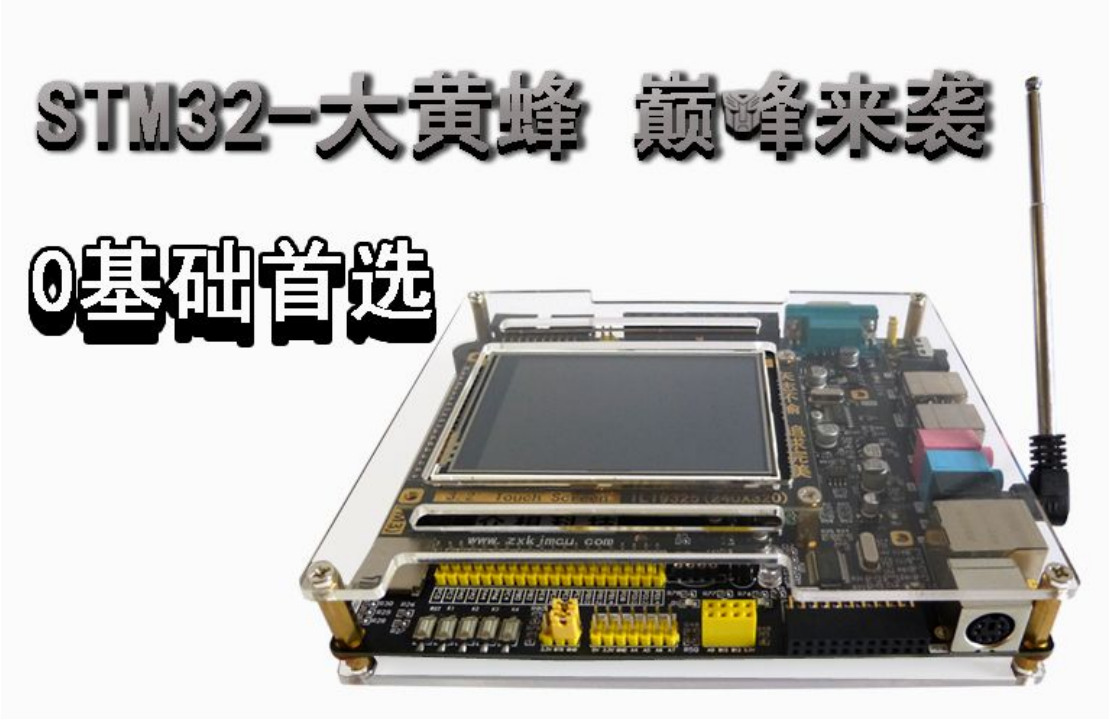
学 ARM 从 STM32 开始

STM32 开发板库函数教程--实战篇



官方网站：http://www.zxkjmcu.com
官方店铺：http://zxkjmcu.taobao.com
官方论坛：http://bbs.zxkjmcu.com
刘洋课堂：http://school.zxkjmcu.com

4.1 点亮第一个发光二极管

本节给大家实现怎样用 STM32 点亮一个发光二极管。实验虽然很简单，但是涉及到的知识点却非常多。比如我们要熟悉 STM32 的开发环境、时钟配置、GPIO 端口配置、延时程序的实现和 C 语言编写规范等。可以说只要我们能够熟悉并掌握上面这些知识，说明我们已经成功的敲开了 STM32 的大门。

关于 STM32 的开发环境与时钟配置，在前面的章节中已经给大家详细的介绍了，在这里就不再叙述了。

4.1.1 首先给大家解释两个专业术语：端口和 I/O 引脚。

端口：是指具有相同地址的（8/16/32)位 I/O 引脚，可以进行读写访问的。访问端口时也可以（8/16/32)位一次性访问。

引脚：可以特指 CPU 的任意一位管脚，其中也可以指端口中包含的某一位引脚。

端口是从功能上描述单片机，引脚是从电路原理上描述单片机。这就是为什么我们说要测哪个引脚的电压，而不是哪个端口的电压。

STM32 的 GPIO 端口配置比以前我们学习 51 单片机时要复杂的多。就是因为复杂，所以 STM32 的 GPIO 端口配置更灵活，更能适合不同的应用需求。

STM32 有 7 个 GPIO 端口，可分为 GPIOA、GPIOB、GPIOC、GPIOD、GPIOE、GPIOF、GPIOG。每个端口又包含了 0~15 共 16 个不同的引脚。对于不同型号的 STM32 芯片，端口的个数和引脚的数量不同，具体请

参考相应芯片型号的手册。

STM32 的 I/O 引脚可以由软件配置成如下 8 种模式：

1、输入浮空
2、输入上拉
3、输入下拉
4、模拟输入
5、开漏输出
6、推挽输出
7、推挽式复用功能
8、开漏复用功能

每个 I/O 引脚都可以独立的配置。 STM32 的很多 I/O 引脚都是兼容 5V 电平，这些 I/O 引脚在与 5V 电平外设连接的时候很有优势，具体哪些 I/O 引脚是兼容 5V 电平，可以从该芯片的数据手册端口描述章节查到（I/O Level 标 FT 的就是兼容 5V 电平的）。由于我们整套视频教程与书籍都是采用库函数的方法编程序的，所以大家不用考虑怎样通过设置寄存器来配置 I/O 引脚，我们只要掌握怎样调用库函数文件中函数，来配置 I/O 引脚初始化就可以了。

4.1.2 GPIO 引脚配置函数 GPIO_Init

函数名	GPIO_Init
函数原形	void GPIO_Init(GPIO_TypeDef* GPIOx, GPIO_InitTypeDef* GPIO_InitStruct)
功能描述	根据 GPIO_InitStruct 中指定的参数初始化外设 GPIOx 寄存器
输入参数 1	GPIOx：x 可以是 A，B，C，D 或者 E，来选择 GPIO 外设
输入参数 2	GPIO_InitStruct：指向结构 GPIO_InitTypeDef 的指针，包含了外设 GPIO 的配置信息.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

第二个参数结构体的原型如下。

```
typedef struct
{
    uint16_t GPIO_Pin;
    GPIOSpeed_TypeDef GPIO_Speed;
    GPIOMode_TypeDef GPIO_Mode;
}GPIO_InitTypeDef;
```

4.1.2.1 参数 GPIO_Pin

该参数选择待设置的 GPIO 引脚，使用操作符“|”可以一次选中多个引脚。可以使用下表中的任意组合。

GPIO_Pin	描述
GPIO_Pin_None	无引被选中
GPIO_Pin_0	选中引脚 0
GPIO_Pin_1	选中引脚 1
GPIO_Pin_2	选中引脚 2
GPIO_Pin_3	选中引脚 3
GPIO_Pin_4	选中引脚 4
GPIO_Pin_5	选中引脚 5
GPIO_Pin_6	选中引脚 6
GPIO_Pin_7	选中引脚 7
GPIO_Pin_8	选中引脚 8
GPIO_Pin_9	选中引脚 9
GPIO_Pin_10	选中引脚 10
GPIO_Pin_11	选中引脚 11
GPIO_Pin_12	选中引脚 12
GPIO_Pin_13	选中引脚 13
GPIO_Pin_14	选中引脚 14
GPIO_Pin_15	选中引脚 15
GPIO_Pin_All	选中全部引脚

4.1.2.2 参数 GPIO_Speed

GPIO_Speed 用以设置选中引脚的速率。

GPIO_Speed	描述
GPIO_Speed_10MHz	最高输出速率 10MHz
GPIO_Speed_2MHz	最高输出速率 2MHz
GPIO_Speed_50MHz	最高输出速率 50MHz

4.1.2.3 参数 GPIO_Mode

GPIO_Mode 用以设置选中管脚的工作状态。

GPIO_Speed	描述
GPIO_Mode_AIN	模拟输入
GPIO_Mode_IN_FLOATING	浮空输入
GPIO_Mode_IPD	下拉输入
GPIO_Mode_IPU	上拉输入
GPIO_Mode_Out_OD	开漏输出
GPIO_Mode_Out_PP	推挽输出
GPIO_Mode_AF_OD	复用开漏输出
GPIO_Mode_AF_PP	复用推挽输出

4.1.2.4 PB5 引脚配置函数如下：

```
GPIO_InitTypeDef GPIO_InitStructure;

GPIO_InitStructure.GPIO_Pin=GPIO_Pin_5;
GPIO_InitStructure.GPIO_Speed=GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_Mode=GPIO_Mode_Out_PP;
GPIO_Init(GPIOB,&GPIO_InitStructure);
```

4.1.3 结构体定义：

4.1.3.1 结构体(struct)是由一系列具有相同类型或不同类型的数据构成的数据集合，也叫结构。是一种聚合类型，里面可以包含多种数据类型，甚至可以结构体里嵌套结构体。在实际项目中，结构体是大量存在的。研发人员常使用结构体来封装一些属性来组成新的类型。结构体在函数中的作用不是简便，其最主要的作用就是封装。封装的好处就是可以再次利用。让使用者不必关心这个是什么，只要根据定义使用就可以了。

4.1.3.2 就上边的例子，首先要定义一个结构体变量,结构体为 GPIO_InitTypeDef，定义的结构体变量为 GPIO_InitStructure，大家要注意结构体和结构体变量的区别。这个结构体变量中包括了 3 个

参数，分别是 GPIO_Pin、GPIO_Speed 和 GPIO_Mode。由于我们这节实验用的 I/O 引脚是 PB5,所以 GPIO_Pin=GPIO_Pin_5。引脚速率可以任意选择一个,在这里我们选择的是GPIO_Speed=GPIO_Speed_50MHz，如果对引脚的输出速率没有要求，那么在干扰很强的环境中运行，可以降低引脚的速率使系统更加稳定。引脚的工作状态我们选择的是推挽输出模式 GPIO_Mode=GPIO_Mode_Out_PP。当我们把这些参数都配置好以后就可以通过 GPIO_Init 来对端口初始化了。

4.1.3 硬件设计

大黄蜂 STM32 开发板上有 3 个 LED 发光二极管指示灯，PCB 线路板上的标号分别是D1、D2、D3。与原理图对应的标号是(91.PB5-LED1)、(87.PD6-LED2)、(84.PD3-LED3)。原理图上标号的命名规则如下，

我们以 D1 发光二极管为例说明：

MCU 芯片管脚序号(91)+I/O 引脚(PB5)+功能标号(LED1)

LED 原理图

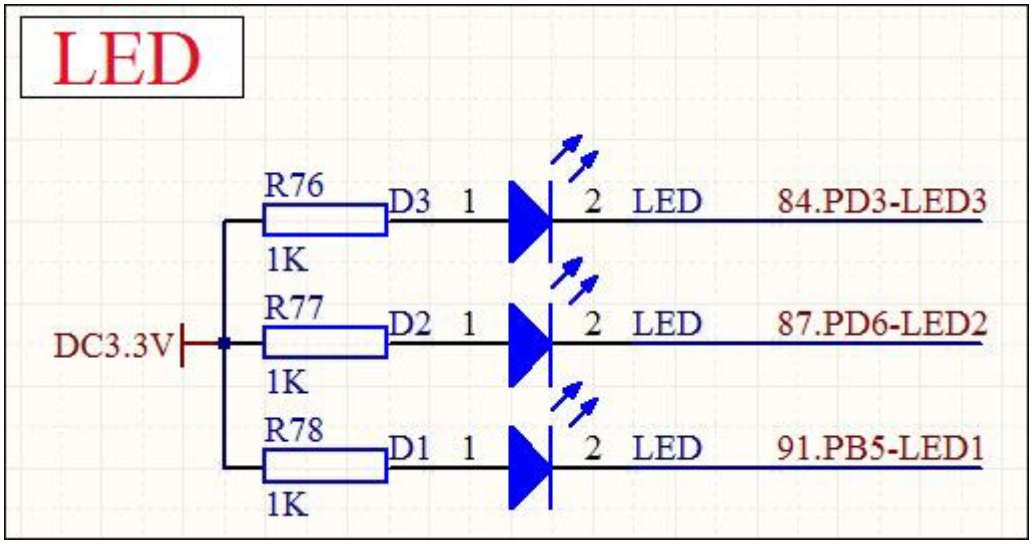


图 4.1

原理图与 PCB 线路板上已经都连接好了，大家不用自己再连接

了，直接把程序下载到大黄蜂 STM32 开发板上运行就可以了。

4.1.4 软件设计

4.1.4.1 STM32 库函数文件

```
stm32f10x_gpio.c
stm32f10x_rcc.c
```

本节实验我们主要用到的库文件，其中 stm32f10x_gpio.h 头文件包含了 GPIO 端口的定义。stm32f10x_rcc.h 头文件包含了系统时钟配置函数以及相关的外设时钟使能函数，所以我们要把这两个头文件对应的 stm32f10x_gpio.c 和 stm32f10x_rcc.c 都加到工程中。

4.1.4.2 自定义头文件

```
pbdata.h
pbdata.c
```

同时我们自己也创建了两个公共的文件，这两个文件主要存放我们自定义的公共函数和全局变量，以方便以后每个功能模块之间传递参数。

4.1.4.3 pbdata.h 文件里的内容是

```
#ifndef _pbdata_H
#define _pbdata_H
#include "stm32f10x.h"
void delay(u32 nCount);//延时函数
#endif
```

语句 #ifndef、#endif 是为了防止 pbdata.h 文件被多个文件调用时出现错误提示。如果不加这两条语句，当两个文件同时调用 pbdata 文件时，会提示重复调用错误。 stm32f10x.h 头文件是我们每个工程都需要调用的，里面包括了 STM32 内部寄存器地址的定义。

4.1.4.4 pbdata.c 文件里的内容是

```
#include "pbdata.h" //别忘了引用这个头文件
```

```
//延时程序
void delay(u32 nCount)
{
    for(;nCount!=0;nCount--);
}
```

delay 延时函数可以通过很多种方法实现，由于本节实验对延时的精度要求不高，所以我们就采用了 51 单片机的延时方法。主要就是通过 for 循环占据 CPU 的处理时间最终达到延时的目的。

4.1.5 STM32 系统时钟配置 SystemInit()

每个工程都必须在开始时配置并启动 STM32 系统时钟。关于 STM32 系统时钟我们在前面的章节中已经详细的说明了，在这里就不再叙述了。

4.1.6 GPIO 引脚时钟使能

```
RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB,ENABLE);
```

本节实验只用到了 PB 端口，所以要把 PB 端口的时钟打开。如果大家还想同时点亮其他两个 LED 发光二极管，那么还要把 PD 端口的时钟也打开。

4.1.7 GPIO 管脚电平控制函数

不管你用的是高端的 ARM 还是低端的单片机，也不管你用的是汇编语言还是 C 语言编，我们最终要达到的目的都是要控制 MCU 管脚的输入、输出状态，STM32 当然也不例外。下面介绍给大家两个函数。

```
PB5 管脚输出高电平
GPIO_SetBits(GPIOB,GPIO_Pin_5);

PB5 管脚输出低电平
GPIO_ResetBits(GPIOB,GPIO_Pin_5);
```

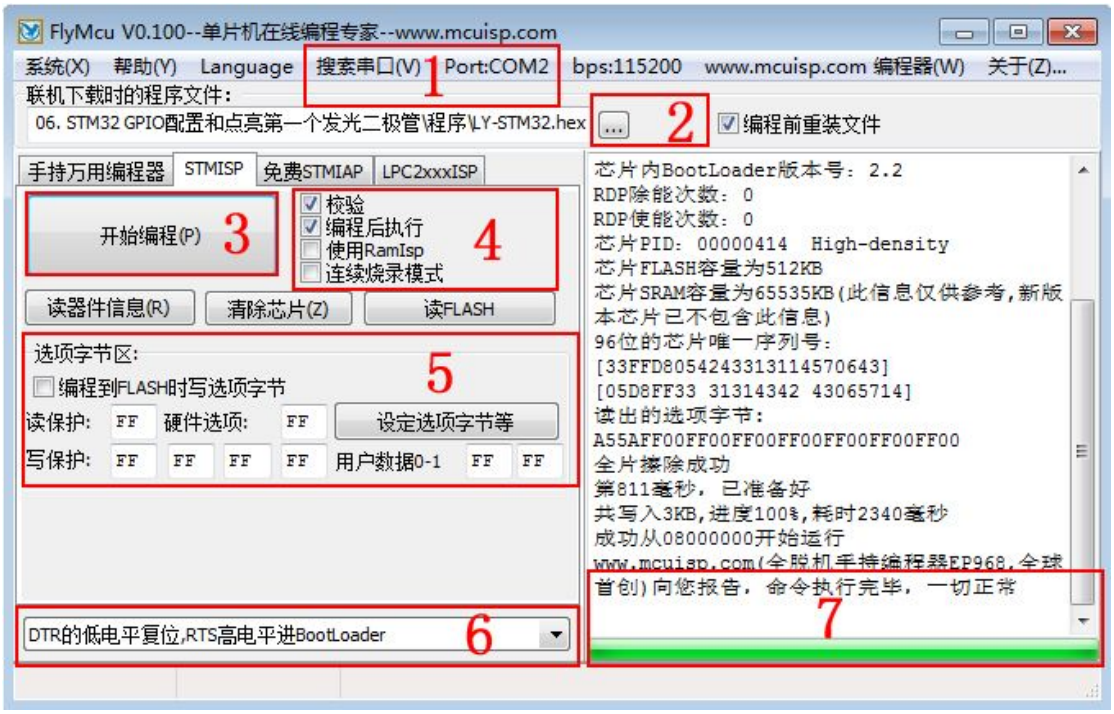
两个函数中传递的参数都是一样的，第一个参数是端口(GPIOB)，第二个参数是管脚序号(GPIO_Pin_5)。

4.1.8 main.c 文件里的内容是

```
#include "pbdata.h"//别忘了引用 pbdata.h 这个头文件
void RCC_Configuration(void);//声明系统时钟初始化函数
void GPIO_Configuration(void);//声明端口初始化函数

int main(void)
{
   RCC_Configuration(); //系统时钟初始化
   GPIO_Configuration();//端口初始化

   while(1)
   {
       //PB5 管脚输出高电平
       GPIO_SetBits(GPIOB,GPIO_Pin_5);

       delay(6000000);//间隔 0.5 秒钟

       //PB5 管脚输出低电平
       GPIO_ResetBits(GPIOB,GPIO_Pin_5);

      delay(6000000);//间隔 0.5 秒钟
   }
}
//定义系统时钟初始化函数
void RCC_Configuration(void)
{
    SystemInit();//初始化系统时钟
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB,ENABLE);//使能 PB
端口时钟
}
//定义端口初始化函数
void GPIO_Configuration(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

   GPIO_InitStructure.GPIO_Pin=GPIO_Pin_5;//配置 GPIO_Pin_5 引脚
   GPIO_InitStructure.GPIO_Speed=GPIO_Speed_50MHz;//引脚速率
50MHz
   GPIO_InitStructure.GPIO_Mode=GPIO_Mode_Out_PP;//推挽输出
   GPIO_Init(GPIOB,&GPIO_InitStructure);//初始化 GPIOB.5 端口
}
```

在 main(void)程序体中代码不多，开始调用了两个函数，分别是系统时钟初始化函数 RCC_Configuration()和端口初始化函数 GPIO_Configuration()，接下来进入 while(1)循环体，在循环体中，每间隔 0.5 秒钟切换一次 PB5 引脚的输出状态，使其不断的高低电平变化输出。当 PB5 引脚输出低电平时 D1 发光二极管点亮，当 PB5 引脚输出高电平时 D1 发光二极管熄灭。

4.1.9 程序下载



请根据上图所指向的 7 个重点区域配置。其中（1）号区域根据自己机器的实际情况选择，我的机器虚拟出来的串口号是 COM2。（2）号区域请自己选择程序所在的文件夹。（7）号区域当程序下载完后，进度条会到达最右边，并且提示一切正常。（4、5、6）号区域一定要按照上图显示的设置。当都设置好以后就可以直接点击（3）号区域的开始编程按钮下传程序了。本节实验的源代码在光盘中：

（LY-STM32 光盘资料\1.课程\1,基础篇\基础篇 06. STM32 GPIO 配置和点亮第

一个发光二极管\程序）

4.1.10 实验效果图

下面是 D1 发光二极管闪烁效果，间隔时间是 0.5 秒钟。