
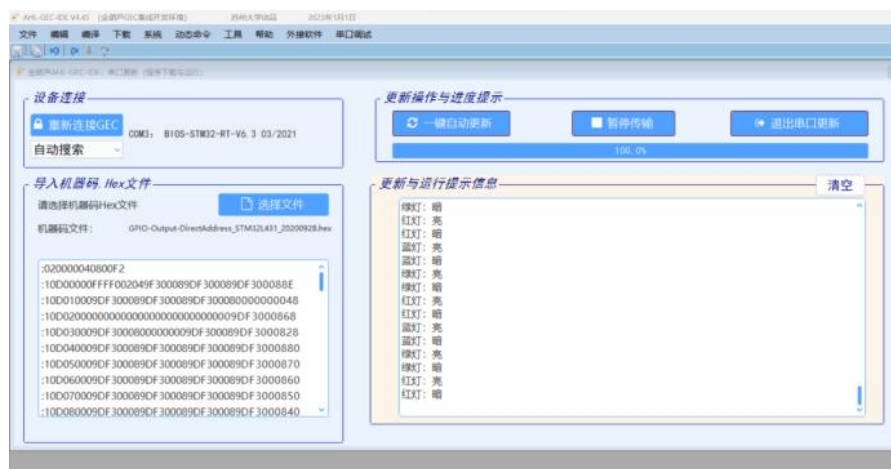


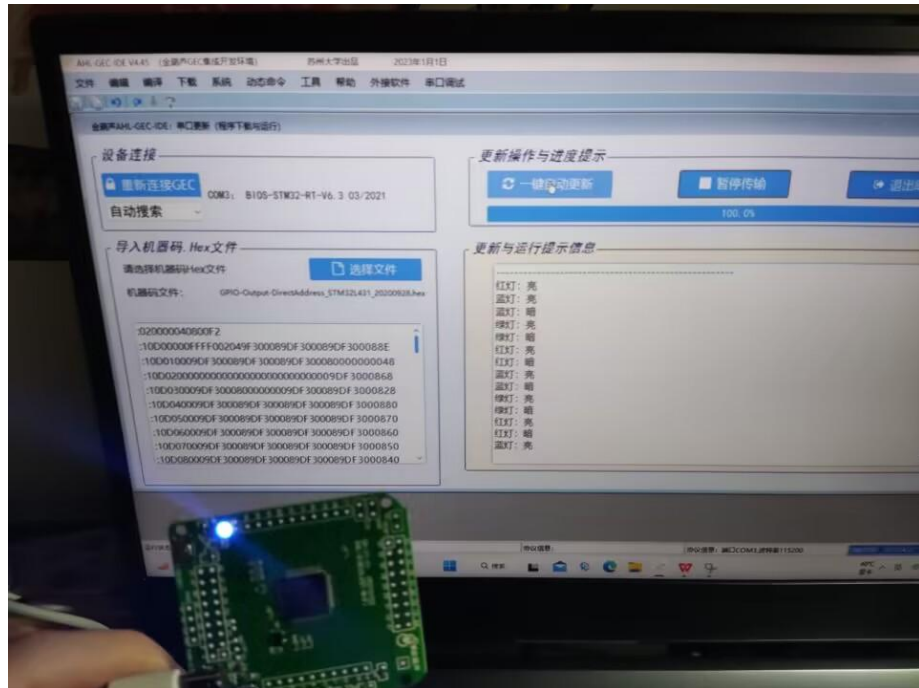
# 嵌入式系统及应用实验报告

学号	202018526	姓名	高树林	专业年级	人工智能 2020 级
实验地点	南楼中栋 108 室		实验时间	2023-04-27	
实验名称	实验一 熟悉实验开发环境及 GPIO 编程				
实验学时	2	实验类型	■验证型    □设计型    ■创新型    ■综合型		
实验目的	1.了解集成开发环境的安装与基本使用方法。 2.掌握 GPIO 构件基本应用方法，理解第一个 C 程序框架结构，了解汇编语言与 C 语言的如何相互调用。 3.掌握硬件系统的软件测试方法，初步理解 printf 输出调试的基本方法。				
实验准备	软件	华水学堂-》课程-》嵌入式系统及应用 资料下载编译器 AHL-GEC-IDE，安装。			
	硬件	实验箱，或者自己课本的配套电路板			
实验内容	本实验通过编程控制 LED 小灯，体会 GPIO 输出作用，可扩展控制蜂鸣器、继电器等；通过编程获取引脚状态，体会 GPIO 输入作用，可用于获取开关的状态。				
实验任务及实验过程	1.验证 (1)从华水学堂资料中，下载第 4 章源代码，将工程 \04-Soft\GPIO-Output-DirectAddress 导入编译器。阅读代码，理解软件如何直接根据硬件资源的地址而实现的点亮一个发光二极管--蓝灯，理解构件式的嵌入式编程过程。 运行截图：（请贴上你的运行截图）				
	<div></div>				



(2)第4章源代码，将工程\04-Soft\GPIO-Output-Component 导入编译器。阅读代码，理解软件如何调用 GPIO 构件实现点亮一个发光二极管--蓝灯，理解构件式的嵌入式编程过程。  
运行截图：（请贴上你的运行截图）





根据对以上两个工程的整体理解，可以基于任意一个工程，改造为其它软件编程直接干预硬件的工程。

## 2.实践性问答题

(1)  $X \&= \sim(1 \ll 3)$ 的作用是什么？ $X |= (1 \ll 3)$ 的作用是什么？在嵌入式开发中，能有什么实际的作用？

$X \&= (1 \ll 3)$ 的作用是将  $X$  的第 3 位（从右往左数，从 0 开始计数）设置为 0。该操作使用按位取反运算符（ $\sim$ ）将 1 左移 3 位，然后使用按位与运算符（ $\&$ ）将结果与  $X$  中对应的位相与，以将第 3 位设置为 0。

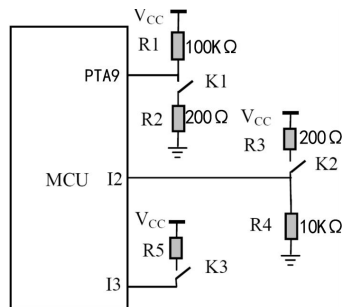
$X |= (1 \ll 3)$ 的作用是将  $X$  的第 3 位设置为 1。该操作使用左移运算符（ $\ll$ ）将 1 左移 3 位，然后使用按位或运算符（ $|$ ）将结果与  $X$  中对应的位相或，以将第 3 位设置为 1。

在嵌入式开发中，这些操作通常用于控制单个寄存器或寄存器集中的特定位。例如，可以使用它们来控制 GPIO（通用输入/输出）引脚的状态或控制设备的某些功能，如打开或关闭 LED 灯。这些操作也可用于与硬件交互时设置或清除特定的标志位或状态位。

(2)请解释一下 volatile 的作用。

在计算机编程中，volatile 是一个关键字，用于指示该变量的值可能会在程序的控制之外被修改。具体来说，当一个变量被声明为 volatile 时，编译器会生成额外的代码来确保每次读取该变量时都会从内存中读取其最新的值，而不是使用缓存或寄存器中的旧值。volatile 可以用于多线程编程中的共享变量，以确保对变量的修改在不同线程之间是可见的。这是因为 volatile 变量的值在每次读取时都必须从内存中获取，而不是从线程的本地缓存中获取。此外，volatile 还可以用于访问硬件设备的变量，因为这些变量的值可能会在程序控制之外被修改，例如，一个硬件寄存器的值可能会被外部设备改变。

(3)请结合 GPIO 的寄存器，详细描述一下，如何获得一个开关是否被按下的开关量状态？请描述编程基本步骤。比如下图中的端口 A 的第 9 引脚（PTA9），需要被设置为输入状态，根据 PTA9 的状态，可以知道开关 K1 是否被按下，实现和对外界环境的交互。



编程步骤如下：

首先，需要配置 GPIO 模式寄存器（MODER）来设置引脚的输入输出模式。对于端口 A 的第 9 引脚（PTA9），需要将 MODER 寄存器的第 18 位和第 19 位设置为 0b00，即输入模式。MODER 寄存器的地址为 0x4002 0000。

接下来，需要配置 GPIO 输入数据寄存器（IDR）来读取引脚的电平状态。对于端口 A 的第 9 引脚（PTA9），可以通过读取 IDR 寄存器的第 9 位来获取其电平状态。IDR 寄存器的地址为 0x4002 0010。

最后，可以通过判断 IDR 寄存器的第 9 位是否为 1 来判断开关 K1 是否被按下。

```
1. #include "stm32f10x.h"
2.
3. int main(void)
4. {
5.     // 使能 GPIOA 时钟
6.     RCC->APB2ENR |= RCC_APB2ENR_IOPAEN;
7.
8.     // 配置 GPIOA 的第 9 引脚为输入模式
9.     GPIOA->CRL &= ~(GPIO_CRL_MODE9 | GPIO_CRL_CNF9);
10.    GPIOA->CRL |= GPIO_CRL_CNF9_0;
11.
12.    while (1)
13.    {
14.        // 读取 GPIOA 的 IDR 寄存器的第 9 位，即端口 A 的第 9 引脚
           (PTA9) 的状态
15.        if (GPIOA->IDR & GPIO_IDR_IDR9)
16.        {
17.            // 端口 A 的第 9 引脚（PTA9）被按下
18.        }
19.        else
20.        {
21.            // 端口 A 的第 9 引脚（PTA9）未被按下
22.        }
```

	<div>23.     } 24. }</div>
实验总结 (收获)	<p>在本次实验中，我通过学习熟悉了实验开发环境及 <b>GPIO</b> 编程。这是一次非常有意义的实验，让我深刻理解了硬件编程的重要性，同时也让我更加熟悉了开发环境的使用。</p> <p>在实验中，我使用了一些常见的开发工具，这些工具为我提供了很好的开发环境，使我能够更加高效地编写代码和调试程序。此外，我还学习了使用 <b>Stm32</b> 开发板进行 <b>GPIO</b> 编程的方法。通过这种方式，我可以通过树莓派来控制外部设备（三色灯）。</p> <p>在实验过程中，我发现 <b>GPIO</b> 编程需要严格遵守硬件的规范，以确保程序的稳定性和可靠性。如果在编程过程中出现错误，可能会导致外部设备受损或无法正常工作。因此，我在修改代码时，一定要非常仔细，确保程序的正确性。在本次实验中，我还学习了如何进行调试和测试。通过使用调试器，我可以更加快速地找到问题所在，并进行相应的修复。此外，我还学习了如何使用模拟器进行测试，以确保程序的正确性和稳定性。</p> <p>总的来说，本次实验让我收获颇丰。通过学习熟悉实验开发环境及 <b>GPIO</b> 编程，我深刻理解了硬件编程的重要性，并掌握了一些常见的开发工具和编程方法。在未来的学习和工作中，这些知识和经验将对我非常有帮助。</p>

**备注：**此表格，可以根据自己的实验情况，进行格式调整。