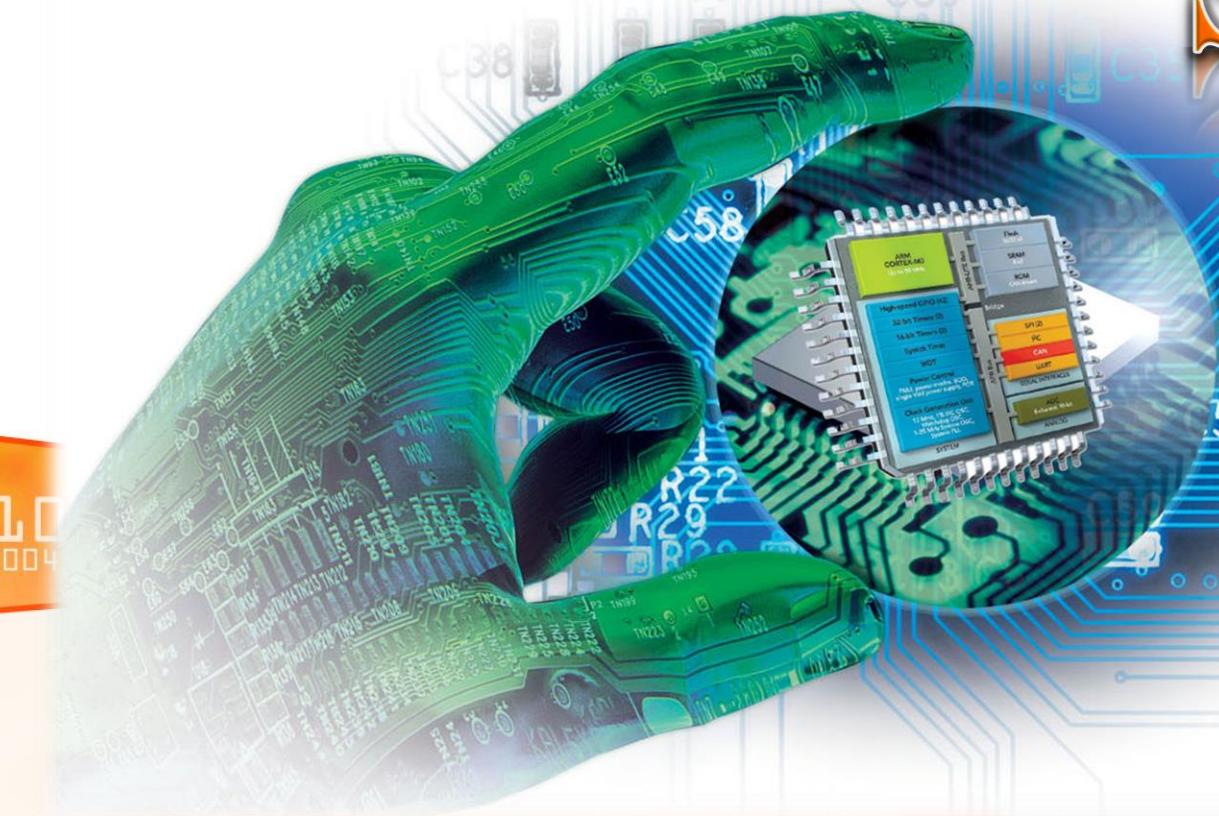


# مُتَحَكِّمَات

# STM32

3



# موجو عاٹن المعاڑة:

- Exceptions Overview
- Micro-Coded Interrupts
- Exception Types
- System Exceptions
- متڪم NVIC
- أنماط عمل المعالج
- مصطلاح الـ Preemption
- Interrupt Lifecycle
- Configure Interrupts

# نظرة عامة عن مفهوم الاستثناءات: Exceptions Overview

- المقاطعة هي حدث غير متزامن يتسبب في إيقاف تنفيذ الكود الحالى اعتمادا على مبدأ الأولوية
- تنشأ المقاطعات من خلال الـ hardware أو من البرنامج نفسه الـ software
- يتم التحكم بالمقاطعات من خلال الـ NVIC(Nested Vectored Interrupt Controller)
- توفر معالجات ARM هيكيلية خاصة للتعامل مع الاستثناءات/ المقاطعات وتدعى Micro-Coded Architecture حيث يتم تكديس المقاطعات ثم البدء بمعالجتها حسب أولويتها ثم العودة لاستكمال تنفيذ البرنامج الرئيسي وكل ذلك يتم Hardware أي بدون التدخل من قبل المعالج مما يعني سرعة استجابة أعلى للاستثناءات وتخفيض الضغط عن المعالج بالإضافة إلى مرونة أعلى في العمل والقدرة على دعم أنظمة الزمن الحقيقي RTOS

# نظرة عامة عن مفهوم الاستثناءات: Exceptions Overview

يتم الدخول إلى المقاطعة والخروج منها عن طريق الـ **Hardware** من خلال الخطوات التالية:

- ❑ حفظ واستعادة (processor context) وتشمل حالة المعالج لحظة حدوث المقاطعة بما في ذلك حالة وقيم المسجلات ) عند حدوث المقاطعة وعند العودة منها لاستكمال تنفيذ البرنامج الرئيسي انطلاقاً من التعليمة التي تم التوقف عنها عند حدوث المقاطعة.
- ❑ يُسمح باكتشاف الوصول المتأخر للمقاطعات/ الاستثناءات وخدمتها بناءً على مستوى الأولوية لها.
- ❑ يُسمح بخدمة المقاطعة المعلقة التالية عند الانتهاء من خدمة المقاطعة الحالية دون الحاجة لإعادة مرحلة (حفظ/ استعادة) حالة المعالج (processor context) مما يزيد من سرعة الاستجابة للمقاطعات وتدعي هذه الخاصية بـ **tail-chaining**

# أنواع الاستثناءات: Exception Types

هناك نوعين من الاستثناءات هما:

□ يتم توليدها من قبل المعالج نفسه داخلياً system exception

□ المقاطعات وتأتي من العالم الخارجي للمعالج interrupts

هناك 15 استثناء من نوع system exception تدعمها معالجات CORTEX-M بالإضافة إلى 240 مقاطعة ، لذا فإن معالجات CORTEX-M تدعم 255 استثناء، بحيث:

□ الاستثناء رقم واحد هو لا RESET

□ فقط 9 استثناءات من نوع system exception يتم التعامل معها والـ 6 الباقية هي للاستخدامات المستقبلية.

□ الاستثناء رقم 16 هو للمقاطعة رقم 1 IRQ1).

# System Exceptions

RESET هو استثناء من نوع system exception، يتم طلب هذا الاستثناء عند تغذية المعالج أو من خلال الضغط على زر الـ **Reset**

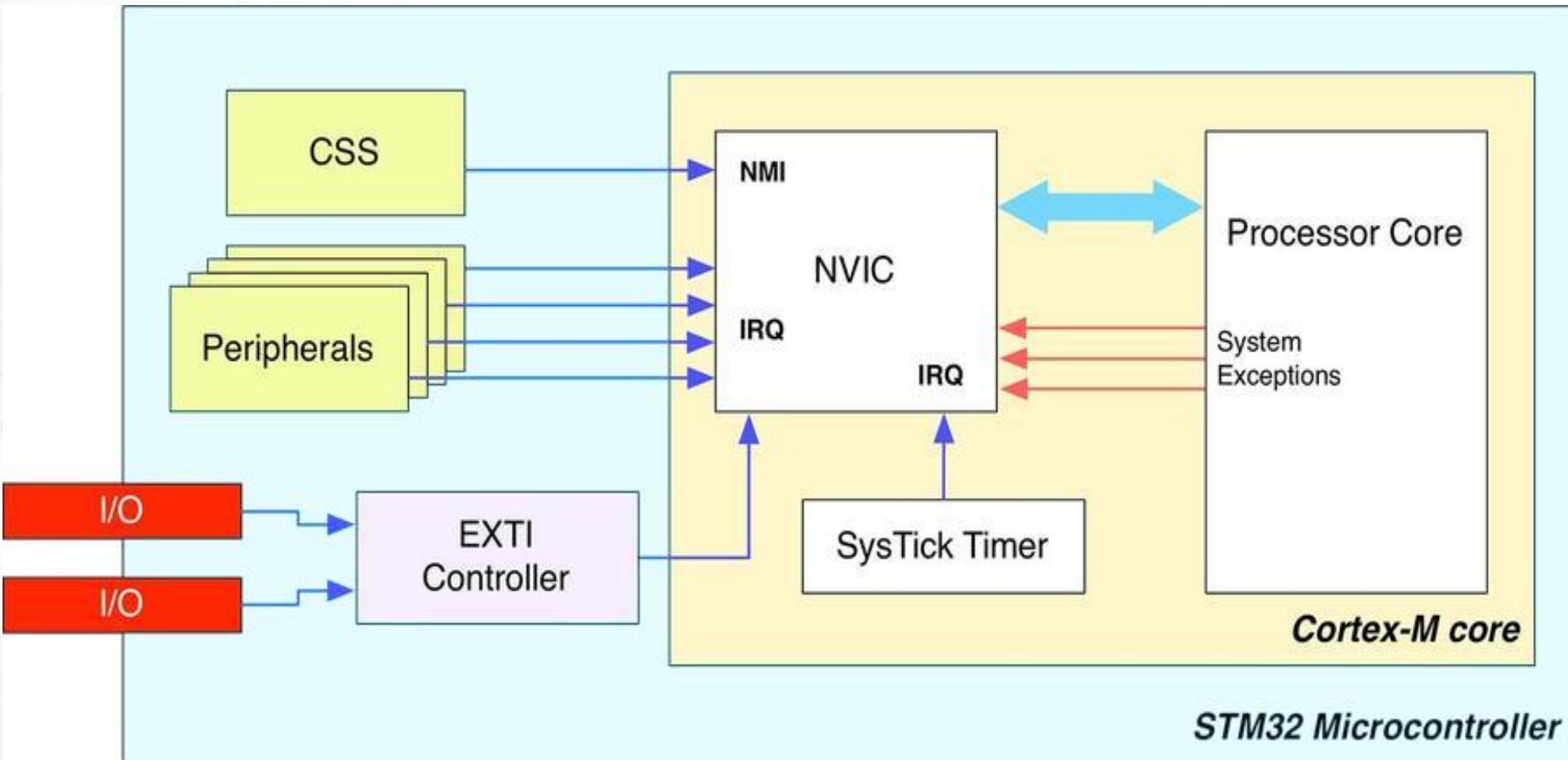
**Non-Maskable Interrupt (NMI)** : هي الاستثناءات التي لا يمكن إلغاء تفعيلها، يتم قدح هذا الاستثناء عند حدوث خطأ ما عند تنفيذ أحد Exception Handler، ولها أعلى أولوية بعد استثناء الـ Reset، في متحكمات STM32 يتم ربط استثناءات الـ NMI مع نظام حماية الساعة Clock security حيث الـ CSS هي وحدة طرفية تقوم بفحص حالة الساعة الخارجية HSE وفي حال اكتشاف مشكلة ما فيها تقوم بتعطيل HSE وتفعيل الساعة الداخلية HSI.

# متحكم NVIC (Nested Vectored Interrupt Controller)

- هو المتحكم المسؤول عن تحديد أولويات المقاولات Interrupt
- هي عبارة عن الاستثناءات التي يتم توليدها من قبل IRQ(الطرفيات أو من قبل البرنامج
- وتحسين أداء المعالج MCU وتقليل زمن استجابة المقاولة
- يوفر NVIC أيضا خطوات محددة للتعامل مع المقاولات التي تحدث أثناء تنفيذ مقاولات أخرى أو عندما يكون المعالج في مرحلة استعادة حاليه السابقة واستئناف عمليته المعلقة التي توقف عندها بسبب حدوث المقاولة

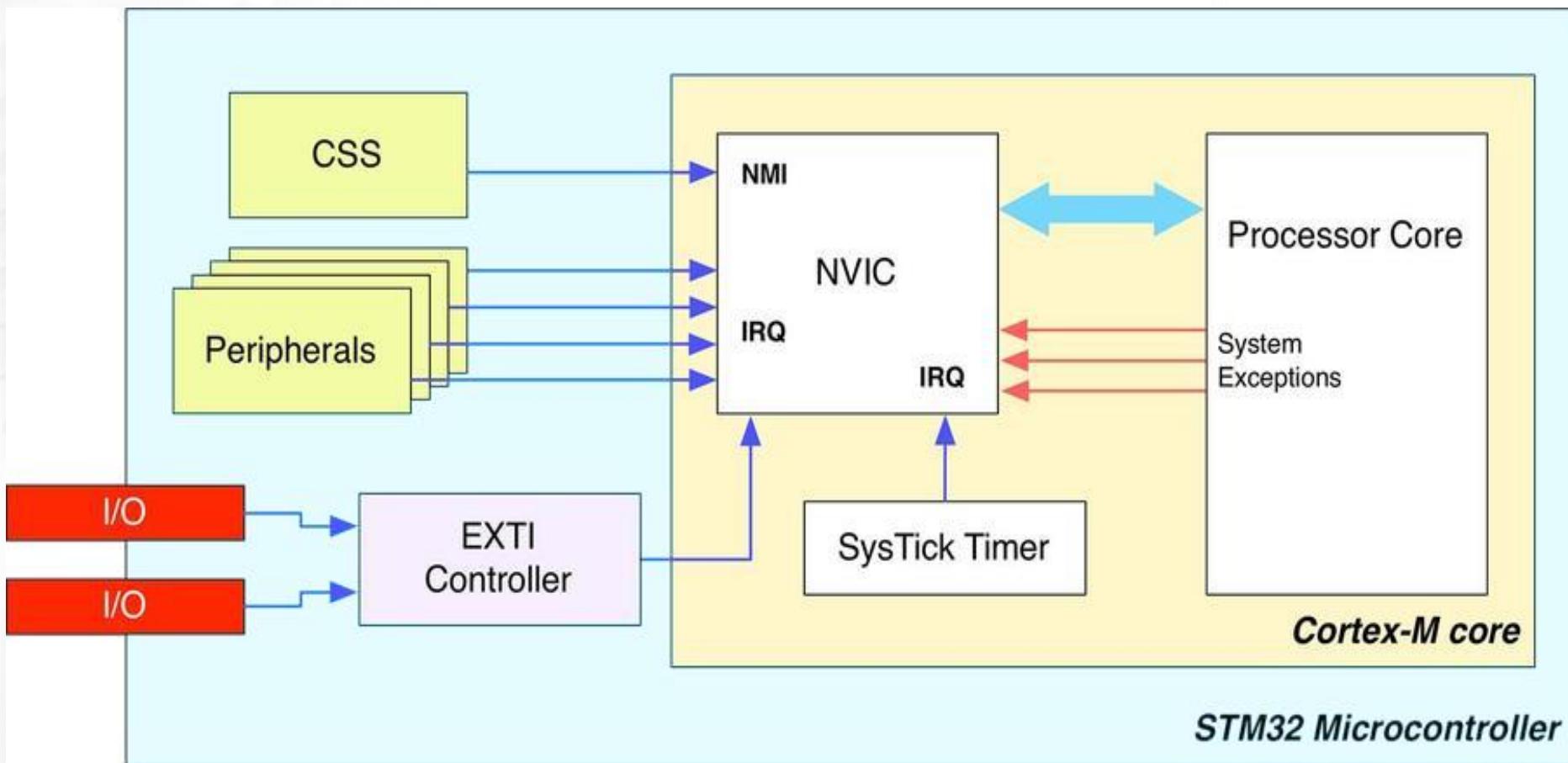
# متحكم NVIC (Nested Vectored Interrupt Controller)

يتم ربط مقاطعة مع Clock Security System (CSS)  خطوط مقاطعات الـ Non-Maskable Interrupt وهي المقاطعات التي لا يمكن تجاهلها أو إلغاء تفعيلها (NMI)



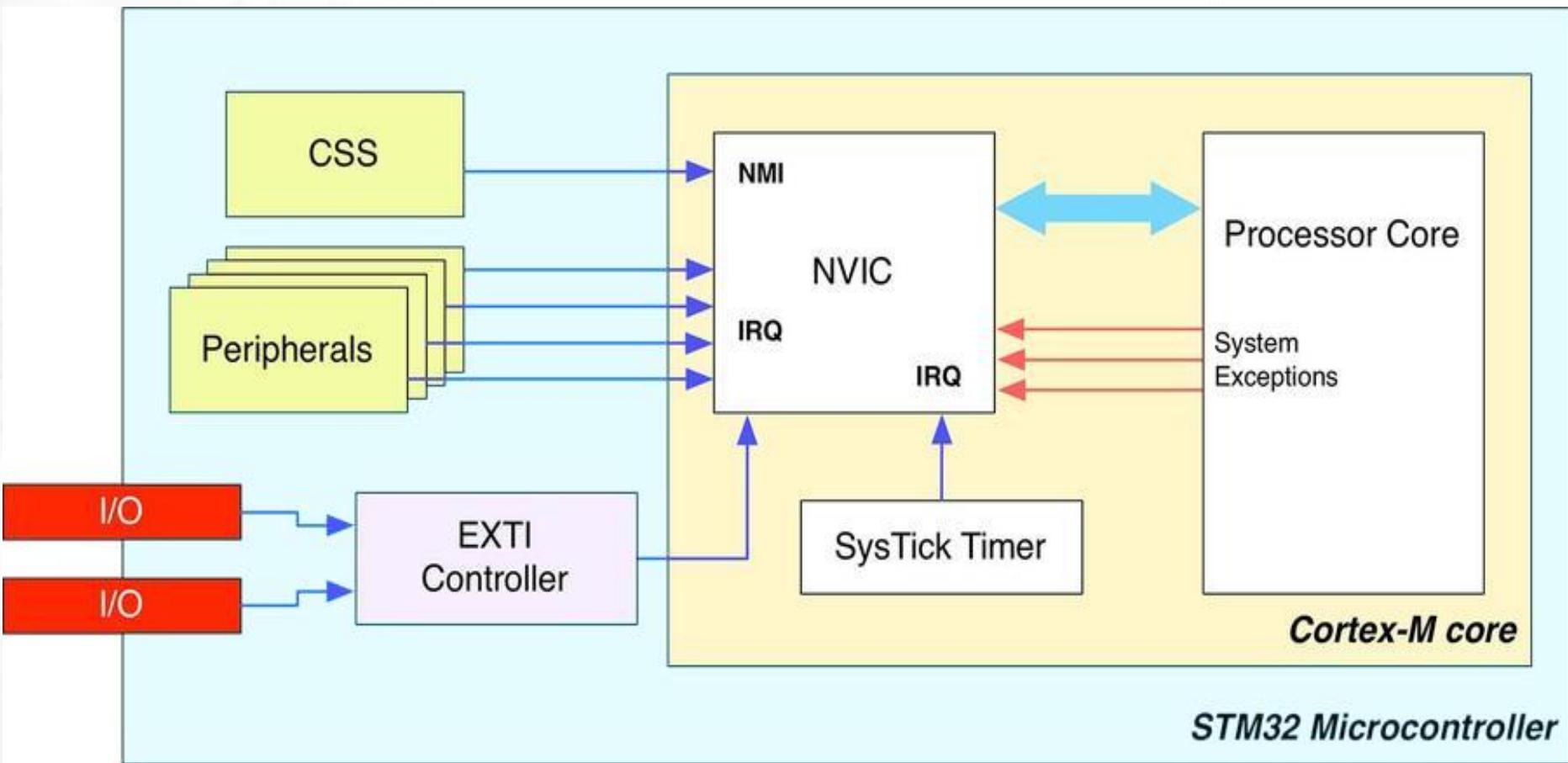
# متحكم NVIC (Nested Vectored Interrupt Controller)

□ يتم ربط مقاطعات الطرفيات (مثل مقاطعات المؤقتات، المبدلات التشابهية رقمية وغيرها من الطرفيات) مع خطوط Interrupt Requests (IRQ)



# متحكم NVIC (Nested Vectored Interrupt Controller)

المقاطعات الخارجية القادمة من الـ **GPIO** يتم ربطها مع **External Interrupt/Event Controller (EXTI)** ليتم بعد ذلك ربطها مع خطوط **IRQ**، كما هو موضح بالشكل التالي



# جدول أشعة المقاطعة interrupts table Vectors

Number	Exception Type	Priority	Function
1	Reset	-3	Reset
2	NMI	-2	Non-Maskable Interrupt
3	Hard Fault	-1	All faults that hang the processor
4	Memory Fault	Configurable	Memory issue
5	Bus Fault	Configurable	Data bus issue
6	Usage Fault	Configurable	Data bus issue
7 ~ 10	Reserved	—	Reserved
11	SVCall	Configurable	System service call (SVC instruction)
12	Debug	Configurable	Debug monitor (via SWD)
13	Reserved	—	Reserved
14	PendSV	Configurable	Pending request for System Service call
15	SysTick	Configurable	System Timer
16 ~ 240	IRQ	Configurable	Interrupt Request

# المقاطعات الخارجية External Interrupts

- يحتوي متحكم STM32 على 16 خط للمقاطعات الخارجية هي من الخط 0 حتى الخط 15
- تشير أرقام الخطوط إلى أرقام اطراف ال-GPIOS
- هذا يعني أن الطرف PA0 متصل بالخط LINE0 والطرف PA13 متصل بالطرف LINE13، أيضاً PC0 و PB0 متصلين بالخط LINE0، بمعنى أن جميع الأطراف ذات الأرقام متصلة بالخط (LINE0 حيث يعبر X عن اسم المنفذ) ، أيضاً جميع الأطراف ذات الأرقام PX3 متصلة بالخط LINE3 وهكذا...
- في كل خط من خطوط المقاطعات الخارجية (كل مجموعة) يتحقق أن واحد pin يقوم بتوليد المقاطعة وعلى البرنامج أن يكتشف أي pin قام بتوليد المقاطعة

# المقاطعات الخارجية External Interrupts

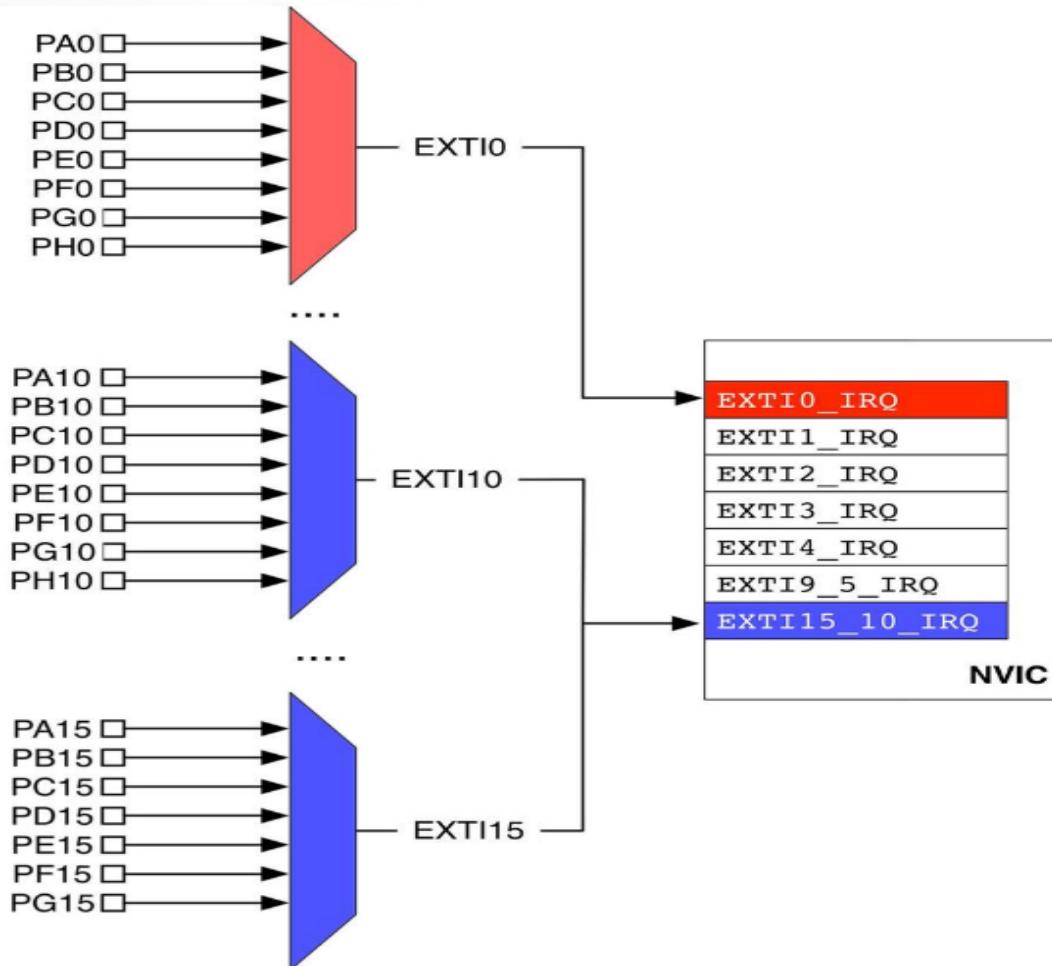
يجب الانتباه للملاحظات التالية:

☐ الأطراف LINE0... PA0, PB0, PC0... جميعها متصلة بالخط PA0 لذا

في اللحظة الواحدة

بإمكانك توليد مقاطعة على طرف واحد فقط من هذه الأطراف.

☐ الأطراف PA5, PA0 متصلين على خطين مختلفين من خطوط المقاطعة لذا يمكنك استخدامهم في نفس اللحظة لتوليد المقاطعة.



# المقاطعات الخارجية External Interrupts

يمكن قذح المقاطعة عند الجبهة الصاعدة أو الهابطة أو عند كليهما

`GPIO_MODE_IT_RISING`



`GPIO_MODE_IT_RISING_FALLING`

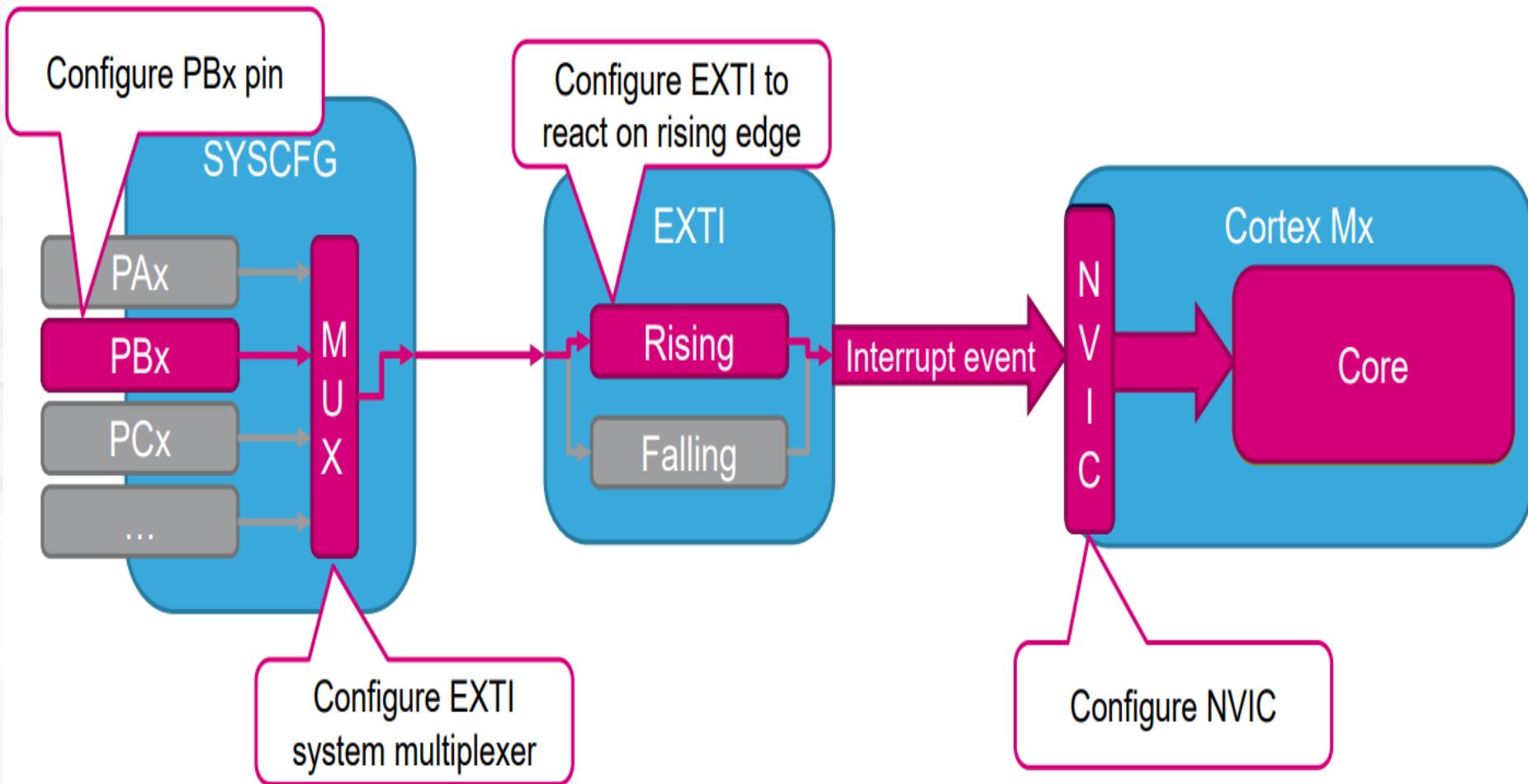


`GPIO_MODE_EVT_FALLING`



# المقاطعات الخارجية External Interrupts

فتحدث المقاطعة الخارجية وفقاً للآلية التالية:



# أنماط عمل المعالج Processor mode

للمعالج نمطي عمل أساسين:

في Thread Mode : يكون المعالج في نمط Thread Mode

الوضع الطبيعي له عند تنفيذ الكود أو عند عمل him reset

Handler Mode : يدخل المعالج في نمط Handler Mode

عند حدوث استثناء / مقاطعة حيث يتم حفظ المكان الذي تم توقف

المعالج عنده كما يتم حفظ حالة المسجلات والأعلام flags بشكل

آلي وعن طريق hardware ليضمن استجابة سريعة لحدث

المقاطعة

# Exception Behavior

عند حدوث استثناء ما، يتم إيقاف تنفيذ التعليمات الحالية ويتم توجيه المعالج لجدول أشعة المقاطعة حيث:

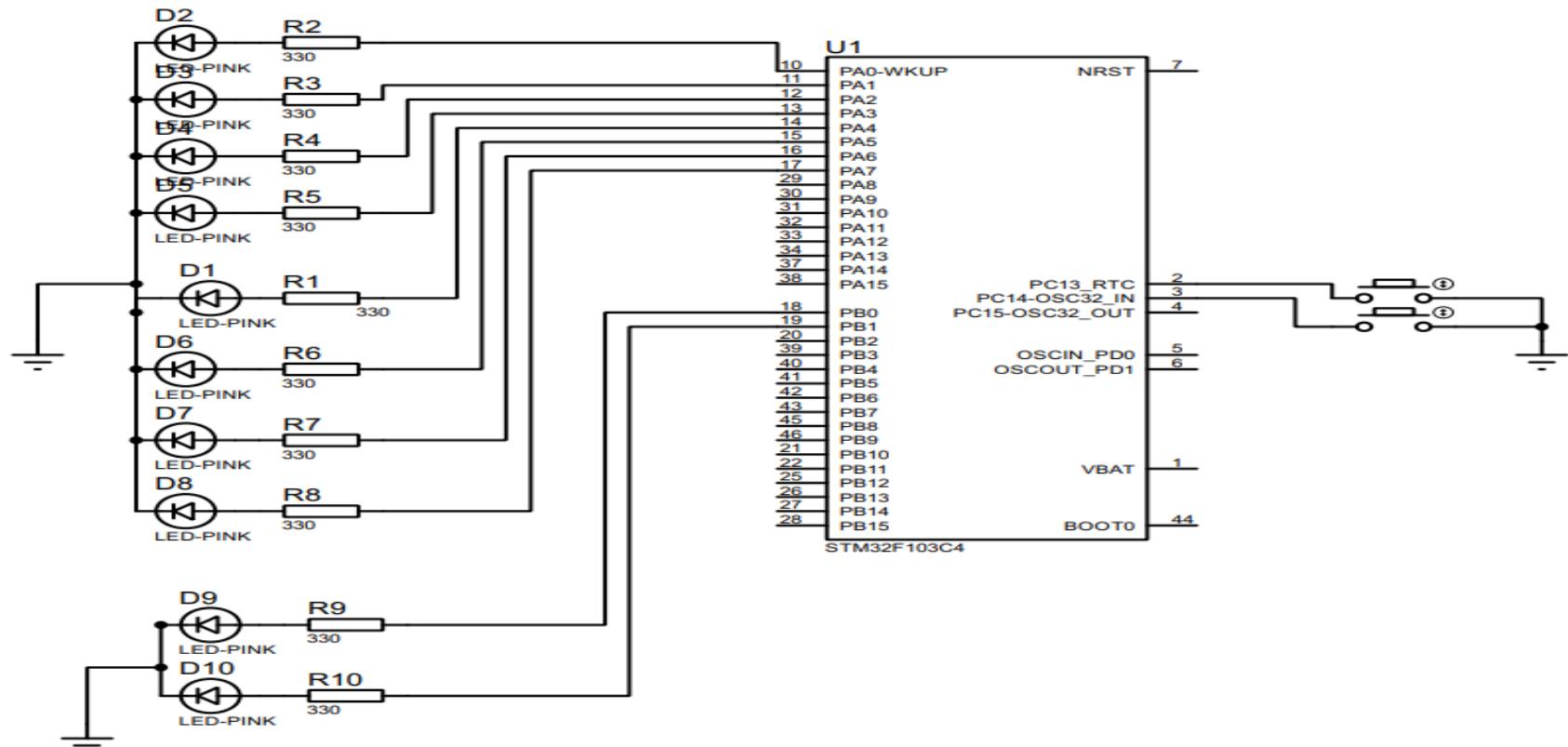
- يقوم المعالج بإنهاe التعليمية الحالية التي يقوم بتنفيذها طالما أنها لا تحتاج لأكثر من دورة تعليمية
- يتم حفظ حالة المعالج (عنوان التعليمية التي وصل لعندتها حالياً) إلى المكدس
- يتم تحميل عنوان برنامج خدمة المقاطعة الحاصلة من جدول أشعة المقاطعة
- يتم البدء بتنفيذ برنامج خدمة المقاطعة ISR، ويستغرق تنفيذ كامل هذه العملية 12 cycles في معالجات
- يقوم برنامج خدمة المقاطعة بتصغير علم المقاطعة التي قامت باستدعائه

# Exception Behavior

- في نهاية برنامج خدمة المقاطعة يتم التأكيد من عدم وجود مقاطعة في الانتظار من أجل الانتقال لتنفيذ برنامج خدمة المقاطعة التالية دون استعادة حالة المعالج السابقة بهدف زيادة سرعة الاستجابة للمقاطعة.
- يتم تنفيذ تعليمات **EXC\_RETURN** لاستعادة حالة المعالج قبل حدوث المقاطعة
- تستغرق العودة من المقاطعة (استعادة حالة المعالج) في معالجات ARM Cortex-M3/M4 حوالي 10 clock cycles

# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجية

□ زر طوارئ لتفعيل برنامج الطوارئ حيث يتم تنفيذه بغض النظر عن عمل النظام (إنارة اللدات) و بعد الانتهاء من برنامج خدمة المقاطعة (الطوارئ) يعود للبرنامج الرئيسي ليتابع عمله



# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

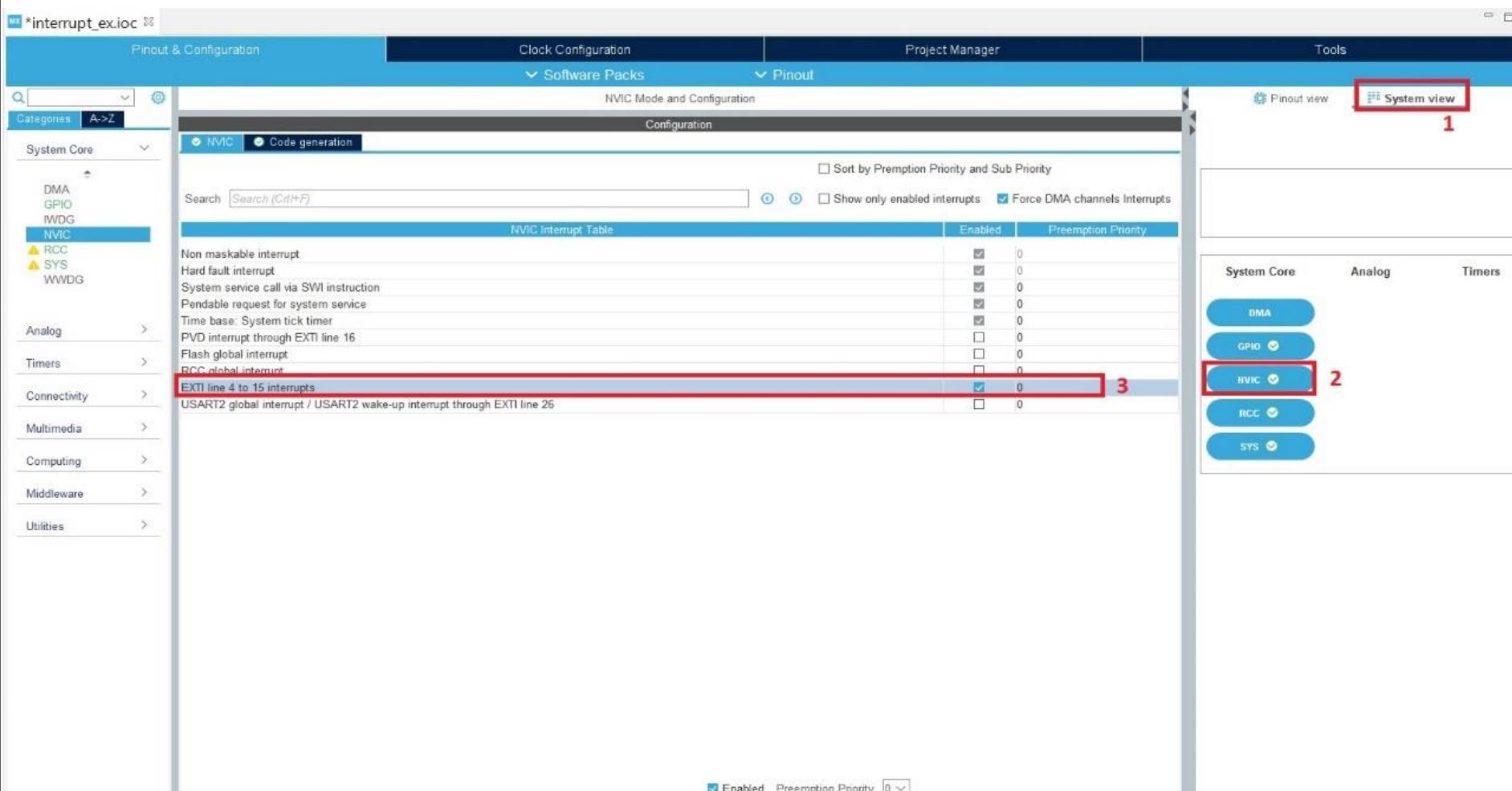
نقوم بضبط الأقطاب PB0:PB1، والأقطاب PA0:PA6 كأقطاب  
خرج ، والقطب PC13 كقطب دخل مع تفعيل مقاومة الرفع  
الداخلية، والقطب PC14 كقطب مقاطعة خارجية

The screenshot shows the STM32CubeMX software interface. On the left, there's a navigation bar with categories like DMA, GPIO, IWDG, NVIC, RCC, SYS, and WWDG. The main area has tabs for Pinout & Configuration, Clock Configuration, Project Manager, and Tools. The Pinout & Configuration tab is active, displaying a table of pins and their configurations. The table includes columns for Pin Name, Signal on Pin, GPIO output level, GPIO mode, GPIO Pull-up/Pull-down, Maximum output, User Label, and Modified. Rows include PA0-WKUP, PA1, PA2, PA3, PA4, PA5, PA6, PB0, PB1, PC13-TAMPER-..., and PC14-OSC32\_IN. The PC13-TAMPER-... row is selected, showing Input mode, Pull-up, and int\_button under User Label. To the right, a detailed diagram of the STM32F103C4Tx microcontroller package is shown, highlighting pins PA3, PA4, PA5, PA6, PA7, PB0, PB1, PB2, PB10, PB11, VSS, and VDD. Pins PA0, PA1, and PA2 are also labeled as GPIO\_Output.

Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output	User Label	Modified
PA0-WKUP	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PA1	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PA2	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PA3	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PA4	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PA5	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PA6	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PB0	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PB1	n/a	Low	Output Push Pull	No pull-up and n...	Low		
PC13-TAMPER-...	n/a	n/a	Input mode	Pull-up	n/a	int_button	<input checked="" type="checkbox"/>
PC14-OSC32_IN	n/a	n/a	External Interrupt...	Pull-up	n/a	int_button	<input checked="" type="checkbox"/>

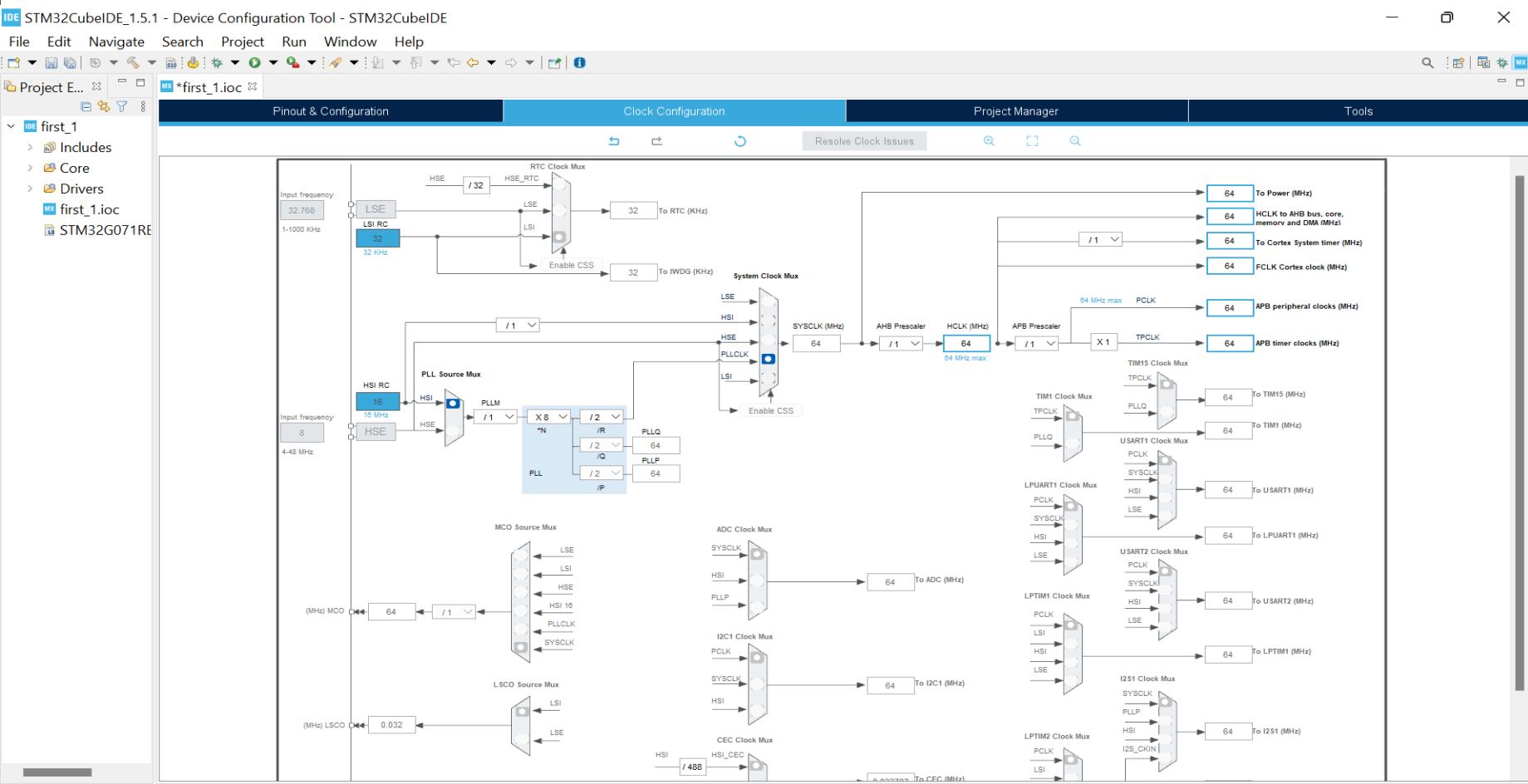
# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجية

قم بفتح NVIC Tab ثم قم بتفعيل المقاولات الخارجية، يمكنك أيضاً إعادة ضبط مستويات الأولوية للمقاولات:



## التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

## نقوم بضبط تردد الساعة للمتحكم



# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

□ نقوم بالضغط على Project...Generate أو من Ctrl+s، ليتم حفظ المشروع وتوليد الكود ثم نقوم بإضافة الجزء التالي:

```
#include "main.h"

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin == GPIO_PIN_14)
        {HAL_GPIO_WritePin(GPIOB,
GPIO_PIN_0|GPIO_PIN_1 , GPIO_PIN_SET);
}
```

برنامج خدمة المقاطعة

# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

```
int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
    while (1)
    {
        if(HAL_GPIO_ReadPin(GPIOC,
GPIO_PIN_13)==0)
        {
            GPIOA->ODR = 0X0001;
            HAL_Delay(50);
        }
        //*****
    }
}
```

# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

```
GPIOA->ODR = 0X0002;
```

```
HAL_Delay(50);
```

```
//*****
```

```
GPIOA->ODR = 0X0004;
```

```
HAL_Delay(50);
```

```
//*****
```

```
GPIOA->ODR = 0X0008;
```

```
HAL_Delay(50);
```

```
//*****
```

```
GPIOA->ODR = 0X0010;
```

```
HAL_Delay(50);
```

```
//*****
```

# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

```
GPIOA->ODR = 0X0020;  
HAL_Delay(50);  
//*****  
GPIOA->ODR = 0X0040;  
HAL_Delay(50);  
//*****  
GPIOA->ODR = 0X0080;  
HAL_Delay(100);  
{}
```

# التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

else

{

GPIOA->ODR = 0X0000;

}

}

}

Thank you for listening