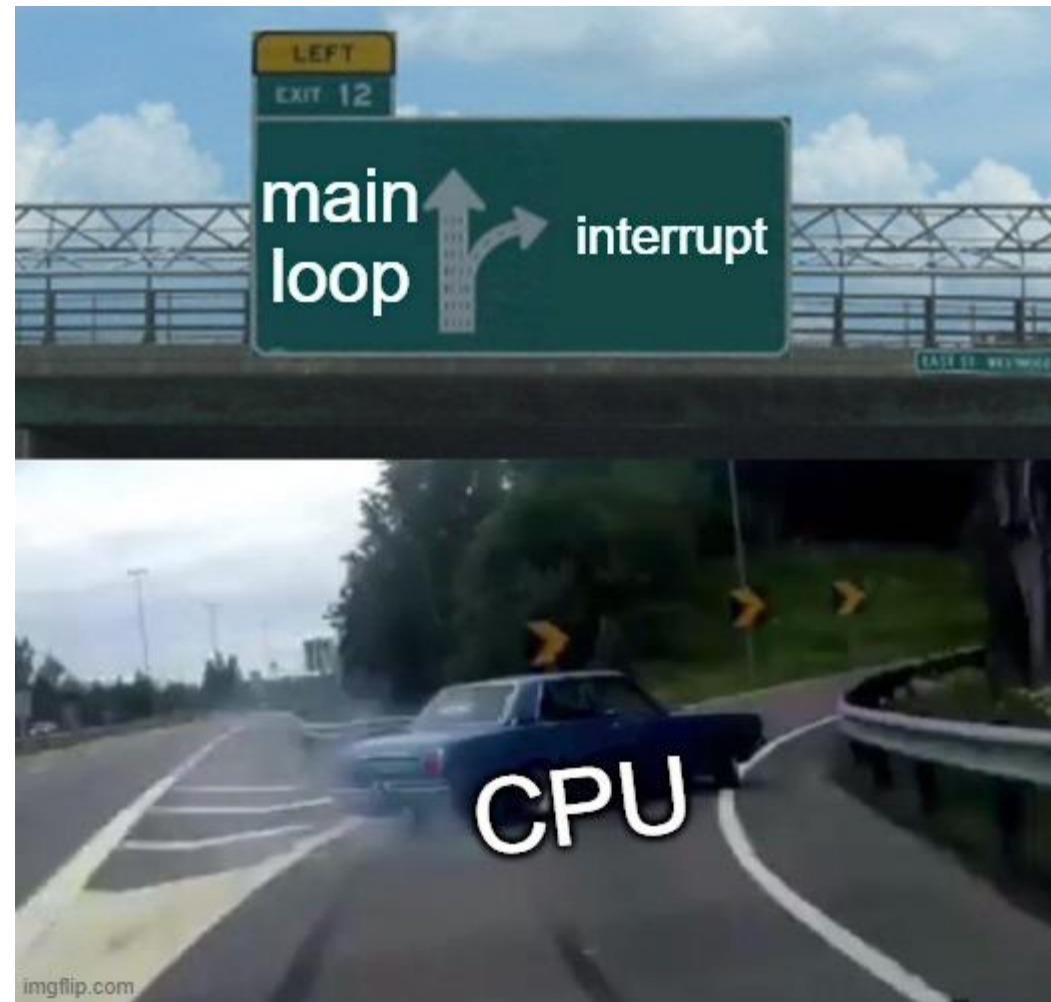


6.Interrupt

DR.SOMSIN THONGKRAIRAT



life.augmented



Asynchronous programming

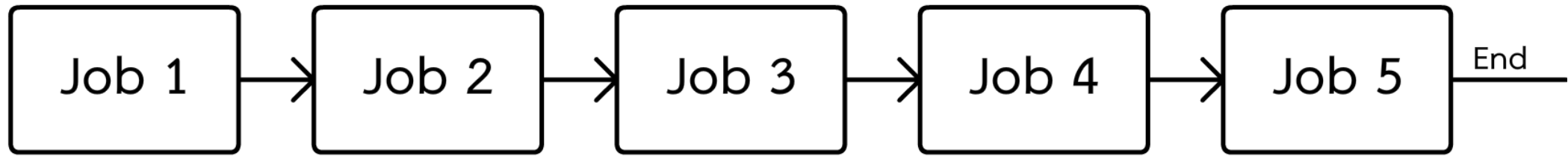
นิยาม -> Something that not ordered , อะไรก็ตามที่ไม่เรียงตามลำดับ

เช่น มีงานที่ 1 2 3 4 5 ที่ต้องทำ

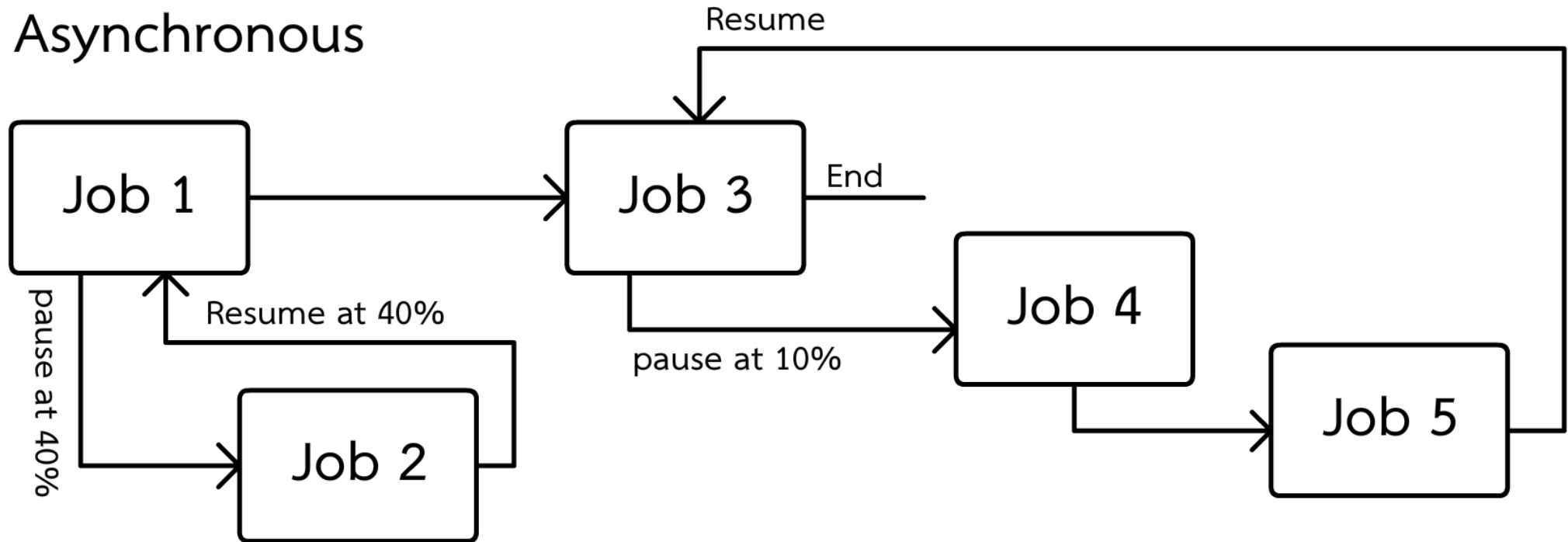
synchronous -> ทำตามลำดับ ทำงานที่ 1 เสร็จ ค่อยทำงานต่อไป

Asynchronous -> ทำอะไรก่อนก็ได้ หรือทำๆ หยุดๆ ก็ได้

Synchronous



Asynchronous



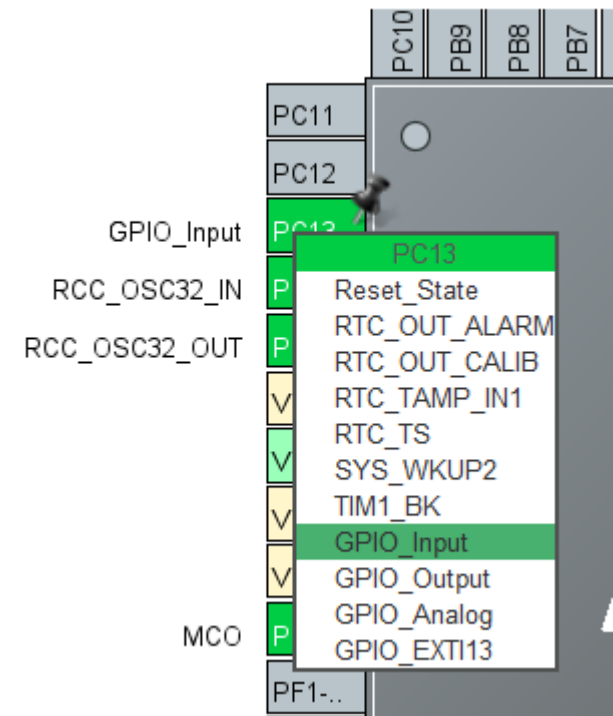
Polling method (Synchronous method)

IO toggle program (กดปุ่มแล้วไฟติด ปลั่อยปุ่มแล้วไฟดับ)

เขียนอย่างไร? Polling method

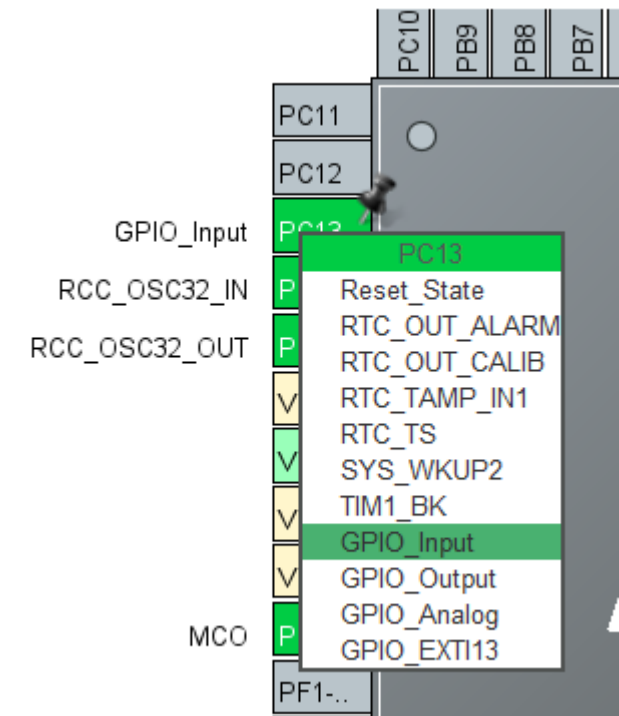
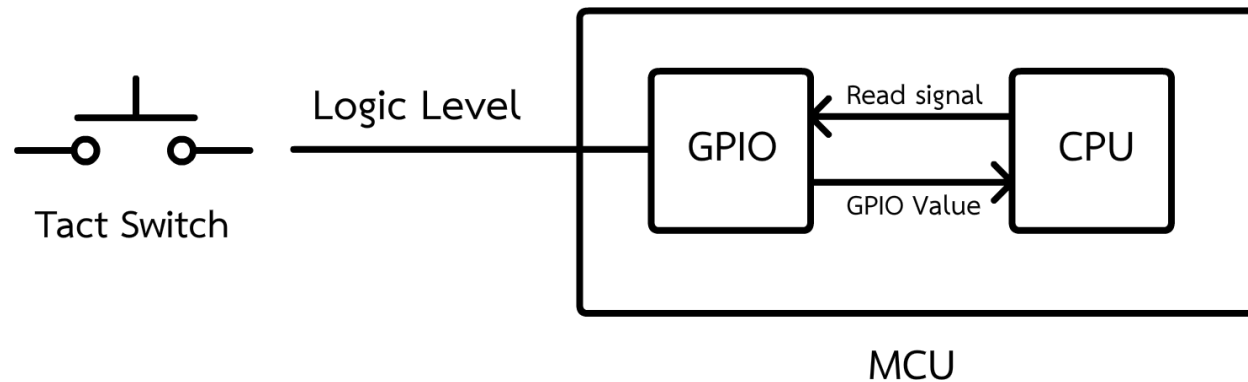
```
int state = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13);  
HAL_GPIO_WritePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin, state);  
HAL_Delay(1000); // do another Task
```

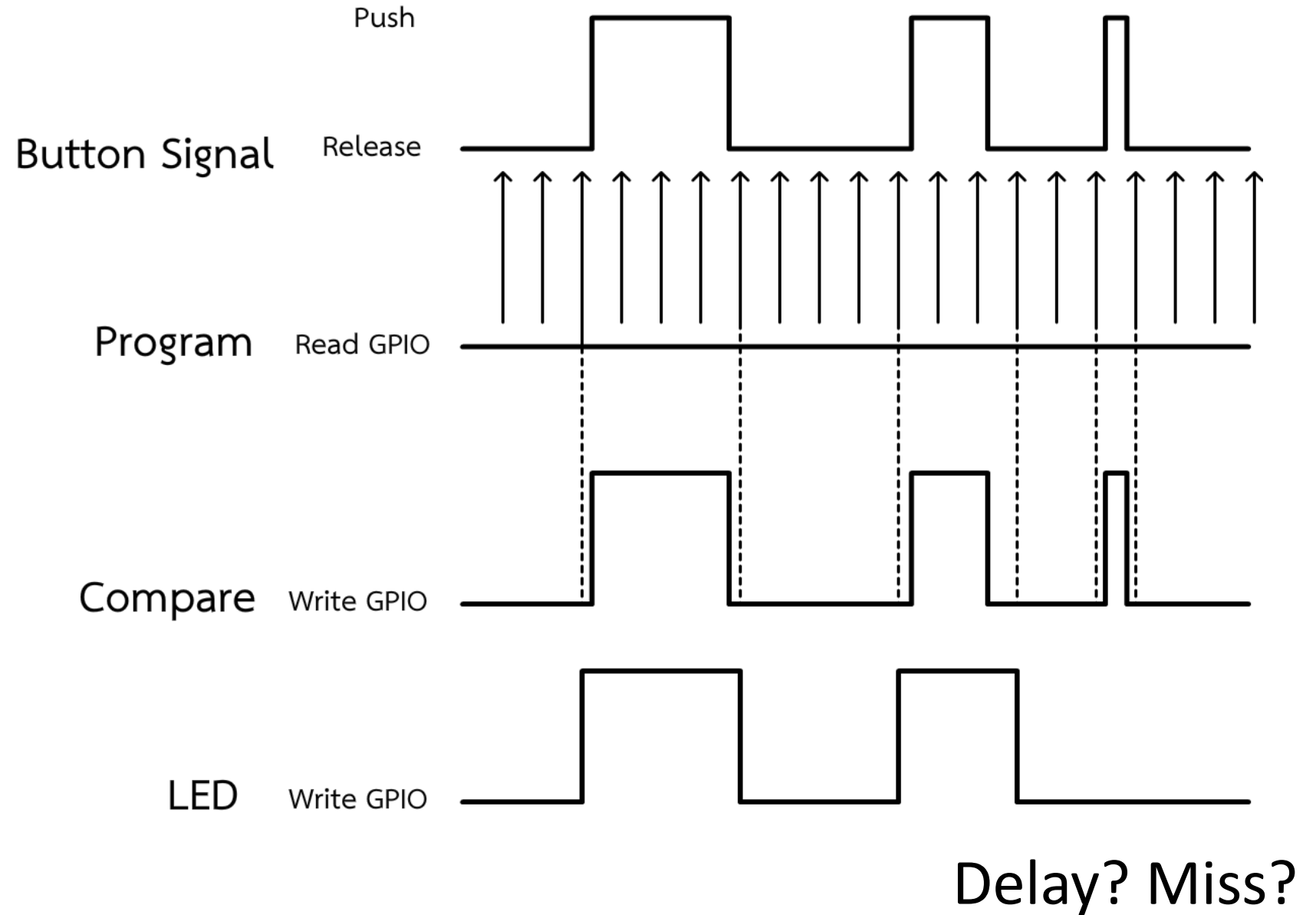
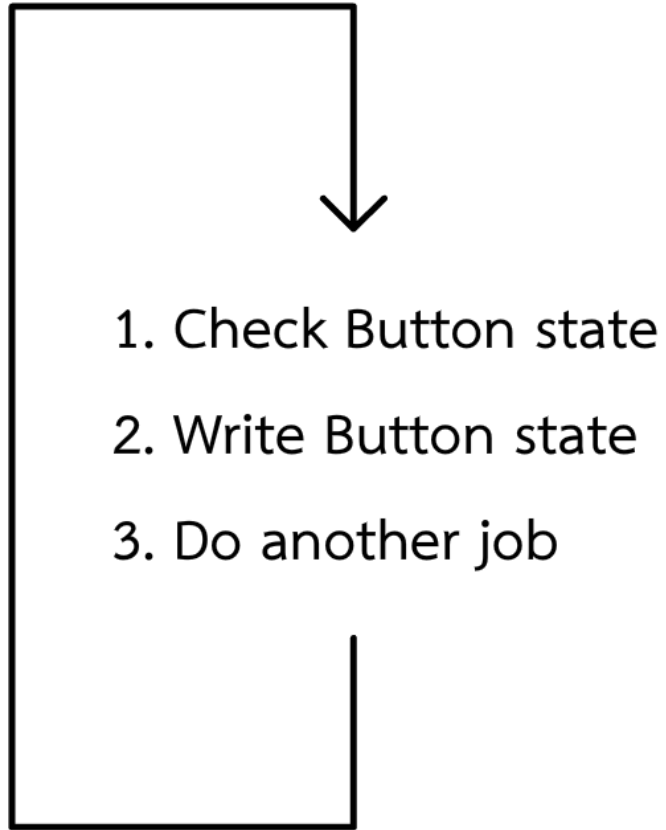
ผลลัพธ์เป็นอย่างไร?



Polling Button

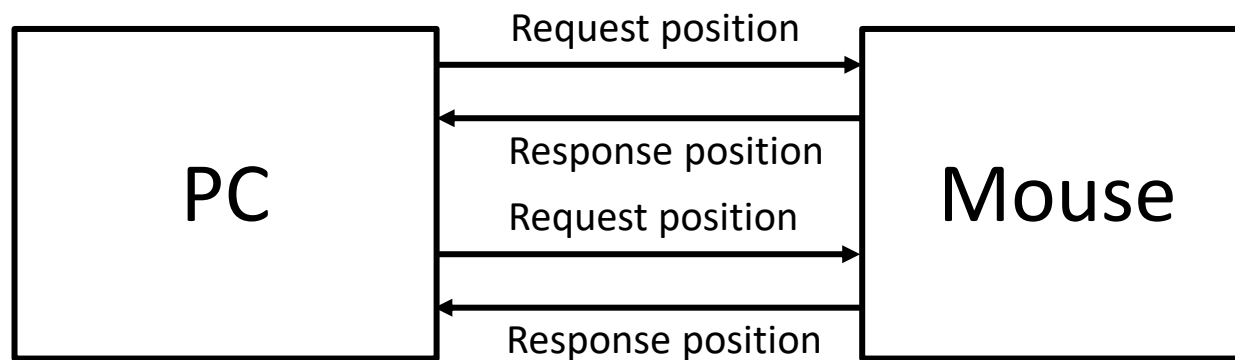
CPU จะอ่านข้อมูลจาก GPIO เรื่อย ๆ แม้ Stage หรือ logic ของ GPIO จะเหมือนเดิมก็ตาม





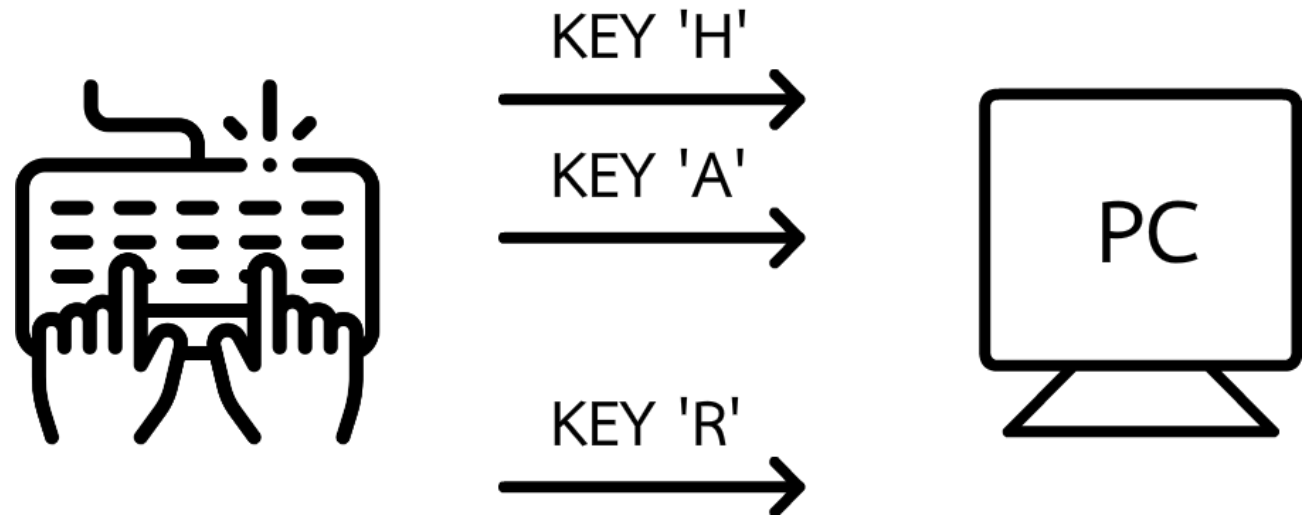
Polling method

- ความถี่ในการ sampling สม่ำเสมอ เช่น Mouse
- ข้อมูลที่ต้องการเก็บสถิติอย่างสม่ำเสมอ
- ข้อมูลที่ต้องการความถี่หรือ sampling rate คงที่ เพื่อนำไปวิเคราะห์ต่อ (ADC)



Asynchronous method

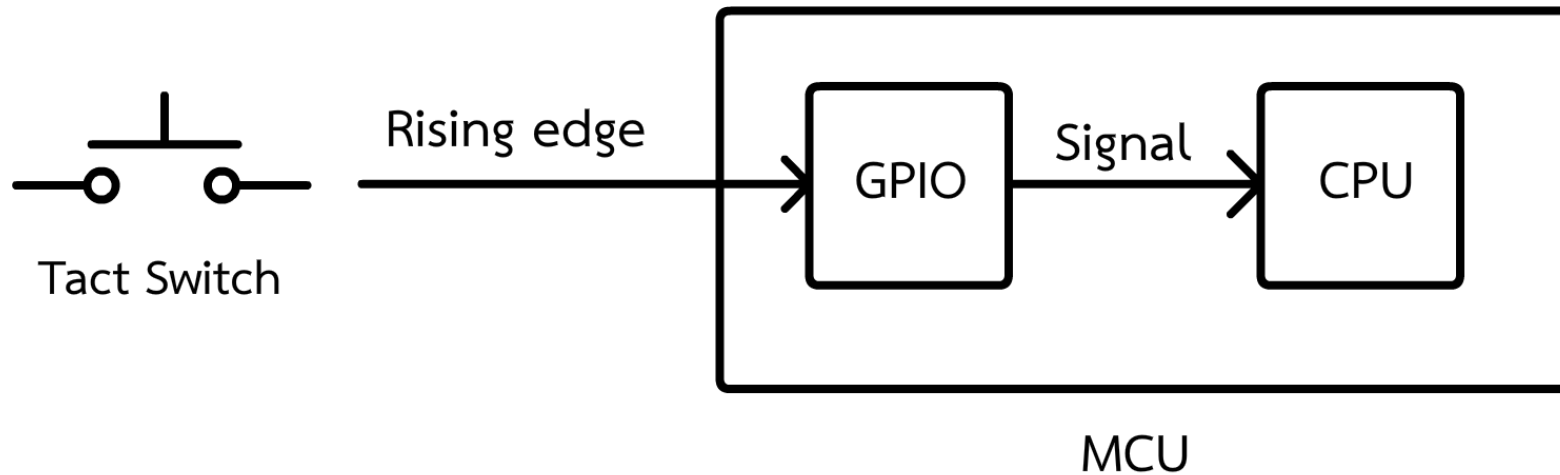
Asynchronous method คือการใช้ *signal*(สัญญาณ) หรือ *event* เป็น จุดเริ่มต้นของการทำงาน โดยที่ peripheral เป็นตัวเริ่มการสื่อสาร เช่น การกด Keyboard แล้ว USB ส่ง interrupt มายัง PC



* Polling method CPU จะเป็นตัวเริ่มต้นการสื่อสาร

Asynchronous method

CPU จะได้รับสัญญาณเมื่อ เกิดการเปลี่ยนแปลงของสัญญาณตามที่ได้ config เอาไว้ หรือ มี event เกิดขึ้น

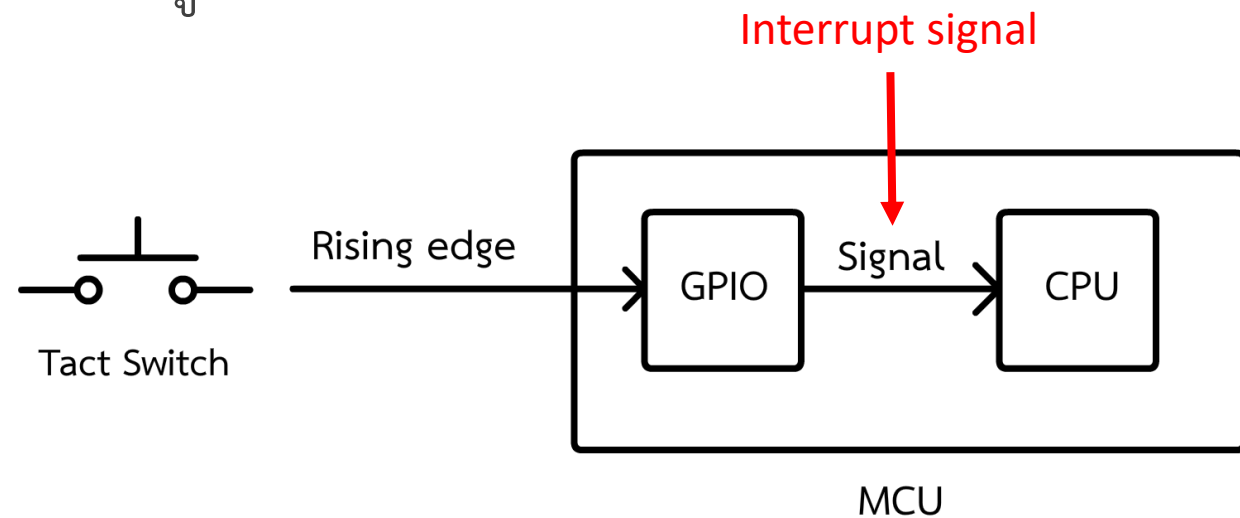


Interrupt

Asynchronous method in MCU

Using interrupt signal trigger CPU to response with event (change)

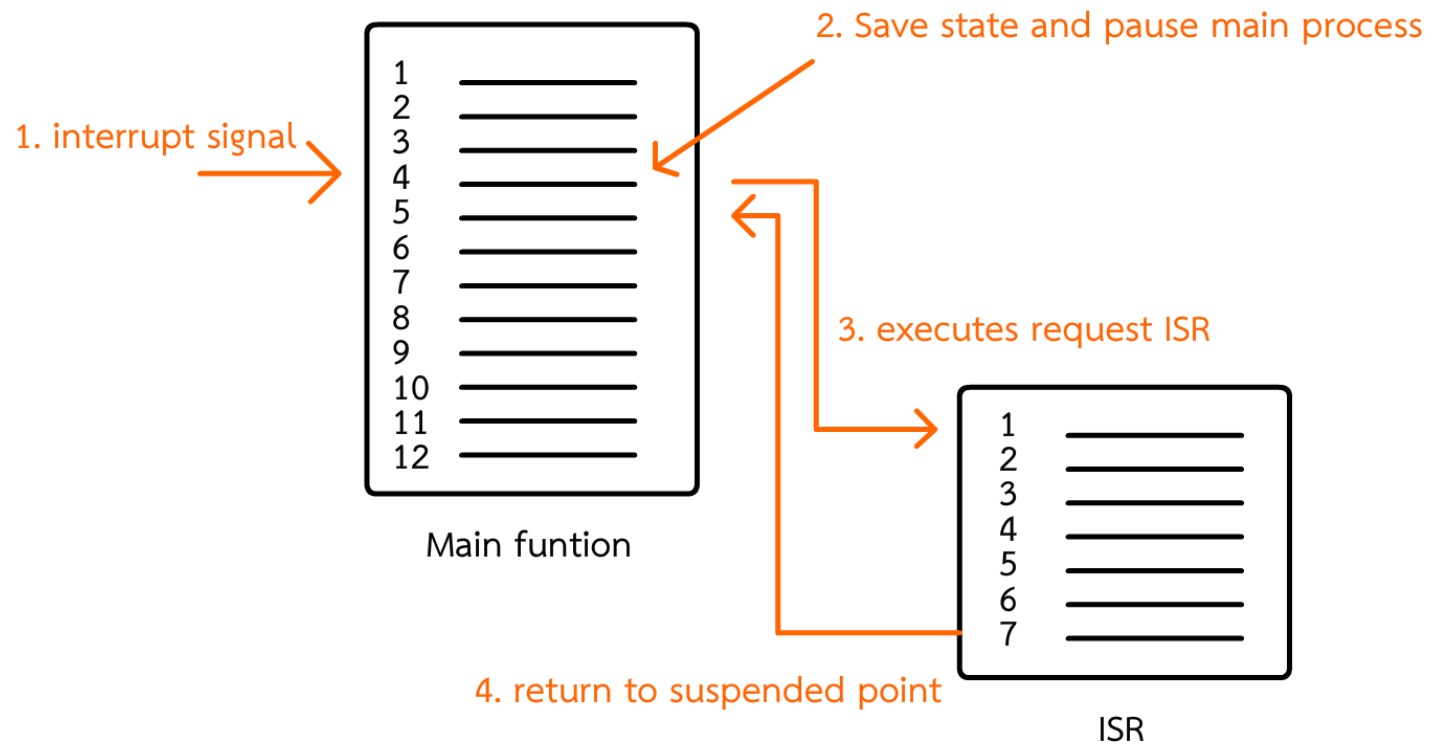
ใช้สัญญาณ interrupt เพื่อทำให้ CPU รู้ถึง Event

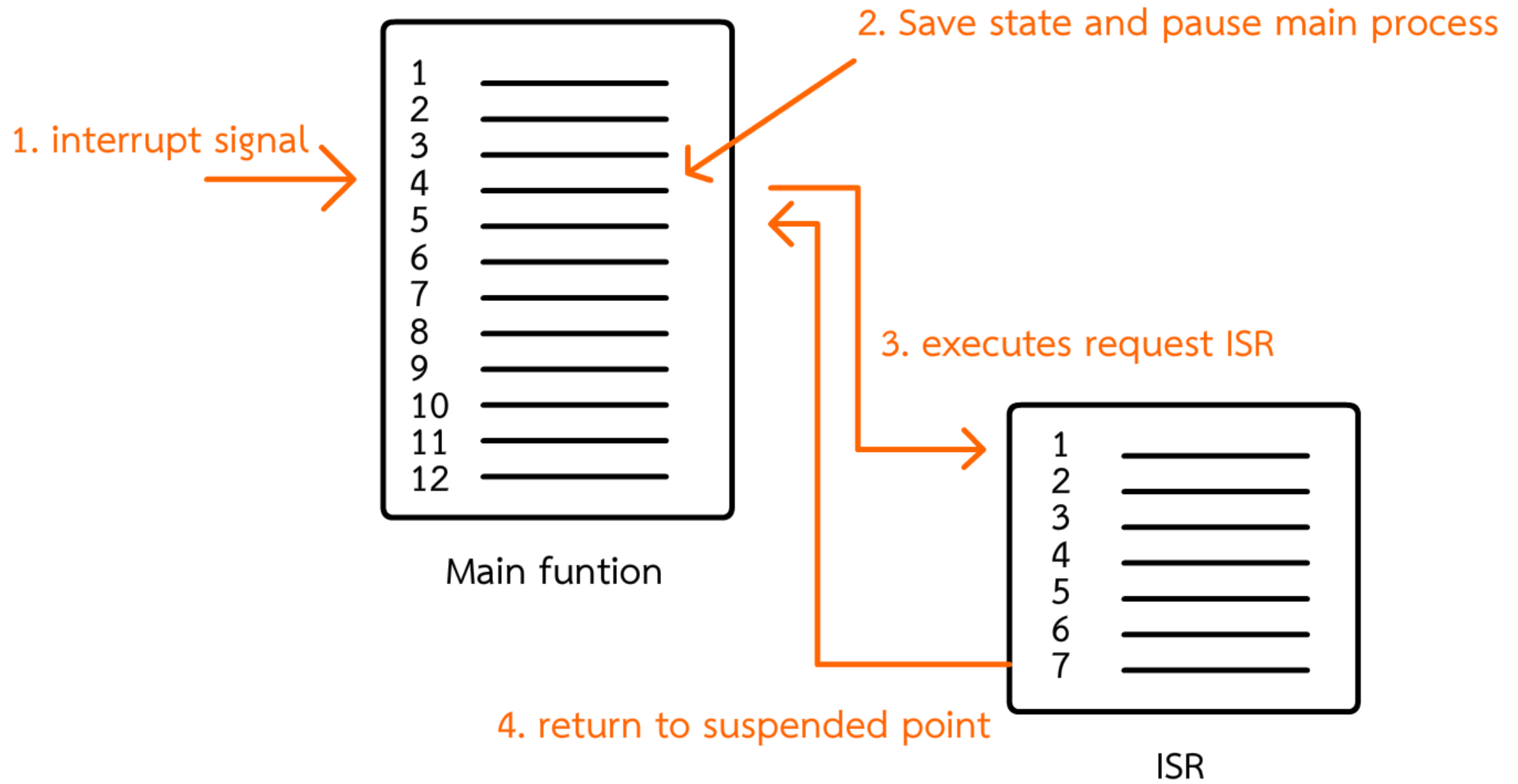


ISR (interrupt service routine)

ISR function or routine that
response an interrupt signal

ISR คือ function หรือการทำงานที่
ตอบสนองกับ สัญญาณ interrupt





ISR in CUBE ide

In Callback function form , อยู่ในรูปแบบของ Callback function

```
281 /* USER CODE BEGIN 4 */
282 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
283 {
284     if (htim == &htim2 )
285     {
286         HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin);
287     }
288 }
289 /* USER CODE END 4 */
```

เมื่อ เกิด Timer interrupt (OVF interrupt)
Callback function นี้จะถูกเรียก

LAB 2 GPIO Interrupt

1. Set pin as GPIO_EXTI

2. Set set interrupt mode

3. Enable External interrupt

The screenshot shows the 'Pinout & Configuration' window for an STM32 microcontroller. The 'Categories' list on the left includes System Core, Analog, Timers, Connectivity, Multimedia, Computing, Middleware and Software Packs, and Utilities. The 'Pinout' tab is active, showing a pin list. A red arrow points to the 'GPIO_EXTI13' option for pin PC13.

Pin	Signal	GPIO o...	GPIO mo...	G...	Maxim...	Fa...	User Label	Modified
PA5	n/a	Low	Output P...	N...	High	n/a	LED_GREEN	<input checked="" type="checkbox"/>
PC13	n/a	n/a	External ...	N...	n/a	n/a		<input checked="" type="checkbox"/>

PC13 Configuration :

GPIO mode: External Interrupt Mode with Rising edge trigger detection

GPIO Pull-up/Pull-down: External Interrupt Mode with Rising edge trigger detection

User Label: External Interrupt Mode with Falling edge trigger detection

External Event Mode with Rising edge trigger detection

External Event Mode with Falling edge trigger detection

External Event Mode with Rising/Falling edge trigger detection

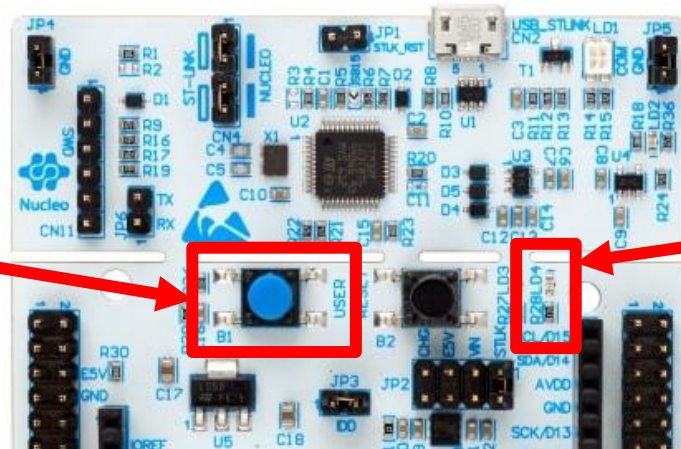
NVIC

NVIC Interrupt Table	Enabled	Preemption Priority
EXTI line 4 to 15 interrupts	<input checked="" type="checkbox"/>	0

Coding

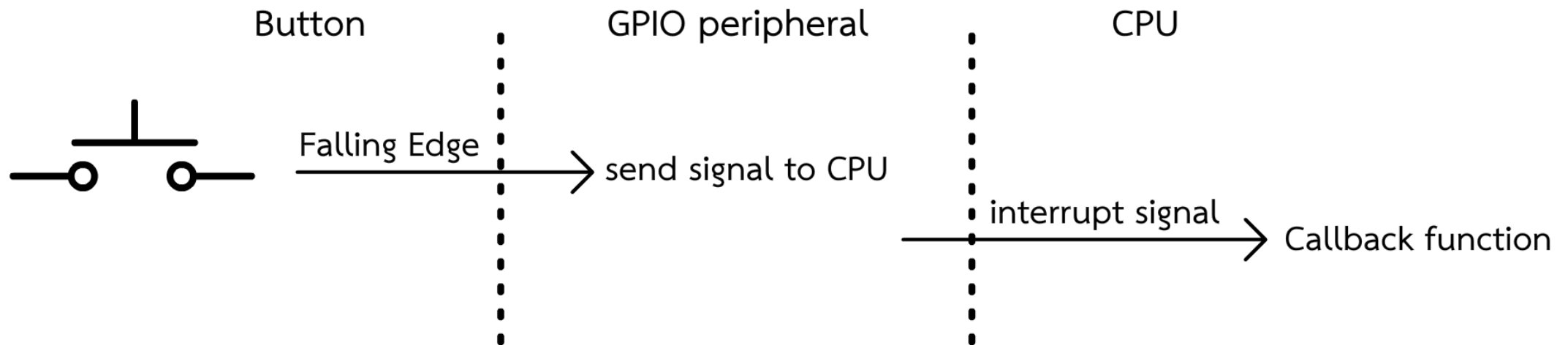
```
/* USER CODE BEGIN 4 */  
void HAL_GPIO_EXTI_Falling_Callback(uint16_t GPIO_Pin)  
{  
    if(GPIO_Pin == GPIO_PIN_13) {  
        HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin);  
    }  
}  
/* USER CODE END 4 */
```

Press Button



Press Button

Timing



CPU call callback function immediately after receive interrupt signal

CPU จะเรียกใช้ Callback function ทันทีหลังจากได้รับสัญญาณ interrupt

Main function?

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    /* USER CODE END WHILE */

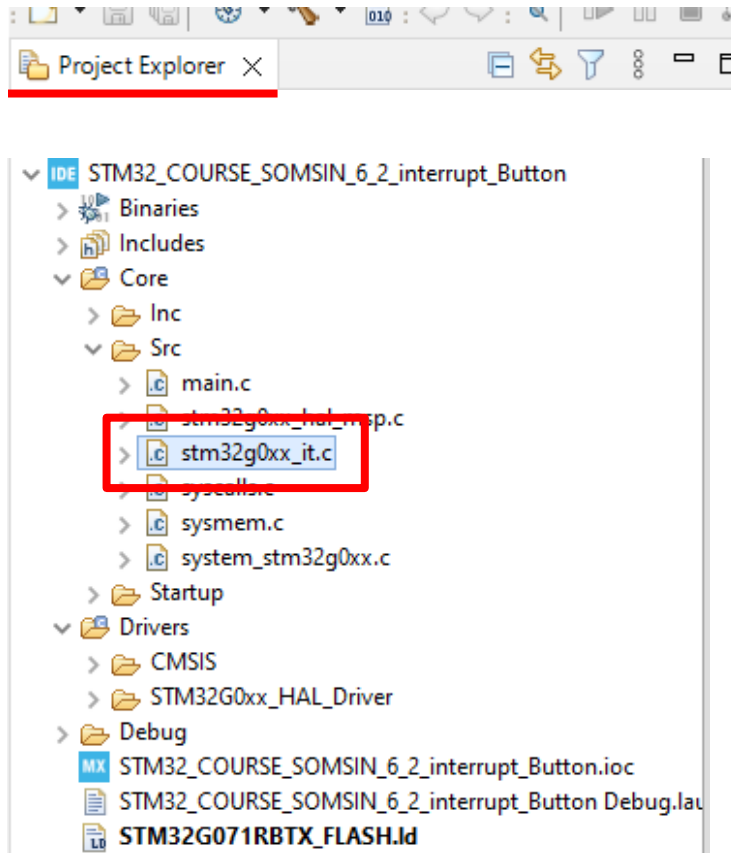
    /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```

Empty!

Yep. Nothing in main loop function
But it's running with ISR (callback
function)

ไม่มีอะไรอยู่ใน main loop

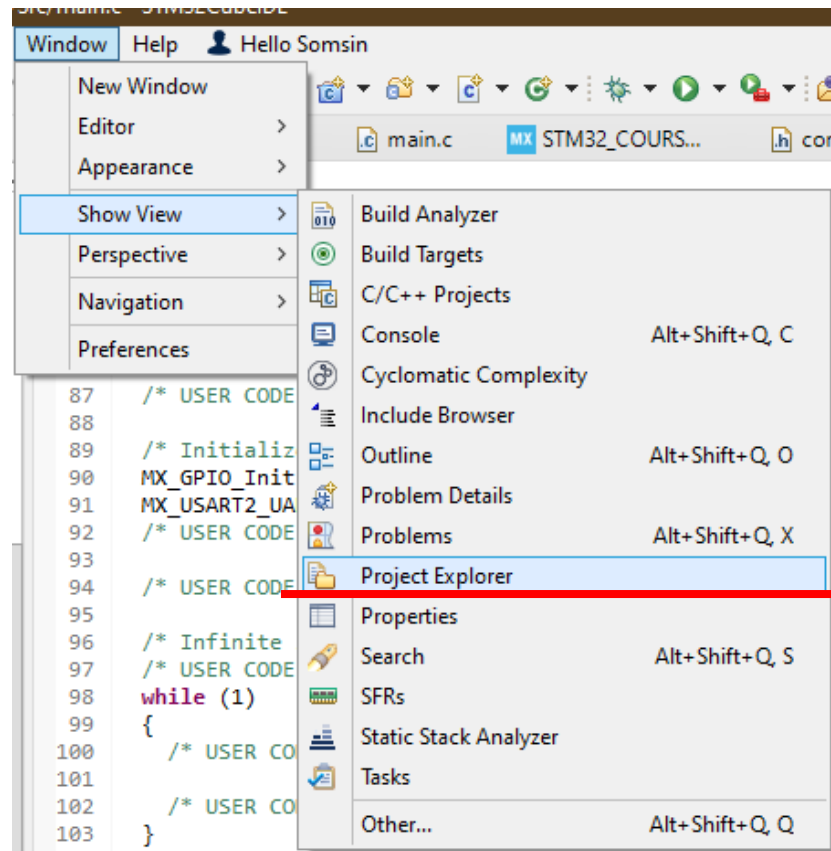
How to know callback function



```
143 /**
144  * @brief This function handles EXTI line 4 to 15 interrupts.
145  */
146 void EXTI4_15_IRQHandler(void)
147 {
148     /* USER CODE BEGIN EXTI4_15_IRQn 0 */
149
150     /* USER CODE END EXTI4_15_IRQn 0 */
151     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_13);
152     /* USER CODE BEGIN EXTI4_15_IRQn 1 */
153
154     /* USER CODE END EXTI4_15_IRQn 1 */
155 }
```

Handler is here

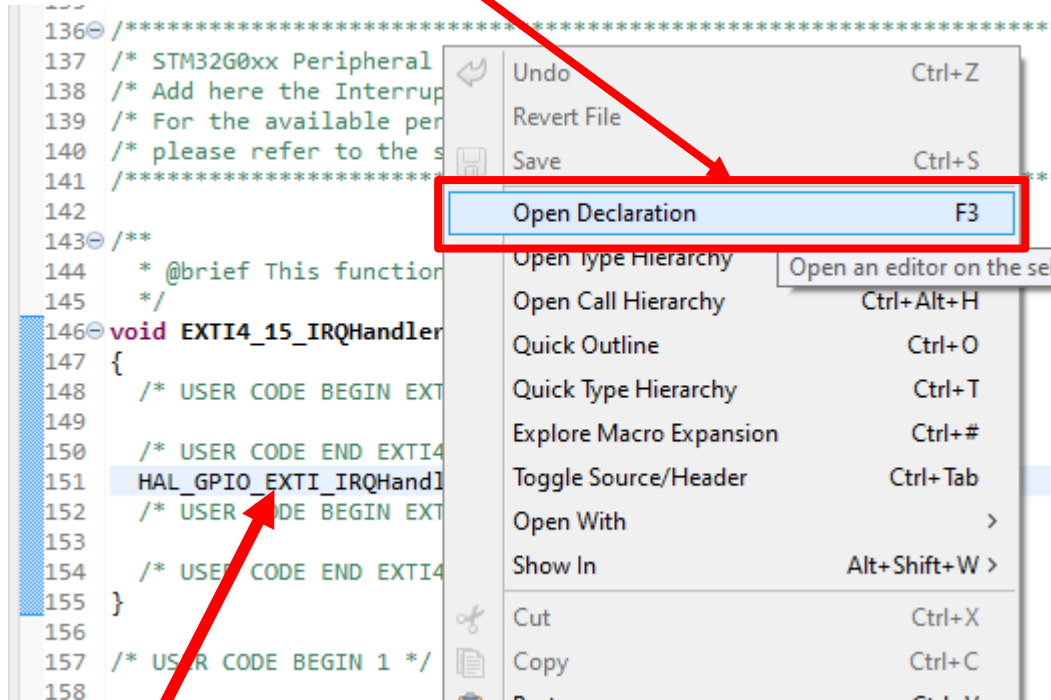
Note to open Project explorer



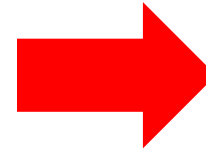
Project Explorer

How to know callback function

2. Select Open Declaration



1.Right Click function



```
487 void HAL_GPIO_EXTI_IRQHandler(uint16_t GPIO_Pin)
488 {
489     /* EXTI line interrupt detected */
490     if (__HAL_GPIO_EXTI_GET_RISING_IT(GPIO_Pin) != 0x00u)
491     {
492         __HAL_GPIO_EXTI_CLEAR_RISING_IT(GPIO_Pin);
493         HAL_GPIO_EXTI_Rising_Callback(GPIO_Pin);
494     }
495
496     if (__HAL_GPIO_EXTI_GET_FALLING_IT(GPIO_Pin) != 0x00u)
497     {
498         __HAL_GPIO_EXTI_CLEAR_FALLING_IT(GPIO_Pin);
499         HAL_GPIO_EXTI_Falling_Callback(GPIO_Pin);
500     }
501 }
502
503 /**
504  * @brief EXTI line detection callback.
505  * @param GPIO_Pin Specifies the port pin connected to corresponding EXTI line.
506  * @retval None
507  */
508 weak void HAL_GPIO_EXTI_Rising_Callback(uint16_t GPIO_Pin)
509 {
510     /* Prevent unused argument(s) compilation warning */
511     UNUSED(GPIO_Pin);
512
513     /* NOTE: This function should not be modified, when the callback is needed,
514      the HAL_GPIO_EXTI_Rising_Callback could be implemented in the user file
515     */
516 }
517
518 /**
519  * @brief EXTI line detection callback.
520  * @param GPIO_Pin Specifies the port pin connected to corresponding EXTI line.
521  * @retval None
522  */
523 weak void HAL_GPIO_EXTI_Falling_Callback(uint16_t GPIO_Pin)
524 {
525     /* Prevent unused argument(s) compilation warning */
526     UNUSED(GPIO_Pin);
527
528     /* NOTE: This function should not be modified, when the callback is needed,
529      the HAL_GPIO_EXTI_Falling_Callback could be implemented in the user file
530     */
531 }
```

All related callback is in here

Weak function

behaves like a normal function unless there is another non-weak same name function define in system. system will execute non-weak function

เหมือน function ทั่วไปทุกประการ แต่ ถ้ามี function ชื่อเดียวกันที่เป็น non-weak function ระบบจะทำการเรีย function ที่ non-weak

Weak function

```
21 ~ __weak print_name (){  
22 |     printf("somsin");  
23 | }  
24  
25 ~ print_name (){  
26 |     printf("thanavit");  
27 | }  
28  
29 ~ int main(){  
30 |     print_name();  
31 | }
```

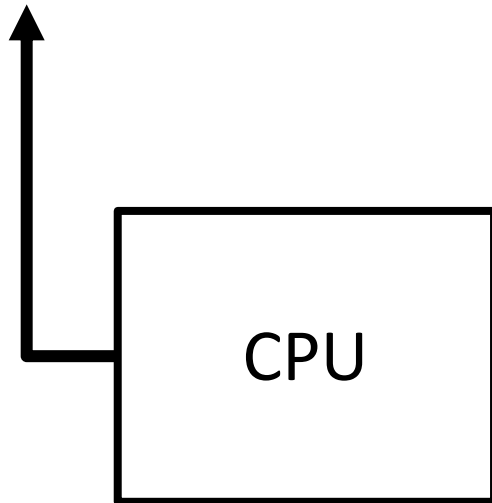
OUTPUT : thanavit

```
20  
21 print_name (){  
22 |     printf("somsin");  
23 | }  
24  
25 print_name (){  
26 |     printf("thanavit");  
27 | }  
28  
29 int main(){  
30 |     print_name();  
31 | }
```

OUTPUT : error duplicate declaration!

Coding

```
235 /* USER CODE BEGIN 4 */
236 void HAL_GPIO_EXTI_Falling_Callback(uint16_t GPIO_Pin)
237 {
238     if(GPIO_Pin == GPIO_PIN_13) {
239         HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin);
240     }
241 }
242 /* USER CODE END 4 */
```



```
523 __weak void HAL_GPIO_EXTI_Falling_Callback(uint16_t GPIO_Pin)
524 {
525     /* Prevent unused argument(s) compilation warning */
526     UNUSED(GPIO_Pin);
527
528     /* NOTE: This function should not be modified, when the callback is needed,
529             the HAL_GPIO_EXTI_Falling_Callback could be implemented in the user file
530     */
531 }
```

System will call out callback in main.c not weak function

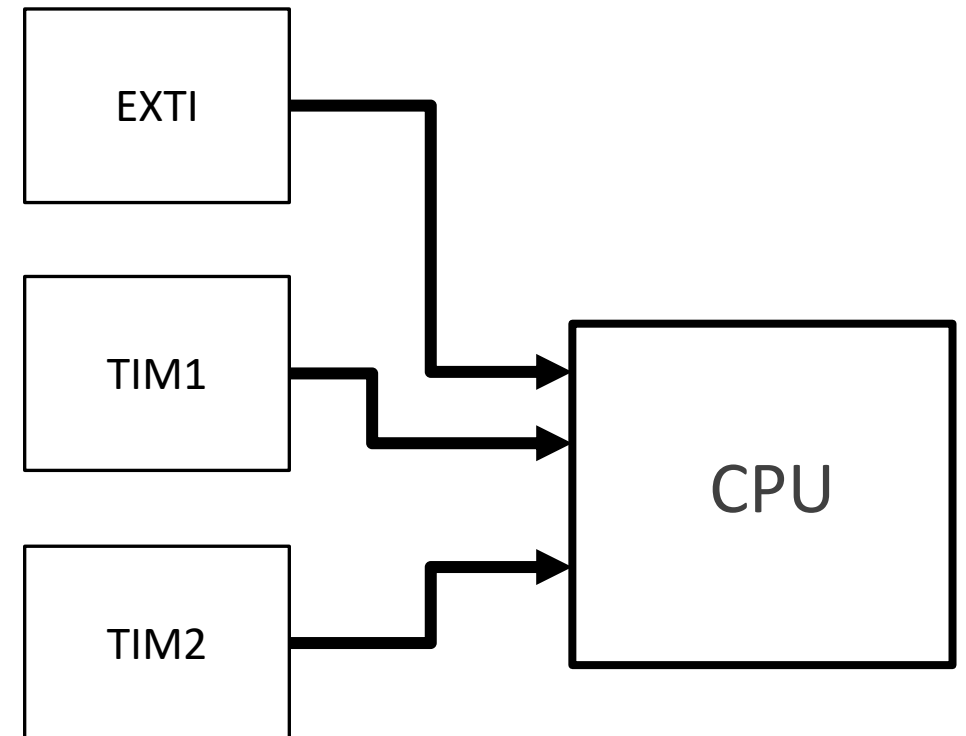
ระบบจะทำการเรียก callback function ใน main.c เท่านั้น และจะไม่เรียก weak function

Multiple interrupt

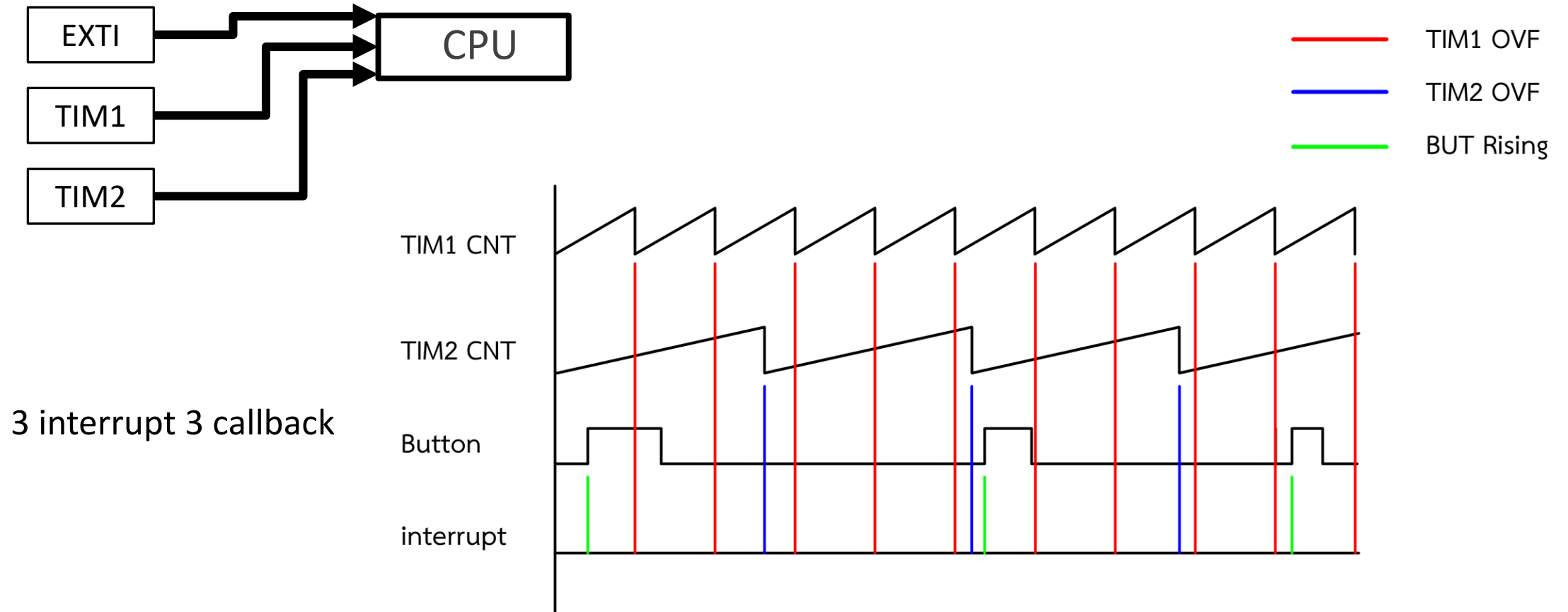
No problem!

CPU can handle multiple interrupt signal

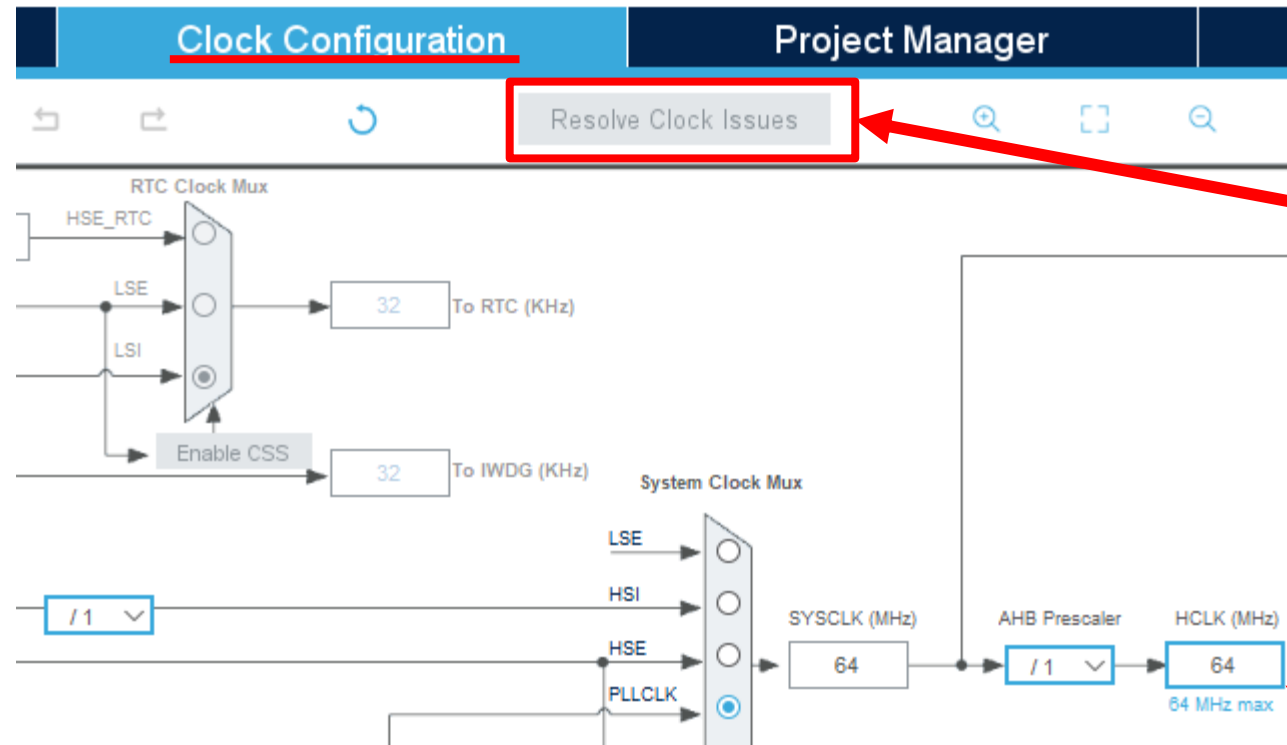
CPU สามารถรองรับได้มากกว่า 1 interrupt signal



LAB 3 timer interrupt



LAB 3 timer interrupt



2.Resolve Clock source

1.Set main Clock to 64 MHz

LAB 3 timer interrupt (TIM1)

1.Enable TIM 1 Clock source

2.Set Period to 30Hz

The screenshot shows the STM32CubeMX configuration window. On the left, under 'Timers', TIM1 is selected with a checkmark. In the center, the 'Mode' tab is active, showing 'Slave Mode' as 'Disable' and 'Trigger Source' as 'Disable'. The 'Clock Source' is set to 'Internal Clock'. Below this, the 'Configuration' tab is active, showing 'Reset Configuration' and 'Parameter Settings' (checked). The 'Counter Settings' are expanded, showing 'Prescaler (PSC - 16 bits va...)' as 640 - 1, 'Counter Mode' as 'Up', 'Counter Period (AutoReloa...)' as 3333 - 1, 'Internal Clock Division (CKD)' as 'No Division', 'Repetition Counter (RCR - ...)' as 0, and 'auto-reload preload' as 'Disable'.

TIM1 Mode and Configuration

3.Enable Interrupt

The screenshot shows the 'TIM1 Mode and Configuration' window. The 'Mode' tab is active, showing 'Slave Mode' as 'Disable' and 'Trigger Source' as 'Disable'. The 'Clock Source' is set to 'Internal Clock'. Below this, the 'Configuration' tab is active, showing 'Reset Configuration' and 'Parameter Settings' (checked). The 'NVIC Interrupt Table' is expanded, showing 'TIM1 break, update, trigger and commutation interrupts' with a checkmark in the 'Enabled' column.

LAB 3 timer interrupt (TIM2)

1.Enable TIM 2 Clock source

2.Set Period to 1Hz

3.Enable Interrupt

The image displays three screenshots from the STM32CubeMX IDE, illustrating the configuration steps for the TIM2 timer interrupt.

Screenshot 1: Timer Selection and Clock Source

- The 'Timers' list on the left shows TIM2 selected with a checkmark.
- The 'Clock Source' is set to 'Internal Clock'.

Screenshot 2: Counter Settings

- The 'Counter Settings' section shows the 'Counter Period (AutoReload)' set to 10000, which corresponds to a 1Hz frequency.

Screenshot 3: NVIC Interrupt Table

- The 'NVIC Interrupt Table' shows the 'TIM2 global interrupt' enabled (checked box).

LAB 3 timer interrupt (USART2)

The screenshot shows the STM32CubeMX configuration interface. On the left, the 'Connectivity' tree lists various peripherals, with 'USART2' selected and highlighted by a red box. On the right, the 'Reset Configuration' tab is active, and the 'NVIC Settings' sub-tab is selected. The 'NVIC Interrupt Table' is displayed with the following entries:

	Enabled	Preemption Priority
DMA1 channel 1 interrupt	<input type="checkbox"/>	0
USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26	<input checked="" type="checkbox"/>	0

The 'Enabled' checkbox for the USART2 interrupt is checked and highlighted by a red box. A red arrow points from the text '1.Enable Interrupt' below to this checkbox.

1.Enable Interrupt

LAB 3 timer Coding

```
99  /* USER CODE BEGIN 2 */
100 HAL_TIM_Base_Start_IT(&htim1);
101 HAL_TIM_Base_Start_IT(&htim2);
102 /* USER CODE END 2 */
103
104 /* Infinite loop */
105 /* USER CODE BEGIN WHILE */
106 while (1)
107 {
108     /* USER CODE END WHILE */
109
110     /* USER CODE BEGIN 3 */
111 }
112 /* USER CODE END 3 */
...
341 /* USER CODE BEGIN 4 */
342 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
343     if(htim == &htim1){
344         HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin);
345     }
346     if(htim == &htim2){
347         HAL_UART_Transmit(&huart2, "haruhi\r\n", 8,1000);
348     }
349 }
350
351 void HAL_GPIO_EXTI_Rising_Callback(uint16_t GPIO_Pin)
352 {
353     if(GPIO_Pin == GPIO_PIN_13) {
354         HAL_UART_Transmit(&huart2, "Somsin\r\n", 8,1000);
355     }
356 }
357 /* USER CODE END 4 */
```

```
/* USER CODE BEGIN 2 */
HAL_TIM_Base_Start_IT(&htim1);
HAL_TIM_Base_Start_IT(&htim2);
/* USER CODE END 2 */

/* USER CODE BEGIN 4 */
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
    if(htim == &htim1){
        HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin);
    }
    if(htim == &htim2){
        HAL_UART_Transmit(&huart2, "haruhi\r\n", 8,1000);
    }
}

void HAL_GPIO_EXTI_Rising_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin == GPIO_PIN_13) {
        HAL_UART_Transmit(&huart2, "Somsin\r\n", 8,1000);
    }
}
/* USER CODE END 4 */
```

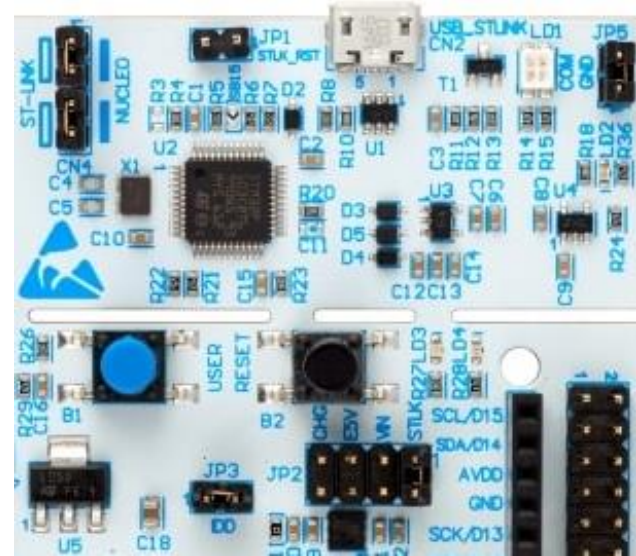
LAB 3 timer

Feature

- Blink 15 Hz
- Print “Haruhi” 1Hz
- Print “Somsin” when Push button

3 task at a time

ทำงาน 3 อย่างในเวลาเดียวกัน



Received/Sent data

```
haruhi  
Somsin  
haruhi  
Somsin  
Somsin  
Somsin  
haruhi  
Somsin  
haruhi  
haruhi
```

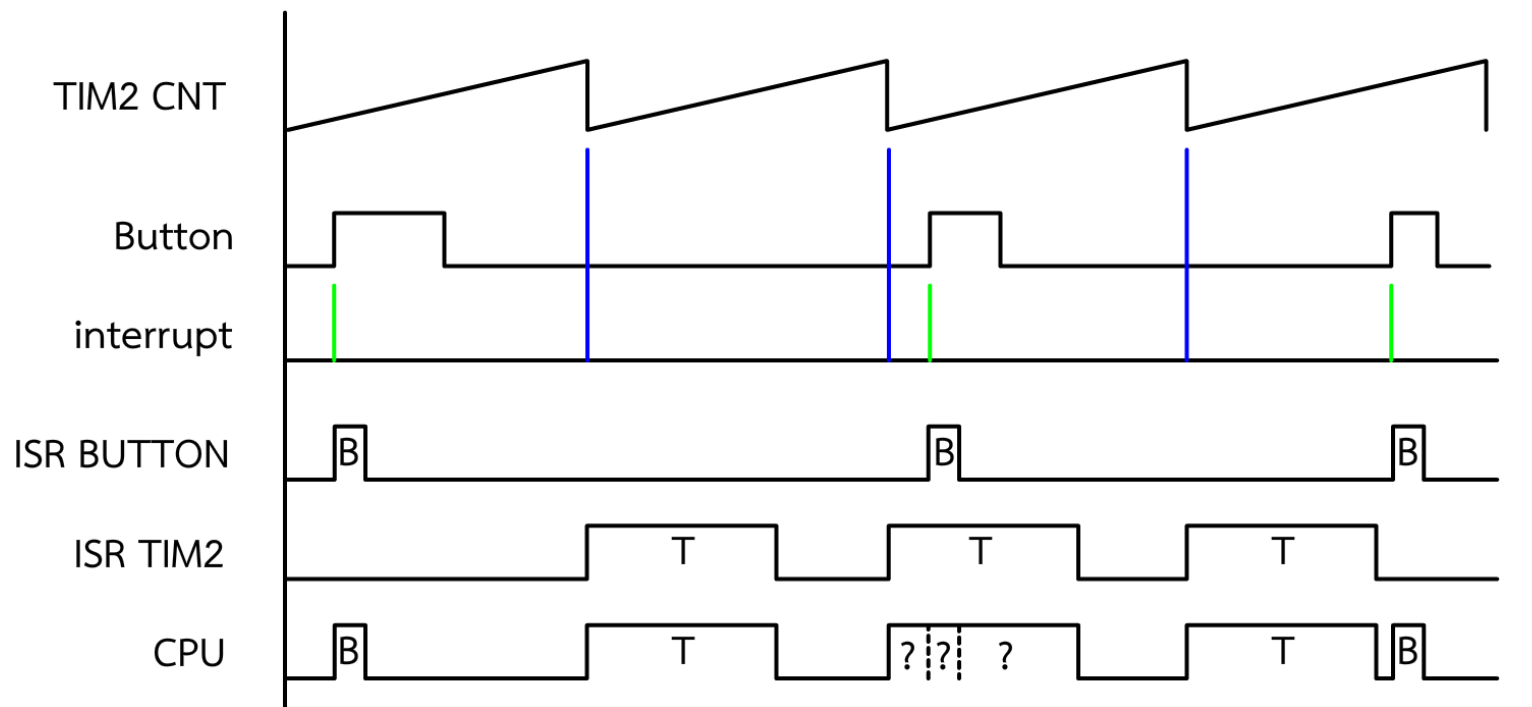

Priority

ISR TIM2 -> 2000 ms

ISR BUTTON -> 20 ms

TIM2 OVF

BUT Rising

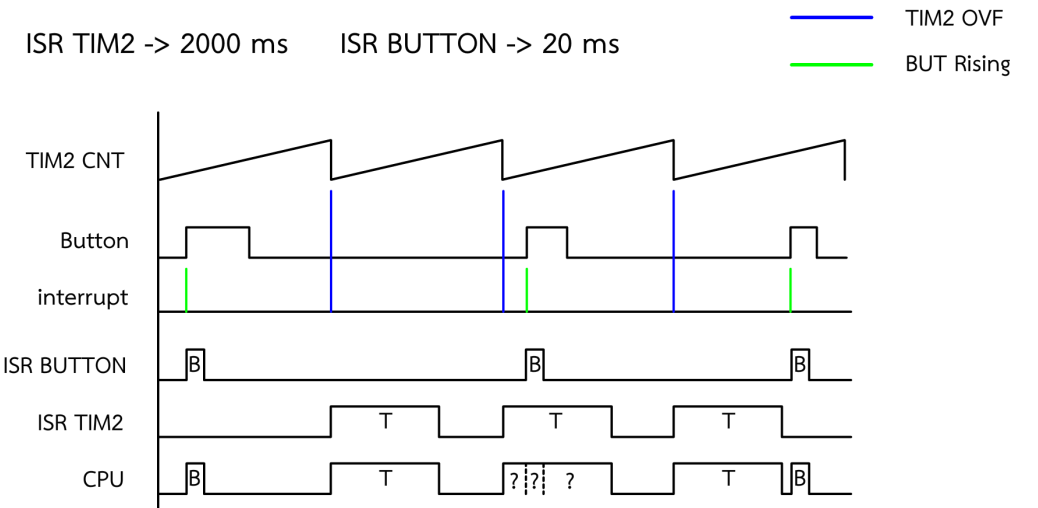
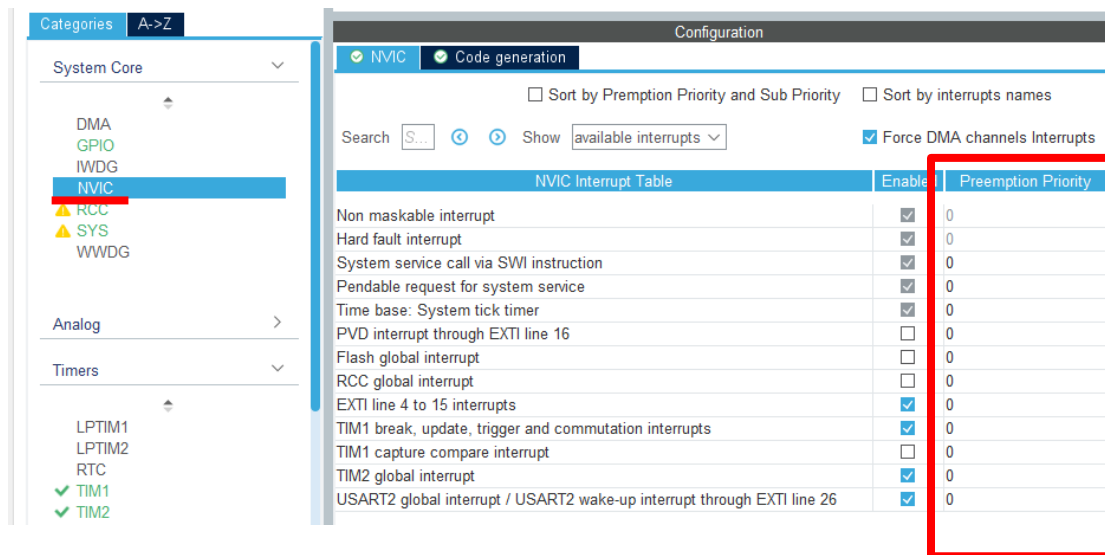


What happen if 2 interrupt occurs at the same time?

เกิดอะไรขึ้นถ้าเกิด interrupt ขึ้น
พร้อมกัน 2 อัน

Priority

CPU 1 ตัวสามารถทำงานได้เพียงแค่ 1 อย่าง เท่านั้น
ดังนั้นจึงมีสิ่งที่เรียกว่า interrupt Priority เกิดขึ้น

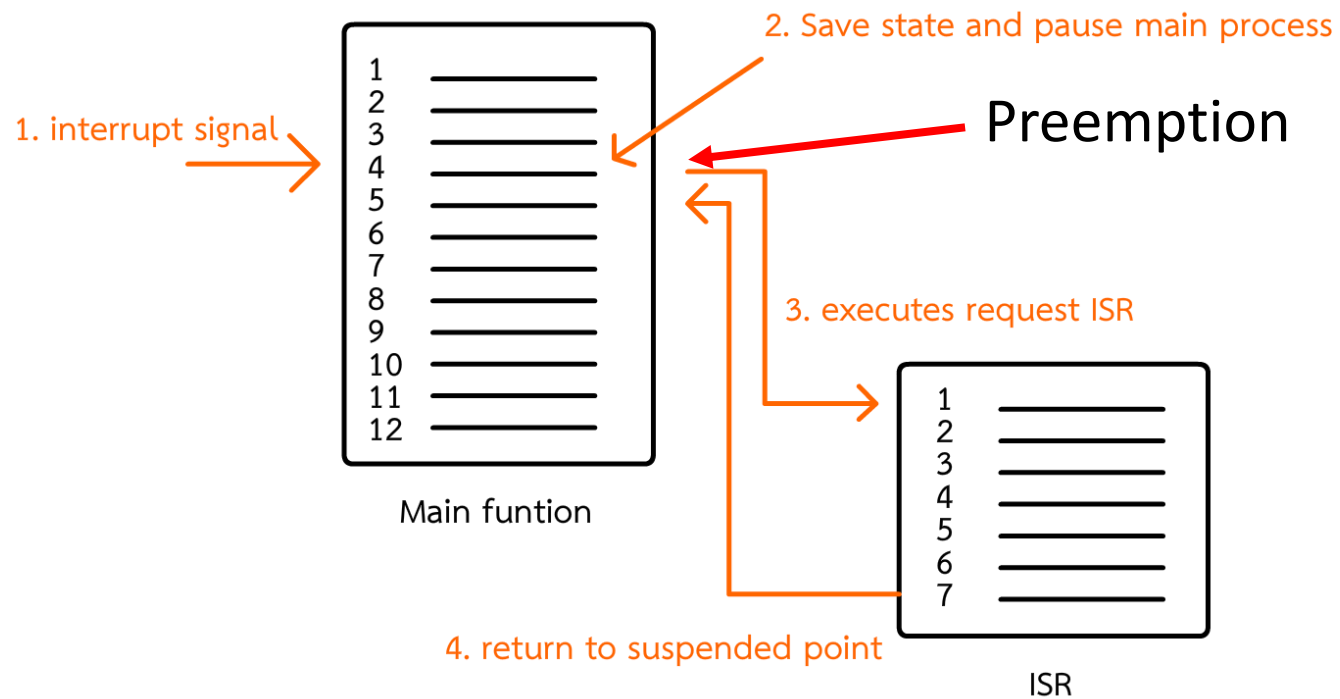


Priority

คือ ตัวบ่งชี้ความสำคัญของ ISR หรือ section สามารถบอกลำดับการทำงานของ interrupt ได้ (ยิ่งตัวเลขน้อยยิ่งสำคัญมากกว่า หรือ สูงกว่า)

Priority of ISR or code section indicate order for interrupt. (Lower number mean higher priority)

Preemption คือ การออกจาก task หรือ งานเดิม เพื่อไปทำงานใน task ใหม่ หรือ section ที่กำหนดไว้



Preempt (V.)

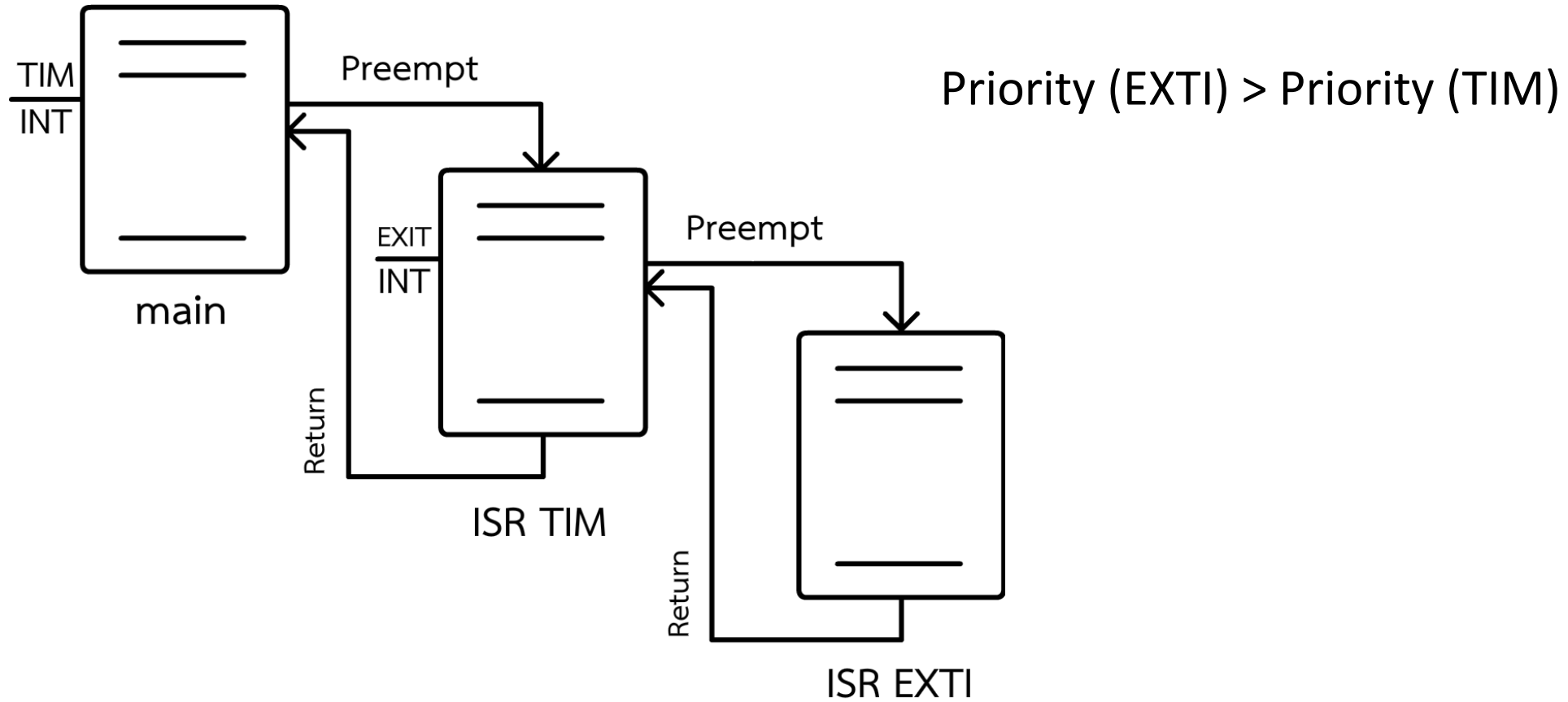
Preemption (N.)

Priority

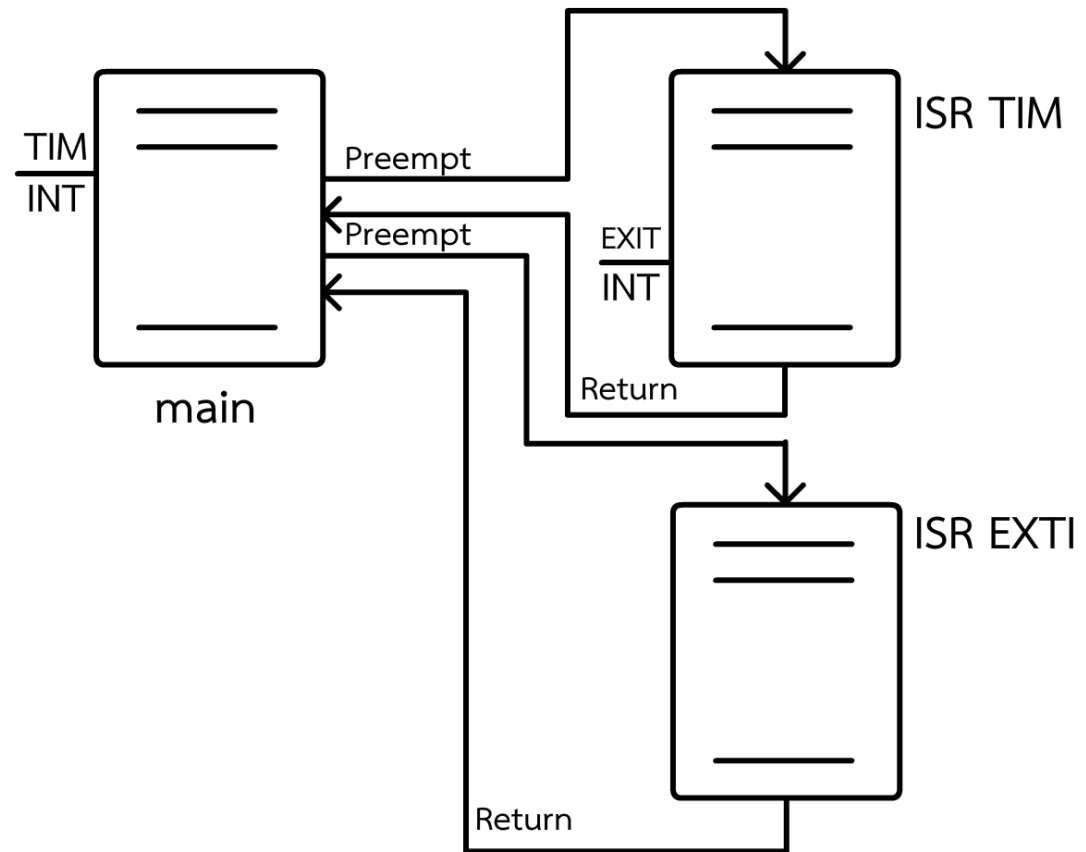
If current section has lower priority than incoming interrupt, the incoming will preempt otherwise it will preempt once when incoming ISR finished.

หาก section ปัจจุบัน มี priority ที่ต่ำกว่า priority ของ interrupt ที่เข้ามา interrupt ที่เข้ามา จะได้สิทธิ์ในการทำงาน แต่หาก section ปัจจุบัน มี priority ที่สูงกว่า ระบบจะรอให้ ISR จบก่อนจึงให้สิทธิ์ทำงาน 1 ครั้ง

Priority

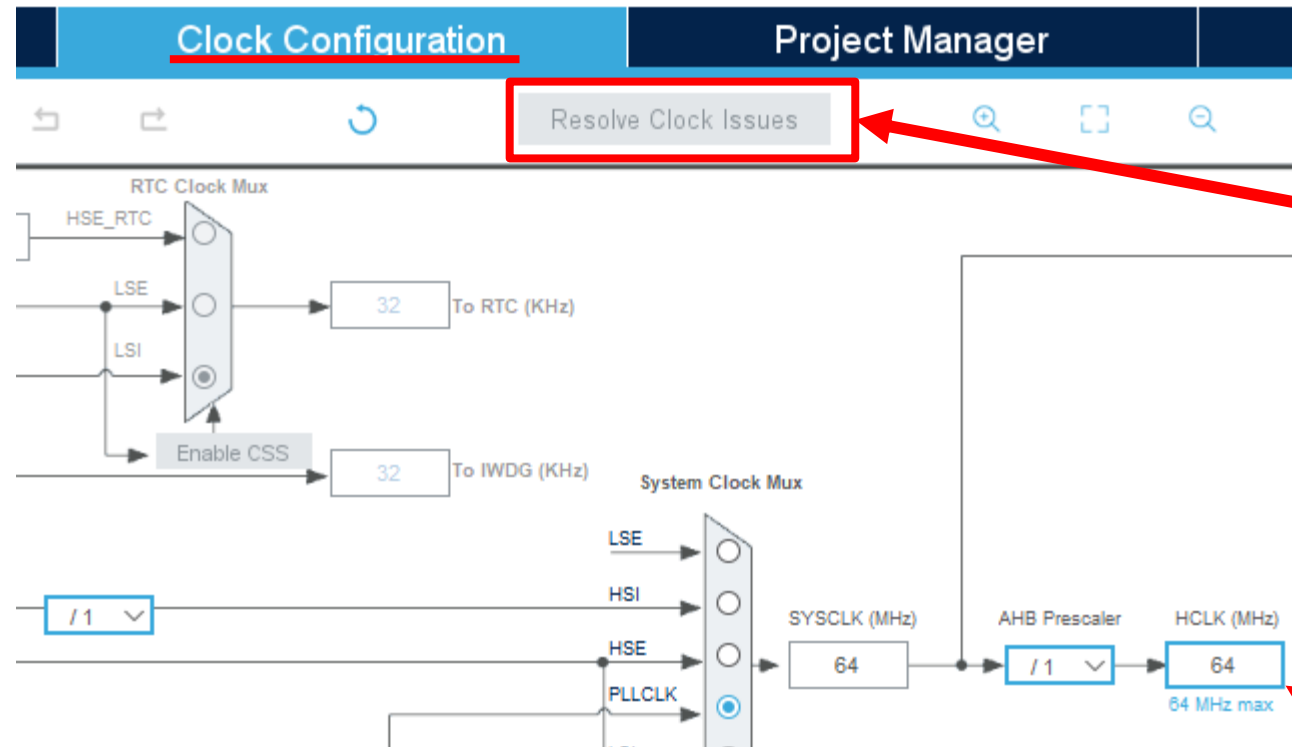


Priority



Priority (EXTI) < Priority (TIM)

LAB 4 Priority



2.Resolve Clock source

1.Set main Clock to 64 MHz

LAB 4 Priority TIM config

The image shows the STM32CubeMX interface with three red annotations and arrows pointing to specific configuration steps:

- 1.Enable TIM 2 Clock source**: Points to the 'Internal Clock' option in the 'Clock Source' field of the 'TIM2 Mode and Configuration' window.
- 2.Set Period to 0.1Hz**: Points to the 'Counter Settings' table in the 'TIM2 Mode and Configuration' window.
- 3.Enable Interrupt**: Points to the 'TIM2 global interrupt' row in the 'NVIC Interrupt Table'.

Left Panel (Pinout & Configuration):

- Categories: A->Z
- System Core >
- Analog >
- Timers >
- LPTIM1
- LPTIM2
- RTC
- TIM1**
- TIM2** (checked)
- TIM3
- TIM6
- TIM7
- TIM14
- TIM15
- TIM16
- TIM17

Top Panel (Clock Configuration):

- Software Packs
- Pinout
- TIM2 Mode and Configuration
- Mode
- Slave Mode: Disable
- Trigger Source: Disable
- Clock Source: **Internal Clock**
- Configuration
- Reset Configuration
- Parameter Settings (checked)
- User Constants (checked)
- NVIC Settings (checked)
- DMA Settings (checked)

Right Panel (TIM2 Mode and Configuration):

- Mode
- Slave Mode: Disable
- Trigger Source: Disable
- Clock Source: Internal Clock
- Configuration
- Reset Configuration
- Parameter Settings (checked)
- User Constants (checked)
- NVIC Settings (checked)**
- DMA Settings (checked)

Bottom Panel (NVIC Interrupt Table):

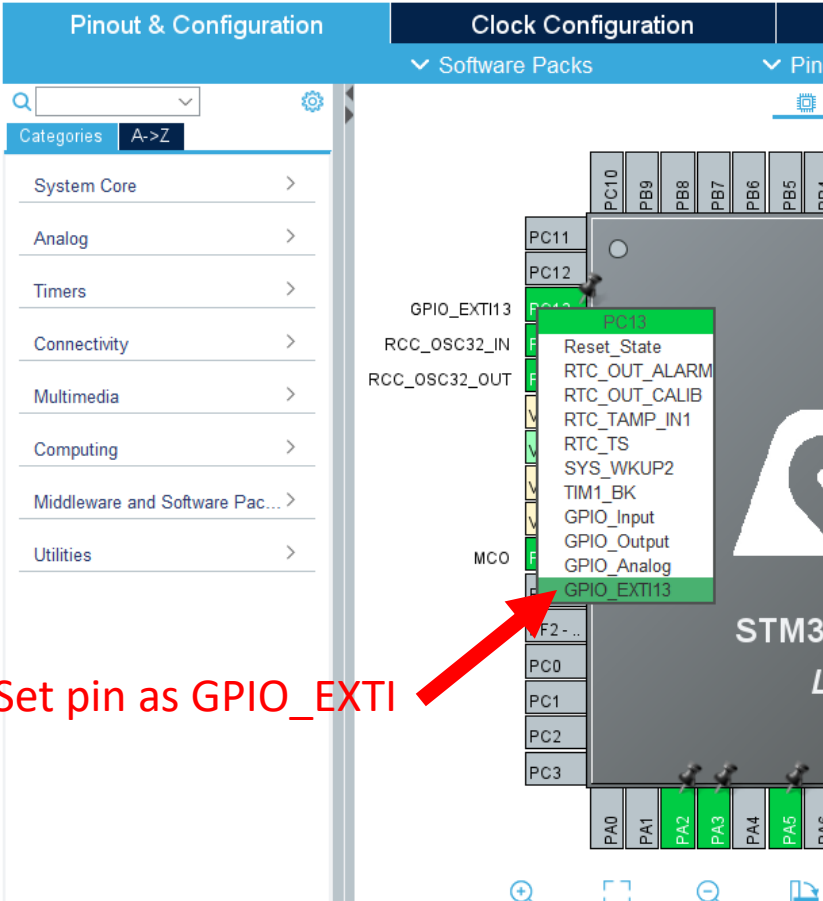
Interrupt	Enabled	Preemption Priority
TIM2 global interrupt	<input checked="" type="checkbox"/>	3

Counter Settings Table:

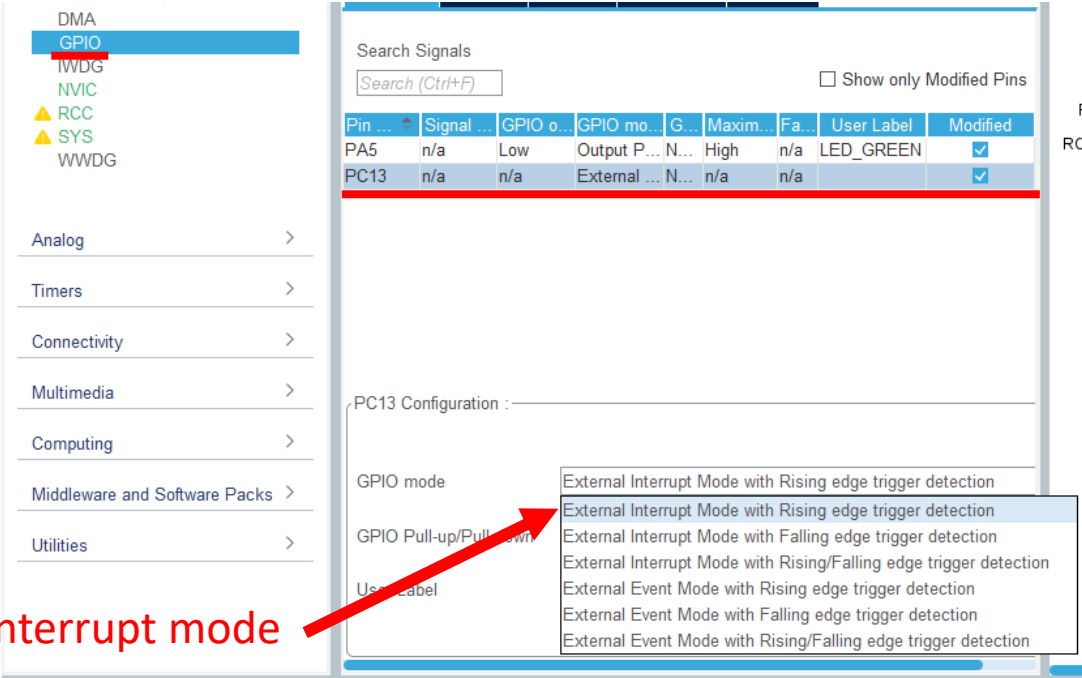
Parameter	Value
Prescaler (PSC - 16 bits value)	6400 - 1
Counter Mode	Up
Counter Period (AutoReload Register)	100000 - 1
Internal Clock Division (CKD)	No Division
auto-reload preload	Disable

LAB 4 Priority GPIO Config


1. Set pin as GPIO_EXTI



2. Set set interrupt mode



3. Enable External interrupt



Pin	Signal	GPIO o...	GPIO mo...	G...	Maxim...	Fa...	User Label	Modified
PA5	n/a	Low	Output P...	N...	High	n/a	LED_GREEN	<input checked="" type="checkbox"/>
PC13	n/a	n/a	External ...	N...	n/a	n/a		<input checked="" type="checkbox"/>

GPIO mode
External Interrupt Mode with Rising edge trigger detection
External Interrupt Mode with Rising edge trigger detection
External Interrupt Mode with Falling edge trigger detection
External Interrupt Mode with Rising/Falling edge trigger detection
External Event Mode with Rising edge trigger detection
External Event Mode with Falling edge trigger detection
External Event Mode with Rising/Falling edge trigger detection

GPIO	RCC	SYS	USART	NVIC
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

NVIC Interrupt Table	Enabled	Preemption Priority
EXTI line 4 to 15 interrupts	<input checked="" type="checkbox"/>	0

LAB 4 USART2 Config

The screenshot displays the STM32CubeMX configuration interface. On the left, the 'Connectivity' tree shows various peripherals, with 'USART2' selected and highlighted by a red box. The main panel shows the 'Reset Configuration' window with the 'NVIC Settings' tab active. The 'NVIC Interrupt Table' is visible, listing two interrupts: 'DMA1 channel 1 interrupt' and 'USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26'. The 'Enabled' column for the USART2 interrupt is checked, indicated by a blue checkmark and a red box. A red arrow points to this checkmark with the text '1.Enable Interrupt'.

NVIC Interrupt Table		Enabled	Preemption Priority
DMA1 channel 1 interrupt		<input type="checkbox"/>	0
USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26		<input checked="" type="checkbox"/>	0

1.Enable Interrupt

LAB 4 Priority Config

Pinout & Configuration | Clock Configuration | Project Manager

Software Packs | Pinout

NVIC Mode and Configuration

Configuration

✓ NVIC | ✓ Code generation

☐ Sort by Preemption Priority and Sub Priority ☐ Sort by interrupts names

Search Show ☒ Force DMA channels Interrupts

NVIC Interrupt Table	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"/>	2
TIM2 global interrupt	<input checked="" type="checkbox"/>	3
USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26	<input checked="" type="checkbox"/>	0

1.Set EXTI and TIM2 interrupt priority

*TIM2 has priority higher than EXTI

TIM2 มี priority สูงกว่า EXTI

*system tick มี priority เป็น 0

*main function มี priority ต่ำที่สุด
(สามารถถูก preempt ได้จากทุกคน)

Coding

```
95 HAL_TIM_Base_Init(&htim2);
96 /* USER CODE BEGIN 2 */
97 HAL_TIM_Base_Start_IT(&htim2);
98 /* USER CODE END 2 */
99
100 /* Infinite loop */
101 /* USER CODE BEGIN WHILE */
102 while (1)
103 {
104     /* USER CODE END WHILE */
105
106     /* USER CODE BEGIN 4 */
107 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
108     unsigned char out_string[20] = "haruhi1\r\n";
109     if(htim == &htim2){
110         for(int i=0;i<5;i++){
111             HAL_UART_Transmit(&huart2, out_string, 9,1000);
112             out_string[6]++;
113             HAL_Delay(1000);
114         }
115     }
116 }
117
118 void HAL_GPIO_EXTI_Rising_Callback(uint16_t GPIO_Pin)
119 {
120     if(GPIO_Pin == GPIO_PIN_13) {
121         HAL_UART_Transmit(&huart2, "Somsin\r\n", 8,1000);
122     }
123 }
124 /* USER CODE END 4 */
```

```
/* USER CODE BEGIN 2 */
HAL_TIM_Base_Start_IT(&htim2);
/* USER CODE END 2 */

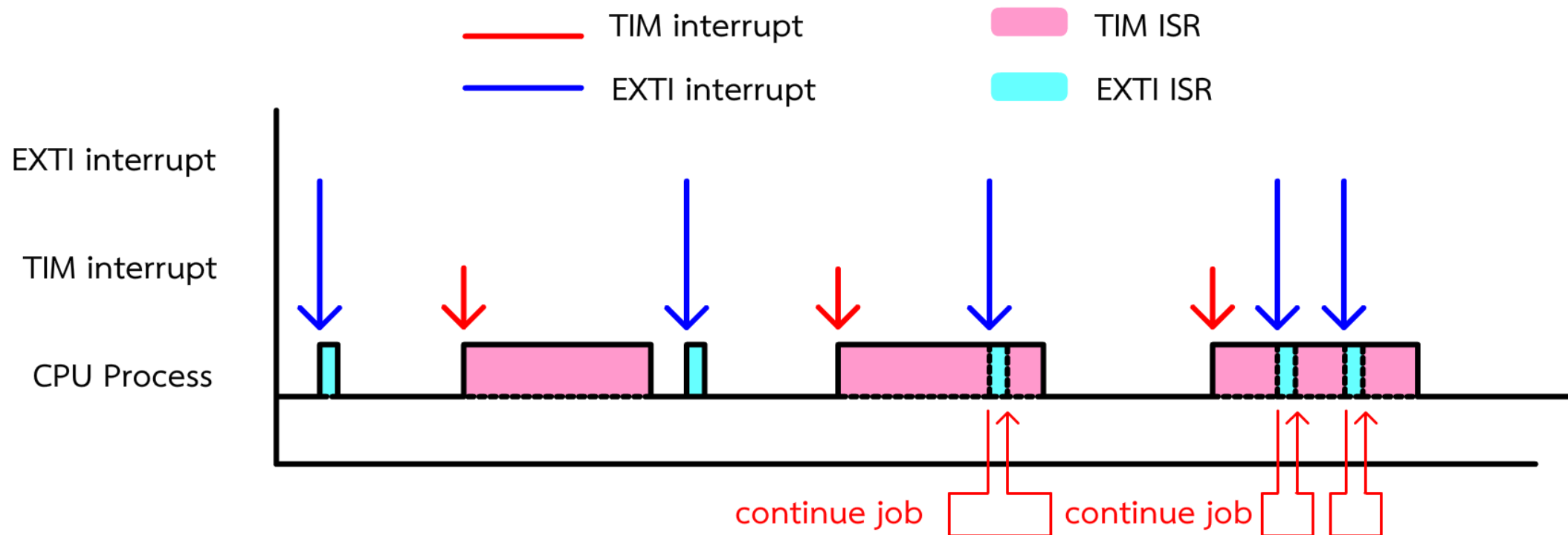
/* USER CODE BEGIN 4 */
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
    unsigned char out_string[20] = "haruhi1\r\n";
    if(htim == &htim2){
        for(int i=0;i<5;i++){
            HAL_UART_Transmit(&huart2, out_string, 9,1000);
            out_string[6]++;
            HAL_Delay(1000);
        }
    }
}

void HAL_GPIO_EXTI_Rising_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin == GPIO_PIN_13) {
        HAL_UART_Transmit(&huart2, "Somsin\r\n", 8,1000);
    }
}
/* USER CODE END 4 */
```

LAB 4

TIM ISR -> interrupt every 10 sec do 5 sec job (print Haruhi1 - 5)

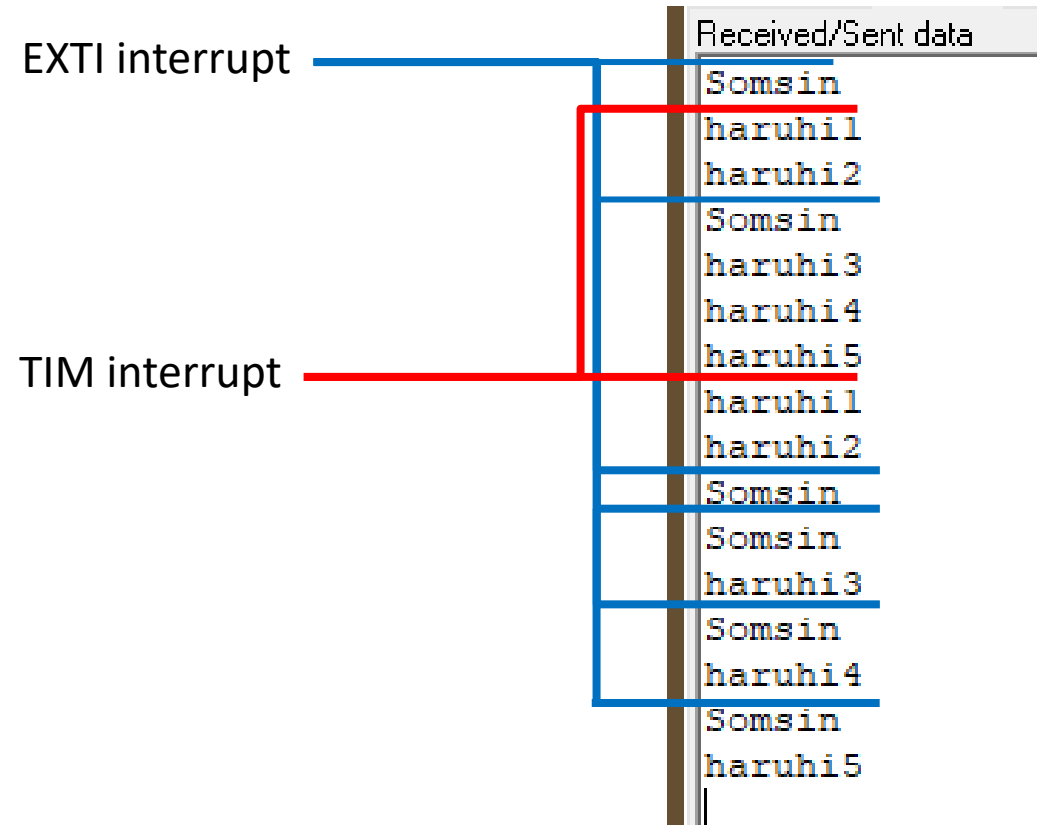
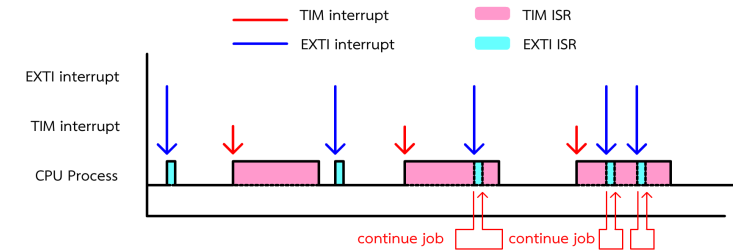
EXTI ISR -> print Somsin, interrupt every time button release.



LAB 4

TIM ISR -> interrupt every 10 sec
do 5 sec job (print Haruhi1 - 5)

EXTI ISR -> print Somsin, interrupt every time button release.



LAB 4 Priority Config

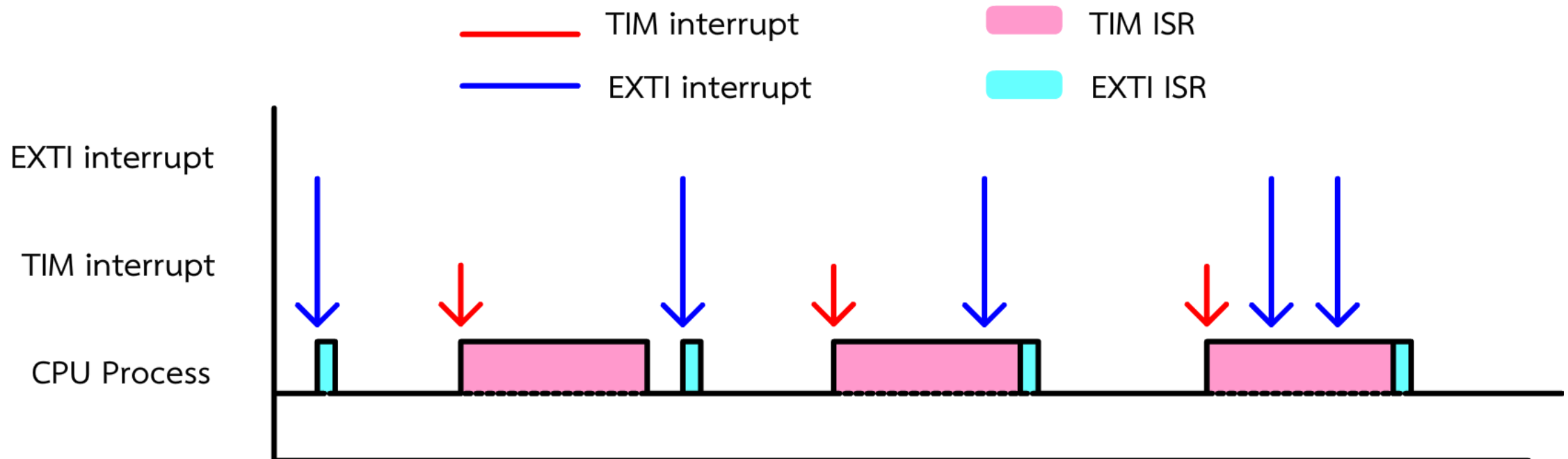
The screenshot shows the STM32CubeIDE interface with the 'NVIC Mode and Configuration' window open. The 'NVIC' tab is selected, and the 'NVIC Interrupt Table' is displayed. A red box highlights the 'EXTI line 4 to 15 interrupts' and 'TIM2 global interrupt' rows, with an arrow pointing to the 'TIM2 global interrupt' row.

NVIC Interrupt Table	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"/>	3
TIM2 global interrupt	<input checked="" type="checkbox"/>	2
USART12 global interrupt / USART12 wake-up interrupt through EXTI line 20	<input checked="" type="checkbox"/>	0

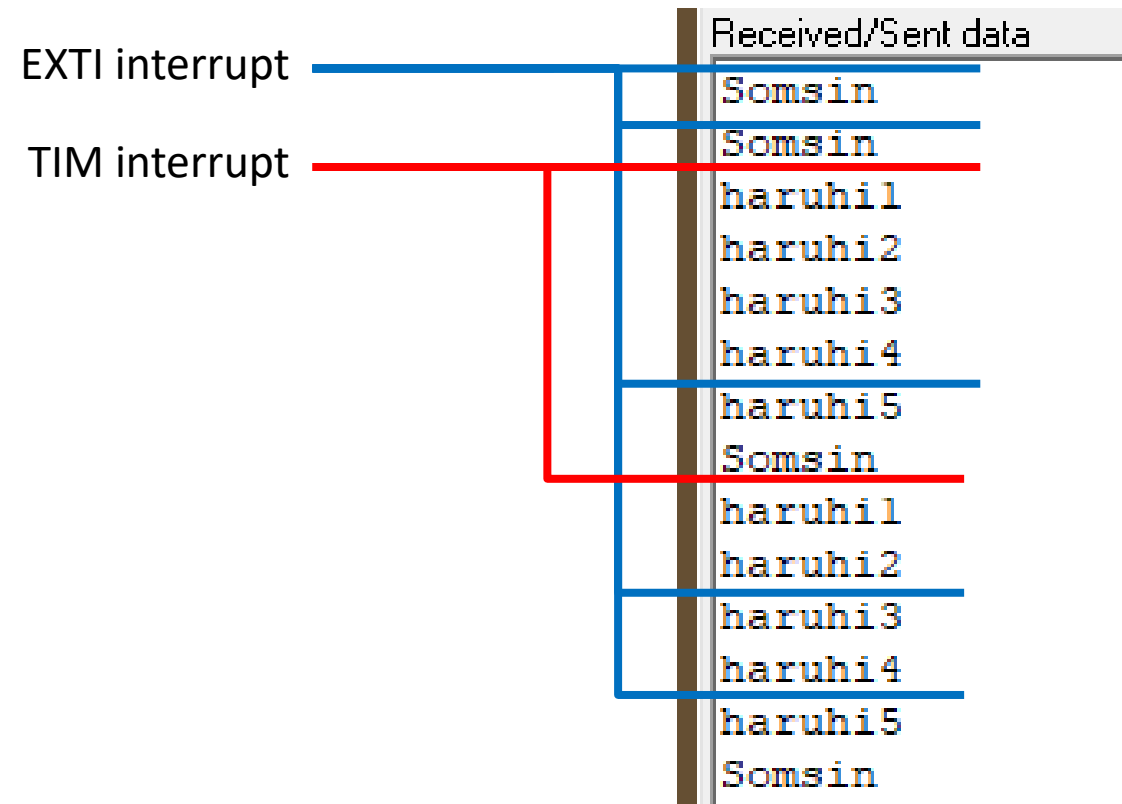
Try swap priority

*TIM2 has priority lower than EXTI
TIM2 มี priority ต่ำกว่า EXTI

LAB 4



LAB 4



Shortest ISR

เพื่อเลี่ยงปัญหา interrupt collision เราควรให้ code ที่อยู่ในส่วนของ ISR สั้นที่สุดเท่าที่จะเป็นไปได้ (ห้ามมี delay)

To avoid interrupt collision ISR should be as fast as possible code (mustn't contain delay)

Further problem

- race condition (using same register across task)
- multiple variable access (semaphore and mutex)
- scheduler
- etc.



OS!

LAB

สร้างโปรแกรมไฟกระพริบโดย

- หากมีการกดปุ่ม 1 ครั้ง ให้ไฟกระพริบทั้งหมด 3 ครั้ง
- การกระพริบ 1 ครั้ง จะต้องใช้เวลา 2 วินาที (ติด 1 วินาที ดับ 1 วินาที)
- การกดปุ่มสามารถ stack ได้ (หากกดปุ่มในขณะที่ไฟยังกระพริบไม่ครบ 3 ครั้ง ให้เพิ่มการกระพริบเข้าไป 3 อีกครั้ง) (กดปุ่ม 2 ครั้งจะต้องเห็นไฟกระพริบ 6 ครั้ง ไม่ว่าจะกดตอนไหน , กดปุ่ม 9 ครั้งจะต้องเห็นไฟกระพริบ 9 ครั้ง ไม่ว่าจะกดตอนไหน)