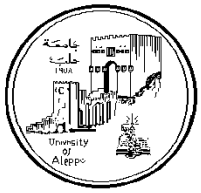
جامعة حلب
كلية الهندسة الكهربائية والالكترونية
قسم هندسة التحكم والأتمتة
مخبر التحكم



مقرر المتحكمات المصغرة

الجلسة الثالثة

السنة الرابعة ميكاترونيك

2023/2022

مدرس المقرر: _____

الغاية من الجلسة

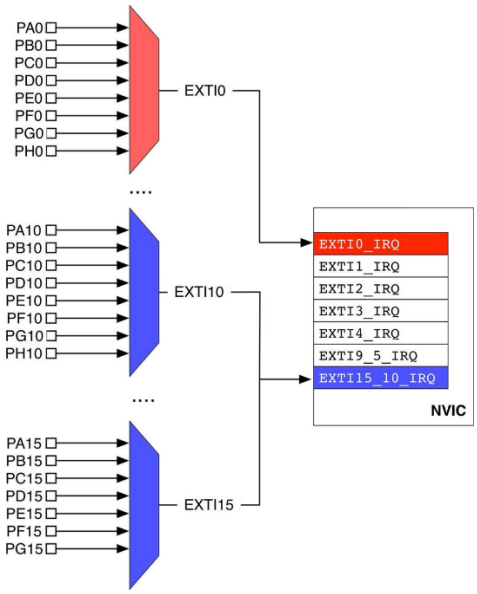
1- التعرف على المقاطعات الخارجية في متحكمات STM32 بحيث يكون قادراً على تهيئة هذه المقاطعات في المتحكم لاستثمارها حسب الحاجة.
2- التطبيق1: بناء لنظام يحتوي على زر طوارئ لتفعيل برنامج للطوارئ حيث يتم تنفيذه بغض النظر عن عمل النظام باستخدام بيئة المحاكاة proteus وباستخدام البورد التطويري.
3- التعرف إلى شاشة 7-seg
4- التطبيق2: بناء لنظام يحتوي على شاشة 7-seg لتعمل كعداد من 0 حتى 9 باستخدام بيئة المحاكاة proteus وباستخدام البورد التطويري.
5- التطبيق3: تشغيل وإطفاء جهاز من خلال زر لحظي واحد باستخدام المقاطعة إظهار حالة الجهاز على شاشة الإظهار باستخدام بيئة المحاكاة proteus وباستخدام البورد التطويري.
6- التطبيق4: عداد زوار و إظهار العدد على شاشة 7-seg بخانتين ,حيث ان الكباسين موصلين على اقطاب المقاطعة الخارجية (INT0,INT1) ويتم تفعيل المقاطعتين عند الجبهة الهابطة باستخدام بيئة المحاكاة proteus وباستخدام البورد التطويري.

1- تعريف المقاطعة:

هي الاستجابة لحدث ما أثناء تنفيذ البرنامج الرئيسي void main وذلك عن طريق تنفيذ برنامج فرعي يسمى برنامج خدمة المقاطعة. و غالباً ما نصـــادف المقاطعات في الحياة اليومية وكمثال عليها يمثل رنين الهاتف مقاطعة تعبر عن وجود مكالمة هاتفية تنتظر الرد عليها ويمثل برنامج خدمة المقاطعة في المتحكم الرد على هذه المكالمة.

المقاطعات الخارجية INTx:

يحتوي متحكم STM32 على 16 خط للمقاطعات الخارجية هي من الخط 0 حتى الخط 15حيث تشير أرقام الخطوط إلى أرقام اطراف الـــ GPIOS يتم تجميع المقاطعات الخارجية ضمن خطوط متصلة مع أطراف المتحكم المصغر GPIO، وبما أن للمعالج عدة أطراف GPIOS لذا فإن كل خط من خطوط المقاطعات الخارجية EXTI هو عبارة عن خط مشترك بين عدة أطراف Pins.



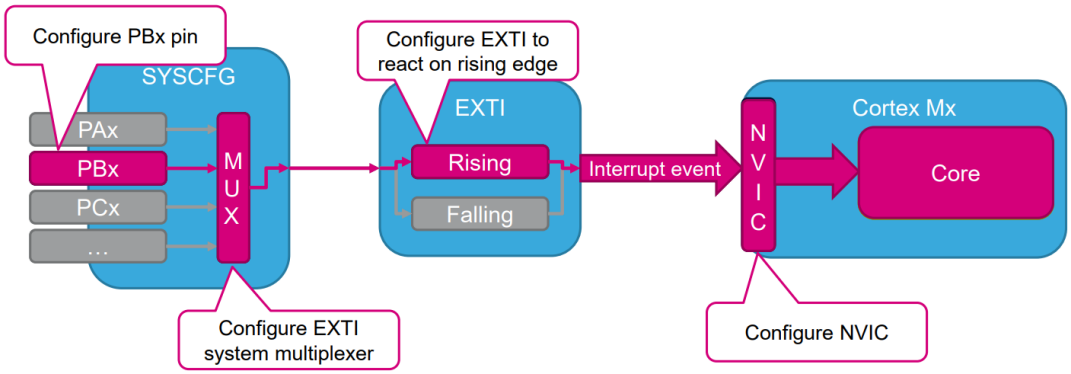
في كل خط من خطوط المقاطعات الخارجية (كل مجموعة) يحق لـ pin واحد أن يقوم بتوليد المقاطعة وعلى البرنامج أن يكتشـــف أي pin قام بتوليد المقاطعة حيث يمكن قدح المقاطعة عند الجبهة الصـــاعدة rising edge أو الهابطة falling edge أو عند كليهما ، هذا يعني أن الطرف PA0 متصل بالخط LINE0 والطرف PA13 متصل بالطرف LINE13 ، أيضـاً PB0 و PC0 متصـلين بالخط LINE0 ، بمعنى أن جميع الأطراف ذات الأرقام Px0 متصـلة بالخط LINE0 (حيث يعبر x عن اسـم المنفذ) ، أيضـاً جميع الأطراف ذات الأرقام Px3 متصـلة بالخط LINE3 وهكذا...

لكن يجب الانتباه للملاحظات التالية:

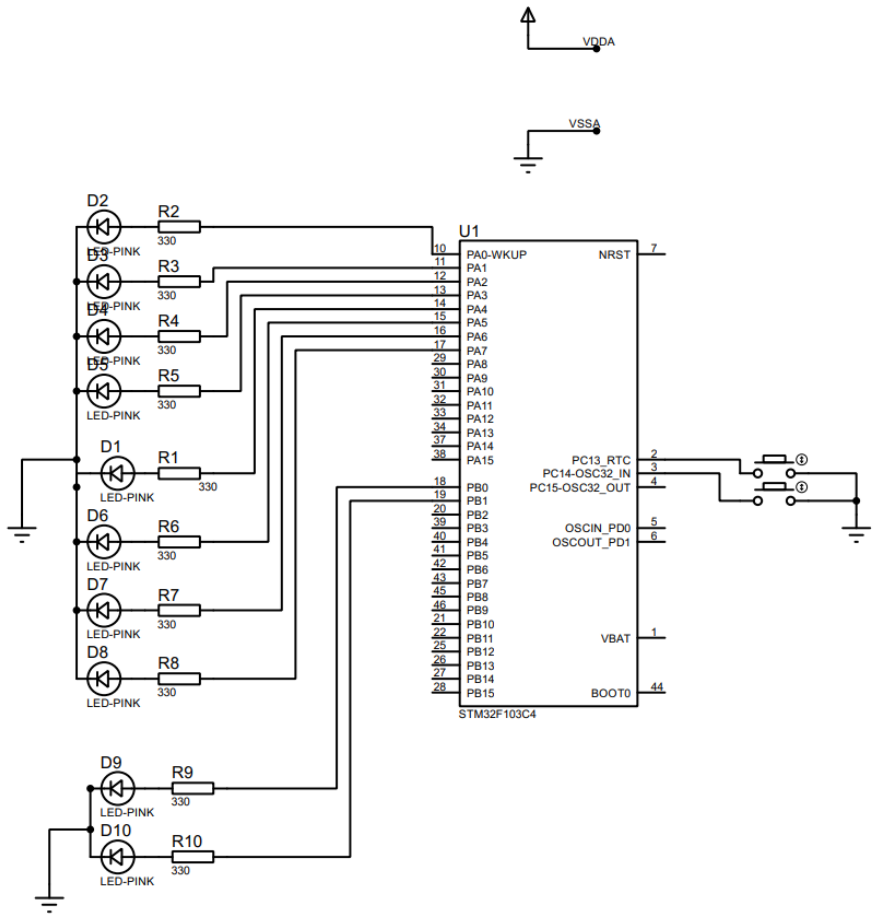
1. الأطراف PA0،PB0، PC0... جميعها متصـــلة بالخط LINE0 لذا في اللحظة الواحدة بإمكانك توليد مقاطعة على طرف واحد فقط من هذه الأطراف.

2. الأطراف PA0، PA5 متصـــلين على خطين مختلفين من خطوط المقاطعة لذا يمكنك اســـتخدامهم في نفس اللحظة لتوليد المقاطعة.

في حال حدوث المقاطعة فإن المعالج يتوقف عن تنفيذ الكود الحالي ويقوم بمعالجة المقاطعة من خلال تنفيذ برنامج خدمة المقاطعة (Interrupt Service Routines (ISR) والذي يتم تحديد عنوانه في الذاكرة من خلال جدول أشعة المقاطعة المعرف مسبقاً (Vector Interrupt Table (VIC)).



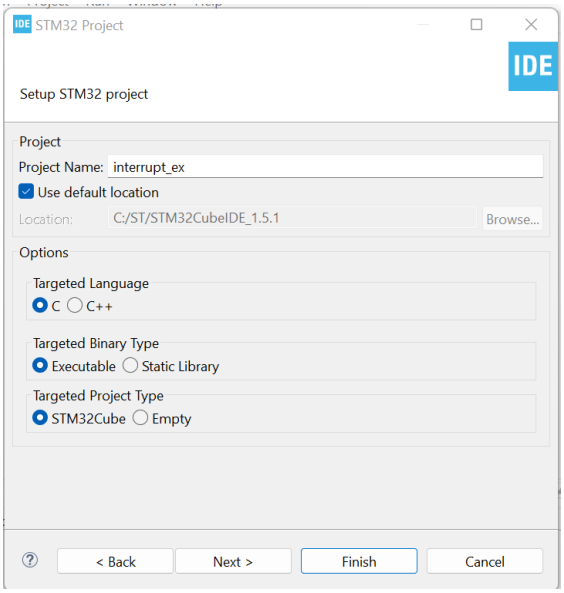
2- **التطبيق العملي1 :** زر طوارئ لتفعيل برنامج للطوارئ حيث يتم تنفيذه بغض النظر عن عمل النظام (إنارة اللدات) و بعد الانتهاء من برنامج خدمة المقاطعة (الطورائ) يعود للبرنامج الرئيسي ليتابع عمله وذلك باستخدام بيئة المحاكاة proteus ثم باستخدام البورد التطويري:



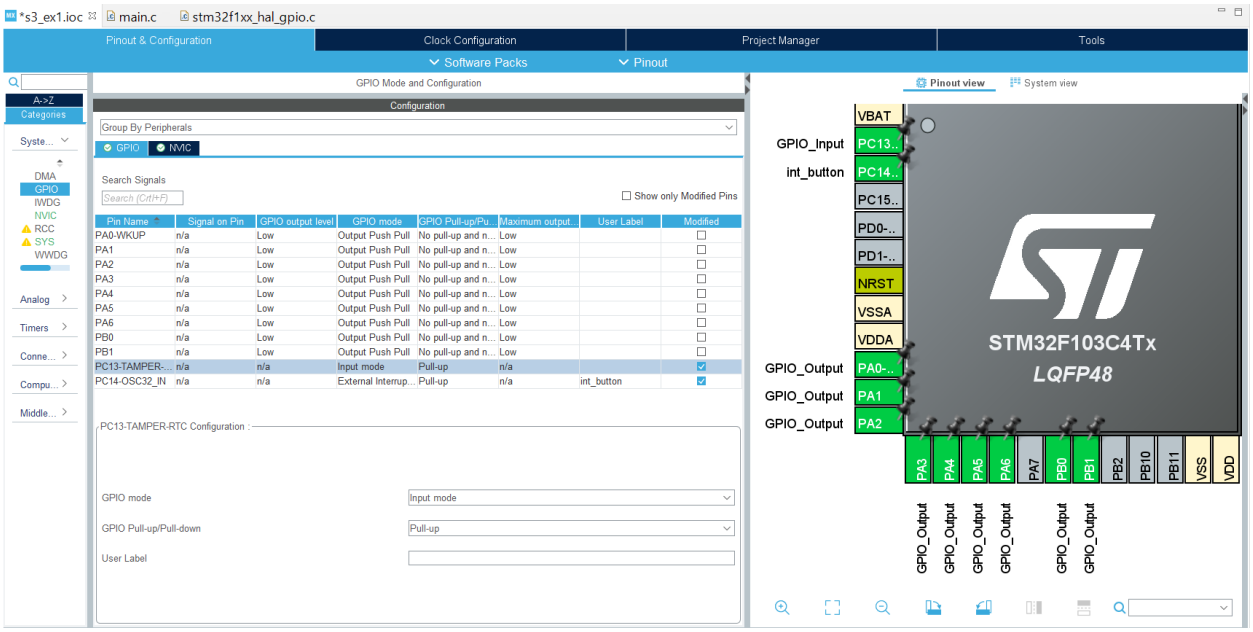
1. **ضبط إعدادات المشروع:**

الخطوة الأولى: فتح بيئة STM32CubeIDE وإنشاء مشروع جديد ثم قم باختيار المتحكم المناسب من أجل بيئة المحاكاة اختر المتحكم stm32f103c4

الخطوة الثانية: اختيار اسم للمشروع

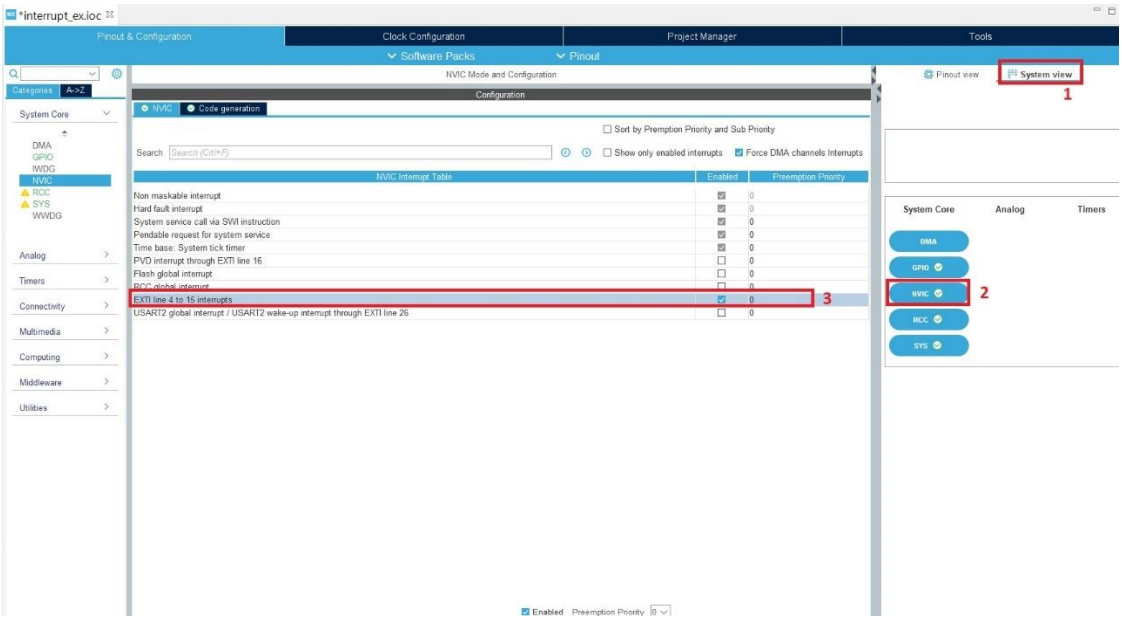


الخطوة الثالثة: اختر الأقطاب PA0:PA6 ، والأقطاب PB0:PB1 لضـــبطها كأقطاب خرج ، والقطب PC13 كقطب دخل مع تفعيل مقاومة الرفع الداخلية، والقطب PC14 كقطب مقاطعة خارجية:

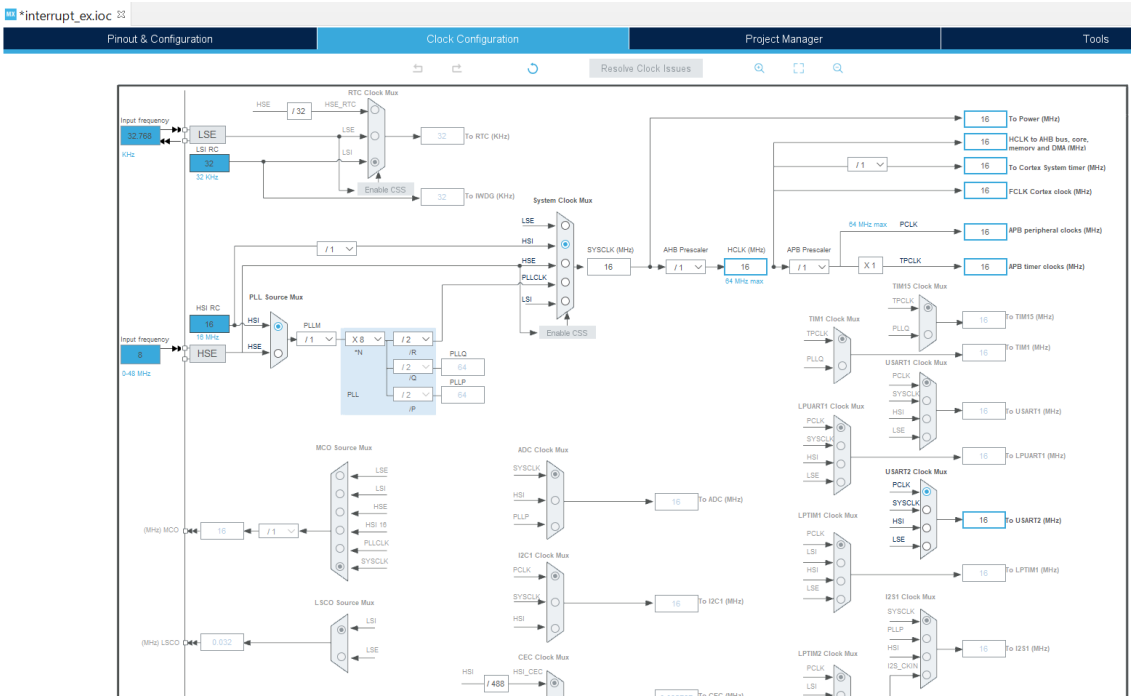


بما أننا قمنا بتفعيل مقاومة الرفع الداخلية لقطب المقاطعة لذا سنقوم بتفعيل المقاطعة عند الجبهة الهابطة.

الخطوة الرابعة: قم بفتح NVIC Tab ثم قم بتفعيل المقاطعات الخارجية، يمكنك أيضـــاً إعادة ضـــبط مســـتويات الأولوية للمقاطعات:

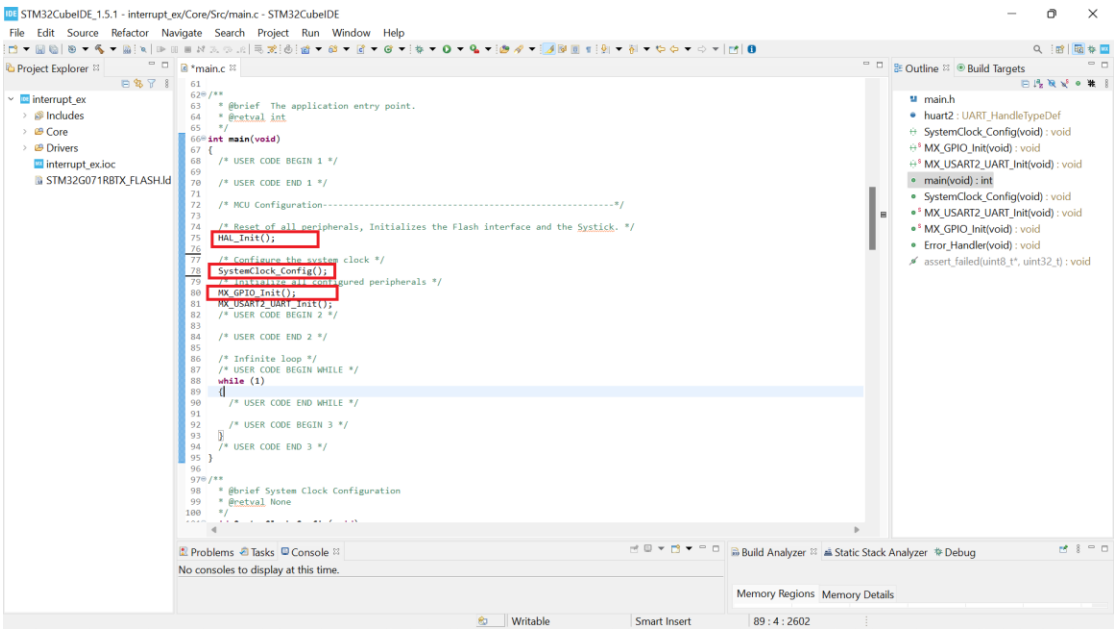


الخطوة الخامسة: قم بضبط ساعة النظام على الساعة الداخلية(HSI) واختر التردد 16MHZ



الخطوة السادسة: قم بتوليد الكود اعتماداً على الإعدادات التي قمت بضبطها من خلال الضغط على زري Ctrl+S

نلاحظ أن ملف الـــــ main.c يحتوي على تهيئة وتضـمين مكتبة HAL وضـبط سـاعة النظام بالإضـافة إلى تهيئة المداخل والمخارج والطرفيات كما في الشكل التالي:

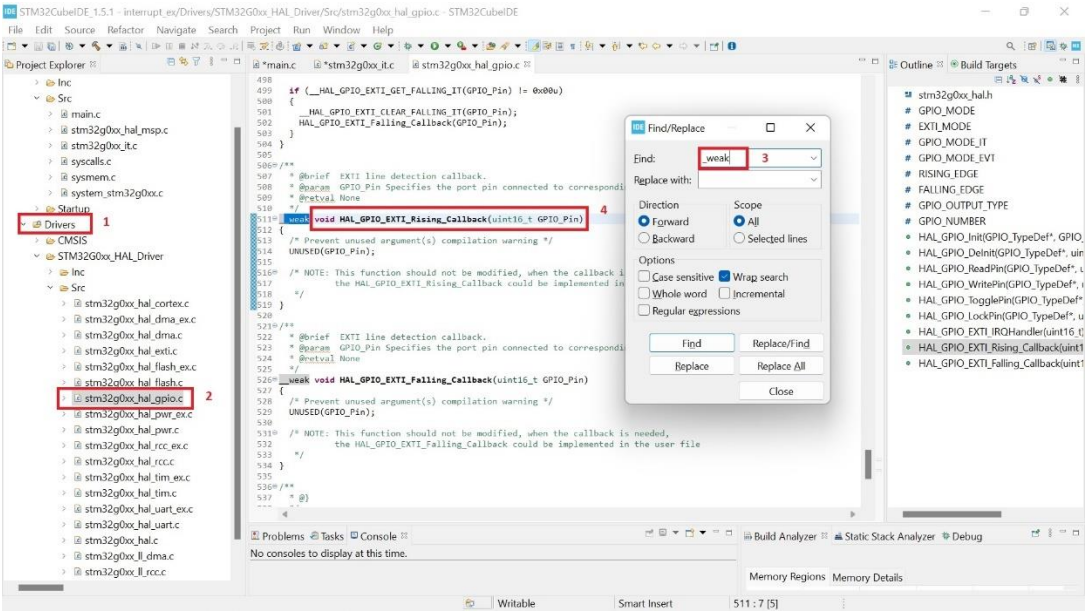


الخطوة السابعة :

1- نختار المجلد Drivers

2- نقوم بفتح الملف stm32g0xx_hal_gpios.c (عند استخدام المتحكم stm32f1xx سيكون اسم الملف stm32f1xx_hal_gpios.c)

3- نبحث عن الـ function الذي يبدأ بكلمة weak_ ونحدد اسم الـ function وننسخه

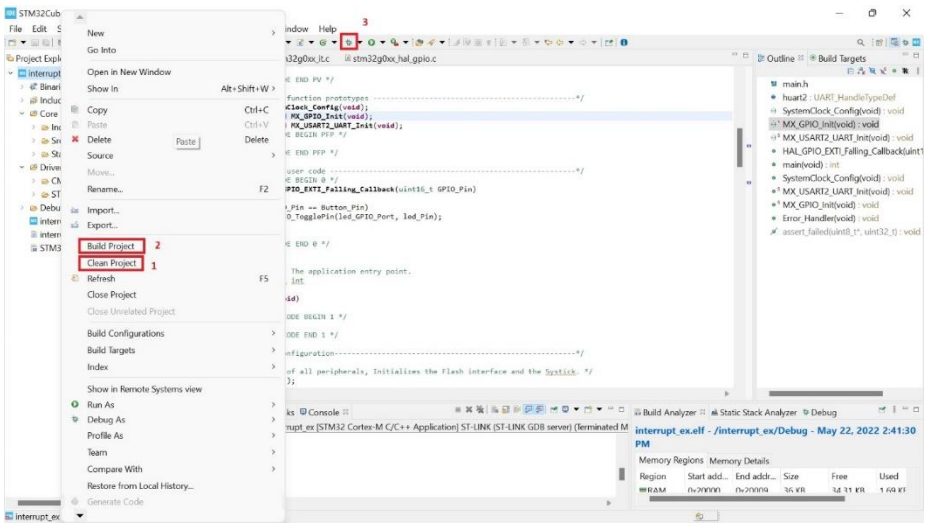


تعني كلمة weak_ أي سيتم استدعاؤه في حال لم يكتب المستخدم برنامج خدمة مقاطعة يحمل نفس الاسم.

4- نقوم بلصق الـــ function ضمن ملف الـــ main.c إما قبل أو بعد الـــ int main() مع إزالة كلمة weak_ ونضـــع فيه الأوامر التي نريد تنفيذها عند طلب المقاطعة مثلاً هنا قمنا بفحص القطب الذي حدثت عنده المقاطعة ثم قمنا باســـتدعاء تابع HAL الذي يقوم بعكس الحالة المنطقية لليد عند حدوث المقاطعة.

الخطوة الثامنة: التأكد من خلو الكود من أي أخطاء برمجية من خلال تنظيف الكود clean project ثم ترجمة الكود من خلال Build Project من أجل توليد ملف hex. ومن ثم برمجته بالمتحكم stm32f103c4 ضمن بيئة المحاكاة proteus

الخطوة التاسعة استخدام البورد التطويري: قم بإعادة نفس الخطوات السابقة ولكن هذه المرة قم باختيار المتحكم stm32G0B1CEU وفي آخر خطوة قم برفع الكود للبورد التطويري من خلال نافذة الـ debug



الكود بالكامل:

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

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin ==GPIO_PIN_14)
    {
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0|GPIO_PIN_1 , GPIO_PIN_SET);

    }
}

int main(void)
{
    HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
   while (1)
  {
        if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)==0)
        {
                GPIOA->ODR = 0X0001;
                HAL_Delay(50);
                //**************
                GPIOA->ODR = 0X0002;
                HAL_Delay(50);
                //**************
                GPIOA->ODR = 0X0004;
```

```
                HAL_Delay(50);
                //**************
                GPIOA->ODR = 0X0008;
                HAL_Delay(50);
                //*************
                GPIOA->ODR = 0X0010;
                HAL_Delay(50);
                //************
                GPIOA->ODR = 0X0020;
                HAL_Delay(50);
                //***********
                GPIOA->ODR = 0X0040;
                HAL_Delay(50);
               //************
                GPIOA->ODR = 0X0080;
                HAL_Delay(100);

        }
        else
        {
              GPIOA->ODR = 0X0000;
        }

  }

}
```

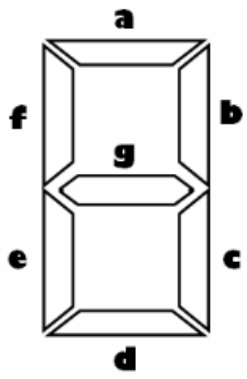
ملاحظة:

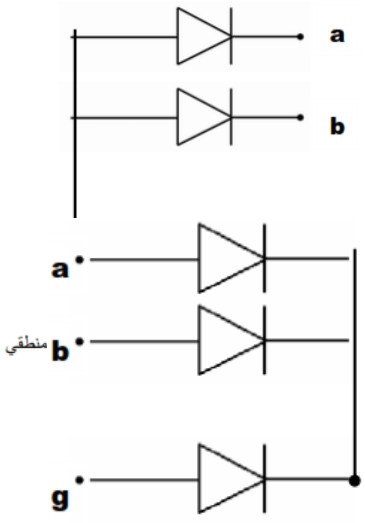
لزيادة سرعة الاستجابة للمقاطعات عليك الاستغناء عن مكتبة HAL واستخدام المسجلات بشكل مباشر.

أيضاً لا يمكن استخدام التأخير الزمني من مكتبة HAL ضمن برنامج خدمة المقاطعة.

3- التعرف إلى شاشة 7-seg :

تستخدم شاشات الإظهار الرقمية ذات السبع قطع 7-Segment Led لإظهار الأعداد ومثال عليها استخدامها لإظهار عدد الزوار في معرض أو إظهار التوقيت كما في الساعة الرقمية أو إظهار درجة الحرارة أو قيمة الفولت والتيار في ساعات القيم الكهربائية وغيرها الكثير، وهي عبارة عن سبعة لدات موزعة بترتيب يمكن من تشكيل الأرقام من 0 حتى 9 وتسمى اللدات بالأحرف a,b,c,…g كما في الشكل وتقسم هذه الشاشات إلى نوعين:





الشاشات ذات المصاعد المشتركة: بهذه الحالة تكون جميع مصاعد اللدات موصولة إلى قطب مشترك يسمى المصعد المشترك ويكون مهبط كل لد موصول إلى قطب خارجي خاص مسؤول عن إضاءة هذا اللد.

الشاشات ذات المهابط المشتركة:

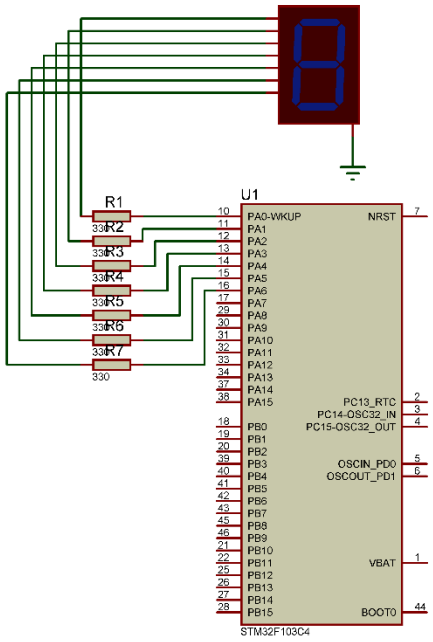
بهذه الحالة تكون جميع مهابط اللدات موصولة إلى قطب مشترك يسمى المهبط المشترك ويكون مصعد كل لد موصول إلى قطب خارجي خاص مسؤول عن إضاءة هذا اللد.

التوصيل:

من أجل الشاشات ذات المهابط المشتركة نقوم بتوصيل القطب المشترك إلى 0v ونوصل الأقطاب a,b,c,..,g إلى منفذ الخرج الرقمي وليكن مثلا المنفذ A وبحسب القيمة على هذا المنفذ أي حسب القيمة الخارجة من المنفذ الرقمي ستضاء اللدات الموصولة معها ويوضح الجدول التالي توصيل شاشة ذات مهابط مشتركة.

القيمة ست عشريا على المنفذ A	PA7 dot	PA6 g	PA5 f	PA4 e	PA3 d	PA2 c	PA1 b	PA0 a	الرقم على الشاشة
0x3F	0	0	1	1	1	1	1	1	0
0x06	0	0	0	0	0	1	1	0	1
0x5B	0	1	0	1	1	0	1	1	2
0x4F	0	1	0	0	1	1	1	1	3
0x66	0	1	1	0	0	1	1	0	4
0x6D	0	1	1	0	1	1	0	1	5
0x7D	0	1	1	1	1	1	0	1	6
0x07	0	0	0	0	0	1	1	1	7
0x7F	0	1	1	1	1	1	1	1	8
0x6F	0	1	1	0	1	1	1	1	9

4- التطبيق العملي2 : برمجة شاشة واحدة ذات مصاعد مشتركة لتعمل كعداد يعد من 0 حتى 9 كل زمن معين وتتكرر العملية بشكل دائم وذلك باستخدام بيئة المحاكاة proteus ثم باستخدام البورد التطويري:

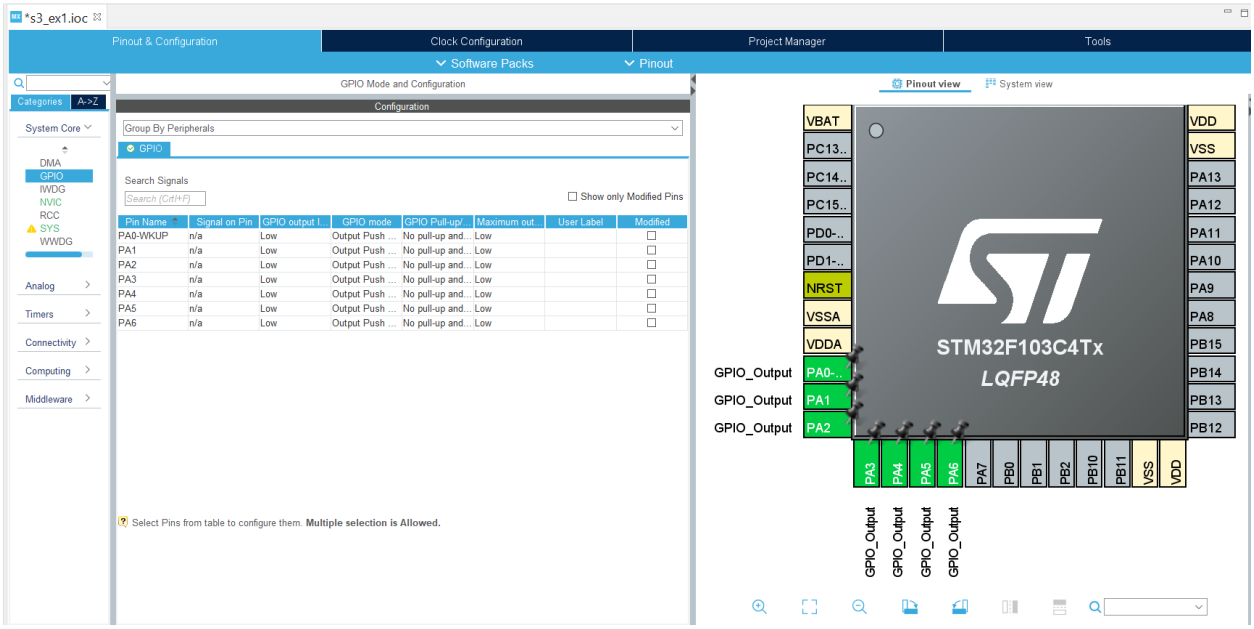


ضبط إعدادات المشروع:

الخطوة الأولى: فتح بيئة STM32CubeIDE وإنشاء مشروع جديد ثم قم باختيار المتحكم المناسب من أجل بيئة المحاكاة اختر المتحكم stm32f103c4

الخطوة الثانية: اختيار اسم للمشروع

الخطوة الثالثة: اختر الأقطاب PA0:PA6 لضبطها كأقطاب خرج



الخطوة الرابعة: قم بضبط ساعة النظام على الساعة الداخلية(HSI) واختر التردد 8MHZ

الخطوة الخامسة: قم بتوليد الكود اعتماداً على الإعدادات التي قمت بضبطها من خلال الضغط على زري Ctrl+S

يصبح الكود النهائي:

```
#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 = 0x003F; //Displaying 0
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x0006; //Displaying 1
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x005B; //Displaying 2
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x004F; //Displaying 3
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x0066; //Displaying 4
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x006D; //Displaying 5
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x007D; //Displaying 6
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x0007; //Displaying 7
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x007F; //Displaying 8
                HAL_Delay(1000);     //One second delay
                GPIOA ->ODR = 0x0067; //Displaying 9
                HAL_Delay(1000);     //One second delay
   }
}
```

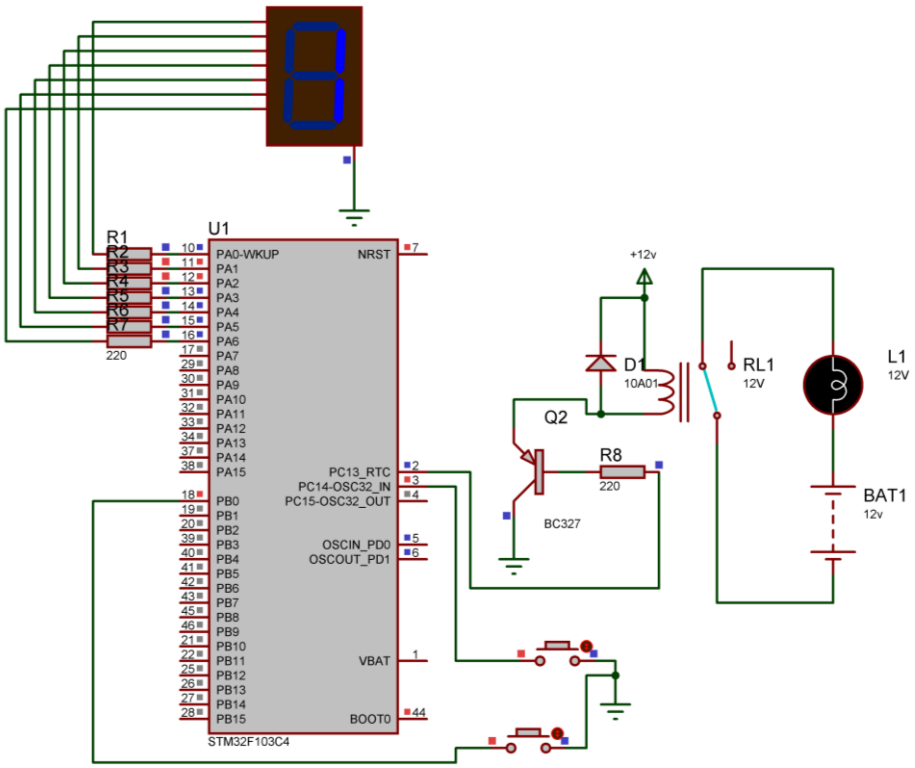
الخطوة الســادســة: التأكد من خلو الكود من أي أخطاء برمجية من خلال تنظيف الكود clean project ثم ترجمــة الكود من خلال Build Project من أجـل توليـد ملف hex. ومن ثم برمجتــه بــالمتحكم stm32f103c4 ضمن بيئة المحاكاة proteus

ملاحظة:

- اخترنا التردد 8MHZ ضمن STM32cubeIDE وضمن الـ PROTEUS كي تكون المحاكاة بالزمن الحقيقي
- ضمن برنامج الـ PROTEUS قم بضبط إعدادات التيار للـ 7seg على 0.01mA كي يتمكن المتحكم من تشغيلها.

الخطوة الســابعة اســتخدام البورد التطويري: قم بإعادة نفس الخطوات الســابقة ولكن هذه المرة قم باختيار المتحكم stm32G0B1CEU وفي آخر خطوة قم برفع الكود للبورد التطويري من خلال نافذة الـ debug

التطبيق العملي3 : تشغيل جهاز من خلال زر لحظي موصول على القطب PC14 باستخدام المقاطعة، وإطفاؤه من خلال زر لحظي موصول على القطب PB0 أيضاً باستخدام المقاطعة، و إظهار حالة الجهاز على شاشة الإظهار بحيث يتم كتابة الرقم (1) في حال كان الخرج مفعل (On) و إظهار الرقم (0) في حال كان الخرج مطفئ (Off)، وذلك باستخدام بيئة المحاكاة proteus ثم باستخدام البورد التطويري

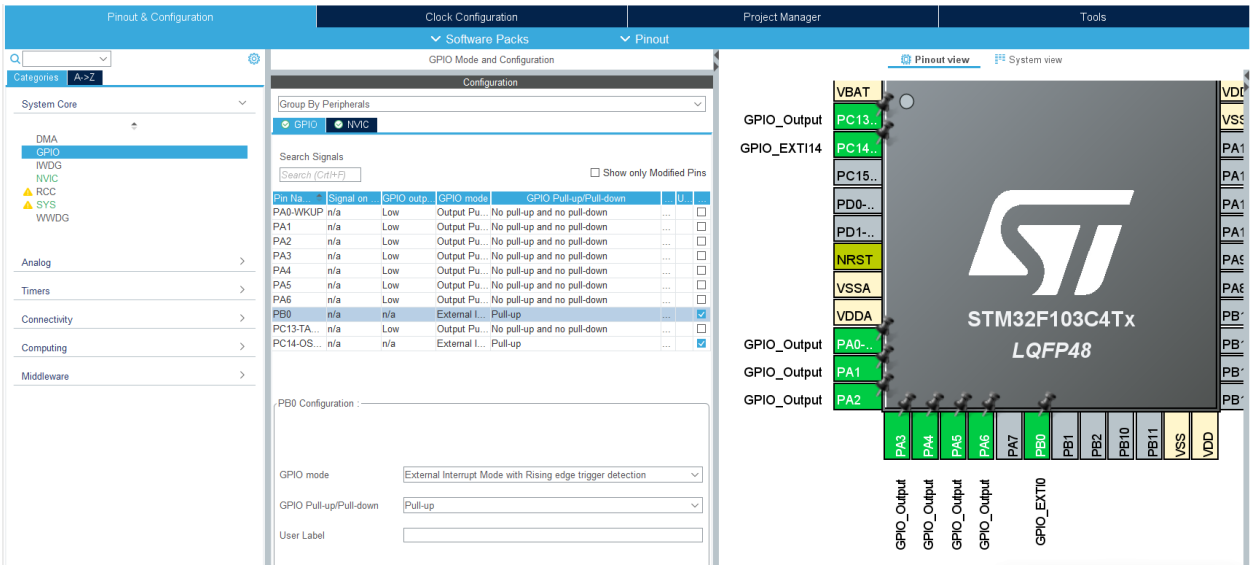


ضبط إعدادات المشروع:

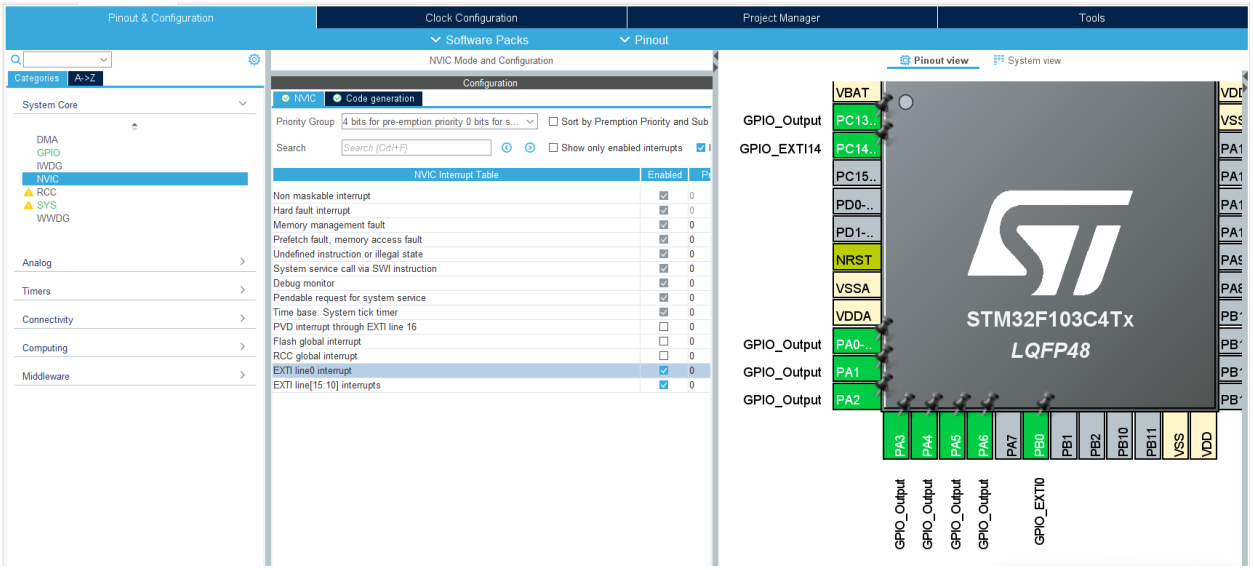
الخطوة الأولى: فتح بيئة STM32CubeIDE وإنشـاء مشـروع جديد ثم اختيار المتحكم المناسـب من أجل بيئة المحاكاة اختر المتحكم stm32f103c4

الخطوة الثانية: اختيار اسم للمشروع

الخطوة الثالثة: اختر الأقطاب PA0:PA6 لضـبطها كأقطاب خرج، وضـبط القطب PC13 كقطب خرج، والقطب PC14 كقطب دخل



قم بتفعيل المقاطعات على الأقطاب:



الخطوة الرابعة: قم بضبط ساعة النظام على الساعة الداخلية(HSI) واختر التردد 8MHZ

الخطوة الخامسة: قم بتوليد الكود اعتماداً على الإعدادات التي قمت بضبطها من خلال الضغط على زري Ctrl+S

يصبح الكود النهائي:

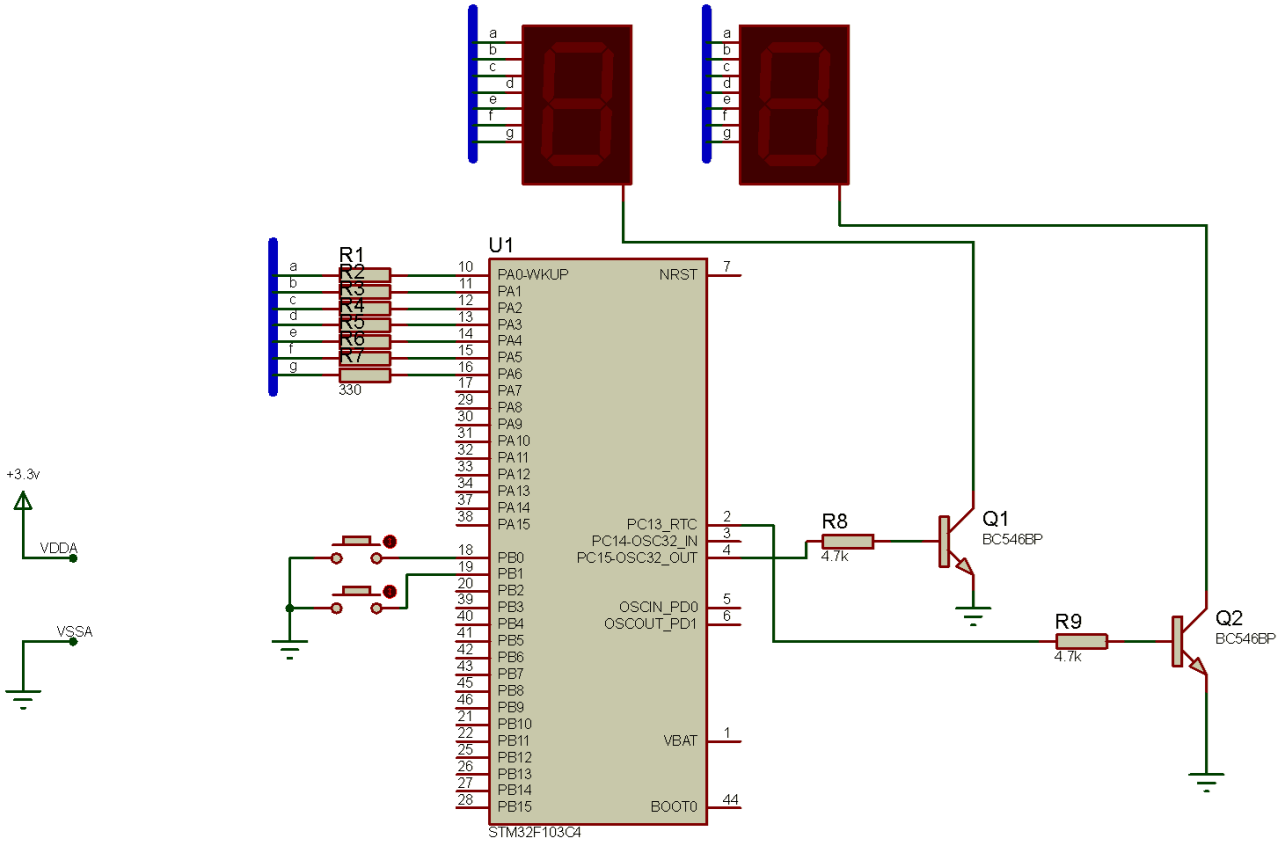
```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
```

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin ==GPIO_PIN_14)
    {

        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_SET);
         GPIOA->ODR = 0x003f; //Displaying 0

    }
    else  if(GPIO_Pin ==GPIO_PIN_0)
       {

             HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_RESET);
              GPIOA->ODR = 0x0006; //Displaying 1
       }

 }


int main(void)
{

  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  GPIOA->ODR = 0x003f; //Displaying 0
    while (1)
  {

  }
}
```

الخطوة السـادسـة: التأكد من خلو الكود من أي أخطاء برمجية من خلال تنظيف الكود clean project ثم ترجمـة الكود من خلال Build Project من أجـل توليـد ملف hex. ومن ثم برمجتـه بـالمتحكم stm32f103c4 ضمن بيئة المحاكاة proteus

الخطوة السـابعة اسـتخدام البورد التطويري: قم بإعادة نفس الخطوات السـابقة ولكن هذه المرة قم باختيار المتحكم stm32G0B1CEU وفي آخر خطوة قم برفع الكود للبورد التطويري من خلال نافذة الـ debug

التطبيق العملي4: عداد زوار و إظهار العدد على شاشة 7-seg بخانتين ,حيث ان الكباسين موصلين على اقطاب المقاطعة الخارجية (INT0,INT1) ويتم تفعيل المقاطعتين عند الجبهة الهابطة ، وذلك باستخدام بيئة المحاكاة proteus ثم باستخدام البورد التطويري

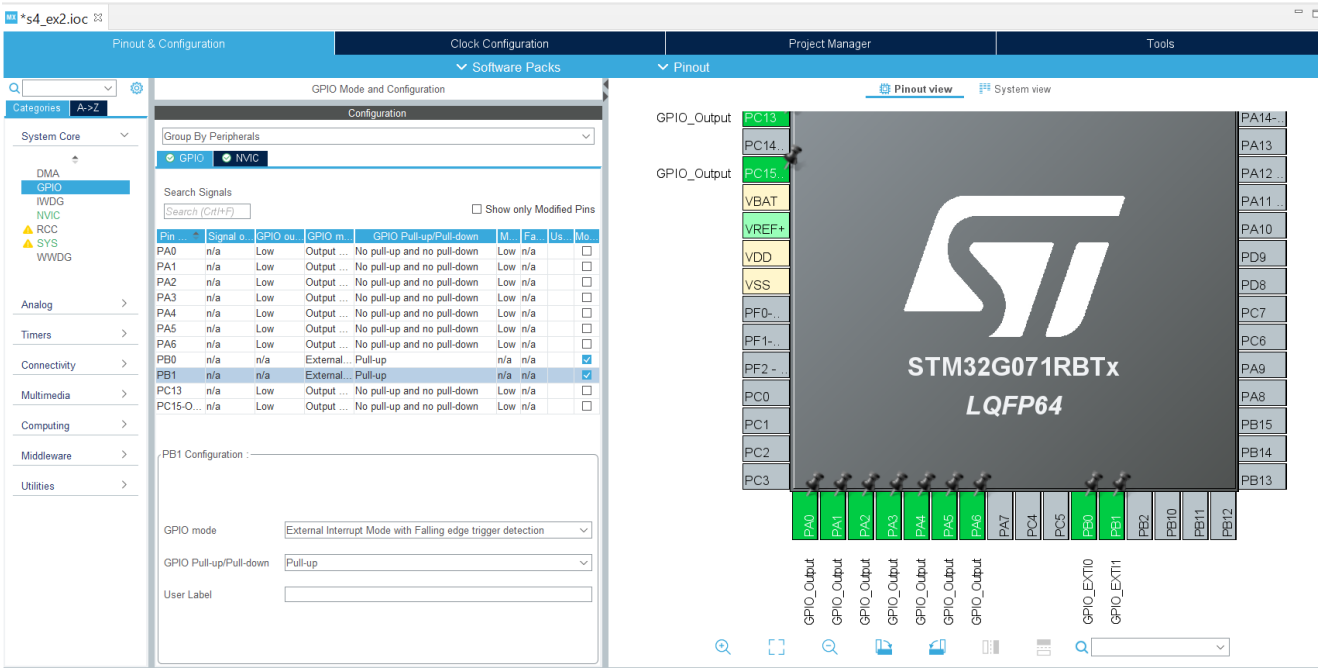


ضبط إعدادات المشروع:

الخطوة الأولى: فتح بيئة STM32CubeIDE وإنشـــاء مشـــروع جديد ثم اختيار المتحكم المناســـب (STM32F103C4) عند اســـتخدام المحاكاة ، و المتحكم STM32G0 عند التطبيق العملي على البورد التطويري

الخطوة الثانية: اختيار اسم للمشروع

الخطوة الثالثة: اختر الأقطاب PA0:PA6 لضبطها كأقطاب خرج



الخطوة الرابعة: قم بضبط ساعة النظام على الساعة الداخلية(HSI) واختر التردد 8MHZ

الخطوة الخامسة: قم بتوليد الكود اعتماداً على الإعدادات التي قمت بضبطها من خلال الضغط على زري Ctrl+S

يصبح الكود النهائي:

```
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
int8_t i=0;
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin ==GPIO_PIN_0)
    {
        i++;
        if(i==100)
         i=99;
    }
    else  if(GPIO_Pin ==GPIO_PIN_1)
       {
     if(i>0)
        i--;
        }
 }

int main(void)
```

```
{
    HAL_Init();
   SystemClock_Config();
   MX_GPIO_Init();
  /* USER CODE BEGIN WHILE */
   unsigned char
decode[10]={0x003f,0x0006,0x005B,0x004f,0x0066,0x006d,0x007d,0x0007,
0x007f,0x006f};
   unsigned char a,b;
  while (1)
  {
        a=i%10;
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_SET);
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_15, GPIO_PIN_RESET);
        GPIOA->ODR = decode[a]; //Displaying ones
        HAL_Delay(5);
        b=i/10;
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_15, GPIO_PIN_SET);
        GPIOA->ODR = decode[b]; //Displaying tens
        HAL_Delay(5);
  }

  /* USER CODE END WHILE */
}
```

الخطوة السادسة: التأكد من خلو الكود من أي أخطاء برمجية من خلال تنظيف الكود clean project ثم ترجمة الكود من خلال Build Project من أجل توليد ملف hex. ومن ثم برمجته بالمتحكم stm32f103c4 ضمن بيئة المحاكاة proteus

الخطوة السابعة استخدام البورد التطويري: قم بإعادة نفس الخطوات السابقة ولكن هذه المرة قم باختيار المتحكم stm32G0B1CEU وفي آخر خطوة قم برفع الكود للبورد التطويري من خلال نافذة الـ debug