点亮 LED

编译：netlhx

本教程以 ST 的 NUCLEO F072RB 为硬件平台，结合 STM32CubeMX 及 MDK 来学习 STM32。为了更好的学习，建议从 ST 官网下载如下资料备查：

- NUCLEO F072RB 原理图：MB1136.PDF
- NUCLEO F072RB 数据手册：DM00090510.PDF
- NUCLEO F072RB 参考手册：DM00031936.PDF

软件版本如下：

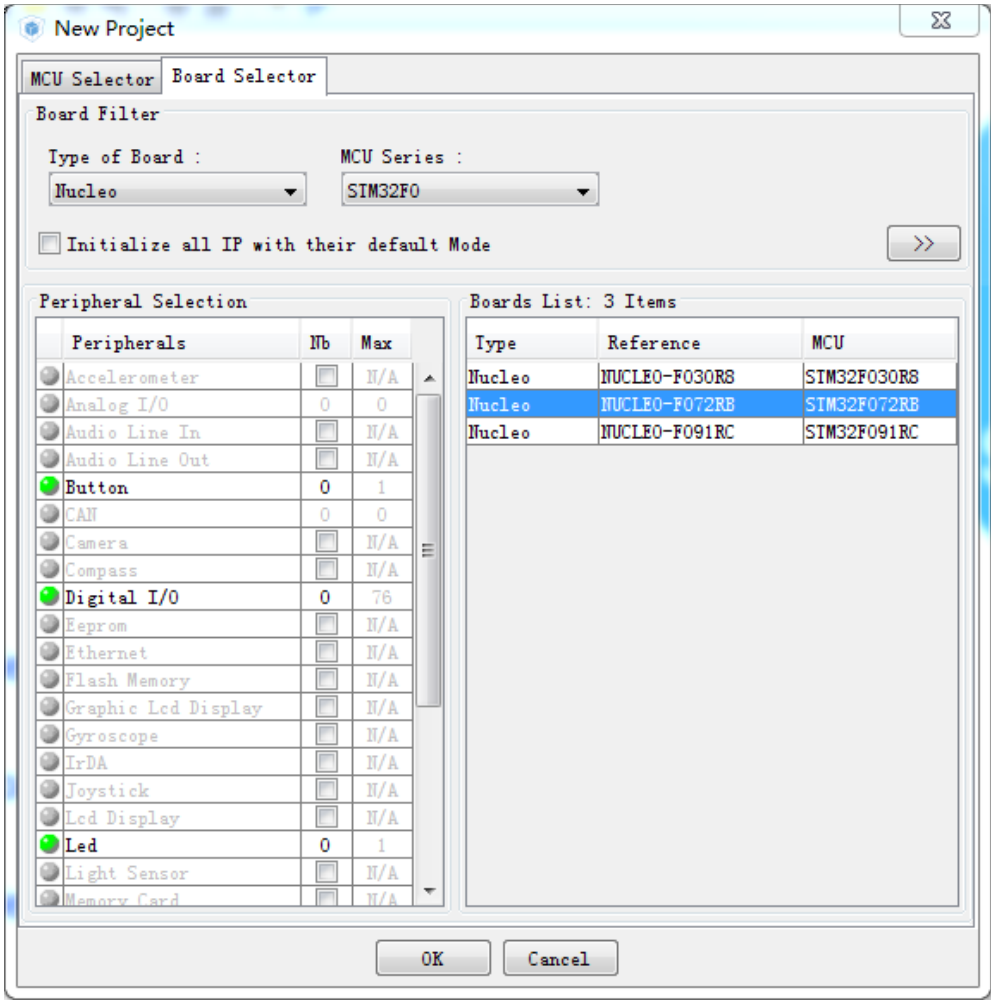
- MDK 4.74
- STM32CubeMX 4.5

创建工程

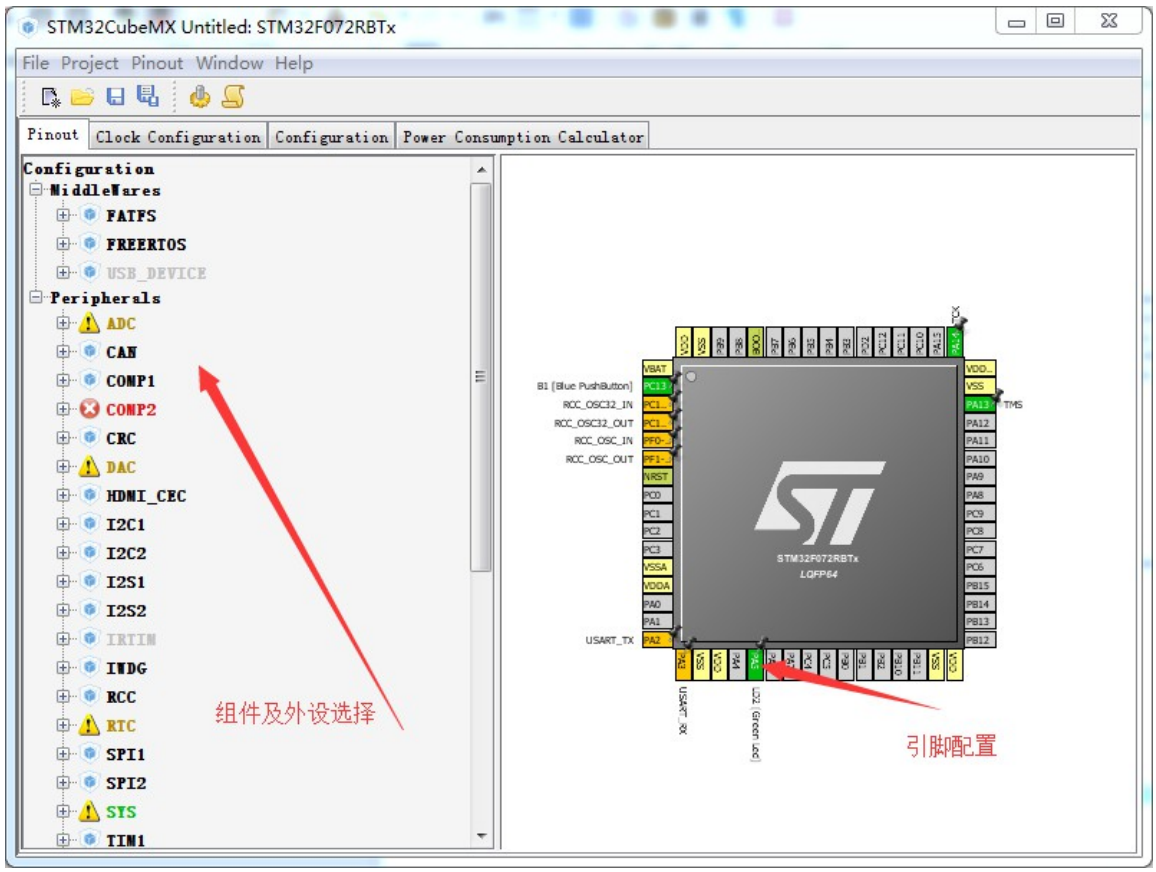
STM32CubeMX 是 ST 推出的一款图形化编程工具，其目的是更好的解放程序员。

选择板型

启动 STM32CubeMX，点击“New Project”，按下图所示选择好开发板类型：



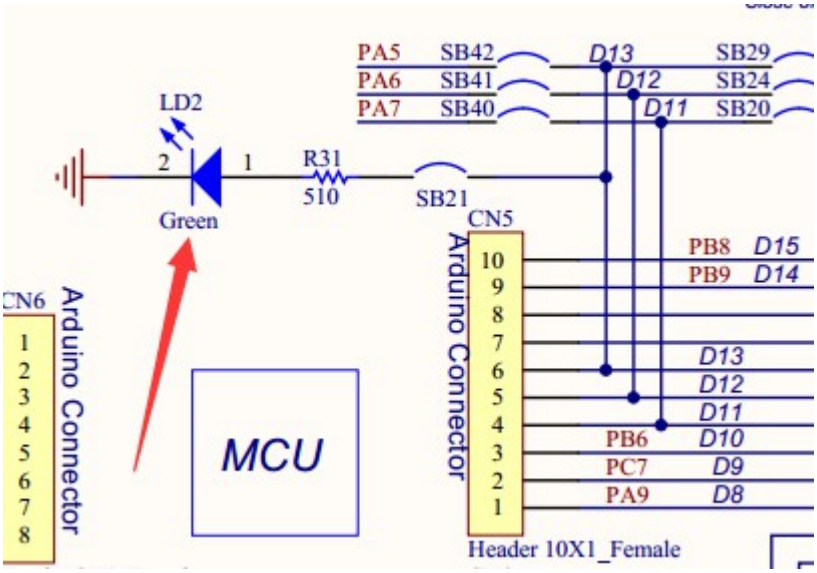
接下来出现的是 STM32CubeMX 的外设及引脚配置窗口。



这里，左边是组件及外设选择设置，通过选择并启用相应的组件或外设来打开 MCU 上的相应功能；右边是引脚选择及配置，很多外设都有对应的复用引脚，可以直接在引脚上设定相应功能。

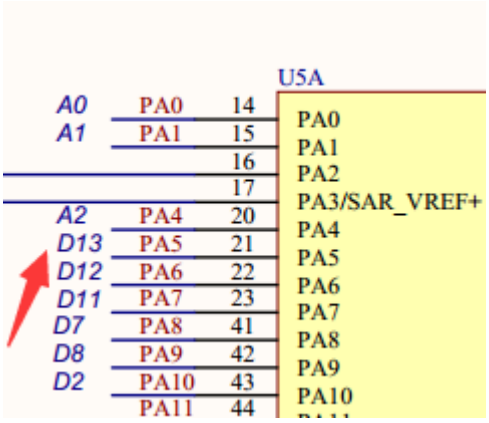
选择组件及外设

本文的任务是点个 LED 灯，所以要操作的直接对象就是 GPIO 口。打开板子原理图吧，找啊找……



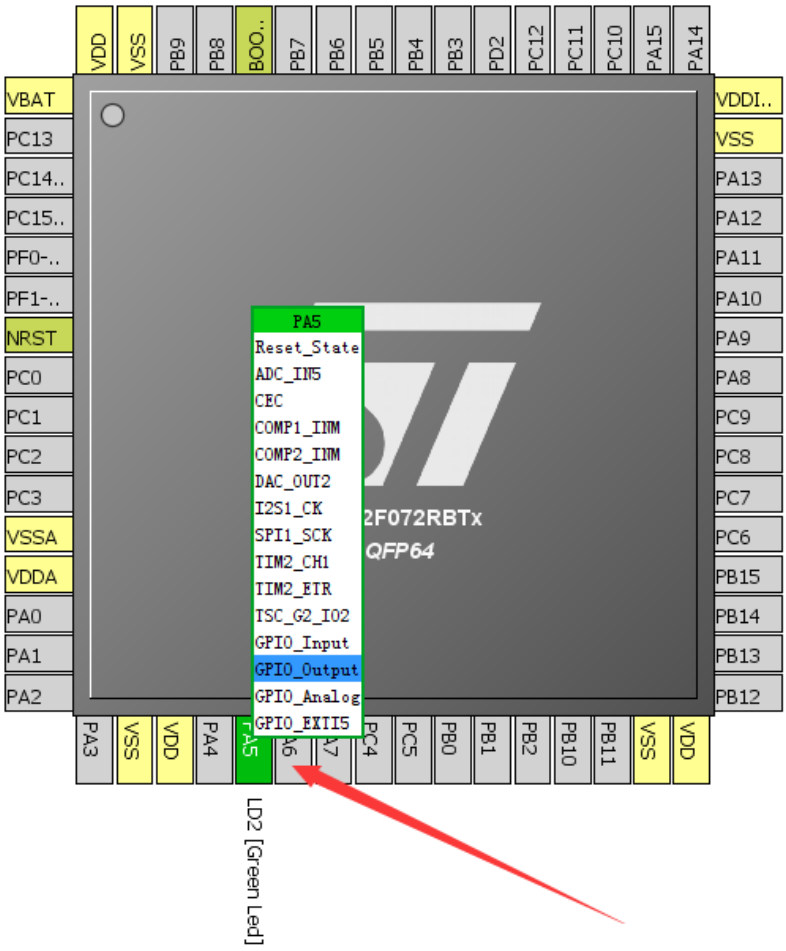
板子上的 LD2 就是我们要操作的指示灯，对应哪个 GPIO 引脚呢？

LD2 对应的网络号为 D13，再到 MCU 上去找 D13 吧，啊，终于找到你了，原来是 PA5。



要知道，ST32 系列 MCU 为了最大限度减少能耗，将不需要的外设及引脚电源都是关闭了的，所以要使 GPIO 工作，需要配置 RCC，即复位与时钟控制系统。需要注意的是，NUCLEO F072RB 是默认是没有外部晶振的，如果要使用的话，需要自己焊接。所以本文使用的是 HSI 时钟，这个属于内部资源，开机就可以工作。

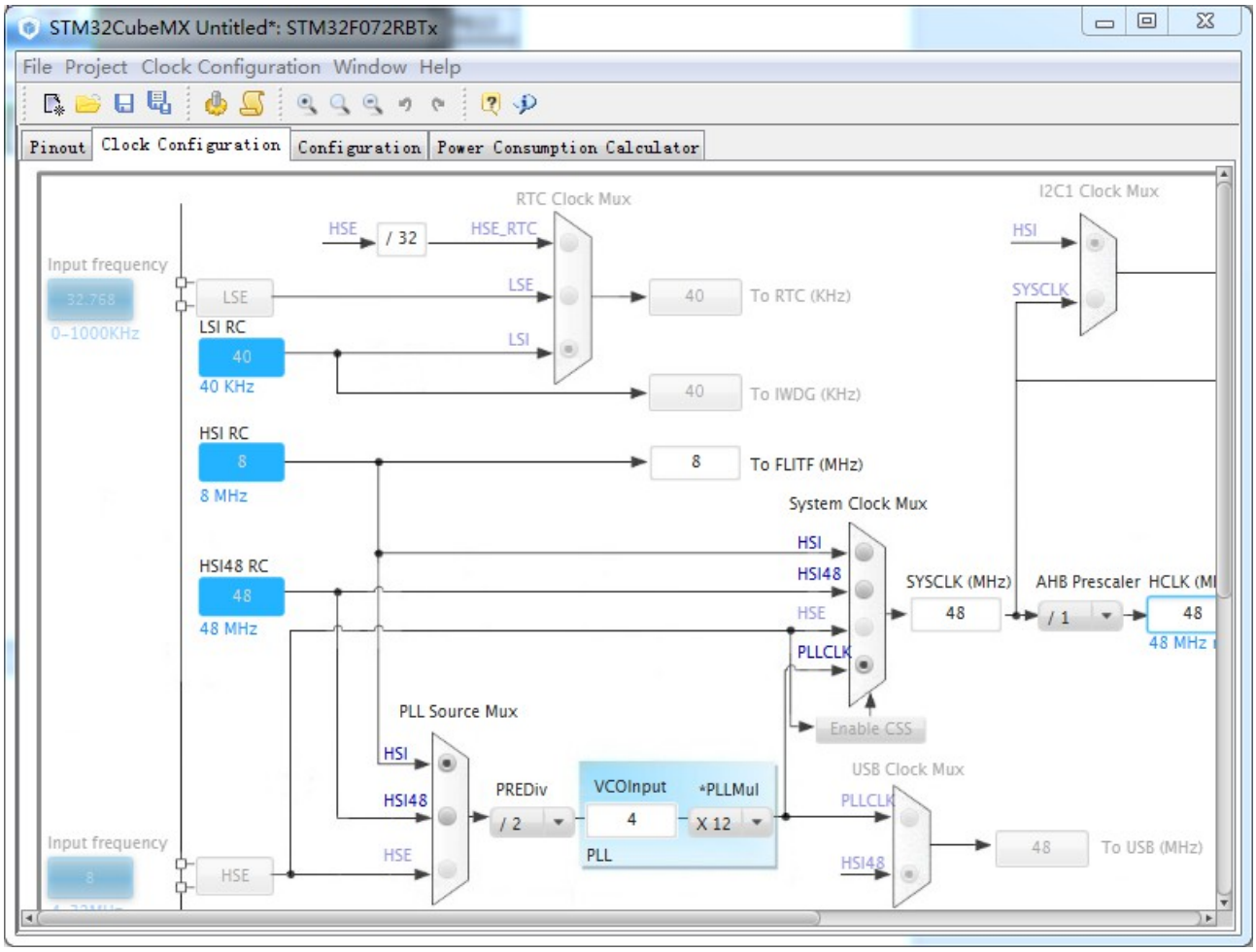
所以实际上，要点亮 LED 灯，只要配置 GPIO 即可。在引脚上单击可以配置引脚的工作模式，点亮 LED 属于 GPIO 的输出功能，所以这里选择 GPIO_Output。



配置时钟

STM32CubeMX 的第二个配置选项卡是时钟选择与配置。

STM32 支持使用内部或外部时钟源，前面已说过，NUCLEO F072RB 没有焊接外部时钟晶振，所以只有 HSI 可以配置，HSI 的频率是固定的 8M，所以我们能够修改的也只有分频系统几个有限的可以配置，这里就用默认的，不修改了。



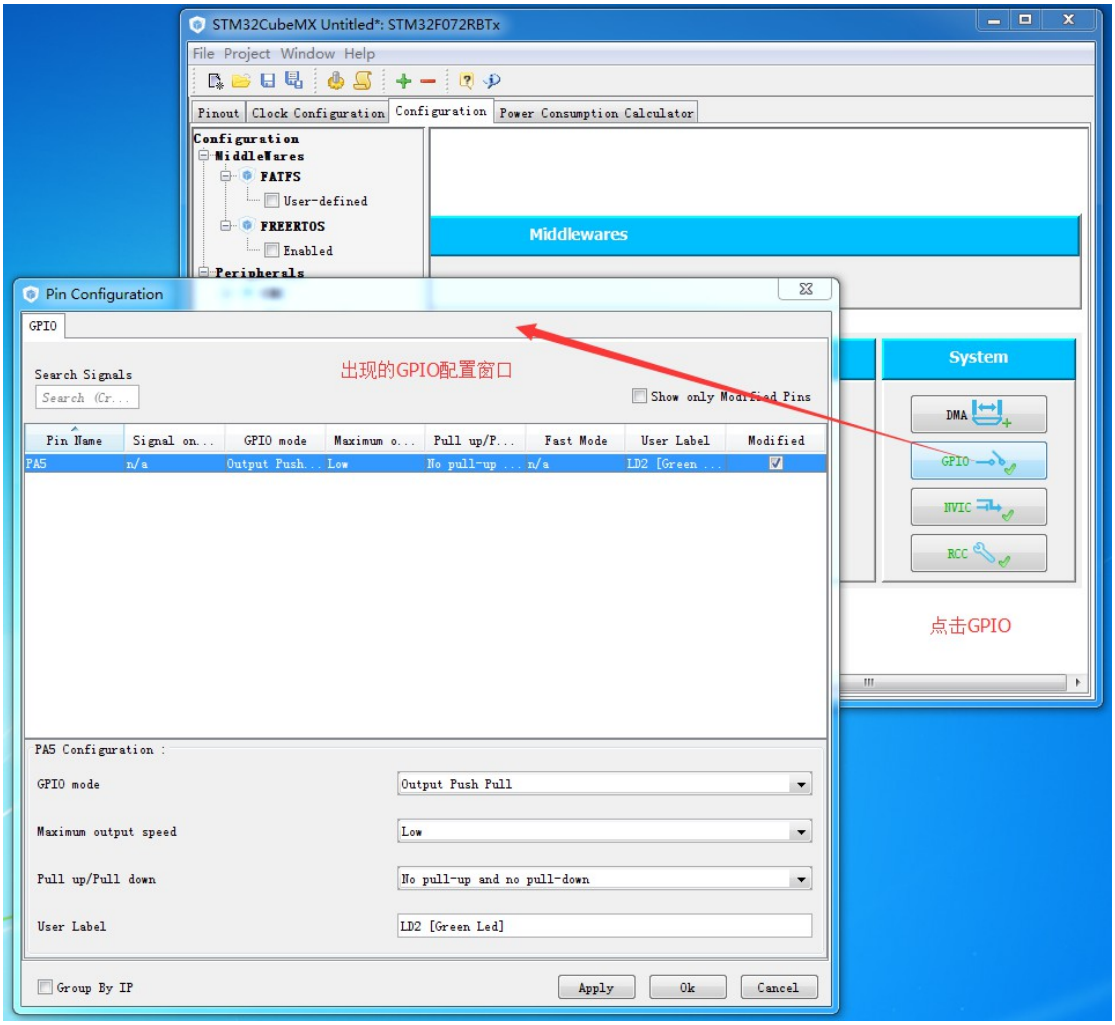
配置外设及引脚

TM32CubeMX 的第三个配置选项卡是配置外设的具体参数。

单击 GPIO 按钮，出现 GPIO 配置参数，里面包括下面一些内容：

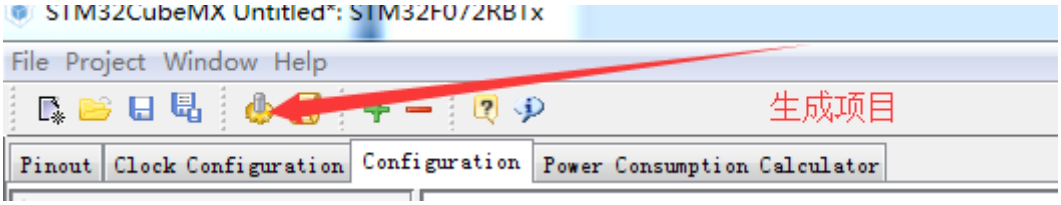
- GPIO 工作模式：输出还是输入
- GPIO 工作最大速度：决定能耗的多少
- 上下拉：是否启用内部下、下拉功能
- 用户标签：标注之用，无实际意义

关于 GPIO 的工作模式及上下拉等，这里就不解释了……。



好吧，配置得差不多了，记得保存。

生成项目



点击生成项目按钮，弹出设置对话框。

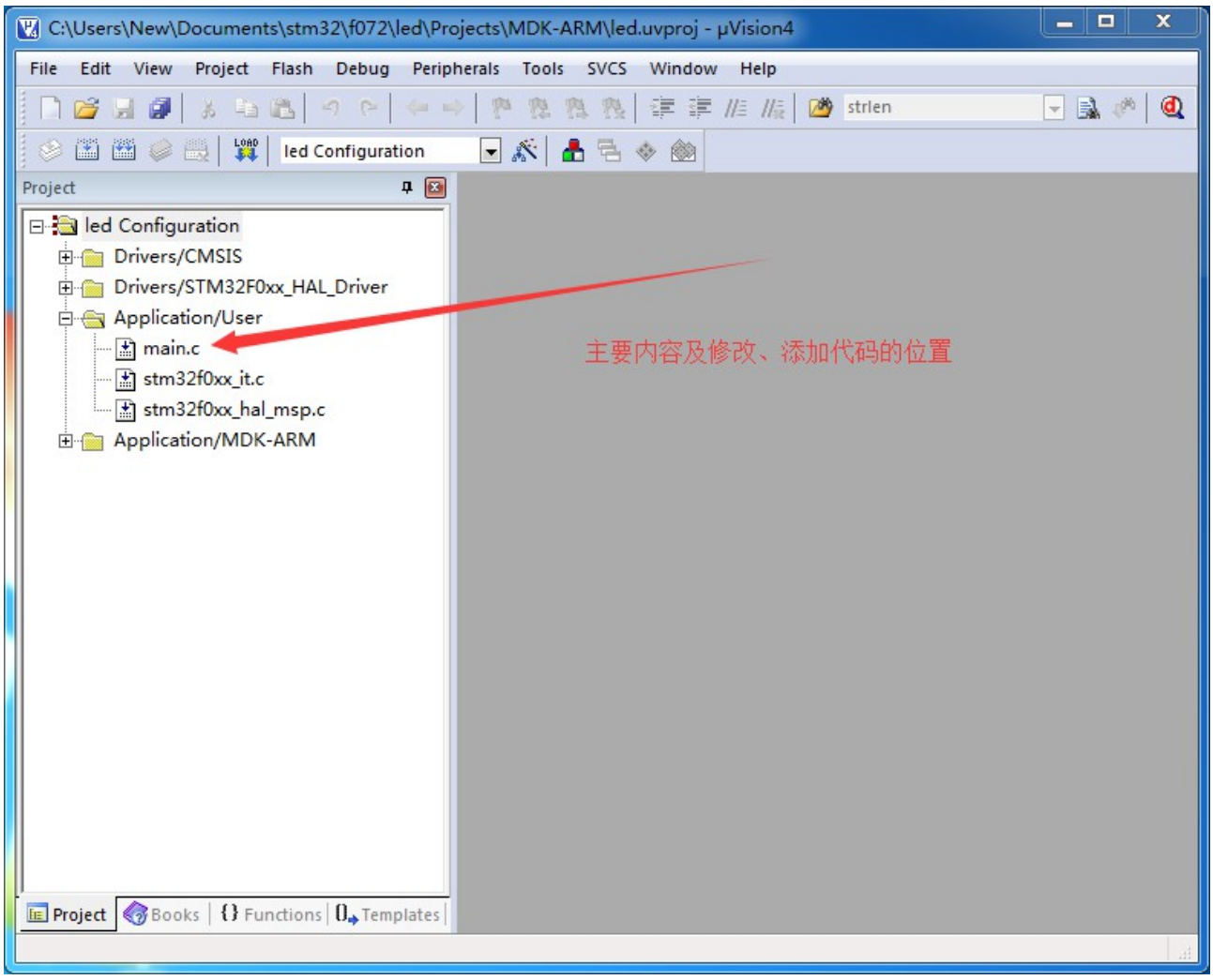
设置的内容包括：

- 项目名称，比如 LED
- 项目位置，项目文件的位置
- 项目类型：MDK 还是 IAR 等。

项目创建好之后，会提示你是否打开当前创建的项目。

代码添加

打开项目文件后的样子



项目文件构成

自动生成的项目文件，里面包含很多的内容，就用户而言，我们需要关心如下几个：

- main.c：主函数文件，里面包含一些常规的初始化代码，这是 STM32CubeMX 根据我们指定的参数自动创建的代。
- stm32f0xx_it.c：中断服务子程序 ISR 所在位置，如果要写中断代码，一般放到这里面。
- stm32f0xx_hal_msp.c：MCU 支持文件，一些 MCU 相关的配置都放到这里面。

主程序文件 main.c 分析

下面是主文件里的部分代码。

```
/* Includes ------------------------------------------------------------------*/
#include "stm32f0xx_hal.h"

/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private variables ---------------------------------------------------------*/

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);

/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

int main(void)
{

  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------
*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick.
*/
  HAL_Init();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
```

```
  MX_GPIO_Init();

  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* USER CODE BEGIN 3 */
  /* Infinite loop */
  while (1)
  {

  }
  /* USER CODE END 3 */

}
```

注意注释里包含 USER CODE 处，这里指明，如果用户要在自动生成的代码里添加自己的功能代码，应该插在这些注释的中间，这样 STM32CubeMX 下次重新生成代码后才不会覆盖这些内容！

尝试在 main.c 中找到并修改如下代码，红色部分是添加的新代码，功能是点亮 LED2。

```
  /* USER CODE BEGIN 2 */
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);

  /* USER CODE END 2 */
```

其它一些函数，从名称上可以大概看出其功能，比如 HAL_Init()是做一些全局的初始化工作，而 SystemClock_Config()的作用则是配置系统时钟。

好了，编译并下载到板子上会发现 LD2 亮了。

下面代码是 LED 闪烁，添加并修改后验证。

```
  /* USER CODE BEGIN 3 */
  /* Infinite loop */
  while (1)
  {
       HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);
       HAL_Delay(500);
       HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
       HAL_Delay(500);

  }
  /* USER CODE END 3 */
```

Over!