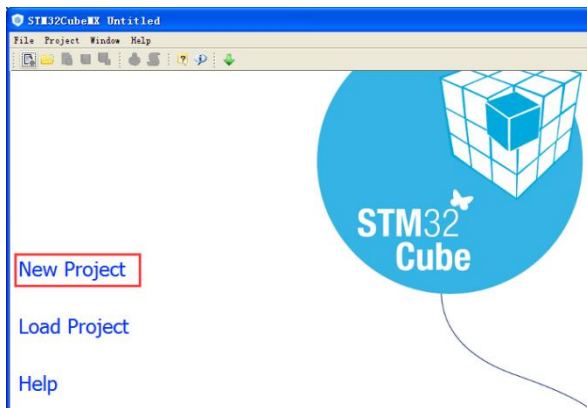


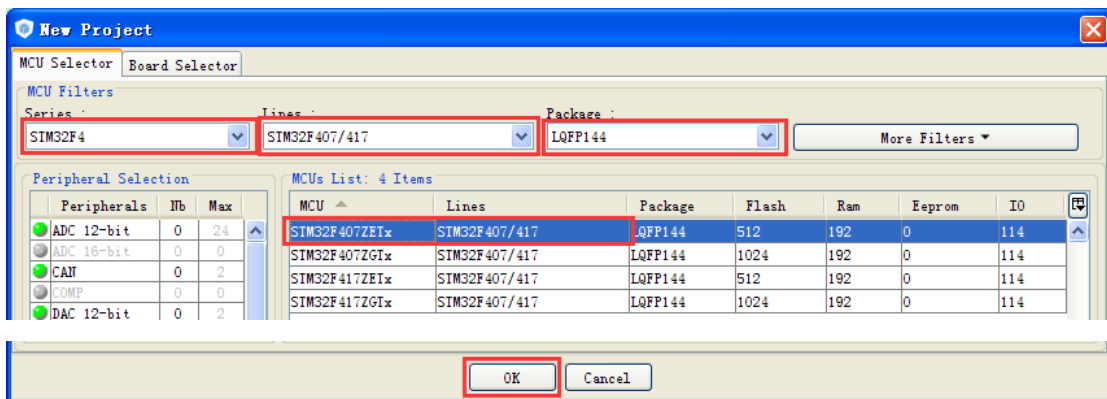
STM32Cube 学习之一：点灯

假设已经安装好 STM32CubeMX 和 STM32CubeF4 支持包。

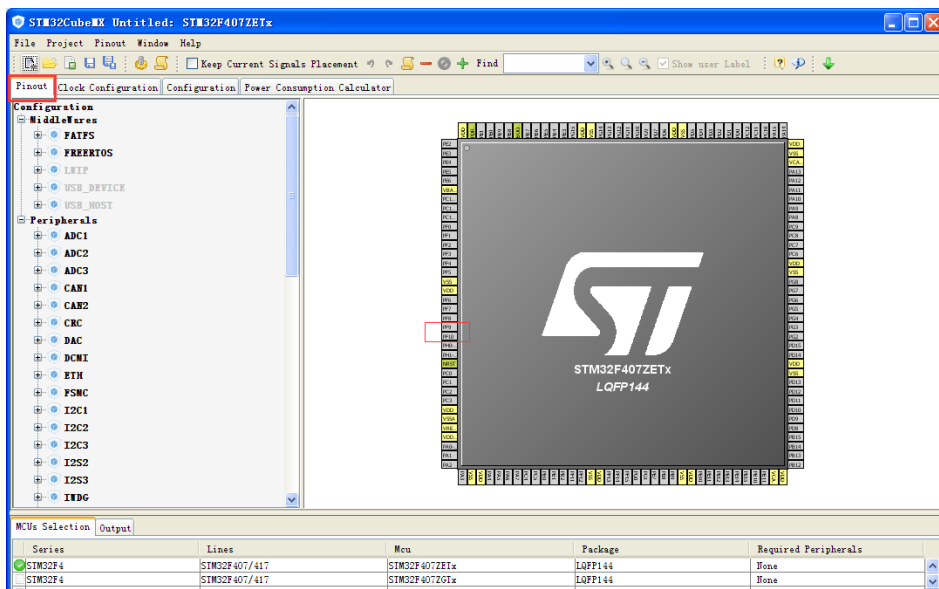
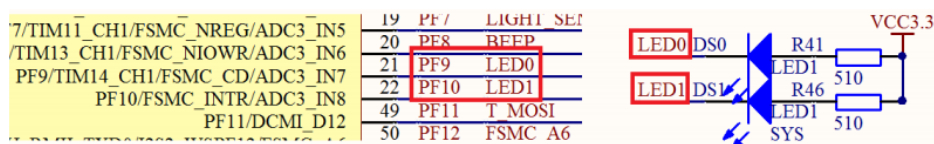
Step1. 打开 STM32CubeMX，点击“New Project”。

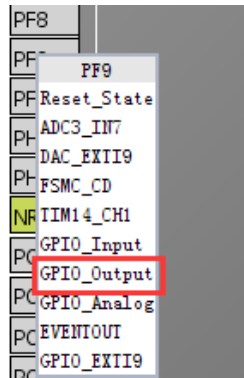


Step2. 选择芯片型号，STM32F407ZETx。

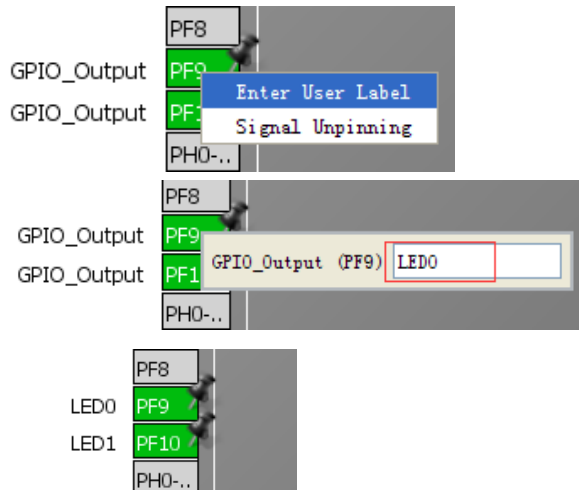


Step3. 在 Pinout 界面配置 GPIO，PF9 和 PF10 为输出。

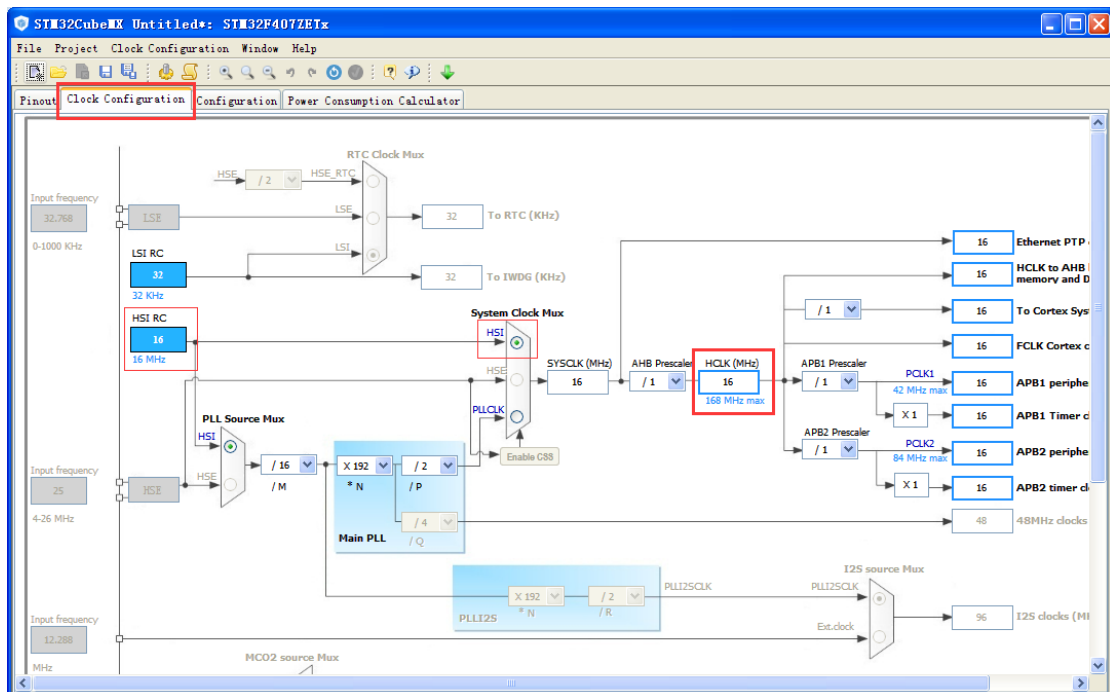




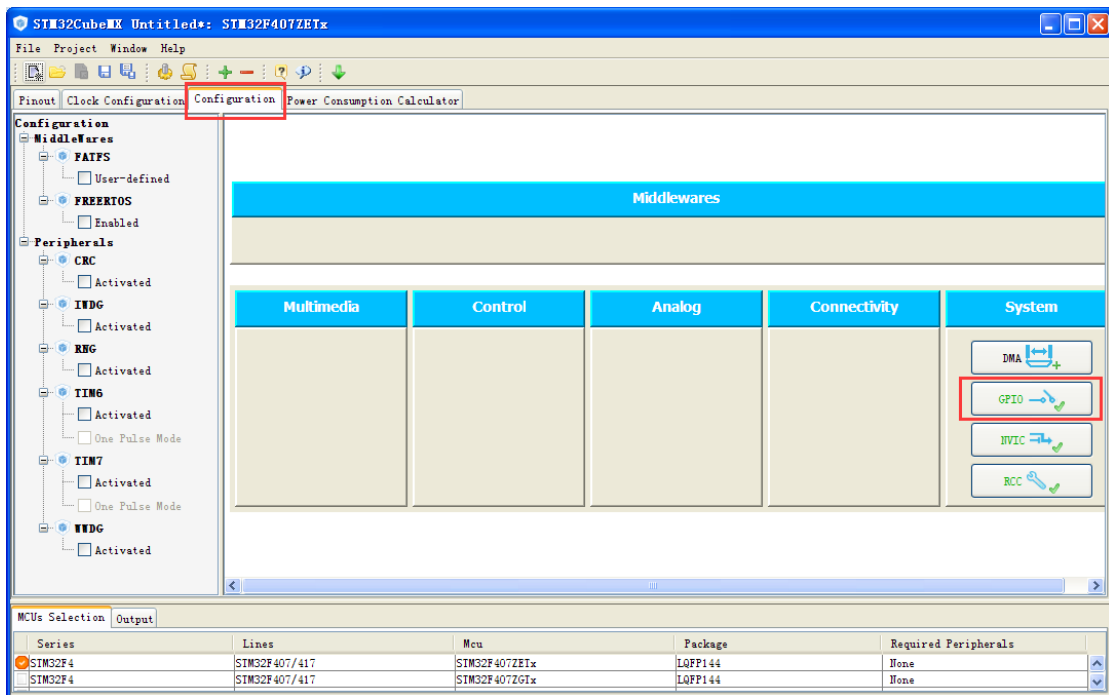
点击右键配置用户标签，分别为 LED0 和 LED1。



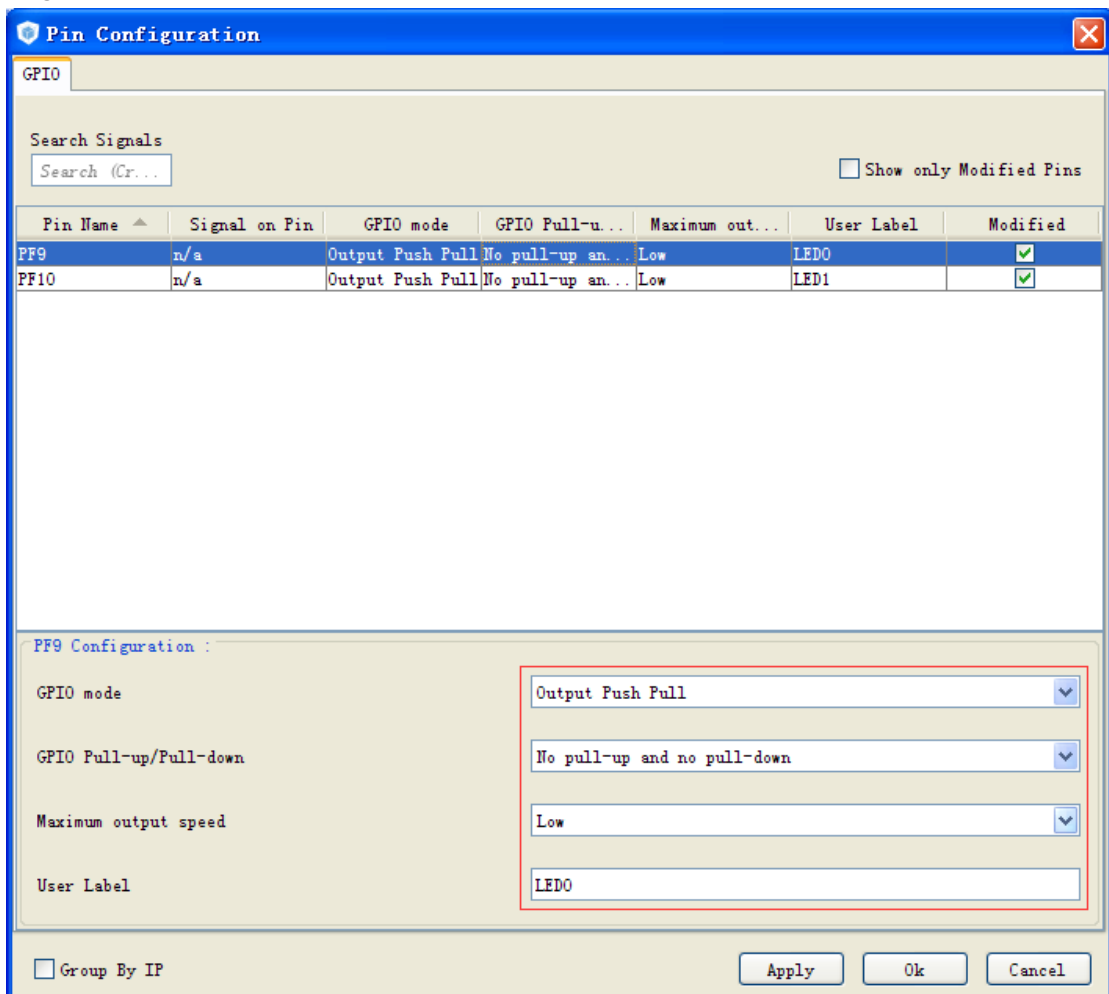
Step4.配置时钟树，在此先使用默认的内部 16M 时钟源，内核时钟 16M。



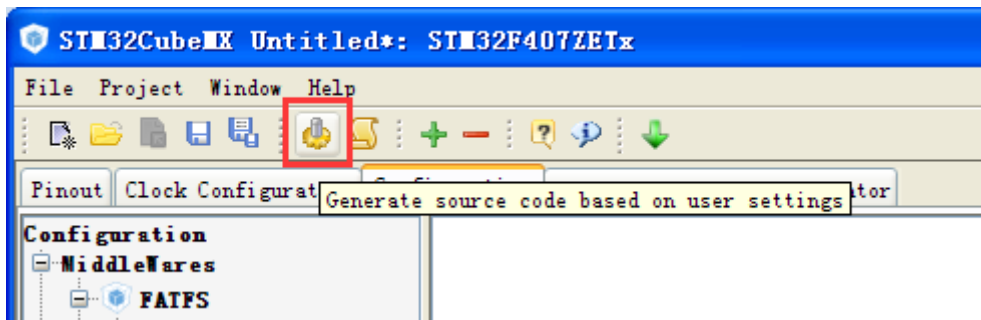
Step5.配置 GPIO 的速度和上拉/下拉电阻。



在 Configuration 界面点击“GPIO”按钮，进入 GPIO Pin 配置界面。

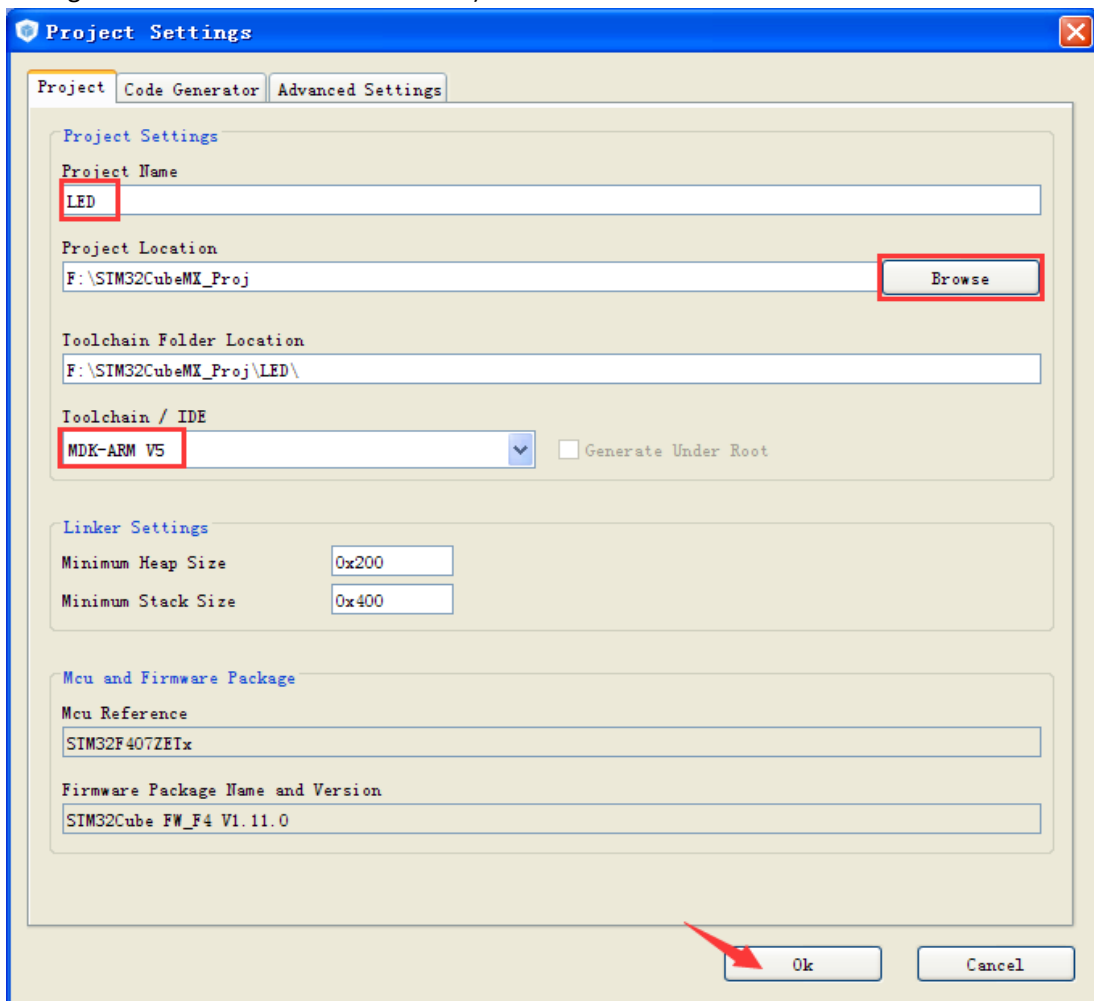


Step6.生成源代码。

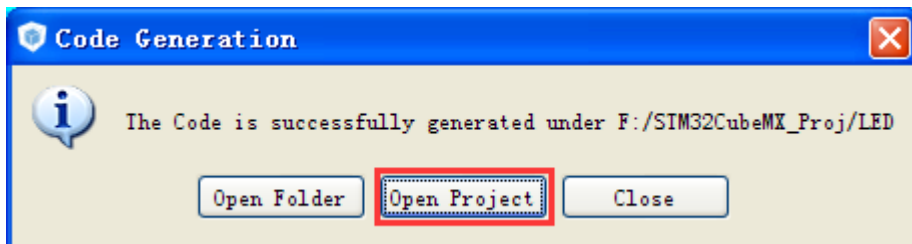


点击创建源代码的工具按钮，进入工程设置。

输入工程名称“LED”，点击“Browse”设置工程保存路径，选择 IDE 为 MDK-ARM V5，点击 OK。注意：一定要保证 CubeMX 和所使用的器件支持包的匹配，否则无法正常生成代码。本例中 CubeMX 是 V4.14，使用的 STM32CubeF4 为 V1.11.0。(注：因为 Cube 可能会更新版本，修正用户使用过程中发现的 Bug，所以建议尽量使用最新版本。)

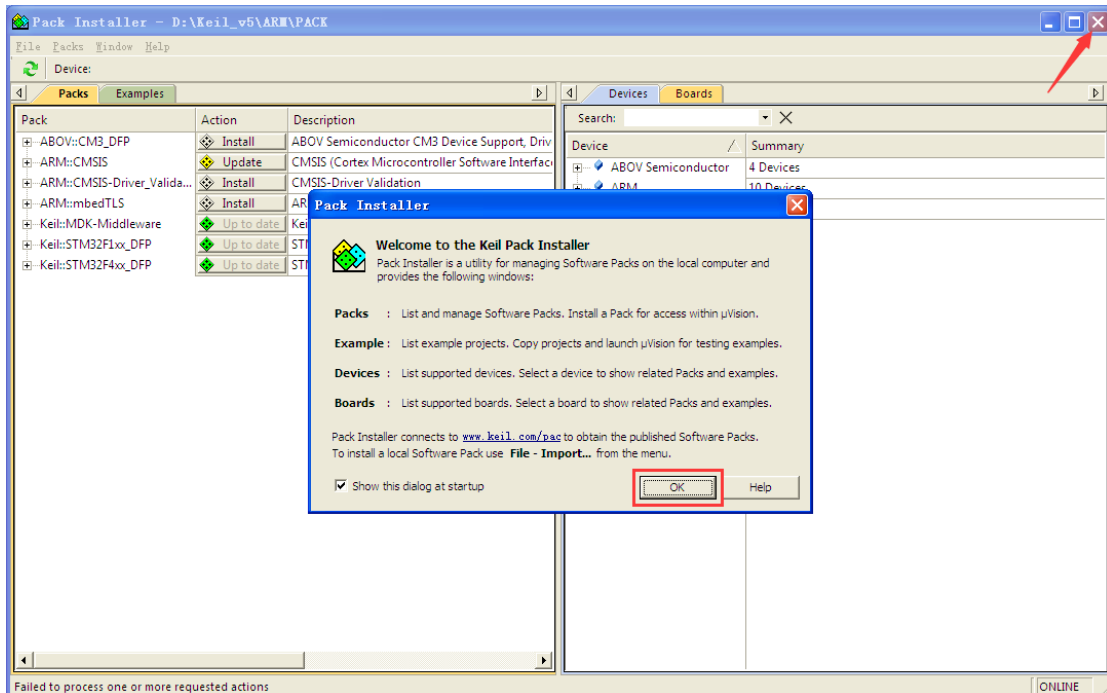


生成源代码完成后，会弹出如下对话框。点击“Open Project”即可在 MDK 中打开工程。



Step7.打开 MDK 工程。

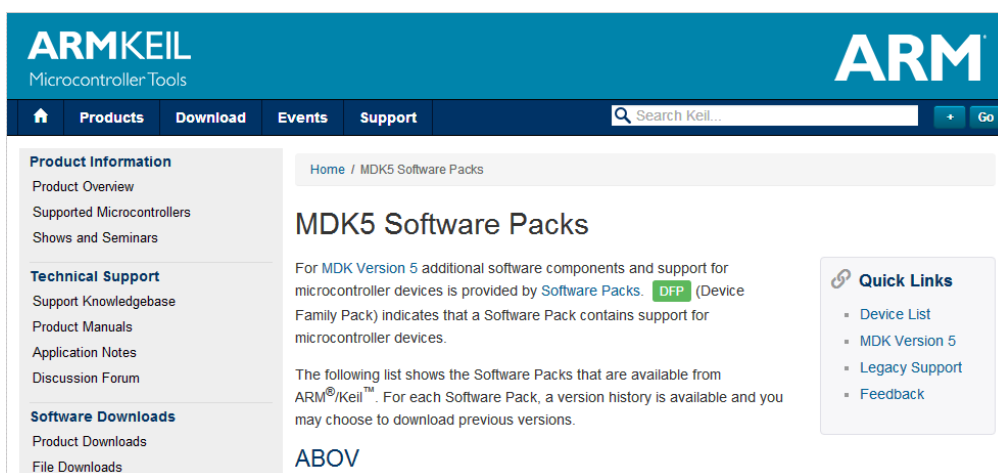
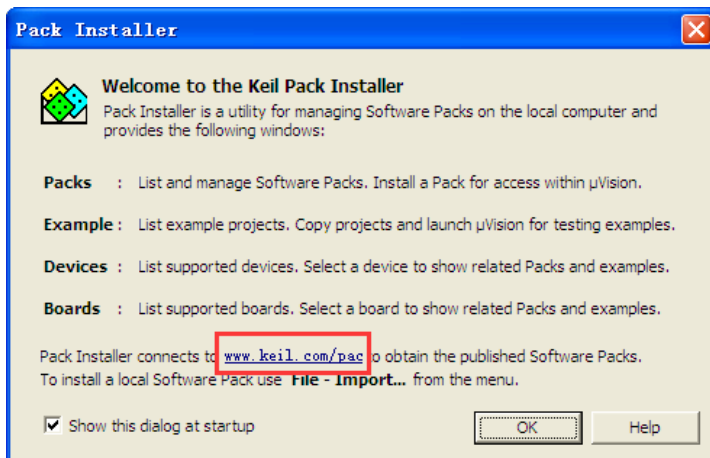
在生成工程后，第一次在 MDK 中打开时，会出现如下对话框。



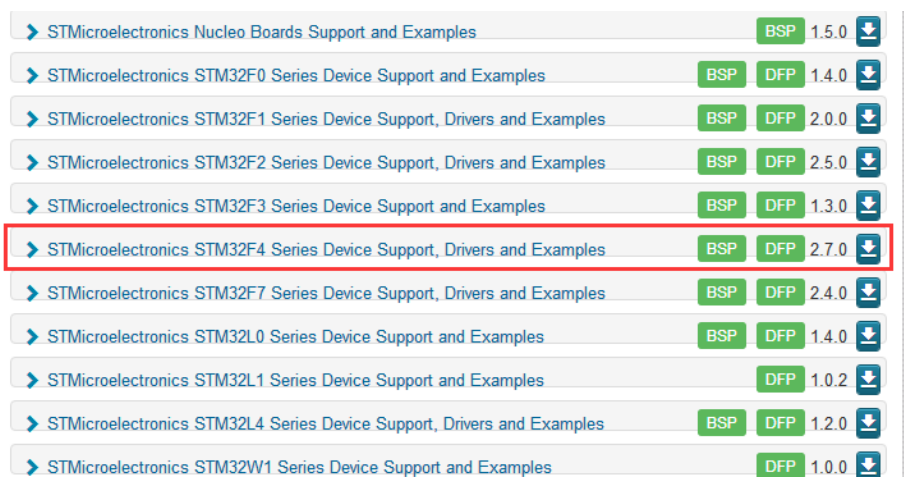
由于 MDK V5 将 IDE 和芯片支持包分开了，要支持芯片就要先安装对应的支持包。

如果已经安装了 STM32F4xx 的支持包，直接点击 OK 和关闭按钮关闭上面的两个窗口。

如果没有安装，这点击下图中的链接，进入 keil 官网的支持包网页进行下载。



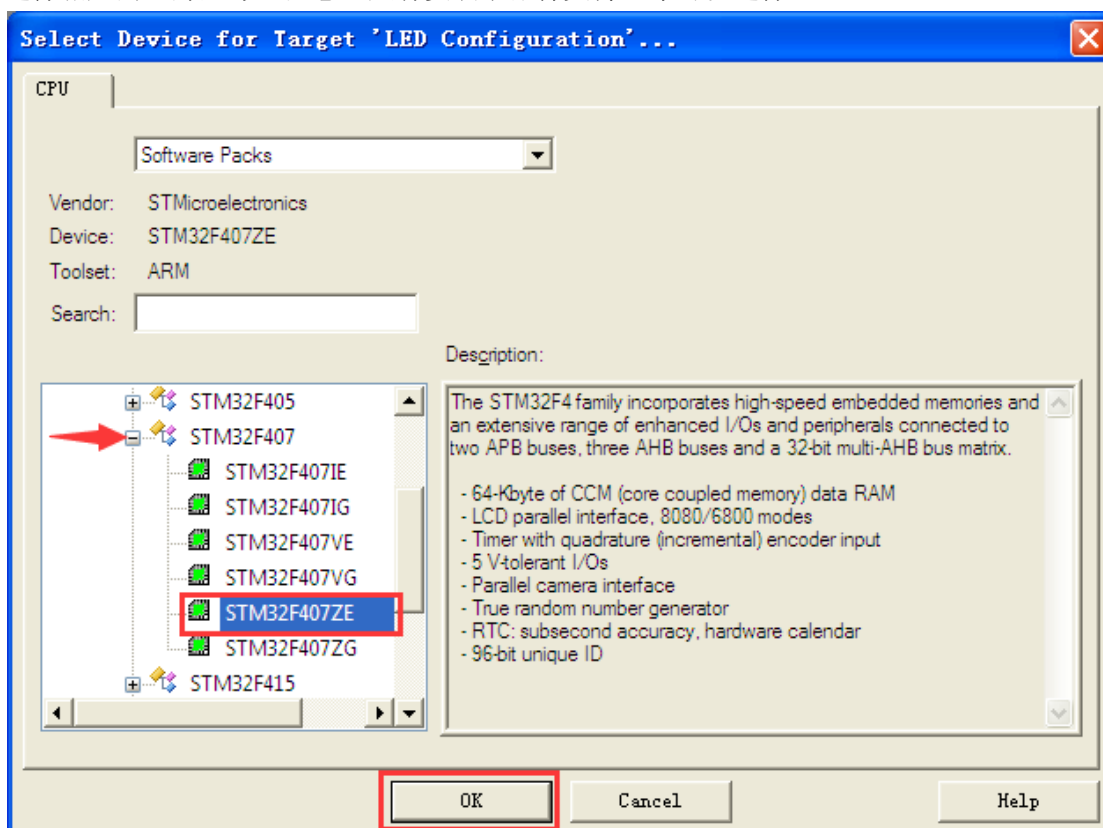
在网页中找到 STM32F4 系列器件的支持包下载，下载完成之后直接双击安装即可。



关闭 Step7 中遇到的两个窗口后，还会弹出如下窗口，点击“是”。

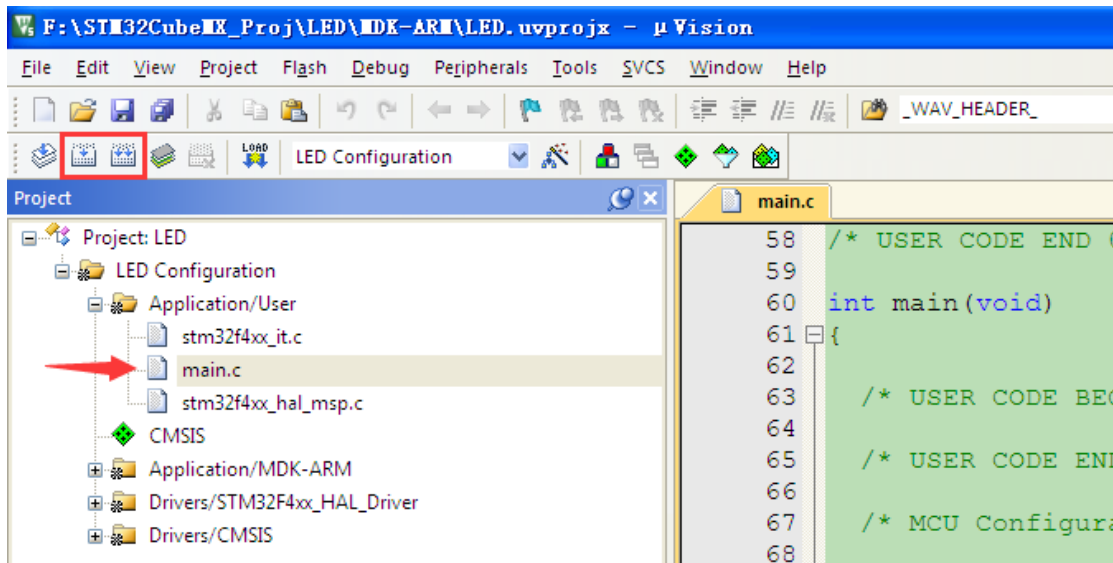


然后选择相应的芯片型号。注意：只有安装了器件支持包才可以选择。

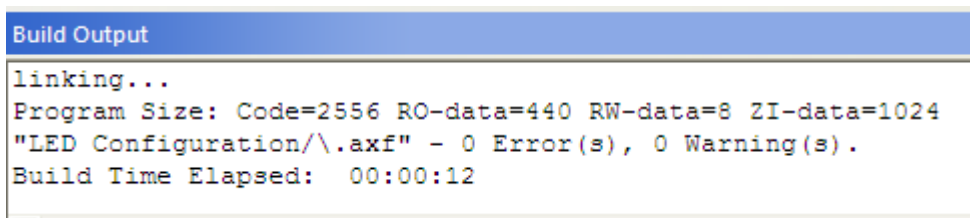


Step8.编译工程。

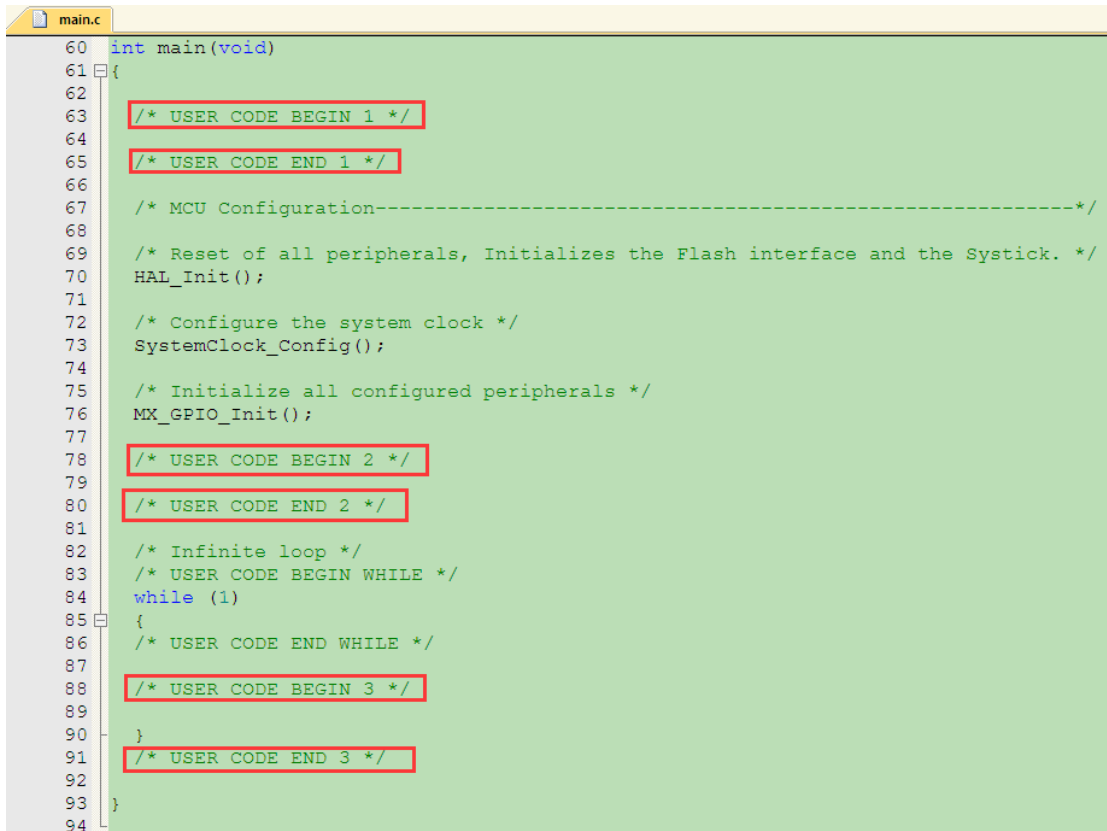
点击 MDK 中的编译或者全部编译按钮，对工程进行编译。



0 错误, 0 警告。

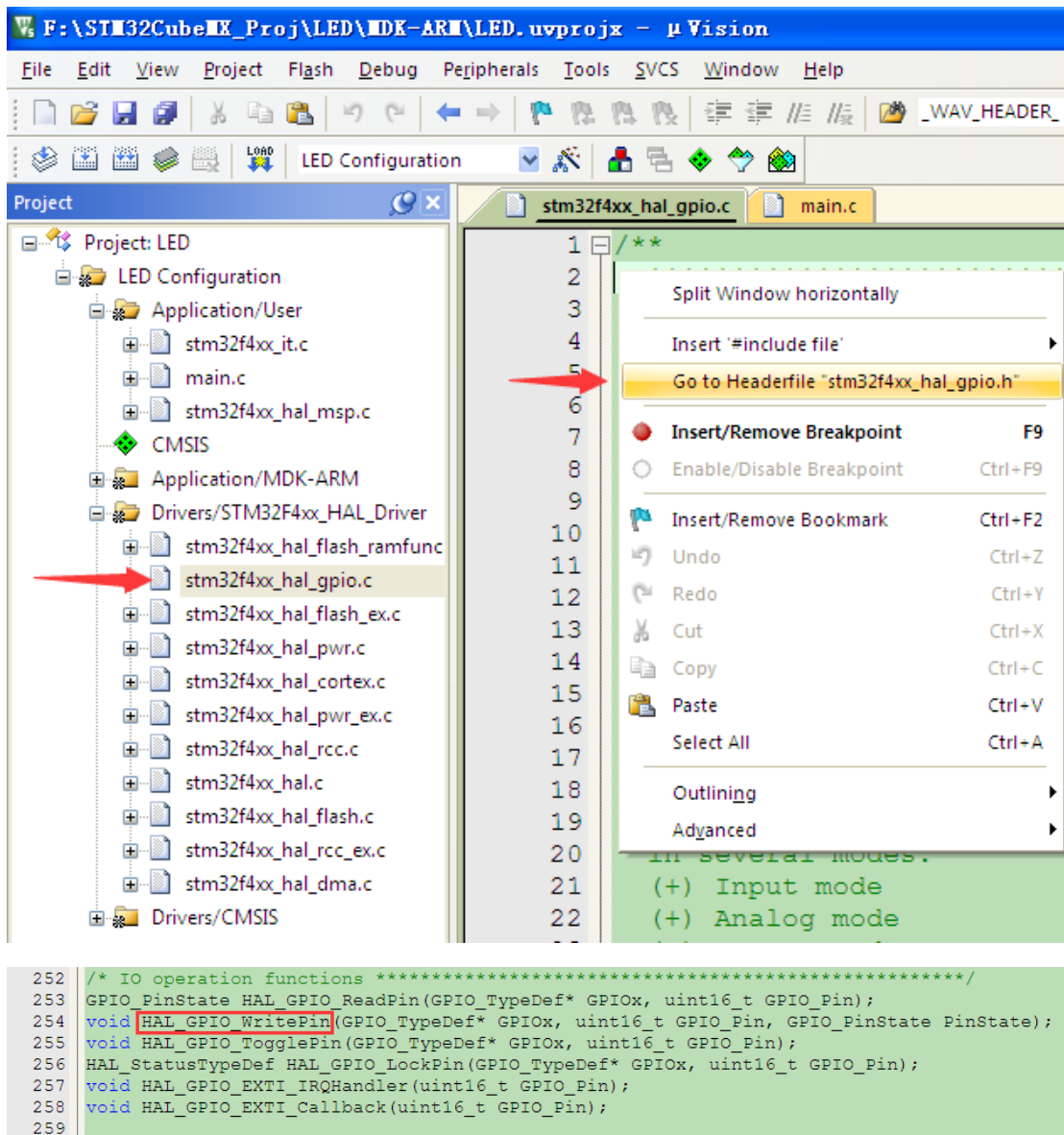


Step9.添加用户功能代码。

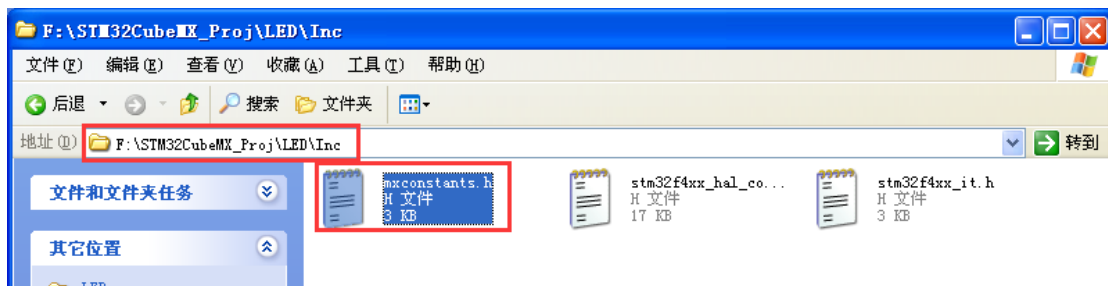


在 CubeMX 生成的文件中添加用户代码的时候, 必须是写在/* USER CODE BEGIN n*/和/* USER CODE

END n*/之间。这样如果需要改变 Cube 工程的配置，在重新生成代码时，在这两句注释之间的语句不会被覆盖。而用户新建或添加的文件不会受到影响。



打开 stm32f4xx_hal_gpio.c，并通过在该文件中点击右键，选择 Go to Headfile...可以打开相应的头文件。然后可以找到 GPIO 操作的 HAL 库函数，HAL_GPIO_WritePin()。复制到 main 文件的 while(1)中/* USER CODE BEGIN 3*/之后。



CubeMX 还生成了一个文件 mxconstants.h，包含了用户配置 GPIO 的宏定义。在 GPIO 操作时，直接使用即可。


```

39  /* Private define -----
40
41  #define LED0_Pin GPIO_PIN_9
42  #define LED0_GPIO_Port GPIOF
43  #define LED1_Pin GPIO_PIN_10
44  #define LED1_GPIO_Port GPIOF
45  /* USER CODE BEGIN Private defines */
46
47  /* USER CODE END Private defines */

```

在 while(1)中添加如下用户代码，实现的功能是 PF9 和 PF10 引脚上的两个 LED 轮流点亮，周期是 2 秒。

```

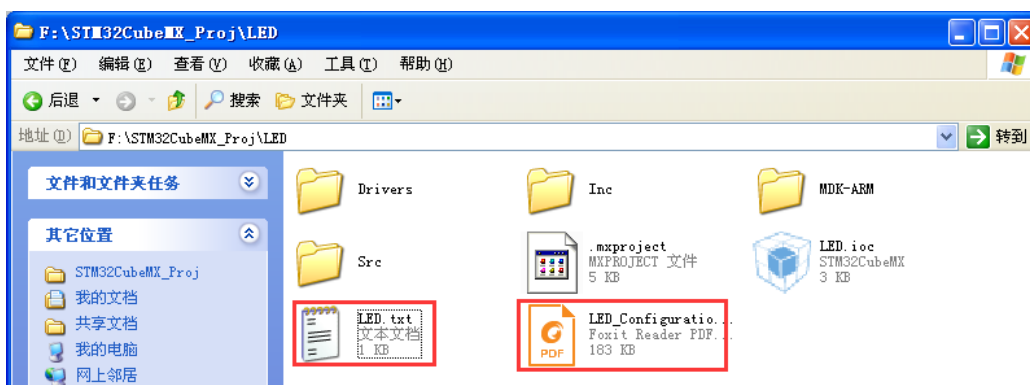
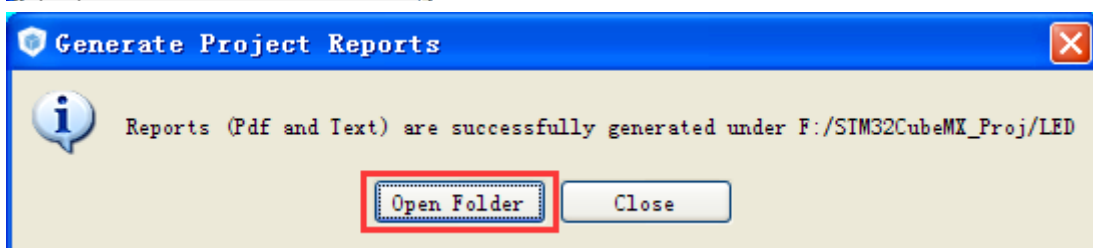
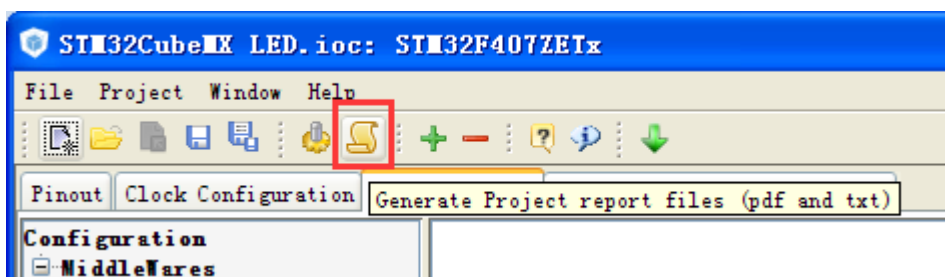
84  while (1)
85  {
86      /* USER CODE END WHILE */
87
88      /* USER CODE BEGIN 3 */
89      HAL_GPIO_WritePin(LED0_GPIO_Port, LED0_Pin, GPIO_PIN_RESET);
90      HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_SET);
91      HAL_Delay(1000);
92      HAL_GPIO_WritePin(LED0_GPIO_Port, LED0_Pin, GPIO_PIN_SET);
93      HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_RESET);
94      HAL_Delay(1000);
95  }
96  /* USER CODE END 3 */

```

其中，HAL_Delay()函数是 HAL 库的延时函数，单位是 1 毫秒。

至此，点灯实验完成。

另外，点击 CubeMX 的生成报表工具，可以生成相关的配置报表。一个是 txt 文件，一个是 pdf，其内容可到工程目录查看。



另外，在使用 CubeMX 开发 STM32 的时候，可以参考官方提供的例程。
到官网下载芯片的 Cube 支持包，如 F4 的是 stm32cubef4.zip，解压之后得到 STM32Cube_FW_F4_V1.11.0 文件夹。

共享 新建文件夹			
名称	修改日期	类型	大小
STM32Cube_FW_F4_V1.11.0	2016/2/3 0:03	文件夹	
stm32cubef4.zip	2016/3/31 20:51	WinRAR ZIP 压缩...	317,570 KB

打开 STM32CubeF4 V1.11\STM32Cube_FW_F4_V1.11.0\Projects，该文件夹下是官方出的各种开发板的参考例程，及一个工程列表说明网页文件。

共享 新建文件夹			
名称	修改日期	类型	大小
STM32F4-Discovery	2016/2/2 23:53	文件夹	
STM32F401-Discovery	2016/2/2 23:54	文件夹	
STM32F401RE-Nucleo	2016/2/2 23:54	文件夹	
STM32F410xx-Nucleo	2016/2/2 23:55	文件夹	
STM32F411E-Discovery	2016/2/2 23:56	文件夹	
STM32F411RE-Nucleo	2016/2/2 23:56	文件夹	
STM32F429I-Discovery	2016/2/3 0:00	文件夹	
STM32F429ZI-Nucleo	2016/2/3 0:01	文件夹	
STM32F446ZE-Nucleo	2016/2/3 0:03	文件夹	
STM324x9I_EVAL	2016/2/2 23:45	文件夹	
STM324xG_EVAL	2016/2/2 23:51	文件夹	
STM32446E_EVAL	2016/2/2 23:30	文件夹	
STM32446E-Nucleo	2016/2/2 23:26	文件夹	
STM32469I_EVAL	2016/2/2 23:37	文件夹	
STM32469I-Discovery	2016/2/2 23:32	文件夹	
WIN32	2016/2/3 0:03	文件夹	
STM32CubeProjectsList.html	2016/2/2 20:09	360 se HTML Do...	159 KB

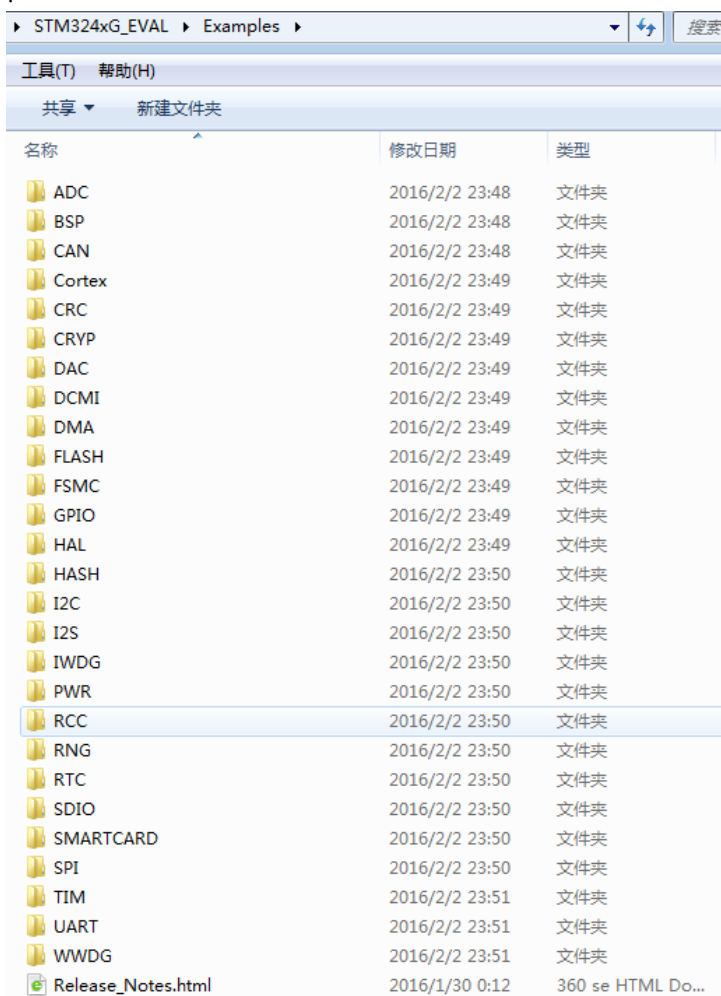
例如，打开 STM324xG_EVAL 文件夹，里面是 STM324xG 评估板的例程。

STM32Cube_FW_F4_V1.11.0 ▸ Projects ▸ STM324xG_EVAL ▸			
工具(T) 帮助(H)			
共享 新建文件夹			
名称	修改日期	类型	
Applications	2016/2/2 23:47	文件夹	
Demonstrations	2016/2/2 23:48	文件夹	
Examples	2016/2/2 23:51	文件夹	
Templates	2016/2/2 23:51	文件夹	

其中 Applications 文件夹中，是 FatFs、FreeRTOS 等中间件和应用层例程。

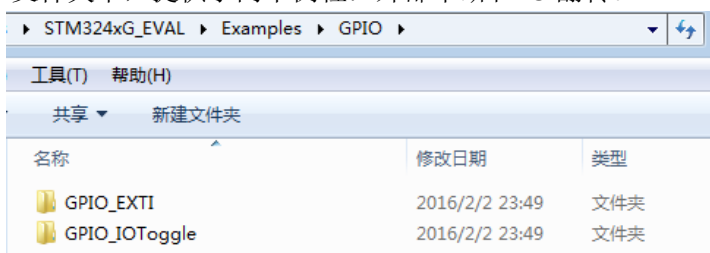
STM324xG_EVAL ▸ Applications ▸			
工具(T) 帮助(H)			
共享 新建文件夹			
名称	修改日期	类型	
Camera	2016/2/2 23:45	文件夹	
Display	2016/2/2 23:45	文件夹	
EEPROM	2016/2/2 23:45	文件夹	
FatFs	2016/2/2 23:45	文件夹	
FreeRTOS	2016/2/2 23:46	文件夹	
IAP	2016/2/2 23:46	文件夹	
LibJpeg	2016/2/2 23:46	文件夹	
LwIP	2016/2/2 23:46	文件夹	
PolarSSL	2016/2/2 23:47	文件夹	
STemWin	2016/2/2 23:47	文件夹	
USB_Device	2016/2/2 23:47	文件夹	
USB_Host	2016/2/2 23:48	文件夹	
Release_Notes.html	2016/1/30 0:14	360 se HTML Do...	

Examples 文件夹中，是各个外设的使用例程。



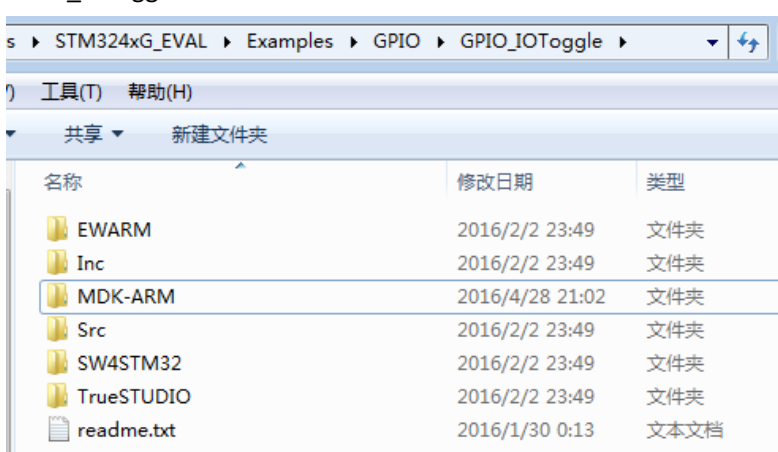
名称	修改日期	类型
ADC	2016/2/2 23:48	文件夹
BSP	2016/2/2 23:48	文件夹
CAN	2016/2/2 23:48	文件夹
Cortex	2016/2/2 23:49	文件夹
CRC	2016/2/2 23:49	文件夹
CRYP	2016/2/2 23:49	文件夹
DAC	2016/2/2 23:49	文件夹
DCMI	2016/2/2 23:49	文件夹
DMA	2016/2/2 23:49	文件夹
FLASH	2016/2/2 23:49	文件夹
FSMC	2016/2/2 23:49	文件夹
GPIO	2016/2/2 23:49	文件夹
HAL	2016/2/2 23:49	文件夹
HASH	2016/2/2 23:50	文件夹
I2C	2016/2/2 23:50	文件夹
I2S	2016/2/2 23:50	文件夹
IWDG	2016/2/2 23:50	文件夹
PWR	2016/2/2 23:50	文件夹
RCC	2016/2/2 23:50	文件夹
RNG	2016/2/2 23:50	文件夹
RTC	2016/2/2 23:50	文件夹
SDIO	2016/2/2 23:50	文件夹
SMARTCARD	2016/2/2 23:50	文件夹
SPI	2016/2/2 23:50	文件夹
TIM	2016/2/2 23:51	文件夹
UART	2016/2/2 23:51	文件夹
WWDG	2016/2/2 23:51	文件夹
Release_Notes.html	2016/1/30 0:12	360 se HTML Do...

GPIO 文件夹中，提供了两个例程：外部中断和 IO 翻转。



名称	修改日期	类型
GPIO_EXTI	2016/2/2 23:49	文件夹
GPIO_IOToggle	2016/2/2 23:49	文件夹

打开 GPIO_IOToggle 文件夹，打开 MDK-ARM 文件夹下的 MDK 工程。



名称	修改日期	类型
EWARM	2016/2/2 23:49	文件夹
Inc	2016/2/2 23:49	文件夹
MDK-ARM	2016/4/28 21:02	文件夹
Src	2016/2/2 23:49	文件夹
SW4STM32	2016/2/2 23:49	文件夹
TrueSTUDIO	2016/2/2 23:49	文件夹
readme.txt	2016/1/30 0:13	文本文档

主函数在进入 while(1)之前，主要是系统时钟的配置，GPIO 的时钟使能和其他配置。

```
66 int main(void)
67 {
68     /* STM32F4xx HAL library initialization:
69      - Configure the Flash prefetch, instruction and Data caches
70      - Configure the SysTick to generate an interrupt each 1 msec
71      - Set NVIC Group Priority to 4
72      - Global MSP (MCU Support Package) initialization
73     */
74     HAL_Init();
75
76     /* Configure the system clock to 168 MHz */
77     SystemClock_Config();
78
79     /* -1- Enable GPIOG, GPIOC and GPIOI Clock (to be able to program
80      __HAL_RCC_GPIOG_CLK_ENABLE();
81      __HAL_RCC_GPIOC_CLK_ENABLE();
82      __HAL_RCC_GPIOI_CLK_ENABLE();
83
84     /* -2- Configure PG.6, PG.8, PI.9 and PC.7 IOs in output push-pull mode to
85      drive external LEDs */
86     GPIO_InitStruct.Pin = (GPIO_PIN_6 | GPIO_PIN_8);
87     GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
88     GPIO_InitStruct.Pull = GPIO_PULLUP;
89     GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
90
91     HAL_GPIO_Init(GPIOG, &GPIO_InitStruct);
92
93     GPIO_InitStruct.Pin = GPIO_PIN_9;
94     GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
95     GPIO_InitStruct.Pull = GPIO_PULLUP;
96     GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
97
98     HAL_GPIO_Init(GPIOI, &GPIO_InitStruct);
99
100    GPIO_InitStruct.Pin = GPIO_PIN_7;
101    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
102    GPIO_InitStruct.Pull = GPIO_PULLUP;
103    GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
104
105    HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
106
```

进入 while(1)之后，是 GPIO 的翻转操作。

```
107 /* -3- Toggle PG.6, PG.8, PI.9 and PC.7 IOs in an infinite loop */
108 while (1)
109 {
110     HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_6);
111     /* Insert delay 100 ms */
112     HAL_Delay(100);
113     HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_8);
114     /* Insert delay 100 ms */
115     HAL_Delay(100);
116     HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_9);
117     /* Insert delay 100 ms */
118     HAL_Delay(100);
119     HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_7);
120     /* Insert delay 100 ms */
121     HAL_Delay(100);
122 }
```

在使用 CubeMX 开发 STM32 的时候，片上外设的初始化代码由 CubeMX 生成，即 while(1)之前的这些代码是不需要关心的。除非在调试过程中需要进行小的改动，因为目前如果使用 CubeMX 进行配置修改，就要对整个工程进行全编译，有时候这是不必要的。

在参考上述的例程的时候，主要是参考 while(1)中的代码。

