

جامعة حلب  
كلية الهندسة الكهربائية والإلكترونية  
قسم هندسة التحكم والأتمتة  
مخبر التحكم

## مقرر المتحكمات المصغرة الجلسة السادسة

السنة الرابعة ميكاترونيك

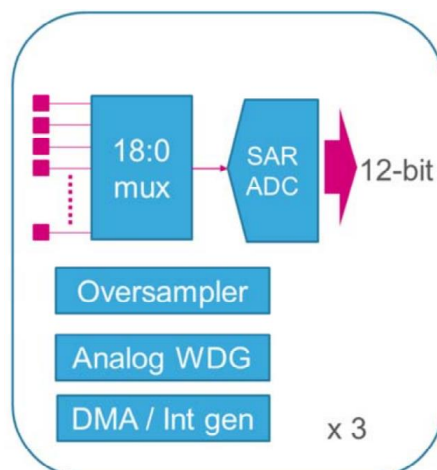
2023/2202

## الغاية من الجلسة

- 1- التعرف على المبدلات التشابيهية الرقمية في متحكمات STM32
- 2- أنماط عمليات التحويل ADC conversion modes
- 3- طرق قراءة المبدل التشابيهي الرقمي
- 4- التطبيق العملي الأول: التحكم بشدة إضاءة ليد موصول على أحد أقطاب الـ PWM من خلال مقاومة ضوئية موصولة على أحد أقطاب الدخل التشابيهي باستخدام نمط الـ Polling باستخدام البورد التطويري
- 5- التطبيق العملي الثاني: إعادة التطبيق السابق باستخدام نمط المقاطعة Interrupt
- 6- التطبيق العملي الثالث: مراقبة درجة حرارة الغرفة وعرضها على شاشة LCD

## 1- التعرف على المبدلات التشابيهية الرقمية في متحكمات STM32:

المبدلات التشابيهية الرقمية عبارة عن دارات الكترونية تقوم بتحويل الجهد التشابيهي على دخلها إلى قيمة رقمية بالنظام الثنائي مقابلة لمستوى الجهد، فبمجرد قرح المبدل التشابيهي الرقمي يبدأ بأخذ العينات samples ويقوم بعملية تدعى التكميم ليقابل كل مستوى من الجهد بما يناسبه من القيم الرقمية. تحتوي متحكمات STM32G0 على مبدل تشابيهي رقمي وحيد من نوع Successive approximation ADC(SAR) بدقة 12بت وما يقارب الـ 19 قناة للمبدل



تسمح المبدلات التشابيهية الرقمية للمتحكم STM32G0 باستقبال القيم التشابيهية القادمة من الحساسات، حيث تقوم بتحويلها إلى القيم الرقمية المقابلة لها، فللمبدل ما يقارب الـ 19 قناة تحويل أي 19 دخل تشابيهي للمتحكم بإمكانه استقبال القيم التشابيهية من خلالها

يتم حساب جهد الدخل التشابيهي من خلال العلاقة التالي:

$$V_{in} = ADC_{Res} * (V_{ref}/4096)$$

حيث :

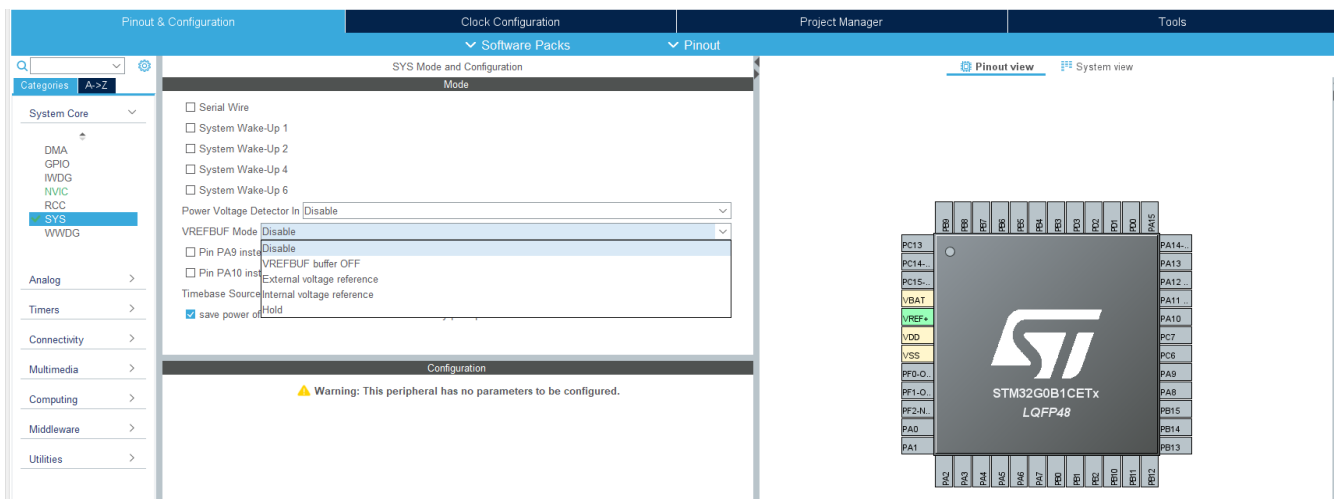
$ADC_{Res}$  : القيمة الرقمية الناتجة عن عملية التحويل التشابيهي الرقمي

الجهد المرجعي  $V_{ref}$ 

Features	Description
Input channel	Up to 16 external (GPIOs) and 3 internal channels
Type of conversion	12-bit successive approximation
Conversion time	400 ns, 2.5 Msamples/s (when $f_{ADC\_CLK} = 35\text{ MHz}$ , 12 bits)
Functional mode	Single, Continuous, Scan, and Discontinuous
Triggers	Software or external trigger (Timers & IOs)
Special functions	Analog watchdogs, Hardware oversampling, and Self-calibration
Data processing	Interrupt generation and DMA requests
Low-power modes	Wait, Auto-off, and Power-down

هناك عدة احتمالات ممكنة للجهد المرجعي:

1. Disabled : وهو الافتراضي وفي هذه الحالة يكون الجهد المرجعي متصل داخلياً مع VDDA وهو نفسه VDD أي جهد تغذية المتحكم 3.3v
2. Buffer off : في هذه الحالة يكون الجهد المرجعي موصول داخلياً مع الـ VSSA
3. External Voltage reference : من خلال وصل جهد مرجعي خارجي إلى القطب  $V_{ref}$
4. Internal Voltage reference : استخدام مولد الجهد المرجعي الداخلي: يوجد داخل متحكمات STM32 مولد جهد مرجعي مدمج بداخلها يقوم بتوليد جهد مرجعي ثابت ومستقر حتى عند تغذيته من بطارية ويمكن استخدامه مع المبدل التشابهي الرقمي ADC والمبدل الرقمي التشابهي DAC ، و يعطي في خرجه إما 2.5v أو 1.8v أو 1.5v ، كما يمكنه أن يغذي أحمال خارجية باستقرار تيار لا يتجاوز الـ 4mA .



عند استخدام مولد الجهد المرجعي الداخلي يجب وصل مكثفات على القطب  $V_{ref+}$  وفي هذه الحالة لن تحتاج لوصلة دائرة خارجية لتوليد الجهد المرجعي

## المعايرة الذاتية Self-calibration:

يوفر المبدل خاصية المعايرة الذاتية والتي تقلل بشكل كبير الأخطاء الناتجة عن تغيرات مكثف الشحن الداخلي internal capacitor، ونقوم عادة في بداية الكود باستخدام دالة من دوال HAL للقيام بعملية المعايرة:

```
HAL_ADCEx_Calibration_Start(&hadc1);
```

## سرعة التحويل Conversion speed:

يحتاج المبدل التشابهي الرقمي على الأقل 1.5 clock cycles لأخذ العينات و 12.5 clock cycles للتحويل من أجل دقة 12 بت، بمعنى آخر ومن أجل تردد الساعة الأعظمي للمبدل 35MHZ يمكن أن تصل سرعة أخذ العينات إلى 2.5mega samples/s

## طرق قراءة المبدل التشابهي الرقمي ADC conversion modes:

يوجد ثلاث طرق رئيسية لقراءة الـ ADC هي:

1. **Polling method**: تعتبر الطريقة الأسهل في كتابة الكود لقراءة القيمة القادمة من إحدى القنوات التشابهية، ولكنها ليست الأكثر فعالية، حيث علينا أن نبدأ بعملية التحويل وتوقف الـ CPU عن تنفيذ الكود وتنتظر لحين الانتهاء من عملية التحويل حينها يمكن للـ CPU استكمال تنفيذ الكود الرئيسي.

2. **The interrupt Method**: تعتبر هذه الطريقة طريقة فعالة لاستخدام المبدل التشابهي الرقمي، حيث نقوم بقدرح المبدل فقط ويمكن للـ CPU أن تستكمل تنفيذ الكود والمهام المطلوبة منها لحين انتهاء عملية التحويل عندها سيقوم المبدل بطلب مقاطعة وستتوجه الـ CPU لبرنامج خدمة المقاطعة وتحتفظ بناتج عملية التحويل ليتم معالجته.

### 3. DMA Method

2- **التطبيق العملي الأول**: سنقوم في هذا التطبيق بالتحكم بشدة إضاءة ليد موصول على أحد أقطاب الـ PWM (القطب PA0) من خلال مقاومة ضوئية موصولة على أحد أقطاب الدخل التشابهي سنقوم بتنفيذ المشروع وفقاً للتسلسل التالي:

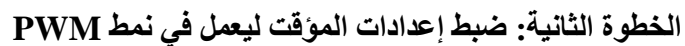
- ضبط تردد ساعة المتحكم
- ضبط قطب الدخل التشابهي (PA7) في نمط التحويل لمرة واحدة Single conversion mode حيث سنربط مقاومة ضوئية معه.
- ضبط الـ timer2 في نمط PWM على القناة CH1 (وسنربط معه ليد).

سنقوم بتنفيذ المشروع بطريقتي Polling, interrupt، حيث سنقوم في البداية بقراءة القيمة القادمة من المقاومة المتغيرة الموجودة على القناة CH7 للمبدل التشابهي ومن ثم سنقوم بإسناد هذه القيمة لمسجل المؤقت CCR والذي من خلاله يتم تحديد دورة التشغيل dutycycle والتي تحدد شدة إضاءة الليد.

## الطريقة الأولى: استخدام نمط الـ Polling:

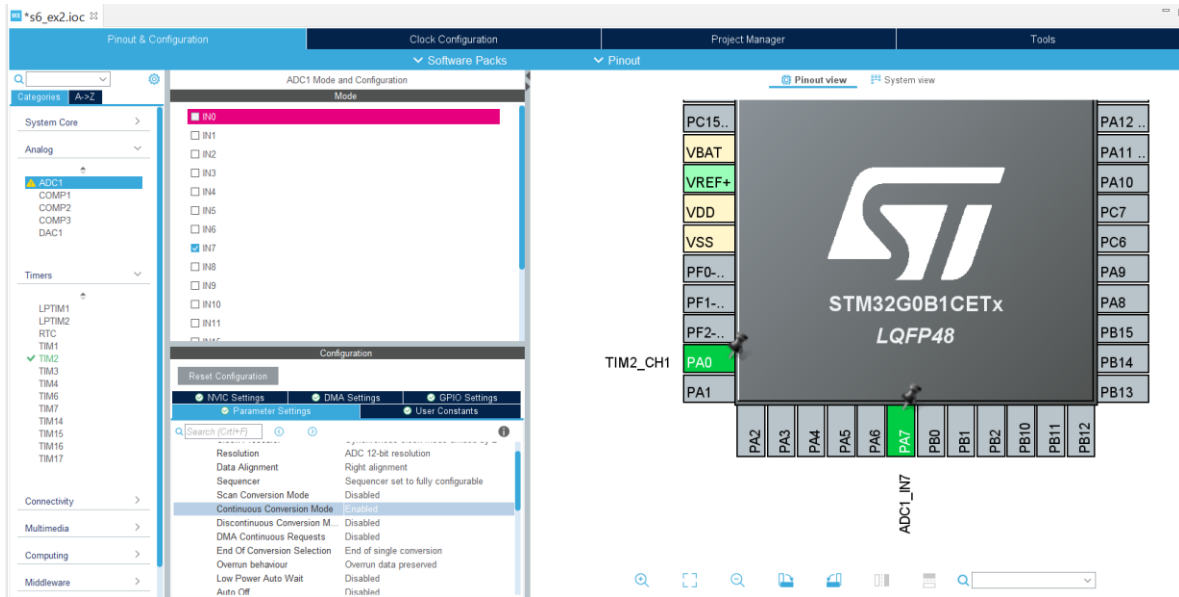
سنقوم بضبط الإعدادات من خلال أداة CubeMx المدمجة داخل بيئة STM32CubeIDE وفقاً للخطوات التالية:

**الخطوة الأولى:** قم بفتح برنامج STM32CubeIDE ومن ثم قم بإنشاء مشروع جديد من نافذة File ثم New ثم STM32Project ثم قم باختيار المتحكم المصغر أو من خلال اختيار اسم اللوحة المستخدمة وهي في حالتنا STM32G0B1CE كما في الشكل التالي:

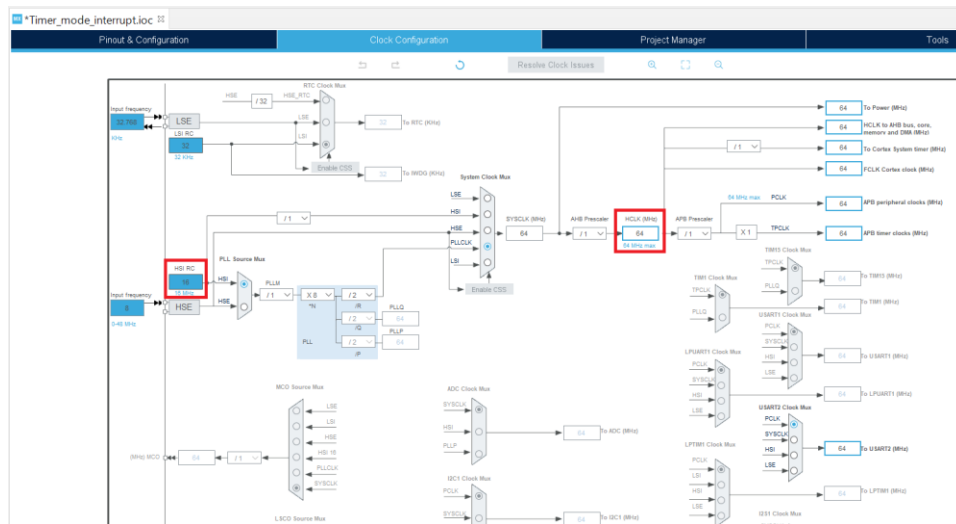


The screenshot displays the STM32CubeMX software interface. The main window is titled "TIM2 Mode and Configuration". It has two tabs: "Mode" and "Configuration". The "Mode" tab is active, showing a list of settings for the TIM2 timer. The settings include: Slave Mode (Disable), Trigger Source (Disable), Clock Source (Internal Clock), Channel1 (PWM Generation CH1), Channel2 (Disable), Channel3 (Disable), Channel4 (Disable), Combined Channels (Disable), and Use ETR as Clearing Source (Disable). The "Configuration" tab is also visible, showing settings for NVIC, DMA, and GPIO. The "Pinout" view is shown on the right, highlighting the PA0 pin.

**الخطوة الثالثة:** قم بضبط إعدادات المبدل التشابهي الرقمي ADC على القناة التشابهية السابعة CH7 بالشكل التالي:



### الخطوة الرابعة: ضبط تردد ساعة المتحكم



### الخطوة الخامسة: توليد الكود بناءً على الإعدادات التي تم ضبطها من خلال ctrl+s

يصبح الكود النهائي بالشكل التالي:

```
#include "main.h"
```

```
ADC_HandleTypeDef hadc1;
TIM_HandleTypeDef htim2;
```

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

تعريف المبدل التشابهي ADC1  
تعريف المؤقت TIM2

```

static void MX_ADC1_Init(void);
static void MX_TIM2_Init(void);

int main(void)
{
    uint16_t AD_RES = 0;

    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
    MX_ADC1_Init();
    MX_TIM2_Init();

    HAL_TIM_PWM_Start(&htim2,
TIM_CHANNEL_1);
    // Calibrate The ADC On Power-Up For Better
Accuracy
    HAL_ADCEX_Calibration_Start(&hadc1);

    while (1)
    {
        // Start ADC Conversion
        HAL_ADC_Start(&hadc1);
        // Poll ADC1 Perihperal & TimeOut = 1mSec
        HAL_ADC_PollForConversion(&hadc1, 1);
        // Read The ADC Conversion Result & Map It
To PWM DutyCycle
        AD_RES = HAL_ADC_GetValue(&hadc1);

        TIM2->CCR1 = (AD_RES<<4);

        HAL_Delay(1);
    }
}

```

التصريح عن الدوال المسؤولة عن تهيئة ساعة المتحكم ، أقطاب الدخل/ الخرج ، تهيئة المبدل التشابهي الرقمي ADC1 ، تهيئة المؤقت TIM2

التصريح عن متحول AD\_RES لنخزن فيه لاحقاً القيمة الناتجة عن المحول التشابهي الرقمي

استدعاء الدوال المسؤولة عن تهيئة ساعة المتحكم ، أقطاب الدخل/ الخرج ، تهيئة المبدل التشابهي الرقمي ADC1 ، تهيئة المؤقت TIM2

بدء عمل المؤقت في نمط الـ PWM وعلى القناة الأولى

بدء المعايرة الذاتية للمبدل التشابهي الرقمي adc1

إعطاء إشارة البدء للمبدل التشابهي الرقمي ليكون جاهز فيما بعد لإجراء عمليات التحويل

طلب عملية تحويل من المبدل التشابهي الرقمي

إسناد ناتج عملية التحويل إلى المتغير AD\_RES

إسناد قيمة المتغير AD\_RES التي تعبر عن قيمة المقاومة الضوئية ، إلى المسجل CCR1 من المؤقت TIM2 ، المسؤول عن دورة التشغيل dutycycle

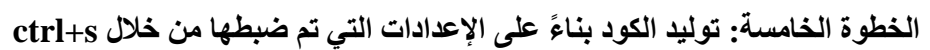
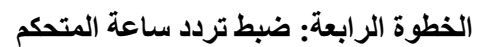
إجراء تأخير زمني بمقدار 1msec

### التطبيق العملي الثاني: إعادة التطبيق الأول ولكن باستخدام نمط الـ Interrupt:

سنقوم بضبط الإعدادات من خلال أداة CubeMx المدمجة داخل بيئة STM32CubeIDE وفقاً للخطوات التالية:

الخطوة الأولى والثانية هي نفس الخطوات الموجودة في الطريقة الأولى

**الخطوة الثالثة:** ضبط إعدادات المبدل التشابهي بنفس الإعدادات السابقة ، فقط سنفعّل المقاطعة الخاصة بالمبدل



التصريح عن متحول عام (لأننا  
سنستخدمه في الدالة الرئيسية وفي  
برنامج خدمة المقاطعة) AD\_RES



<pre> ADC_HandleTypeDef hadc1; TIM_HandleTypeDef htim2;  void SystemClock_Config(void); static void MX_GPIO_Init(void); static void MX_ADC1_Init(void); static void MX_TIM2_Init(void);  int main(void) {     HAL_Init();     SystemClock_Config();     MX_GPIO_Init();     MX_ADC1_Init();     MX_TIM2_Init();      HAL_TIM_PWM_Start(&amp;htim2, TIM_CHANNEL_1);     // Calibrate The ADC On Power-Up For Better Accuracy     HAL_ADCEX_Calibration_Start(&amp;hadc1);     // Start ADC Conversion     HAL_ADC_Start_IT(&amp;hadc1);      while (1)     {          // Update The PWM Duty Cycle With Latest ADC         Conversion Result          TIM2-&gt;CCR1 = (AD_RES&lt;&lt;4);         HAL_Delay(1);     }      void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef*     hadc)     {          // Read &amp; Update The ADC Result         AD_RES = HAL_ADC_GetValue(&amp;hadc1);     } </pre>	<p>لنخزن فيه لاحقاً القيمة الناتجة عن المحول التناظيري الرقمي</p> <p>تعريف المبدل التناظيري ADC1 تعريف المؤقت TIM2</p> <p>التصريح عن الدوال المسؤولة عن تهيئة ساعة المتحكم ، أقطاب الدخل/الخرج ، تهيئة المبدل التناظيري الرقمي ADC1 ، تهيئة المؤقت TIM2</p> <p>تهيئة مكتبة HAL استدعاء الدوال المسؤولة عن تهيئة ساعة المتحكم ، أقطاب الدخل/الخرج ، تهيئة المبدل التناظيري الرقمي ADC1 ، تهيئة المؤقت TIM2</p> <p>بدء عمل المؤقت في نمط الـ PWM وعلى القناة الأولى</p> <p>بدء المعايرة الذاتية للمبدل التناظيري الرقمي adc1</p> <p>إعطاء إشارة البدء للمبدل التناظيري الرقمي ليكون جاهز فيما بعد لإجراء عمليات التحويل</p> <p>إسناد قيمة المتغير AD_RES التي تعبر عن قيمة المقاومة الضوئية ، إلى المسجل CCR1 من المؤقت TIM2 ، المسؤول عن دورة التشغيل dutycycle إجراء تأخير زمني بمقدار 1sec</p> <p>برنامج خدمة المقاطعة والذي يتم استدعاؤه عند إتمام عملية التحويل التناظيري الرقمي</p> <p>إسناد ناتج عملية التحويل إلى المتغير AD_RES (نحصل على ناتج التحويل التناظيري الرقمي من برنامج خدمة المقاطعة)</p>
--	---

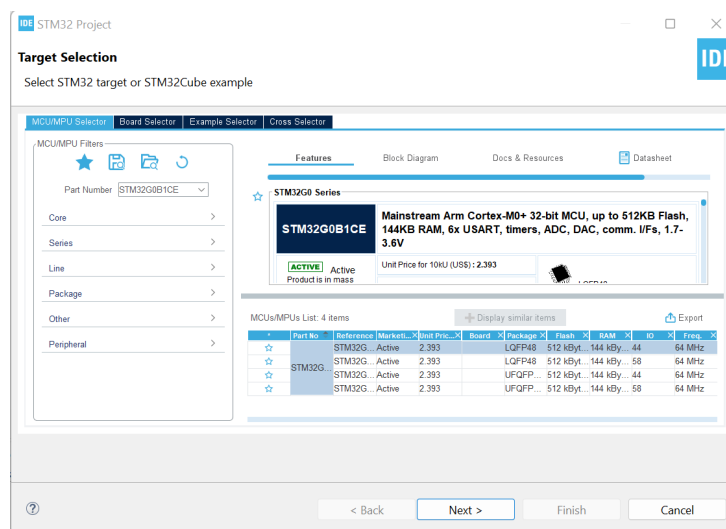
### 3- التطبيق العملي الثالث: سنقوم في هذا التطبيق بمراقبة جهد متغير موصول على أحد أقطاب الدخل التشابهي ADC وعرض القيمة على شاشة lcd.

سنقوم بتنفيذ المشروع وفقاً للتسلسل التالي:

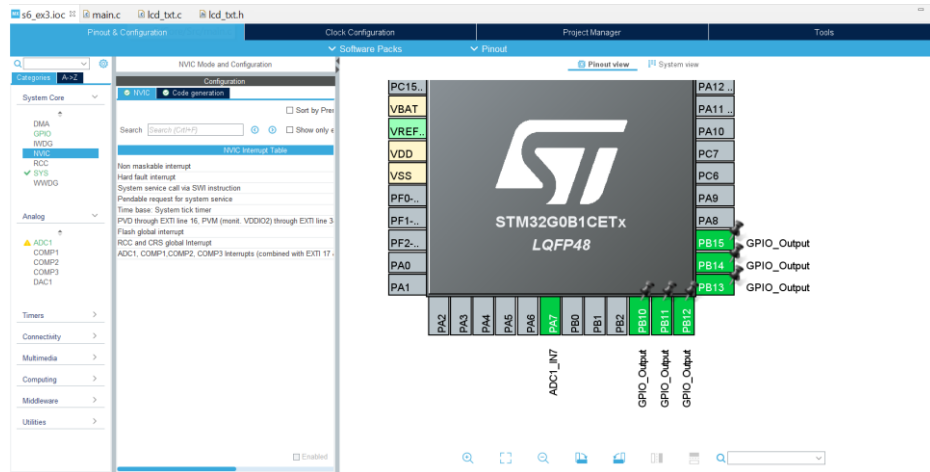
- ضبط تردد ساعة المتحكم
- ضبط قطب الدخل التشابهي ( PA7 ) في نمط المقاطعة .
- إضافة مكتبة الـ lcd إلى الكود

**ضبط إعدادات المشروع:**

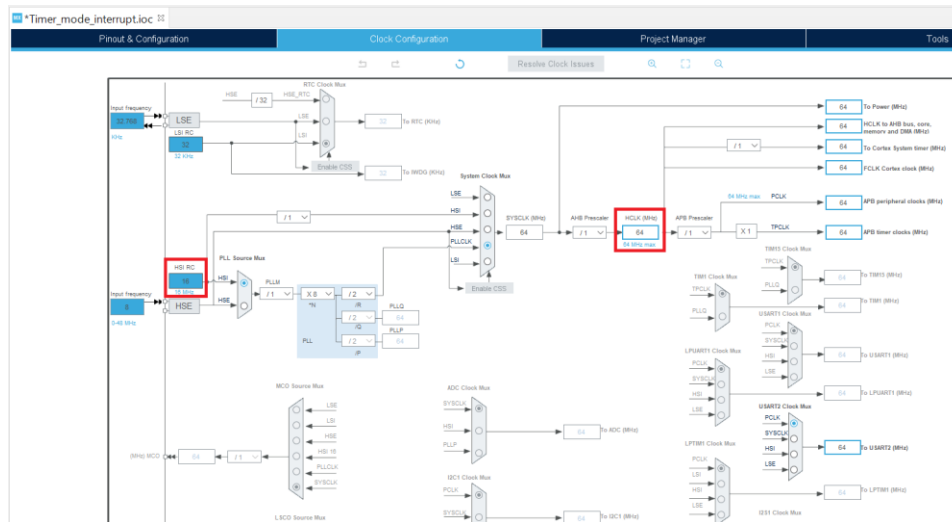
**الخطوة الأولى:** قم بفتح برنامج STM32CubeIDE ومن ثم قم بإنشاء مشروع جديد من نافذة File ثم New ثم STM32Project ثم قم باختيار المتحكم المصغر أو من خلال اختيار اسم اللوحة المستخدمة وهي في حالتنا STM32G0B1CE كما في الشكل التالي:



**الخطوة الثالثة:** قم بضبط إعدادات المبدل التشابهي الرقمي ADC على القناة التشابهيّة السابعة CH7 ، اختر الأقطاب PB10:PB15 لضبطها كأقطاب خرج ليتم وصل شاشة الـ lcd معها، بالشكل التالي:



### الخطوة الرابعة: ضبط تردد ساعة المتحكم



الخطوة الخامسة: توليد الكود بناءً على الإعدادات التي تم ضبطها من خلال ctrl+s، ثم إضافة مكتبة الـ lcd والتعليمات الخاصة ببدء المبدل التشابهي ومعايرته و أخذ القراءات من المقاومة المتغيرة وعرضها على شاشة الـ

lcd

```

#include "main.h"
#include "lcd_txt.h"
ADC_HandleTypeDef hadc1;
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_ADC1_Init(void);
int main(void)
{
  uint16_t adc_value= 0;
  uint16_t voltage=0;

  unsigned char buffer[16];
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_ADC1_Init();
  lcd_init();
  lcd_clear();
  // Calibrate The ADC On Power-Up For Better Accuracy
  HAL_ADCEX_Calibration_Start(&hadc1);
  while (1)
  {
    // Start ADC Conversion
    HAL_ADC_Start(&hadc1);
    // Poll ADC1 Perihperal & TimeOut = 1mSec
    HAL_ADC_PollForConversion(&hadc1, 1);
    // Read The ADC Conversion Result & Map It To PWM DutyCycle
    adc_value = HAL_ADC_GetValue(&hadc1);
    voltage= adc_value*(0.8057); //3300/4096

    sprintf(buffer, "ADC=%04d", adc_value);
    lcd_puts(0,0,buffer);
    sprintf(buffer, "mVolt=%04d", voltage);
    lcd_puts(1,0,buffer);

    HAL_Delay(100);
  }
}

```