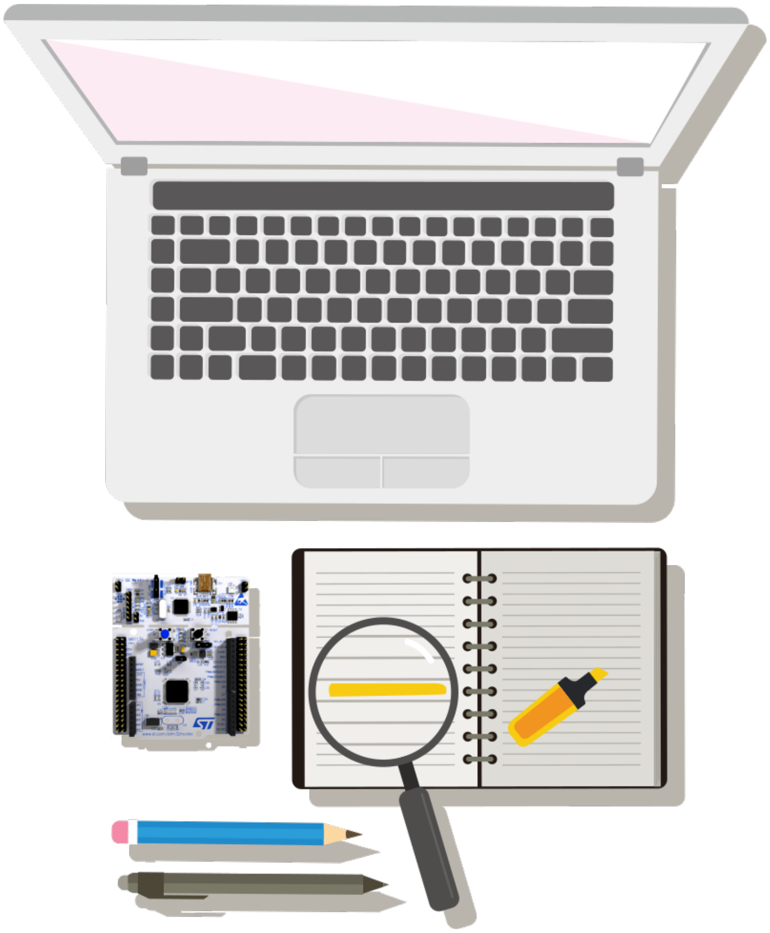
In-class activity 1B

Name: Ben Soto



Requirements

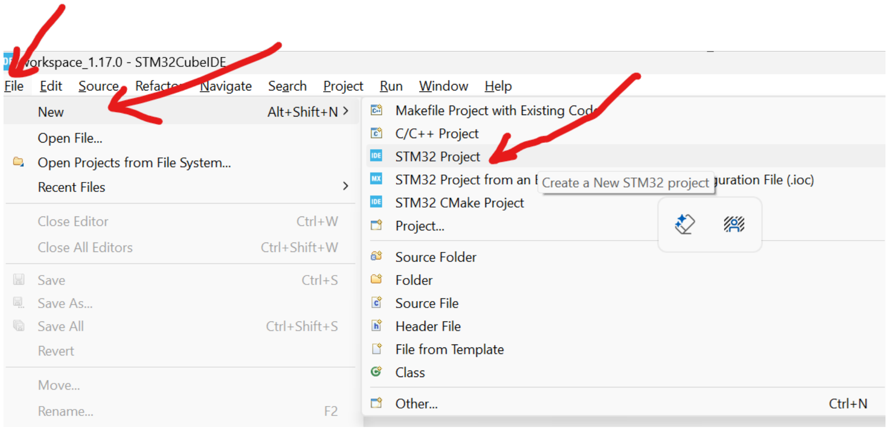
- STM32 L476RG Nucleo board
- STM32CubeIDE
- Lecture slides and hands-on files in class.

Instructions

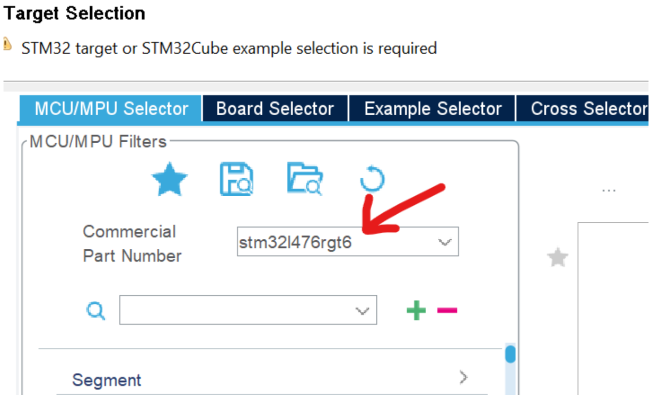
Understand the GPIO register and how to use the debugger to observe input signal.

Step 1: (Create a project and enable SWD)

Connect the board to an USB port.
Launch STM32CubeIDE and choose the default working directory.
Click on File, New, then STM32 Project as below.



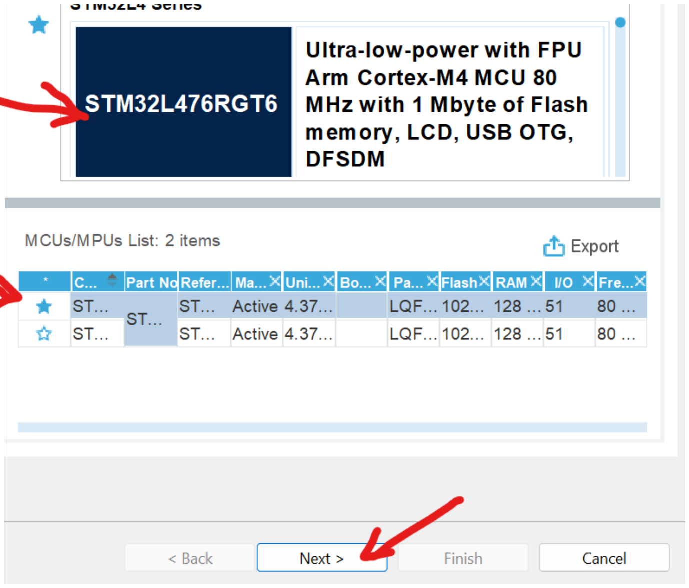
In the target selection, type STM32L476RGT6 as below.


Target Selection

STM32 target or STM32Cube example selection is required

Commercial Part Number: stm32l476rgt6

Choose the first one in the list and click on Next.



STM32L4 Series

STM32L476RGT6

Ultra-low-power with FPU Arm Cortex-M4 MCU 80 MHz with 1 Mbyte of Flash memory, LCD, USB OTG, DFSDM

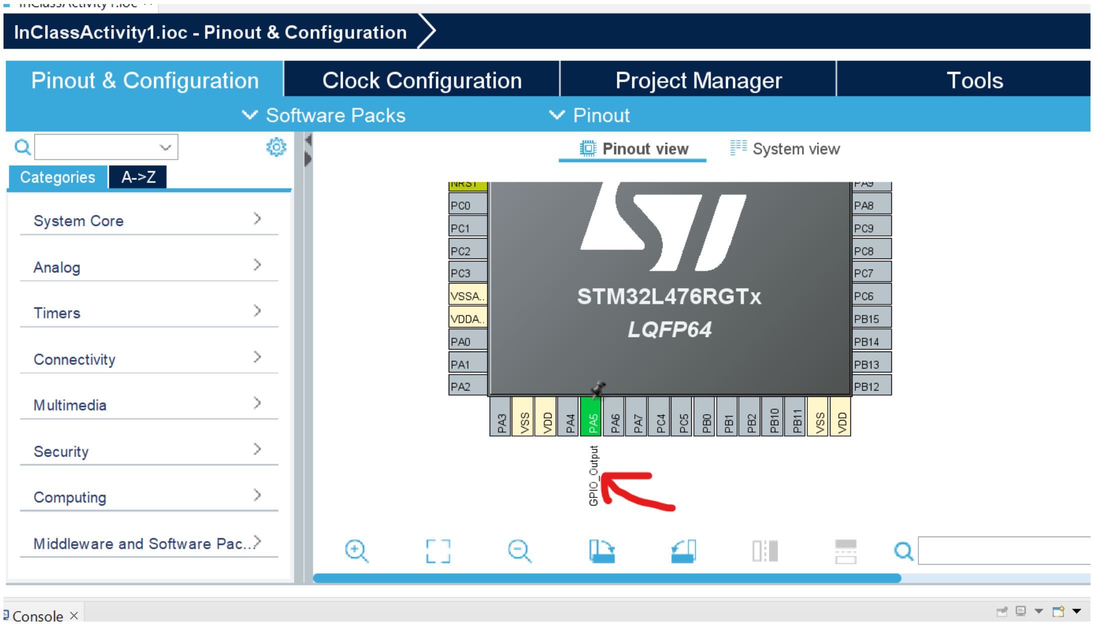
MCUs/MPUs List: 2 items

*	C...	Part No	Refer...	Ma...	Uni...	Bo...	Pa...	Flash	RAM	I/O	Fre...
★	ST...	ST...	ST...	Active	4.37...		LQF...	102...	128 ...	51	80 ...
☆	ST...		ST...	Active	4.37...		LQF...	102...	128 ...	51	80 ...

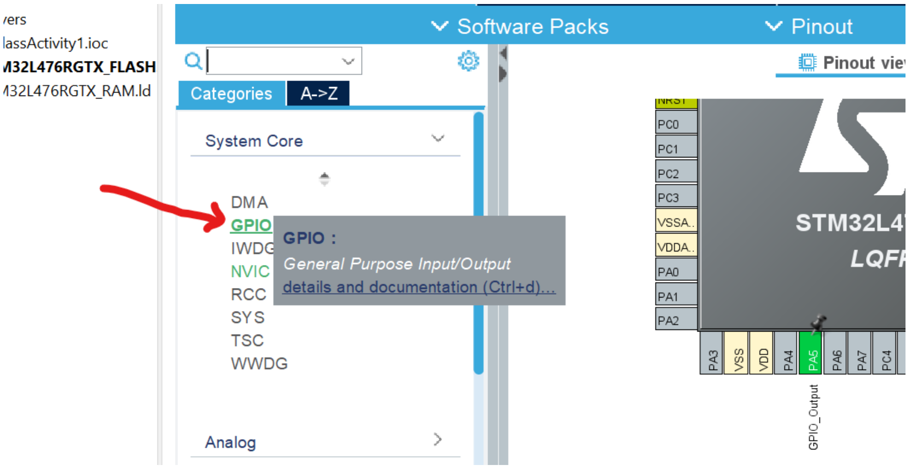
Create a Project Name such as InClassActivity1B then click on Finish.

Step 2 (Configure the on-board LED on PA5)

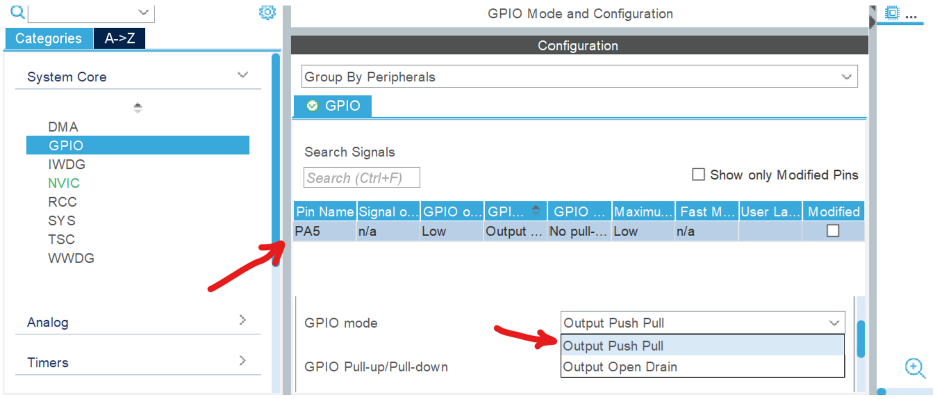
Click on the PA5 and configure it as GPIO_Output as below.



Step 3: Click GPIO to check PA5 configuration.



Ch

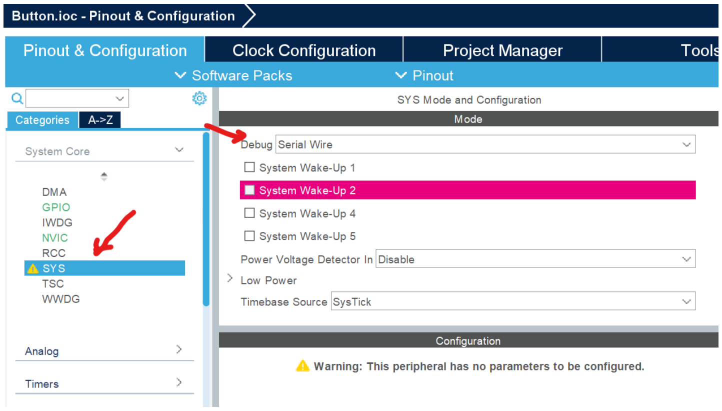


Pin Name	Signal o...	GPIO o...	GPI...	GPIO ...	Maximu...	Fast M...	User La...	Modified
PA5	n/a	Low	Output ...	No pull-...	Low	n/a		☐

GPIO mode: Output Push Pull

Step 4: Choose PA5 output as Push Pull.

Step 5: Enable SWD (for debugger) under System Core.



Debug: Serial Wire

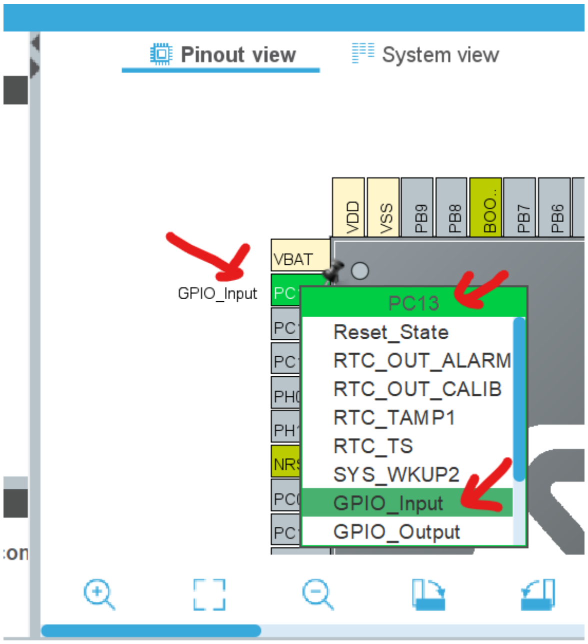
☐ System Wake-Up 1
☐ System Wake-Up 2
☐ System Wake-Up 4
☐ System Wake-Up 5

Power Voltage Detector In: Disable

Timebase Source: SysTick

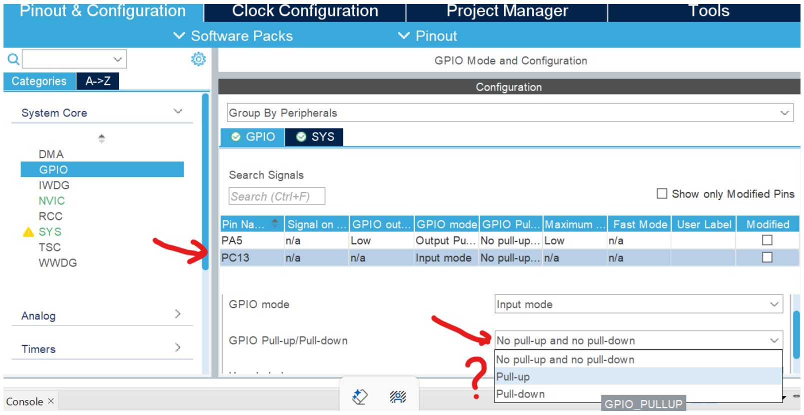
Warning: This peripheral has no parameters to be configured.

Step 6: Review the datasheet from canvas to find where the “blue” pushbutton is connected and make the pin as input.



The “blue” button is connected to GPIOC pin 13. Make it input as shown above.

Step 7: Check the schematic diagram to decide if the button input needs an internal pull-up. (Screenshot and paste under the figure below with the schematic diagram where makes you to decide the need or no need of a pull-up resistor for PC13.)

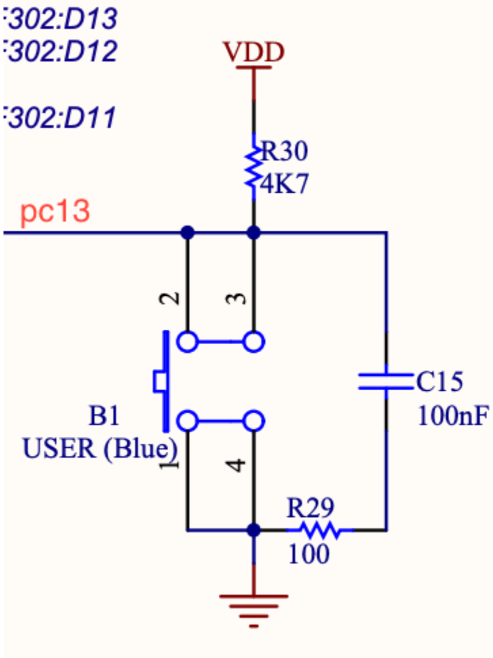


Pin Na...	Signal on ...	GPIO out...	GPIO mode	GPIO Pul...	Maximum ...	Fast Mode	User Label	Modified
PA5	n/a	Low	Output Pu...	No pull-up...	Low	n/a		□
PC13	n/a	n/a	Input mode	No pull-up...	n/a	n/a		□

GPIO mode: Input mode

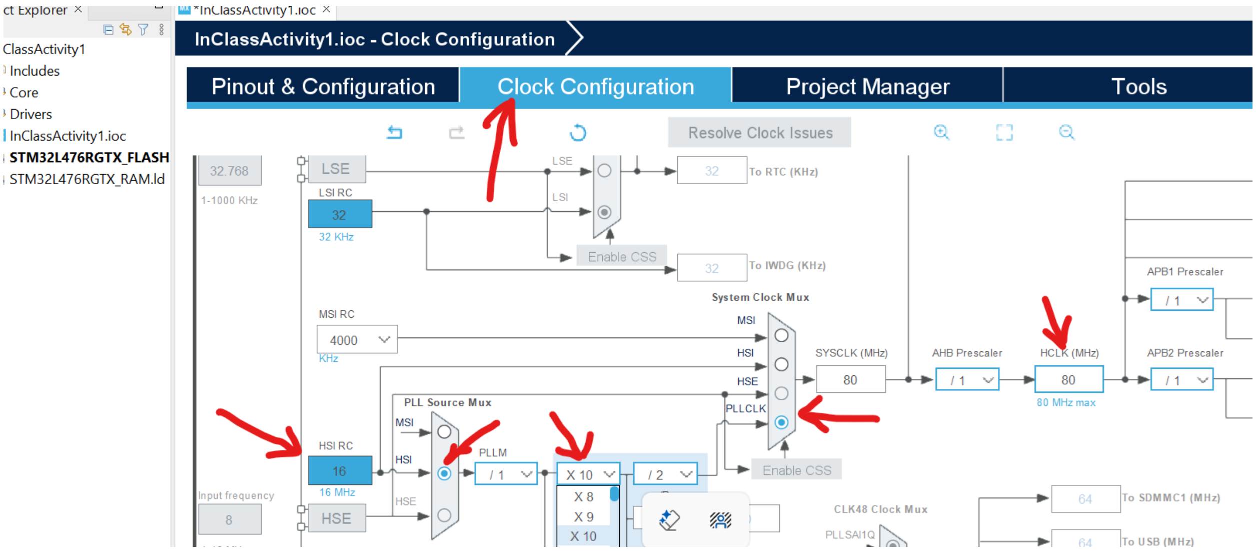
GPIO Pull-up/Pull-down: No pull-up and no pull-down

- No pull-up and no pull-down
- Pull-up
- Pull-down

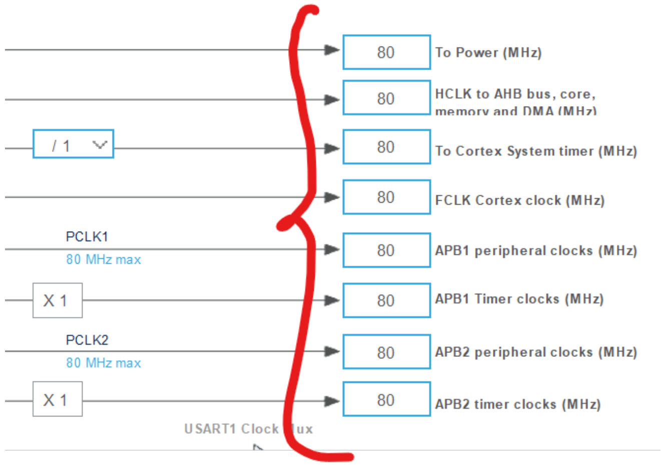


No need a pull up resistor since there is already one (R30).

Step 8: Set up clock to be High Speed Internal with PLL to bump to 80MHz to all buses.

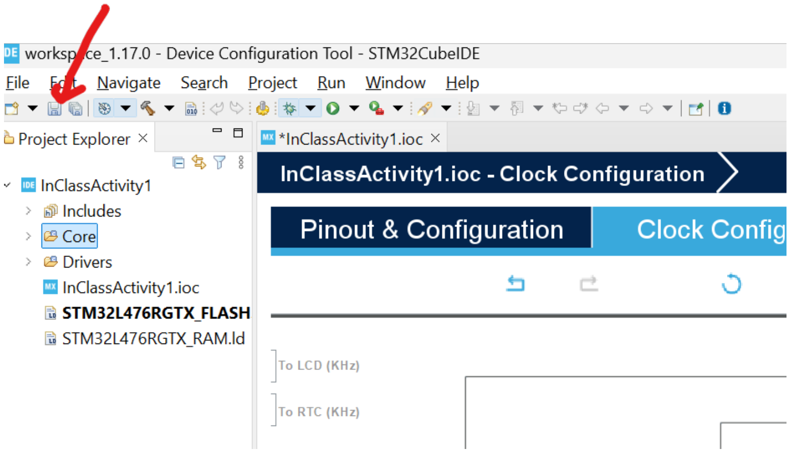


Make sure all buses run at 80MHz.



80 To Power (MHz)
80 HCLK to AHB bus, core, memory and DMA (MHz)
/ 1 — 80 To Cortex System timer (MHz)
80 FCLK Cortex clock (MHz)
PCLK1 80 MHz max — 80 APB1 peripheral clocks (MHz)
X 1 — 80 APB1 Timer clocks (MHz)
PCLK2 80 MHz max — 80 APB2 peripheral clocks (MHz)
X 1 — 80 APB2 timer clocks (MHz)

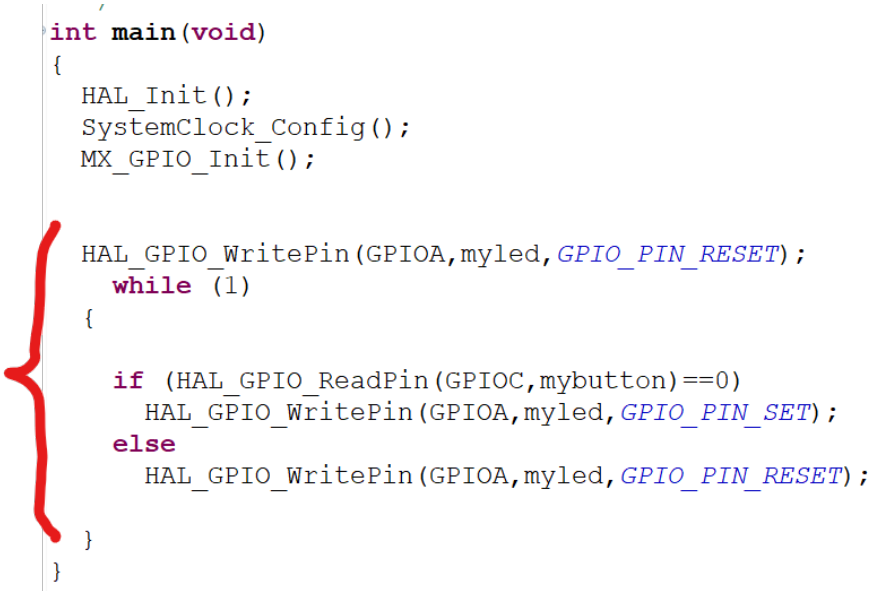
Step 9: After configuration (ioc) is done, click on the diskette to save and let it generate code as well as open perspective as shown.



Step 10: Click on Core, Src, main.c in the Project Explorer pane and type codes as below.

```
#define myled GPIO_PIN_5
#define mybutton GPIO_PIN_13
```

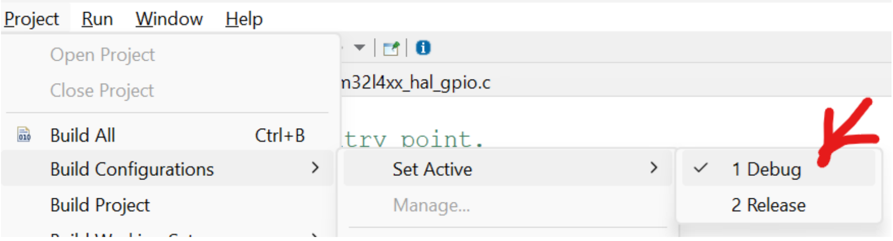
…



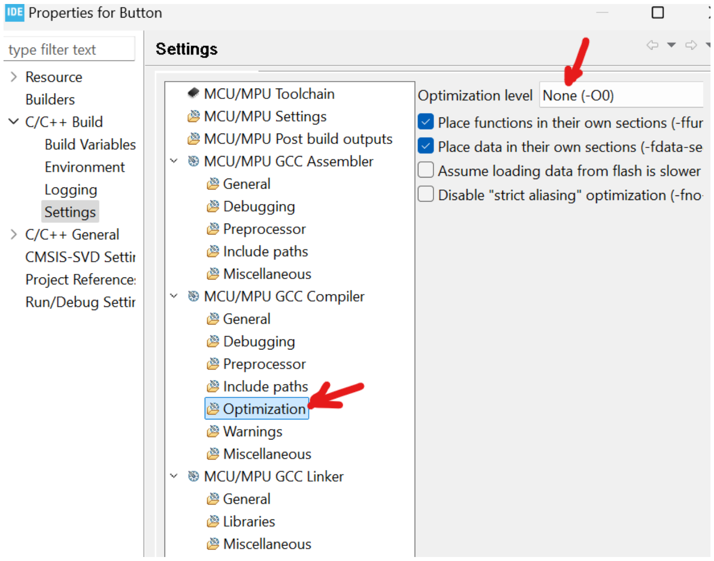
```
int main(void)
{
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();


  HAL_GPIO_WritePin(GPIOA,myled,GPIO_PIN_RESET);
    while (1)
  {

    if (HAL_GPIO_ReadPin(GPIOC,mybutton)==0)
      HAL_GPIO_WritePin(GPIOA,myled,GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOA,myled,GPIO_PIN_RESET);

  }
}
```

Use “// “ to add comment below each line. The comments should explain what the instruction intends to do. Screenshot and paste your code and comments below.

Step 11: Click on Project, Build Configurations, Set Active and Debug.



Step 12: Click on Project, Build All.
Step 13: Clock Project, Properties, then C/C++ Build then Settings. Make Optimization level to None under GCC C compiler.



Optimization level None (-O0)

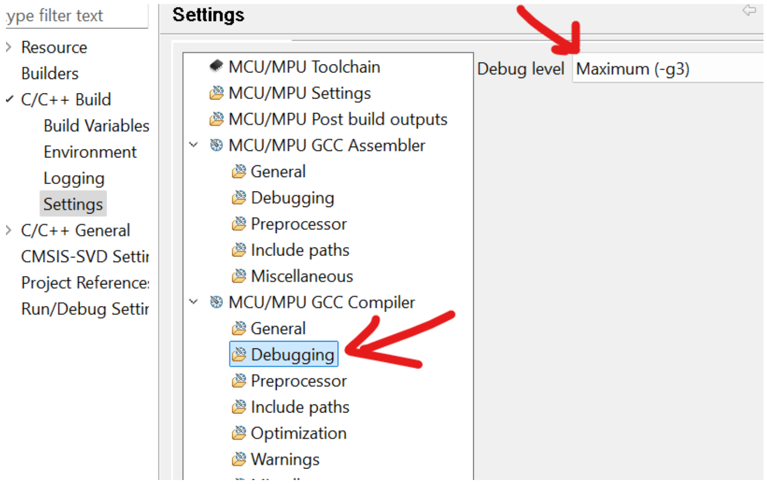
☑ Place functions in their own sections (-ffur

☑ Place data in their own sections (-fdata-se

□ Assume loading data from flash is slower

□ Disable "strict aliasing" optimization (-fno

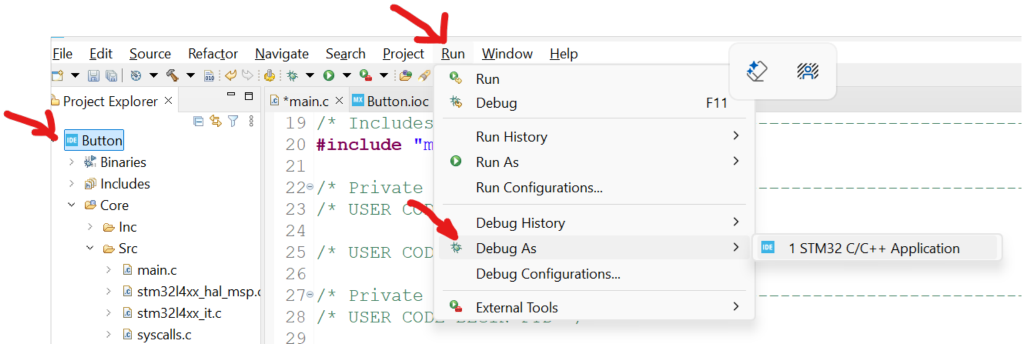
Make Debugging to Maximum.



Debug level Maximum (-g3)

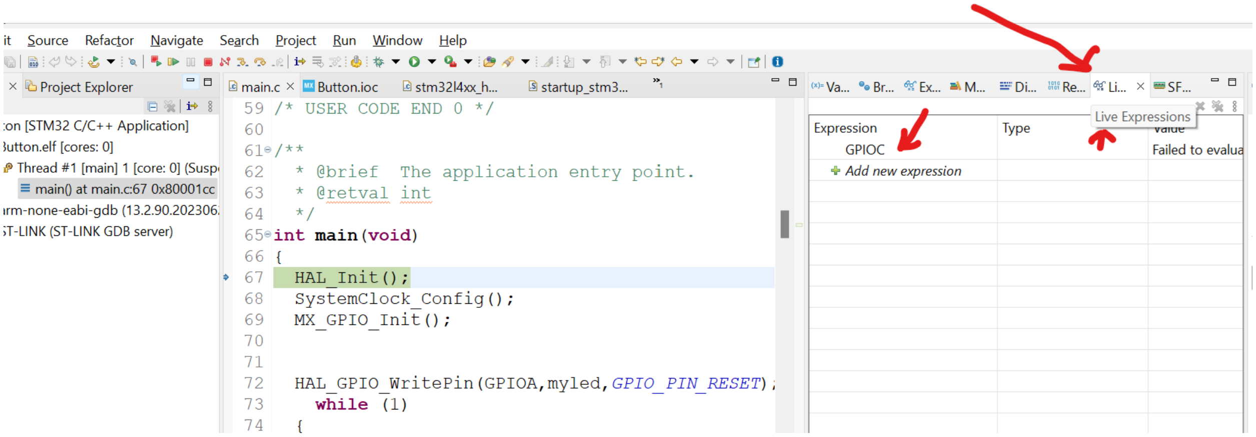
Then Project, Build All again.

Step 14:
Make sure the project name on the left is highlighted. Then, click on the Run, Debug As.



```
/* Includes
#include "m
/* Private
/* USER CO
/* USER COD
/* Private
/* USER COD
```

Step 15: Click “Switch” then the debugger window will be open as below.



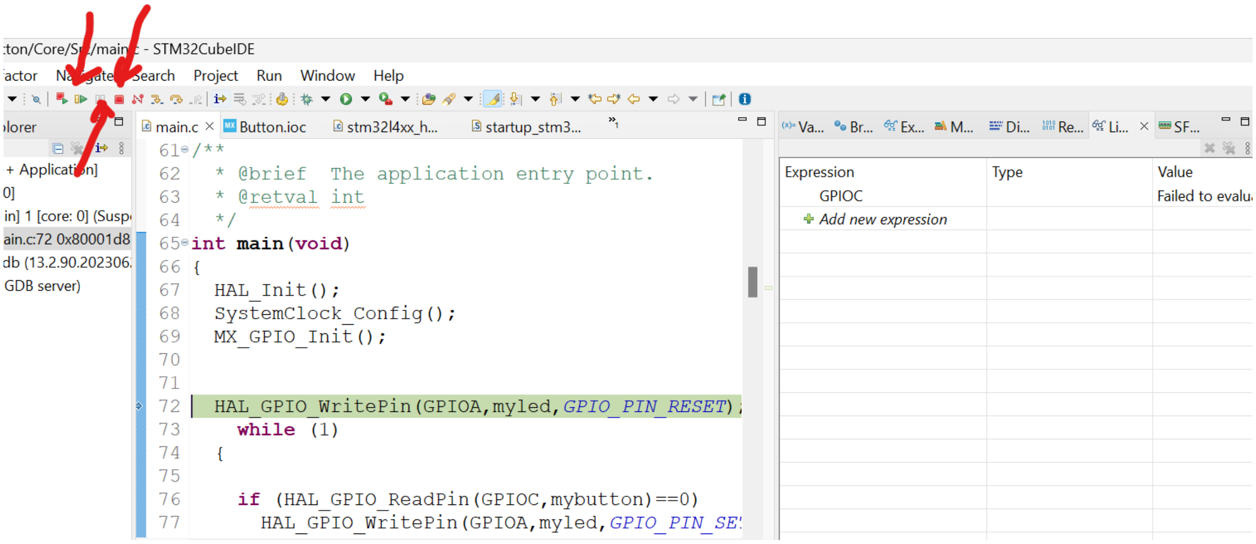
:on [STM32 C/C++ Application]
3utton.elf [cores: 0]
Thread #1 [main] 1 [core: 0] (Susp
main() at main.c:67 0x80001cc
arm-none-eabi-gdb (13.2.90.202306
ST-LINK (ST-LINK GDB server)

```
/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();


  HAL_GPIO_WritePin(GPIOA,myled,GPIO_PIN_RESET);
    while (1)
    {
```

Expression	Type	Value
GPIOC		Failed to evalua
Add new expression		

Click on Live Expressions and type GPIOC as shown above.

Step 16: Use F6 or the following icons as shown to debug.



+ Application]
0]
in] 1 [core: 0] (Susp
ain.c:72 0x80001d8
db (13.2.90.202306.
GDB server)

```
/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();


  HAL_GPIO_WritePin(GPIOA,myled,GPIO_PIN_RESET);
    while (1)
    {

      if (HAL_GPIO_ReadPin(GPIOC,mybutton)==0)
        HAL_GPIO_WritePin(GPIOA,myled,GPIO_PIN_SE
```

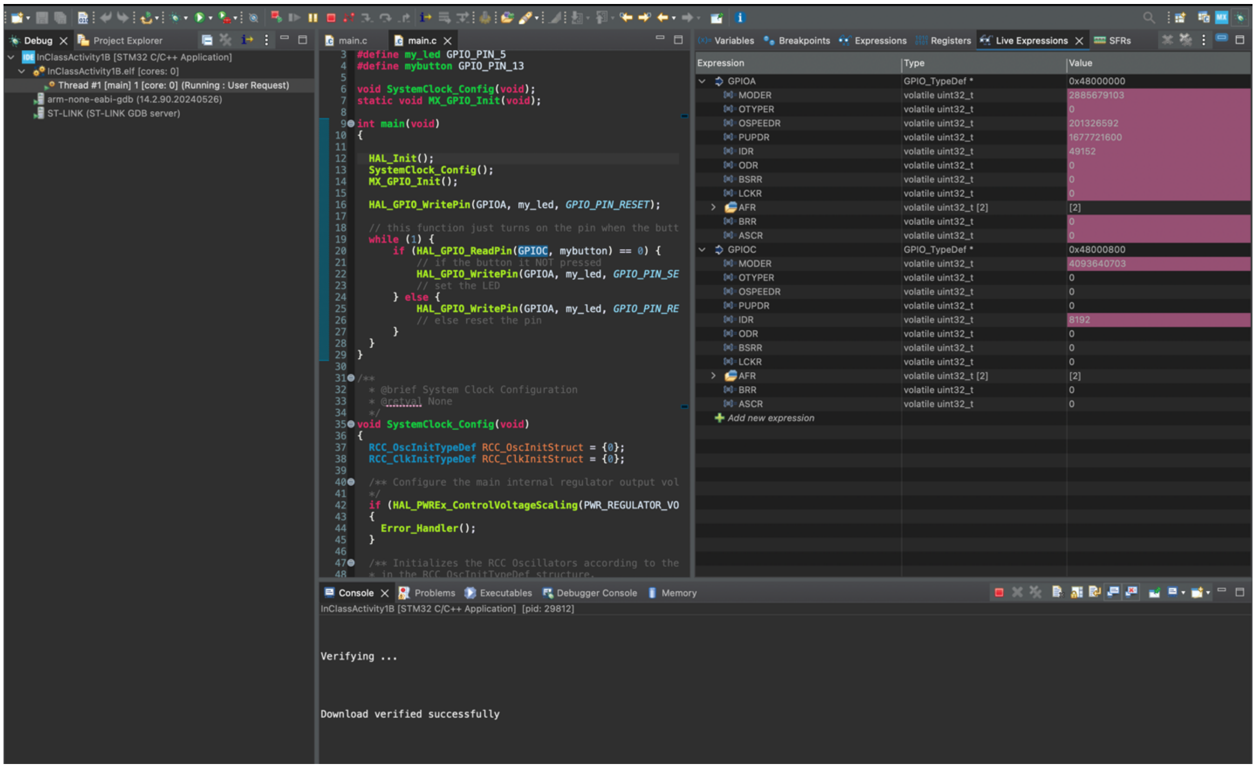
Expression	Type	Value
GPIOC		Failed to evalu
Add new expression		

Step 17: Observe the GPIOC-> IDR. This exercise demonstrate real-time data observation and the LED should be turned on when the pushbutton is pressed.

Screenshot the GPIOC->IDR and GPIOA->ODR below to verify when the button is pushed or released and the status of the LED.

Submit this document for grading.

No Press:



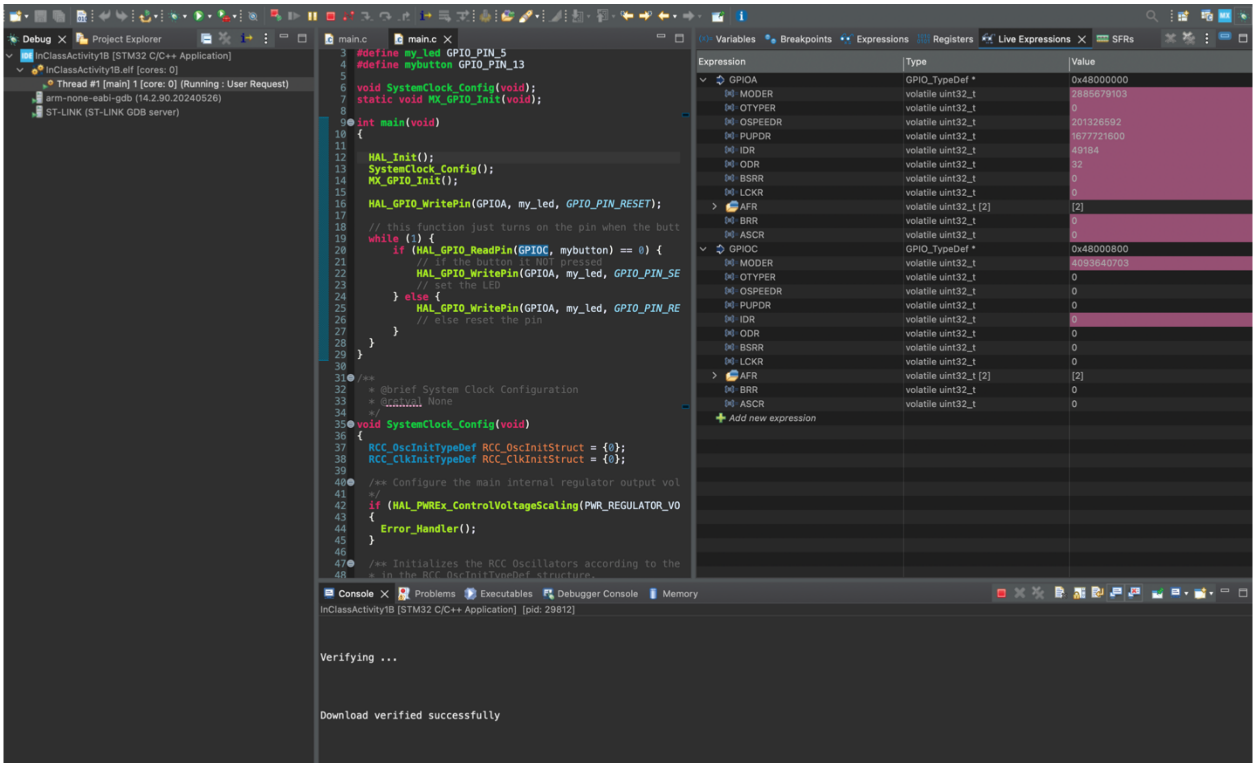
```
InClassActivity1B [STM32 C/C++ Application]
  InClassActivity1B.elf [cores: 0]
    Thread #1 [main] 1 [core: 0] (Running : User Request)
  arm-none-eabi-gdb (14.2.90.20240526)
  ST-LINK (ST-LINK GDB server)
```

```c
#define my_led GPIO_PIN_5
#define mybutton GPIO_PIN_13

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

int main(void)
{

  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();

  HAL_GPIO_WritePin(GPIOA, my_led, GPIO_PIN_RESET);

  // this function just turns on the pin when the butt
  while (1) {
      if (HAL_GPIO_ReadPin(GPIOC, mybutton) == 0) {
          // if the button it NOT pressed
          HAL_GPIO_WritePin(GPIOA, my_led, GPIO_PIN_SE
          // set the LED
      } else {
          HAL_GPIO_WritePin(GPIOA, my_led, GPIO_PIN_RE
          // else reset the pin
      }
  }
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output vol
  */
  if (HAL_PWREx_ControlVoltageScaling(PWR_REGULATOR_VO
  {
    Error_Handler();
  }

  /** Initializes the RCC Oscillators according to the
  * in the RCC OscInitTypeDef structure.
```

Expression	Type	Value
GPIOA	GPIO_TypeDef *	0x48000000
MODER	volatile uint32_t	2885679103
OTYPER	volatile uint32_t	0
OSPEEDR	volatile uint32_t	201326592
PUPDR	volatile uint32_t	1677721600
IDR	volatile uint32_t	49152
ODR	volatile uint32_t	0
BSRR	volatile uint32_t	0
LCKR	volatile uint32_t	0
AFR	volatile uint32_t [2]	[2]
BRR	volatile uint32_t	0
ASCR	volatile uint32_t	0
GPIOC	GPIO_TypeDef *	0x48000800
MODER	volatile uint32_t	4093640703
OTYPER	volatile uint32_t	0
OSPEEDR	volatile uint32_t	0
PUPDR	volatile uint32_t	0
IDR	volatile uint32_t	8192
ODR	volatile uint32_t	0
BSRR	volatile uint32_t	0
LCKR	volatile uint32_t	0
AFR	volatile uint32_t [2]	[2]
BRR	volatile uint32_t	0
ASCR	volatile uint32_t	0
Add new expression		

```
InClassActivity1B [STM32 C/C++ Application]  [pid: 29812]


Verifying ...


Download verified successfully
```

With Press:



```
InClassActivity1B [STM32 C/C++ Application]
  InClassActivity1B.elf [cores: 0]
    Thread #1 [main] 1 [core: 0] (Running : User Request)
  arm-none-eabi-gdb (14.2.90.20240526)
  ST-LINK (ST-LINK GDB server)
```

```c
#define my_led GPIO_PIN_5
#define mybutton GPIO_PIN_13

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

int main(void)
{

  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();

  HAL_GPIO_WritePin(GPIOA, my_led, GPIO_PIN_RESET);

  // this function just turns on the pin when the butt
  while (1) {
      if (HAL_GPIO_ReadPin(GPIOC, mybutton) == 0) {
          // if the button it NOT pressed
          HAL_GPIO_WritePin(GPIOA, my_led, GPIO_PIN_SE
          // set the LED
      } else {
          HAL_GPIO_WritePin(GPIOA, my_led, GPIO_PIN_RE
          // else reset the pin
      }
  }
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output vol
  */
  if (HAL_PWREx_ControlVoltageScaling(PWR_REGULATOR_VO
  {
    Error_Handler();
  }

  /** Initializes the RCC Oscillators according to the
  * in the RCC OscInitTypeDef structure.
```

Expression	Type	Value
GPIOA	GPIO_TypeDef *	0x48000000
MODER	volatile uint32_t	2885679103
OTYPER	volatile uint32_t	0
OSPEEDR	volatile uint32_t	201326592
PUPDR	volatile uint32_t	1677721600
IDR	volatile uint32_t	49184
ODR	volatile uint32_t	32
BSRR	volatile uint32_t	0
LCKR	volatile uint32_t	0
AFR	volatile uint32_t [2]	[2]
BRR	volatile uint32_t	0
ASCR	volatile uint32_t	0
GPIOC	GPIO_TypeDef *	0x48000800
MODER	volatile uint32_t	4093640703
OTYPER	volatile uint32_t	0
OSPEEDR	volatile uint32_t	0
PUPDR	volatile uint32_t	0
IDR	volatile uint32_t	0
ODR	volatile uint32_t	0
BSRR	volatile uint32_t	0
LCKR	volatile uint32_t	0
AFR	volatile uint32_t [2]	[2]
BRR	volatile uint32_t	0
ASCR	volatile uint32_t	0
Add new expression		

```
InClassActivity1B [STM32 C/C++ Application]  [pid: 29812]


Verifying ...


Download verified successfully
```

Lab resources:

ST Reference Documentation:

STM32L4xxxx Reference Manual	RM0351
STM32L476xx Datasheet	DS10198
STM32L4A6xx Datasheet	DS11584
STM32L496xx Datasheet	DS11585
NUCLEO-L476RG Users Manual	UM1724
NUCLEO-L4x6ZG Users Manual	UM2179
STM32CubeIDE Users Manual	UM2609