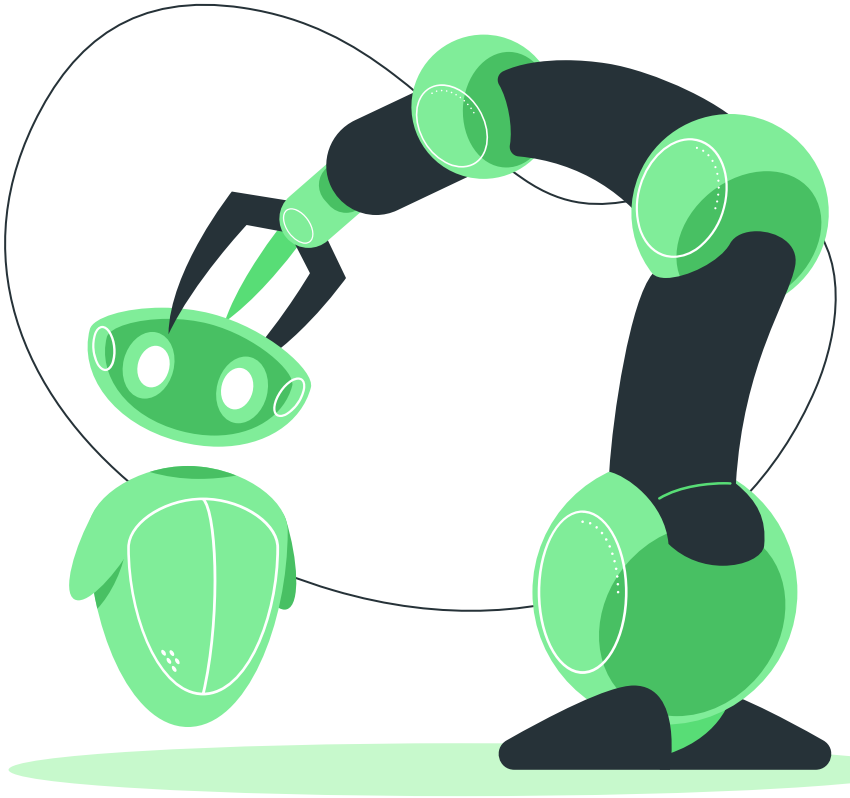
Freeways
free software club

STM32 Workshop

Session 2

By **Moktar SELLAMI**



Freeways
free software club

Plan

1. Quiz
2. Setup ST account
3. STM32Cube IDE Installation
4. HAL
5. HAL APIs
6. Blinking an LED
7. Reading Button state

Quiz

Quiz

1. What does GPIO stand for?

A) General Processor Input/Output

B) General Purpose Input/Output

C) Generic Port Input/Output

D) General Programmable Input/Output

Quiz

2. On an STM32 microcontroller, how are GPIO pins grouped?

A) Into clusters called PORTS (e.g., GPIOA, GPIOB)

B) Into clusters called BANKS

C) Into clusters called MODULES

D) Into clusters called UNIT

Quiz

3. If an internal green LED is connected to "PD12" meaning:

A) It is connected to Peripheral Device 12.

B) It is connected to Pin Diagram 12

C) It is connected to Power Domain 12.

D) It is connected to Port D, Pin 12..

Quiz

4. Which of the following is NOT one of the four main GPIO modes on an STM32?

A) Input

B) Output

C) Analog

D) Serial

E) Alternate Function

Quiz

5. In a GPIO output configuration, what is a key characteristic of the "Push-Pull" mode?

A) It can only sink current, connecting the pin to ground for a logic low.

B) It uses an external resistor to pull the line high when inactive.

C) It can actively drive the pin both high (to VCC) and low (to GND).

D) It is used exclusively for analog signals like audio.

Quiz

6. Why is an internal "Pull-Up" resistor often used with a GPIO pin configured as an input?

A) To increase the pin's output current for driving an LED.

B) To set a known default logic state (high) when no external signal is present.

C) To convert the pin into an analog input for a sensor.

D) To protect the pin from electrostatic discharge (ESD).

Quiz

7. In the context of STM32's HAL, which functions would you use to change the state of an LED ?

A) HAL_GPIO_ReadPin()

B) HAL_GPIO_WritePin()

C) HAL_GPIO_TogglePin()

D) HAL_GPIO_LockPin()

Quiz

Answers:

1 - B

2- A

3- D

4 - D

5- C

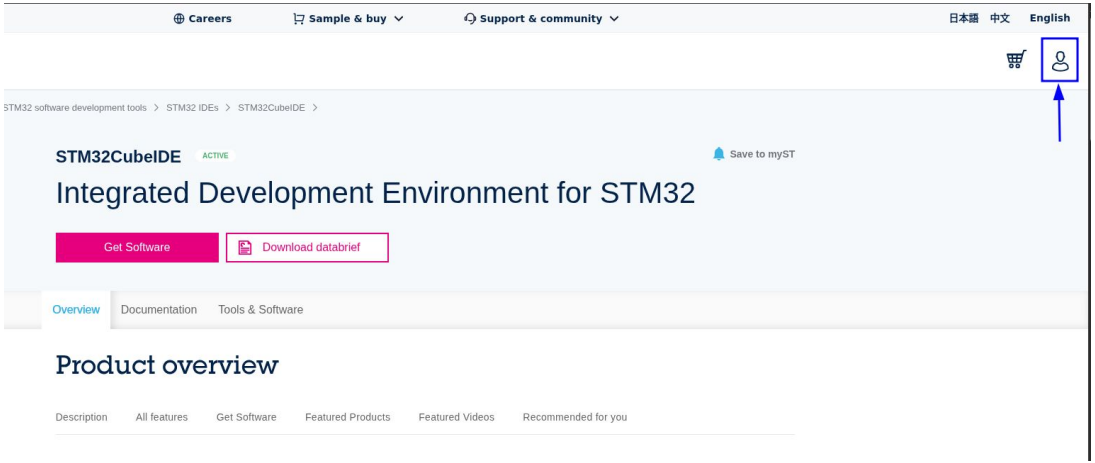
6- B

7 - B & C

Setting up the Environment: STM32 CubeIDE

0- You need a lot of patience.

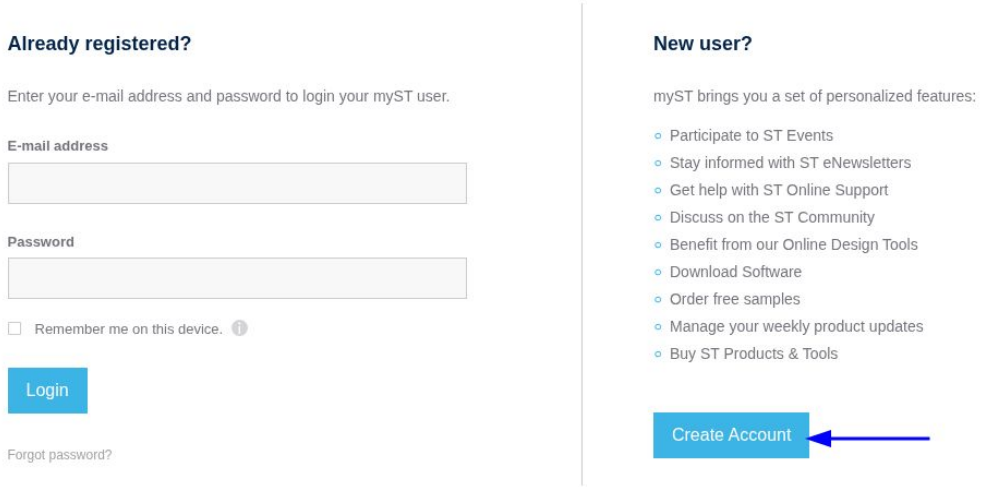
1- Go this link and create a account, you will get an email verification.



Setting up the Environment: STM32 CubeIDE

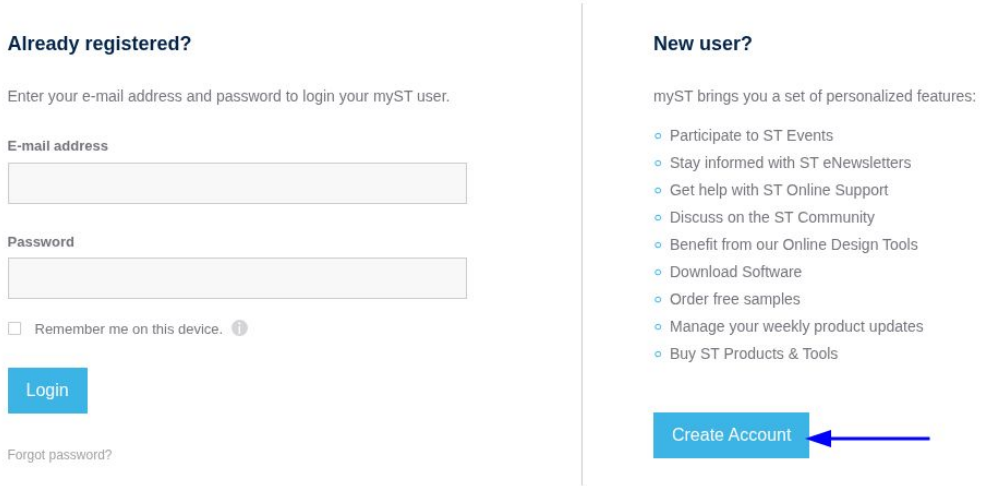
2- Create the account, fill in the necessary information

Fill in only the necessary info that are indicated with *



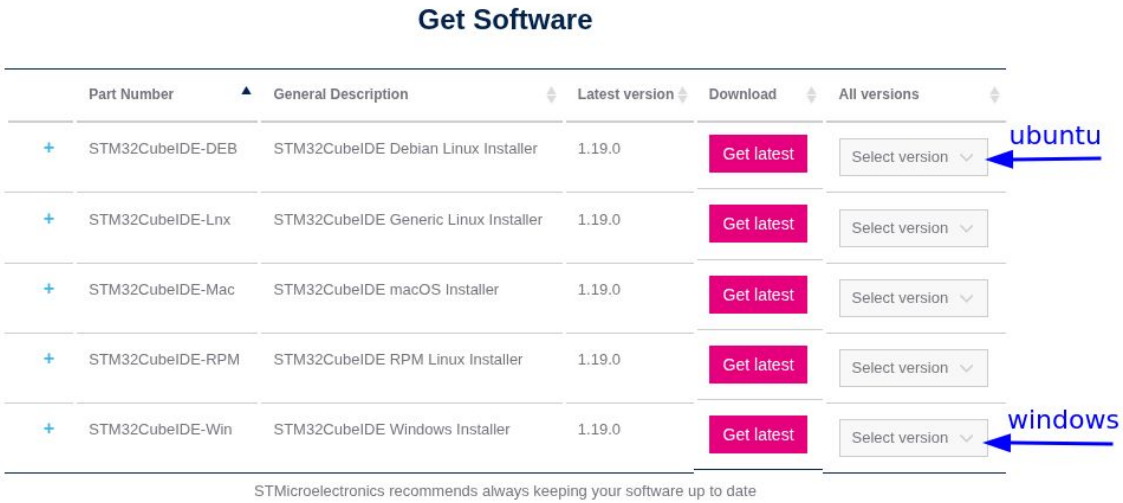
Setting up the Environment: STM32 CubeIDE

3- After some time you will receive and email of verification, verify it and your St account should be created.



Setting up the Environment: STM32 CubeIDE

4- Go this link again and download the Latest STM32 CubeIDE version



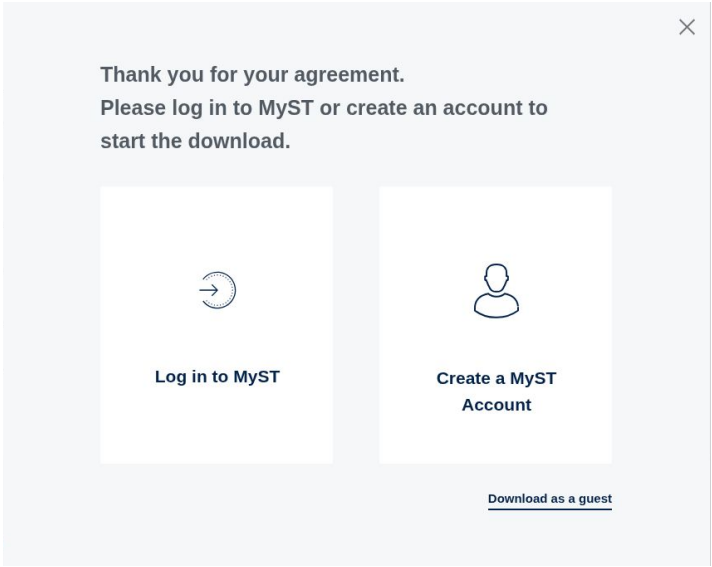
Setting up the Environment: STM32 CubeIDE

5- Accept to the Licence agreement

6- It i will prompt you to Login or use a guest choose Log in to MyST

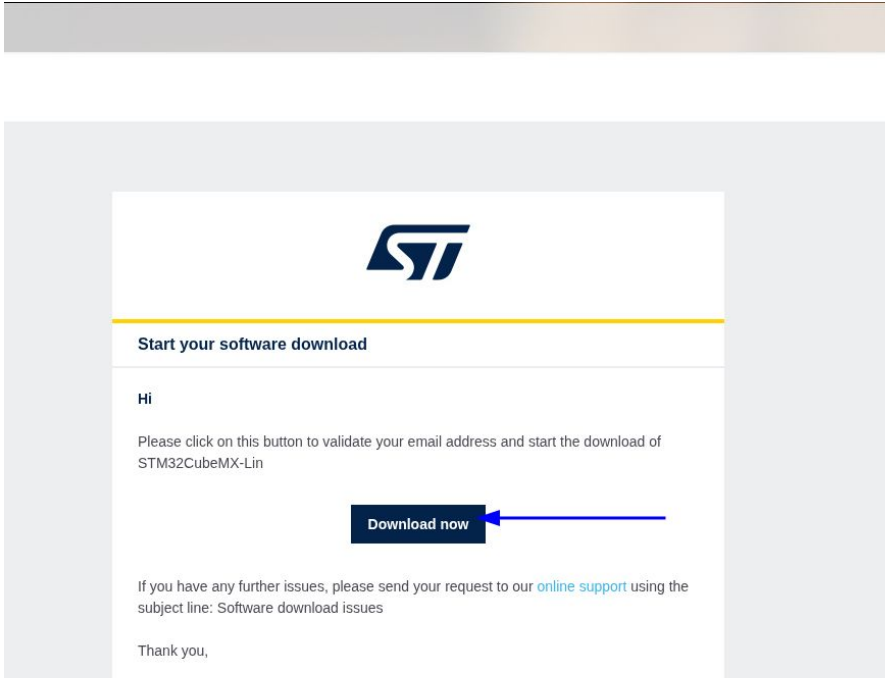
Note:

We create an account because, CubeIDE requires it sometimes



Setting up the Environment: STM32 CubeIDE

7- An Email should be sent to you, give it some time .

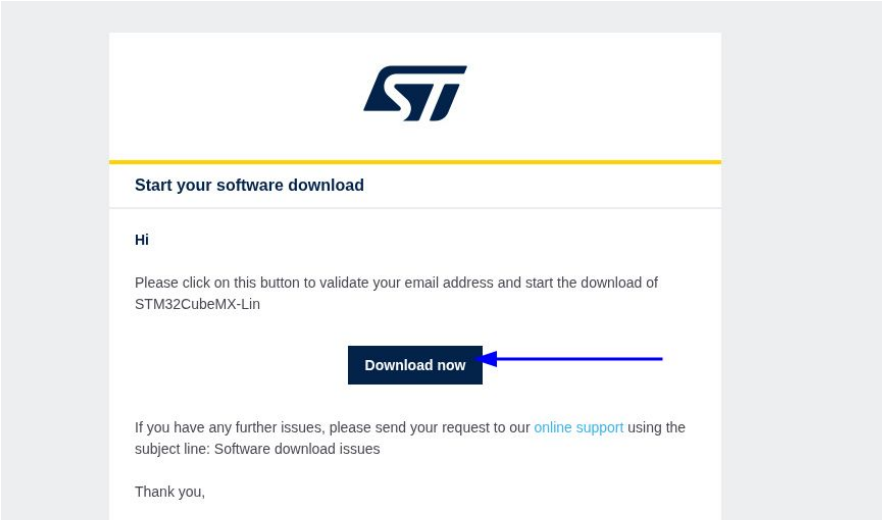


Setting up the Environment: STM32 CubeIDE

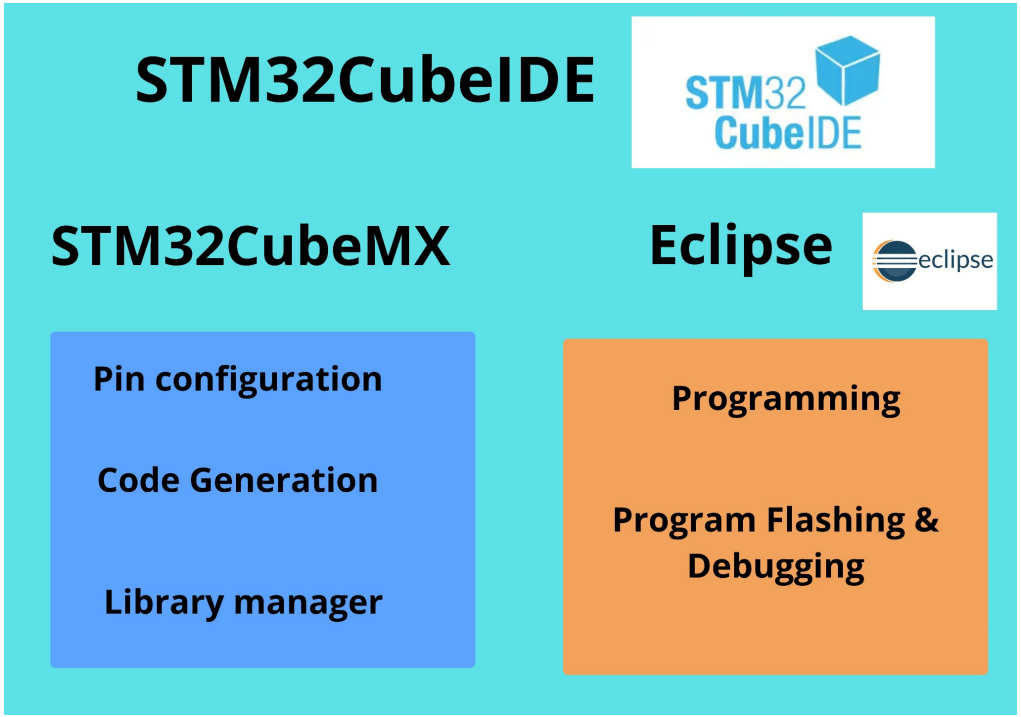
8- it will open a new session on you browser and hopefully, is will start downloading

Note:

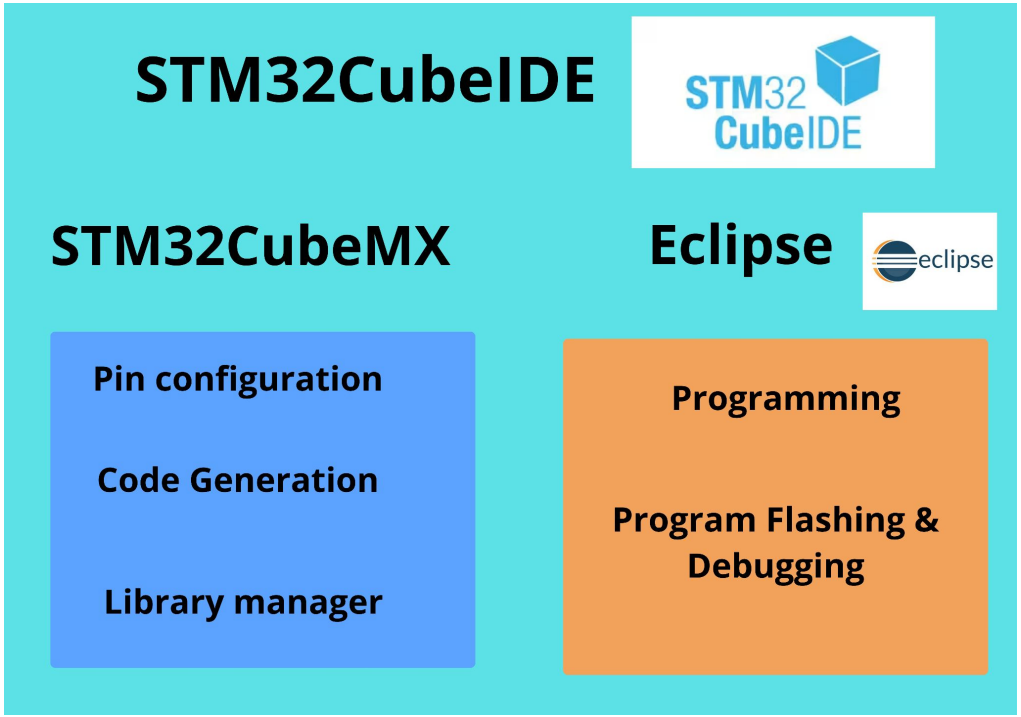
In case it didn't start, restart the downloading process.



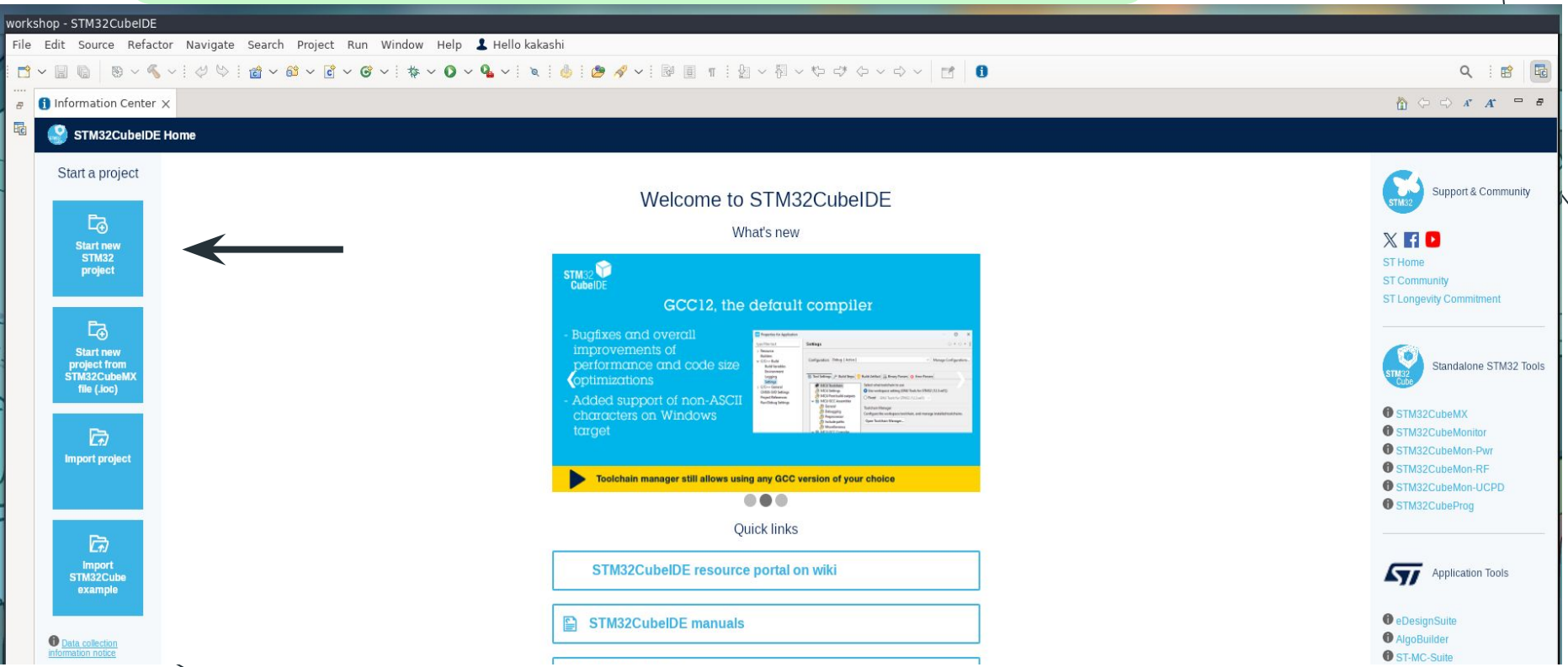
STM32Cube IDE



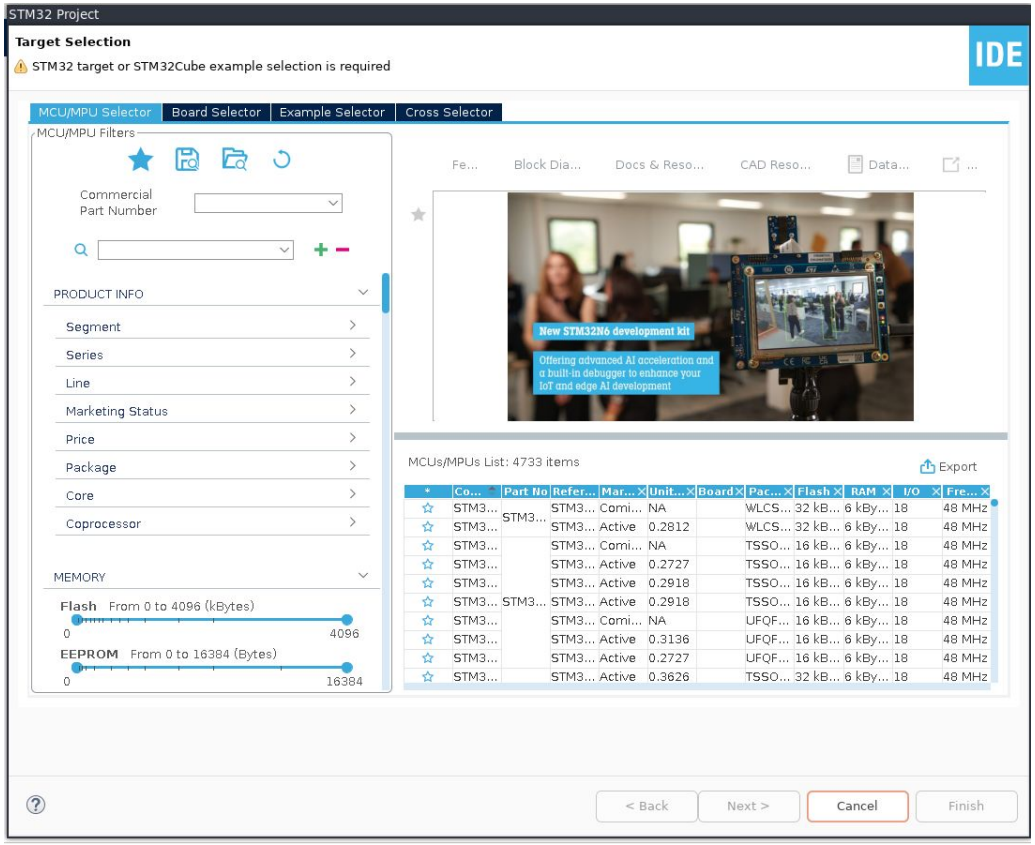
STM32Cube IDE



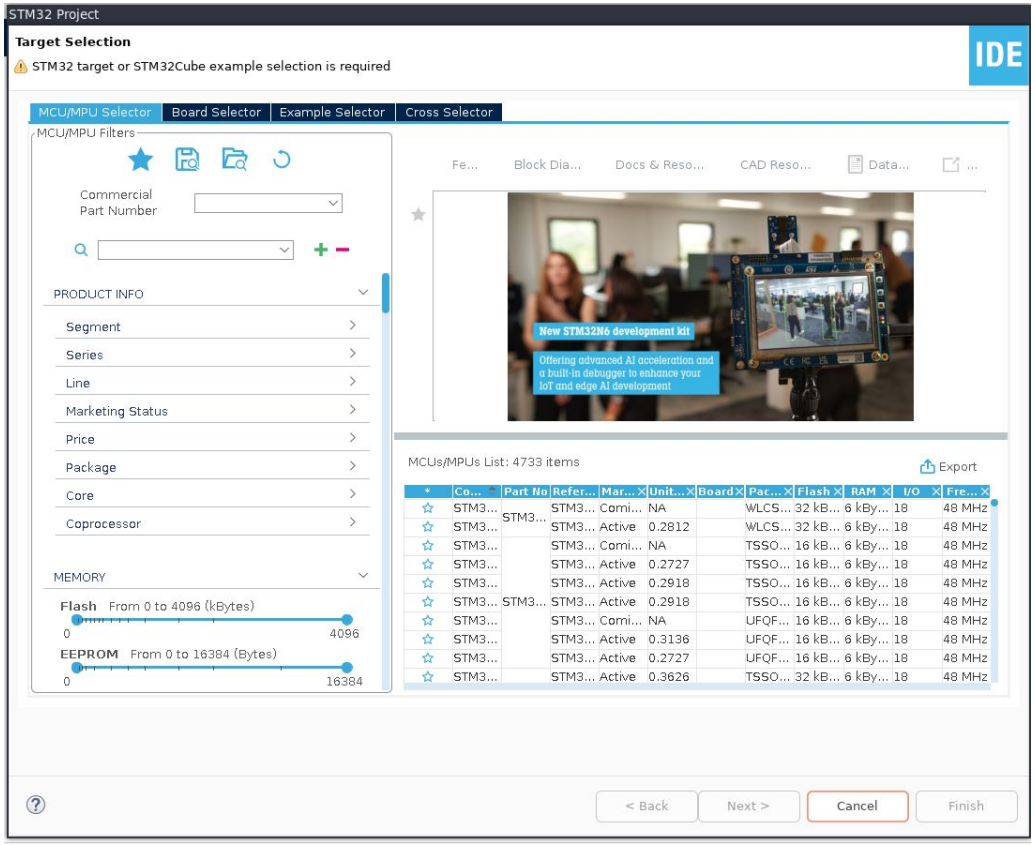
STM32Cube IDE: new project



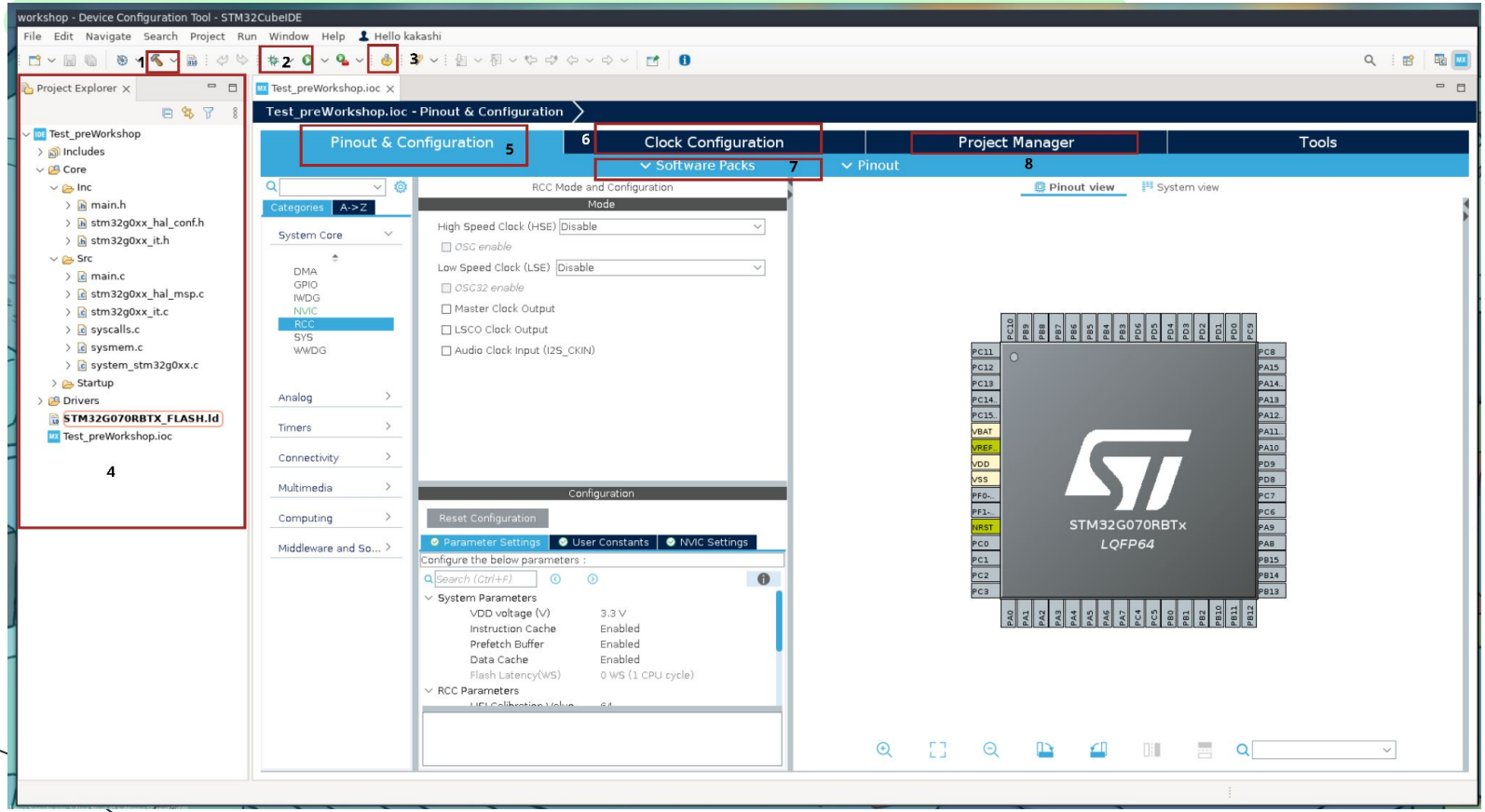
STM32Cube IDE: select target



STM32Cube IDE: Creating the project

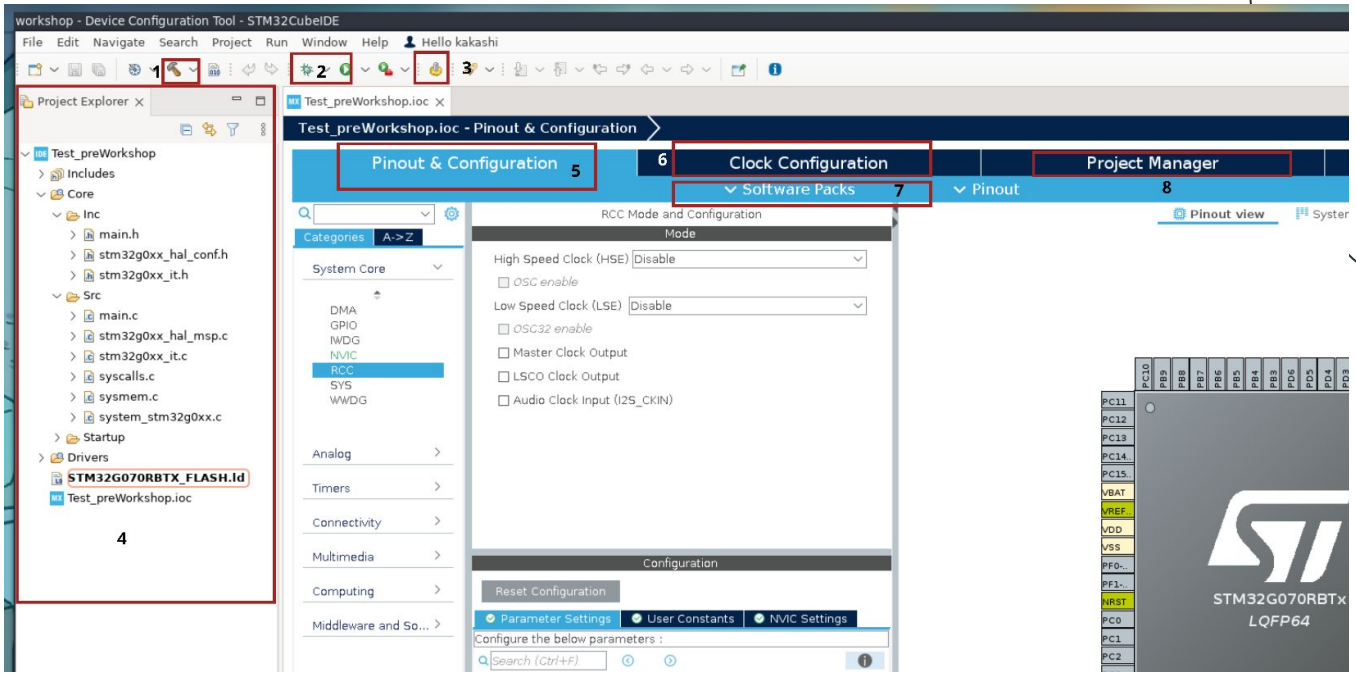


STM32Cube IDE: Creating the project



STM32Cube IDE: Creating the project

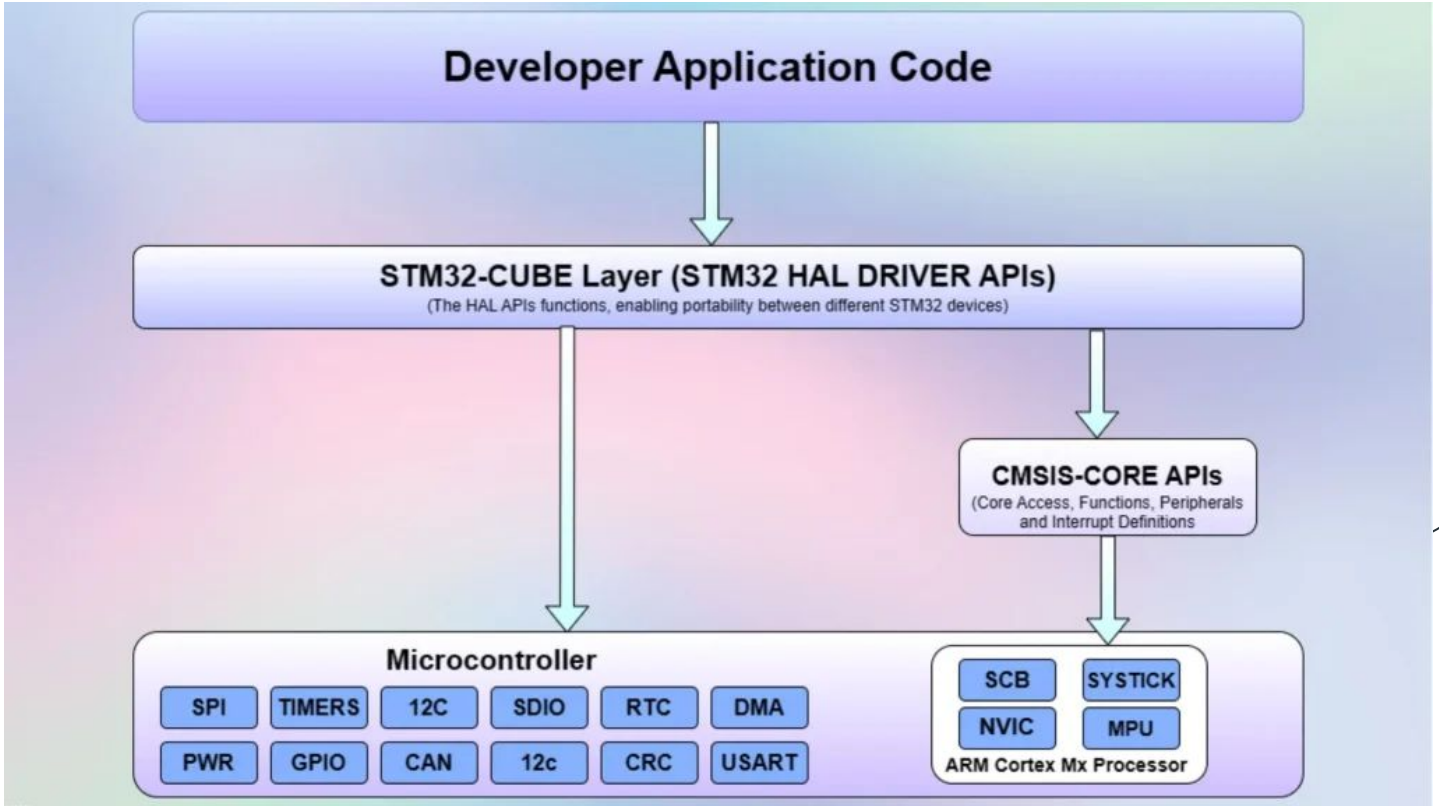
1- Build
2- Run & debug
3- Generate code
4- Project structure
5- Pinout config
6- clock config
7- Software packs
8- Project manager



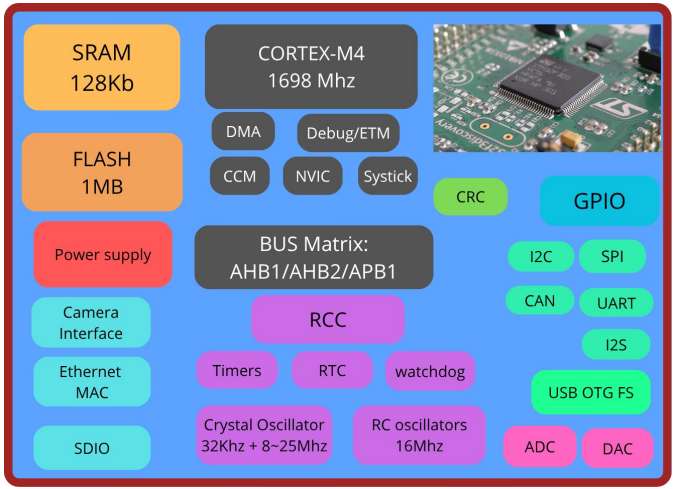
HAL

HAL is a software layer provided by STMicroelectronics that abstracts the hardware details of STM32 microcontrollers. It provides a standardized set of functions to interact with peripherals (GPIO, UART, I2C, etc.) regardless of the specific STM32 chip you're using.

HAL


Developer Application Code
STM32-CUBE Layer (STM32 HAL DRIVER APIs)
(The HAL APIs functions, enabling portability between different STM32 devices)
CMSIS-CORE APIs
(Core Access, Functions, Peripherals and Interrupt Definitions
Microcontroller
SPI
TIMERS
12C
SDIO
RTC
DMA
PWR
GPIO
CAN
12c
CRC
USART
SCB
SYSTICK
NVIC
MPU
ARM Cortex Mx Processor

Programming: HAL



HAL APIs

Programming

PIn Configuration and Setup

main

My Logic

Programming: Pin Configuration

Enable BUS Clock for GPIO

GPIO Configuration

Main Program

HAL:

GPIO InitStruct

```
/**
  * @brief GPIO Init structure definition
  */
typedef struct
{
  uint32_t Pin;       /*!< Specifies the GPIO pins to be configured.
                           This parameter can be any value of @ref GPIO_pins_define */

  uint32_t Mode;      /*!< Specifies the operating mode for the selected pins.
                           This parameter can be a value of @ref GPIO_mode_define */

  uint32_t Pull;      /*!< Specifies the Pull-up or Pull-Down activation for the selected pins.
                           This parameter can be a value of @ref GPIO_pull_define */

  uint32_t Speed;     /*!< Specifies the speed for the selected pins.
                           This parameter can be a value of @ref GPIO_speed_define */

  uint32_t Alternate;  /*!< Peripheral to be connected to the selected pins.
                           This parameter can be a value of @ref GPIO_Alternate_function_selection */
}GPIO_InitTypeDef;
```

HAL:

GPIO Mode

```
/** @defgroup GPIO_mode_define GPIO mode define
  * @brief GPIO Configuration Mode
  *        Elements values convention: 0x00WX00YZ
  *           - W  : EXTI trigger detection on 3 bits
  *           - X  : EXTI mode (IT or Event) on 2 bits
  *           - Y  : Output type (Push Pull or Open Drain) on 1 bit
  *           - Z  : GPIO mode (Input, Output, Alternate or Analog) on 2 bits
  * @{
  */
#define  GPIO_MODE_INPUT                        MODE_INPUT
#define  GPIO_MODE_OUTPUT_PP                    (MODE_OUTPUT | OUTPUT_PP)
#define  GPIO_MODE_OUTPUT_OD                    (MODE_OUTPUT | OUTPUT_OD)
#define  GPIO_MODE_AF_PP                        (MODE_AF | OUTPUT_PP)
#define  GPIO_MODE_AF_OD                        (MODE_AF | OUTPUT_OD)

#define  GPIO_MODE_ANALOG                       MODE_ANALOG
```

HAL:

GPIO Speed

```
21
20 /** @defgroup GPIO_speed_define  GPIO speed define
19   * @brief GPIO Output Maximum frequency
18   * @{
17   */
16 #define  GPIO_SPEED_FREQ_LOW         0x00000000U
15 #define  GPIO_SPEED_FREQ_MEDIUM      0x00000001U
14 #define  GPIO_SPEED_FREQ_HIGH        0x00000002U
13 #define  GPIO_SPEED_FREQ_VERY_HIGH   0x00000003U
12 /**
11   * @}
10   */
```

HAL:

GPIO Pin state

```
15 /**·
14   * @brief  GPIO Bit SET and Bit RESET enumeration·
13   */
12 typedef enum
11 {
10   GPIO_PIN_RESET = 0,
 9   GPIO_PIN_SET
 8 }GPIO_PinState;
```

Freeways
free software club

HAL GPIO APIs:

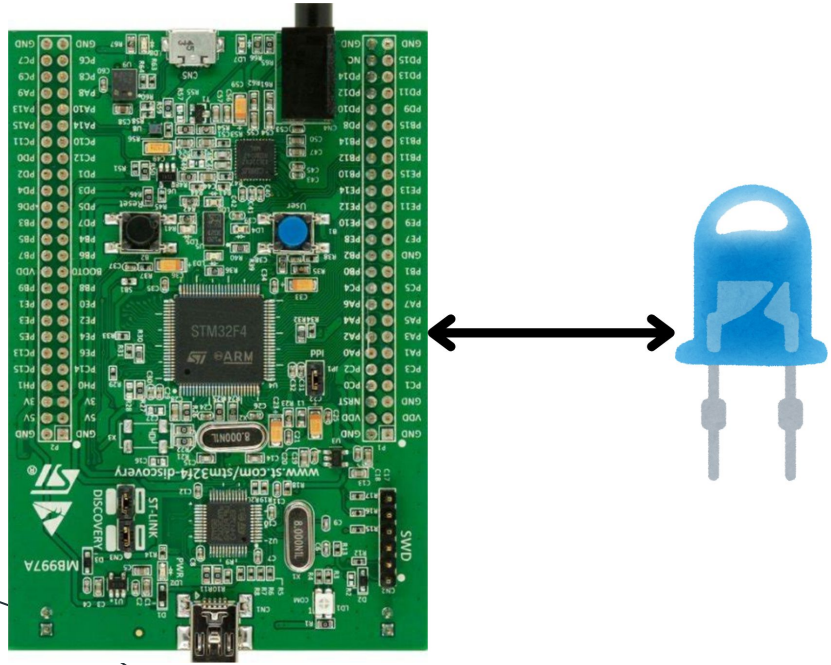
HAL function

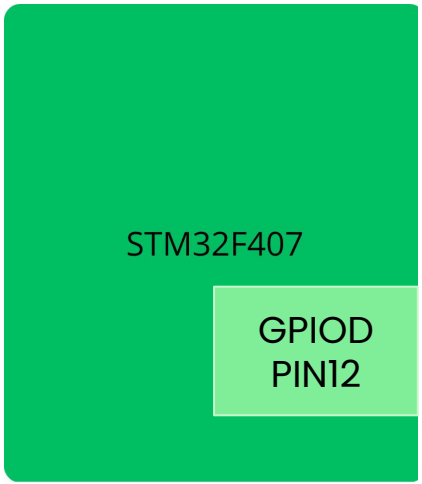
```
/** @addtogroup GPIO_Exported_Functions_Group2
  * @{
  */
/* IO operation functions *****************************************************/
GPIO_PinState HAL_GPIO_ReadPin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
void HAL_GPIO_WritePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState);
void HAL_GPIO_TogglePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
```

```
/** @addtogroup GPIO_Exported_Functions_Group1
  * @{
  */
/* Initialization and de-initialization functions *****************************/
void  HAL_GPIO_Init(GPIO_TypeDef  *GPIOx, GPIO_InitTypeDef *GPIO_Init);
void  HAL_GPIO_DeInit(GPIO_TypeDef  *GPIOx, uint32_t GPIO_Pin);
/**
  * @}
  */
```

GPIO output STM32 and HAL: blinking an LED:

Toggling an LED





GPIO Input STM32 and HAL: Button

Reading button state

STM32F407

GPIOA
PIN0

Freeways

free software club

Thank You