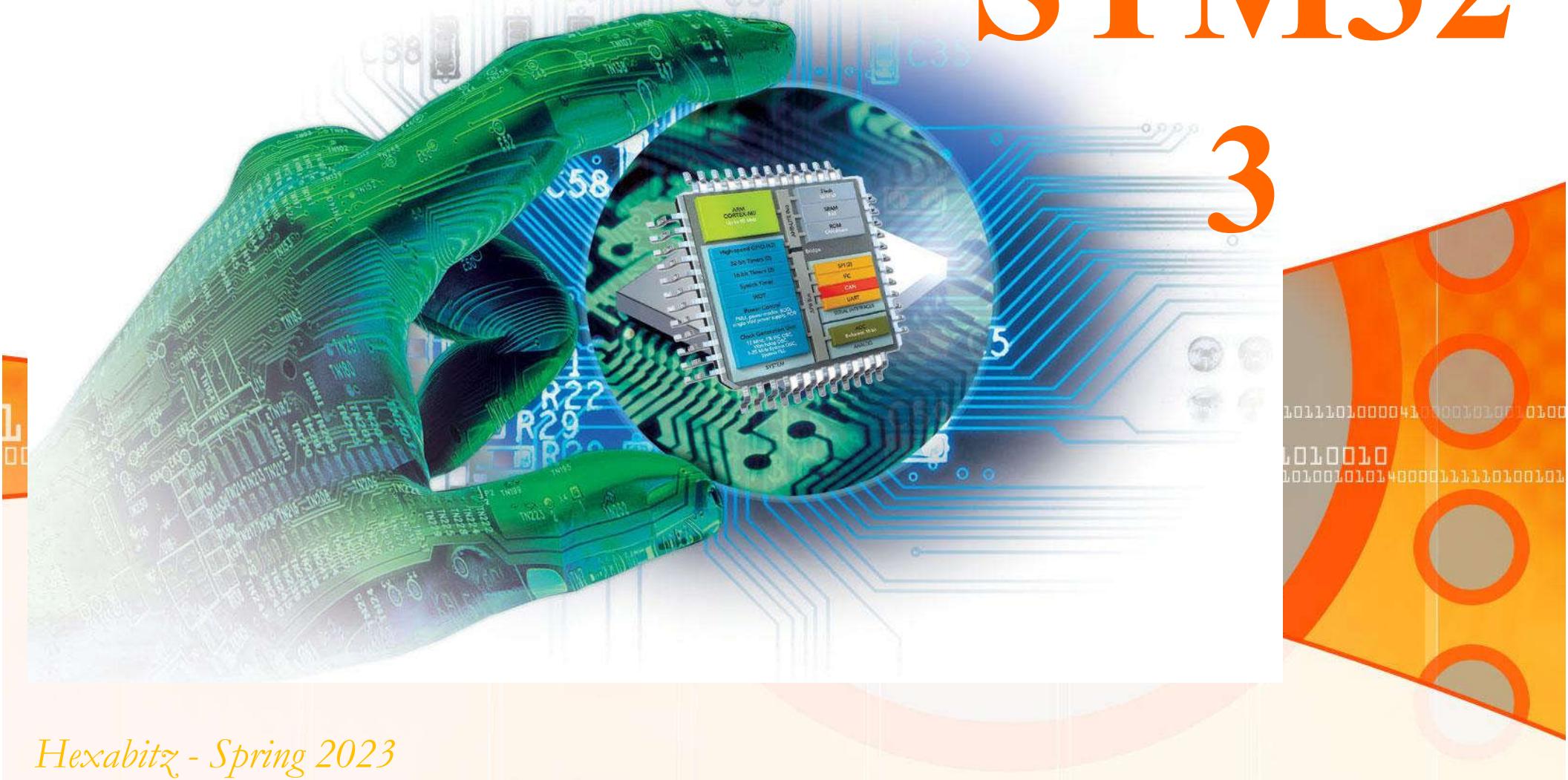


متحكمات

STM32



موضوعات المحاضرة:

- Exceptions Overview
- Micro-Coded Interrupts
- Exception Types
- System Exceptions
- متحكم 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 الحالي اعتماداً 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 وتحفيض 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 لاـ RESET
- فقط 9 استثناءات من نوع system exception يتم التعامل معها والـ 6 الباقيه هي للاستخدامات المستقبالية.
- 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 قدرح 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

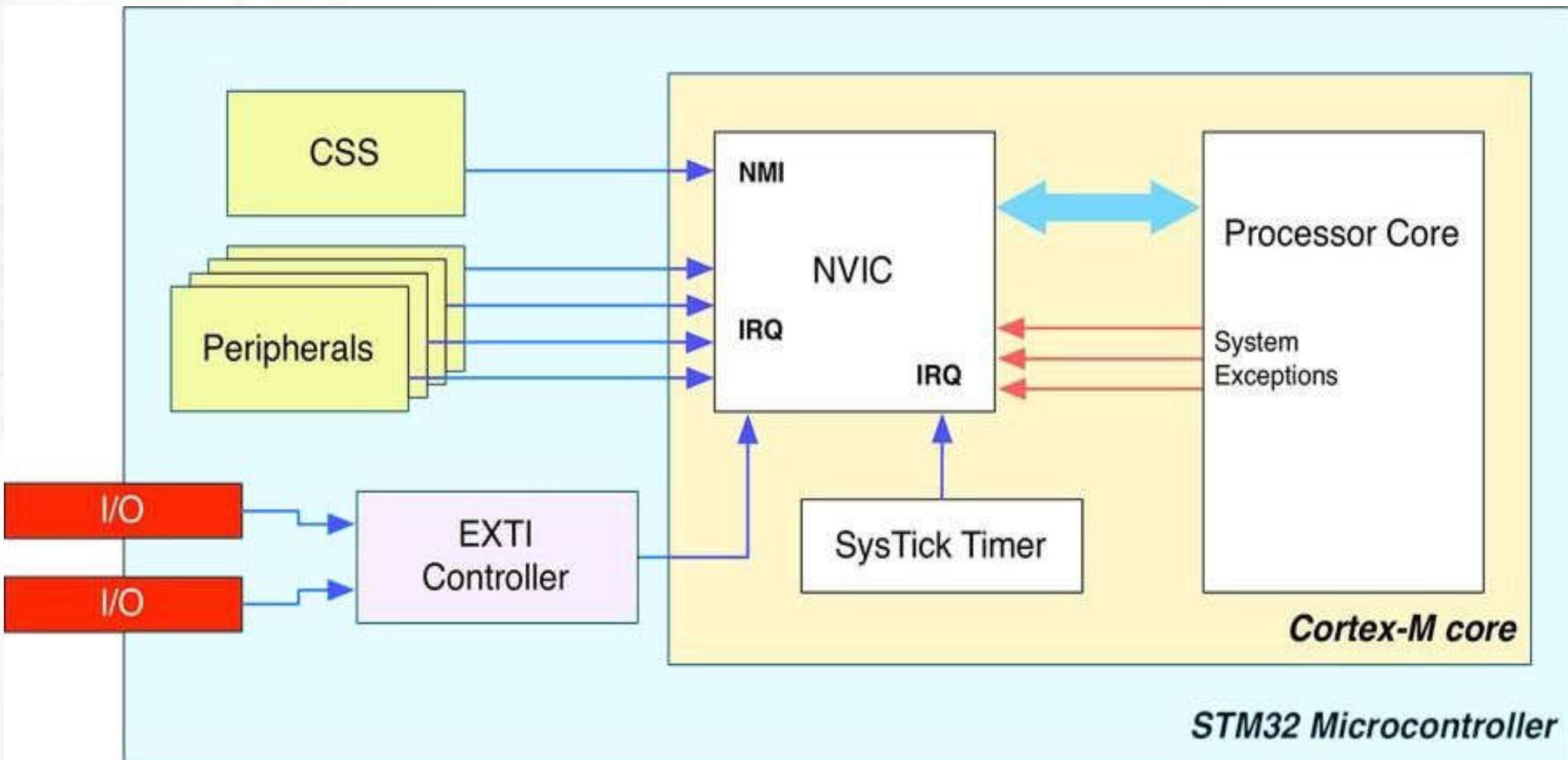
- **System tick** هذا الاستثناء يستخدم دائماً في أنظمة الزمن الحقيقي RTOS، حيث يحتاج كل نظام RTOS إلى مؤقت لجدولة أنشطة النظام بحيث يقوم بـتوليد مقاطعة في كل فترة زمنية لترك تنفيذ المهمة الحالية وتنفيذ مهمة أخرى ويكون هذا المؤقت متصل داخلياً بنواة CORTEX-M وليس وحدة خارجية كباقي الطرفيات.
- **Interrupt (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 (IRQ) هي عبارة عن الاستثناءات التي يتم توليدها من قبل 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

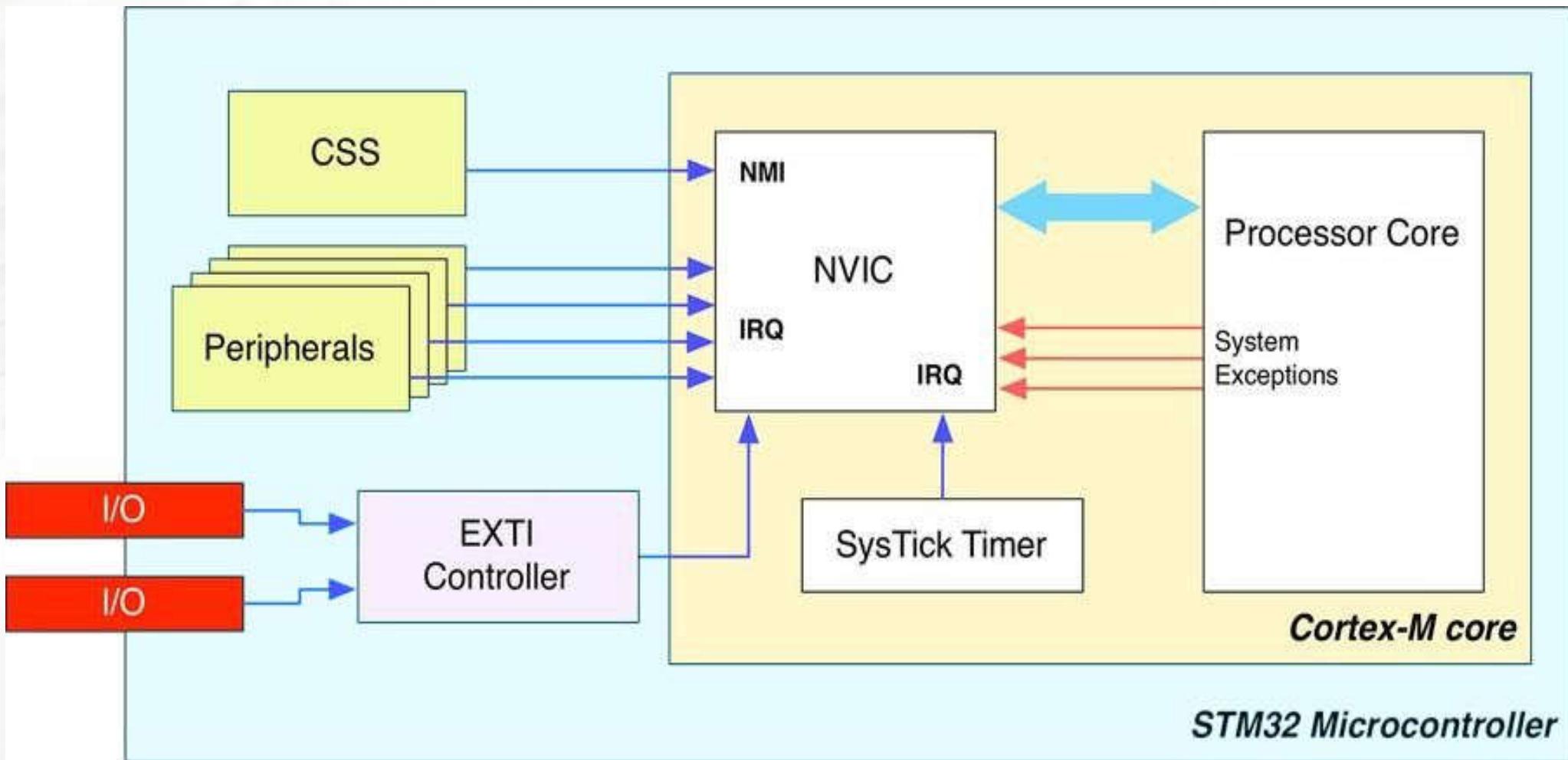
متحكم NVIC (Nested Vectored Interrupt Controller)

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



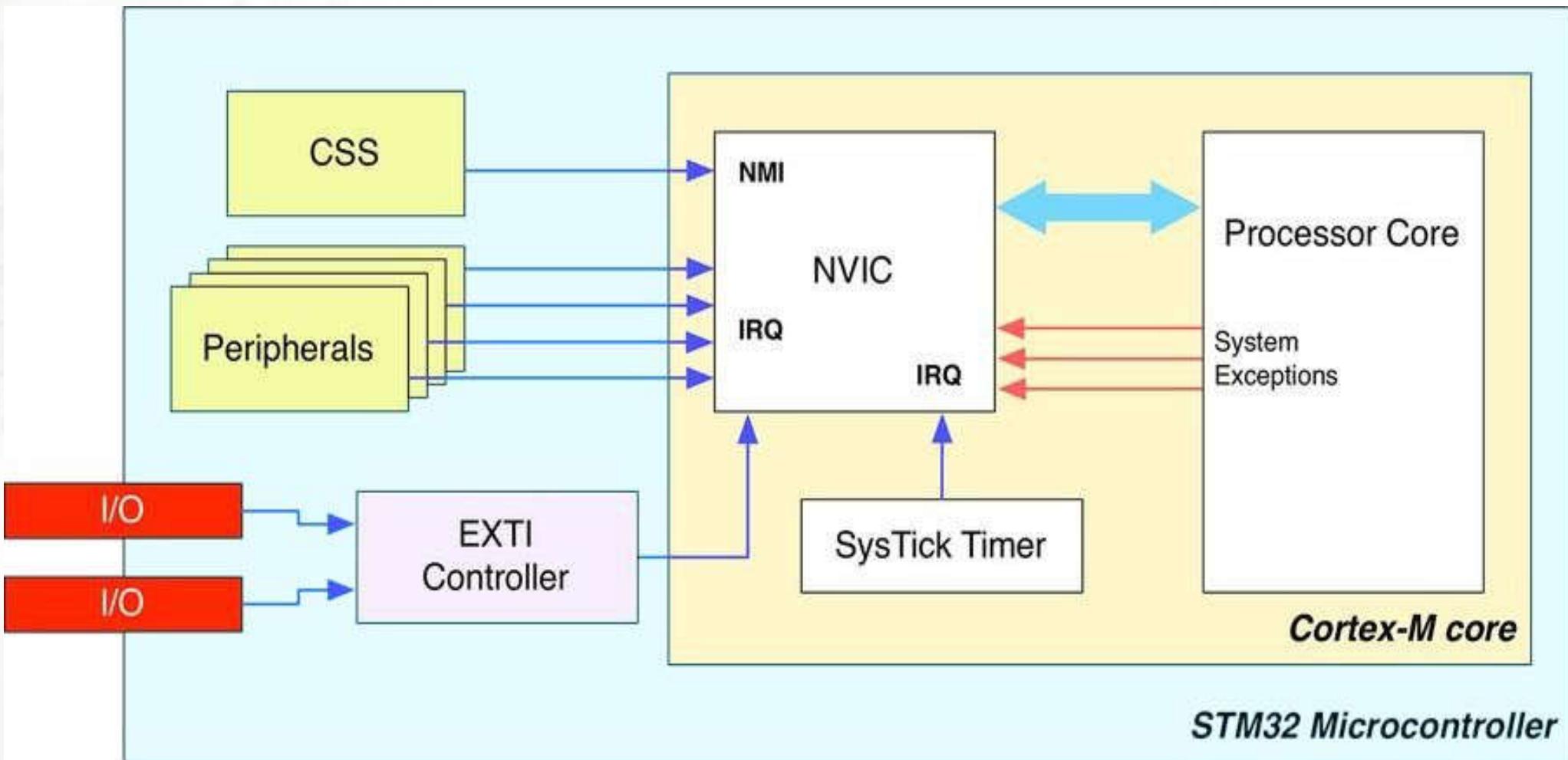
متحكم NVIC (Nested Vectored Interrupt Controller)

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 ل المقاطعات 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 PB0 و PC0 متصلين بالخط LINE0 ، بمعنى أن جميع الأطراف ذات الأرقام PX0 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 واحد أن يقوم بتوليد المقاطعة وعلى البرنامج أن يكتشف أي pin قام بتوليد المقاطعة

المقاطعات الخارجية External Interrupts

يجب الانتباه للملاحظات التالية:

الأطراف PA0، PB0، PC0... جميعها متصلة بالخط LINE0 لذا

في اللحظة الواحدة

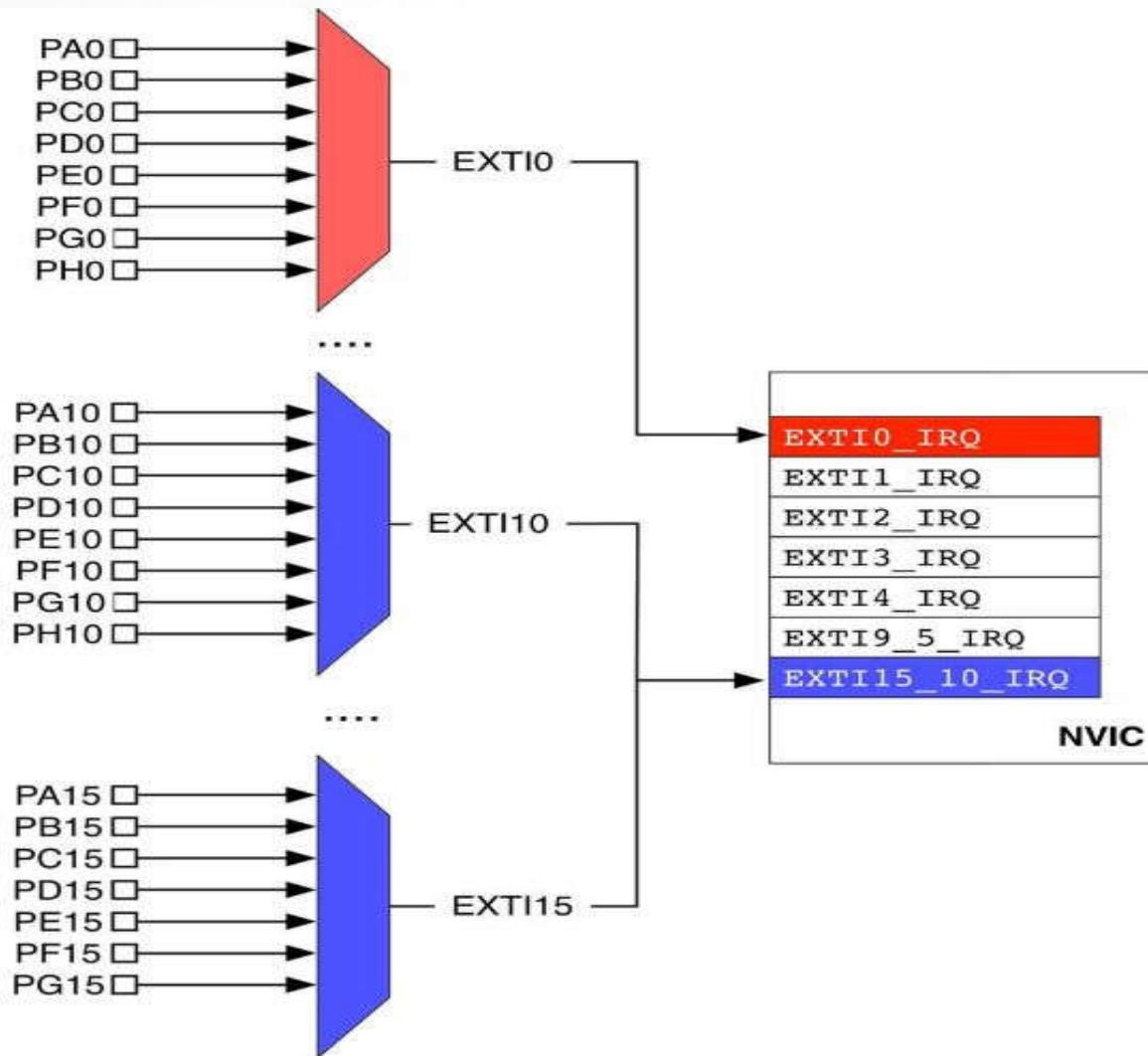
بإمكانك توليد مقاطعة على طرف واحد فقط من هذه الأطراف.

الأطراف PA0، PA5

متصلين على خطين مختلفين

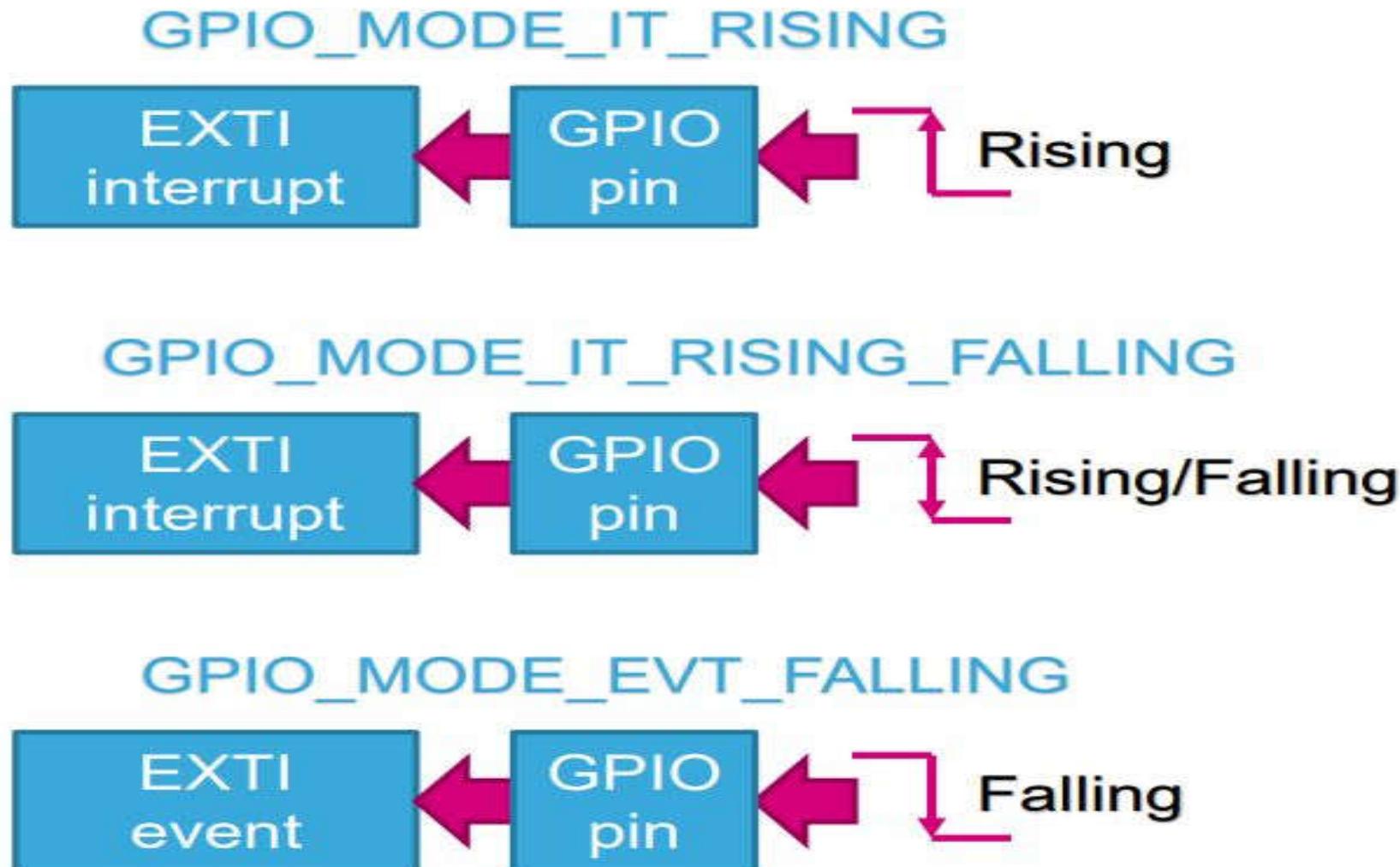
من خطوط المقاطعة لذا

يمكنك استخدامهم في نفس اللحظة لتوليد المقاطعة.



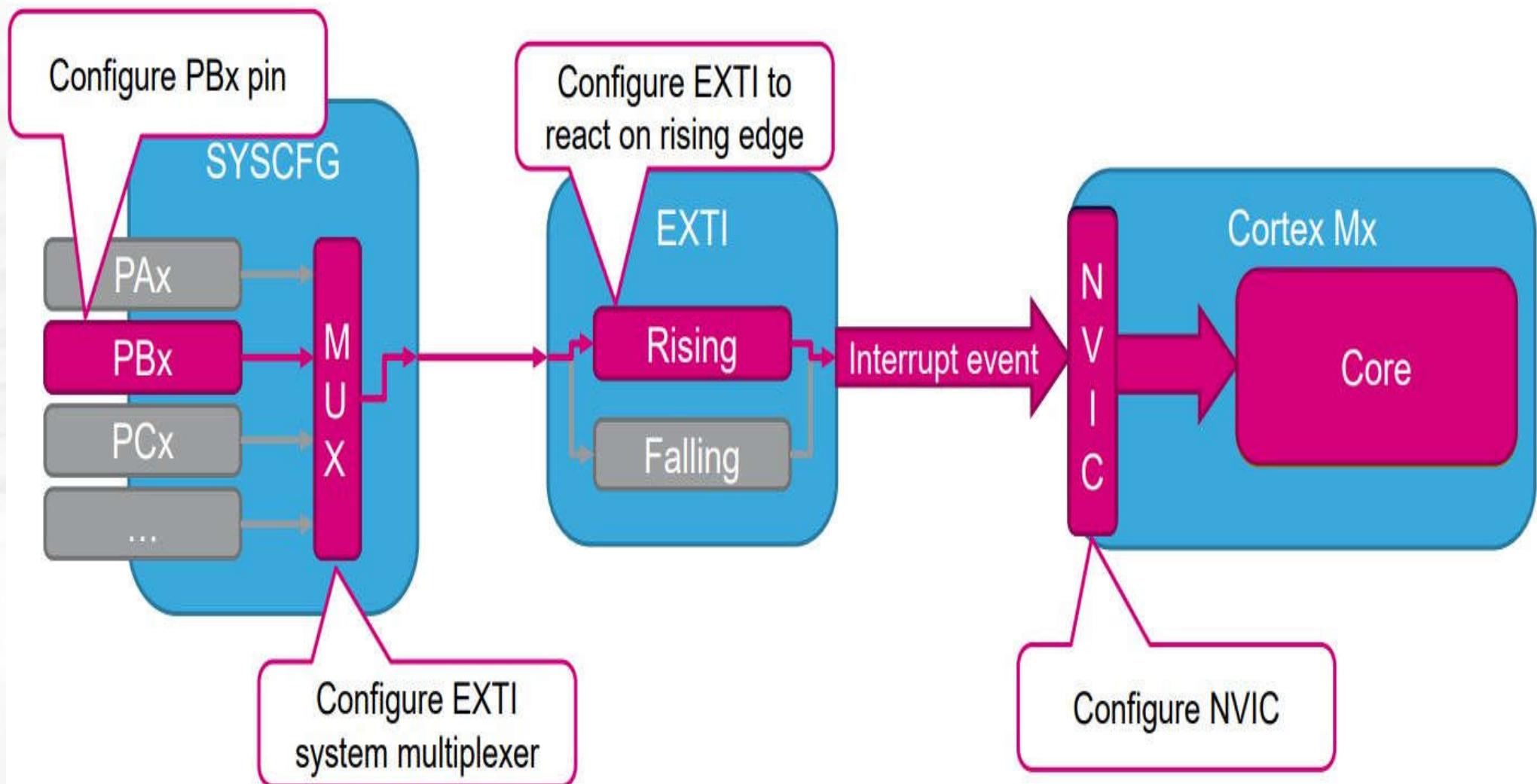
المقاطعات الخارجية External Interrupts

يمكن قذح المقاطعة عند الجبهة الصاعدة rising edge أو الهابطة falling edge □



المقاطعات الخارجية External Interrupts

فتحدث المقاطعة الخارجية وفقاً للآلية التالية:



أنماط عمل المعالج Processor mode

للمعالج نمطي عمل أساسين:

- Thread Mode : يكون المعالج في نمط Thread Mode الوضع الطبيعي له عند تنفيذ الكود أو عند عمل له 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 الـ hardware ليضمن استجابة سريعة لحدث المقاومة

Exception Behavior

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

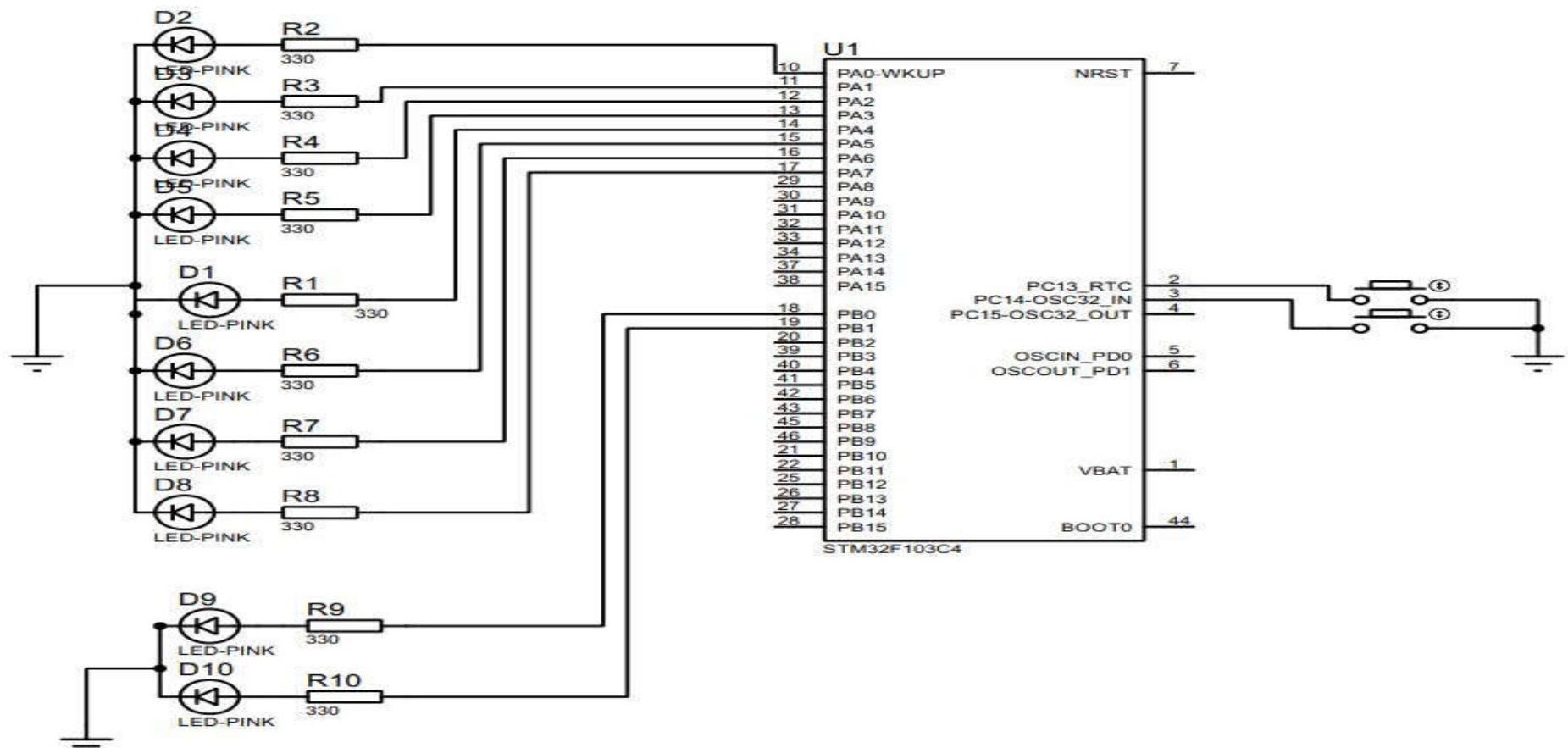
- يقوم المعالج بإنهاء التعليمية الحالية التي يقوم بتنفيذها طالما أنها لا تحتاج لأكثر من دورة تعليمية
- يتم حفظ حالة المعالج (عنوان التعليمية التي وصل لعندها حالياً) إلى المكدس
- يتم تحويل عنوان برنامج خدمة المقاطعة الحاصلة من جدول أشعة المقاطعة
- يتم البدء بتنفيذ برنامج خدمة المقاطعة ISR ، ويستغرق تنفيذ كامل هذه العملية 12 cycles في معالجات
- يقوم برنامج خدمة المقاطعة بتصفير 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 تعليمية 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 الحرجية

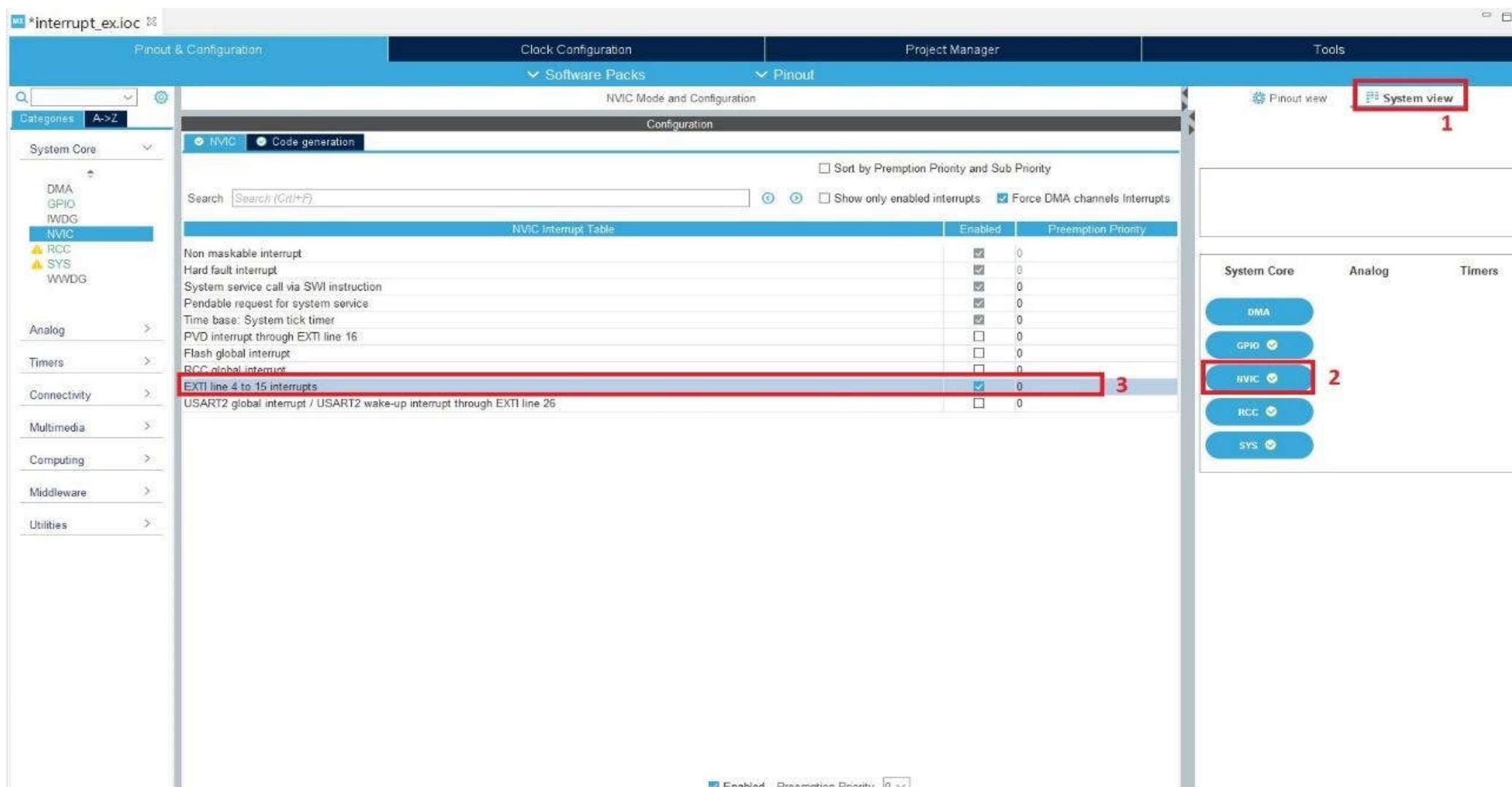
نقوم بضبط الأقطاب PA0:PA6 ، والأقطاب PB0:PB1 كأقطاب خرج ، والقطب PC13 كقطب دخل مع تفعيل مقاومة الرفع الداخلية، والقطب PC14 كقطب مقاطعة خارجية



The screenshot shows the STM32CubeMX software interface. On the left, the 'Pinout & Configuration' tab is active, displaying the 'GPIO Mode and Configuration' table. The table lists various pins (PA0-WKUP to PA6, PB0-PB1, PC13-TAMPER-..., PC14-OSC32_IN) with their current configurations: Signal on Pin, GPIO output level, GPIO mode, GPIO Pull-up/Pull-down, Maximum output, User Label, and Modified status. The row for PC13-TAMPER-... has 'Input mode' and 'Pull-up' selected under its configuration. On the right, the 'Pinout' tab is active, showing the physical pin layout for the STM32F103C4Tx LQFP48 package. The pins are grouped by function: GPIO_Input (VBAT, PC13.., PC14.., PC15.., PD0.., PD1.., NRST, VSSA, VDDA), GPIO_Output (PA0.., PA1, PA2, PA3, PA4, PA5, PA6, PA7, PB0, PB1, PB2, PB10, PB11, VSS, VDD). The PC13 pin is highlighted in green as an input pin.

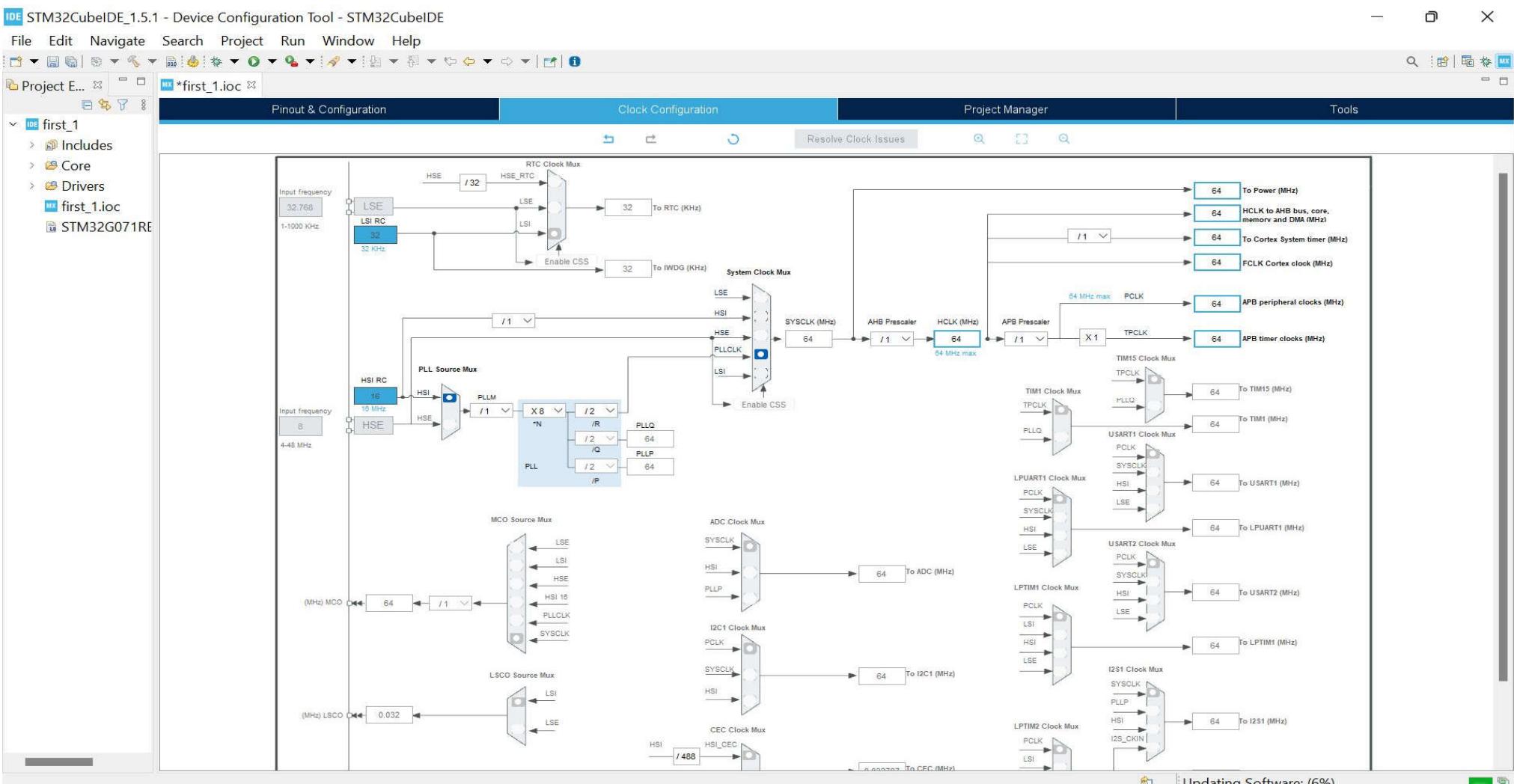
التطبيق الأول: برمجة زر للطوارئ لاستخدامه في الحالات الحرجية

قم بفتح Tab NVIC 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 code** 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
```

```
{
```

```
{
```

```
{
```

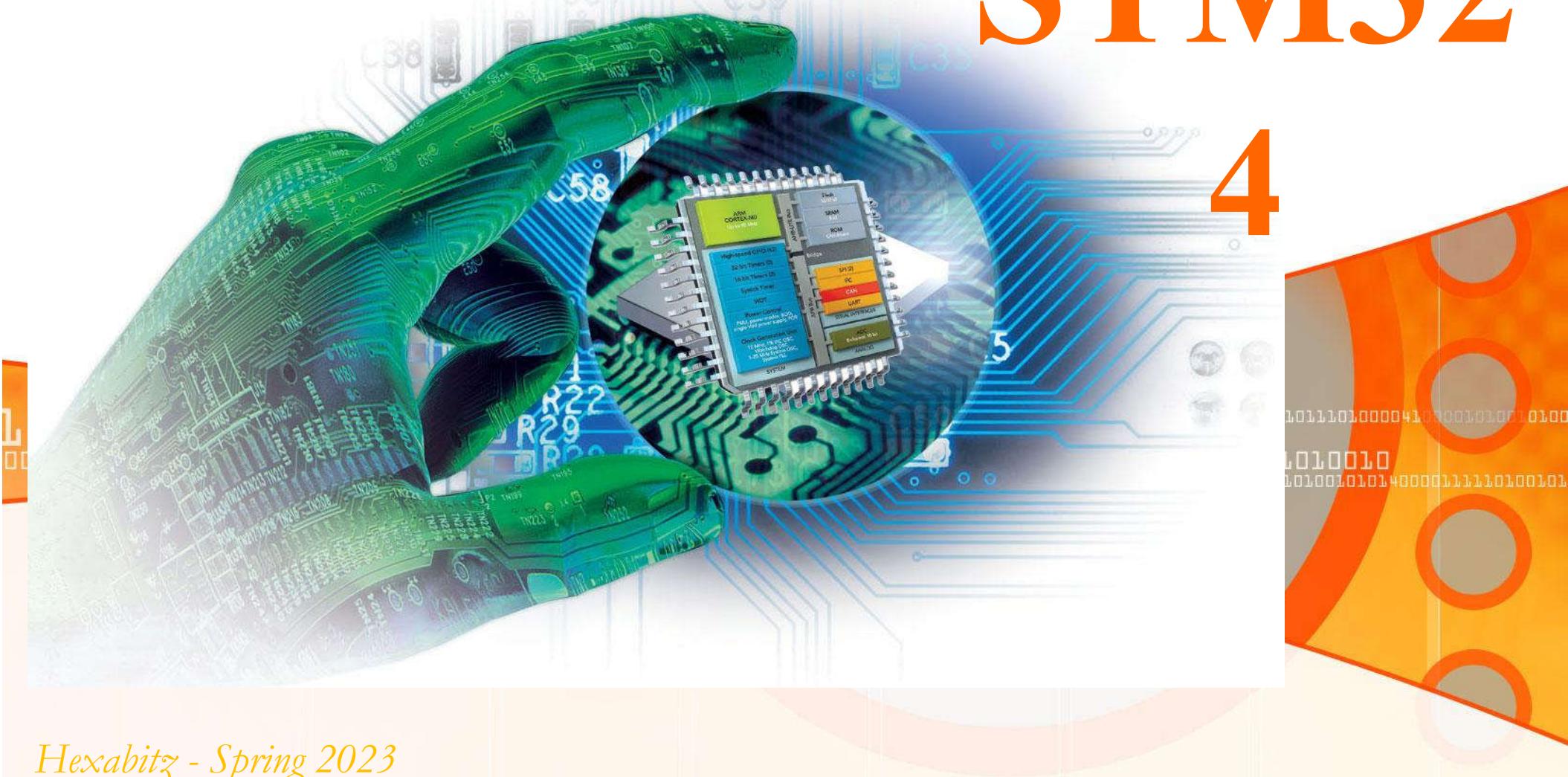
Thank you for listening

متحكمات

STM32

4

5



مُوْضوِعَاتِ الْمَحَاضِرَةِ:

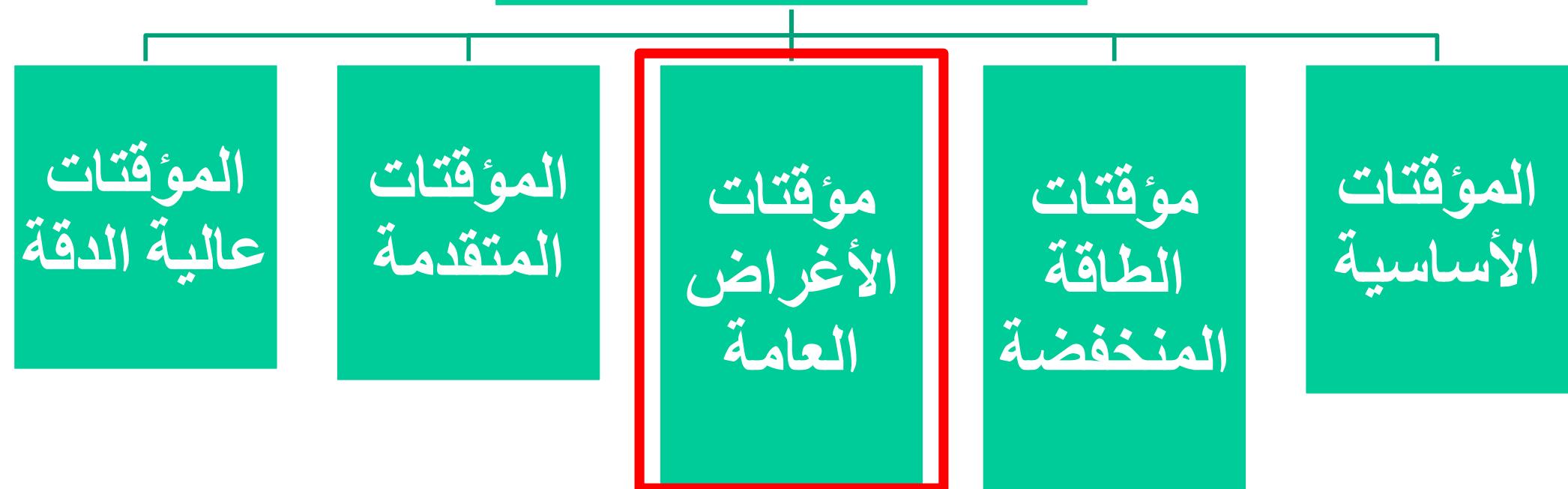
- الأنواع المختلفة للمؤقتات
- مؤقتات الأغراض العامة
- أنماط العمل المختلفة للمؤقتات في متحكمات STM32
- ساعة الزمن الحقيقي (RTC)
- تطبيقات عملية

المؤقتات Timers

- يوجد في متحكمات STM32 العديد من المؤقتات المختلفة كل منها بإمكانها العمل بأنماط مختلفة
- المؤقت عبارة عن عداد يبدأ بالعد من الصفر ويزداد بمقدار عدة واحدة مع كل نبضة ساعة للمتحكم
- بإمكانه العد التصاعدي و التنازلي على حد سواء
- تسمح خاصية ال prescaler أو المقسم الترددية بتقسيم تردد الساعة على عدد معين يتم اختياره بين ال 0 و 65535

الأنواع المختلفة للمؤقتات

الأنواع المختلفة للمؤقتات



مُؤقتات الأغراض العامة :General Purpose Timers

- المُؤقتات في هذه المجموعة تكون إما 16 أو 32 بت(بناءً على عائلة STM32)،
- يمتلك عداد تصاعدي/ تنازلي بطول 16 بت قابل لإعادة التحميل تلقائياً auto-reload counter
- مقسم جهد بطول 16 بت يستخدم لتقسيم تردد الساعة للمتحكم على أي عدد يتراوح بين 1 و 65535

أنماط العمل المختلفة للمؤقتات في متحكمات STM32

Input
Capture

Output
Compare

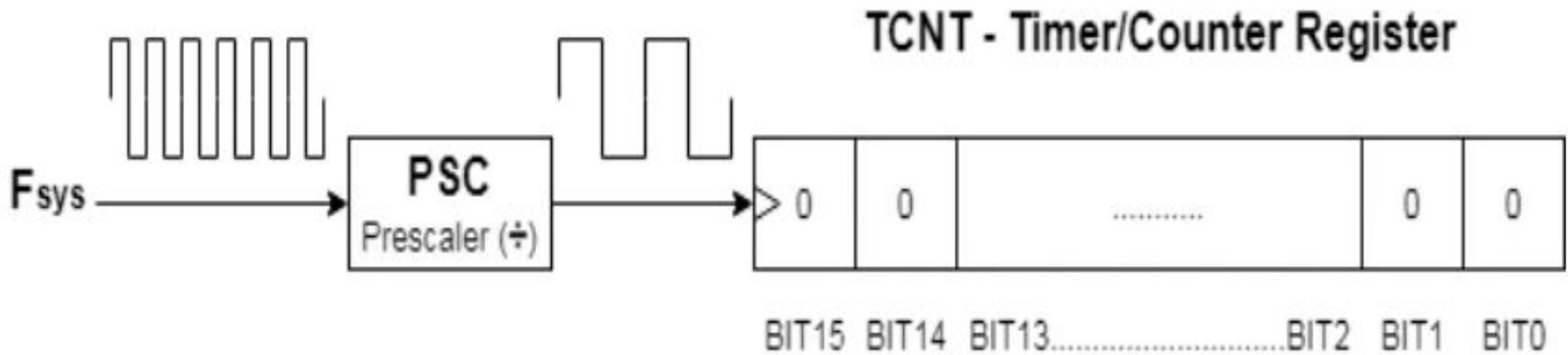
نط
PWM

نط
العداد

نط المؤقت

نـمـطـ الـمـؤـقـتـ Timer

- عند عمل المؤقت بنـمـطـ Timer ، فإن المسـجـلـ TCNT تـزـدـادـ قـيـمـتـهـ بـمـقـدـارـ وـاحـدـ مـعـ كـلـ نـبـضـةـ سـاعـةـ لـلـمـؤـقـتـ
- يـكـونـ مـصـدـرـ السـاعـةـ لـلـمـؤـقـتـ فـيـ هـذـهـ الـحـالـةـ دـاـخـلـيـ قـادـمـ مـنـ سـاعـةـ المـتـحـكـمـ



- حيـثـ يـقـومـ بـالـعـدـ مـنـ الصـفـرـ إـلـىـ الـقـيـمـةـ المـحـدـدـةـ فـيـ حـقـلـ الـAـ Period(preload)ـ أـثـنـاءـ تـهـيـئـةـ الـمـؤـقـتـ ،ـ وـأـكـبـرـ قـيـمـةـ يـمـكـنـ أـنـ يـصـلـ إـلـيـهـ تـحدـدـ حـسـبـ طـوـلـ الـمـؤـقـتـ،ـ حـيـثـ الـمـؤـقـتـ 16ـ بـتـ يـمـكـنـهـ

الـعـدـ إـلـىـ 0xfffffـ وـالـمـؤـقـتـ ذـوـ 32ـ بـتـ يـمـكـنـهـ العـدـ إـلـىـ 15ـ
Hexabitz - Spring 2023 4

نقط المؤقت Timer

يعتمد تردد (سرعة العد) على المقسم الترددی Prescaler حيث يتم تقسيم تردد ساعة المؤقت على واحدة من القيم المتاحة وهي من 1 حتى 65535 (حيث أن مسجل الـ Prescaler بطول 16 بت)

عندما يصل العداد إلى القيمة المحددة Period(preload) يحدث ما يسمى بالطفحان overflow أي يقوم بالتصغير والعد مرة أخرى من الصفر و يتم رفع العلم الخاص بالطفحان Update Event(UEV)

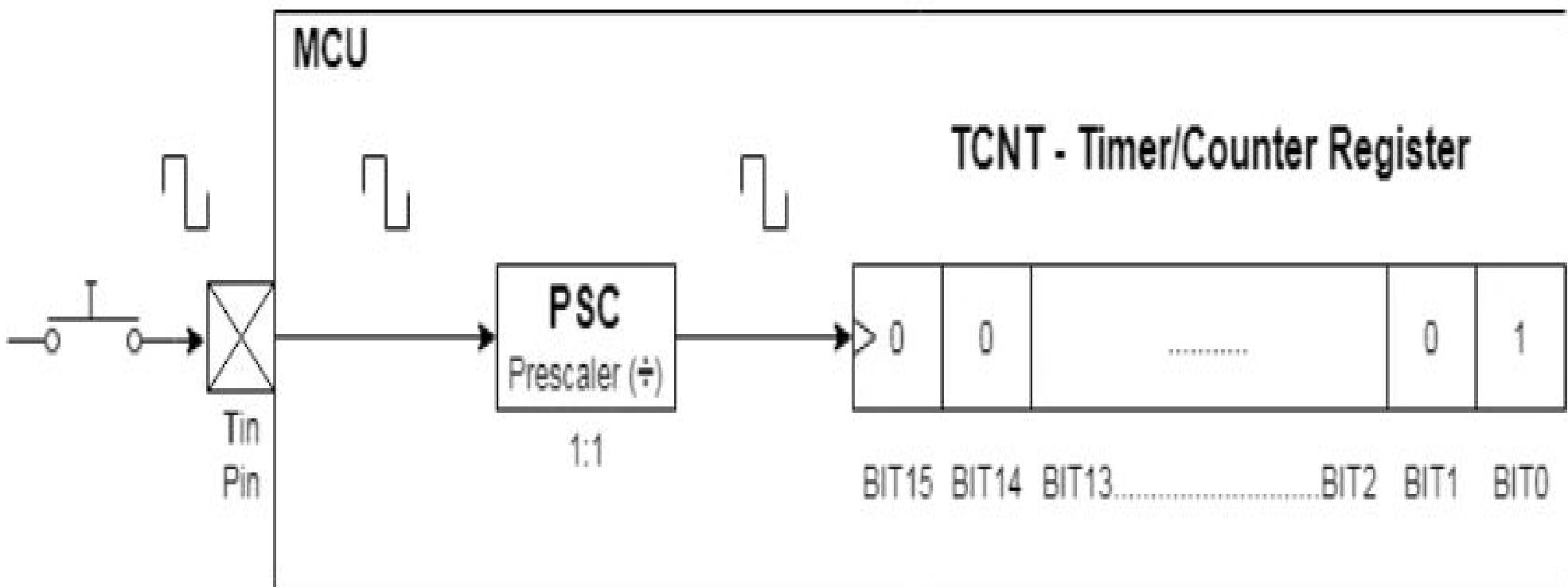
نقط المؤقت Timer

□ على سبيل المثال ، لنفترض أن تردد الساعة للمتحكم مضبوط على **preload** وقيمة الـ Prescaler تساوي 48 MHz والـ تساوي 500 سيفوت كل: overflow المؤقت كل:

$$T_{out} = \frac{48000 \times 500}{48000000} = 0.5sec$$

نطع العداد Counter

يمكن للمؤقت أن يعمل بنطع Counter وفي هذه الحالة سيكون مصدر الساعة للمؤقت عبارة عن إشارة خارجية ممكن أن تكون قادمة من مفتاح لحظي، عندها ستزداد قيمة المؤقت مع كل جبهة صاعدة/هابطة عند ضغط المفتاح اللحظي وبالتالي سيعد المؤقت عدد المرات التي تم فيها ضغط المفتاح اللحظي



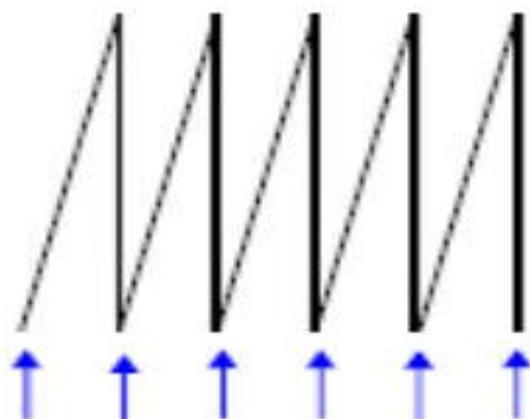
نط العداد Counter

ويمكنه أن يعمل بثلاث أنماط مختلفة هي:

□ نمط العد التصاعدي Up-counting Mode

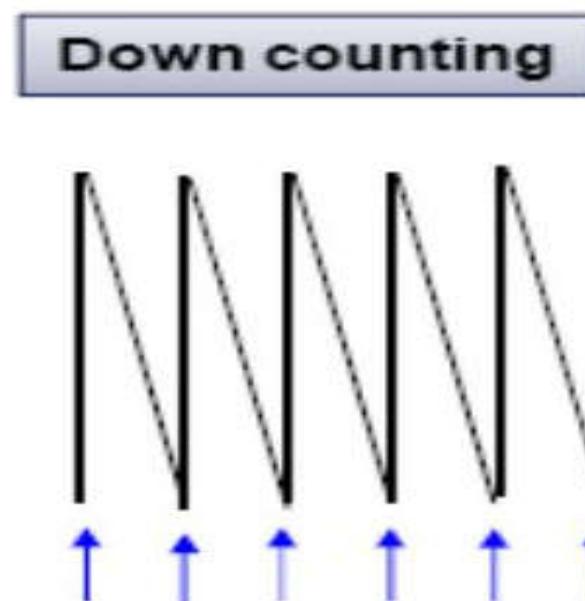
في هذا النمط فإن العداد يبدأ بالعد من الصفر مع كل نبضة قادمة على قطب الدخل ويستمر حتى يصل إلى القيمة المخزنة مسبقاً الموجودة في المسجل Overflow (TIMx_ARR) ، ثم يعود للقيمة صفر ويولد حدث الطفحان Update event مع كل طفحان للعداد كمل يتم توليد حدث

Up counting



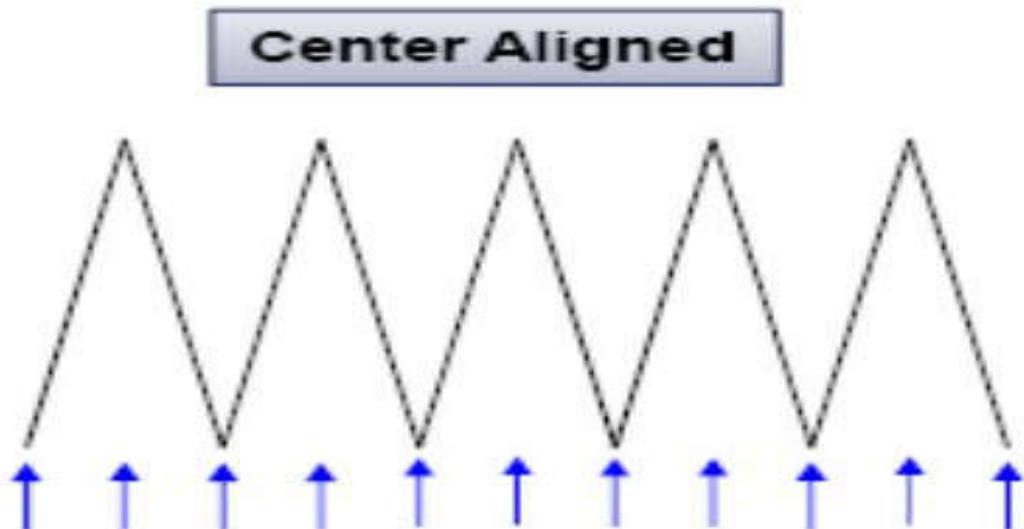
نط العد التنازلي Down-counting Mode

في هذا النط فإن العداد يبدأ بالعد من القيمة المخزنة في المسجل TIMx-ARR مع كل نبضة قادمة على قطب الدخل ويستمر ليصل إلى الصفر ثم يعود ليبدأ من القيمة المخزنة سابقاً ويولد حدث Underflow event أيضاً يتم توليد حدث Underflow



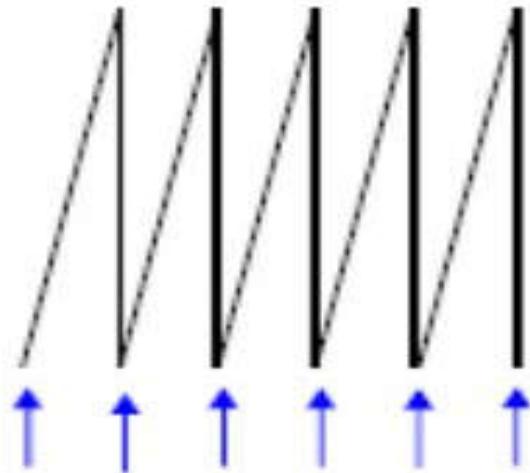
نط العد التصاعدي تنازلي :Center-Aligned Mode □

في هذا النط فإن العد يبدأ بالعد التصاعدي من الصفر ويستمر بالعد مع كل نبضة قادمة على قطب الدخل حتى يصل إلى القيمة المخزنة سابقاً **auto-reload value** في المسجل TIMx-ARR ثم يبدأ بالعد التنازلي من القيمة المخزنة سابقاً **auto-reload value** مع كل نبضة قادمة على قطب الدخل وحتى يصل إلى الصفر عندها يتم توليد حدث Underflow ثم يعود للعد التصاعدي من الصفر وهكذا...،

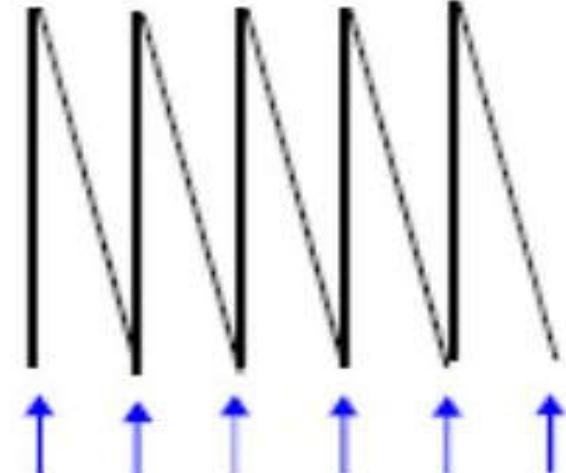


نط^ط العد^{اد} Counter

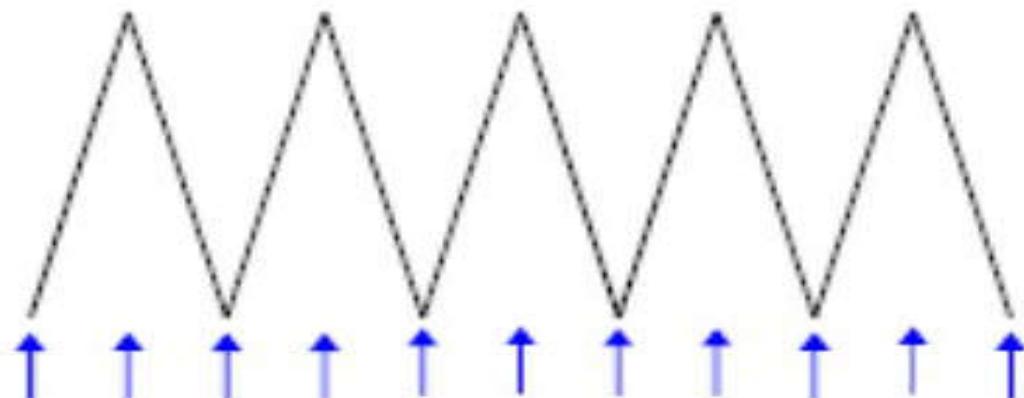
Up counting



Down counting

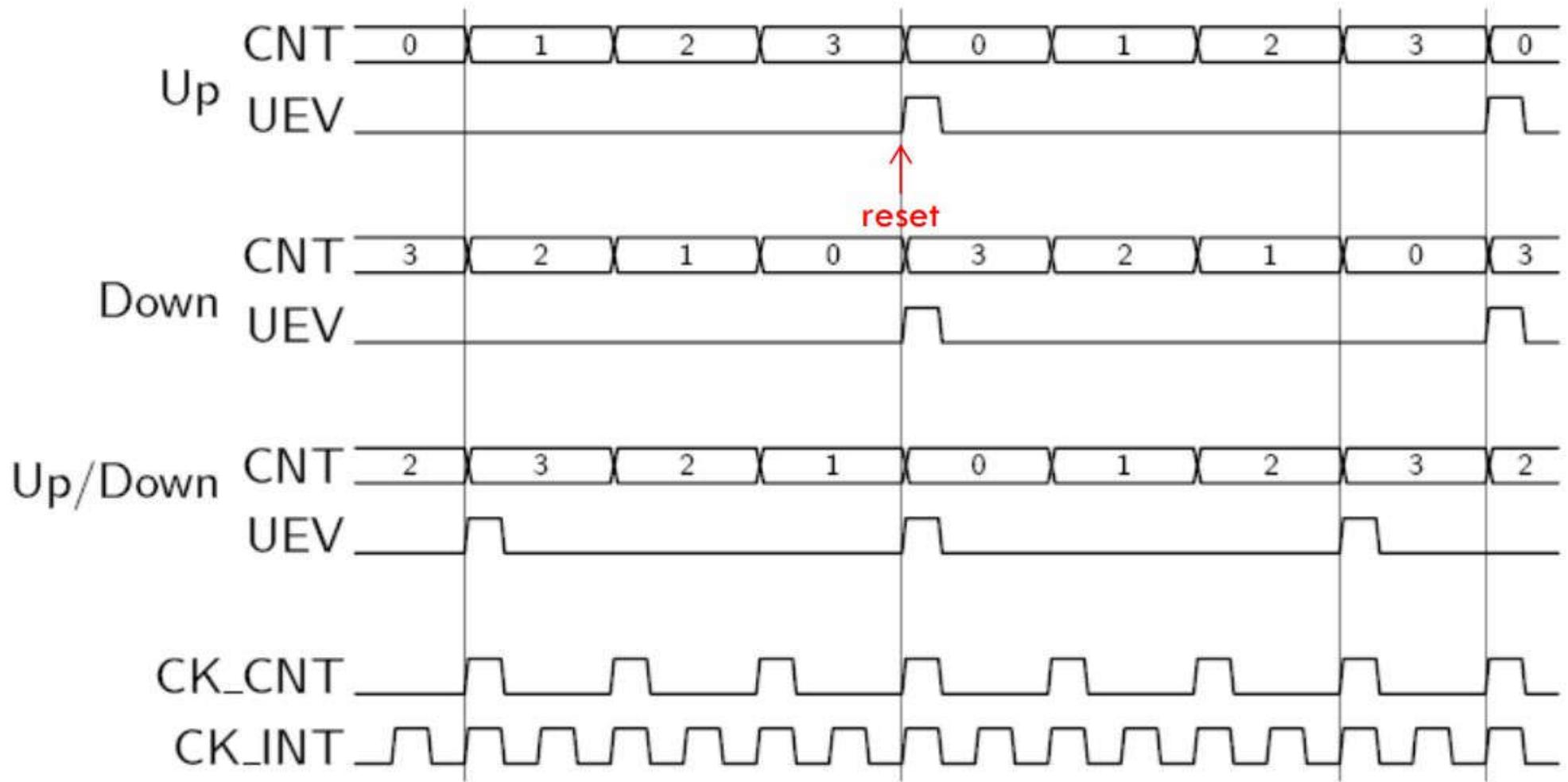


Center Aligned



نطع العداد Counter

ويكون المخطط الزمني لأنماط العد الثلاث : □

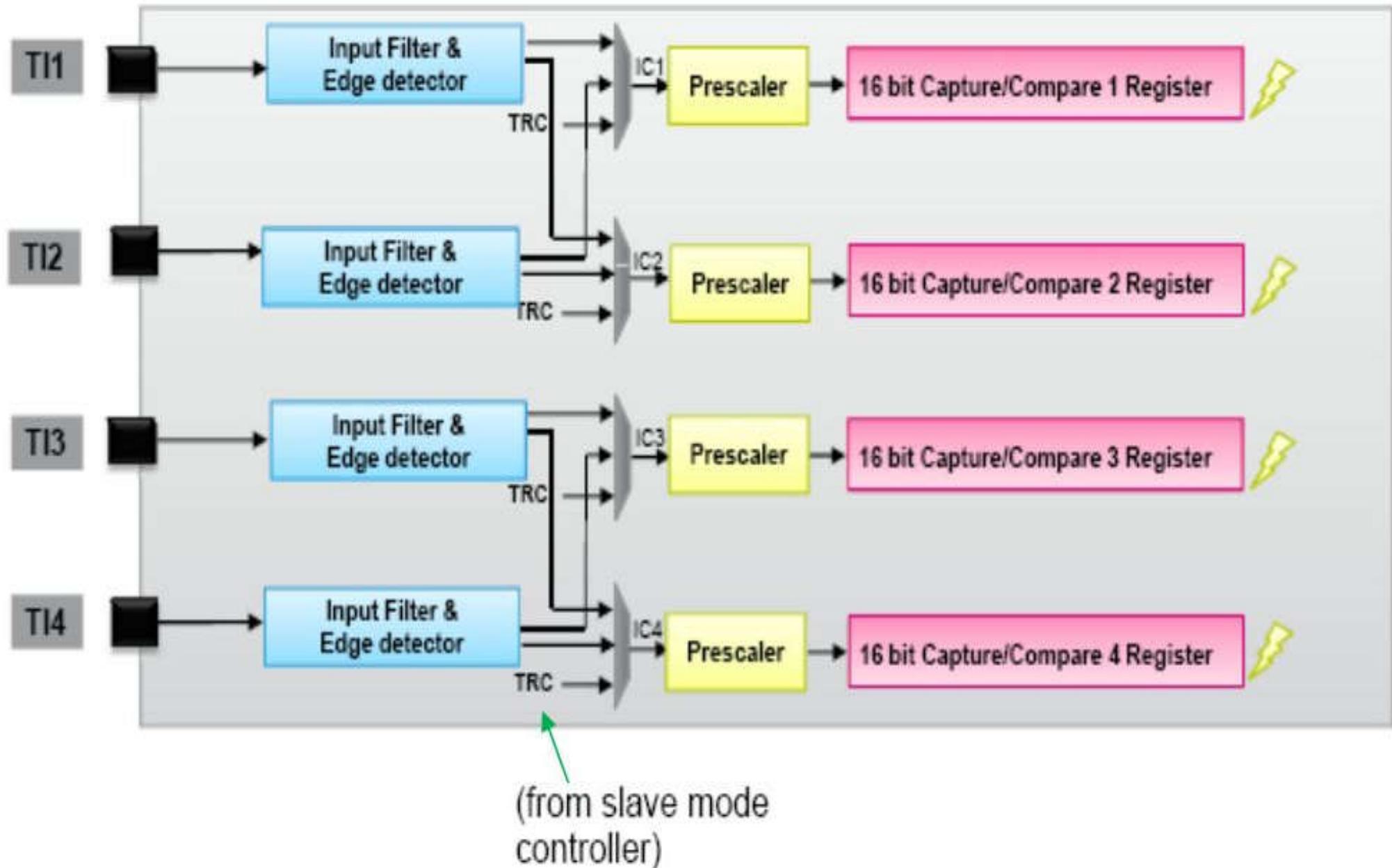


Counter Modes (ARR=3, PSC=1)

نط^ط Input Capture mode

- في نط^ط Input Capture يستقبل المؤقت نبضات الساعة الخاصة به من مصدر داخلي (ساعة المتحكم بعد استخدام المقسم التردد^y)
- يستمر بالعد إلى أن يحدث حدث معين (جبهة صاعدة/ جبهة هابطة) على قطب المتحكم الخاص بقناة الـ Input Capture Channel عند^{ها} يتم حفظ القيمة التي وصل إليها المؤقت إلى مسجل capture register
- لكل مؤقت في متحكمات STM32 عدة قنوات input (channels) مرتبطة بأقطاب outputcapture compare المتحكم يمكن معرفتها من خلال الـ datasheet الخاصة بالمتحكم

Input Capture mode نمط



نقط PWM mode

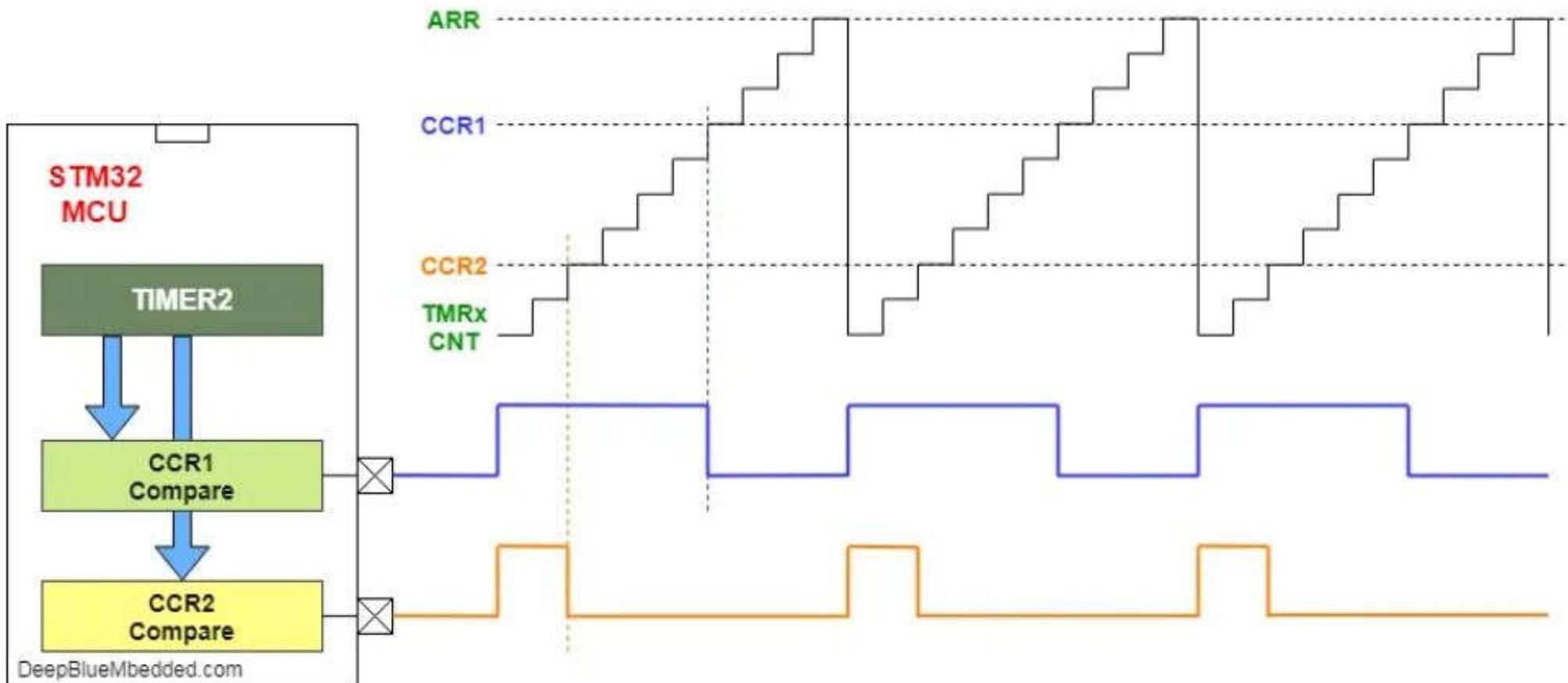
في نمط PWM mode يستقبل المؤقت نبضات الساعة الخاصة به من الساعة الداخلية للمتحكم حيث يبدأ بالعد من الصفر ويزداد مع كل نبضة ساعة للمتحكم (طبعاً مع مراعاة إعدادات المقسم الترددية للمؤقت)

يتم وضع قطب الخرج الخاص بالـ PWM في وضع HIGH ويبقى كذلك إلى أن يصل العداد إلى القيمة المخزنة في المسجل CCRx عندما يصبح قطب الخرج في وضع LOW إلى أن يصل العداد إلى القيمة المخزنة في المسجل ARRx ، وهذا

يدعى شكل الإشارة الناتجة بالـ PWM(Pulse Width Modulation) ، حيث يتم التحكم بالتردد من خلال تردد الساعة الداخلية للنظام، والمقسم الترددية Prescaler بالإضافة إلى قيمة المسجل (Auto Reload register) ، كما يتم تحديد قيمة دورة التشغيل即 duty cycle من خلال قيمة المسجل الـ CCR1

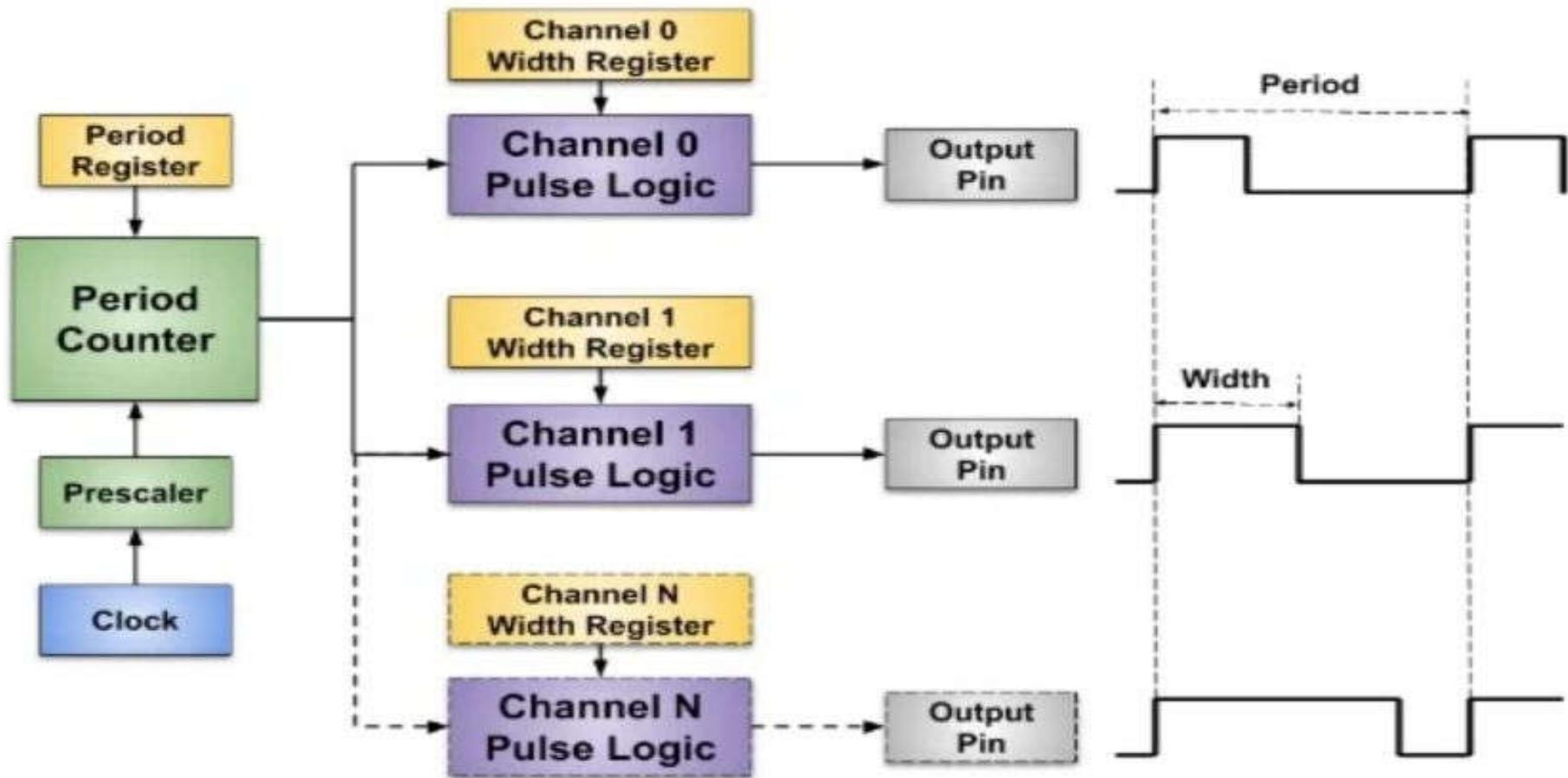
نقط PWM mode

يوضح المخطط التالي كيفية تأثير قيمة المسجل ARR في دور(تردد) إشارة الـ PWM ، وكيف تؤثر قيمة المسجل CCR1 في قيمة دورة التشغيل duty cycle



PWM mode نمط

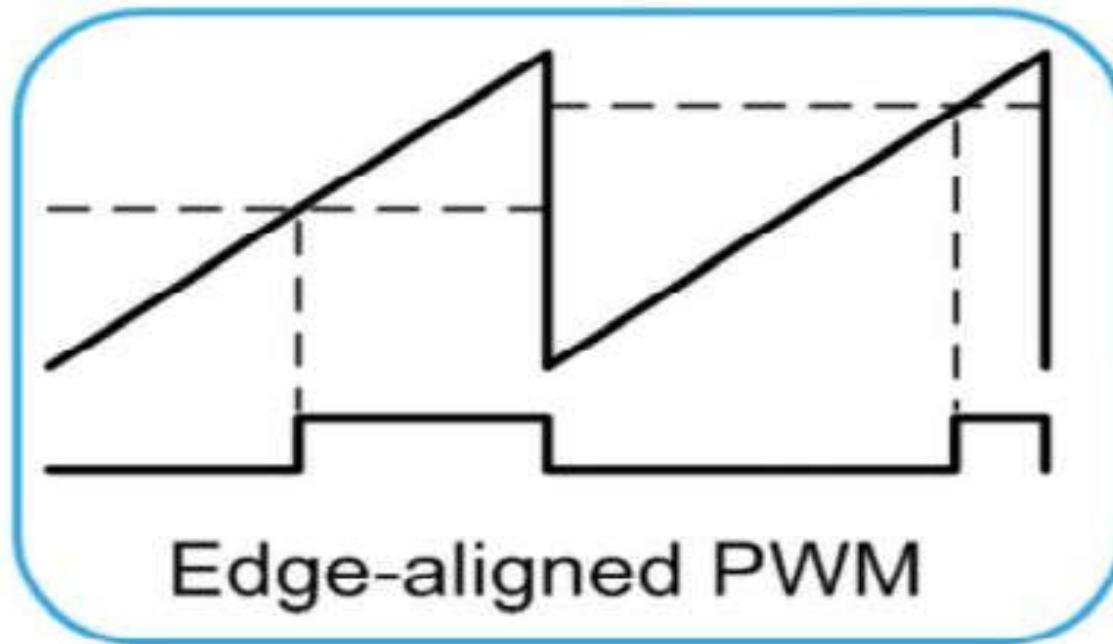
لكل مؤقت من مؤقتات المتحكم STM32 عدة قنوات ، لذا فإن كل مؤقت بإمكانه توليد عدة إشارات PWM لكل منها دورة تشغيل مختلفة ولكن لها نفس التردد وتعمل بالتزامن مع بعضها



PWM mode نمط PWM

أنماط الـ PWM المختلفة:

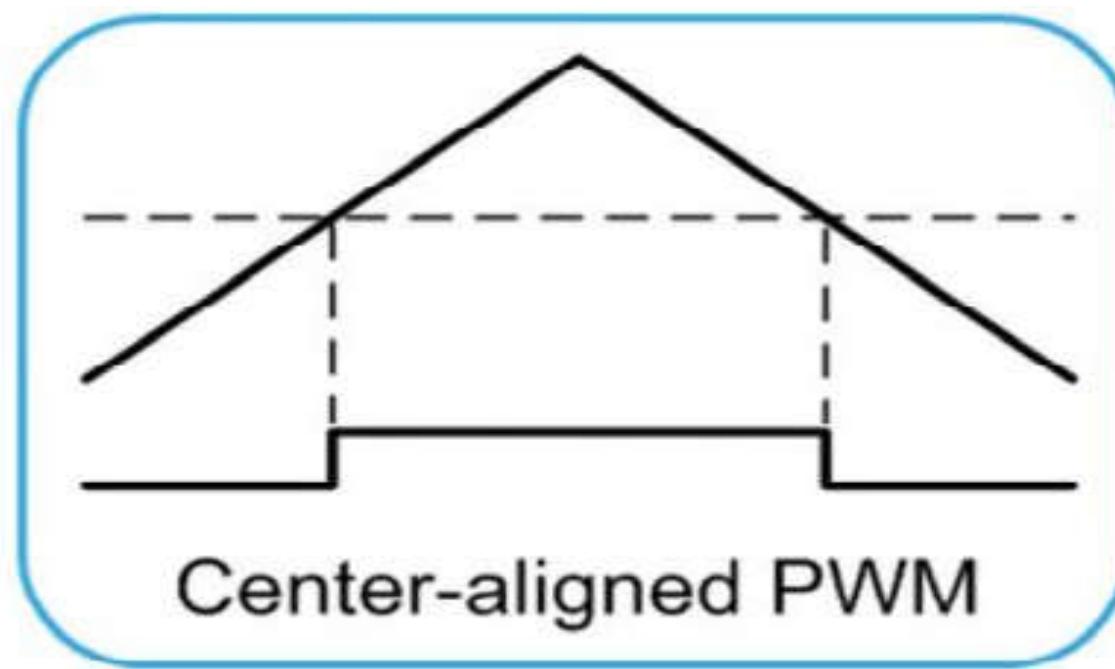
□ في هذا النمط فإن العداد يقوم بالعد بشكل تصاعدي فقط أو تنازلي فقط، وبإمكان المؤقت الواحد أن يولد حتى الـ 6 إشارات PWM لها نفس التردد ولكن بدورات تشغيل مختلفة، وهذه الإشارات جميعها متزامنة باعتبار أن الجبهة الهاابطة هي نفسها لجميع الإشارات.



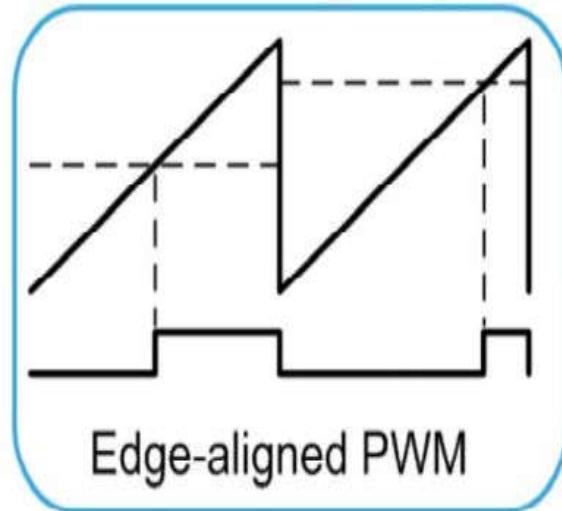
PWM mode نمط PWM

أنماط الـ PWM المختلفة:

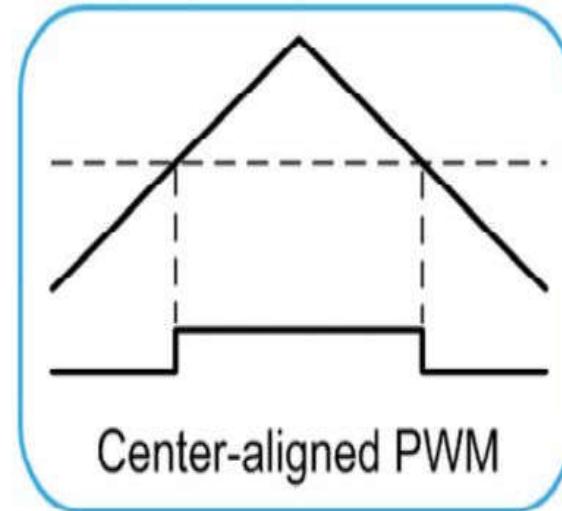
في هذا النمط فإن العداد يقوم بالعد بشكل تصاعدي/تنازلي، وتكون إشارات الـ PWM الناتجة من مؤقت واحد غير متزامنة لأن الجبهة الهاابطة لكل منها مختلفة، لذا فإن أزمنة التبديل لكل إشارة PWM تكون مختلفة عن الإشارة الأخرى



PWM mode طرق

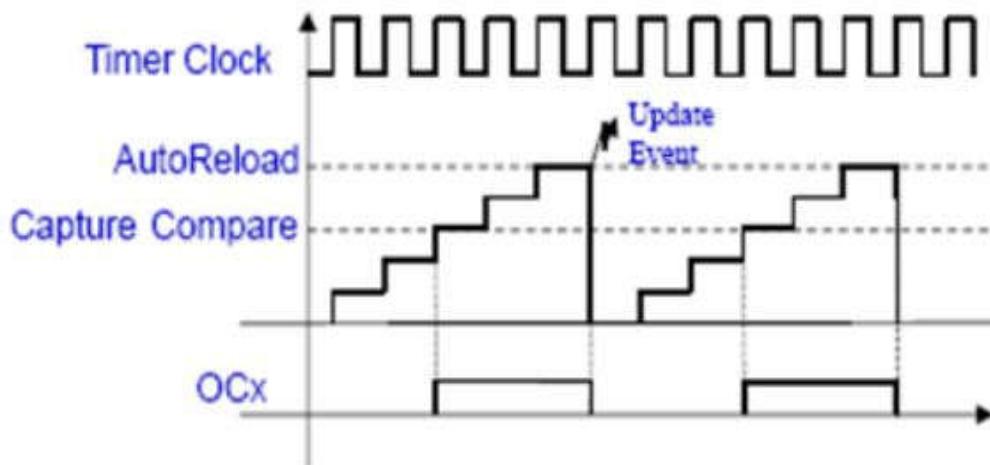


Edge-aligned PWM



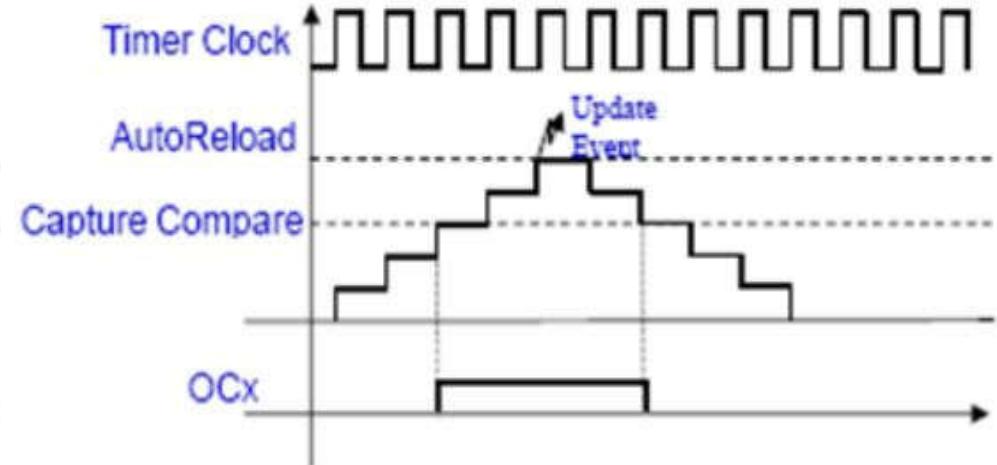
Center-aligned PWM

Edge-aligned Mode



PWM mode 2

Center-aligned Mode



هناك ثلاث أوضاع مختلفة لاستخدام المؤقتات هي:

وضع : Polling أي استخدام المؤقت بدون مقاطعة وفي هذه الحالة يجب فحص القيمة التي وصل إليها العداد بشكل يدوي داخل الكود بشكل مستمر أو يمكن بدلاً من ذلك فحص حالة العلم Flag أيضاً بشكل مستمر داخل الكود مما يؤدي إلى تعطيل العديد من وظائف المتحكم أو قد تسبب في عدم الوصول إلى القيمة المحددة بالضبط، لذا فإننا لن نستخدم هذا الوضع ضمن تطبيقاتنا.

توفر مكتبة HAL الدالة التالية لبدء المؤقت:

```
;()HAL_TIM_Base_Start
```

أوضاع الاستخدام المختلفة للمؤقتات في متحكمات STM32

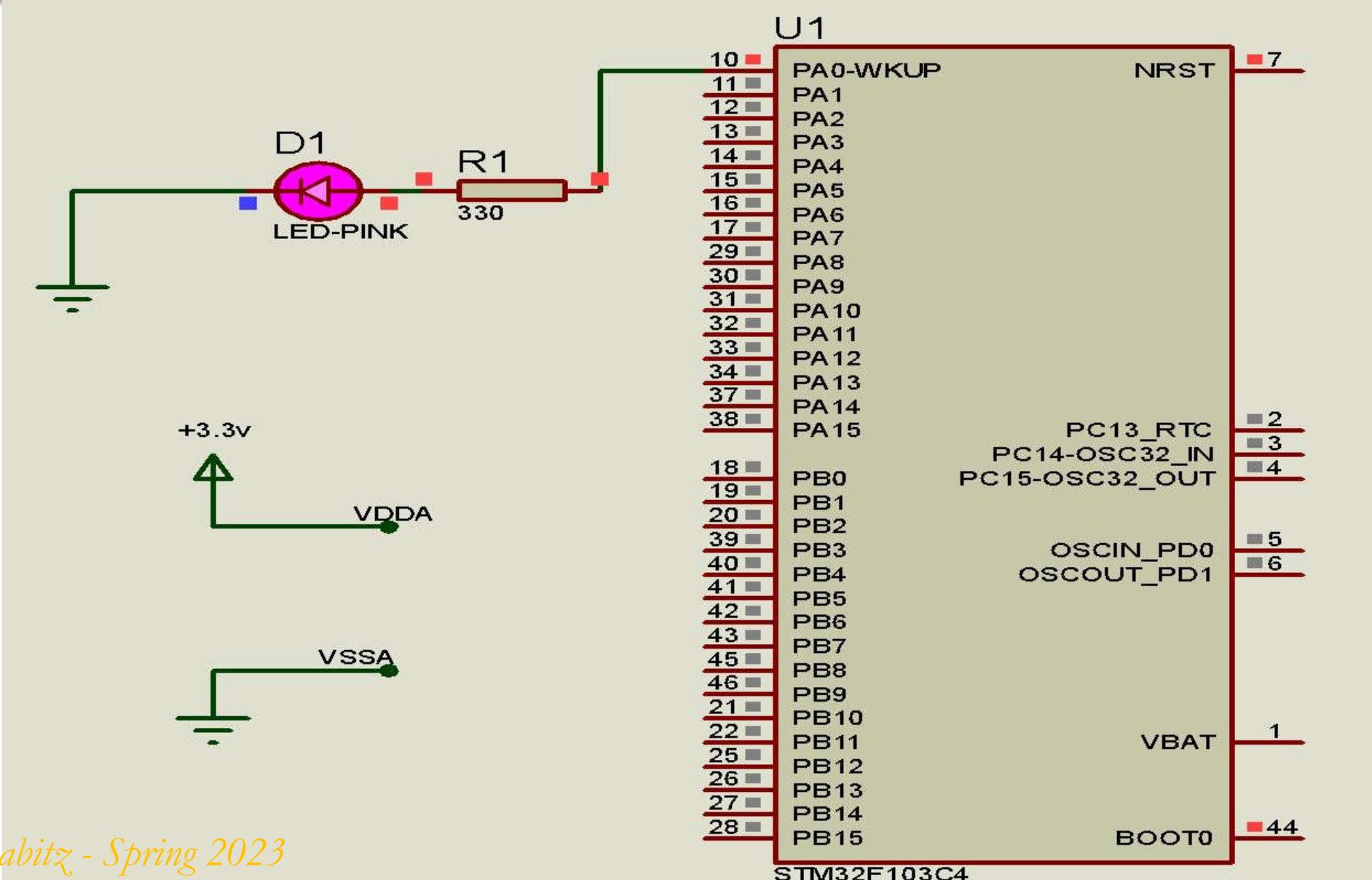
وضع Interrupt : في هذا الوضع عند الوصول إلى Overflow/underflow أو أي من أحداث المقاطة سيتم التوجه آلياً لتنفيذ برنامج خدمة المقاطة، وهذا الوضع الذي سستخدمه في جميع التطبيقات القادمة.

توفر مكتبة HAL الدالة التالية لبدء المؤقت في وضع المقاطة:

```
;()HAL_TIM_Base_Start
```

وضع DMA :

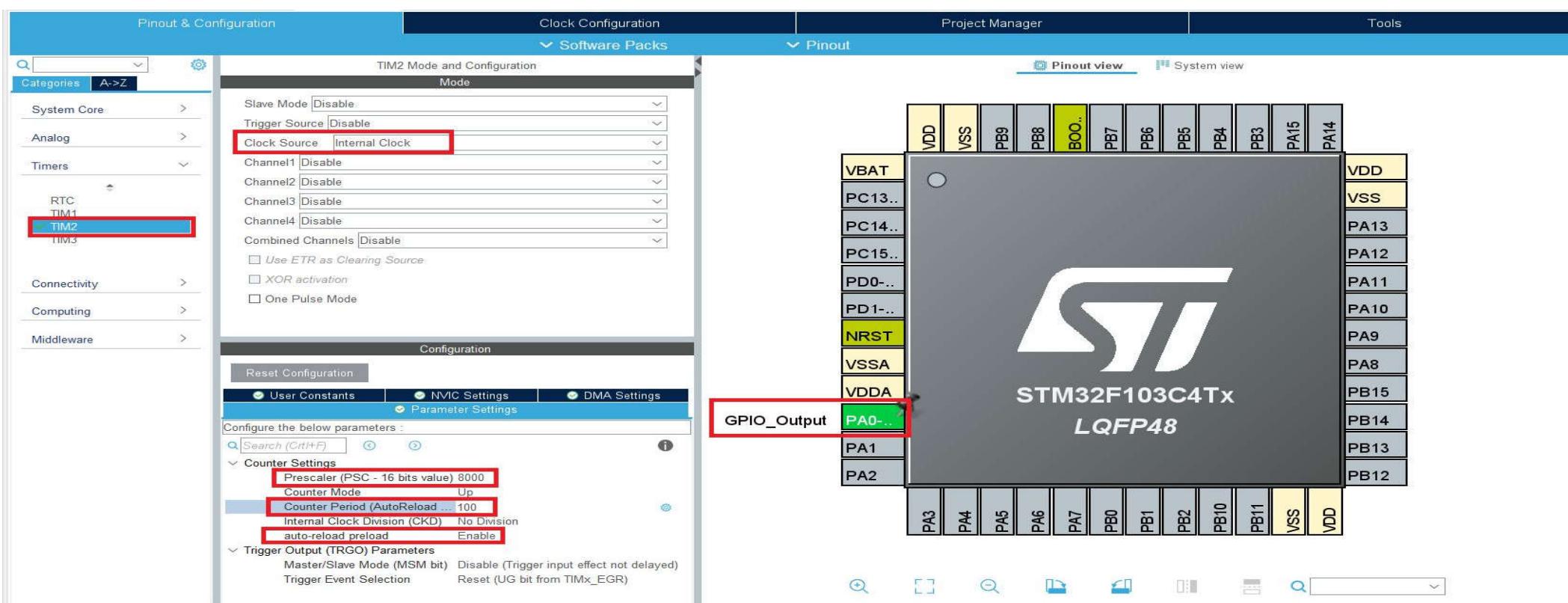
التطبيق العملي الأول : استخدام المؤقت في نمط Timer mode
 وبوضع المقاطةعة لتوليد زمن بدلاً من استخدام دالة `delay()`
 واستخدامه في عمل Toggle للبِد الموصول على القطب PA5



التطبيق العملي : استخدام المؤقت في نمط Timer mode وبوضع المقاطعة لتوليد زمن بدلًا من استخدام دالة delay() واستخدامه في عمل Toggle للبُعد الموصول على القطب PA5

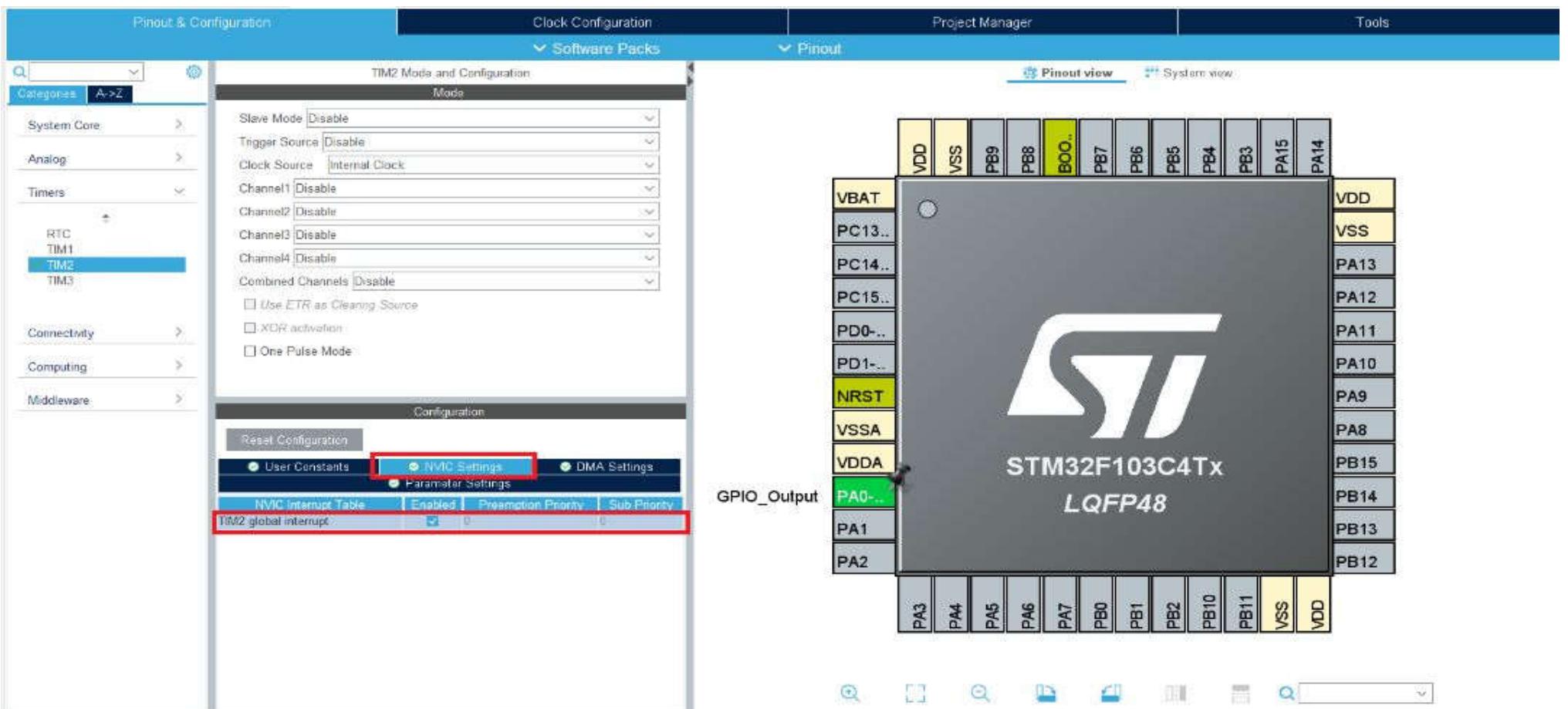
التطبيق العملي : استخدام المؤقت في نمط Timer mode وبوضع المقاطة لـ توليد زمن بدلًا من استخدام دالة delay() واستخدامه في عمل Toggle للبُعد الموصول على القطب PA5

سنقوم باختيار مصدر الساعة للمؤقت داخلي، المقسم الترددية 8000 ، الـ $\text{Preload} = 100$ ، أيضاً سنقوم بتفعيل إعادة التحميل التلقائي، كما في الشكل التالي:

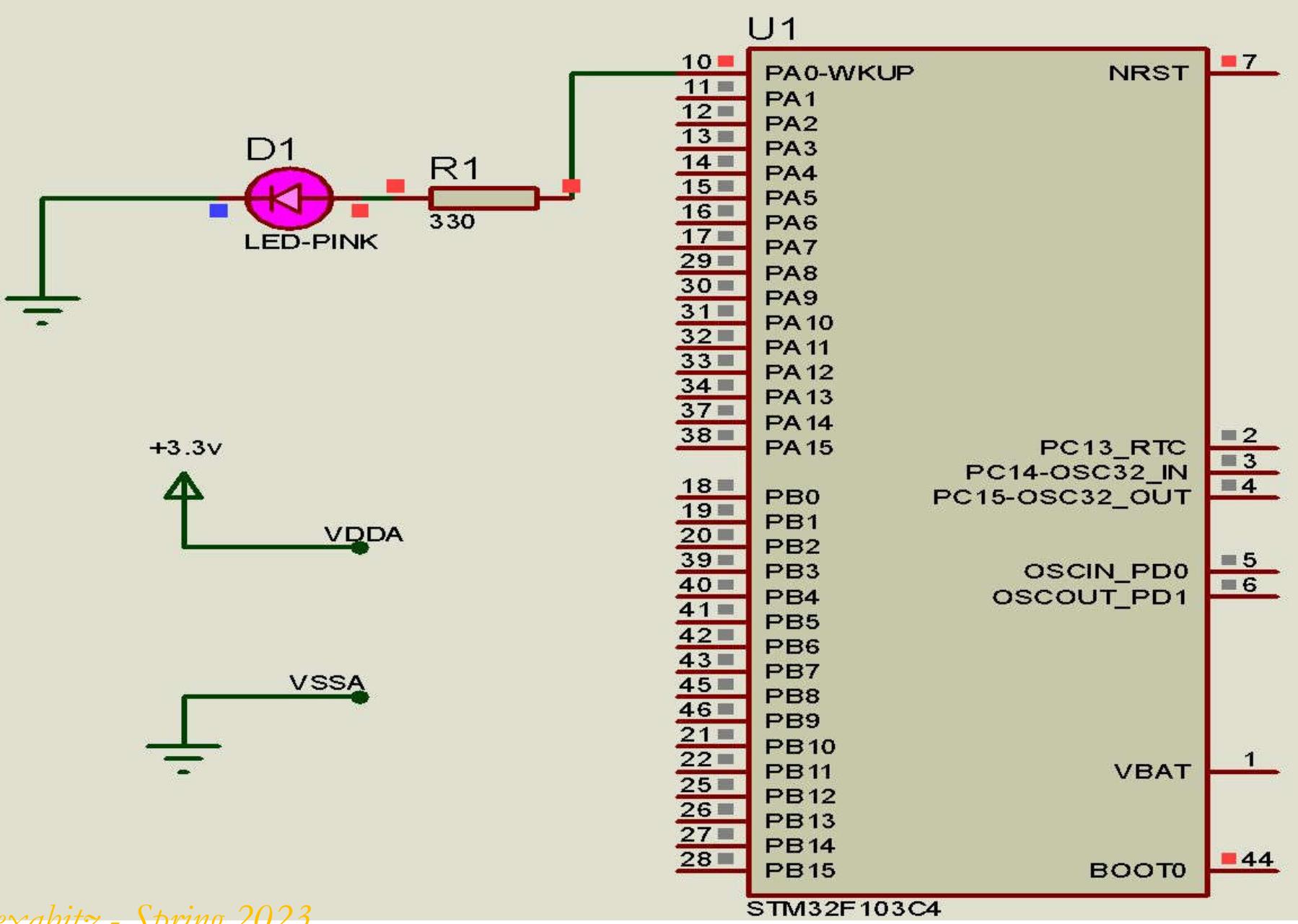


التطبيق العملي : استخدام المؤقت في نمط Timer mode وبوضع المقاطةعة لتوليد زمن بدلًا من استخدام دالة ()delay واستخدامه في عمل Toggle للباد الموصول على القطب PA5

نقوم بتفعيل مقاطعة المؤقت من شريط الـ NVIC tab □



التطبيق العملي الثاني : استخدام المؤقت في نمط PWM mode للتحكم في شدة إضاءة اليد



ساعة الزمن الحقيقي (RTC)

- لوحدة RTC مخرجان لهما القدرة على توليد تنبيهان قابلان للبرمجة ولهمما القدرة على إيقاظ المعالج من كافة أنماط توفير الطاقة
- تحتوي وحدة RTC على مؤقت مدمج قابل للضبط وإعادة تحميل القيمة آلياً والذي يستخدم لتوليد مقاطعات دورية لها القدرة على إيقاظ المعالج ، كما يمكن ضبط دقة هذا المؤقت