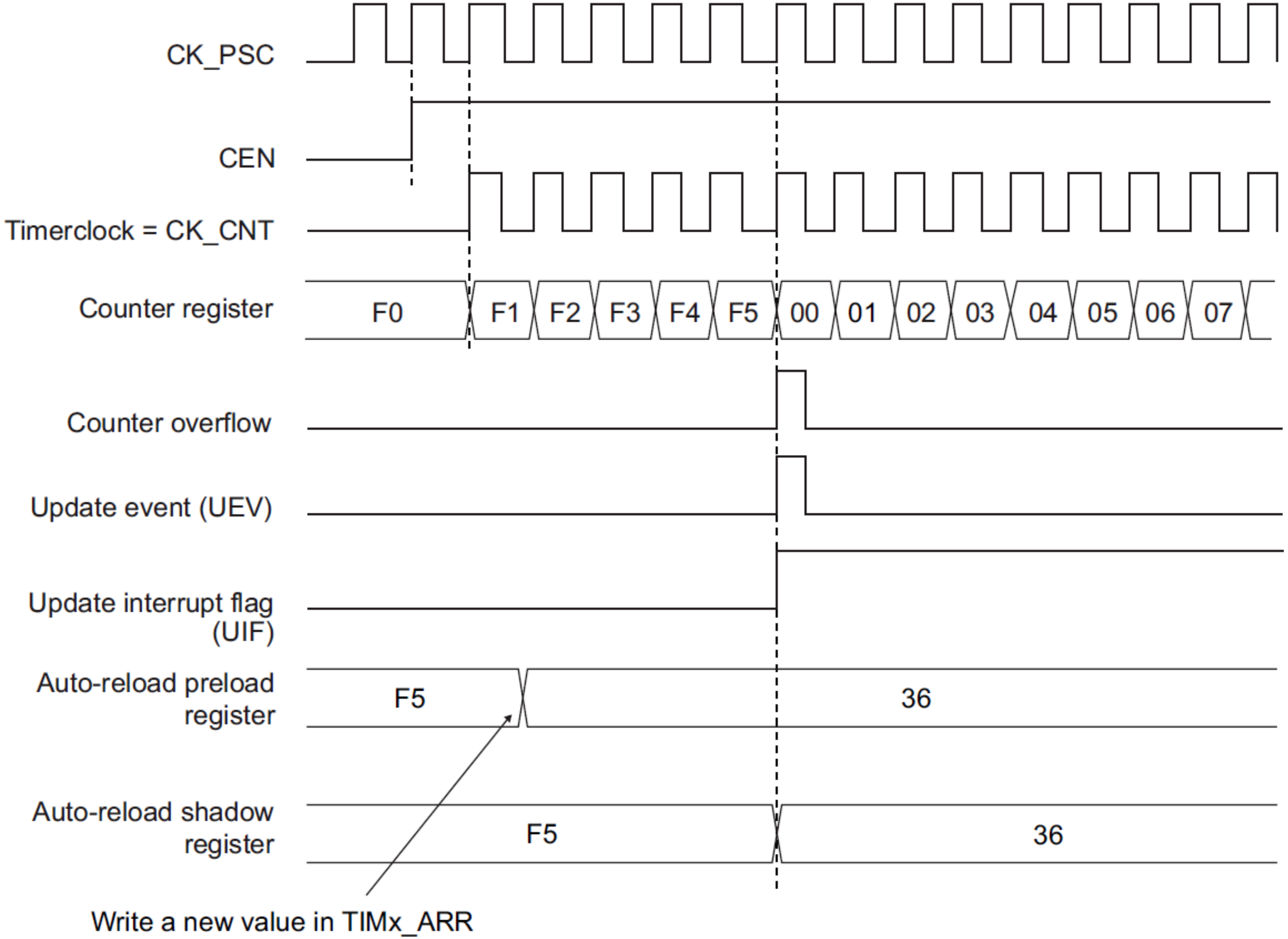
Upcounting mode

ARPE=0

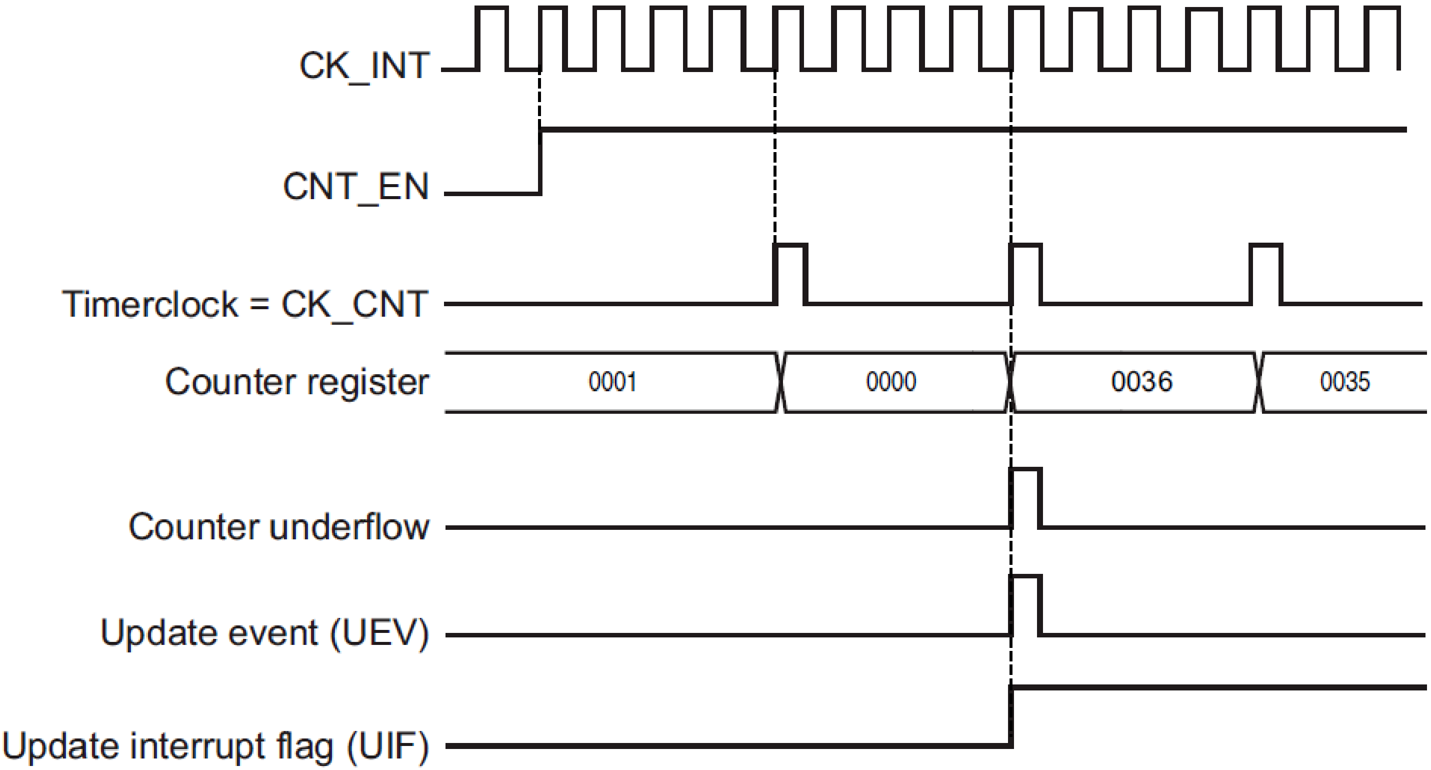


Upcounting mode

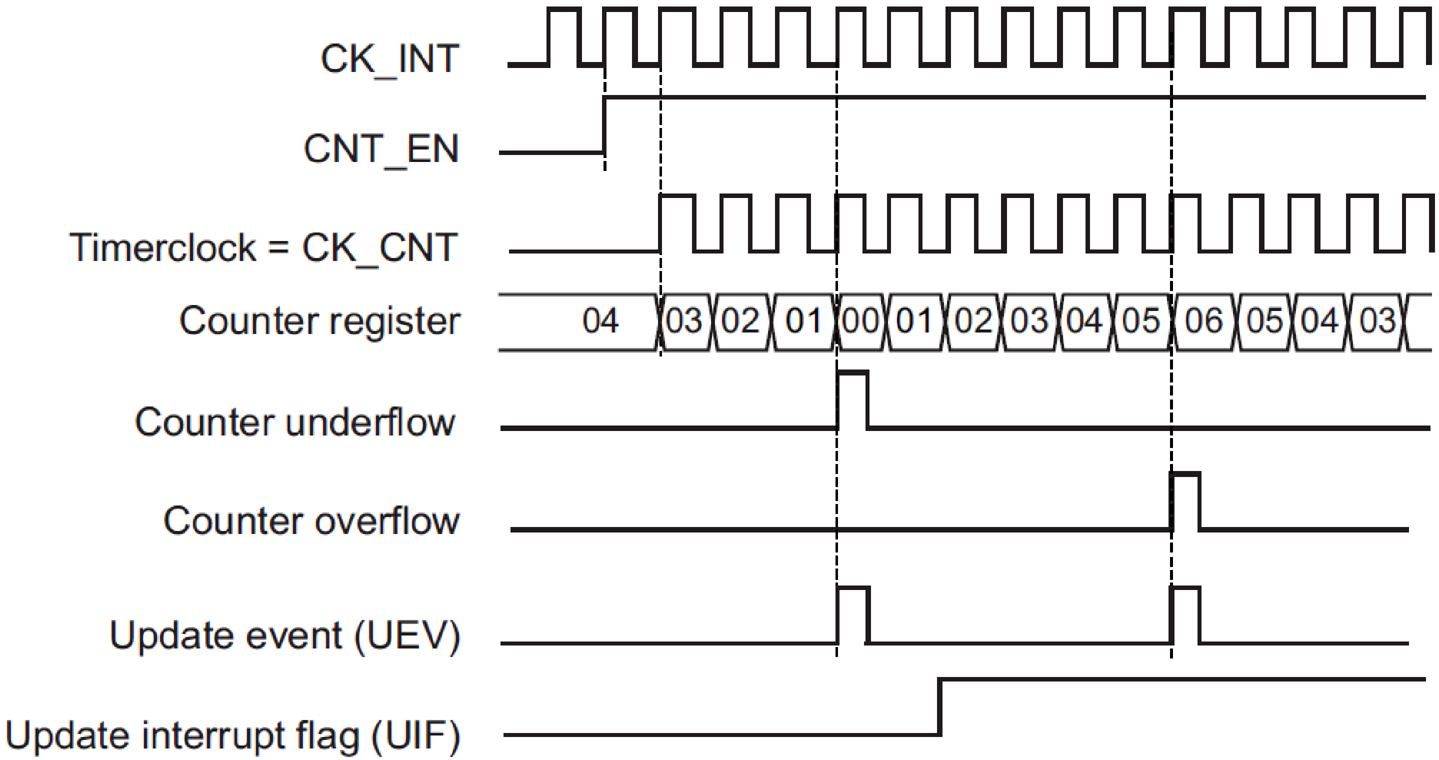
ARPE=1



Downcounting mode



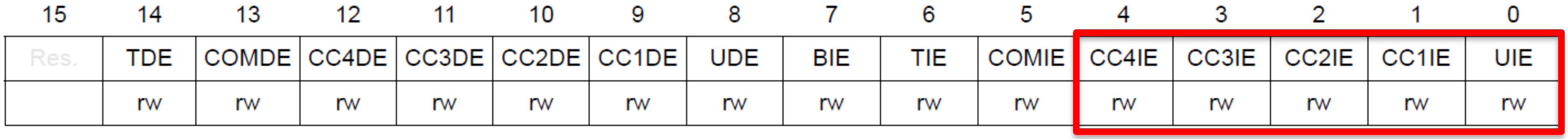
Center-aligned mode (up/down counting)



Timer’s registers

TIM DMA/interrupt enable register (TIMx_DIER)

Address offset: 0x0C



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

UIE: Update interrupt enable

0: Update interrupt disabled

1: Update interrupt enabled

CCxIE: Capture/Compare x interrupt enable

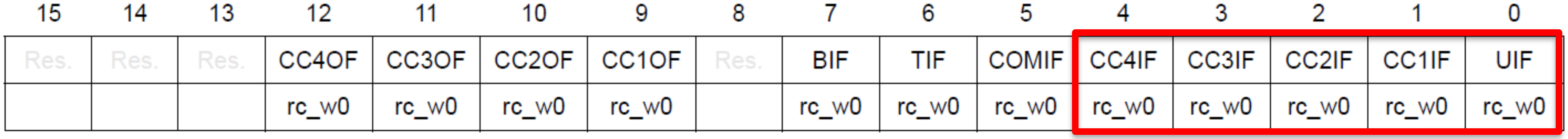
0: CC1 interrupt disabled

1: CC1 interrupt enabled

Timer’s registers

TIM status register (TIMx_SR)

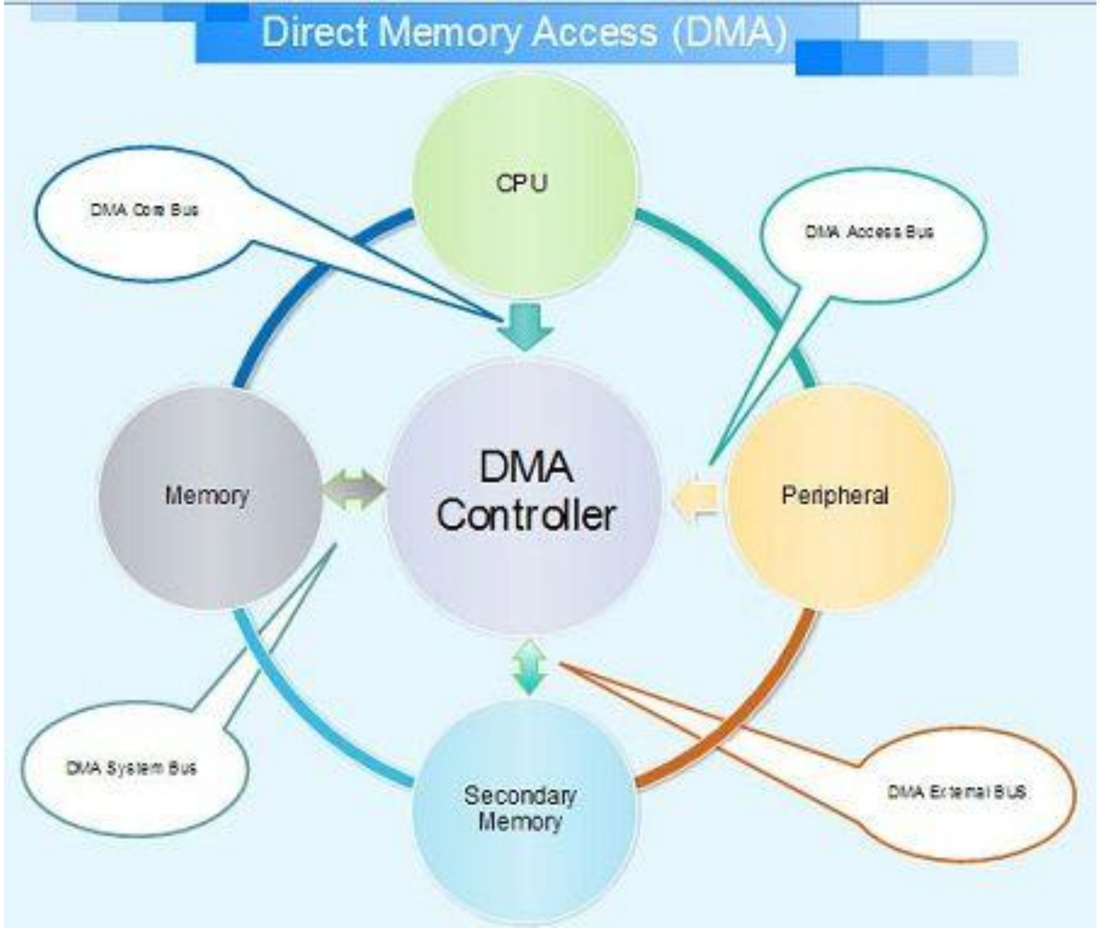
Address offset: 0x10



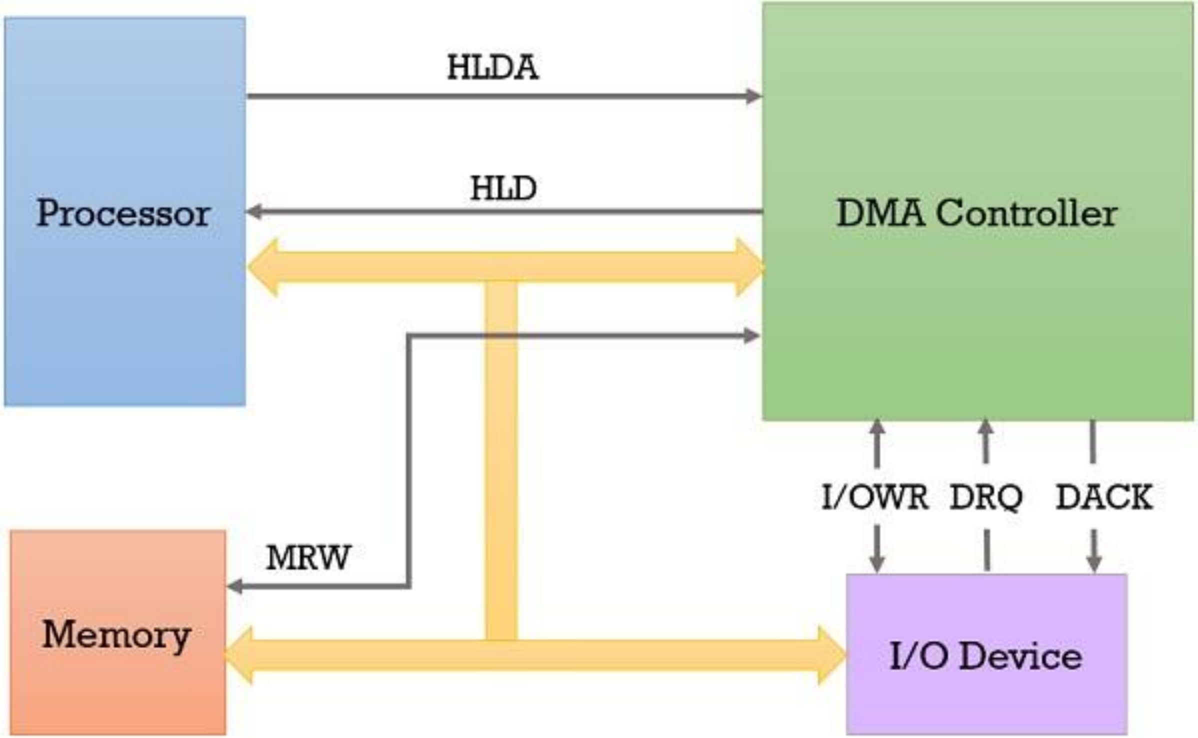
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

UIF: Update interrupt flag
This bit is set by hardware on an update event. It is cleared by software.
0: No update occurred.
1: Update interrupt pending. This bit is set by hardware when the registers are updated
CCxIF: Capture/Compare x interrupt flag
If channel CCx is configured as output:
This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software.
0: No match.
1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register.
When the contents of TIMx_CCR1 are greater than the contents of TIMx_ARR, the CC1IF bit goes high on the counter overflow (in upcounting and up/down-counting modes) or underflow (in downcounting mode)
If channel CC1 is configured as input:
This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register.
0: No input capture occurred
1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)

Direct Memory Access

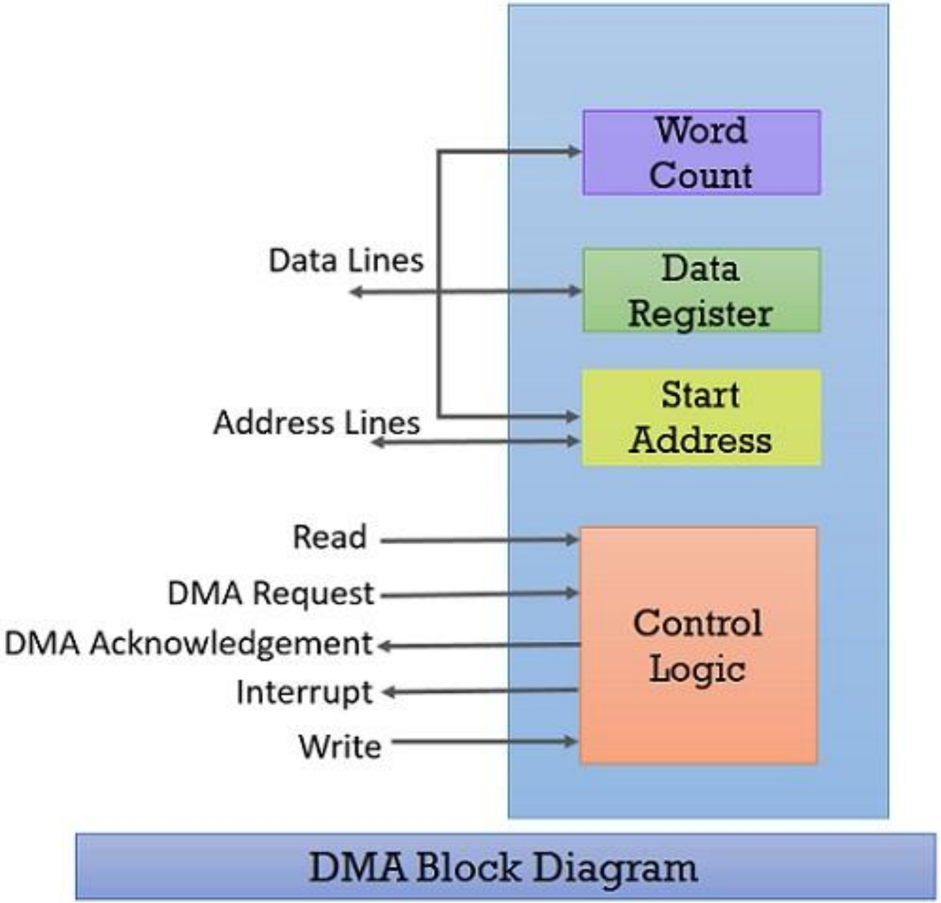


DMA Controller & its Working

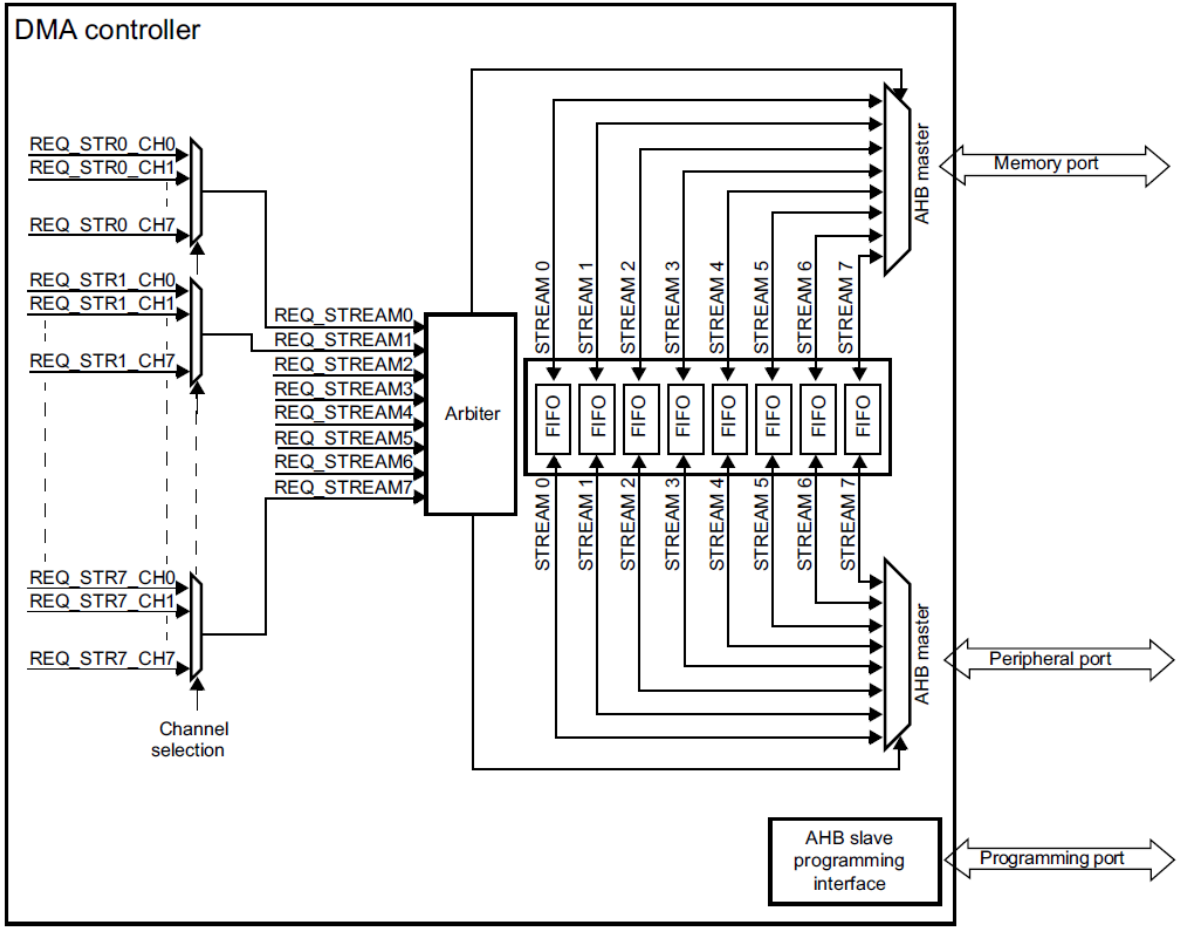


DMA Controller Data Transfer

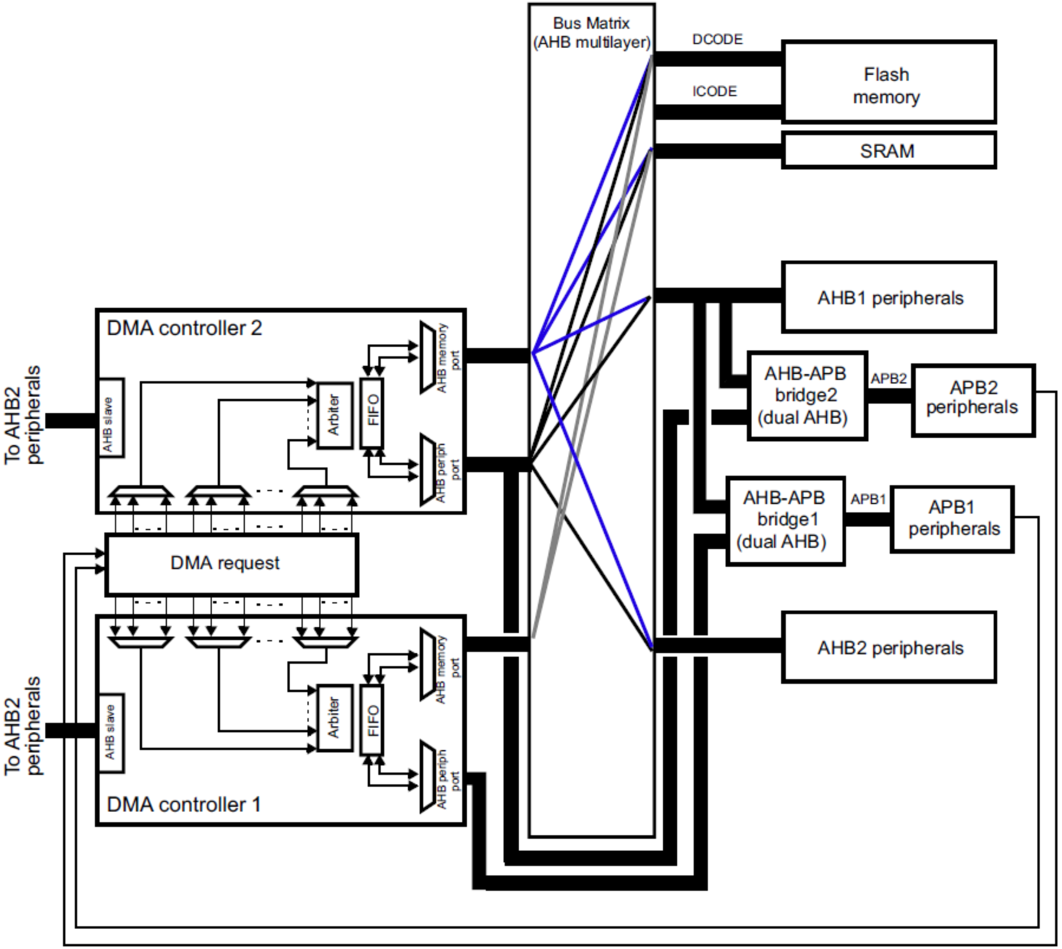
DMA Diagram


Word Count
Data Lines
Data Register
Start Address
Address Lines
Read
DMA Request
DMA Acknowledgement
Interrupt
Write
Control Logic
DMA Block Diagram

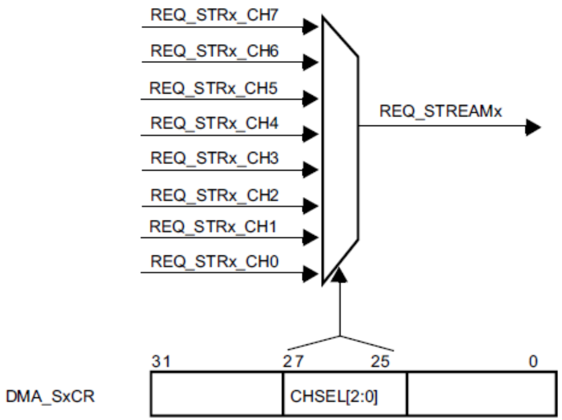
STM32 DMA

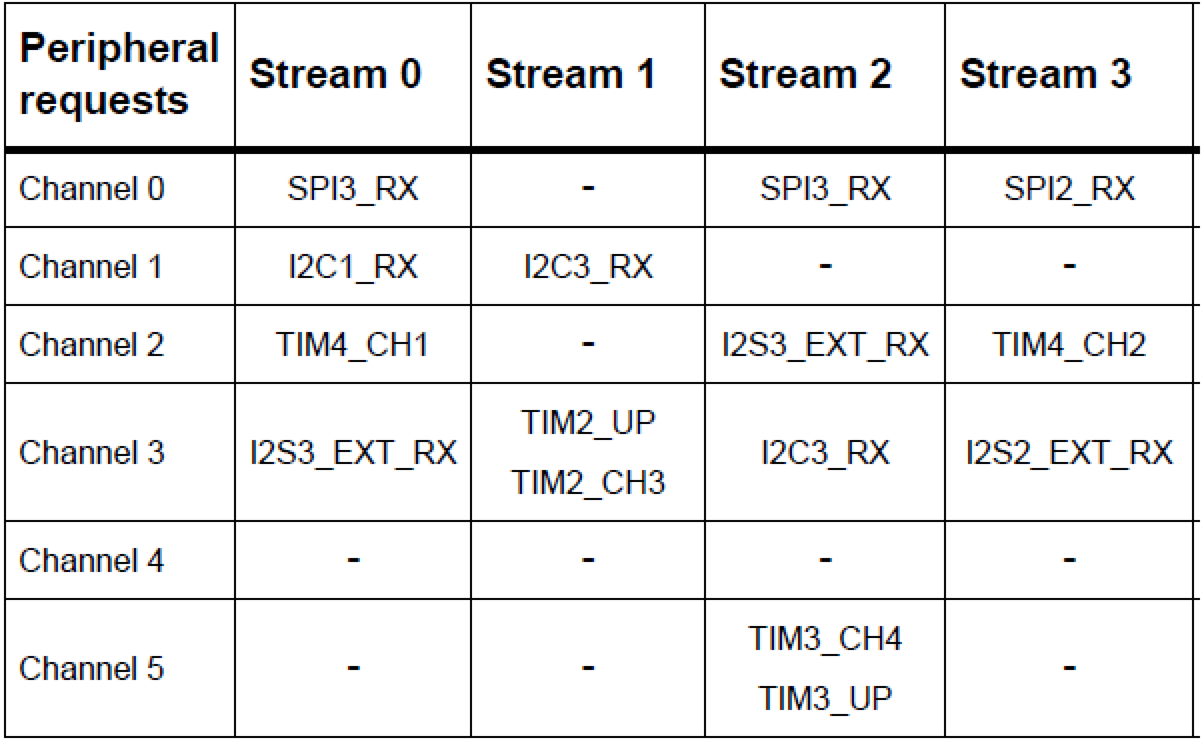

DMA controller
REQ_STR0_CH0
REQ_STR0_CH1
REQ_STR0_CH7
REQ_STR1_CH0
REQ_STR1_CH1
REQ_STR1_CH7
REQ_STR7_CH0
REQ_STR7_CH1
REQ_STR7_CH7
Channel selection
REQ_STREAM0
REQ_STREAM1
REQ_STREAM2
REQ_STREAM3
REQ_STREAM4
REQ_STREAM5
REQ_STREAM6
REQ_STREAM7
Arbiter
STREAM 0
STREAM 1
STREAM 2
STREAM 3
STREAM 4
STREAM 5
STREAM 6
STREAM 7
FIFO
AHB master
Memory port
AHB master
Peripheral port
AHB slave programming interface
Programming port

STM32 DMA


Bus Matrix
(AHB multilayer)
DCODE
ICODE
Flash memory
SRAM
AHB1 peripherals
AHB-APB bridge2 (dual AHB)
APB2
APB2 peripherals
AHB-APB bridge1 (dual AHB)
APB1
APB1 peripherals
AHB2 peripherals
DMA controller 2
AHB slave
Arbiter
FIFO
AHB memory port
AHB periph port
To AHB2 peripherals
DMA request
DMA controller 1
AHB slave
Arbiter
FIFO
AHB memory port
AHB periph port
To AHB2 peripherals

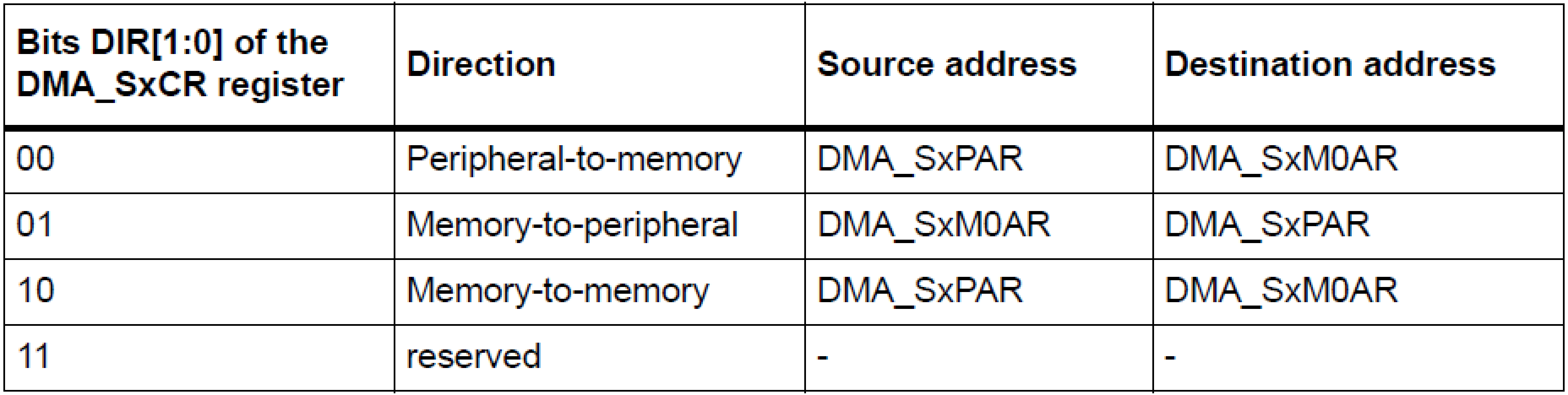
Channel selection





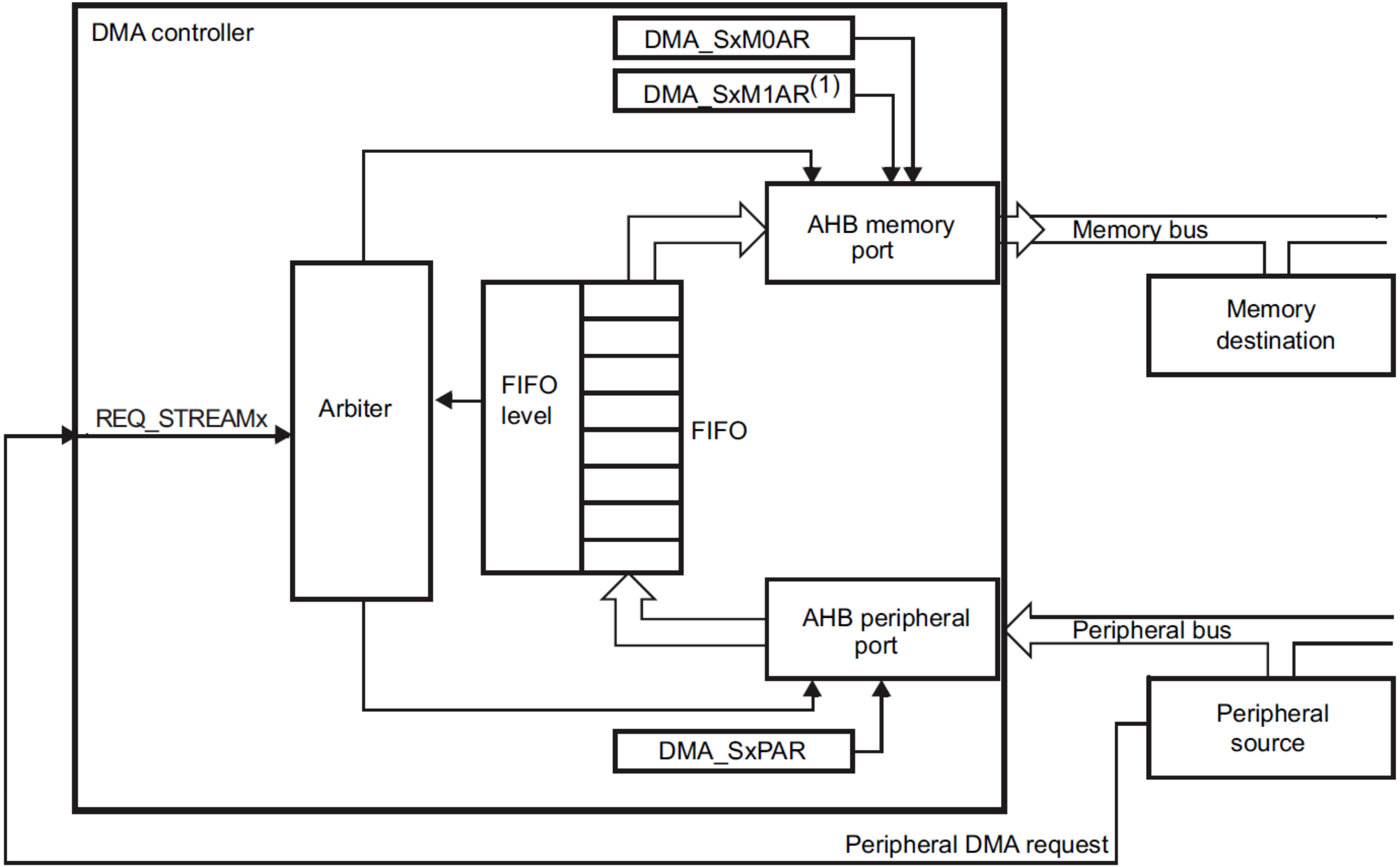
Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3
Channel 0	SPI3_RX	-	SPI3_RX	SPI2_RX
Channel 1	I2C1_RX	I2C3_RX	-	-
Channel 2	TIM4_CH1	-	I2S3_EXT_RX	TIM4_CH2
Channel 3	I2S3_EXT_RX	TIM2_UP TIM2_CH3	I2C3_RX	I2S2_EXT_RX
Channel 4	-	-	-	-
Channel 5	-	-	TIM3_CH4 TIM3_UP	-

Source, destination and transfer modes

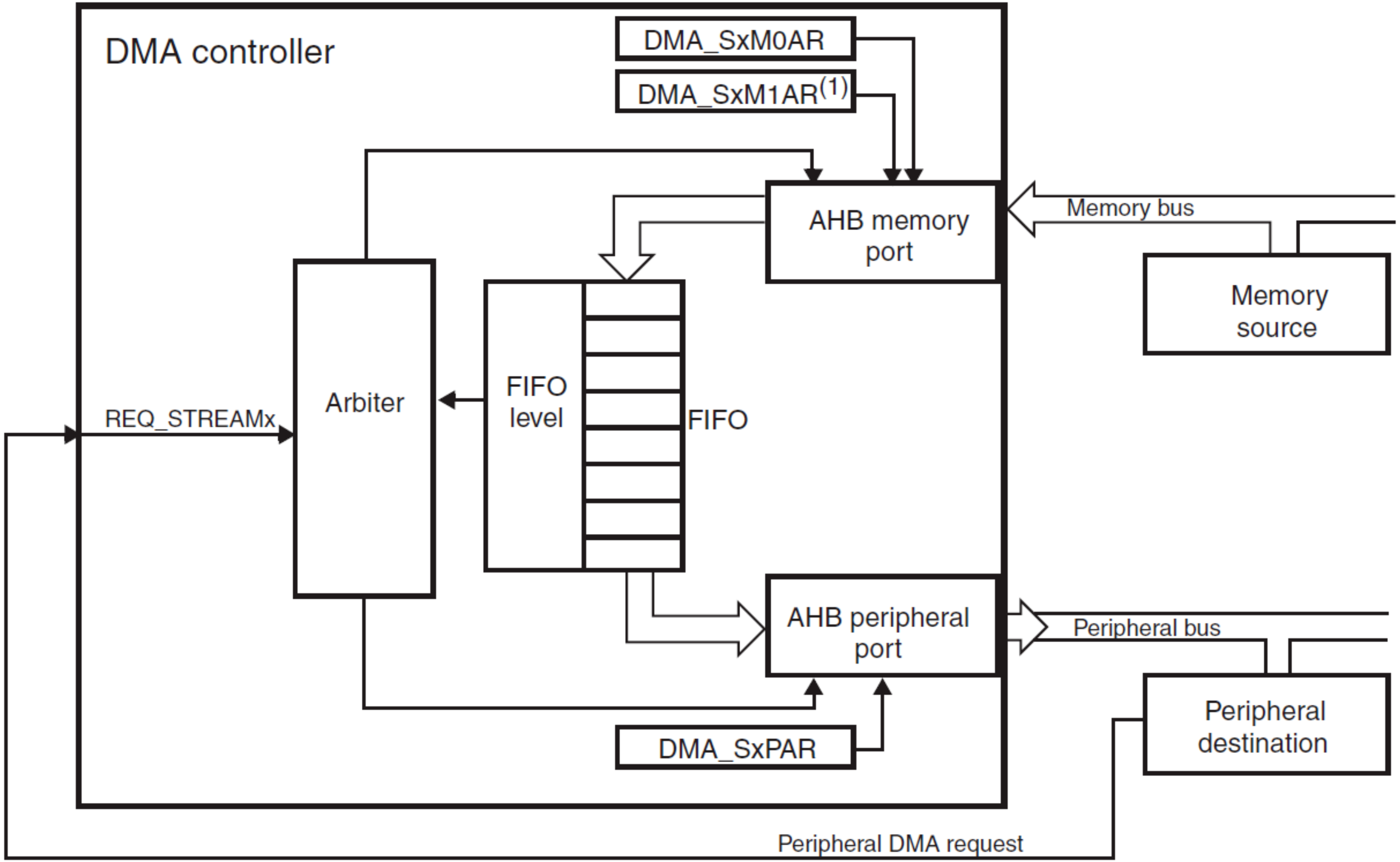


Bits DIR[1:0] of the DMA_SxCR register	Direction	Source address	Destination address
00	Peripheral-to-memory	DMA_SxPAR	DMA_SxM0AR
01	Memory-to-peripheral	DMA_SxM0AR	DMA_SxPAR
10	Memory-to-memory	DMA_SxPAR	DMA_SxM0AR
11	reserved	-	-

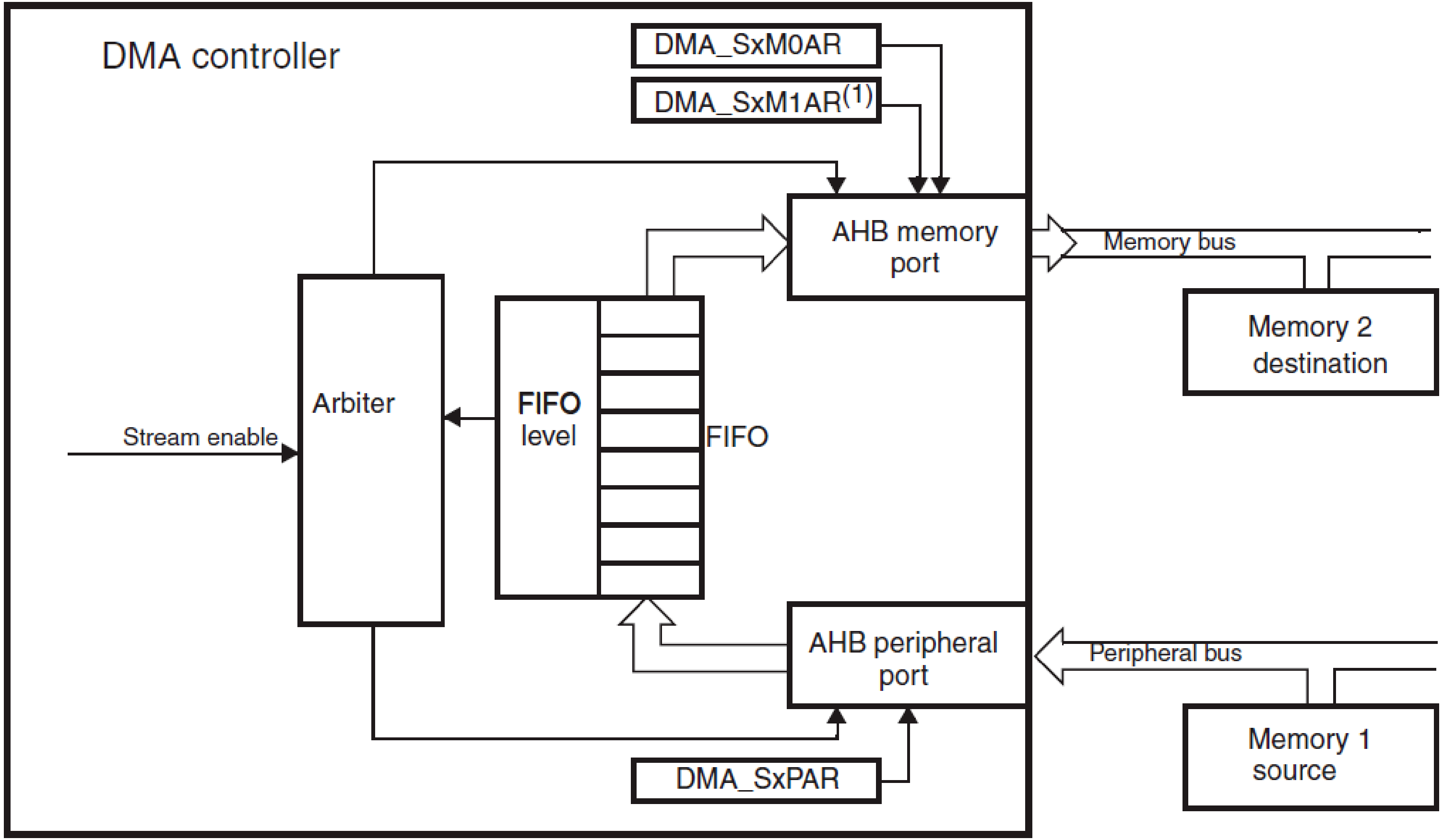
Peripheral-to-memory mode



Memory-to-peripheral mode



Memory-to-memory mode



DMA registers

DMA low/high interrupt status register (DMA_{L/H}ISR)

DMA low/high interrupt flag clear register (DMA_{L/H}IFCR)

DMA stream x configuration register (DMA_SxCR) (x = 0..7)

DMA stream x number of data register (DMA_SxNDTR) (x = 0..7)

DMA stream x peripheral address register (DMA_SxPAR) (x = 0..7)

DMA stream x memory 0 address register (DMA_SxM0AR) (x = 0..7)

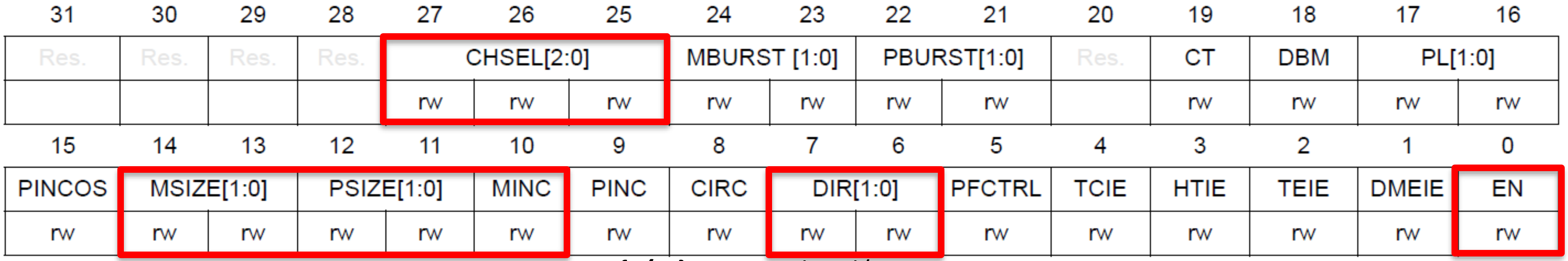
DMA stream x memory 1 address register (DMA_SxM1AR) (x = 0..7)

DMA stream x FIFO control register (DMA_SxFCR) (x = 0..7)

DMA registers

DMA stream x configuration register (DMA_SxCR) (x = 0..7)

Address offset: 0x10 + 0x18 * stream number



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	CHSEL[2:0]			MBURST [1:0]		PBURST[1:0]		Res.	CT	DBM	PL[1:0]	
				rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINCOS	MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR[1:0]		PFCTRL	TCIE	HTIE	TEIE	DMEIE	EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

EN: stream enable

This bit is set and cleared by software.

0: stream disabled

1: stream enabled

{DME/TE/HT/TC}EI: *direct mode error/transfer error/half transfer/transfer complete* interrupt enable

0: interrupt disabled

1: interrupt enabled

DIR: data transfer direction

00: peripheral-to-memory

01: memory-to-peripheral

10: memory-to-memory

11: reserved

{P/M}INC: peripheral/memory increment mode

0: address pointer is fixed

1: address pointer is incremented after each data transfer (increment is done according to PSIZE)

{P/M}SIZE: peripheral/memory data size

00: byte (8-bit)

01: half-word (16-bit)

10: word (32-bit)

11: reserved

CHSEL: channel selection

000: channel 0 selected

001: channel 1 selected

010: channel 2 selected

011: channel 3 selected

100: channel 4 selected

101: channel 5 selected

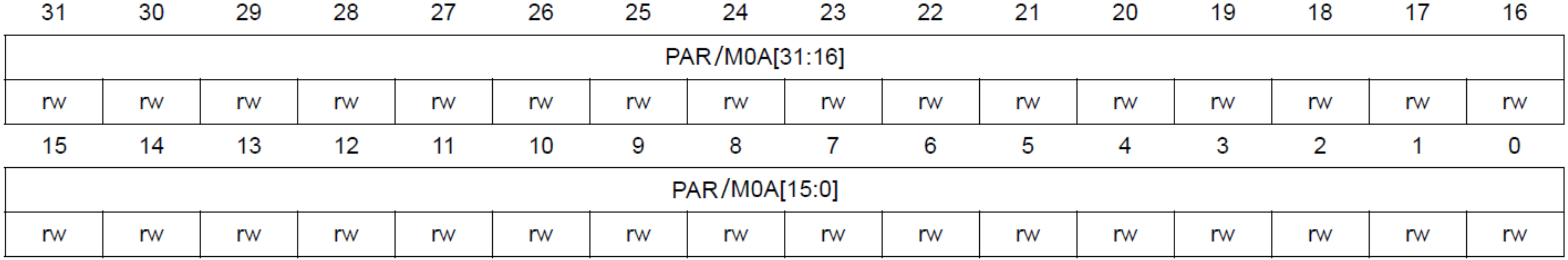
110: channel 6 selected

111: channel 7 selected

DMA registers

DMA stream x peripheral/memory 0 address register (DMA_Sx{P/M0}AR) (x = 0..7)

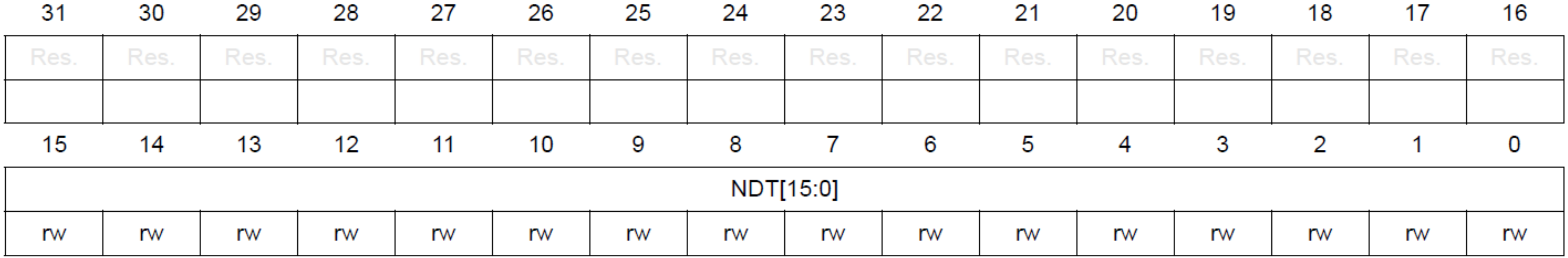
Address offset: 0x18/0x1C + 0x18 * stream number



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAR/M0A[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAR/M0A[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

DMA stream x number of data register (DMA_SxNDTR) (x = 0..7)

Address offset: 0x14 + 0x18 * stream number



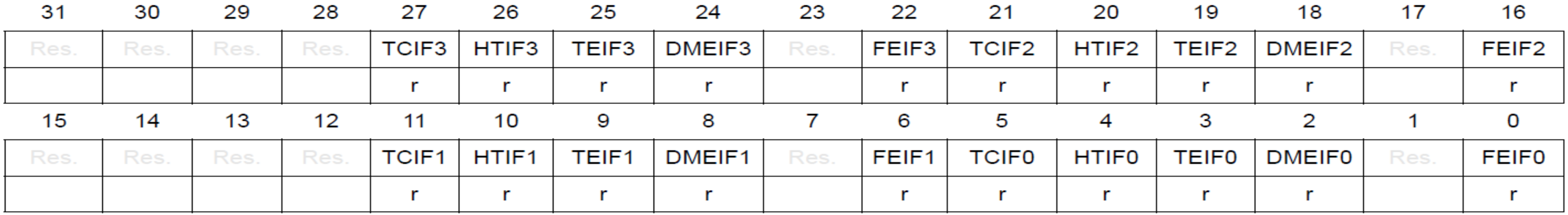
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NDT[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

DMA registers

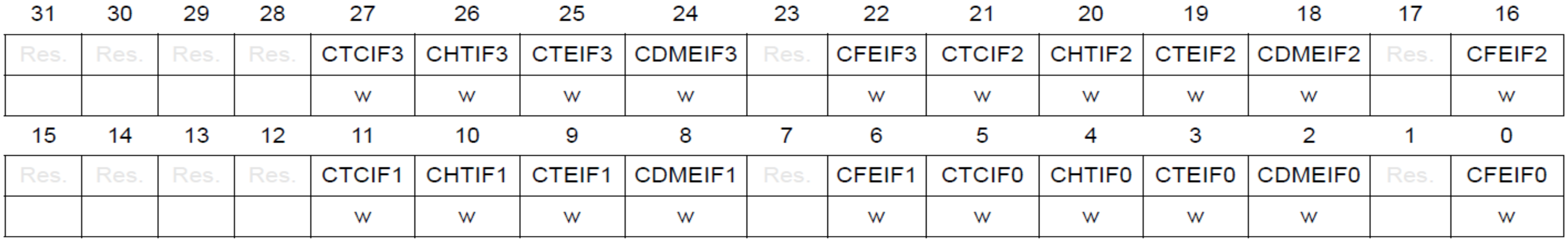
DMA low/high interrupt status register (DMA_{L/H}ISR)

DMA low/high interrupt flag clear register (DMA_{L/H}IFCR)

Address offset: 0x00, 0x04, 0x08, 0x0C



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TCIF3	HTIF3	TEIF3	DMEIF3	Res.	FEIF3	TCIF2	HTIF2	TEIF2	DMEIF2	Res.	FEIF2
				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**
Res.	Res.	Res.	Res.	TCIF1	HTIF1	TEIF1	DMEIF1	Res.	FEIF1	TCIF0	HTIF0	TEIF0	DMEIF0	Res.	FEIF0
				r	r	r	r		r	r	r	r	r		r



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	CTCIF3	CHTIF3	CTEIF3	CDMEIF3	Res.	CFEIF3	CTCIF2	CHTIF2	CTEIF2	CDMEIF2	Res.	CFEIF2
				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**
Res.	Res.	Res.	Res.	CTCIF1	CHTIF1	CTEIF1	CDMEIF1	Res.	CFEIF1	CTCIF0	CHTIF0	CTEIF0	CDMEIF0	Res.	CFEIF0
				w	w	w	w		w	w	w	w	w		w

FEIFx: stream x {clear} FIFO error interrupt flag

DMEIFx: stream x {clear} direct mode error interrupt flag

TEIFx: stream x {clear} transfer error interrupt flag

HTIFx: stream x {clear} half transfer interrupt flag

TCIFx: stream x {clear} transfer complete interrupt flag

DMA Enable algorithm

Steps for start DMA transfer

1. Enable DMA transfer in peripheral device.
2. Config DMA stream:
    1. Select peripheral channel.
    2. Set transfer direction.
    3. Set increase mode for memory or peripheral, if required.
    4. Enable events interrupts, if required.
3. Set peripheral address, memory address and number of bytes for transfer.
4. Start transfer by enabling DMA Stream.

DMA Example

```
uint8_t tx_data[10];

int main()
{   RCC->APB1ENR |= RCC_APB1ENR_USART2EN;
    RCC->AHB1ENR |= RCC_AHB1ENR_DMA1EN;
    USART2->BRR = APB1/USART_BAUDRATE;
    USART2->CR1 = USART_CR1_UE | USART_CR1_TE | USART_CR1_RE;
    USART2->CR3 = USART_CR3_DMAT;

    DMA1_Stream6->CR = DMA_SxCR_DIR_0 | DMA_SxCR_MINC | DMA_SxCR_CHSEL_2;
    DMA1_Stream6->PAR = (uint32_t) &USART2->DR;
    DMA1_Stream6->M0AR = (uint32_t) &tx_data;

    DMA1_Stream6->HIFCR = DMA_HIFCR_CTCIF6 | DMA_HIFCR_CHTIF6;
    DMA1_Stream6->NDTR = 10;
    DMA1_Stream6->CR |= DMA_SxCR_EN;
}
```

Thank you for attention

www.ifmo.ru

ITsMOre than a UNIVERSITY