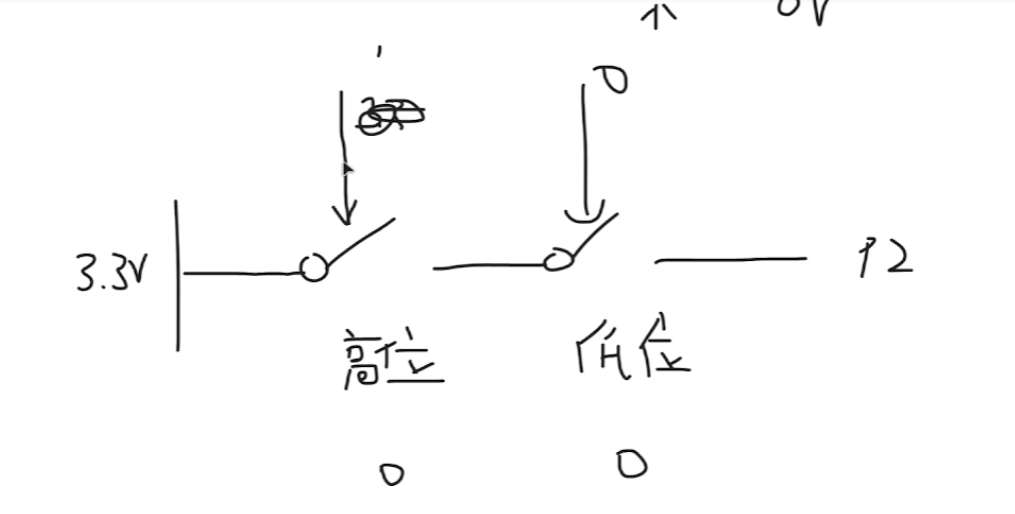
GPIO

何为IO

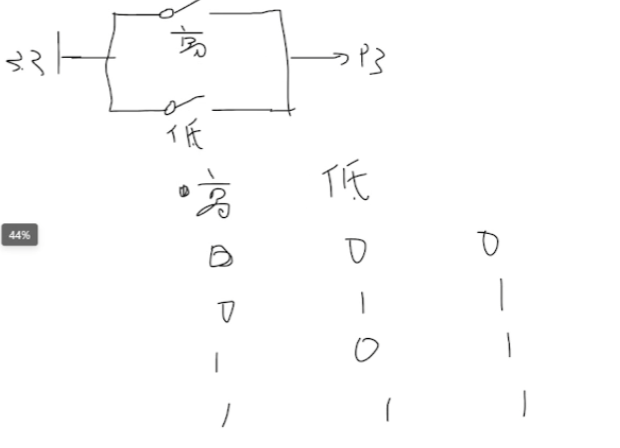
Input and out put ,主管输入和输出

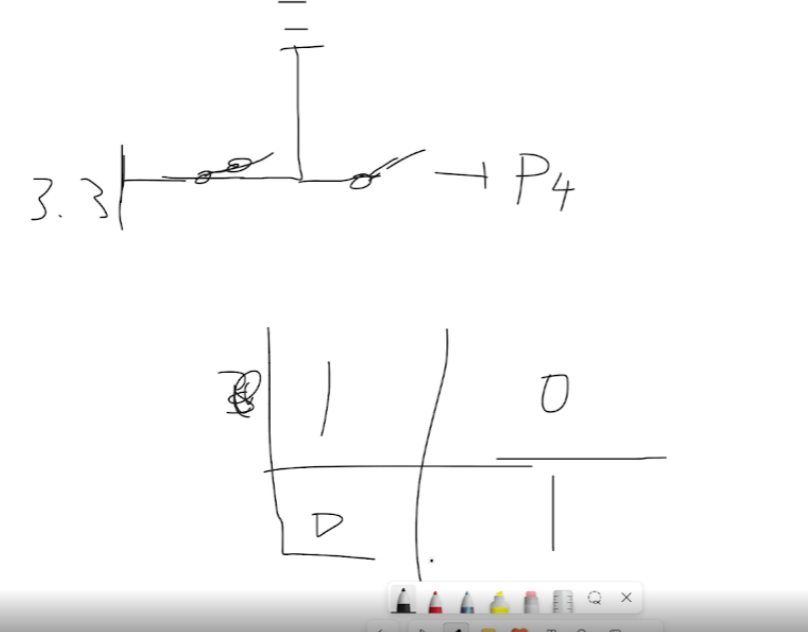
与或非电路

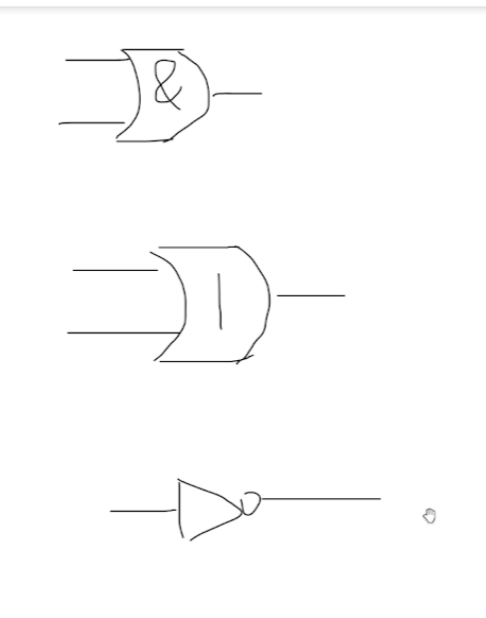




与逻辑 同时开关才可以

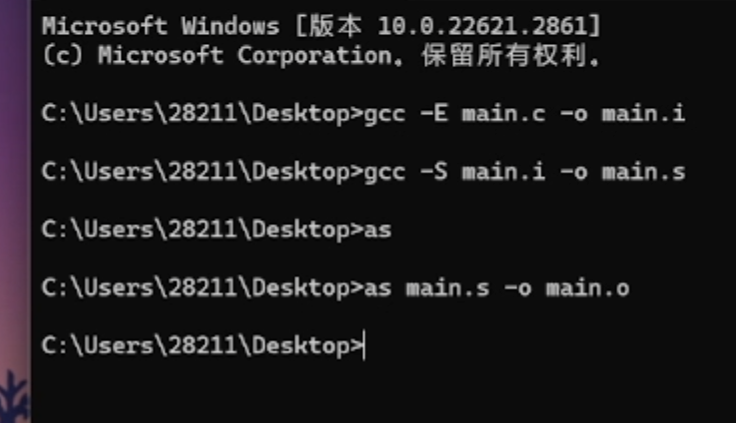
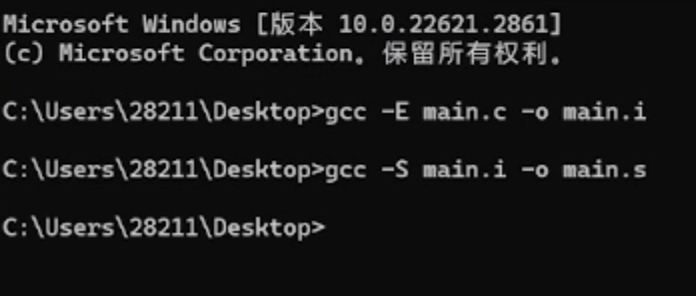
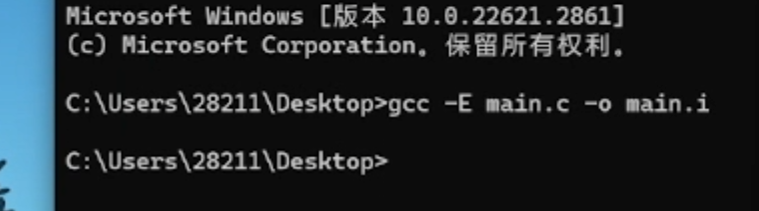
或逻辑 （1=3.3v，0=0v）

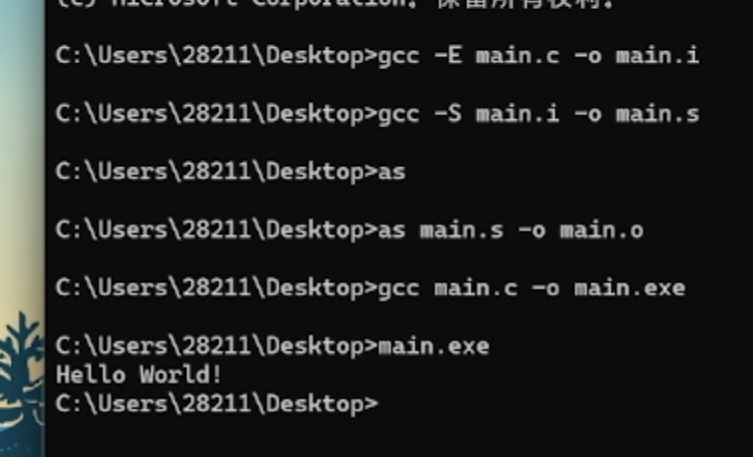
非逻辑

与门 或门 非门

编译过程：把C语言文件~ .i文件（中间链接文件） ~汇编文件 ~二进制文件



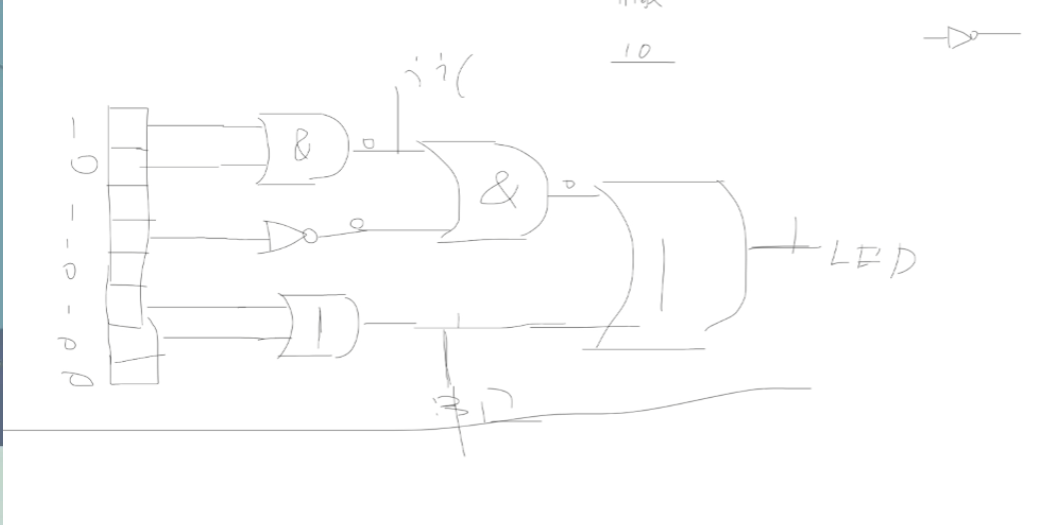
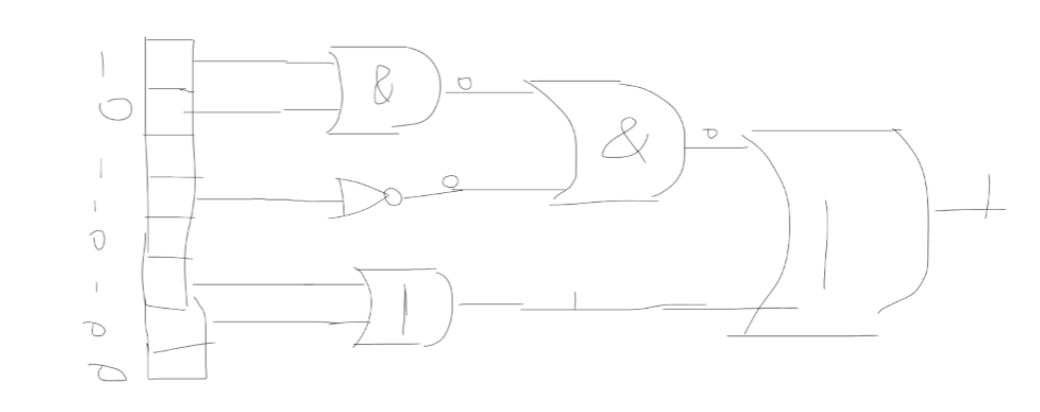


.c->.i->.s->.bin.o.exe

最简单的计算机模型 CPU模型

运行过程

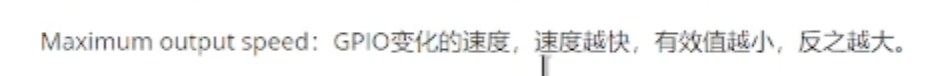
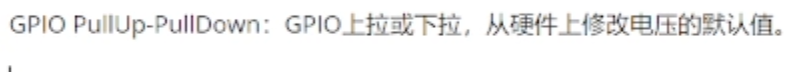
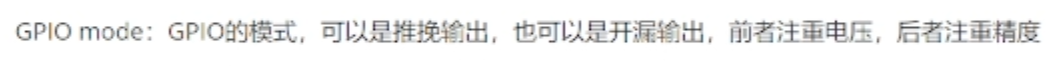
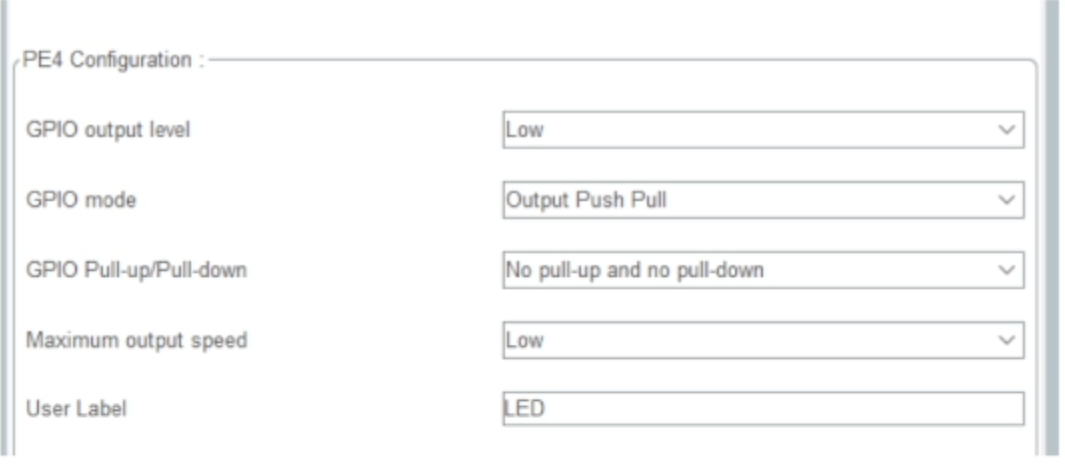
二进制文使用数字信号加载进入cpu，CPU内的逻辑门进行计算得到我们的输出内容，输出内容可以是1或0，逻辑门的一部分代表了一个功能，可以IIC和GPIO用到了同一部分



上拉和下拉

上拉就是默认为高电平，下拉就是默认低电平。

GPIO速度如果是输出电压为主，则速度越低越好，原因是速度快了，电平变化速度也快了，近似于一种PWM的操作， 导致输出电压与高电平持续时间有关，造成一部分电压精度损失。



寄存器（）

内存是一块比较小的区域，它主要做的事情就是把我们程序加载进去，内存的运行速度比外存快的多得多，寄存器与内存结构一样功能也一样，就是位置不一样，位置不一样速度不一样，（寄存器距离cpu近）单片机只有内存没有外存。

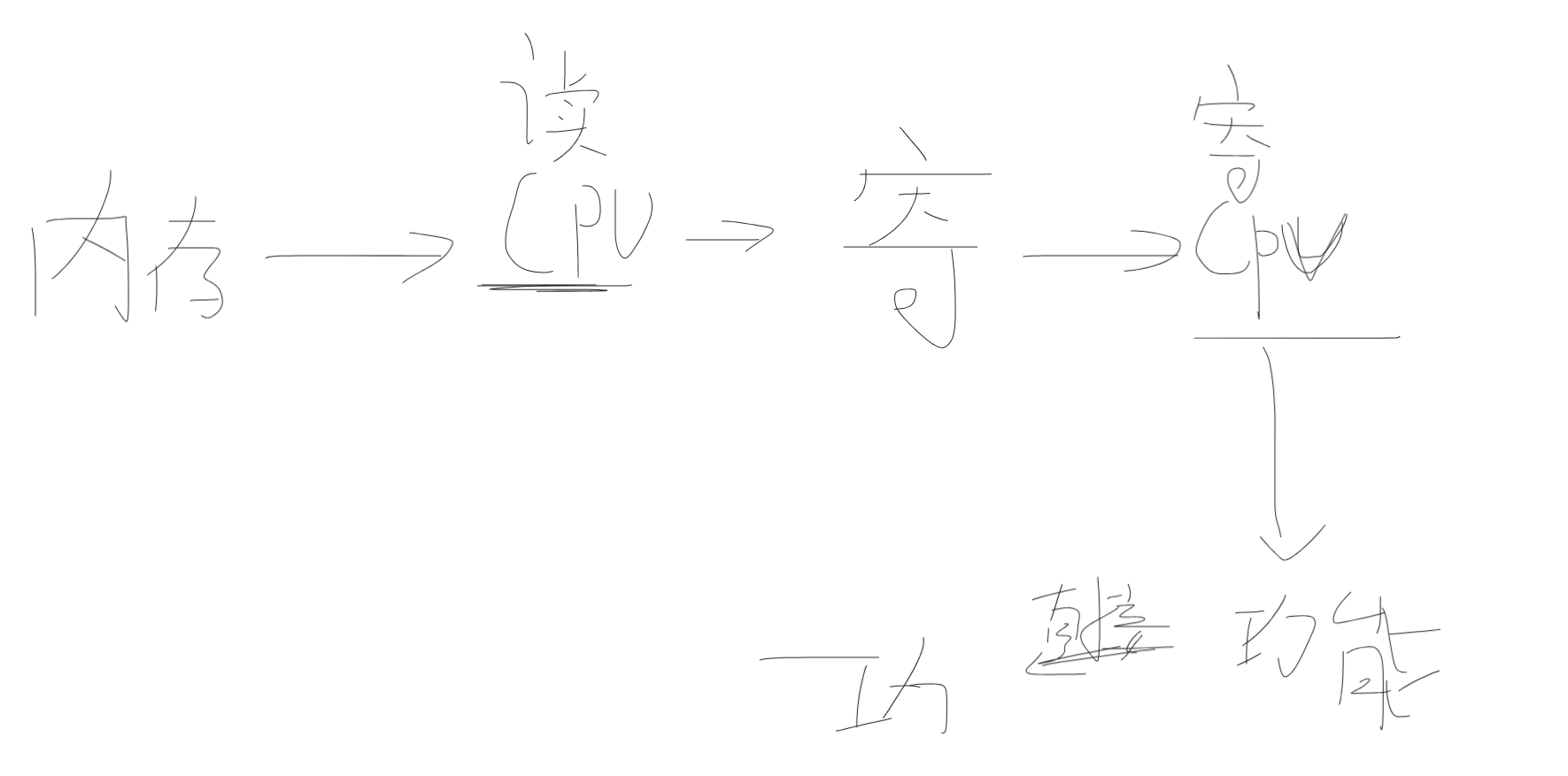
寄存器负责保存程序段，我们一个程序可以很大很大，所以都保存在内存中，每次cpu读取一部分保存在寄存器中，然后按照寄存器的值进行运行。

在代码编译的过程中会把所有无关内容（包括变量名，函数名）都给删除掉，只保留 对寄存器或者内存的操作，也就意味着我们程序的效率更高，。直接操作寄存器速度 快得多！ 对于所有单片机而言，直接操作寄存器就可以直接操作CPU的运行程序一开始可能非常大，经过编译之后也会缩减一部分，但是单片机的性能有限，只 能一段一段的读取，读取到的内容保存在寄存器里面。

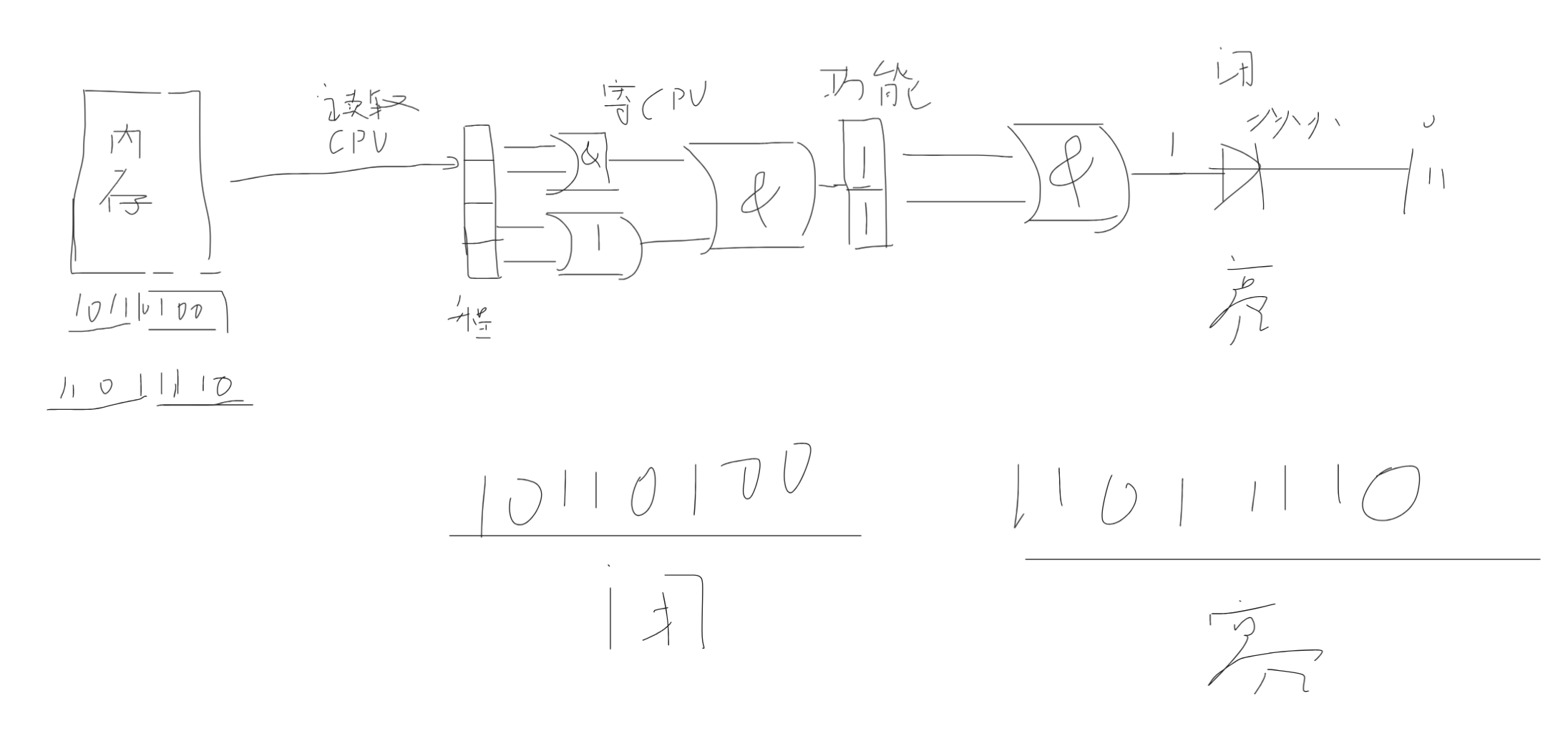
程序寄存器

就是把代码一段一段读取之后保存的位置，二进制形式

功能寄存器

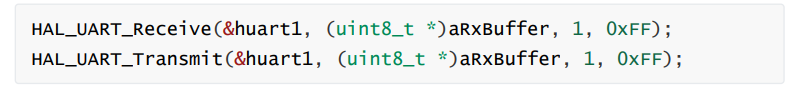
操作设备时，使用到的寄存器，面向设备。 直接操作功能寄存器，也就是直接操作设备。而一般编写代码都是操作程序寄存器， 让CPU去安排怎么操作功能寄存器，最终完成某些功能。 编写的所有代码经过编译就已经列举出所有可能的情况了，非常详细。

代码烧录进内存后，是以二进制形式保存的，经过专门负责读取的CPU放进程序寄存 器内，每次读取只读取程序的一部分。寄存器CPU读取程序寄存器并处理得到功能性 的值，保存进功能寄存器。最终功能寄存器经过逻辑门的计算得到对应的功能编写代码最终产生了效果。 寄存器有读写操作，写操作负责从输入端读取内容并放在锁存器内，读操作负责把锁 存器内容清空并输出到输出端。RS锁存器，Read-Set。

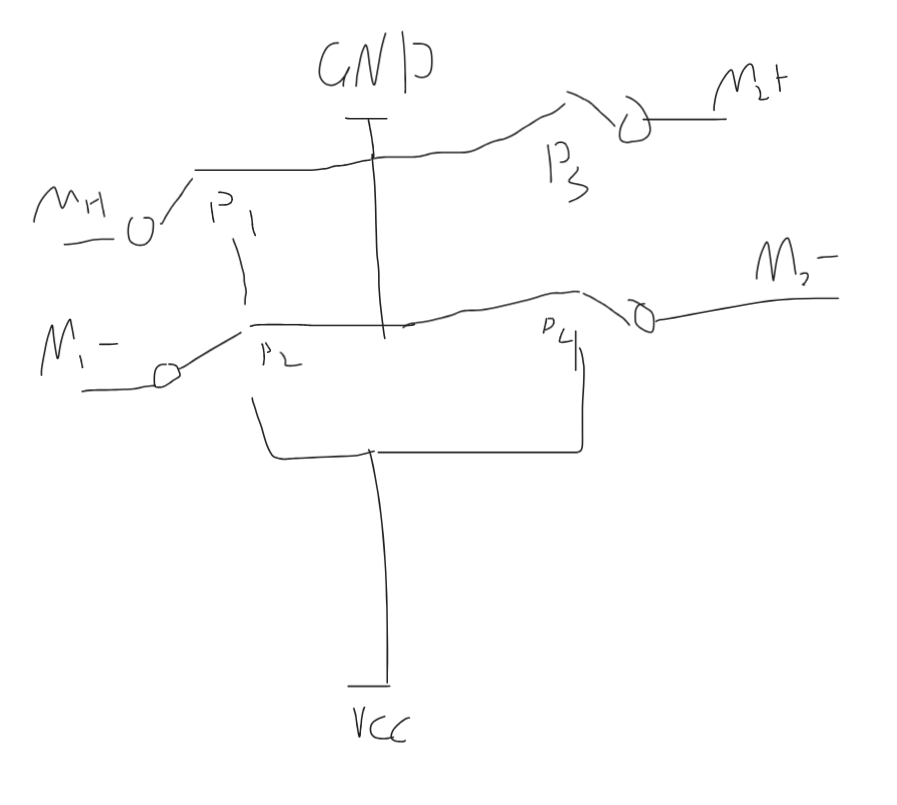


通信寄存器

比如说串口，输入进来的数据都是二进制形式，原因是数据都是ASCII码形式，都可 以用一个无符号的整数来表示，可以计算为二进制数，作为传输的数据，保存在通信 寄存器内，相当于功能寄存器，只不过这里的通信寄存器并不会反应成功能，而是直 接保存在寄存器内供CPU使用。 通信就是二进制的读写。

第一个参数的意思是读取的串口是哪个，也就相对于给通信寄存器指了个位置。第二 个参数是希望保存在程序寄存器的位置，反映到代码上就是代码创建数组的地址。第 三个参数是希望得到数据的大小，也就是循环读取几次通信寄存器，第四个参数是延 迟经过多长时间没读到就自动放弃。

H桥和电机控制



L298N可以控制开关的闭合。



通过修改开关是断开还是闭合就可以产生对应的电位差。