

مُتَحَكِّمَات

STM32

4+



موضوعات المحاضرة:

- Exceptions Overview**
- Micro-Coded Interrupts**
- Exception Types**
- System Exceptions**
- منحكم 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 المتحكم Interrupt Controller) NVIC(Nested Vectored
- 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
- المقاولات 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 الباقي 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.

System Exceptions

□ يتم توليد هذا الاستثناء عند حدوث استثناء ما وفي حال عدم وجود آلية للتعامل مع هذا الاستثناء، على سبيل المثال عند حدوث استثناء معين وفي حال عدم تفعيل Exception Handler الخاص به يتم في هذه الحالة توليد استثناء الـ Hard Fault.

□ يحدث هذا الاستثناء عند محاولة الوصول لموقع غير مصرح بالوصول له في الذاكرة أو عندما يتم انتهاك قاعدة من قواعد حماية الذاكرة أثناء تنفيذ التعليمات البرمجية.

□ يحدث هذا الاستثناء عند حدوث خطأ أثناء الوصول إلى ذاكرة المعطيات أو التعليمات وقد يكون بسبب حدوث خطأ على الناقل Bus في النظام الذاكري.

System Exceptions

□ **Usage Fault** يحدث هذا الاستثناء أثناء تنفيذ التعليمات وقد يكون بسبب استخدام تعليمية غير موجودة أو عند محاولة الوصول لموقع غير مصرح به أو بسبب خطأ في الحالة الناتجة عن تنفيذ التعليمية مثلاً الحصول على قيمة غير صالحة مثل القسمة على صفر.

□ **SVC Call** يتم توليد هذا الاستثناء **System Service Call** عند استدعاء تعليمية **Supervisor Call** وهذه التعليمية تستخدم من قبل نظام الزمن الحقيقي **Real Time Operation (RTC)**.

□ **Debug Monitor** يتم توليد هذا الاستثناء عند استخدام ميزة اكتشاف الأخطاء البرمجية وعندها يكون المعالج في نمط **Monitor Debug Mode**

□₈ **PendSV** وهو استثناء آخر يتعلق بالـ **RTOS**

System Exceptions

□ **System tick** هذا الاستثناء يستخدم دائمًا في أنظمة الزمن الحقيقي RTOS، حيث يحتاج كل نظام RTOS إلى مؤقت لجدولة أنشطة النظام بحيث يقوم بـتوليد مقاطعة في كل فترة زمنية لترك تنفيذ المهمة الحالية وتنفيذ مهمة أخرى ويكون هذا المؤقت متصل داخلياً بنواة CORTEX-M وليس وحدة خارجية كباقي الطرفيات.

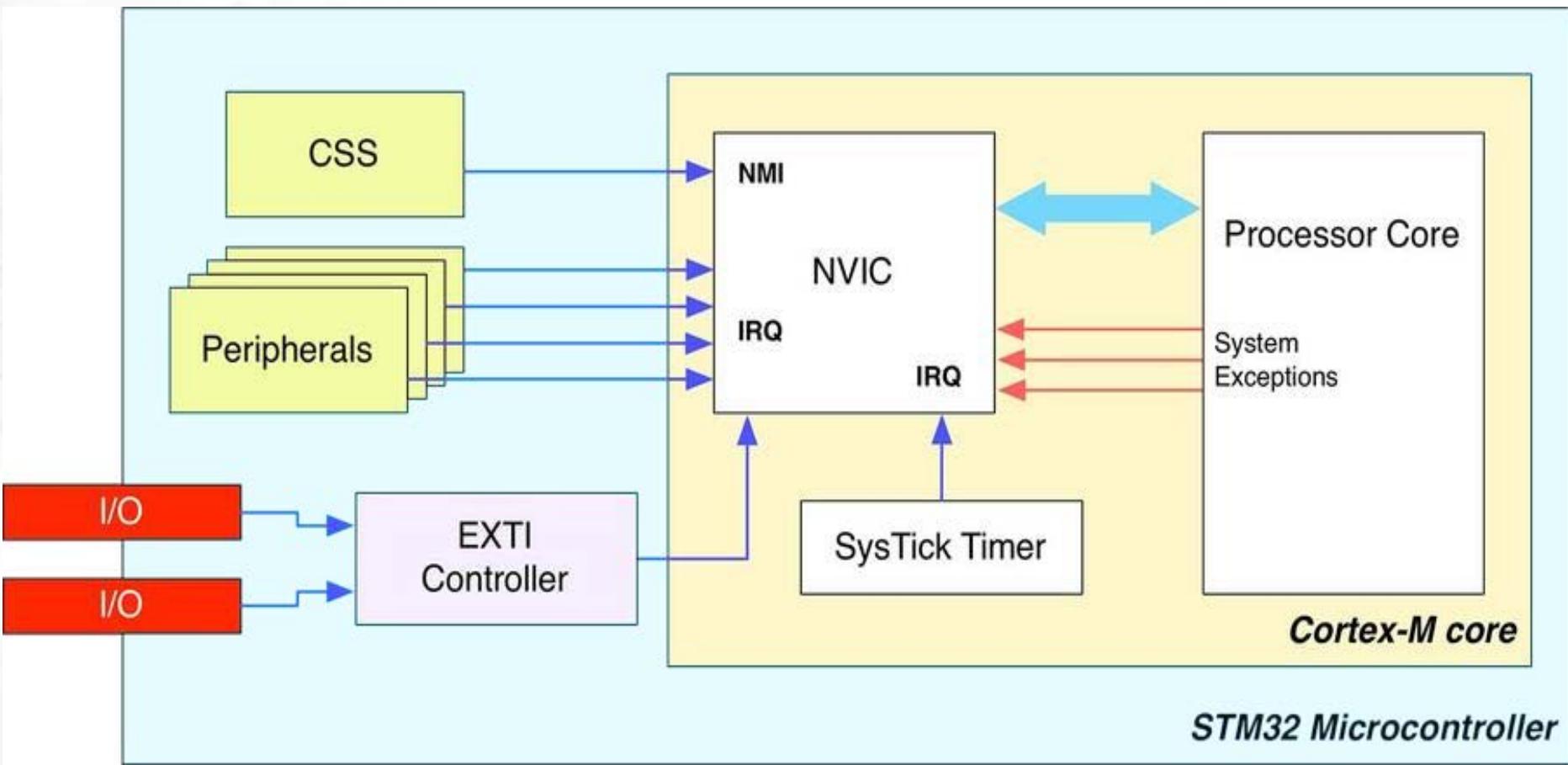
□ **Interrupt (IRQ)** هي عبارة عن الاستثناءات التي يتم توليدها من قبل الطرفيات أو من قبل البرنامج

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

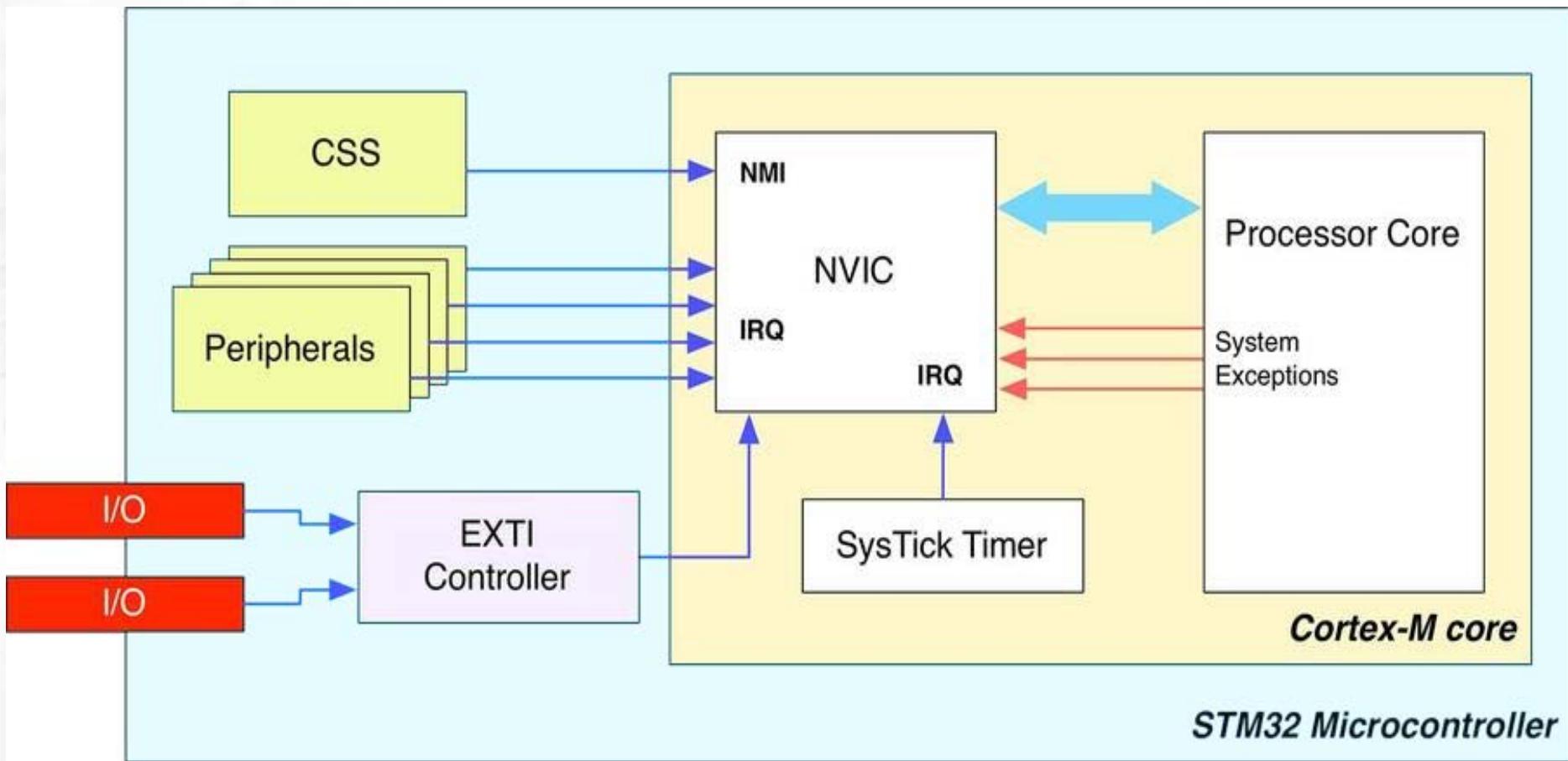
متحكم NVIC (Nested Vectored Interrupt Controller)

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



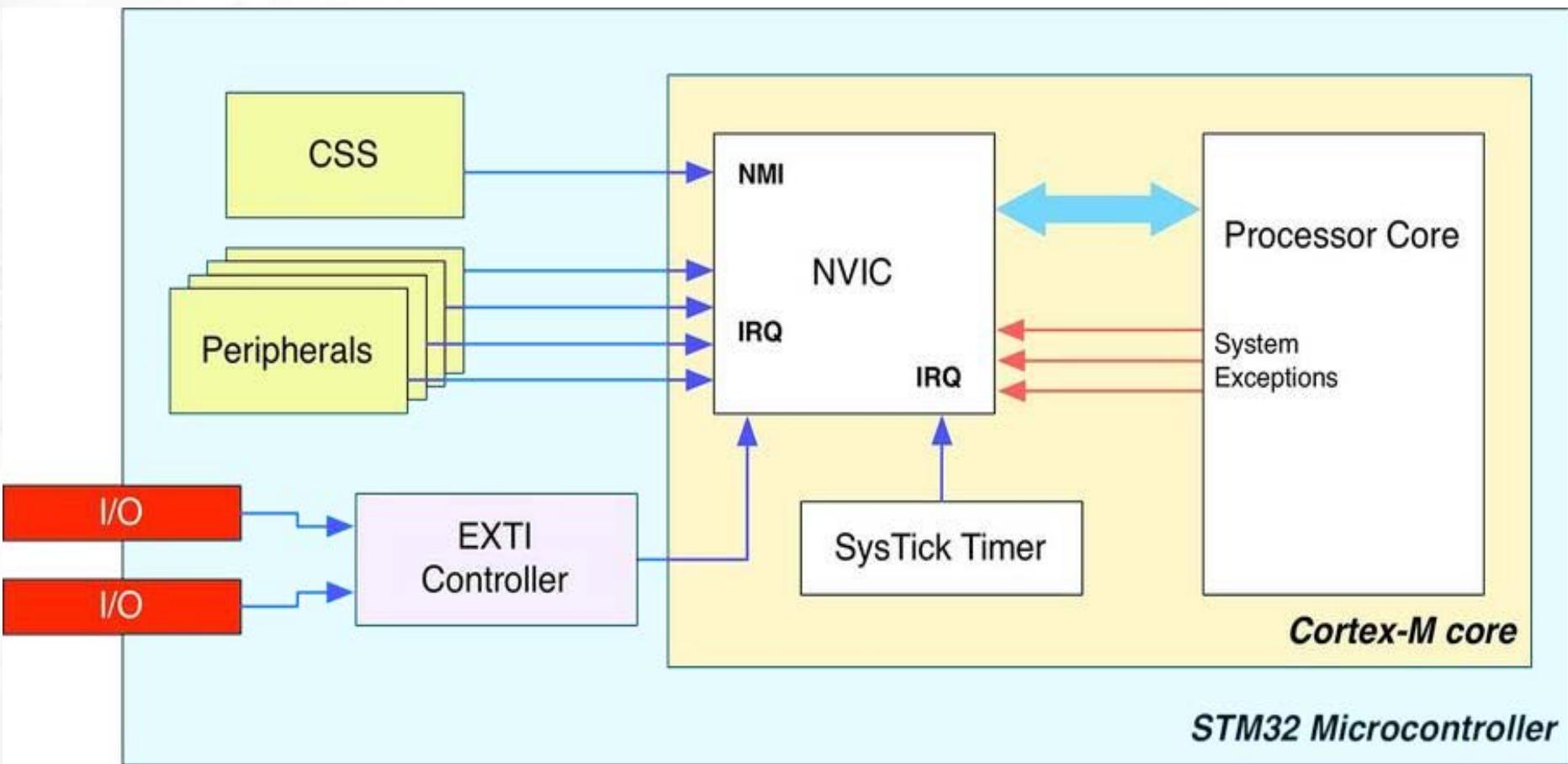
متحكم NVIC (Nested Vectored Interrupt Controller)

□ 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 بعدها 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

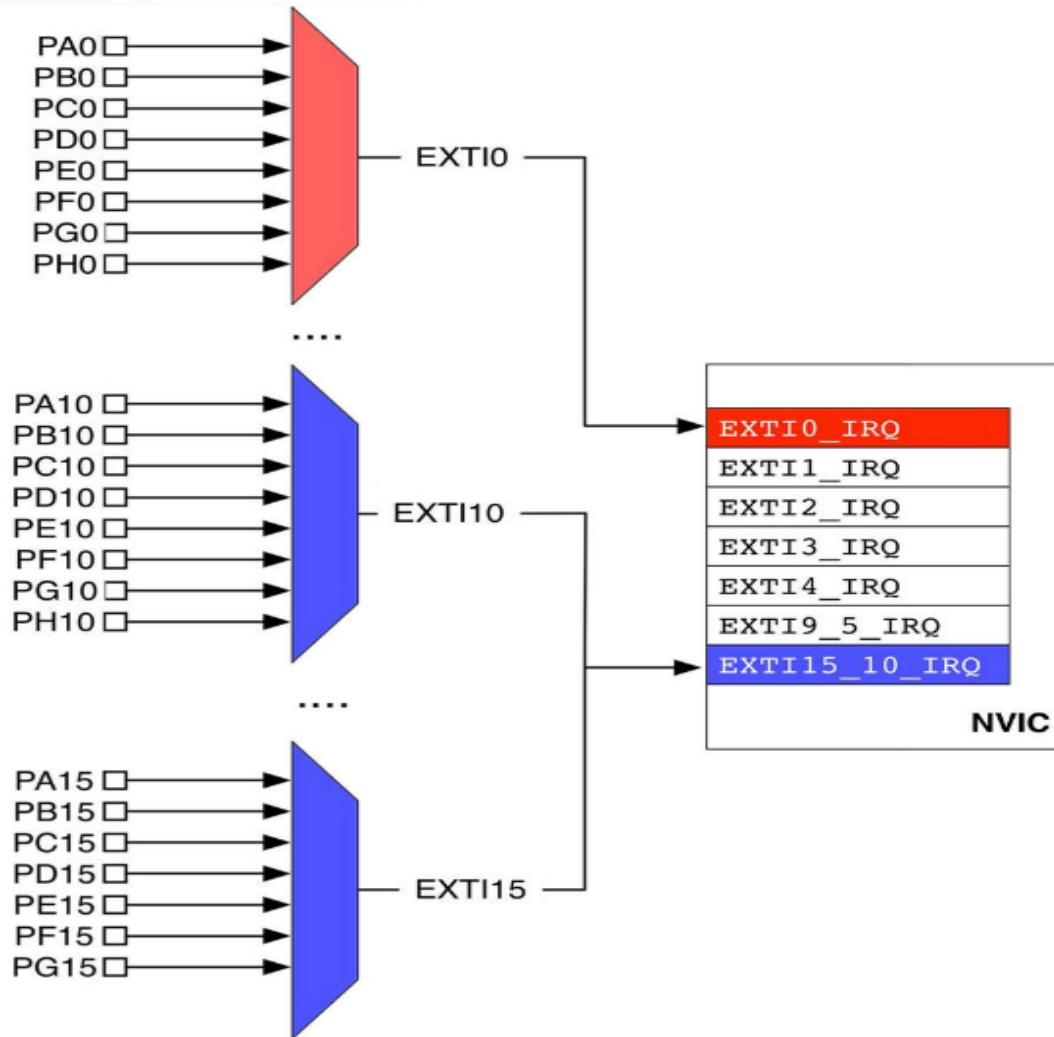
يجب الانتباه للملاحظات التالية:

□ الأطراف LINE0... PA0, PB0, PA0... 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 falling edge أو عند كليهما الهاابطة

GPIO_MODE_IT_RISING



GPIO_MODE_IT_RISING_FALLING

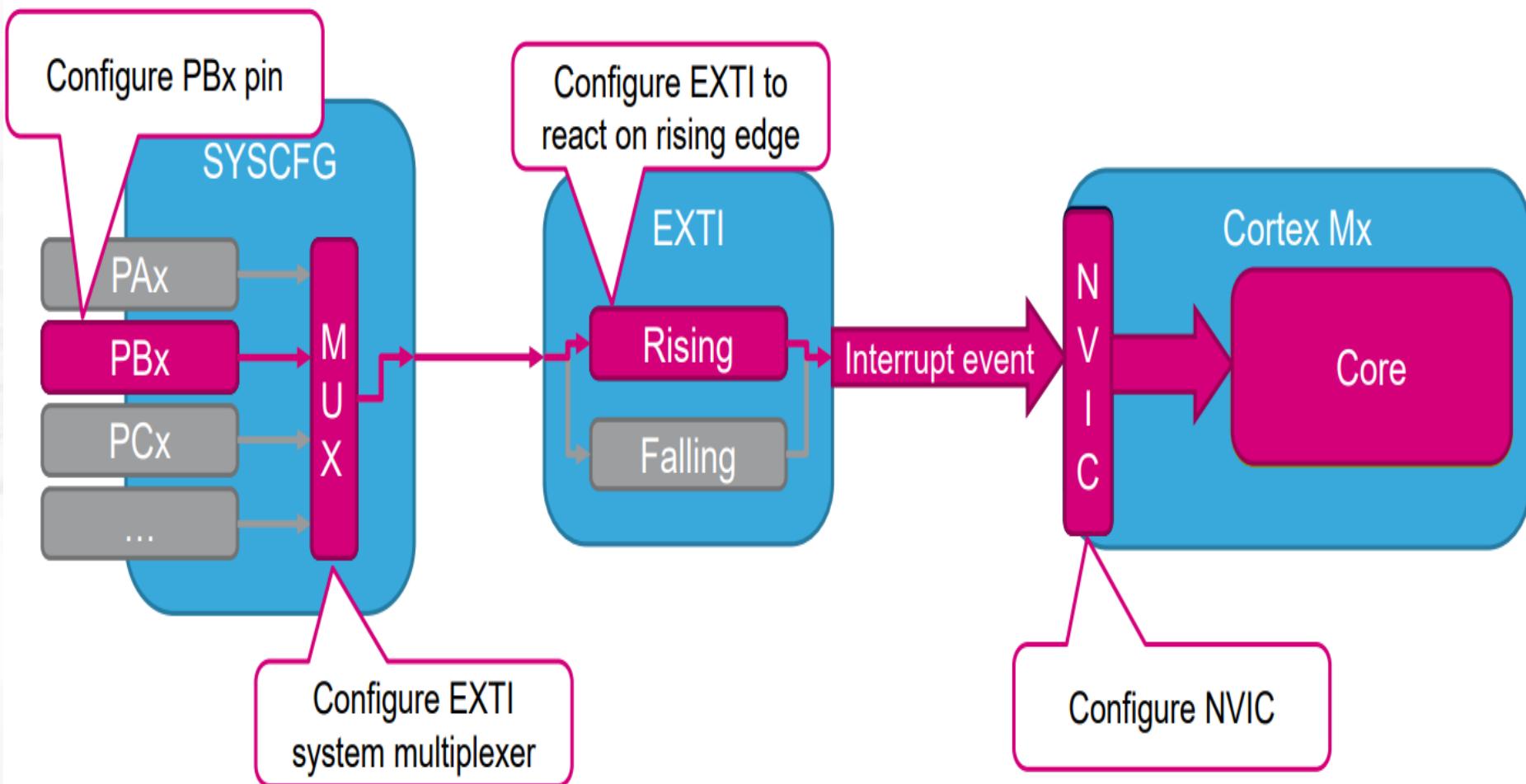


GPIO_MODE_EVT_FALLING



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 reset له

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

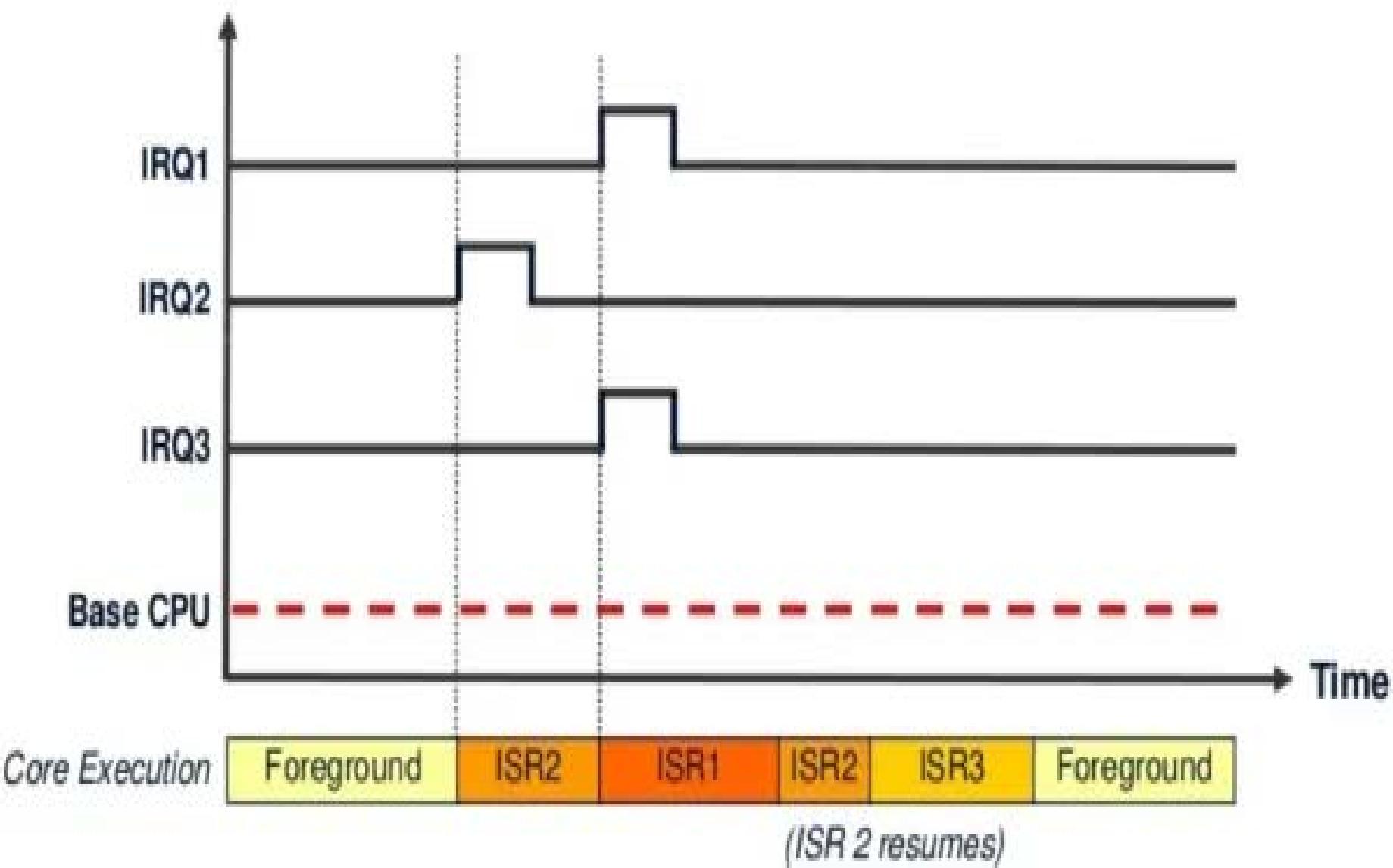
آلي وعن طريق hardware ليضمن استجابة سريعة لحدث
المقاطعة

مصطلاح Preemption

- مصطلح Preemption يعني مقاطعة عمل task أو مهمة معينة بسبب حدوث task أو مهمة أخرى لها أولوية أعلى في التنفيذ وفي هذه الحالة فإن الـ task التي تمت مقاطعتها تسمى preempted
- فعند حدوث عدة مقاطعات أو استثناءات يتم تعليقها جميعاً وتنفيذ الاستثناء أو المقاطعة ذات الرقم الأصغر في جدول أشعة المقاطعات واثناء تنفيذ هذه الاستثناء / المقاطعة من قبل المعالج وفي حال حدوث استثناء/ مقاطعة ذات أولوية أعلى يتم عمل المقاطعة الحالية والذهاب لتنفيذ المقاطعة ذات الأولوية الأعلى، حيث المقاطعة ذات الرقم الأصغر في جدول أشعة المقاطعات لها أولوية أعلى في التنفيذ من قبل المعالج

Preemption مصطلح

Higher Priority

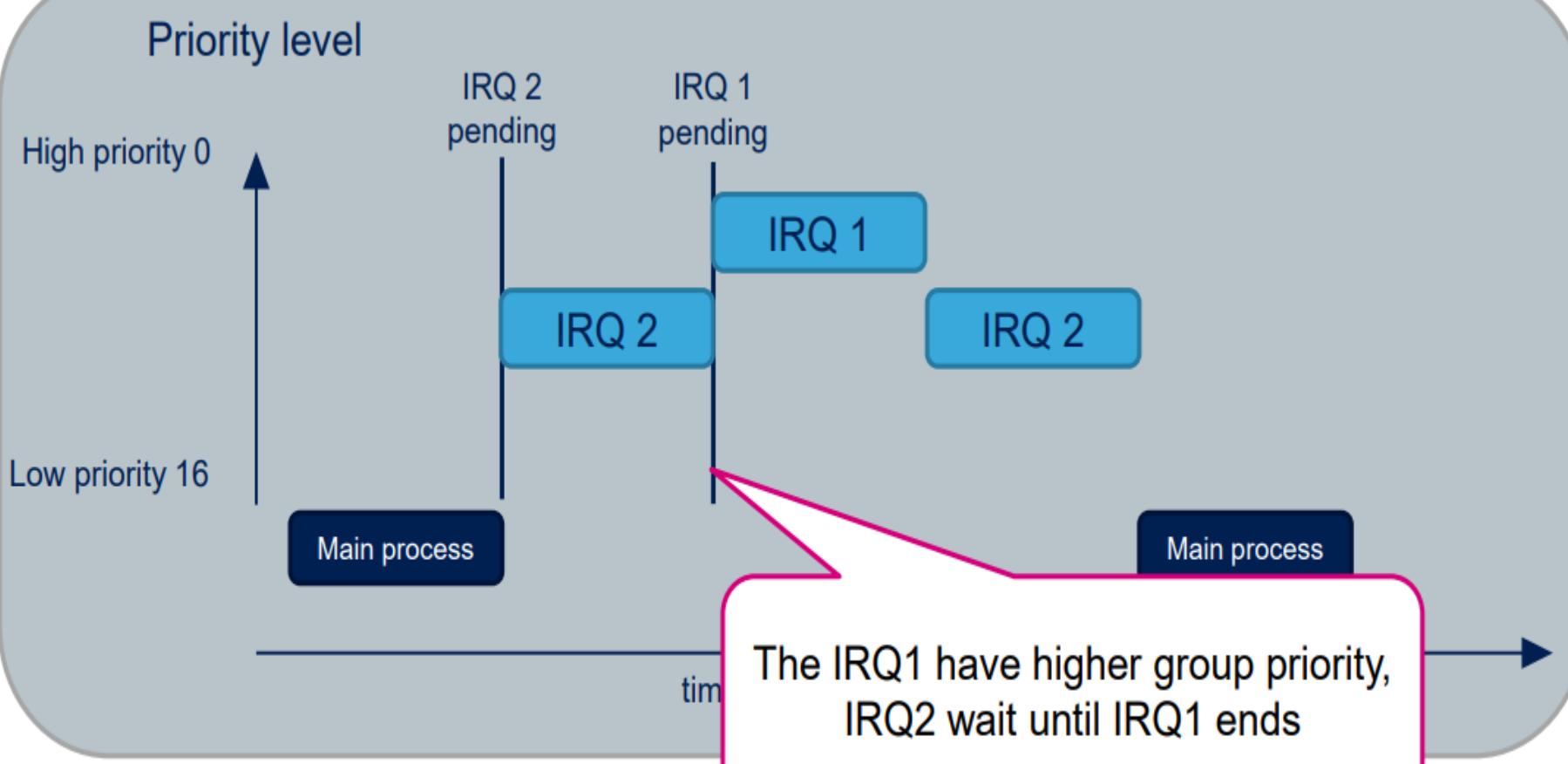


مصطلاح Preemption

أصبح لدينا مصطلحين :

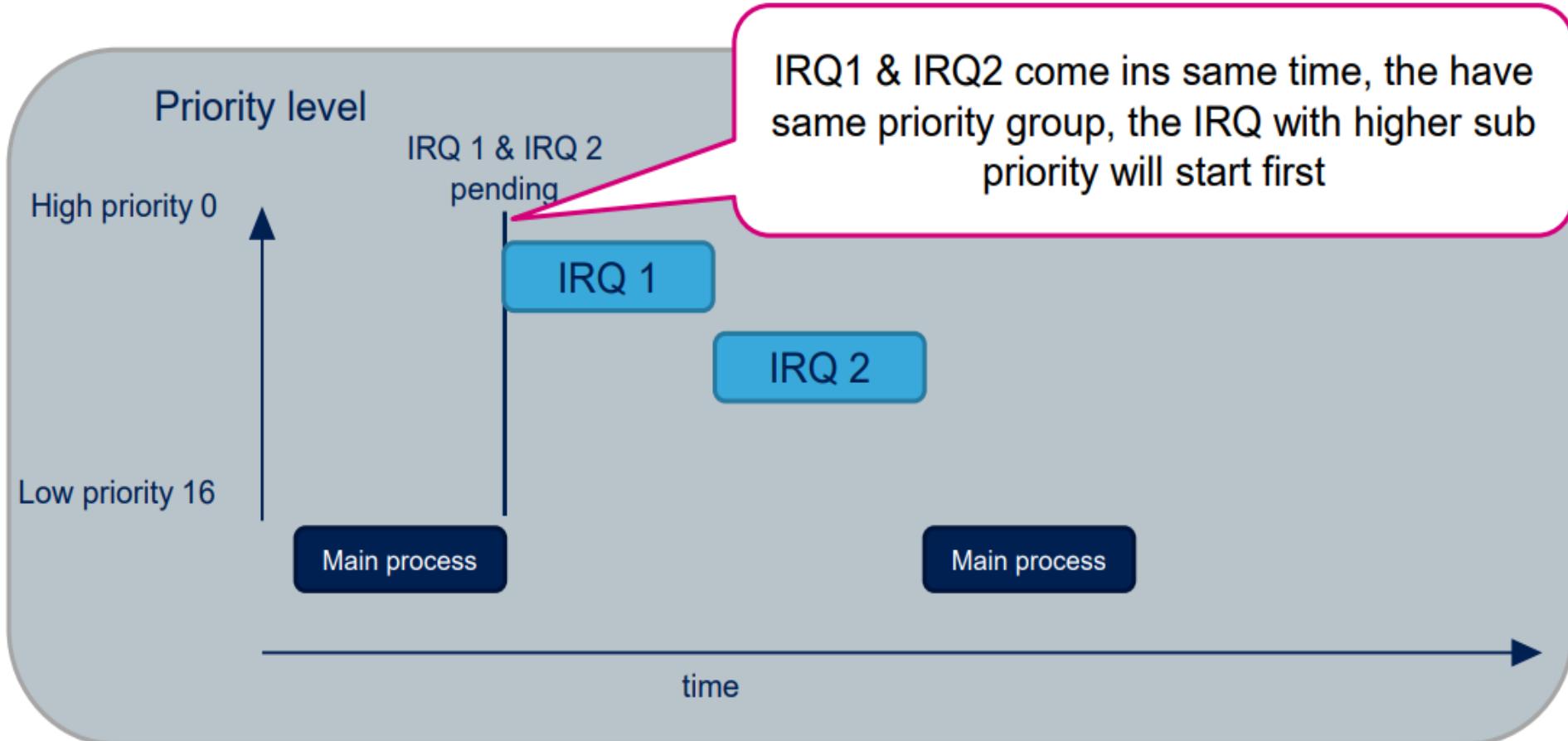
- **Preemption** : يعني مقاطعة استثناء يتم حالياً تنفيذه من قبل المعالج عند حدوث استثناء ذو مستوى أولوية أعلى
- **Pending**: يعني تعليق استثناء حدث ولكن لم يتم البدء بتنفيذه ، لحين تنفيذ الاستثناءات ذات الأولوية الأعلى

Preemption مصطلح



- IRQ1 - Group priority 0
- IRQ2 - Group priority 1

Preemption مصطلح



- IRQ1 - Group priority 0, Sub priority 0
- IRQ2 - Group priority 0, Sub priority 1

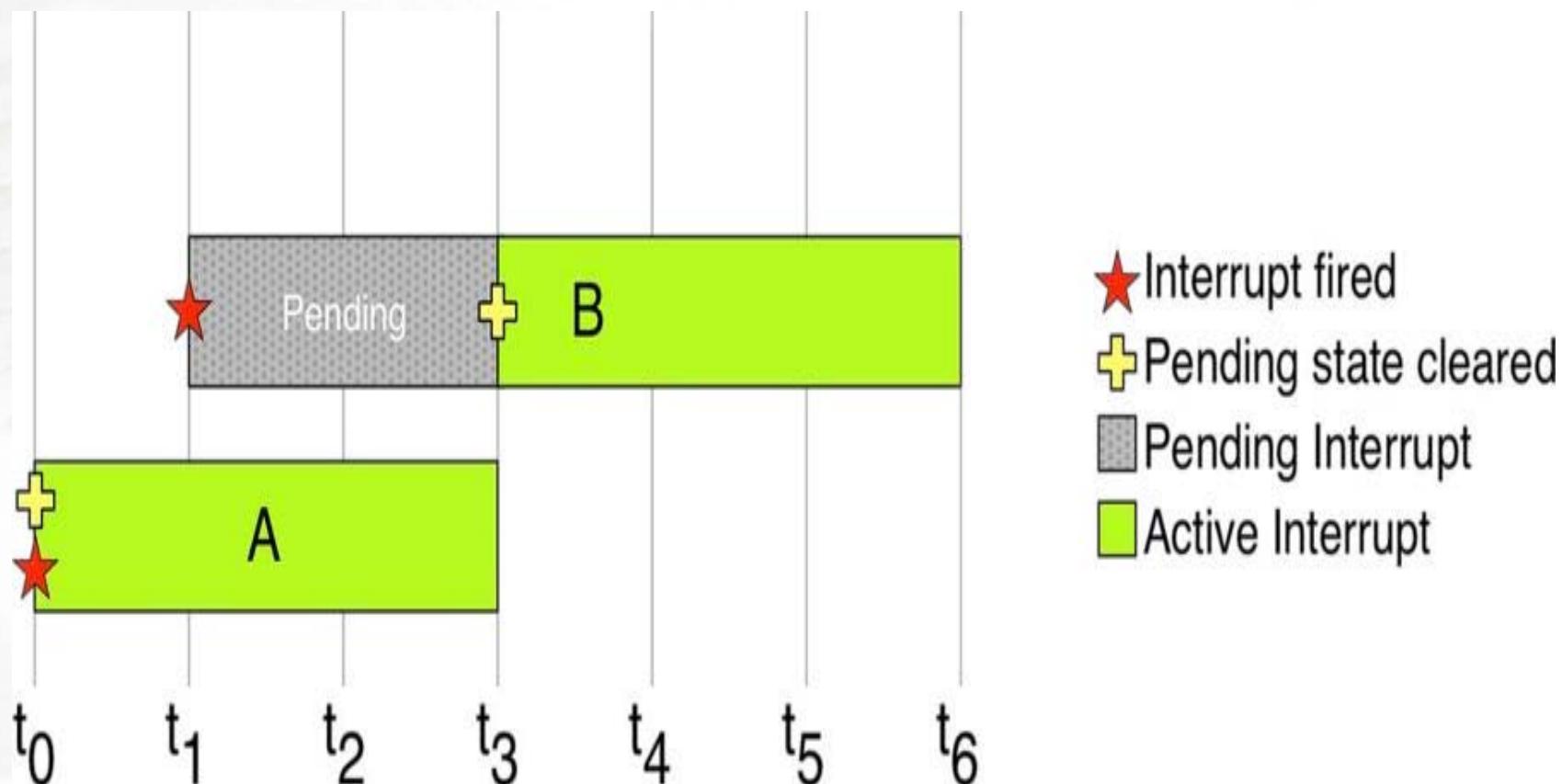
Interrupt Lifecycle دور حياة المقاطعة

يمكن للمقاطعة أن تكون:

- مفعلة enabled أو غير مفعولة disabled (وهو الوضع الافتراضي)
- معلقة (على قائمة الانتظار) Pending أو غير معلقة not pending
- نشطة active أي يتم تنفيذها حالياً أو غير نشطة not active.

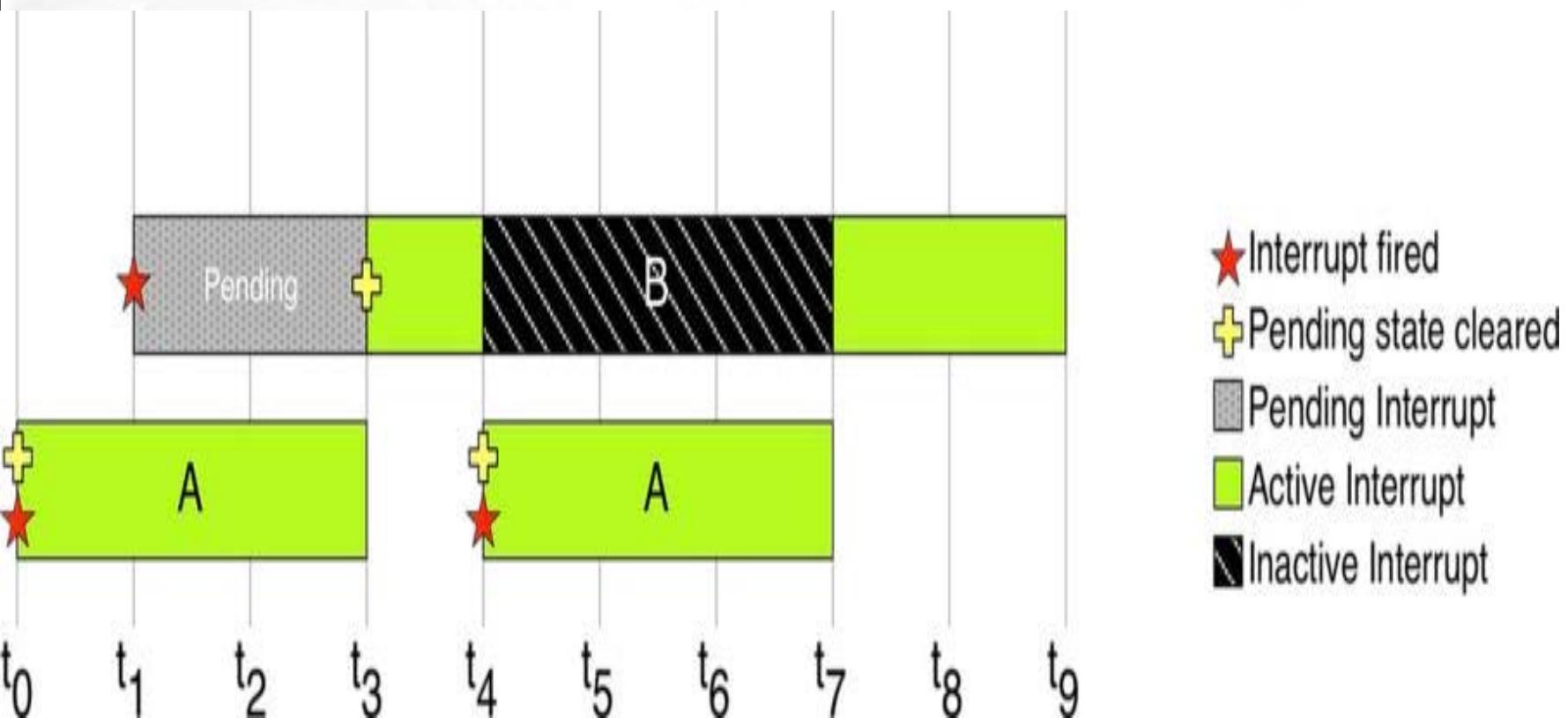
دوره حياة المقاطعة Interrupt Lifecycle

عند حدوث مقاطعة يتم تعليقها لحين يصبح المعالج قادر على خدمتها وفي حال عدم وجود مقاطعات أخرى يتم حالياً تنفيذها، يتم تصفير حالة التعليق pending state تلقائياً من قبل المعالج ومن ثم يبدأ بخدمتها



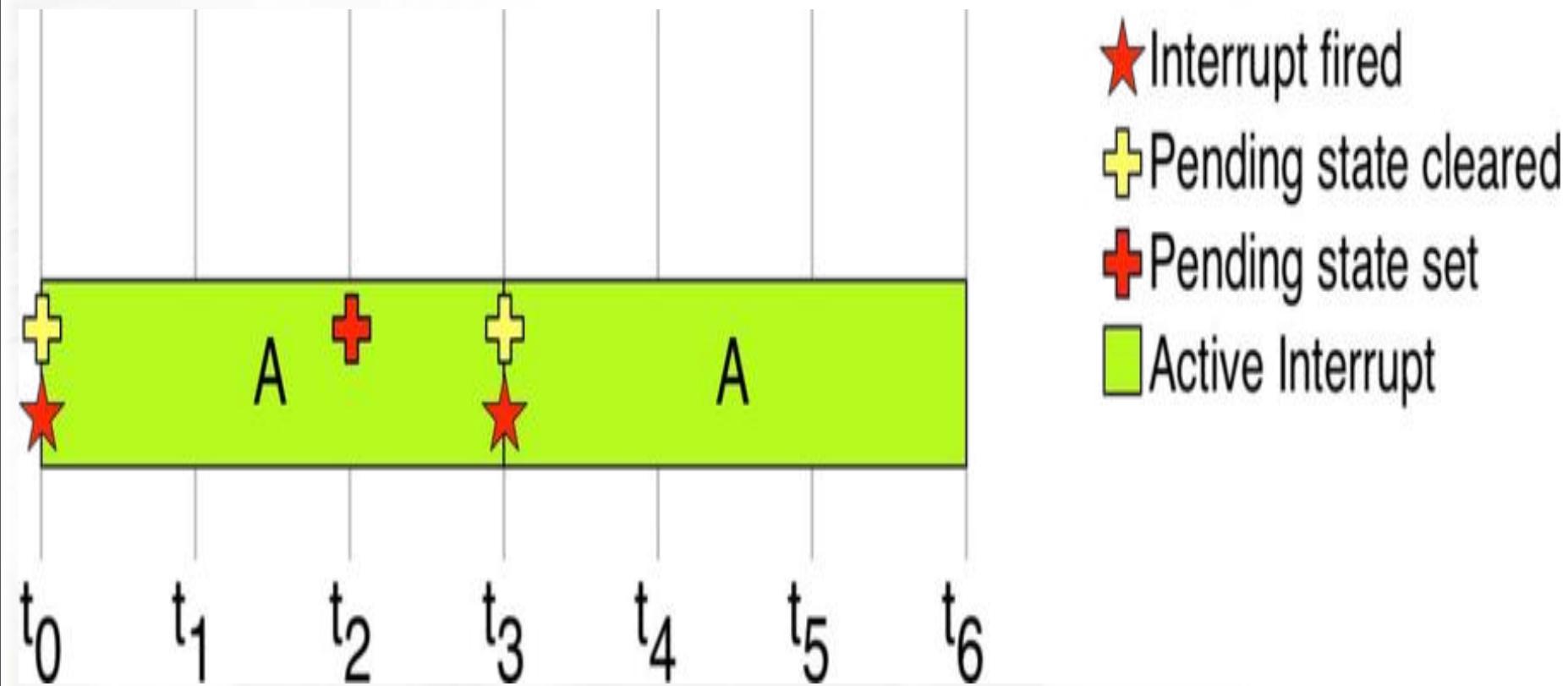
دوره حياة المقاطعة Interrupt Lifecycle

المقاطعات ذات الأولوية الأدنى عليها الانتظار لحين عدم وجود مقاطعات بأولوية أعلى منها ، وعند البدء بتنفيذها يتم وضعها في حالة inactive (preempted) عند حدوث مقاطعة لها أولوية أعلى



دوره حياة المقاطعة Interrupt Lifecycle

□ يمكن قدح المقاطعة مجدداً أثناء تنفيذها من خلال تفعيل بت الانتظار pending bit الخاص بها ، كما يمكن إلغاؤها من خلال مسح بت الانتظار الخاص بها كما هو بالأشكال التالية



Exception Behavior

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

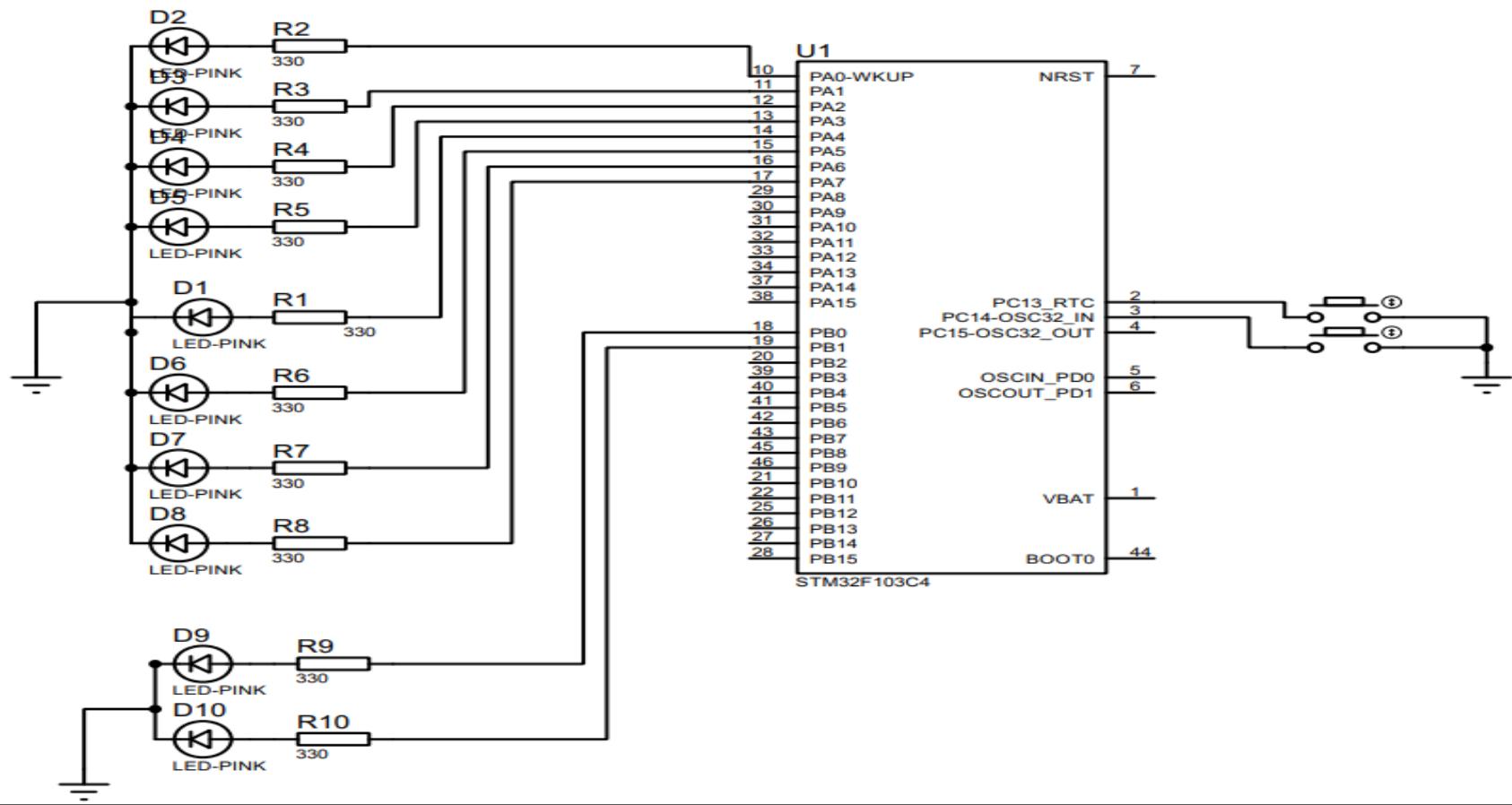
- يقوم المعالج بإنهاe التعليمية الحالية التي يقوم بتنفيذها طالما أنها لا تحتاج لأكثر من دورة تعليمية
- يتم حفظ حالة المعالج (عنوان التعليمية التي وصل لعندتها حالياً) إلى المدرس
- يتم تحميل عنوان برنامج خدمة المقاطعة الحاصلة من جدول أشعة المقاطعة
- يتم البدء بتنفيذ برنامج خدمة المقاطعة ISR، ويستغرق تنفيذ 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 التأكد 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 للطوارئ حيث يتم تنفيذه بغض النظر عن عمل النظام (إنارة اللدات) و بعد الانتهاء من برنامج خدمة المقاطعة (الطارئ) يعود للبرنامج الرئيسي ليتابع عمله



التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

نقوم بضبط الأقطاب PA0:PA6 والأقطاب PB0:PB1 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, the 'Pinout & Configuration' tab is active, displaying a table of GPIO pins. The table includes columns for Pin Name, Signal on Pin, GPIO output level, GPIO mode, GPIO Pull-up/Pull-down, Maximum output current, User Label, and Modified. Rows include PA0-WKUP, PA1, PA2, PA3, PA4, PA5, PA6, PB0, PB1, PC13-TAMPER-..., and PC14-OSC32_IN. The PC13 and PC14 rows have 'Modified' checkboxes checked. Below the table, there's a configuration panel for PC13-TAMPER-RTC.

The right side of the interface shows the 'Pinout view' for an STM32F103C4Tx LQFP48 package. It displays the physical pin layout with labels for VBAT, PC13.., PC14.., PC15.., PD0.., PD1.., NRST, VSSA, VDDA, PA0.., PA1, PA2, PA3, PA4, PA5, PA6, PA7, PB0, PB1, PB2, PB10, PB11, VSS, and VDD. The pins corresponding to the configured GPIOs (PC13, PC14, PA0, PA1, PA2) are highlighted in green.

التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجة

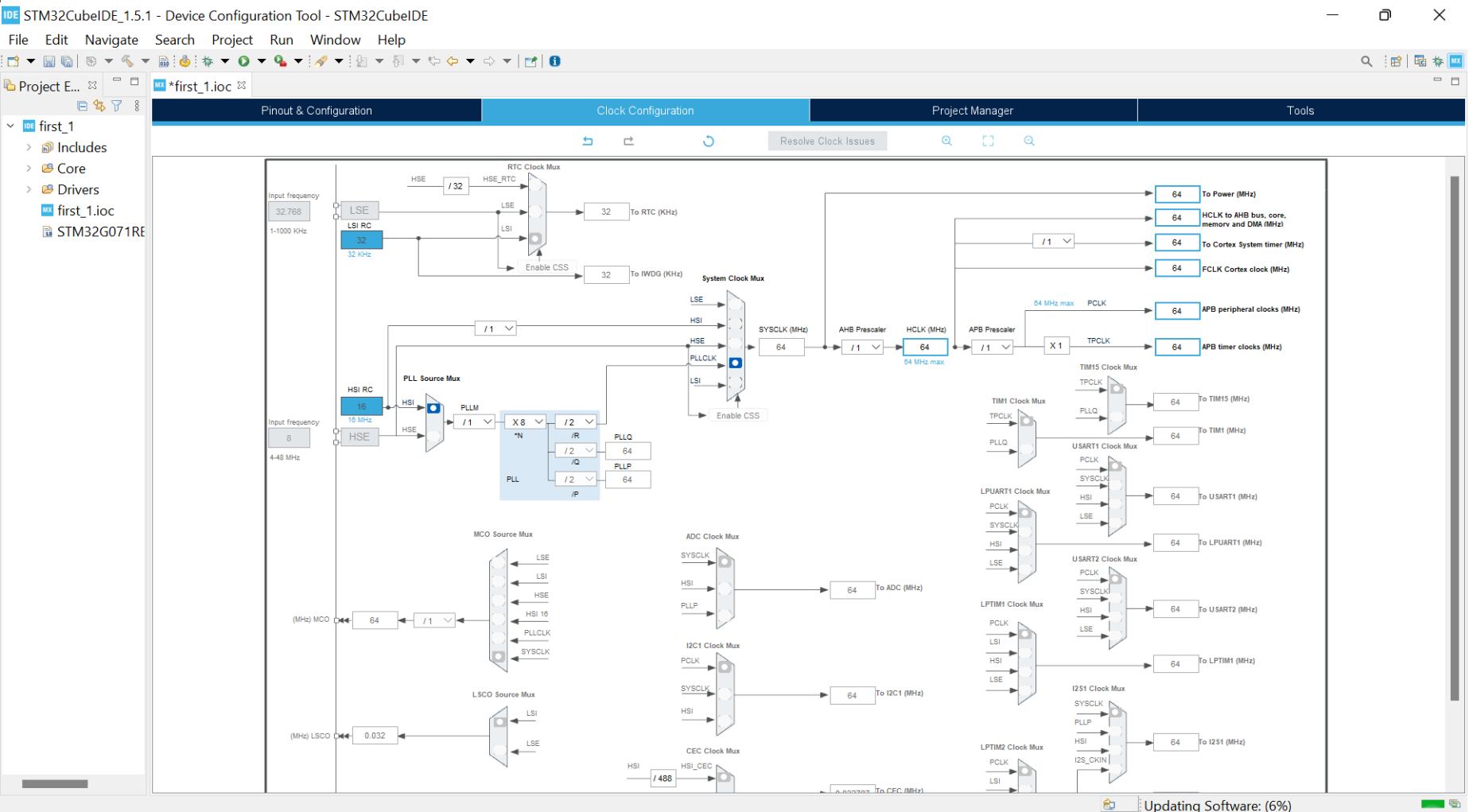
قم بفتح NVIC Tab ثم قم بتفعيل المقاولات الخارجية، يمكنك أيضاً إعادة ضبط مستويات الأولوية للمقاولات:

The screenshot shows the MX Studio interface for interrupt configuration. On the left, a sidebar lists categories like DMA, GPIO, IWDG, NVIC, RCC, SYS, and WWDG. The NVIC category is selected and highlighted in blue. The main window displays the 'NVIC & Configuration' tab, specifically the 'Pinout & Configuration' section under 'Clock Configuration'. A sub-menu 'Software Packs' is open, showing 'NVIC Mode and Configuration'. The 'Pinout' tab is also visible. On the right, there's a 'System view' panel which is currently inactive (disabled). The central area shows the 'NVIC Interrupt Table' with various interrupt types listed. One row, 'EXTI line 4 to 15 interrupts', is highlighted with a red border and has a red number '3' next to it. At the bottom of the table, there's a note about USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26. The bottom right corner of the 'System view' panel also has a red number '2'.

Interrupt Type	Enabled	Preemption Priority
Non maskable interrupt	<input checked="" type="checkbox"/>	0
Hard fault interrupt	<input checked="" type="checkbox"/>	0
System service call via SWI instruction	<input checked="" type="checkbox"/>	0
Pendable request for system service	<input checked="" type="checkbox"/>	0
Time base, System tick timer	<input checked="" type="checkbox"/>	0
PVD interrupt through EXTI line 16	<input type="checkbox"/>	0
Flash global interrupt	<input type="checkbox"/>	0
RCC global interrupt	<input type="checkbox"/>	0
EXTI line 4 to 15 interrupts	<input checked="" type="checkbox"/>	0
USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26	<input type="checkbox"/>	0

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