

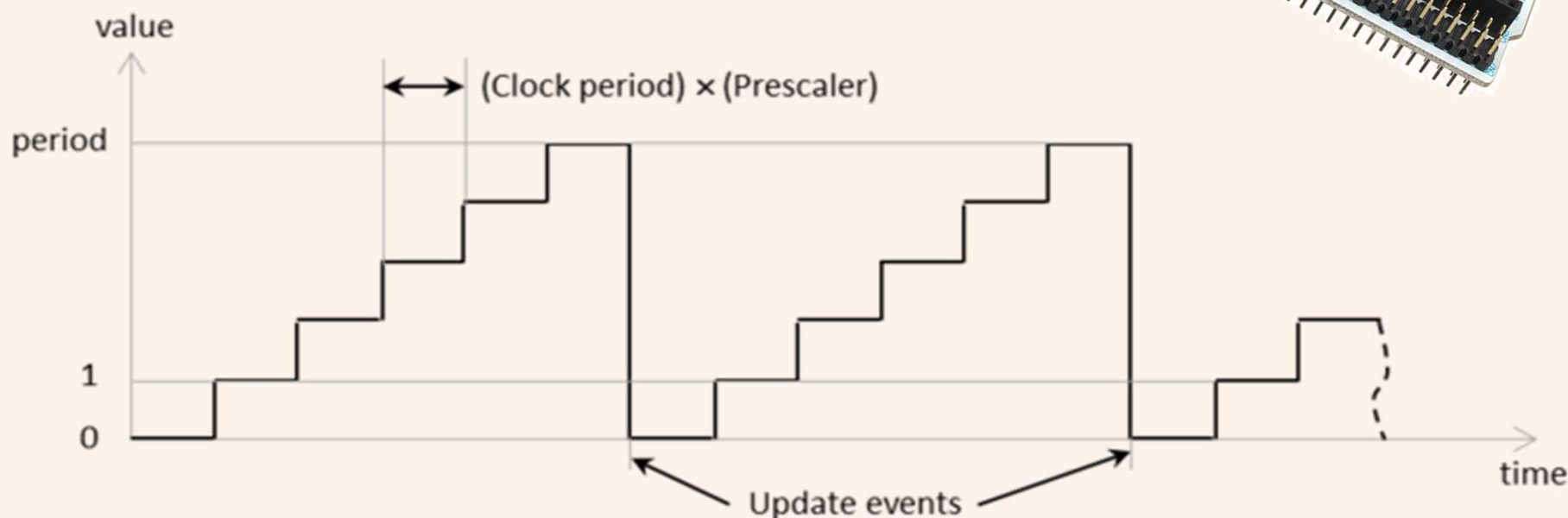
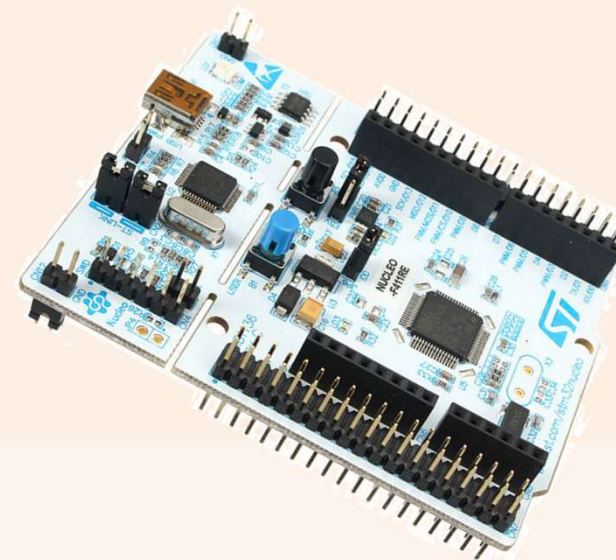
# Competências Transferíveis

## Microcontroladores e Interação com Sensores e Atuadores

V1\_2

Rui Escadas Martins

### Timers and Interrupts



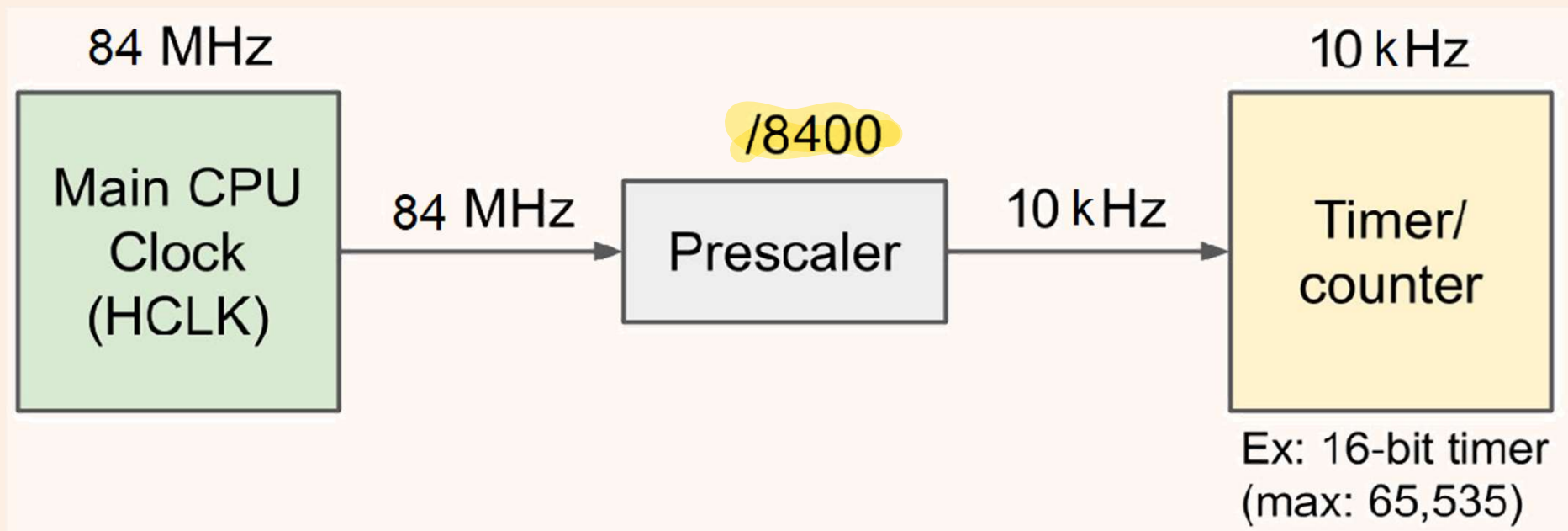
# Timers

O que são:

São periféricos que manipulam contagem de eventos e/ou tempo com muita precisão.

Tipicamente, a referência é derivada do clock interno, mas pode originar de um sinal ligado a um determinado pino (externa) ou a outro oscilador (interno).

Como a frequência de relógio dos microcontroladores é muito elevada é normal dividir-se por um valor inteiro designado por "prescaler"!

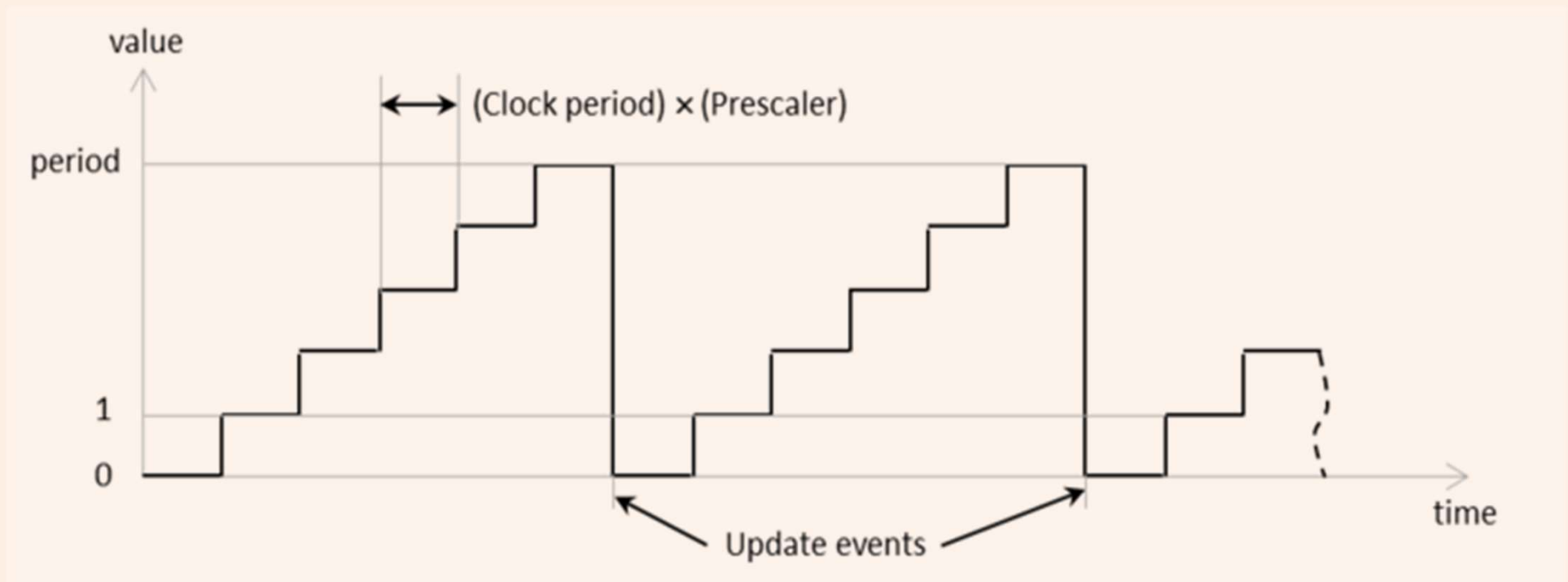


# Timers

O que são:

São periféricos que manipulam contagem de eventos e/ou tempo com muita precisão.

Tipicamente, a referência é derivada do clock interno, mas pode originar de um sinal ligado a um determinado pino (externo) ou a outro oscilador.



# Timers

Configuração:

The screenshot displays the STM32CubeMX Pinout & Configuration window. The left sidebar shows the project tree with 'Timers' expanded and 'TIM2' selected. The main area is titled 'TIM2 Mode and Configuration' and is divided into two sections: 'Mode' and 'Configuration'.

**Mode Section:**

- Slave Mode: Disable
- Trigger Source: Disable
- Clock Source: Internal Clock
- Channel1: Disable
- Channel2: Disable
- Channel3: Disable
- Channel4: Disable
- Combined Channels: Disable
- ☐ Use ETR as Clearing Source
- ☐ XOR activation
- ☐ One Pulse Mode

**Configuration Section:**

- Reset Configuration
- User Constants (checked)
- NVIC Settings (checked)
- DMA Settings (checked)
- Parameter Settings (checked)

Configure the below parameters :

Search (Ctrl+F)

**Counter Settings**

- Prescaler (PSC - 16 bits value): 8400-1
- Counter Mode: Up
- Counter Period (AutoReload value): 100000-1
- Internal Clock Division (CKD): No Division
- auto-reload preload: Enable

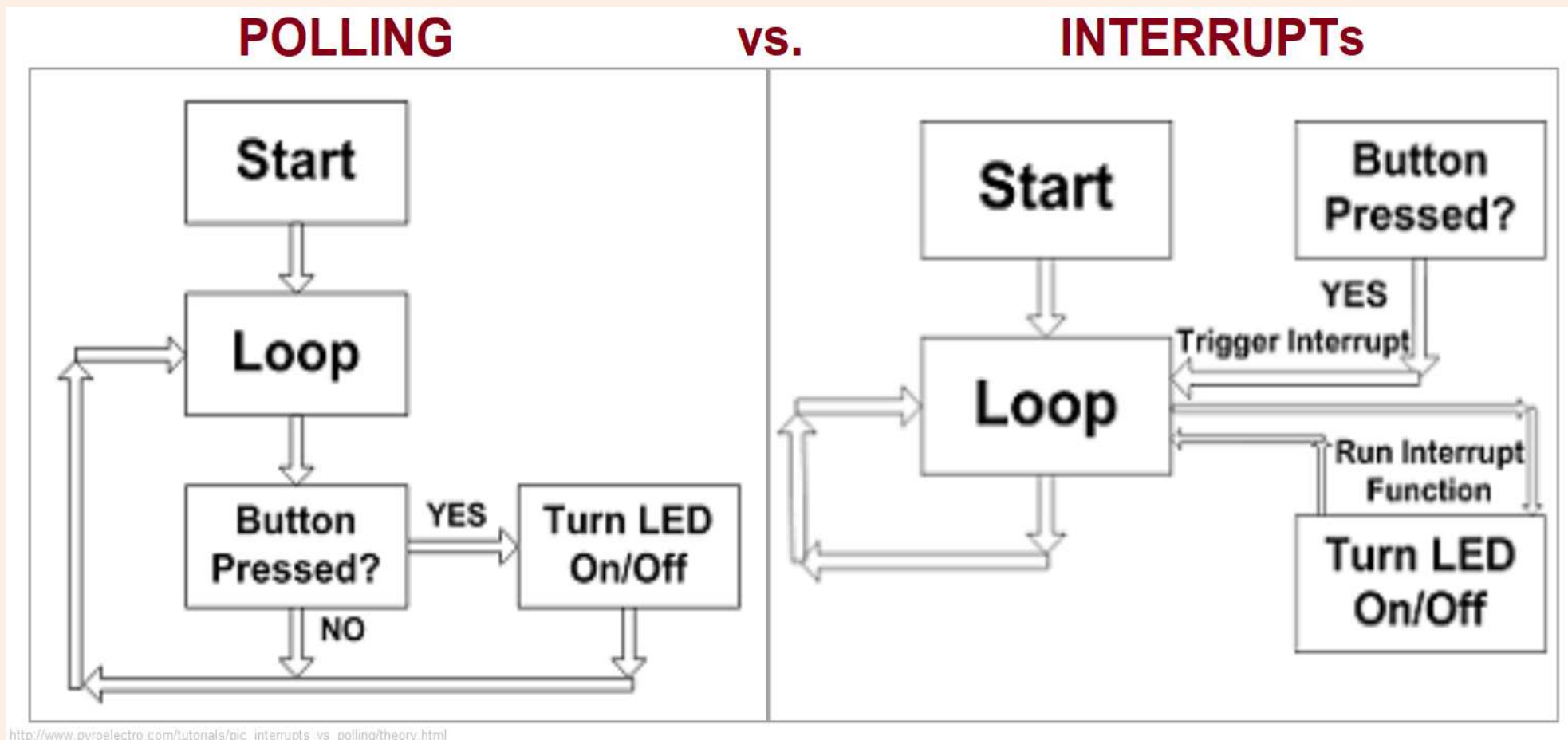
**Trigger Output (TRGO) Parameters**

- Master/Slave Mode (MSM bit): Disable (Trigger input effect not delayed)
- Trigger Event Selection: Reset (UG bit from TIMx\_EGR)

# Interrupts

O que são:

São sinais causados por determinados eventos que **interrompem o normal funcionamento de um programa e forçam a execução de uma rotina especial associada ao referido evento.** São tipicamente mais eficientes do que polling!



# Interrupts

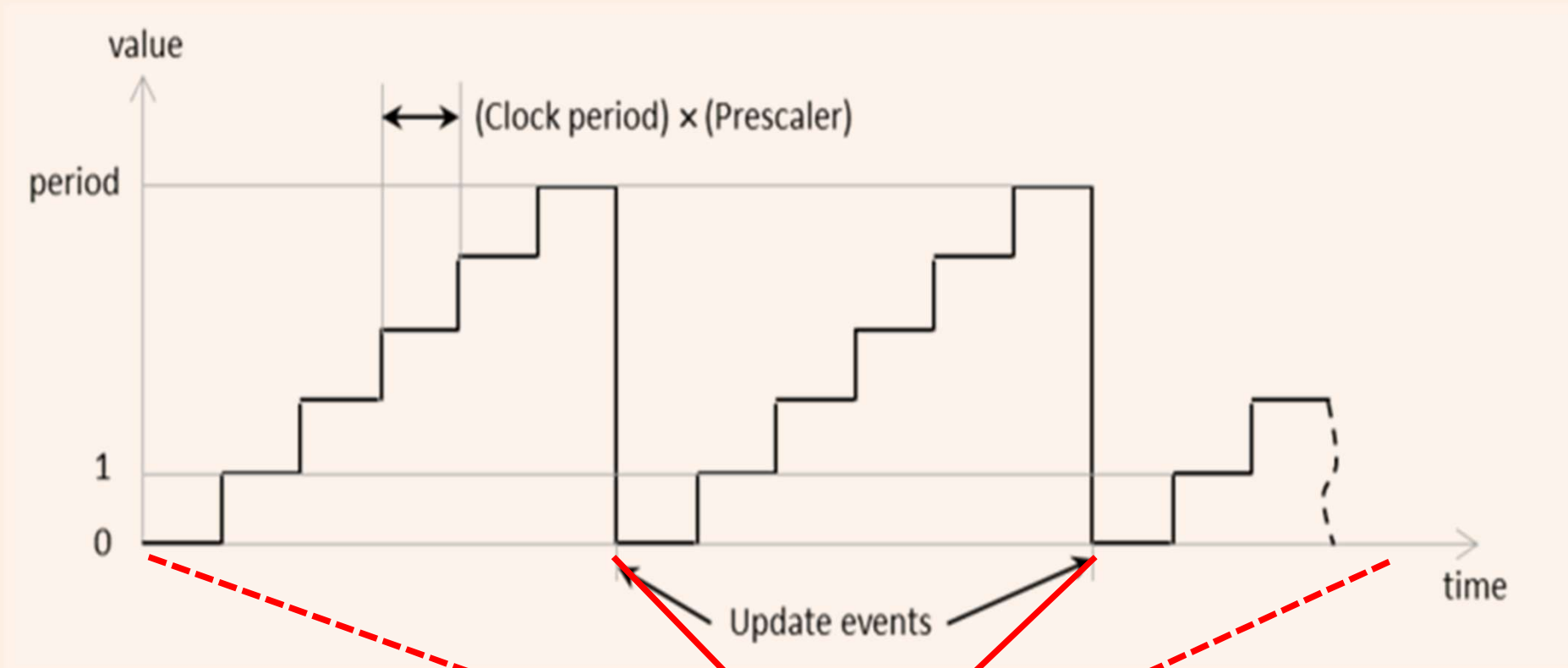
Configuração:

The screenshot shows the STM32CubeMX Pinout & Configuration window. The left sidebar lists various components, with TIM2 selected under the Timers category. The main panel displays the TIM2 Mode and Configuration settings. The Mode section includes dropdown menus for Slave Mode, Trigger Source, Clock Source, and Channels (Channel1, Channel2, Channel3, Channel4, and Combined Channels), all set to Disable. Below these are checkboxes for 'Use ETR as Clearing Source', 'XOR activation', and 'One Pulse Mode', all of which are unchecked. The Configuration section at the bottom shows a 'Reset Configuration' button and a table of settings. The 'NVIC Interrupt Table' is set to 'Enabled', and the 'TIM2 global interrupt' is checked in the 'Enabled' column, which is circled in red.

Configuration			
Reset Configuration			
User Constants			
NVIC Settings			
DMA Settings			
Parameter Settings			
NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
TIM2 global interrupt	<input checked="" type="checkbox"/>		0

# Interrupts

Configuração:



Nestes instantes são gerados interrupts!

# Interrupts

Configuração: Adicionar este Código no ficheiro main.c. Recomenda-se que seja colocado entre: “/\*USER CODE BEGIN 4 \*/” e “/\*USER CODE END 4 \*/”

```
/* USER CODE BEGIN 4 */  
  
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)  
{  
    if (htim->Instance==TIM2)  
    {  
        HAL_UART_Transmit(&huart2, (uint8_t *)"Inside_ISR", strlen("Inside_ISR"), 0xFFFF);  
        TIMER2_FLAG= 1;  
    }  
}  
  
/* USER CODE END 4 */
```



# Interrupts

Configuração: não esquecer de declarar a variável `TIMER2_FLAG` como variável global, tipicamente na zona das “Private variables”.

```
/* Private variables -----*/  
  
/* USER CODE BEGIN PV */  
  
char TIMER2_FLAG= 0;  
  
/* USER CODE END PV */
```

Configuração: não esquecer iniciar o timer e suas interrupções antes do loop infinito, por ex. entre “/\* USER CODE BEGIN 2 \*/” e “/\* USER CODE BEGIN 2 \*/”.

```
MX_TIM2_Init();  
  
/* USER CODE BEGIN 2 */  
  
HAL_TIM_Base_MspInit(&htim2);  
HAL_TIM_Base_Start_IT(&htim2);  
  
/* USER CODE END 2 */
```