第4章 GPIO及第一个汇编程序

4.1 GPIO通用基础知识

4.2 软件干预硬件的方法

4.3 汇编语言工程举例：控制小灯闪烁

实验一：基于汇编程序理解软件如何干预硬件

汇编语言通常被应用在芯片启动过程、操作系统调度以及对程序运行实时性较高的场合。要能够做到从底层透明理解微型计算机运行的基本原理，利用汇编语言进行简单程序设计是必不可少的学习环节。许多人认为汇编语言难以入门，实际上只要掌握基本方法，规范编程，勤于实践，就容易入门。况且，实际编程中，并不是完全使用汇编程序来设计整个应用程序，只是在必要的地方才使用。若完全不了解，则无法完成此类工程。为降低学习难度，本章以运用通用输入输出GPIO构件点亮一盏小灯为例，阐述微型计算机中软件干预硬件的基本方法，并由此给出汇编语言编程模板，以便“照葫芦画瓢”地学习汇编程序设计。

4.1 GPIO通用基础知识

点亮一个发光二极管是最基础的程序，这类程序也有许多实际用途，就是基于软件实现开关量的控制。从微型计算机的视角来看，就是要利用微型计算机一个引脚，控制一盏小灯的亮暗。这个功能，在微型计算机中被称之为通用输入输出，本节首先给出其通用基础知识。

4.1.1 GPIO概念

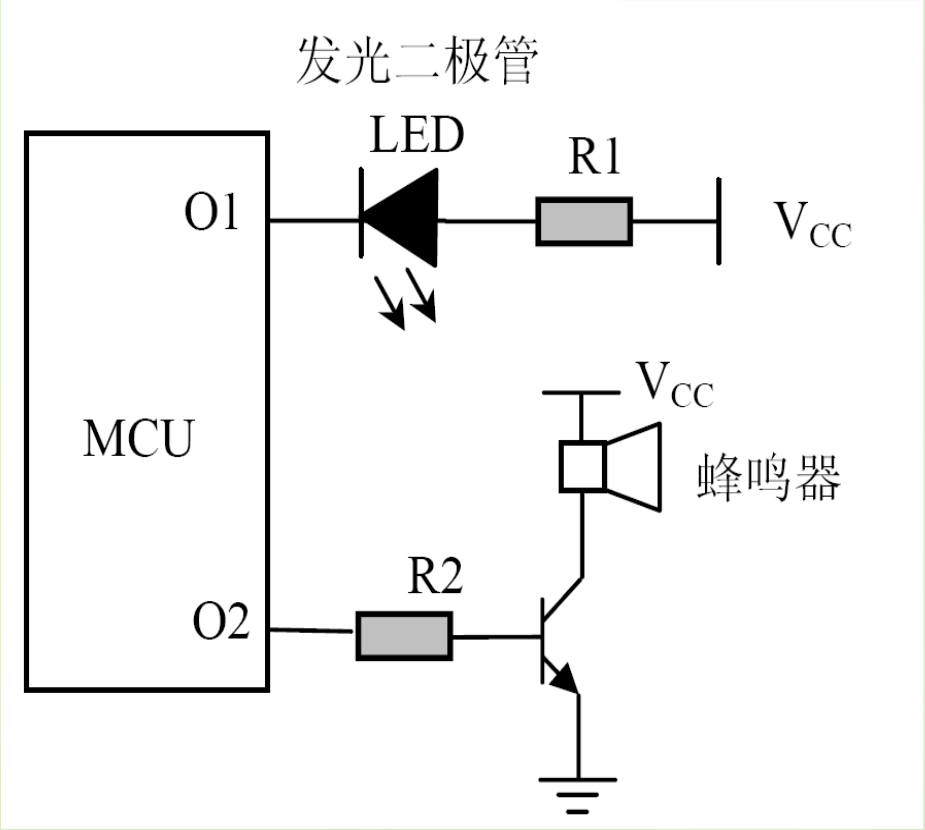
输入/输出（Input/Output，I/O）接口，是MCU同外界进行交互的重要通道，MCU与外部设备的数据交换通过I/O接口来实现。

通用I/O也记为**GPIO**（General Purpose I/O），即基本输入/输出。作为**通用输出引脚**，MCU内部程序通过端口寄存器**控制该引脚状态**，使得引脚输出“1”（高电平）或“0”（低电平），即开关量输出。作为**通用输入引脚**，MCU内部程序可以通过端口寄存器**获取该引脚状态**，以确定该引脚是“1”（高电平）或“0”（低电平），即开关量输入。

思考一下：逻辑1与实际的物理电平存在的对应关系

4.1.2 输出引脚的基本接法

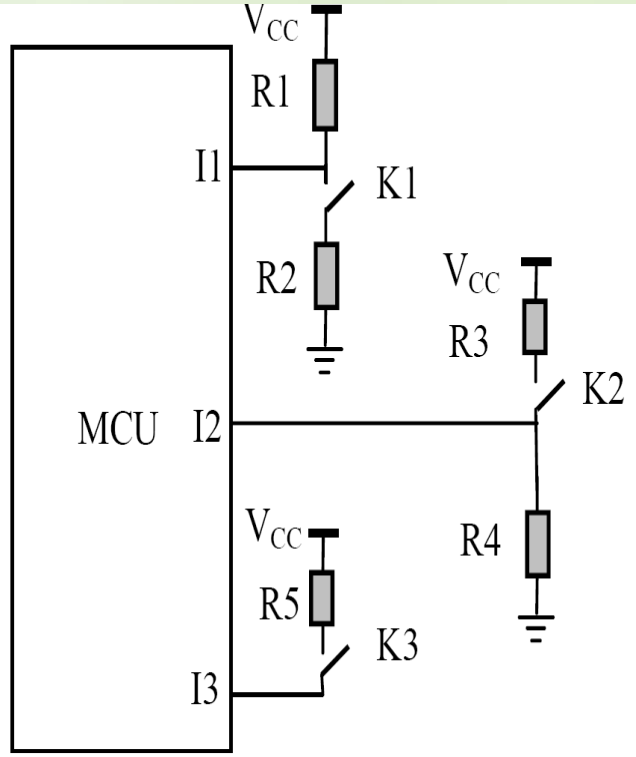
输出引脚O1和O2采用了不同的方式驱动外部器件，一种接法是O1直接驱动发光二极管LED，当O1引脚输出高电平时，LED不亮；当O1引脚输出低电平时，LED点亮。这种接法的驱动电流一般在2mA～10mA。另一种接法是O2通过一个NPN三极管驱动蜂鸣器，当O2引脚输出高电平时，三极管导通，蜂鸣器响；当O2引脚输出低电平时，三极管截止，蜂鸣器不响。



思考一下：什么情况下不用三极管驱动，什么情况下用，如何控制一盏交流220V的灯？

4.1.3 上拉下拉电阻与输入引脚的基本接法

若MCU的某个引脚通过一个电阻接到电源（Vcc）上，这个电阻被称为“上拉电阻”；与之相对应，若MCU的某个引脚通过一个电阻接到地（GND）上，则相应的电阻被称为“下拉电阻”。这种做法使得，悬空的芯片引脚被上拉电阻或下拉电阻初始化为高电平或低电平。



引脚I1通过上拉电阻R1接到Vcc，选择R1>>R2，K1断开时，引脚I1为高电平，K1闭合时，引脚I1为低电平。

引脚I2通过下拉电阻R4接到地，选择R3<<R4，K2断开时，引脚I2为低电平，K2闭合时，引脚I2为高电平。

引脚I3处于悬空状态，K3断开时，引脚I3的电平不确定（这样不好）。

友情提示：大多数情况使用上拉电阻

4.2 软件干预硬件的方法

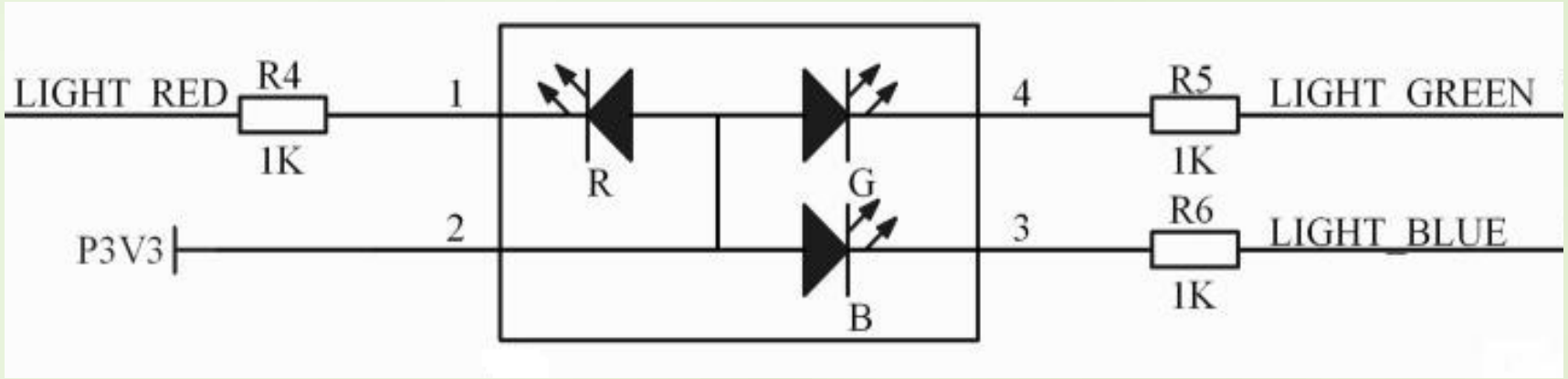
要使用软件方法点亮一盏小灯，实际是个复杂过程，为了把这个复杂过程简单化，首先要制作GPIO构件，然后利用GPIO构件进行应用编程，点亮小灯。本书的目标在于基于底层驱动构件，利用汇编语言进行应用层面编程，以期理解微型计算机运行原理。因此，对于底层驱动构件，重在使用，理解接口，了解其原理即可。

4.2.1 底层驱动构件的概念

学习微型计算机原理的主要目标是用汇编语言编程干预硬件，通过软件获得硬件的状态，通过软件控制硬件的动作。通常情况下，软件与某一硬件模块打交道通过其底层**驱动构件**，也就是封装好的一些函数，编程时通过调用这些函数，干预硬件。这样就把**制作构件**与**使用构件**的工作分开。就像建设桥梁，先做标准预制板一样，这个标准预制板就是构件。

1．软件如何干预硬件？

例如，若要点亮蓝灯，须将LIGHT_BLUE与微型计算机的一个具有GPIO功能的引脚连接起来，通过编程使其引脚为低电平（逻辑0），就能点亮蓝色LED，**这就是软件干预硬件的基本过程**。



若采用从“零”开始编程的方法，需要与端口寄存器直接打交道，对初学者十分困难，可以利用已经做好的GPIO构件，先把LED小灯点亮，然后根据不同学习要求，再理解构件是如何做出来的。

2. 软件构件定义

机械、建筑等传统产业的运作模式是先生产符合标准的构件（零部件），然后将标准构件按照规则组装成实际产品。其中，构件（Component）是核心和基础，复用是必需的手段。

面向构件程序设计工作组提出的软件构件定义：软件构件是一种组装单元，它具有规范的接口规约和显式的语境依赖。软件构件可以被独立地部署并由第三方任意地组装

一般来说，可以将软件构件理解为：在语义完整、语法正确的情况下，具有可复用价值的单位软件，是软件复用过程中可以明确辨别的成分；从程序角度上可以将构件看作是有一定功能、能够独立工作或协同其他构件共同完成的程序体。

对比：软件构件与建房子的砖块

3. 构件的API

在构件制作完成后，如何使用构件进行应用程序的开发呢？它是通过应用程序编程接口来完成的。所谓**应用程序编程接口（Application Programming Interface，API）**是指应用程序与构件之间的衔接约定，应用程序开发人员通过它干预硬件，而无需理解其内部的工作细节。这种衔接约定，就是指构件内部的各个函数名和参数，使用时调用其函数名，并使其参数实例化。

例如，**GPIO构件**中有初始化（gpio_init）、设定引脚状态（gpio_set）、获取引脚状态（gpio_get）等函数，而gpio_set函数的形参为引脚名和状态，可以通过gpio_set函数并将其形参实例化就可以控制小灯的亮暗。

着重理解：（1）函数的形参与实参；
（2）构件函数中为何不能使用全局变量？
（3）函数参数的传值与传址

4. 底层驱动构件的概念

所谓**底层驱动构件**是直接面向硬件操作的程序代码及函数接口的使用说明，规范的底层驱动构件由头文件（.inc）及源程序文件（.s）构成，头文件(.inc)是底层驱动构件简明且完备的使用说明，也就是说，在不需查看源程序文件的情况下，就能够完全使用该构件进行上一层程序的开发。

底层驱动构件若不使用汇编编程，相应组织形式有变化，但实质不变。

为了根能好地进行微机原理的教学，本书的底层驱动构件大部分使用C语言编写，应用程序大部分采用RSIC-V汇编语言编写。

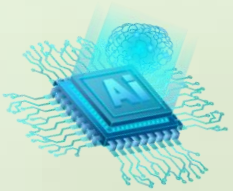
4.2.2 GPIO构件API

嵌入式系统的重要特点是软件硬件相结合，通过软件获得硬件的状态，通过软件控制硬件的动作。应用软件通过**底层驱动构件**，也就是封装好的一些函数，干预硬件。这样就把**制作构件**与**使用构件**的工作分成不同过程。

面对嵌入式人工智能，更是如此。

1. GPIO构件的常用函数

GPIO构件主要API有：GPIO的初始化（gpio_init）、设置引脚状态（gpio_set）、获取引脚状态（gpio_get）等。



2. GPIO构件的头文件gpio.h

```
//=========================================================================
//函数名称：gpio_init
//函数返回：无
//参数说明：port_pin：(端口号)|(引脚号)（如：(PTB_NUM)|(14) 表示为B口14号脚）
//          dir：引脚方向（0=输入，1=输出,可用引脚方向宏定义）
//          state：端口引脚初始状态（0=低电平，1=高电平）
//功能概要：初始化指定端口引脚作为GPIO引脚功能，并定义为输入或输出，若是输出，
//          还指定初始状态是低电平或高电平
//=========================================================================
void gpio_init(uint16_t port_pin, uint8_t dir, uint8_t state);

void gpio_set(uint16_t port_pin, uint8_t state);      //设定引脚状态

uint8_t gpio_get(uint16_t port_pin);                  //获取引脚状态
```

4.2.3 GPIO构件的使用方法

以控制一盏小灯闪烁为例，必须知道两点：一是由芯片的哪个引脚，二是高电平点亮还是低电平点亮，这样就可使用gpio构件控制小灯。

1. 给小灯取名

在user.inc文件中给小灯起名，并确定与MCU连接的引脚，进行宏定义。

```
.equ LIGHT_RED   , (PTB_NUM|12)     /*红色RUN灯使用的端口/引脚*/
```

2. 给灯灯的亮暗取名

```
/*灯状态宏定义（灯亮、灯暗对应的物理电平由硬件接法决定）*/
.equ LIGHT_ON,1                 /*灯亮*/
.equ LIGHT_OFF,0                /*灯暗*/
```

3. 初始化小灯

在main.s文件中初始化小灯的初始状态为输出，且初始为暗。

```
/* 初始化GPIO（蓝灯），三个参数 */
li a0,LIGHT_BLUE          /* 灯的引脚 */
li a1,GPIO_OUTPUT         /* 输出 */
li a2,LIGHT_OFF           /* 初始状态 */
call gpio_init            /* 调用函数 */
```

4. 点亮小灯

```
/* 蓝灯亮 */
li a0,LIGHT_BLUE          /* 第1参数a0 ← LIGHT_BLUE */
li a1,LIGHT_ON            /* 第2参数a1 ← LIGHT_ON */
call gpio_set             /* 调用函数 */
```

4.3 汇编语言工程举例：控制小灯闪烁

实际运行【CH04\GPIO-asm】并解析。

注意：

（1）GPIO构件中主要函数的使用方法；

（2）printf构件的使用方法。

所有同学跟随实际运行理解。

实验一：基于汇编程序理解软件如何干预硬件

在学院网站上传实验报告及实验程序，注意注释规范。
建议将程序文件中所有注释改为//（需在看正常编译条件下）

本章作业：1、2、4、6