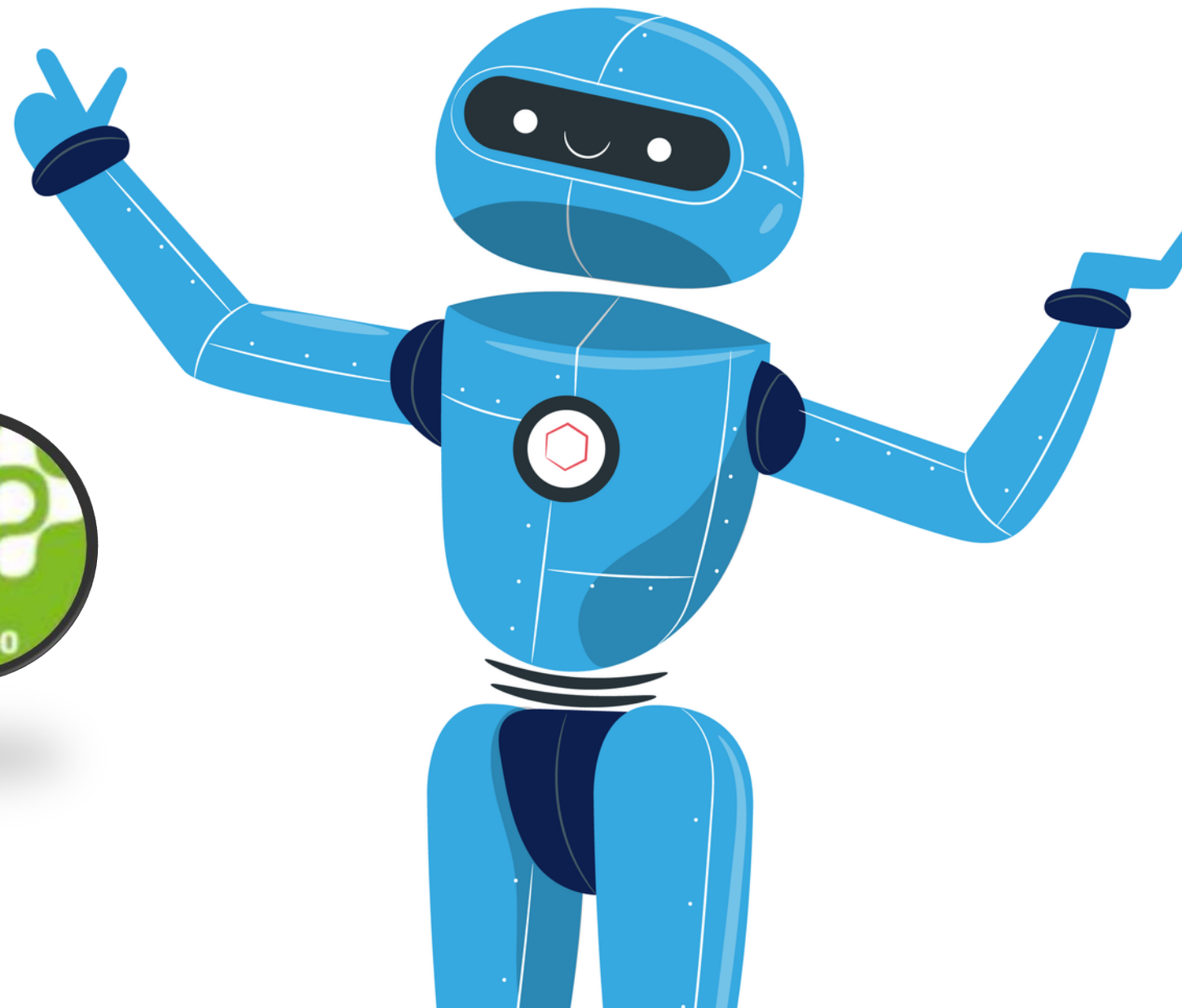


CLASE 6: NVIC II

MICROCONTROLADORES ARM

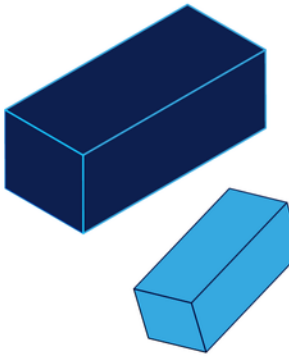


INTERRUPCION DE LA SYSTICK

arm

MICRO-
CONTRO-
LADORES
ARM

INTERRUPCION DE LA SYSTICK



- El SysTick Timer es un periférico interno del CORTEX-M, por lo tanto, para poder generar interrupciones por el SysTick se debe establecer el bit **TICKINT** del registro **CTRL**.

SysTick control and status register (STK_CTRL)

Address offset: 0x00

Reset value: 0x0000 0000

Required privilege: Privileged

The SysTick CTRL register enables the SysTick features.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															COUNT FLAG
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CLKSO URCE	TICK INT	EN ABLE	
												rw	rw	rw	

```
/*enable SysTick_IT*/  
SysTick->CTRL |= SysTick_CTRL_TICKINT_Msk;
```

Bit 1 TICKINT: SysTick exception request enable

0: Counting down to zero does not assert the SysTick exception request

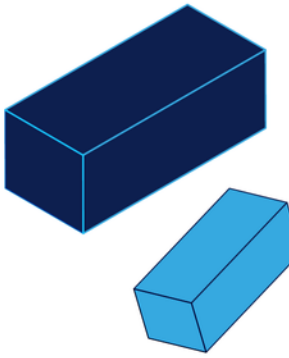
1: Counting down to zero to asserts the SysTick exception request.

Note: Software can use COUNTFLAG to determine if SysTick has ever counted to zero.

arm

MICRO-
CONTRO-
LADORES
ARM

INTERRUPCION DE LA SYSTICK



- Para el cambio de prioridad se usa el registro **SHPR3**, en específico los bits **PRI_15**

System handler priority register 3 (SHPR3)

Address: 0xE000 ED20

Reset value: 0x0000 0000

Required privilege: Privileged

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_15[7:4]				PRI_15[3:0]				PRI_14[7:4]				PRI_14[3:0]			
rw	rw	rw	rw	r	r	r	r	rw	rw	rw	rw	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits 31:24 **PRI_15**: Priority of system handler 15, SysTick exception

Bits 23:16 **PRI_14**: Priority of system handler 14, PendSV

Bits 15:0 Reserved, must be kept cleared

```
/*SysTick Change Priority*/  
//register  
SCB->SHP[(((uint32_t)SysTick_IRQn) & 0xFUL) - 4UL] = (uint8_t)(15<<4) & 0xFFU;  
//CMSIS function  
NVIC_SetPriority(SysTick_IRQn,15);
```

arm

MICRO-
CONTRO-
LADORES
ARM

INTERRUPCION DE LA SYSTICK

EJEMPLO DE CONFIGURACION

```
void Delay_Init(void){
    __disable_irq();
    /*1. deshabilitar el systick*/
    SysTick->CTRL = 0;
    /*2. configurar el valor de reload*/
    SysTick->LOAD = SystemCoreClock / 1000 - 1;
    /*3. seleccionar la fuente de reloj*/
    SysTick->CTRL |= SysTick_CTRL_CLKSOURCE_Msk;
    /*4. poner el registro val*/
    SysTick->VAL = 0;
    /*5. habilitar la interrupcion y el conteo*/
    SysTick->CTRL |= SysTick_CTRL_TICKINT_Msk | \
                    SysTick_CTRL_ENABLE_Msk;
    /*6. (opcional) cambiar la prioridad*/
    NVIC_SetPriority(SysTick_IRQn,15);
    __enable_irq();
    return;
}
```

ISR

```
/*ISR*/
void SysTick_Handler(void){
    if(SysTick->CTRL & SysTick_CTRL_COUNTFLAG_Msk)
        uwTick+= 1;    //se incrementa el valor de la variable en 1

    return;
}
```

arm

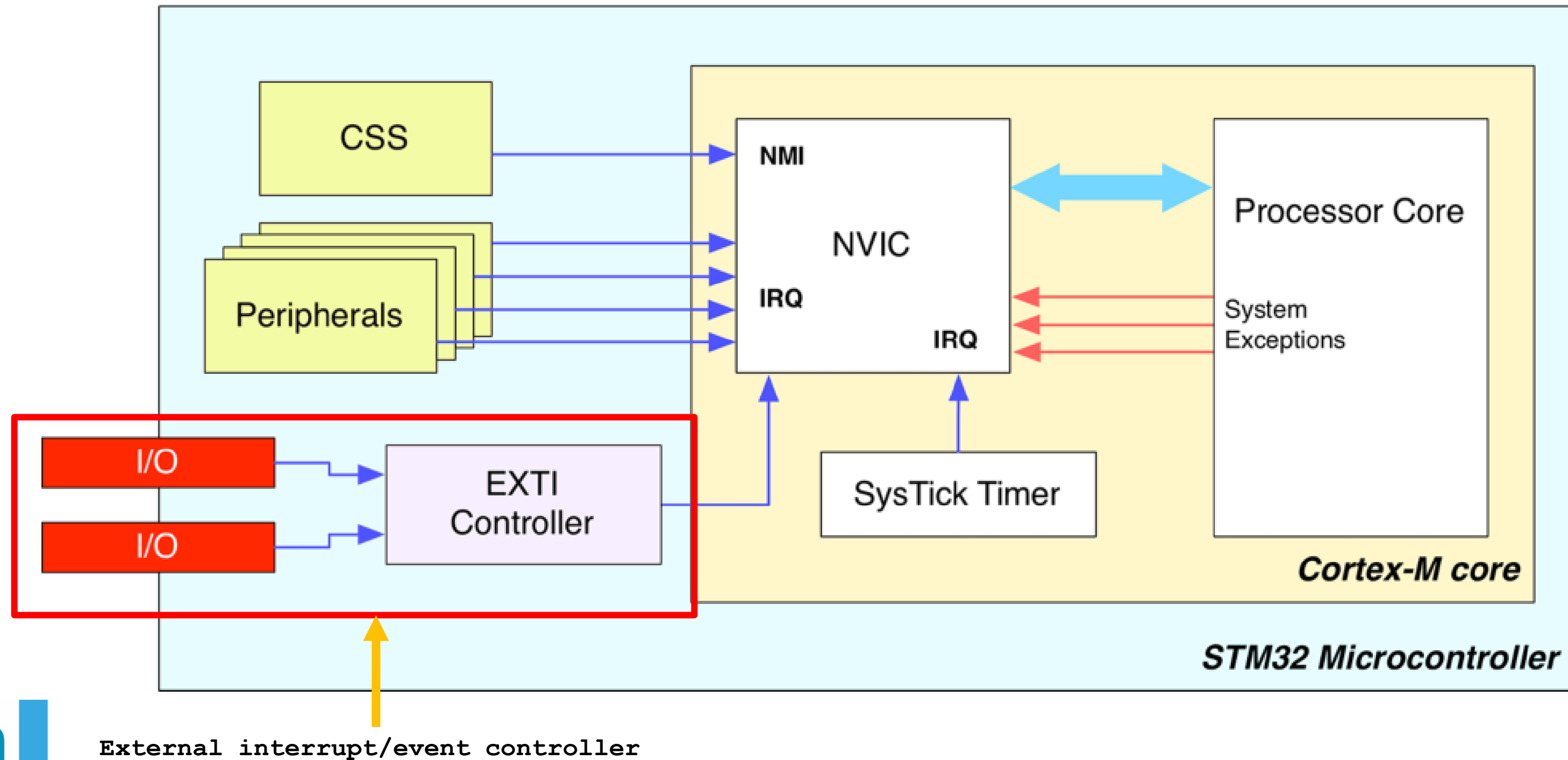
MICRO-
CONTRO-
LADORES
ARM

DESCRIPCION DE LA INTERRUPCION EXTERNA

arm

MICRO-
CONTRO-
LADORES
ARM

INTERRUPCION EXTERNA

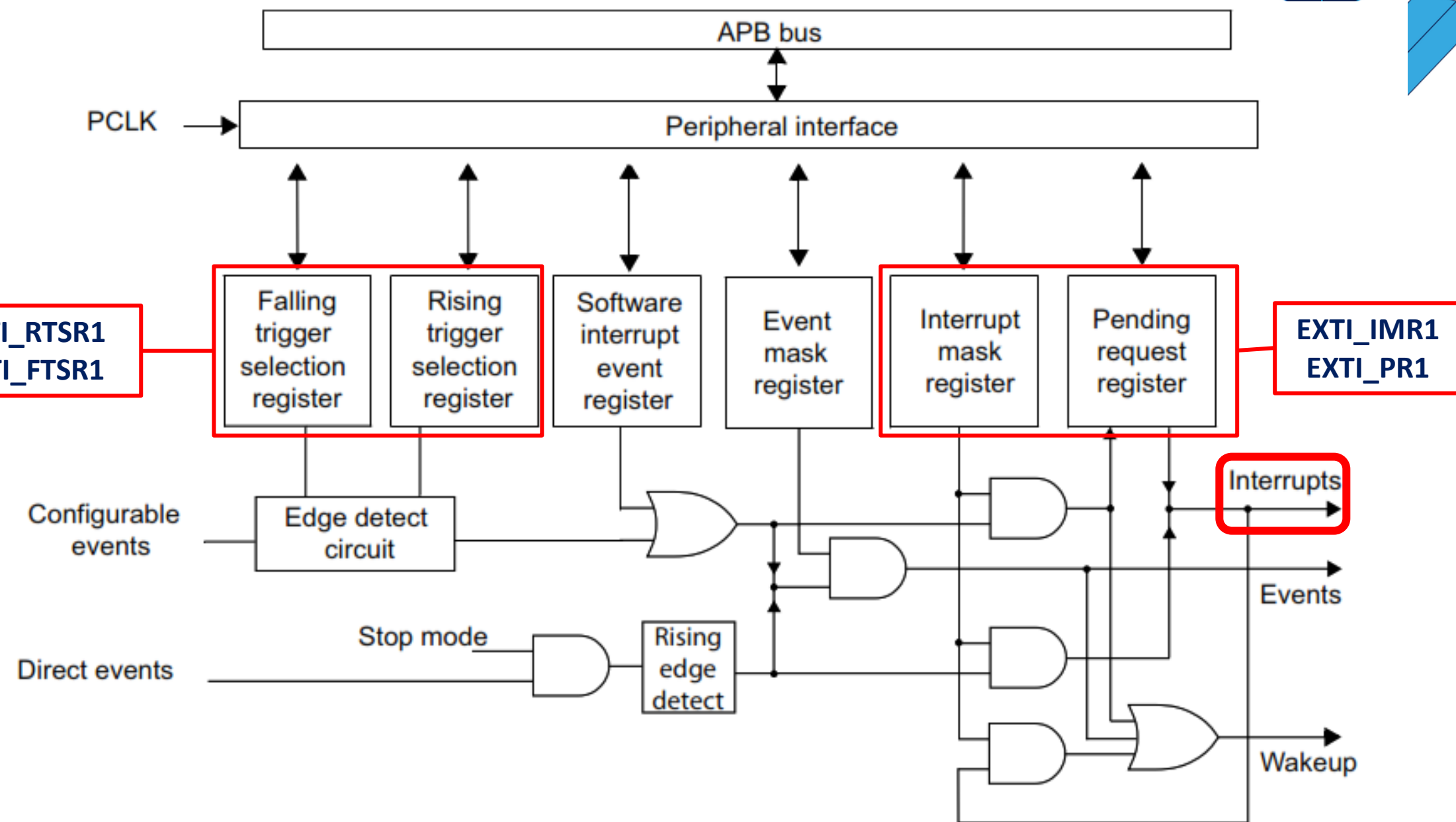
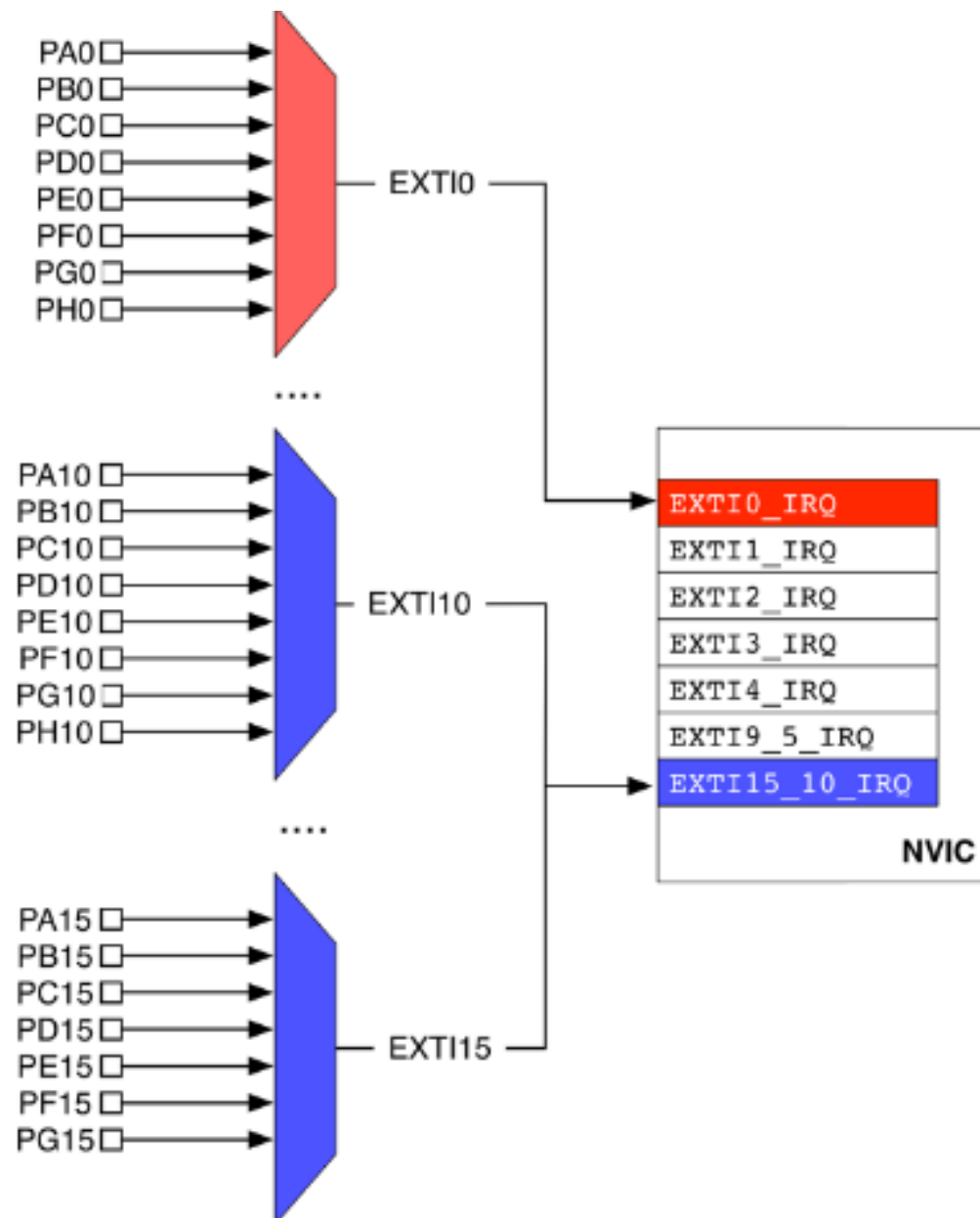


arm

MICRO-
CONTRO-
LADORES
ARM

INTERRUPCION EXTERNA

- Generación de hasta **37** solicitudes de eventos/interrupciones.
 - **25 líneas configurables**
 - **12 líneas directas.**
- Máscara independiente en cada línea de evento/interrupción.
- Flanco ascendente o descendente configurable (solo líneas configurables)
- Bit de estado dedicado (solo líneas configurables)



INTERRUPCION EXTERNA

UBICACIÓN EN LA TABLA DE VECTORES

NUMERO IRQ						DIRECCION EN LA TABLA DE VECTORES
6	13	settable	EXTI0	EXTI Line0 interrupt		0x0000 005C
7	14	settable	EXTI1	EXTI Line1 interrupt		0x0000 005C
8	15	settable	EXTI2	EXTI Line2 interrupt		0x0000 0060
9	16	settable	EXTI3	EXTI Line3 interrupt		0x0000 0064
10	17	settable	EXTI4	EXTI Line4 interrupt		0x0000 0068
23	30	settable	EXTI9_5	EXTI Line[9:5] interrupts		0x0000 009C
40	47	settable	EXTI15_10	EXTI Line[15:10] interrupts		0x0000 00E0

PRIORIDAD

EXTI5, 6, 7, 8 Y 9

EXTI10, 11, 12, 13, 14, 15



INTERRUPCION EXTERNA

SELECCIÓN DE INTERRUPCION

Para configurar una línea como fuente de interrupción, utilice el siguiente procedimiento:

- Configure el bit de máscara correspondiente en el registro **EXTI_IMRx**.
- Configure los bits de selección de disparo de la línea de interrupción (**EXTI_RTsr y EXTI_FTSR**).
- Configure los bits de habilitación y máscara que controlan el canal **IRQ NVIC** asignado al EXTI para que una interrupción proveniente de una de las líneas EXTI pueda reconocerse correctamente.

```
/*Interrupt mask configuration*/
EXTI->IMR1 |= 1U<<5;           //Interrupt request from Line 5 is not masked
/*Set trigger selection*/
EXTI->RTSR1 |= 1U<<5;          //Rising trigger enabled
/*set priority*/
NVIC->IP[23] = 1U<<4;           //priority 1
//NVIC_SetPriority(EXTI9_5_IRQn,1);
/*enable interrupt*/
NVIC->ISER[23>>5U] |= 1U<<(23 & 0x1F);
//NVIC_EnableIRQ(EXTI9_5_IRQn);
```

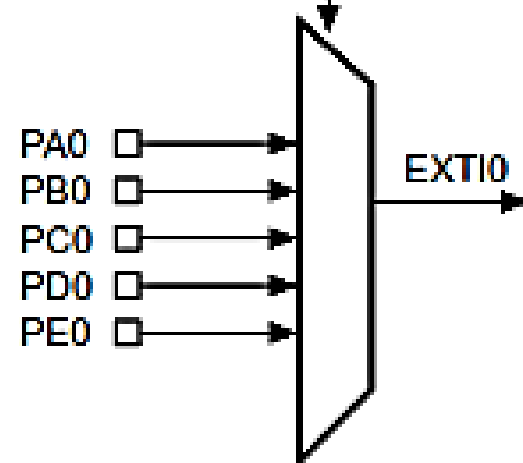
arm

MICRO-
CONTRO-
LADORES
ARM

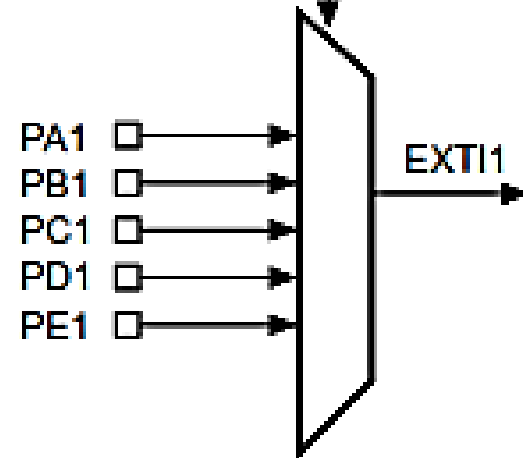
INTERRUPCION EXTERNA

Mapeo de línea de eventos/interrupciones EXTI

EXTI0[3:0] bits in the SYSCFG_EXTICR1 register

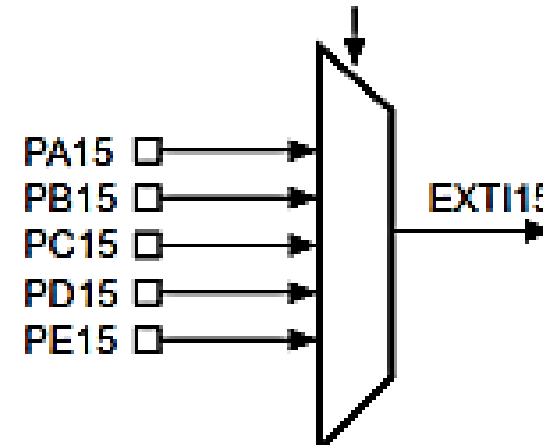


EXTI1[3:0] bits in the SYSCFG_EXTICR1 register



...

EXTI15[3:0] bits in the SYSCFG_EXTICR4 register



SYSCFG external interrupt configuration register 2
(SYSCFG_EXTICR2)

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
Res	EXTI7[2:0]			Res	EXTI6[2:0]			Res	EXTI5[2:0]			Res	EXTI4[2:0]		
	rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw	rw

Bits 6:4 **EXTI5[2:0]**: EXTI 5 configuration bits

These bits are written by software to select the source input for the EXTI5 external interrupt.

000: PA[5] pin

001: PB[5] pin

010: PC[5] pin

011: PD[5] pin

100: PE[5] pin

101: Reserved

110: Reserved

111: Reserved

```
/*EXTI interrupt/event line mapping*/
```

```
RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
```

```
SYSCFG->EXTICR[1] &=~ 0x7U<<4; //PA5 mapping to EXTI5
```

```
//SYSCFG->EXTICR[1] |= SYSCFG_EXTICR2_EXTI5_PA;
```

arm

MICRO-
CONTRO-
LADORES
ARM

INTERRUPCION EXTERNA

Mapeo de línea de eventos/interrupciones EXTI

EXTI line	Line source ⁽¹⁾	Line type
0-15	GPIO	configurable
16	PVD	configurable
17	USB FS wakeup event ⁽³⁾⁽²⁾	direct
18	RTC alarms	configurable
19	RTC tamper or timestamp or CSS_LSE	configurable
20	RTC wakeup timer	configurable
21	COMP1 output	configurable
22	COMP2 output	configurable
23	I2C1 wakeup ⁽³⁾	direct

EXTI line	Line source ⁽¹⁾	Line type
24	I2C2 wakeup ⁽³⁾⁽⁴⁾	direct
25	I2C3 wakeup	direct
26	USART1 wakeup ⁽³⁾	direct
27	USART2 wakeup ⁽³⁾	direct
28	USART3 wakeup ⁽³⁾⁽⁴⁾	direct
30	-	-
29	UART4 wakeup ⁽³⁾⁽⁵⁾	direct
31	LPUART1 wakeup	direct
32	LPTIM1	direct
33	LPTIM2 ⁽⁶⁾	direct
34	SWPMI1 wakeup ⁽³⁾⁽⁷⁾	direct
35	PVM1 wakeup	configurable
36	-	-
37	PVM3 wakeup	configurable
38	PVM4 wakeup	configurable
39	LCD wakeup ⁽⁸⁾	direct
40	I2C4 wakeup ⁽⁵⁾	direct

arm

MICRO-
CONTRO-
LADORES
ARM

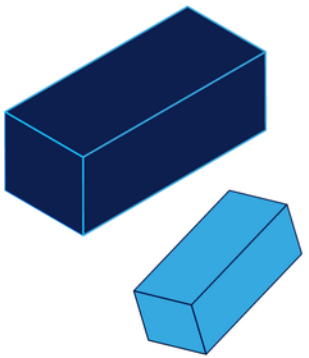
REGISTROS DEL EXTI

arm

MICRO-
CONTRO-
LADORES
ARM

UMAKER | CENTRO DE CAPACITACIÓN
DE DESARROLLO TECNOLÓGICO

REGISTROS EXTI



Interrupt mask register 1 (EXTI_IMR1)

Address offset: 0x00

Reset value: 0xFF820000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IM31	Res.	Res.	IM28	IM27	IM26	IM25	IM24	IM23	IM22	IM21	IM20	IM19	IM18	IM17	IM16
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
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

EXTI15

EXTI0

- Bit 31 IM31:** Interrupt Mask on line 31
0: Interrupt request from Line 31 is masked
1: Interrupt request from Line 31 is not masked
- Bits 30:29** Reserved, must be kept at reset value.
- Bits 28:0 IMx:** Interrupt Mask on line x (x = 28 to 0)
0: Interrupt request from Line x is masked
1: **Interrupt request from Line x is not masked**

arm

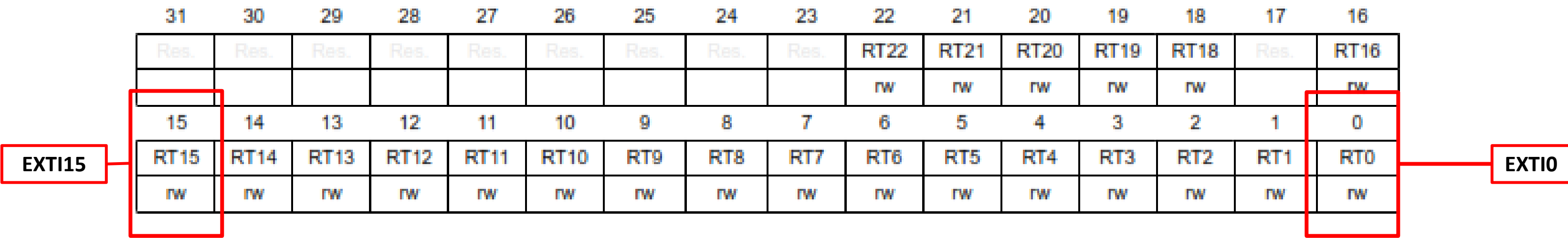
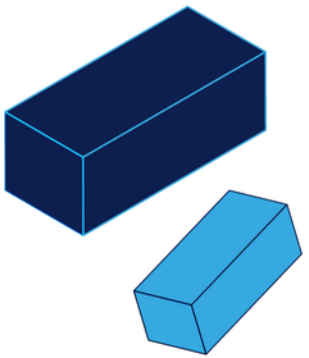
MICRO-
CONTRO-
LADORES
ARM

REGISTROS EXTI

Rising trigger selection register 1 (EXTI_RTSR1)

Address offset: 0x08

Reset value: 0x00000000



Bits 31:23 Reserved, must be kept at reset value.

Bits 22:18 RTx: Rising trigger event configuration bit of line x (x = 22 to 18)
0: Rising trigger disabled (for Event and Interrupt) for input line
1: **Rising trigger enabled** (for Event and Interrupt) for input line

Bit 17 Reserved, must be kept at reset value.

Bits 16:0 RTx: Rising trigger event configuration bit of line x (x = 16 to 0)
0: Rising trigger disabled (for Event and Interrupt) for input line
1: **Rising trigger enabled** (for Event and Interrupt) for input line



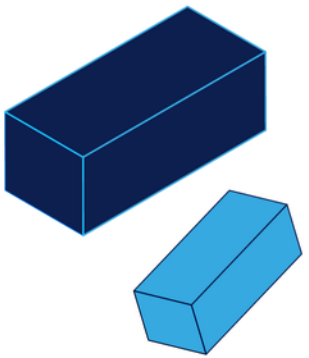


arm

MICRO-
CONTRO-
LADORES
ARM

```
1: Falling trigger enabled (for Event and Interrupt) for input line
```

REGISTROS EXTI



Falling trigger selection register 1 (EXTI_FTSR1)

Address offset: 0x0C

Reset value: 0x0000 0000

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.	FT22	FT21	FT20	FT19	FT18	Res.	FT16
									rw	rw	rw	rw	rw		rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

EXTI15

EXTI0

Bits 31:23 Reserved, must be kept at reset value.

Bits 22:18 FTx: Falling trigger event configuration bit of line x (x = 22 to 18)

0: Falling trigger disabled (for Event and Interrupt) for input line

1: **Falling trigger enabled** (for Event and Interrupt) for input line

Bit 17 Reserved, must be kept at reset value.

Bits 16:0 FTx: Falling trigger event configuration bit of line x (x = 16 to 0)

0: Falling trigger disabled (for Event and Interrupt) for input line

1: **Falling trigger enabled** (for Event and Interrupt) for input line

arm

MICRO-
CONTRO-
LADORES
ARM

REGISTROS EXTI

Pending register 1 (EXTI_PR1)

Address offset: 0x14

Reset value: undefined

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.	PIF22	PIF21	PIF20	PIF19	PIF18	Res.	PIF16
									rc_w1	rc_w1	rc_w1	rc_w1	rc_w1		rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIF15	PIF14	PIF13	PIF12	PIF11	PIF10	PIF9	PIF8	PIF7	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

EXTI15

EXTI0

Bits 31:23 Reserved, must be kept at reset value.

Bits 22:18 PIFx: Pending interrupt flag on line x (x = 22 to 18)

0: No trigger request occurred

1: Selected trigger request occurred

This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a '1' to the bit.

Bit 17 Reserved, must be kept at reset value.

Bits 16:0 PIFx: Pending interrupt flag on line x (x = 16 to 0)

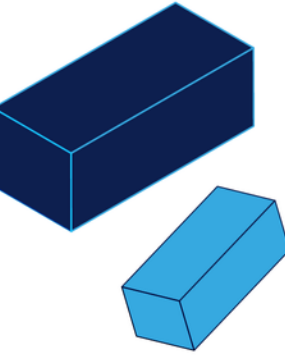
0: No trigger request occurred

1: Selected trigger request occurred

This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a '1' to the bit.

arm

MICRO-
CONTRO-
LADORES
ARM

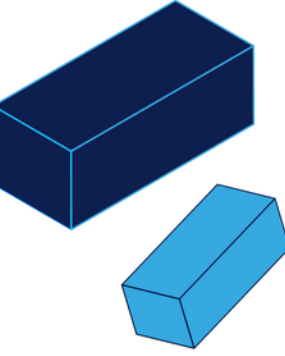


PROGRAMANDO DESDE REGISTROS LA EXTI

arm

MICRO-
CONTRO-
LADORES
ARM

CONFIGURACION DE INTERRUPCION DE UN PIN GPIO



1. El pin deber ser configurado en modo entrada (**GPIOx_MODER**) .
2. Configurar el disparador de borde (FT,RT, FRT) (**EXTI_RTSR**, **EXTI_FTSR**) .
3. Configurar la entrega de la interrupción desde el periférico al procesador(En el periférico) (**EXTI_IMR**) .
4. Asignar un pin a la line EXTIx configurada en el registro **SYSCFG_EXTICRx**
5. Identificar el numero de interrupción (**IRQn**) en el que el procesador acepta la interrupción del pin configurado.
6. Configurar la prioridad de la interrupción(En el procesador) (**NVIC_IPR**) .
7. Habilitar le interrupción (En el procesador) (**NVIC_ISER**) .
8. Implementar el IRQ Handler (ISR) .

arm

MICRO-
CONTRO-
LADORES
ARM

EJEMPLO DE CONFIGURACION

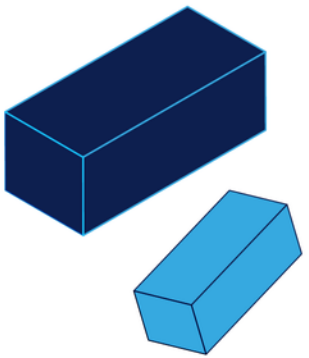
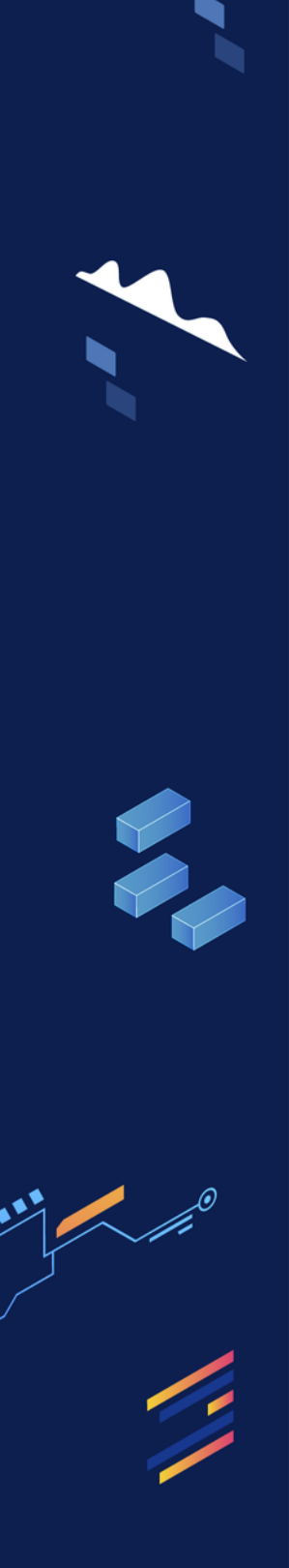
```
void EXTI13_Config(void){
    /*enable clock*/
    RCC->AHB1ENR |= RCC_AHB1ENR_GPIOCEN;
    RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
    /*1. configurar el PC13 como entrada*/
    GPIOC->MODER &=~ GPIO_MODER_MODE13;           //input
    /*2. Configurar el disparador de borde*/
    EXTI->FTSR |= EXTI_FTSR_TR13;                  // 1U<<|3;
    /*3. Configurar la entrega de interrupcion del exti al core*/
    EXTI->IMR |= EXTI_IMR_IM13;
    /*4. Asignar el pin de interrupcion a la linea EXTI13*/
    SYSCFG->EXTICR[3] &=~ SYSCFG_EXTICR4_EXTI13;
    //SYSCFG->EXTICR[3] |= 0x2U<<4;
    SYSCFG->EXTICR[3] |= SYSCFG_EXTICR4_EXTI13_PC;
    /*5. Identificar el IRQn de la linea EXTIx*/
    // EXTI13_IRQn = 40 (EXTI15_10_IRQn)
    /*6. Configurar la prioridad de la interrupcion*/
    NVIC->IP[EXTI15_10_IRQn] = 1U<<4;
    //NVIC->ICPR[EXTI15_10_IRQn>>5U] |= 1U<<(EXTI15_10_IRQn & 0x1FU);
    EXTI->PR |= EXTI_PR_PR13;
    NVIC_ClearPendingIRQ(EXTI15_10_IRQn);
    //NVIC_SetPriority(EXTI15_10_IRQn,1U);
    /*7 Habilitar la interrupcion de la EXTIx*/
    NVIC->ISER[EXTI15_10_IRQn>>5U] |= 1U<<(EXTI15_10_IRQn & 0x1FU);
    //NVIC_EnableIRQ(EXTI15_10_IRQn);
}
```

ISR

```
/******ISR******/
void EXTI15_10_IRQHandler(void){
    if(EXTI->PR & EXTI_PR_PR13){
        EXTI->PR |= EXTI_PR_PR13;
        eventCount++;
        GPIOA->ODR ^= GPIO_ODR_OD5;
    }
}
```

arm

MICRO-
CONTRO-
LADORES
ARM



EJEMPLO: CONTADOR DE EVENTOS MEDIANTE INTERRUPCIONES EXTERNAS

arm

MICRO-
CONTRO-
LADORES
ARM

UMAKER | CENTRO DE CAPACITACIÓN
DE DESARROLLO TECNOLÓGICO