STM32地址映射、位带操作

目录

1：前言

> 以前学51的时候，没有注重地址的这个概念，因为51 寄存器 少，一个reg52.h里面涵盖了你要用到的所有寄存器地址，你只需要去写几个字母调用一下就可以了。我甚至一度以为TMOD就是TMOD，单片机生产出来，程序就是这样写，就好像用手机指纹解锁，你只是把手指放上去，手机就解锁了。你以为理所应当，但是手机却做了很多工作。
> 直到学习32后，由于庞大的寄存器数量，让我不得不去注意到为什么我写几个字母，就可以操作到我想要操作的寄存器上去。SO，地址概念也就被我重视起来。下面只是简单的一些我理解的32地址知识。如有不足，咋~~评论区过两招！！！

2：地址（门牌号）

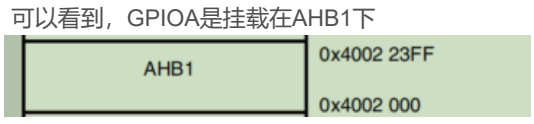
> “地址”：第一印象，就是一张地图。一个你家的门牌号。就是一条可以让你找的你想找到位置的指引。这是在广义上我的理解
> 在计算机狭义上： 内存中每个用于数据存取的基本单位，都被赋予一个唯一的序号，称为地址，也叫做内存地址。
> bala~~bala~~bala~~bala~~
> 地址就是”唯一“的，不管是在现实生活中，还是在单片机，嵌入式里面。都是为了方便你去快速准确找到你想要的。

3：STM32地址映射

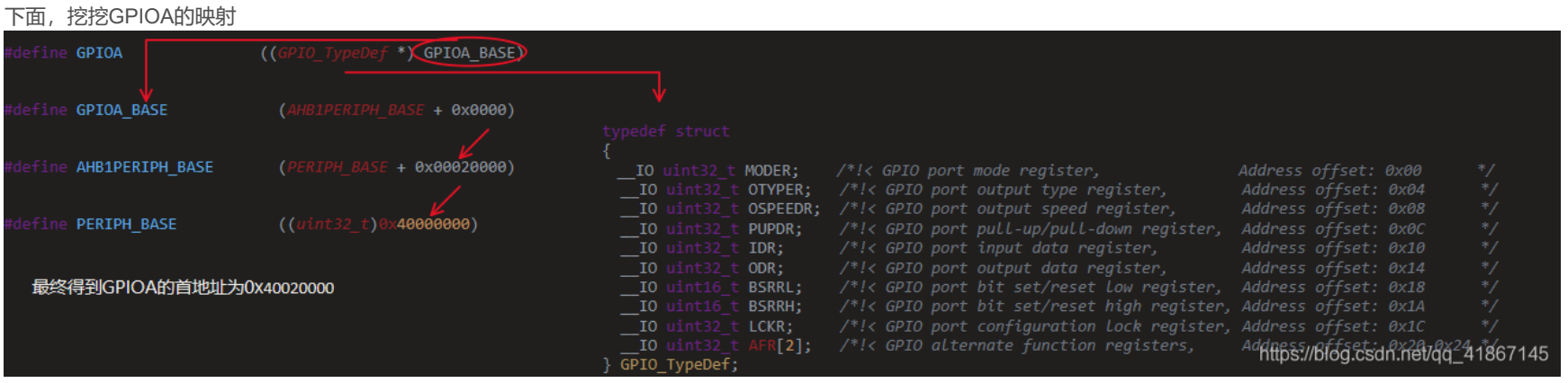
> 不多说，让我们康康，32里面的地址是怎么操作的
> 以GPIOA为例

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA,ENABLE); //使能GPIOA时钟
```

> 可以看到，GPIOA是挂载在AHB1下



> AHB1地址段如上，那么GPIOA肯定地址区间就在这个范围
> 下面，挖挖GPIOA的映射

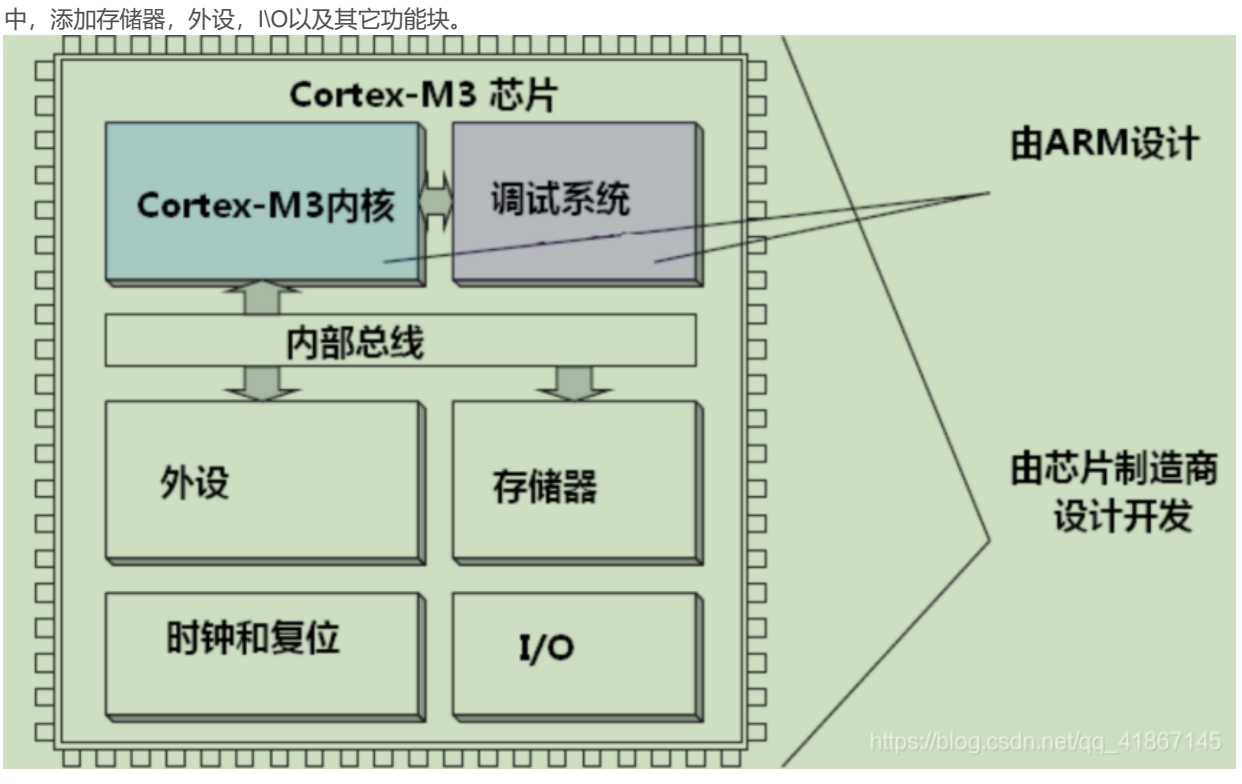


> 巧了，就是AHB1的首地址，这不就好起来了
> 又由于GPIO下各寄存器已经在GPIO_TypeDef的结构体中都有声明，并且按照相对应的地址偏移顺序排列，这样就可以做到诸如GPIOx->MODER，这样的操作来访问到相对应GPIO的MODER寄存器了

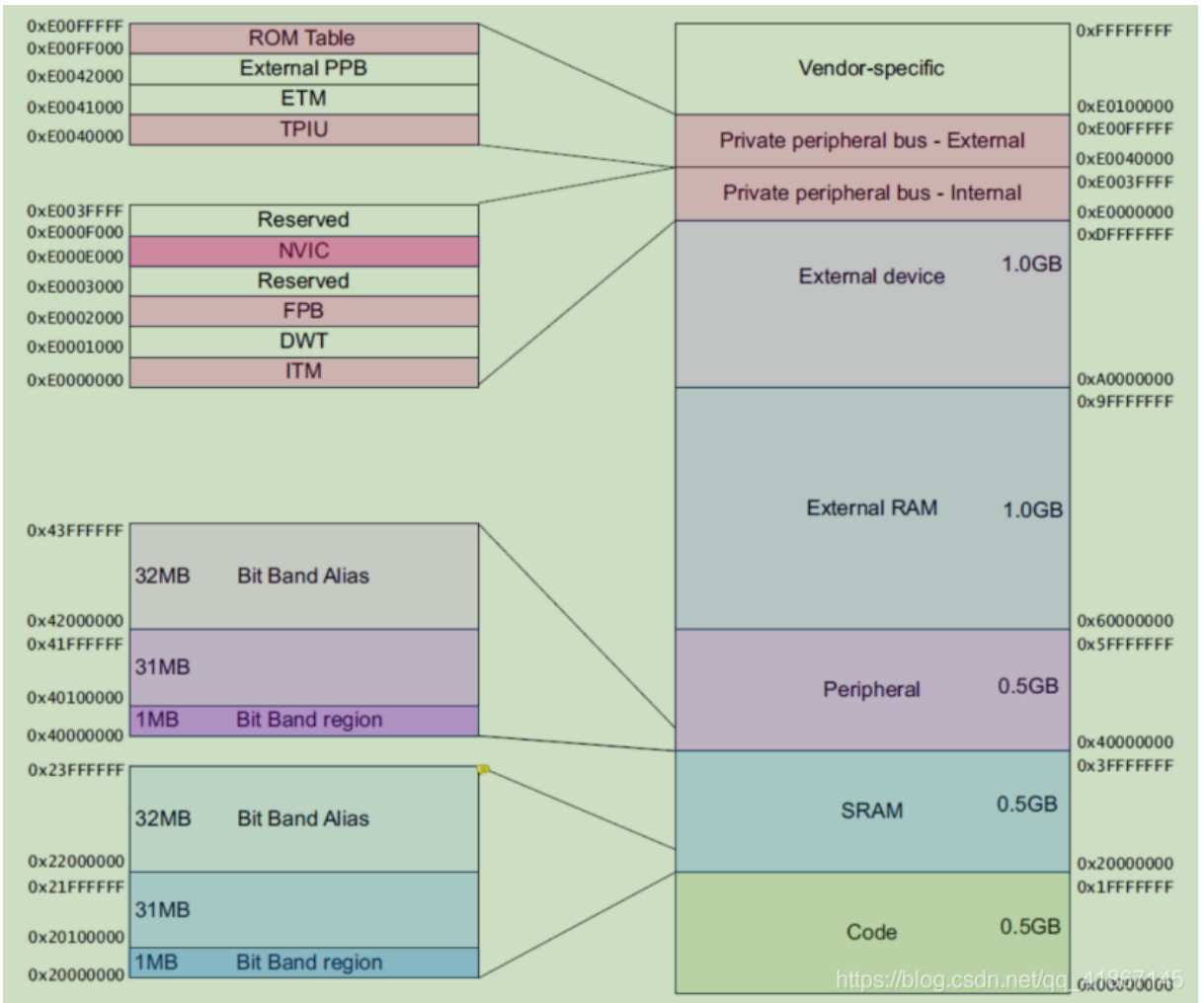
> 像这种通过一层层的将 地址映射 起来的操作32里面一大推，道理都是一样。

4：位带操作原理

> eg:STM32F407ZGT6
> 注：字=4字节=32比特
> STM32F407ZGT6是基于Cortex_M4处理器内核的(我们以M3为例是一样的)，在芯片制造商得到CM3处理器内核的使用授权之后，它们就可以把CM3内核用在自己的硅片设计中，添加存储器，外设，I\O以及其它功能块。



CM3的地址空间是4GB，康康这4GB的粗线条划分：



内部SRAM区的大小是512M，用于让芯片制造商连接片上的SRAM,这个区通过系统总线来访问，在这个区的下部，有一个1M的位带区，该位带区还有一个相应的32M位带别名区。容纳了8M的"位变量"（32M/32bit）=8M;

位带区对应的是 最低的 1MB 地址范围，而位带别名区里面的每个字对应位带区的一个比特

地址空间的另一个512M的范围由片上外设的寄存器使用，这个区也有一条32M的位带别名区。

SO:

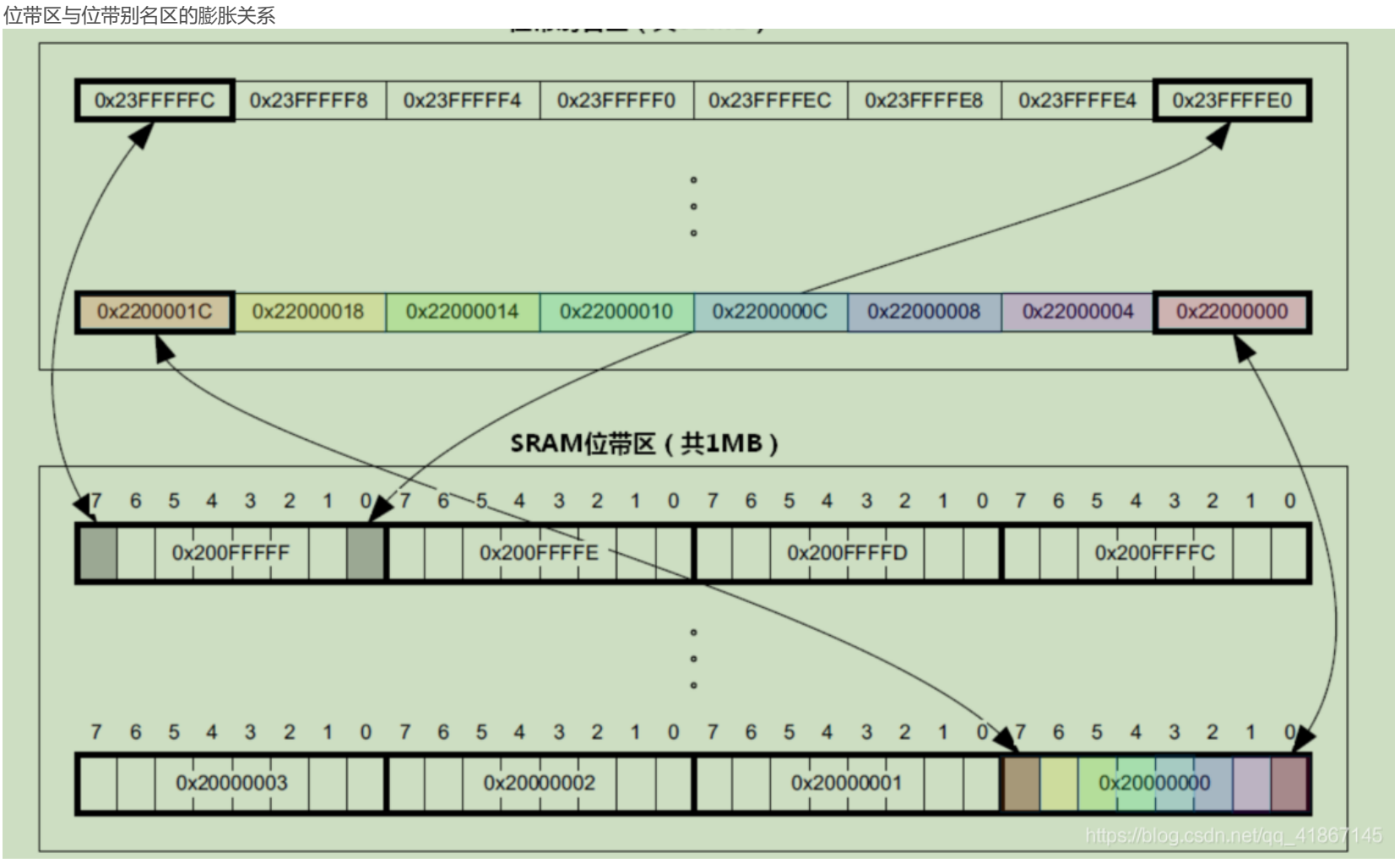
SRAM区的最低1M范围　0X2000 0000~0X200F FFFF

片上外设区的最低1M范围 0X4000 0000~0X400F FFFF

这两个地方支持位带骚操作！！！

位带区中每一个比特"膨胀"为一个32位的字放在位带别名区。当通过位带别名区访问这些字时，就可以达到访问原始比特的目的

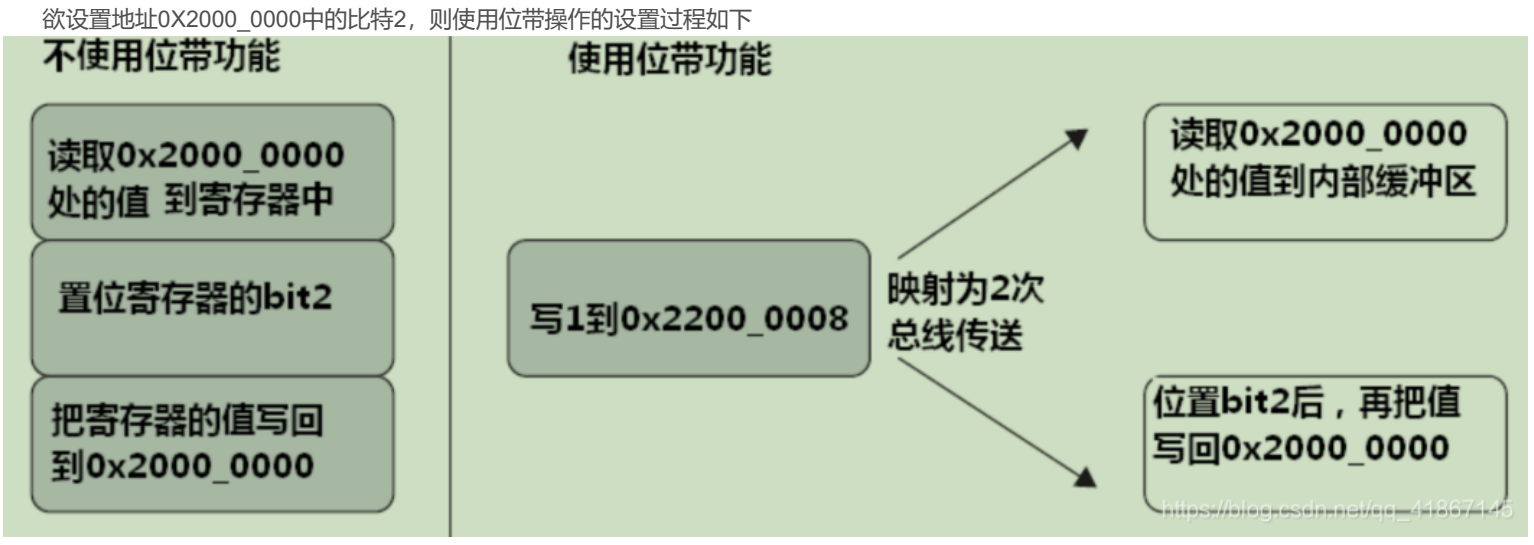
位带区与位带别名区的膨胀关系



把位带区中的每个比特"膨胀"到32位映射到基于0X2200 0000为基址的位带别名区上

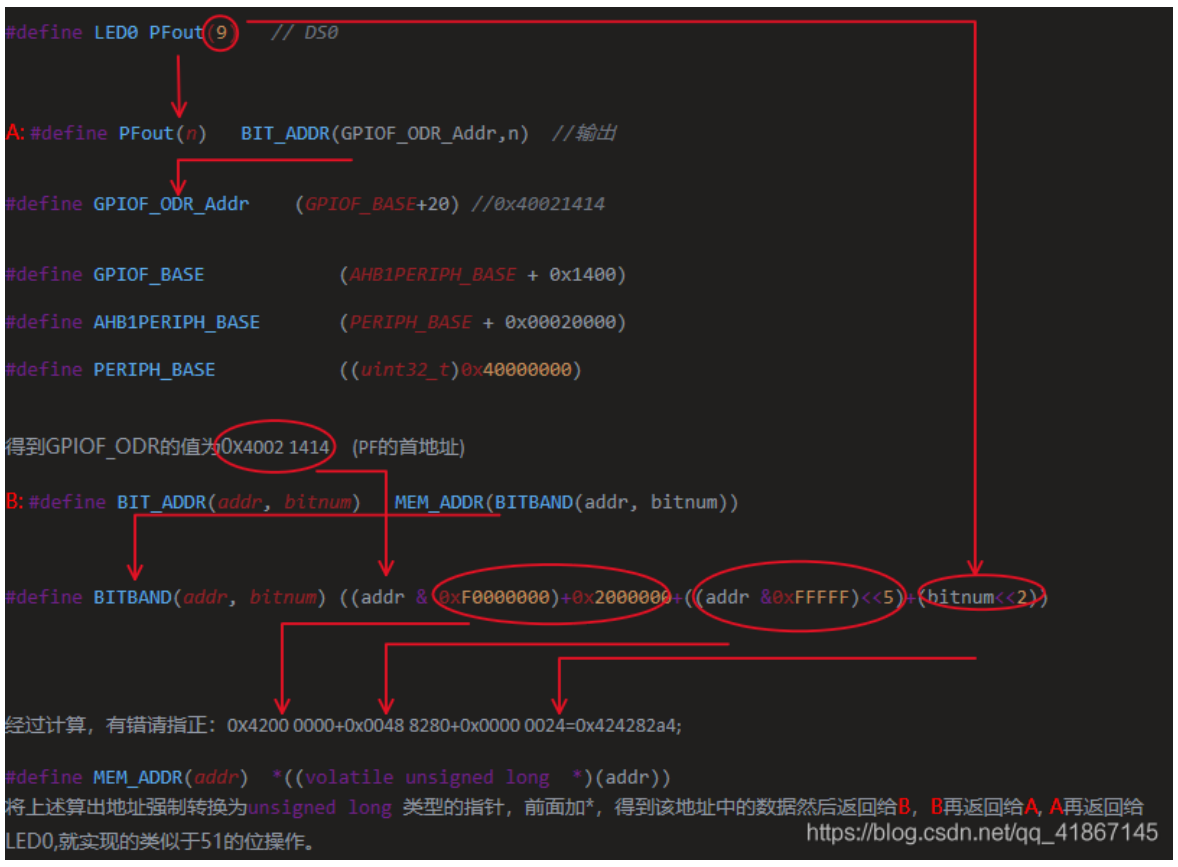
举例：

欲设置地址0X2000_0000中的比特2，则使用位带操作的设置过程如下



这样，可以方便对一个比特的操作。

5：位操作代码实现



可以看到最终是访问地址0X4242 82A4,正是在片上外设区的位带别名区的地址区间内，可以看我上面发的图。
这样，一顿操作就将PF9偷天换日到了片上外设别名区上。仔细一想，有点像C语言里面的函数传地址的概念。
一间房子（地址），我复制了一把钥匙放在另一个人的手上，这样，这个人就可以获取，修改这个房间（地址）里面的内容了。