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

مقرر المتحكمات المصغرة

الجلسة الثانية

السنة الرابعة ميكاترونيك

2023/2022

مدرس المقرر: _____

الغاية من الجلسة

1- التعرف على التوابع المستخدمة من مكتبة HAL للتحكم بالمخارج الرقمية في متحكم STM32
2- بناء تطبيق للتحكم بالمخارج الرقمية باستخدام متحكمات stm32 ومكتبةHAL
3- التحكم بالمخارج الرقمية للمتحكم STM32 دون استخدام مكتبة HAL
4- إعادة التطبيق السابق دون استخدام مكتبة HAL
5- برمجة أقطاب الدخل في متحكمات stm32
6- التعريف بالدوال المستخدمة من مكتبة HAL للتحكم بالمداخل الرقمية في متحكم STM32
7- بناء تطبيق باستخدام متحكمات stm32 ومكتبةHAL
8- التعامل مع أقطاب الدخل في متحكمات stm32 بدون استخدام مكتبة HAL
9- إعادة التطبيق السابق دون استخدام مكتبة HAL
10- بناء عدة تطبيقات

1- التعرف على التوابع المستخدمة من مكتبة HAL للتحكم بالمخارج الرقمية في متحكم STM32:

تم تصميم مكتبة HAL كي تستخلص عناوين الوحدات الطرفية المختلفة من الذاكرة، وإعطاء طريقة ثابتة وسهلة للتعامل مع الوحدات المختلفة، دون الحاجة للتعامل مع المسجلات Registers المختلفة وللتسهيل على المبرمجين.

- **لإعطاء واحد منطقي set أو صفر منطقي reset لقطب محدد من أي منفذ من منافذ المتحكم نقوم باستخدام التابع التالي من مكتبة HAL:**

```
void HAL_GPIO_WritePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState);
```

اسم المنفذ المراد التحكم بأحد أقطابه على سبيل المثال GPIOA

رقم القطب المراد التحكم به على سبيل المثال GPIO_PIN_5

لإعطاء واحد منطقي نكتب GPIO_PIN_SET ولإعطاء صفر منطقي نكتب GPIO_PIN_RESET

مثال1: لكتابة واحد منطقي على القطب رقم10 من المنفذ D نستخدم التابع التالي:

```
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_10, 1);
```

مثال2: لكتابة صفر منطقي على القطب رقم5 من المنفذ A نستخدم التابع التالي:

```
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 0);
```

- **لعكس الحالة المنطقية لأحد الأقطاب نستخدم التابع التالي:**

```
HAL_GPIO_TogglePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin);
```

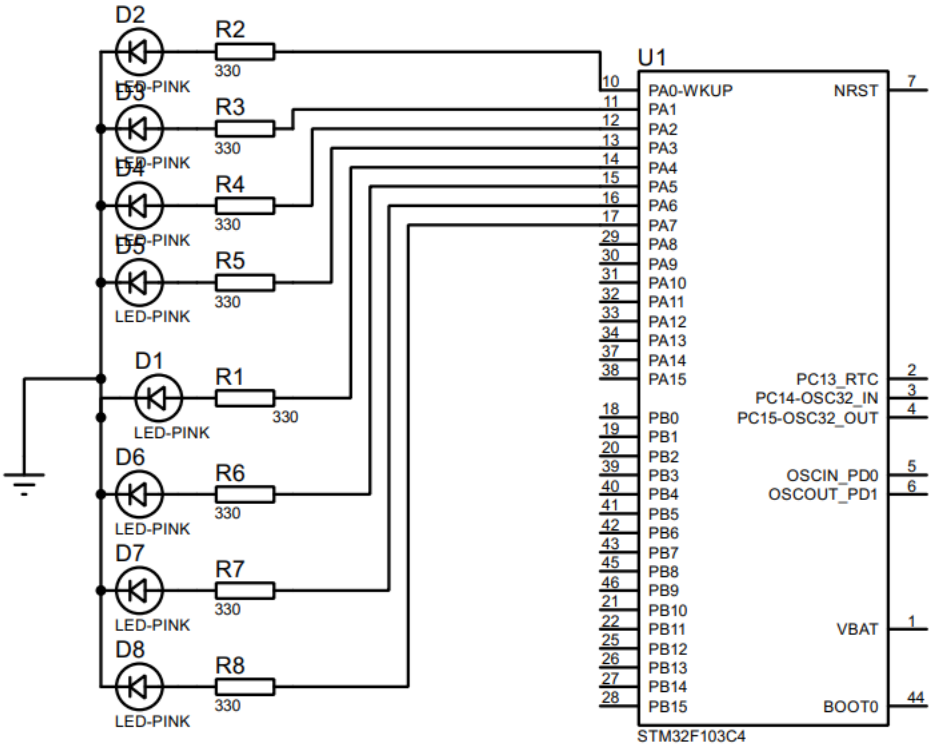
مثال: لعكس الحالة المنطقية للقطب رقم 5 من المنفذ A نستخدم التابع التالي:

HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);

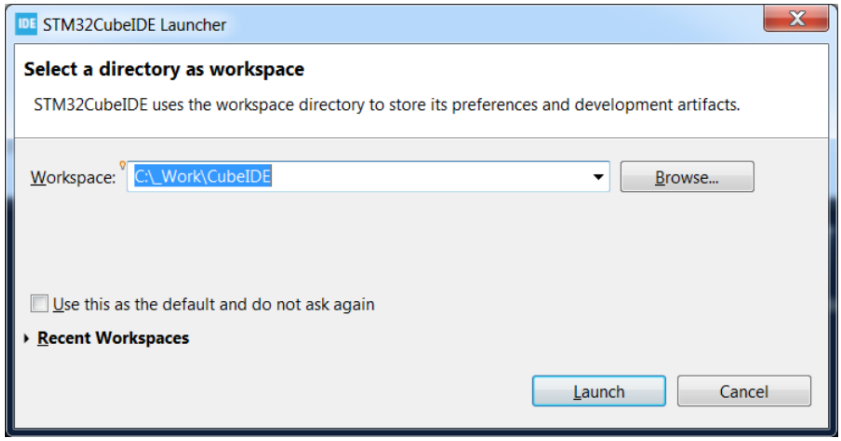
- **لإضافة تأخير زمني بالميللي ثانية نستخدم التابع التالي:**

HAL_Delay(Milliseconds)

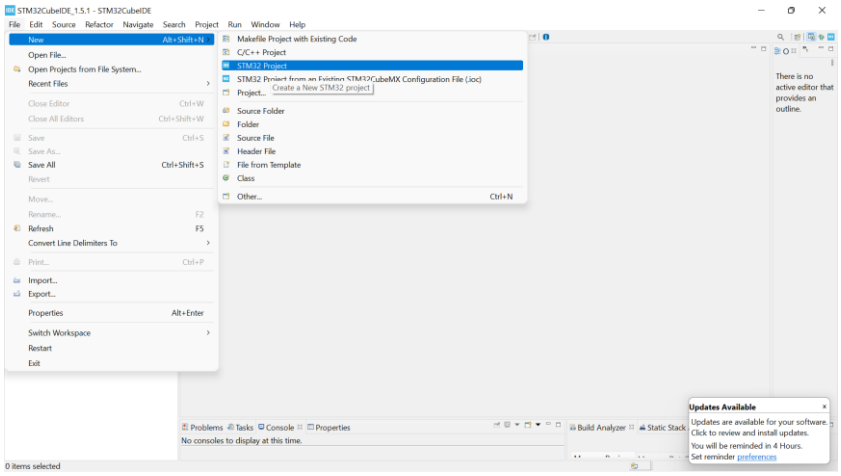
2- التطبيق الأول: تشغيل وإطفاء مجموعة لدات بشكل متسلسل حيث الزمن بين تشغيل الليد والذي يليه 250msec باستخدام بيئة المحاكاة Proteus



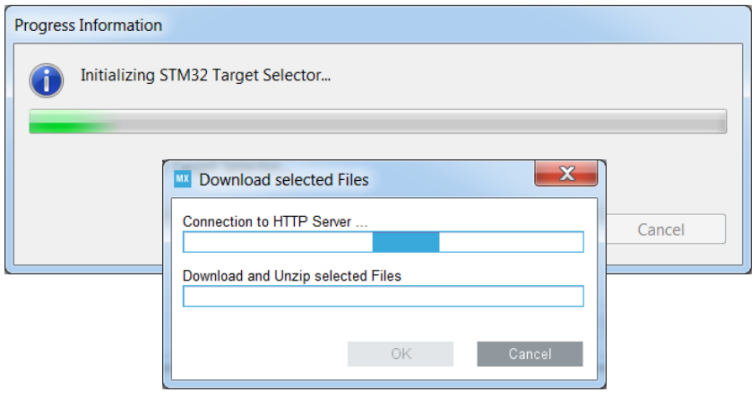
الخطوة الأولى: قم بفتح بيئة STM32cubeIDE



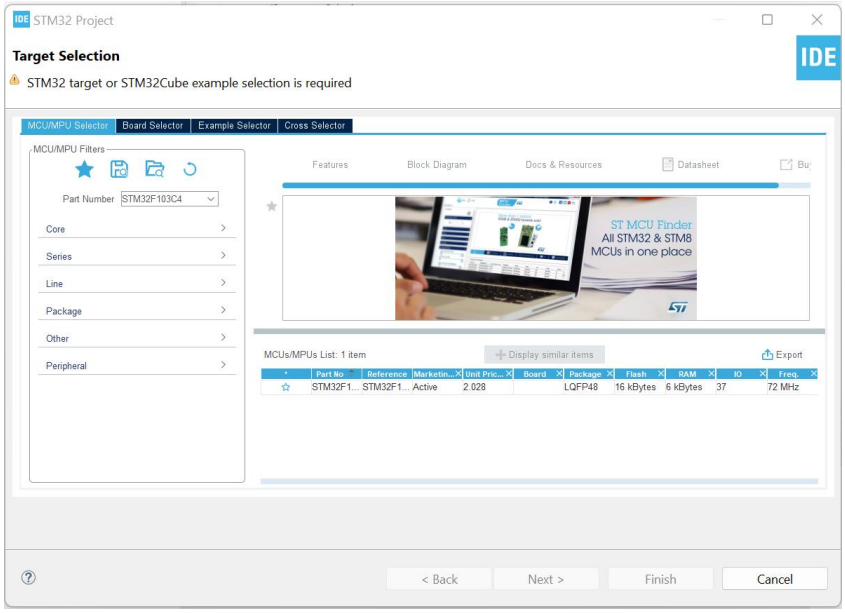
ثم نختار إنشاء مشروع جديد من file… new…. Stm32project كما هو موضح بالشكل التالي:

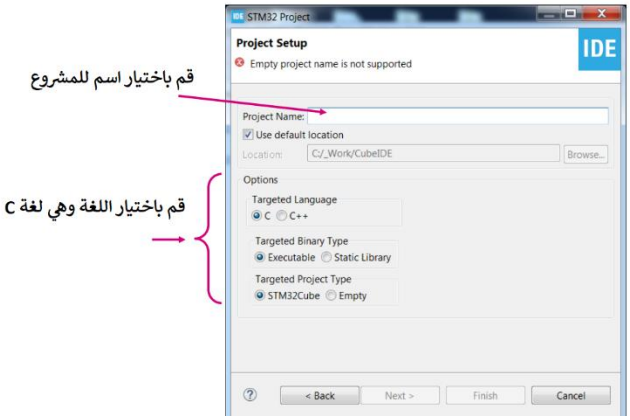


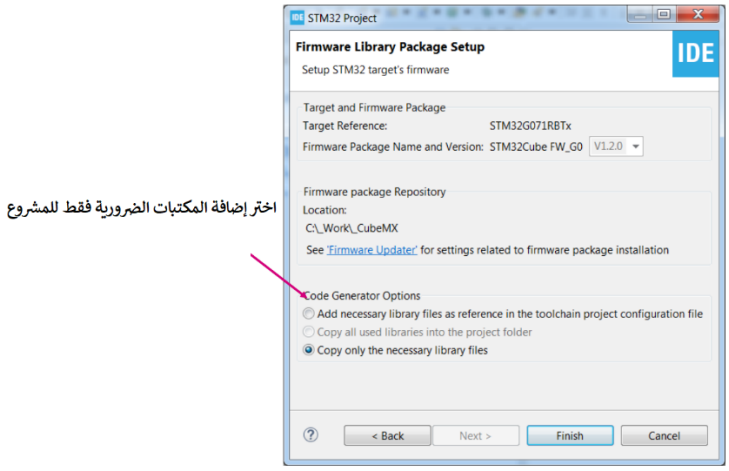
نلاحظ أن البرنامج يبحث عن تحديثات له عبر شبكة الانترنت في حال كان الحاسب متصل بالانترنت:



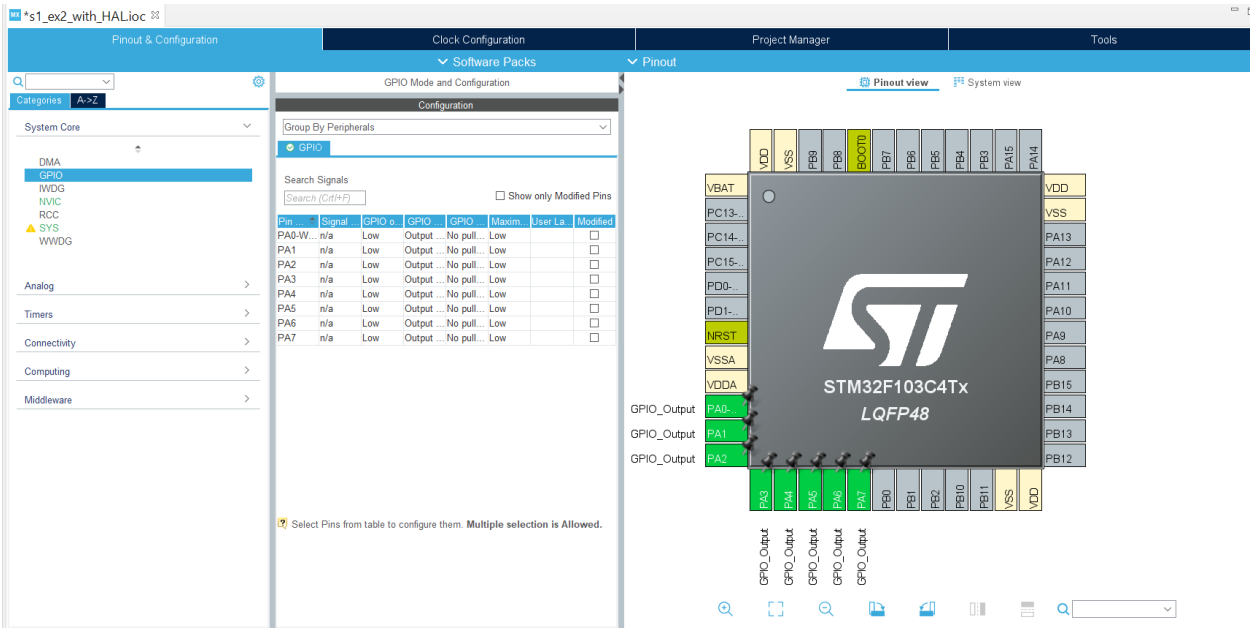
الخطوة الثانية : قم باختيار المتحكم من خلال كتابة اسم المتحكم وهو **STM32F103C4** أو كتابة اسم اللوحة التطويرية في حال استخدام لوحة تطويرية ضمن مربع البحث كما هو موضح بالشكل التالي:



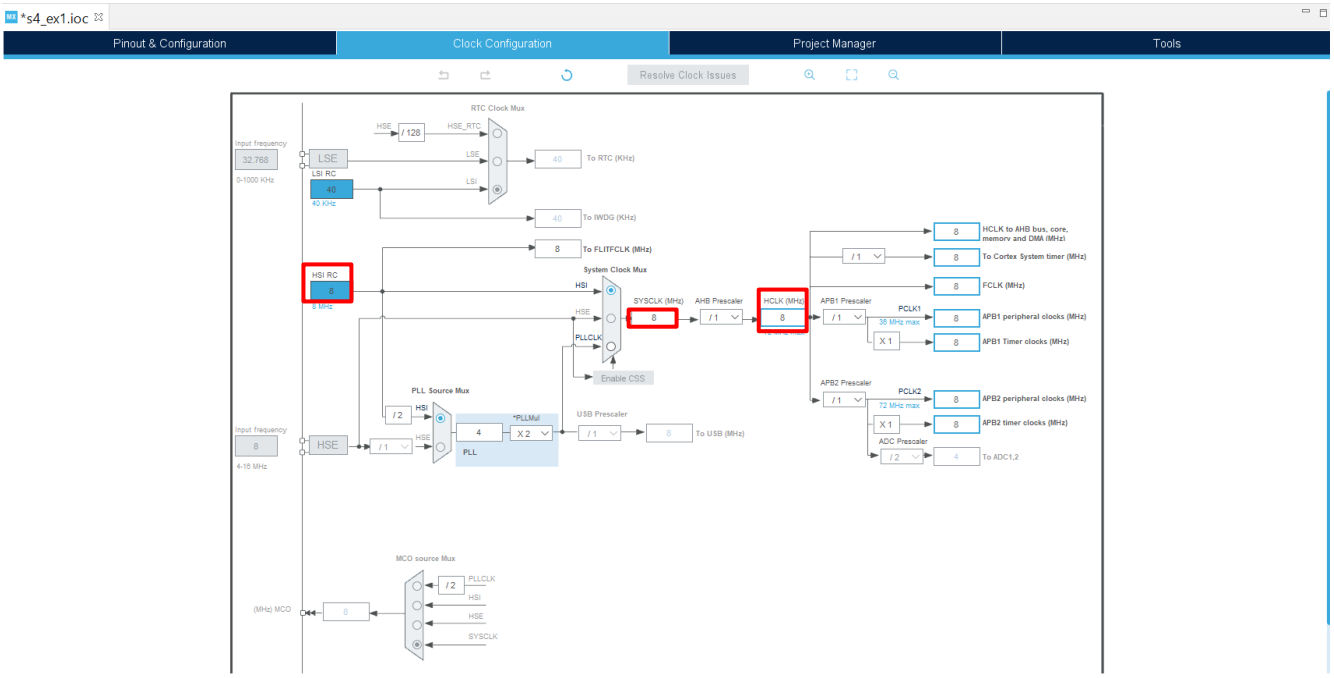




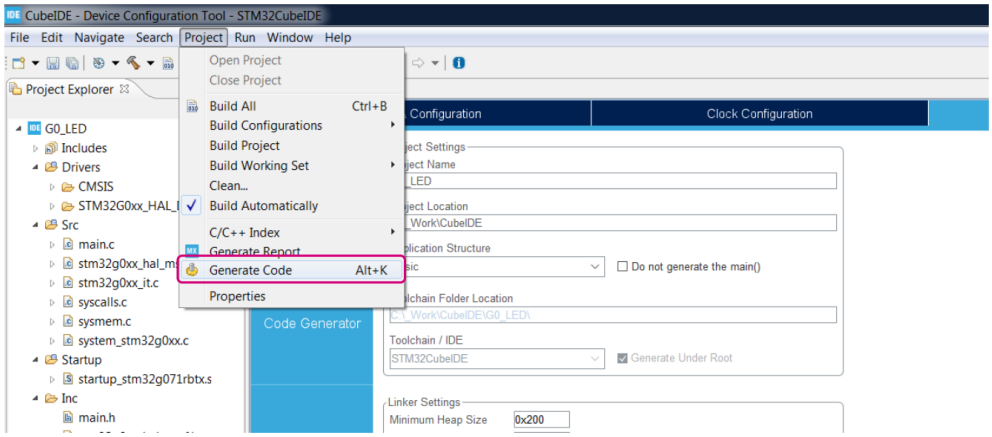
الخطوة الثالثة: قم بتحديد الأقطاب من PA0 حتى PA7 كخرج كما هو موضح بالشكل التالي:



الخطوة الرابعة: قم بضبط تردد الساعة للمتحكم واختر مصدر الساعة الداخلية HSI للمتحكم كما هو موضح بالشكل التالي:



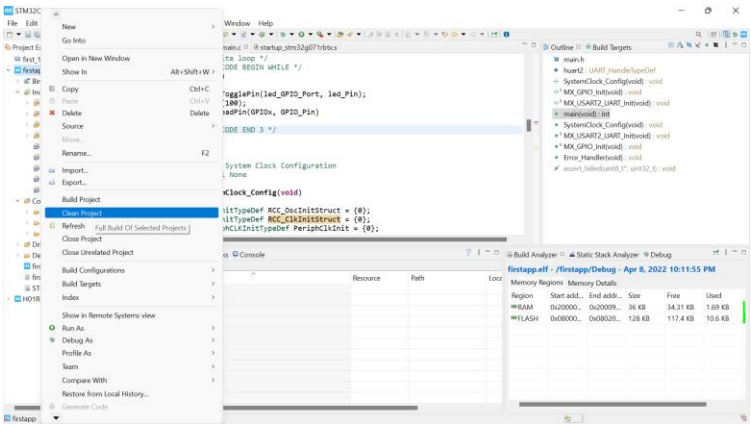
اضغط على Ctrl+s أو من Project...Generate code ، ليتم حفظ المشروع وتوليد الكود وإضافة المكتبات اللازمة، ثم قم بفتح main.c لتعديل الكود بما يناسب مشروعك



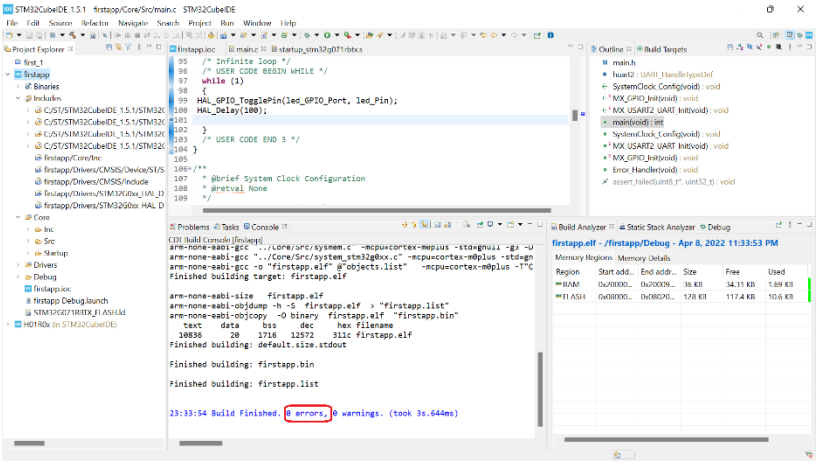
الخطوة الخامسة: نقوم بكتابة الكود المناسب

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
   while (1)
  {
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 1);
        HAL_Delay(250);
        //**************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 1);
        HAL_Delay(250);
        //**************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 1);
        HAL_Delay(250);
        //**************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 1);
        HAL_Delay(250);
        //**************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 1);
        HAL_Delay(250);
        //*************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 1);
        HAL_Delay(250);
        //************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, 1);
        HAL_Delay(250);
       //*************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, 1);
        HAL_Delay(250);
        //************
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, 0);
        HAL_Delay(250); }}
```

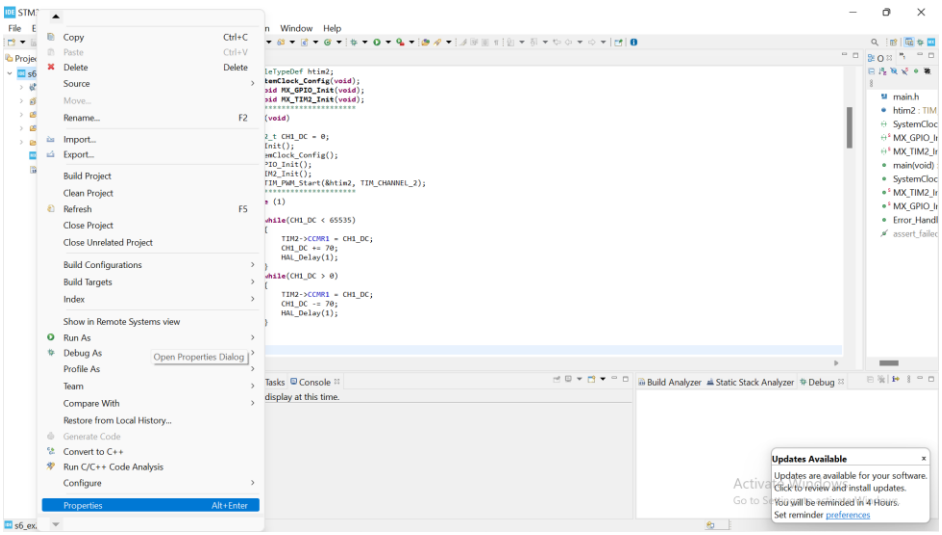
الخطوة السادسة: قم بالنقر بزر الفأرة الأيمن على اسم المشروع ثم اختر Clean project ثم Build project لتتم عملية ترجمة الكود للصيغة الثنائية والتأكد من خلو الكود من الأخطاء اللغوية Syntax errors



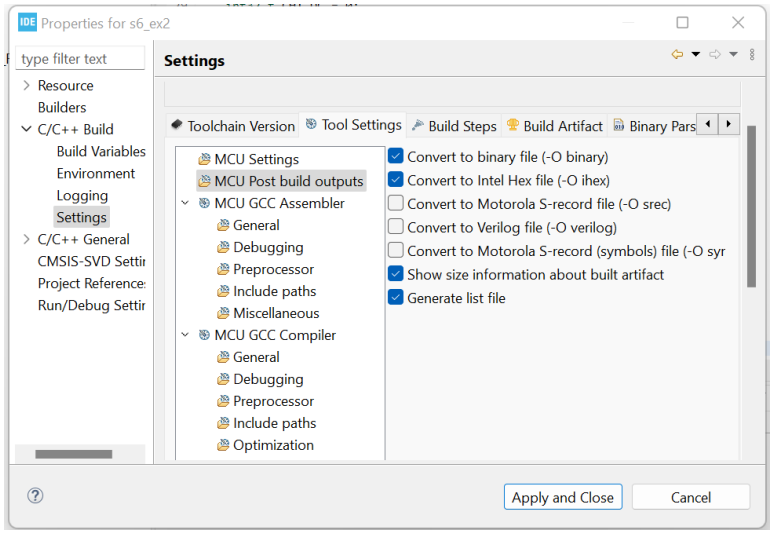
في حال خلو الكود من أخطاء الـ Syntax errors :



في حال لم يتم توليد ملف بامتداد hex. اذهب إلى خصائص المشروع



واختر مايلي:



الخطوة السابعة: ننتقل لرسم الدارة على برنامج proteus : نفتح البرنامج ثم نبدأ بإضافة العناصر اللازمة للتطبيق الذي قمنا به وهي عبارة عن متحكم stm32f103c4 وليدات ومقاومات

لتجريب التطبيق على البورد التجريبي قم بإعادة الخطوات السابقة ولكن اختر في الخطوة الثانية المتحكم STM32G071RB ثم بعد كتابة الكود قم بالضغط على أيقونة الـــ debug لتبدأ دارة ST-Link برفع الكود إلى المتحكم ثم البدء بجلسة debug لمراقبة الكود خطوة بخطوة وتصحيح الأخطاء البرمجية.

-

3- التحكم بالمخارج الرقمية للمتحكم STM32 دون استخدام مكتبة HAL:

هناك مجموعة من المسجلات تستخدم للتحكم بالمخارج الرقمية لمتحكم STM32 سنكتفي فقط بذكر المسجل المسؤول عن عمل set/reset للمنفذ أو لأحد الأقطاب الموجودة فيه (ولمزيد من المعلومات عن باقي المسجلات بإمكانك الرجوع إلى الـ datasheet الخاصة بالمتحكم).

7.4.6 GPIO port output data register (GPIOx_ODR) (x = A..H)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODRy**: Port output data (y = 0..15)
These bits can be read and written by software.
Note: For atomic bit set/reset, the ODR bits can be individually set and reset by writing to the GPIOx_BSRR register (x = A..H).

- نلاحظ أن المسجل GPIOx_ODR عبارة عن مسجل بطول 32bit ، لكن البتات16:31 غير مستخدمة ويجب وضع صفر منطقي في كل منها، أما البتات من 0:15 فهي تعبر عن حالة أقطاب المنفذ المقابل لها، فعلى سبيل المثال لكتابة واحد منطقي على القطب رقم 5 من المنفذ A نكتب:

```
GPIOA->ODR = 0x0020; // Set the Pin PA5
```

- و لكتابة صفر منطقي عليه نكتب:

```
GPIOA->ODR = 0; // Reset the Pin PA5
```

- كما يمكن جعل المنفذ بالكامل بحالة set من خلال كتابة :

```
GPIOA->ODR = 0xFFFF;  // Set the PORTA HIGH
```

- كما يمكن جعل المنفذ بالكامل بحالة reset من خلال كتابة :

```
GPIOA->ODR = 0x0000;  // Reset the PORTA
```

4- إعادة التطبيق الأول دون استخدام مكتبة HAL وباستخدام بيئة المحاكاة proteus

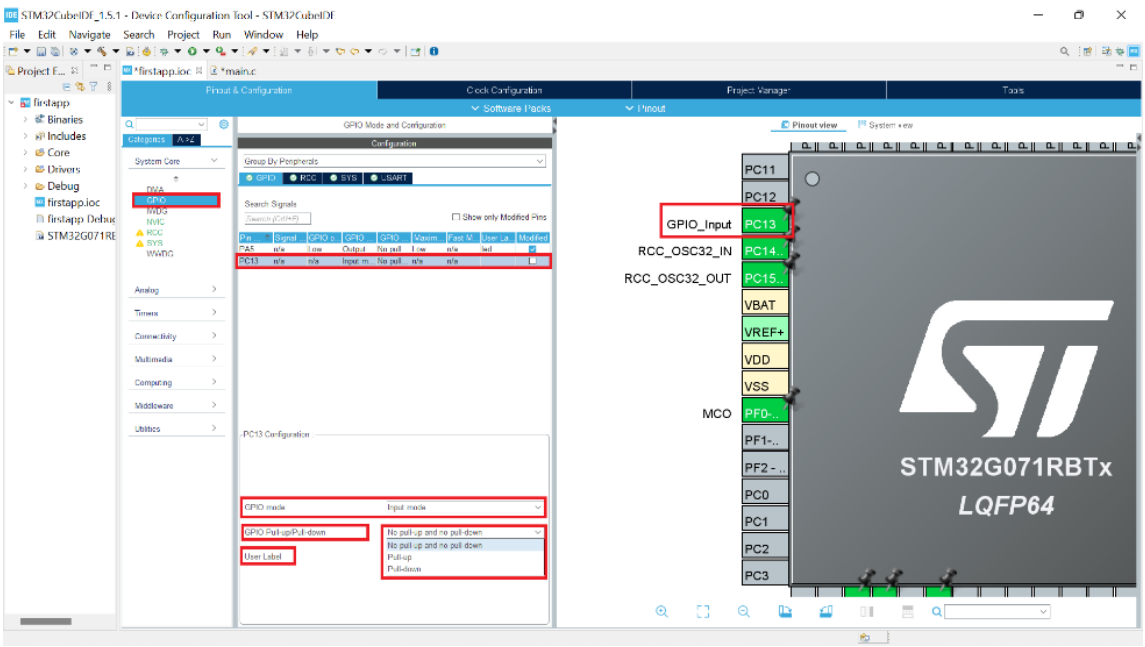
قم بإعادة الخطوات السابقة مع تعديل فقط الكود ليصبح بالشكل التالي:

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
{
    HAL_Init();
    SystemClock_Config();
```

```
  MX_GPIO_Init();
 while (1)
{
      GPIOA->ODR = 0X0001;
         HAL_Delay(250);
    //**************
    GPIOA->ODR = 0X0002;
    HAL_Delay(250);
    //**************
    GPIOA->ODR = 0X0004;
    HAL_Delay(250);
    //**************
    GPIOA->ODR = 0X0008;
    HAL_Delay(250);
    //*************
    GPIOA->ODR = 0X0010;
    HAL_Delay(250);
    //************
    GPIOA->ODR = 0X0020;
    HAL_Delay(250);
    //***********
    GPIOA->ODR = 0X0040;
    HAL_Delay(250);
   //************
    GPIOA->ODR = 0X0080;
    HAL_Delay(250);  }}
```

5- برمجة أقطاب الدخل في متحكمات stm32 :

لجعل أحد الأقطاب قطب دخل رقمي، نقوم بضبط الإعدادات المناسبة من خلال الواجهة الرسومية بالشكل التالي:



حيث نلاحظ وجود ثلاث خيارات للدخل الرقمي وهي:

1- عدم استخدام أي مقاومات للرفع أو الخفض No pull-up and no Pull-down
2- استخدام مقاومة رفع Pull-up
3- استخدام مقاومة خفض Pull-down

وأيضاً لدينا خيار بإعطاء اسم اختياري للقطب.

6- الدوال المستخدمة من مكتبة HAL للتحكم بالمداخل الرقمية في متحكم STM32:

نستخدم التابع التالي لمعرفة حالة الدخل الرقمي على أحد أقطاب المتحكم:

```
HAL_GPIO_ReadPin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin);
```

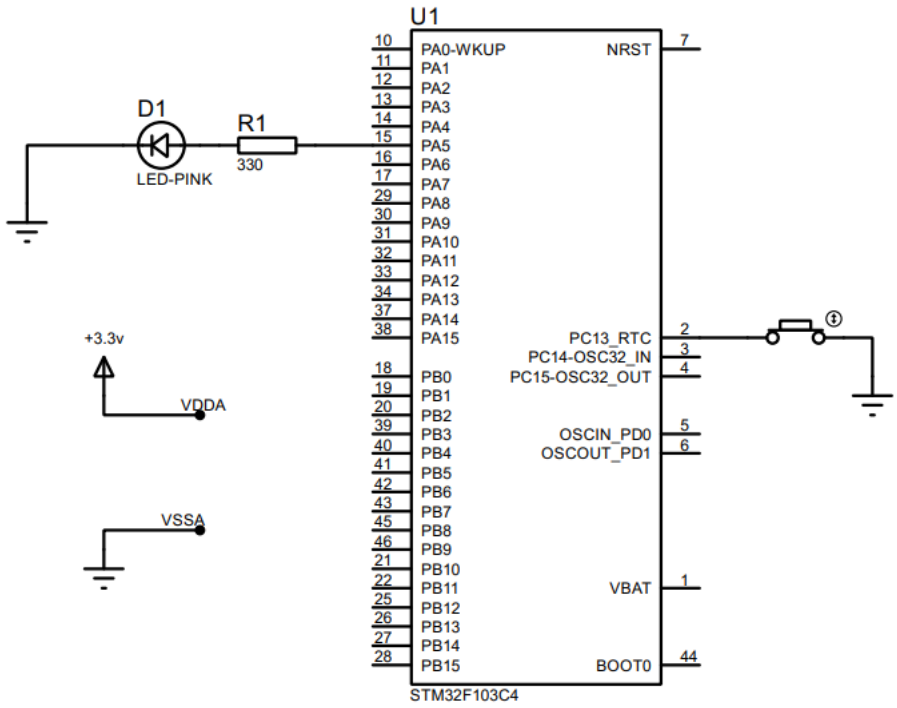
حيث يعيد هذا التابع 0 في حال كانت الحالة المنطقية للقطب في حالة جهد منخفض ويعيد 1 في حال كانت الحالة المنطقية للقطب في حالة جهد مرتفع.

مثال: لقراءة حالة الدخل الرقمي على القطب رقم 13 من المنفذ E نستخدم التابع التالي:

```
HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_13);
```

7- التطبيق الثاني: إضاءة ليد من خلال مفتاح لحظي باستخدام متحكمات stm32 ومكتبةHAL وباستخدام بيئة المحاكاة Proteus

مطلوب كتابة الكود المناسب بحيث يضيء الليد LD4 الموصول على القطب رقم 5 من المنفذ A عند الضغط على المفتاح B1 الموصول على القطب رقم 13 من المنفذ C من المتحكم.

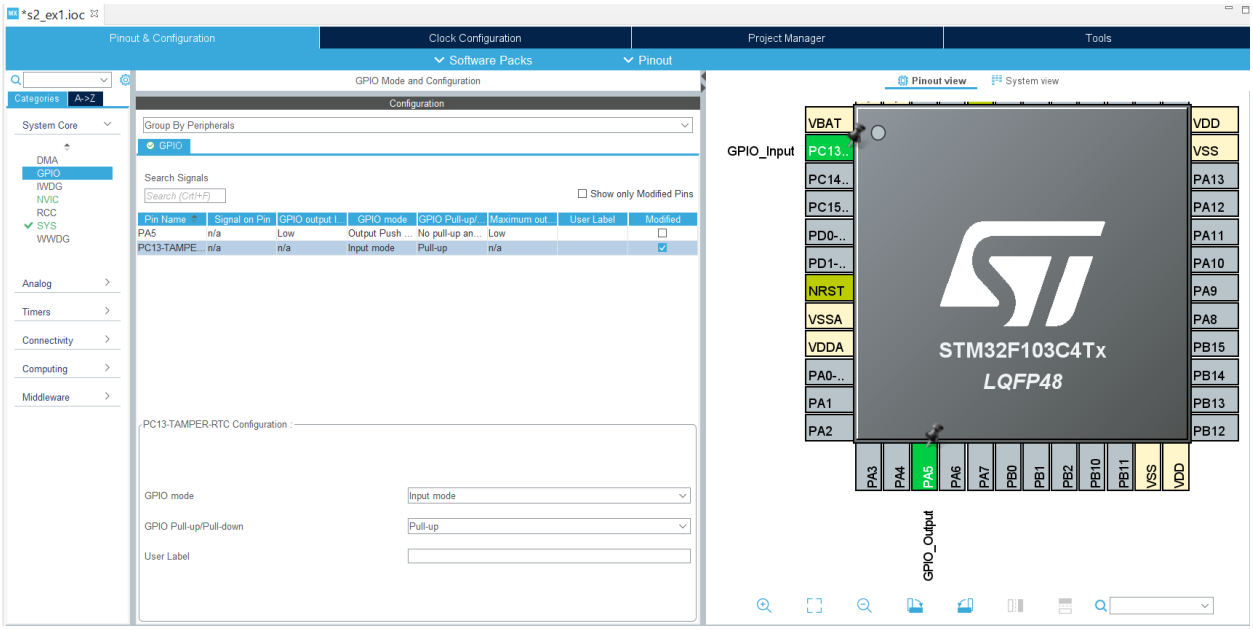


سنقوم باتباع الخطوات التالية:

نقوم بإعادة الخطوات الأولى والثانية كما هي في التطبيق الأول

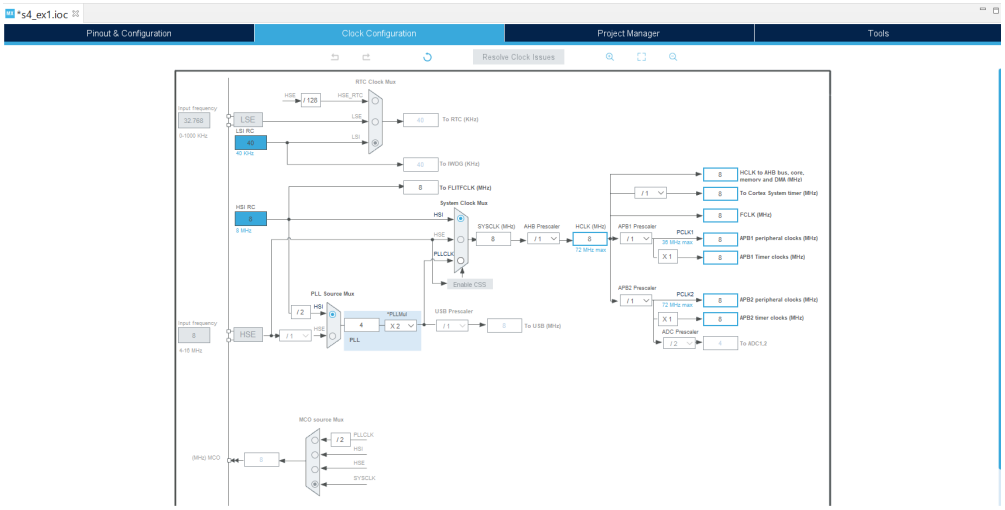
الخطوة الثالثة:

نقوم بضبط الإعدادات المناسبة للمشروع حيث سنقوم بضبط القطب PA5 كقطب خرج من نوع Push-Pull وبدون استخدام مقاومات رفع أو خفض No pull-up no Pull-down وسنعطيه الاسم led ، أيضاً سنقوم بضبط القطب PC13 كقطب دخل وسنقوم بتفعيل مقاومة الرفع الداخلية Pull-up كما في الشكل التالي:



الخطوة الرابعة: ضبط إعدادات الساعة للمتحكم

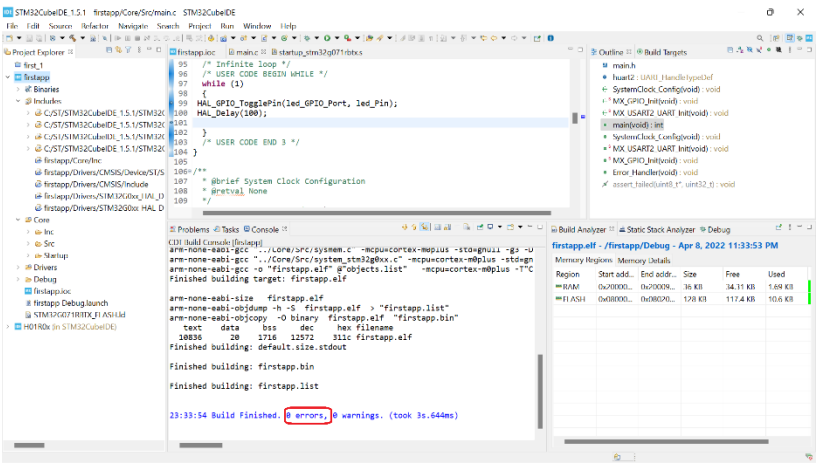
نقوم بضبط تردد الساعة للمتحكم ونختار مصدر الساعة الداخلية HSI كما هو موضح بالشكل التالي:



نضغط على Ctrl+s أو من Project...Generate code ، ليتم حفظ المشروع وتوليد الكود وإضافة المكتبات اللازمة، ثم نقوم بفتح main.c لتعديل الكود بما يناسب المشروع.

الخطوة الخامسة: نقوم بكتابة الكود المناسب ضمن main.c:

في حال خلو الكود من أخطاء الـ Syntax errors :



الخطوة السابعة: ننتقل لرسم الدارة على برنامج proteus : نفتح البرنامج ثم نبدأ بإضافة العناصر اللازمة للتطبيق الذي قمنا به وهي عبارة عن متحكم stm32f103c4 وليد ومقاومة ومفتاح لحظي، ثم نقوم برفع الكود إلى المتحكم

- **لتجريب التطبيق على البورد التجريبي قم بإعادة الخطوات السابقة ولكن اختر في الخطوة الثالثة المتحكم STM32G071RB**

ثم بعد كتابة الكود قم بالضغط على أيقونة الـ debug لتبدأ دارة ST-Link برفع الكود إلى المتحكم ثم البدء بجلسة debug لمراقبة الكود خطوة بخطوة وتصحيح الأخطاء البرمجية.

يصبح الكود النهائي بالشكل التالي:

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
  while (1)
  {

if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)==0)
{
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 1);
}
else
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 0);

     }}
```

8- للتحكم بالمداخل الرقمية للمتحكم STM32 دون استخدام مكتبة HAL:

هناك مجموعة من المسجلات تستخدم للتحكم بالمداخل الرقمية لمتحكم STM32 سنكتفي فقط بذكر المسجل المسؤول عن قراءة حالة المنفذ أو أحد الأقطاب الموجودة فيه (ولمزيد من المعلومات عن باقي المسجلات بإمكانك الرجوع إلى الـ datasheet الخاصة بالمتحكم)، ويدعى Input data register (IDR) وله الشكل التالي:

7.4.5 GPIO port input data register (GPIOx_IDR) (x = A..H)

Address offset: 0x10

Reset value: 0x0000 XXXX (where X means undefined)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **IDRy**: Port input data (y = 0..15)

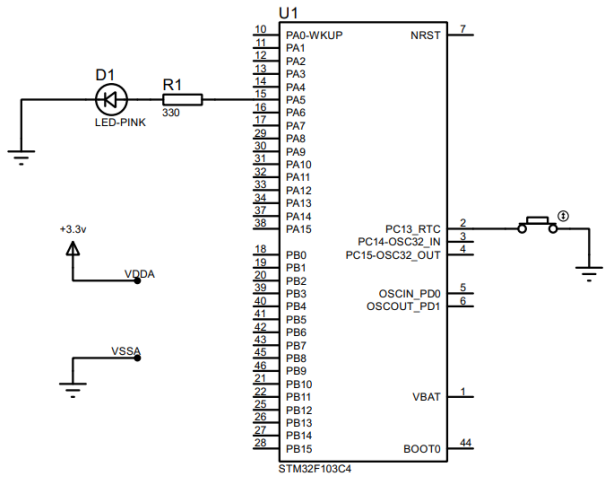
These bits are read-only and can be accessed in word mode only. They contain the input value of the corresponding I/O port.

والمسـجل IDR هو مسـجل للقراءة فقط ، البتات 16:31 غير مسـتخدمين، أما البتات 0:15 فيعبر كل بت عن حالة القطب المقابل له، ففي حال تفعيل مقاومة الرفع الداخلية للقطب عندها سـيكون القطب في حالة HIGH بشـكل دائم، وعند ضغط المفتاح الموصول معه سيصبح القطب في حالة LOW، لذا يجب مراقبة حالة القطب بشكل مستمر لحين يصبح القطب في حالة LOW عندها يكون المفتاح الموصول معه مضغوط.

فلفحص حالة القطب الأول من المنفذ A نستخدم جملة الشرط التالية:

if ((GPIOx->IDR & GPIO_PIN_1) = = 0)

9- إعادة التطبيق الثاني دون استخدام مكتبةHAL وباستخدام بيئة المحاكاة Proteus

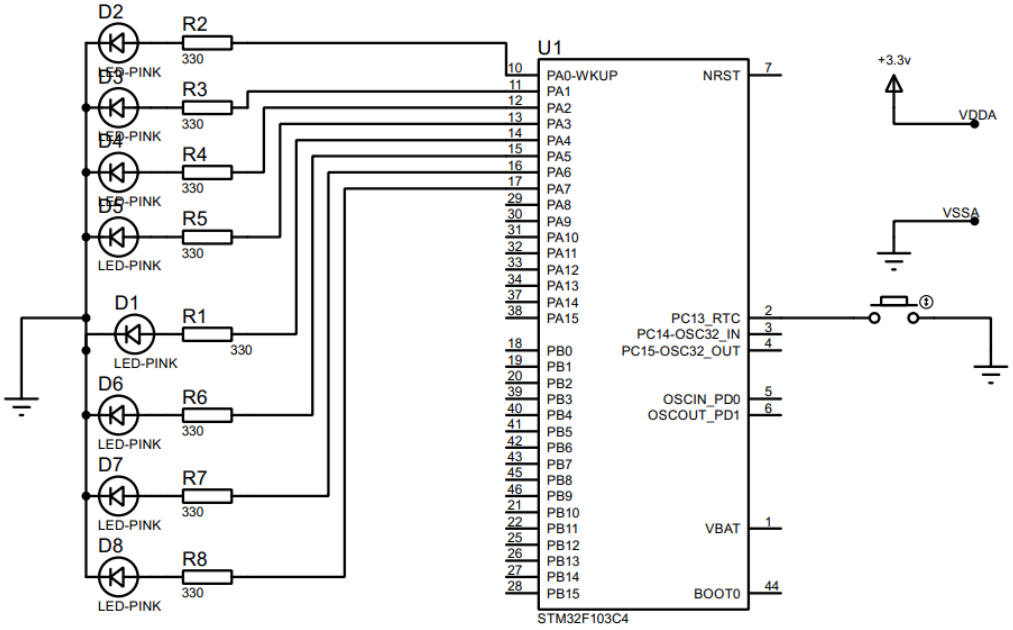


ويكون الشكل النهائي للكود:

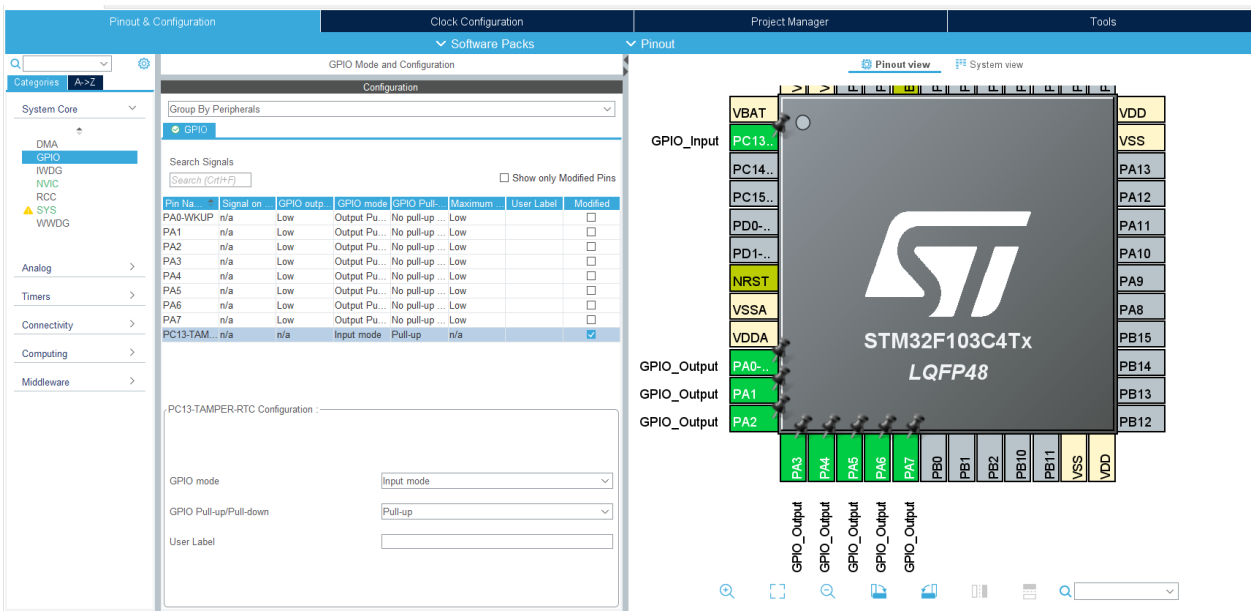
```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
```

```
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
  while (1)
  {
          if (   (GPIOC->IDR & GPIO_PIN_13)  == 0   )

         {
                  GPIOA->ODR = 0x0020;
         }


    else
        GPIOA->ODR = 0;

     }}
```

10-التطبيق الثالث: إضاءة مجموعة من الليدات من خلال زر start



أعد الخطوات السابقة مع ضبط إعدادات الأقطاب بالشكل التالي:

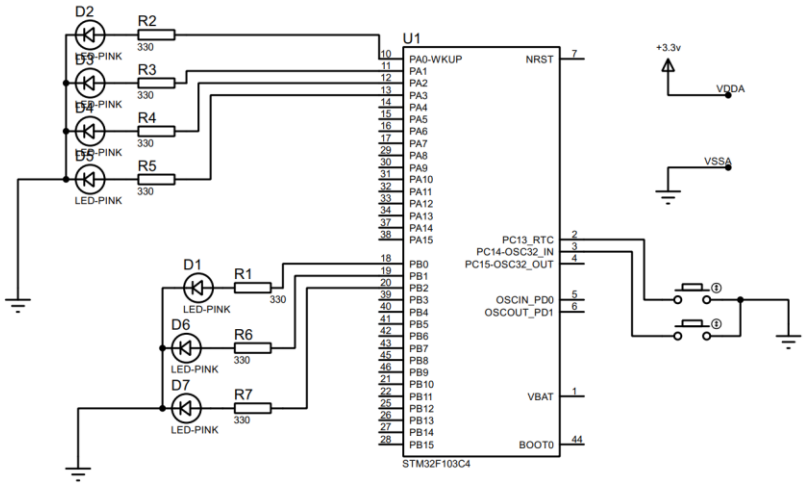


ويكون الشكل النهائي للكود :

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
   while (1)
  {
          if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)==0)
          {
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 1);
                    HAL_Delay(250);
                    //**************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 1);
                    HAL_Delay(250);
                    //**************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 1);
                    HAL_Delay(250);
                    //**************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 1);
                    HAL_Delay(250);
                    //**************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 1);
                    HAL_Delay(250);
                    //*************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 1);
                    HAL_Delay(250);
                    //************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, 1);
                    HAL_Delay(250);
                   //*************
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, 1);
                    HAL_Delay(250);
          }

          else
          {
```

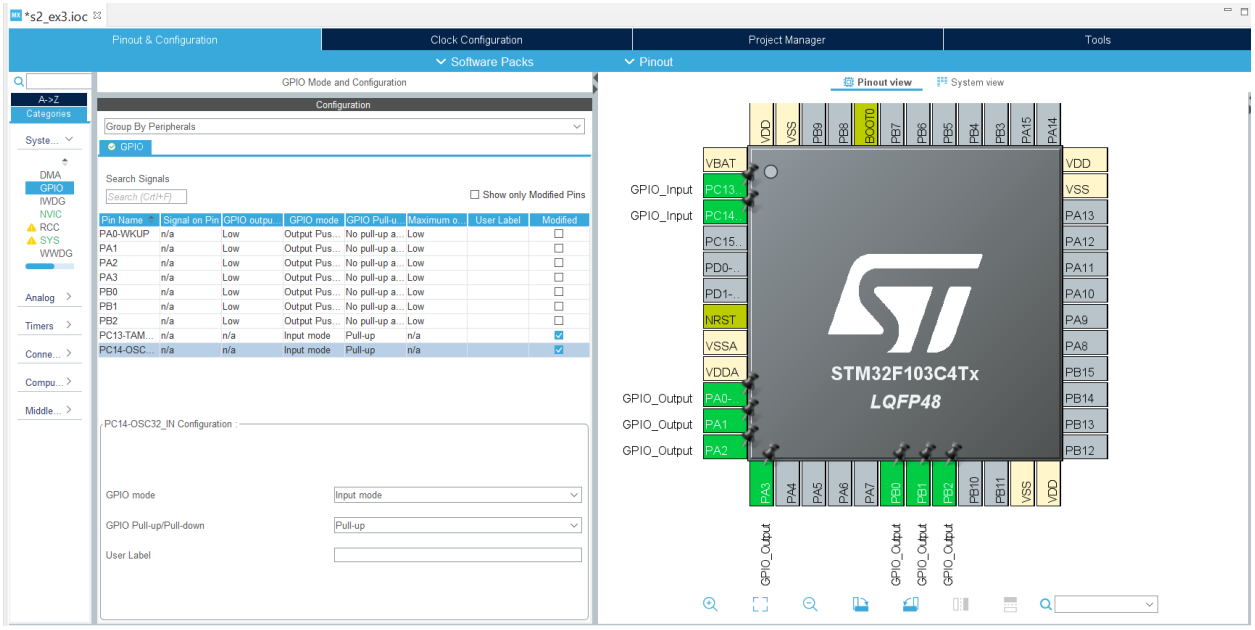
```
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, 0);

    }
}}
```

11-التطبيق الرابع: تشغيل مجموعتين من الليدات بتسلسل معين عند الضغط على أزرار التشغيل



- عند الضغط على الزر الموصول مع الـ C13 يتم تشغيل اللدات الموصولة مع المنفذ A
- عند الضغط على الزر الموصول مع الـ C14 يتم تشغيل اللدات الموصولة مع المنفذ B

أعد الخطوات السابقة مع ضبط إعدادات الأقطاب بالشكل التالي:

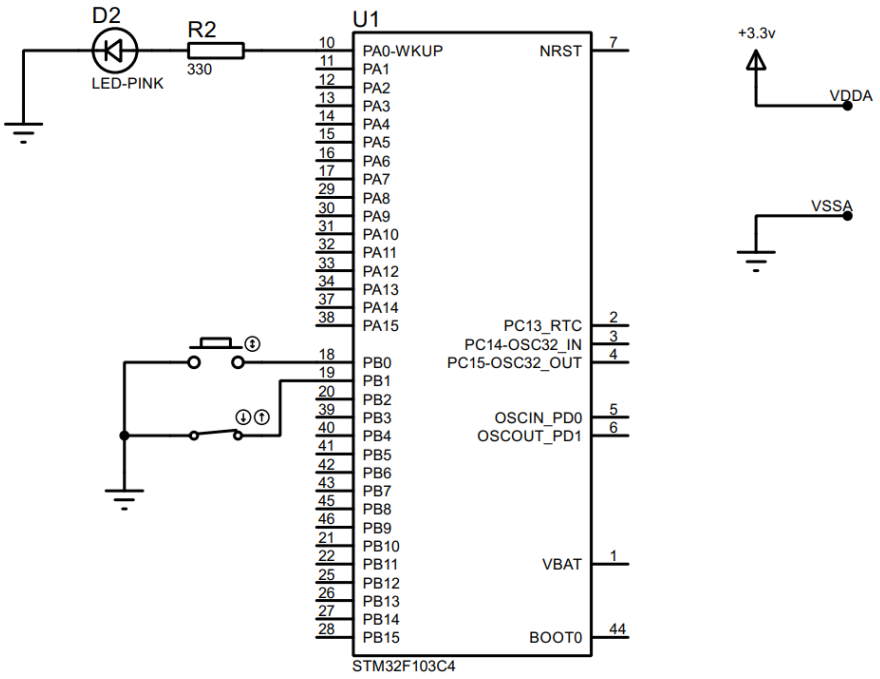


مدرس المقرر: -------

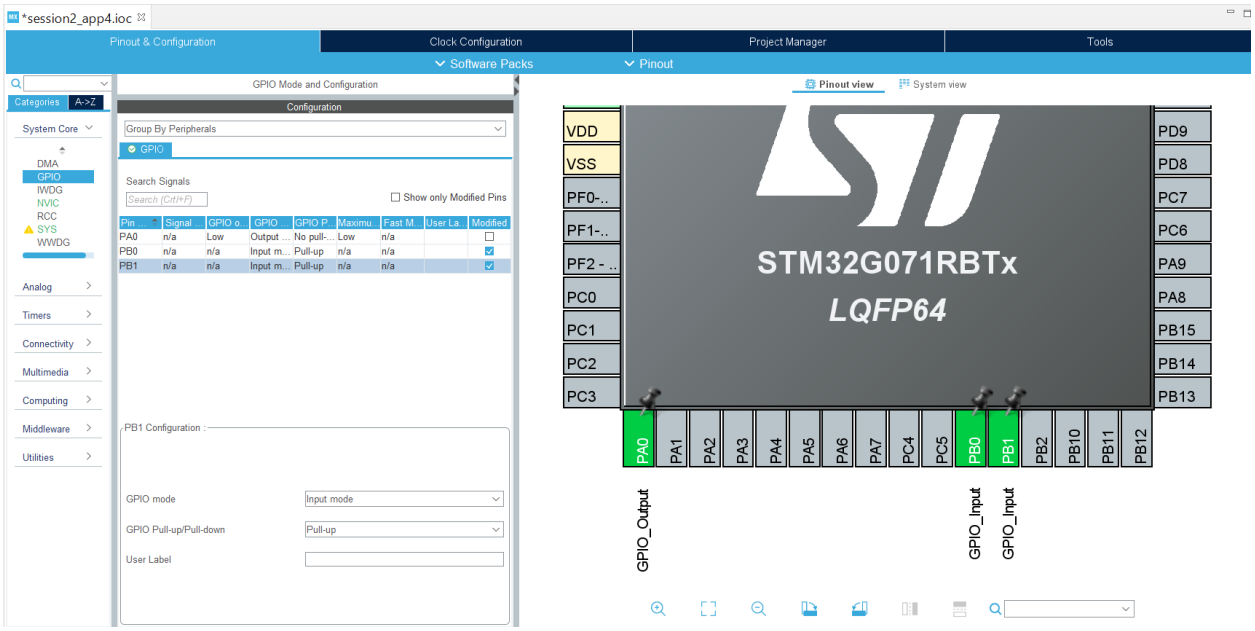
ويكون الشكل النهائي للبرنامج:

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
   while (1)
  {
          if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)==0)
          {
                      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 1);
                      HAL_Delay(250);
                      //**************
                      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 1);
                      HAL_Delay(250);
                      //**************
                      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 1);
                      HAL_Delay(250);
                      //**************
                      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 1);
                      HAL_Delay(250);
                      //*************

          }

          else
          {
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 0);
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 0);
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 0);
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);


          }
          //-------------------------------------------------
          if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_14)==0)
                 {
                             HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, 1);
                             HAL_Delay(250);
                             //**************
                             HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, 1);
                             HAL_Delay(250);
                             //**************
                             HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, 1);
                             HAL_Delay(250);
                                          }

                else
                {
                    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, 0);
                    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, 0);
                    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, 0);
                                }

  }}
```

12- التطبيق الخامس: تشغيل الليد الموصول مع الـ A0 في حال كان الـ Switch و الـ Button (الموصولين على الأقطاب PB0 و PB1) مضغوطين



أعد الخطوات السابقة مع ضبط إعدادات الأقطاب بالشكل التالي:



ويكون الشكل النهائي للبرنامج بالشكل التالي:

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int main(void)
{
    HAL_Init();
```

```
    SystemClock_Config();
    MX_GPIO_Init();
  while (1)
  {
if((HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_0)==0) && (HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_1)==0))
{
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 1);
}
else
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_0, 0);

     }}
```