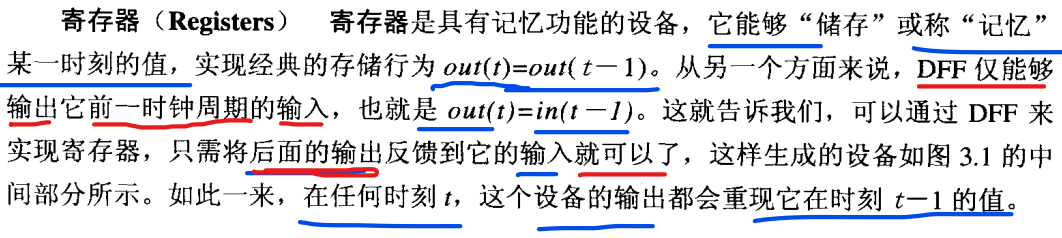
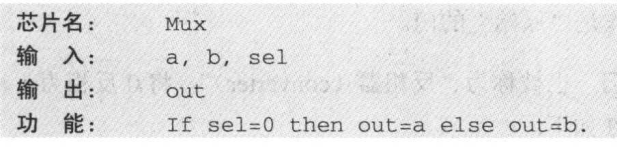
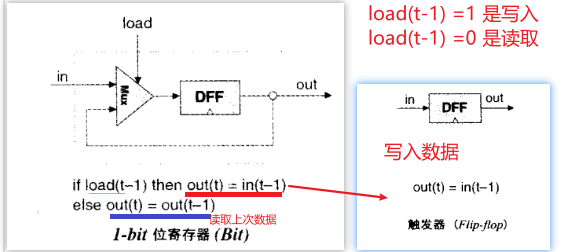
# 第三章 时序逻辑

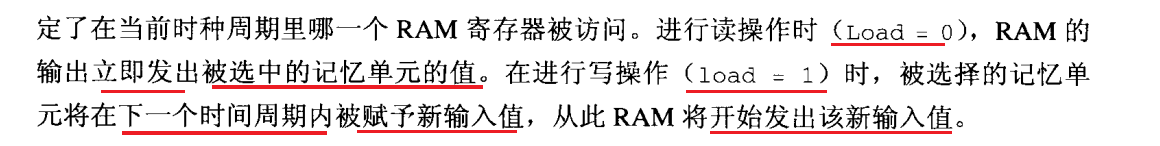
## 理论部分



## 存储器

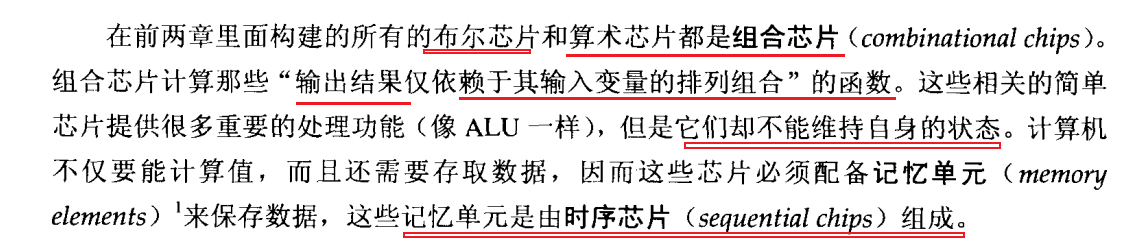


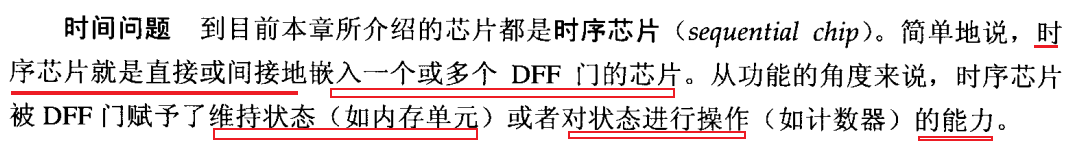




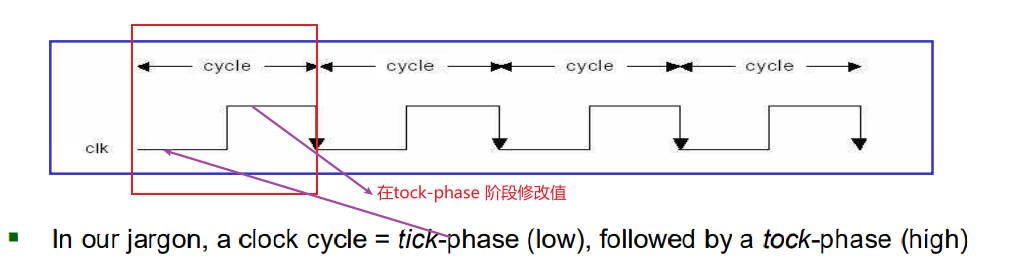
0:读取上一次的值 1:存储新值

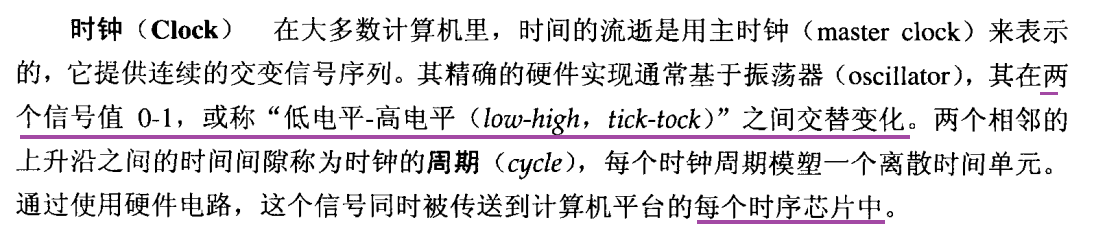
## 触发器





## 时钟

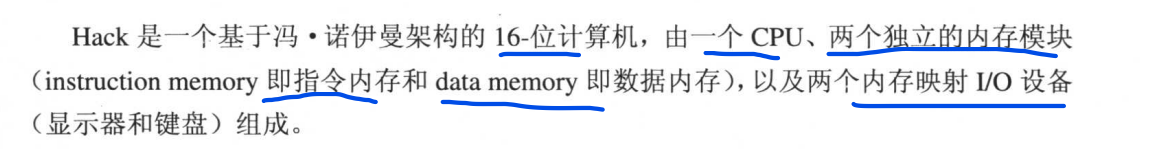


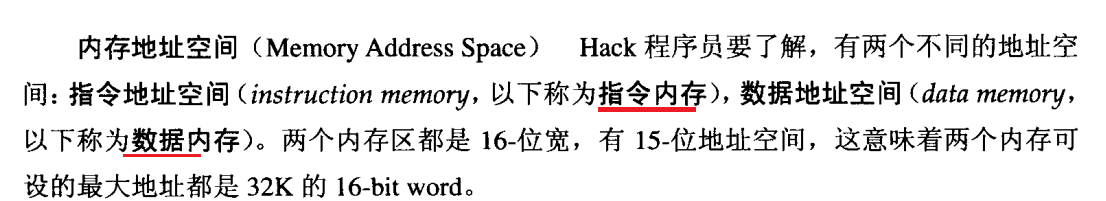


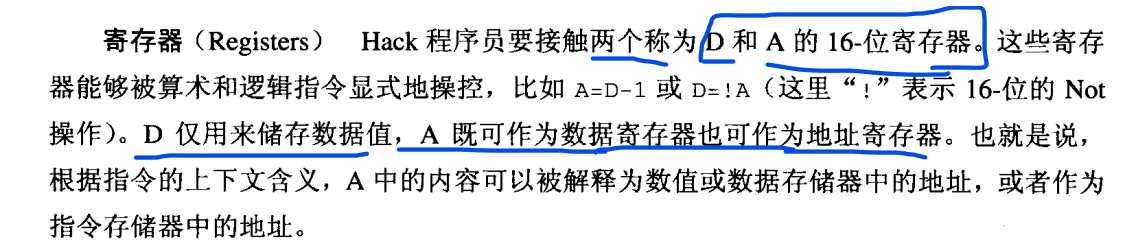
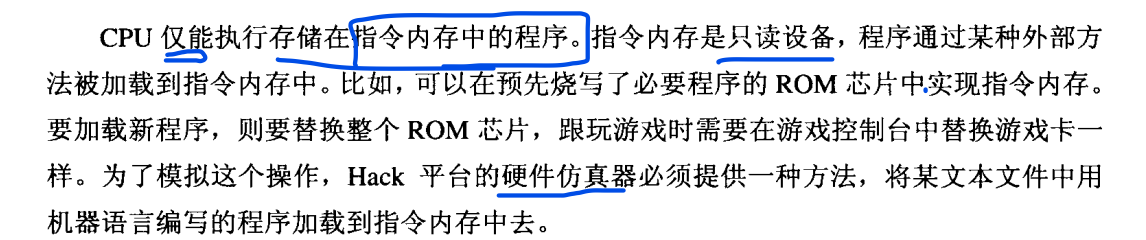




# 第四章 机器语言

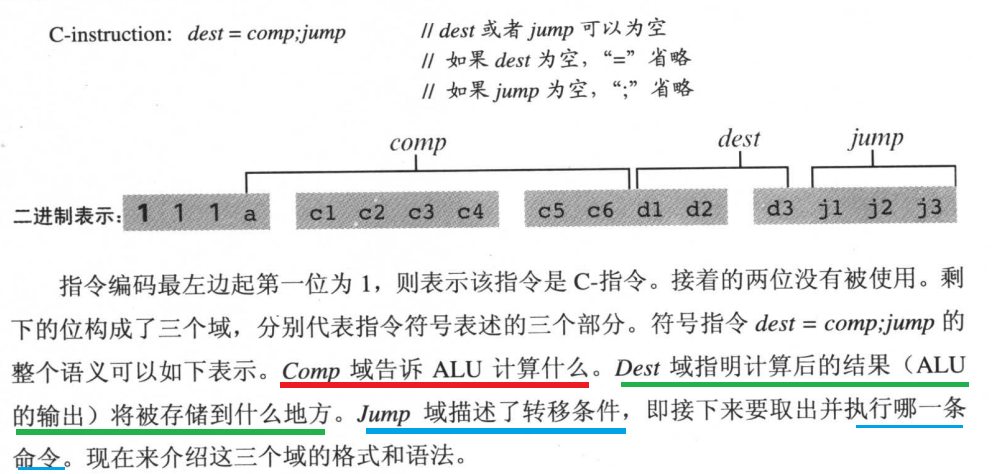




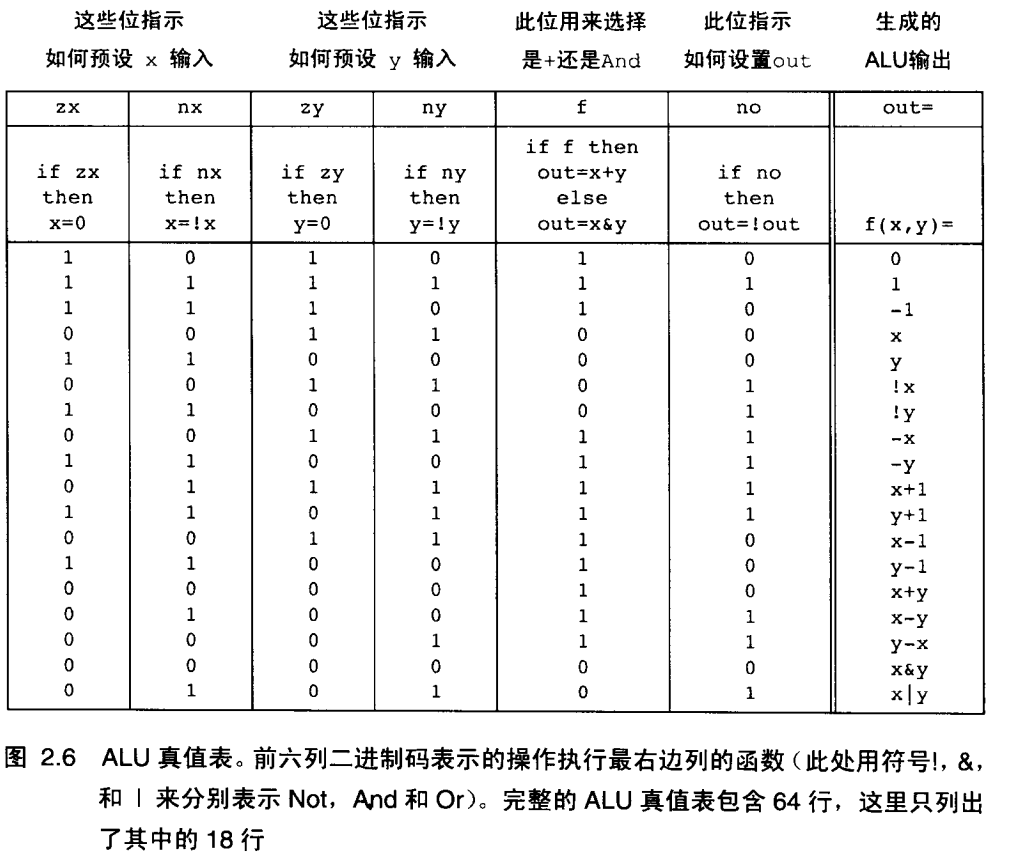


## C指令

### 语法

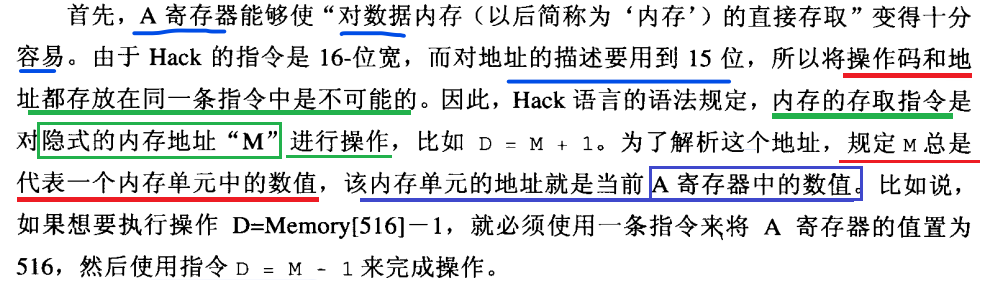


### Comp指令部分



## 理解难点

### 内存操作



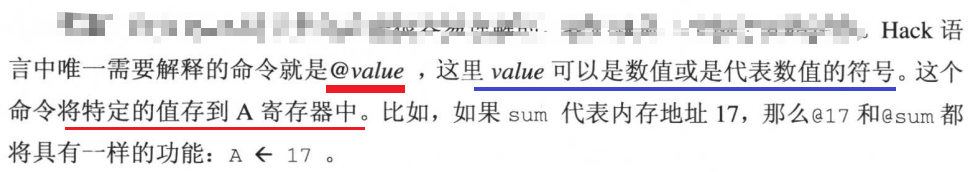
**内存操作 需要用到M(M就是隐式的内存地址)**

M 是内存单元的数值，也就是说他是一个数据，地址存放在A寄存器中。

M = Memory[516]

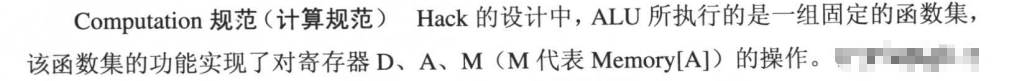
### 定义变量

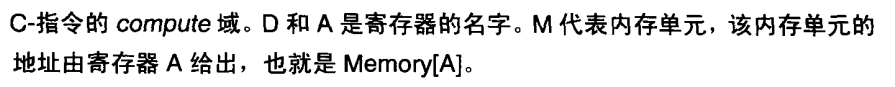
定义变量就是 将内存地址存放到A寄存器中。



也就是说 @value 的赋值操作 需要通过 M进行赋值。步骤如下：

@value 定义一个内存地址，该地址存储在 A寄存器上，赋值操作通过M进行进行赋值，因为M是 A存储器中存储内存地址的值（也就是@value的内存地址），因此M是对内存地址进行赋值操作的。

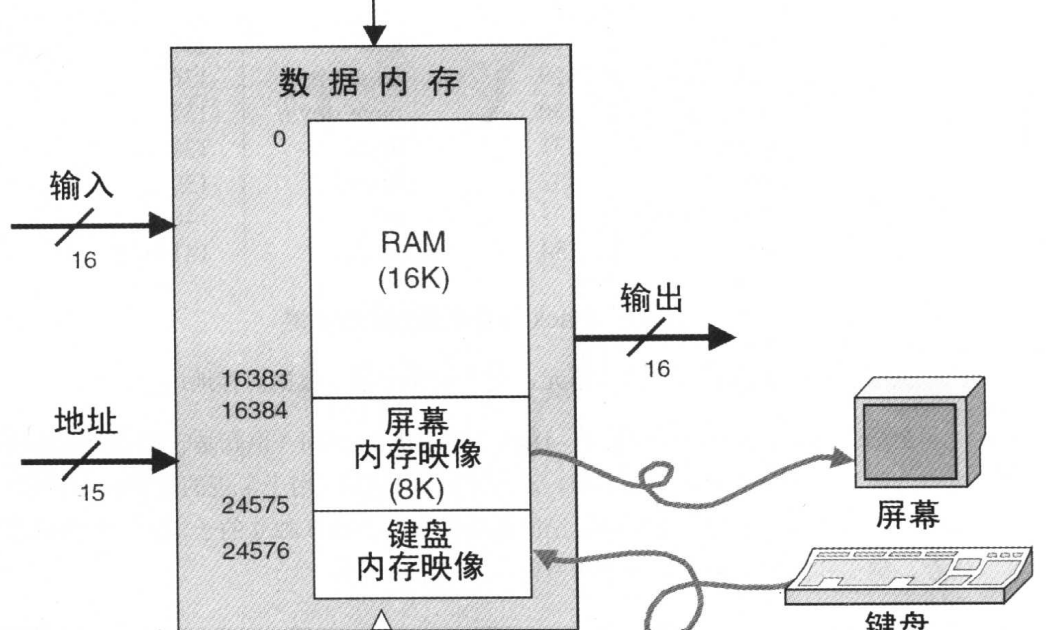
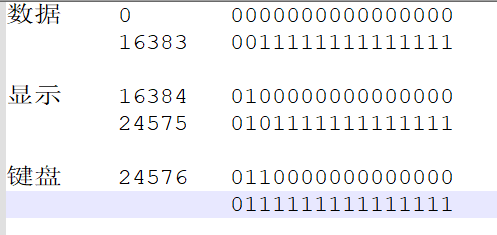




# 第五章 计算机体系结构

## 内存

### 区域划分

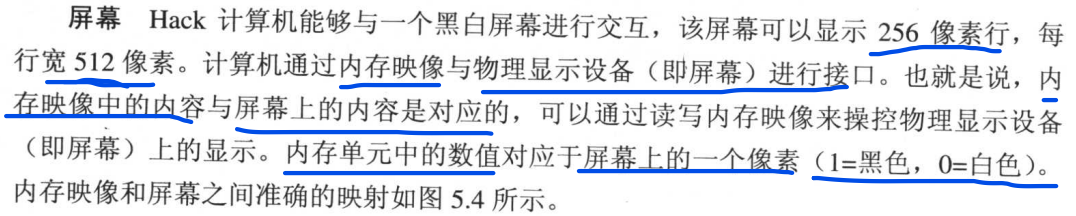
 

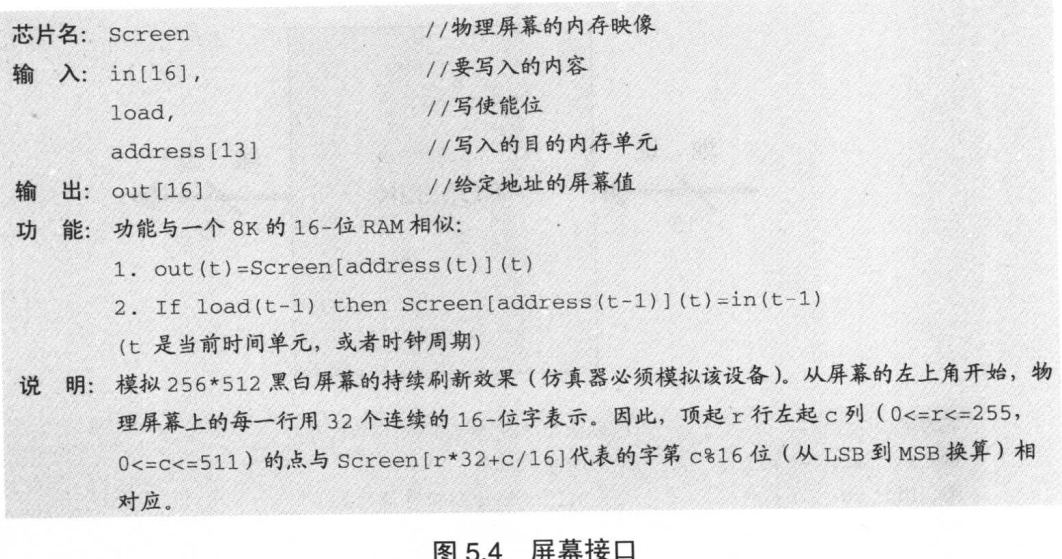
数据内存：是2^14=16384数量，也就是16K的内存大小

屏幕内存：是2^13=8192数量，也就是8K的内存大小

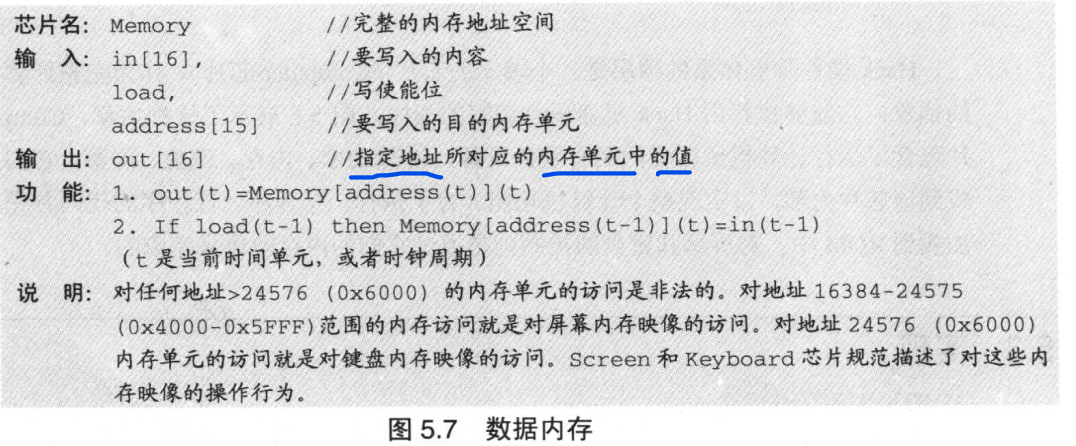
由于内存地址必须为正数，最高位必须为0，因此数据内存的地址为14位，显示器内存地址13位

### 屏幕内存规定

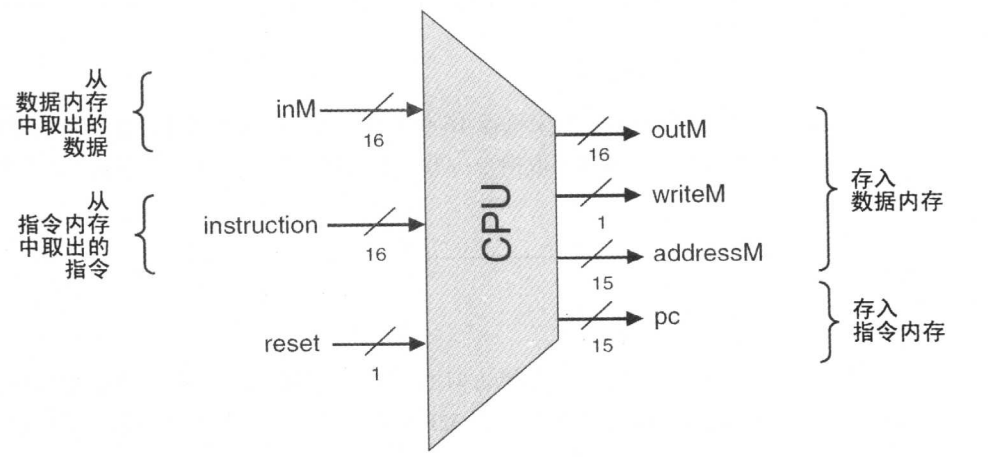


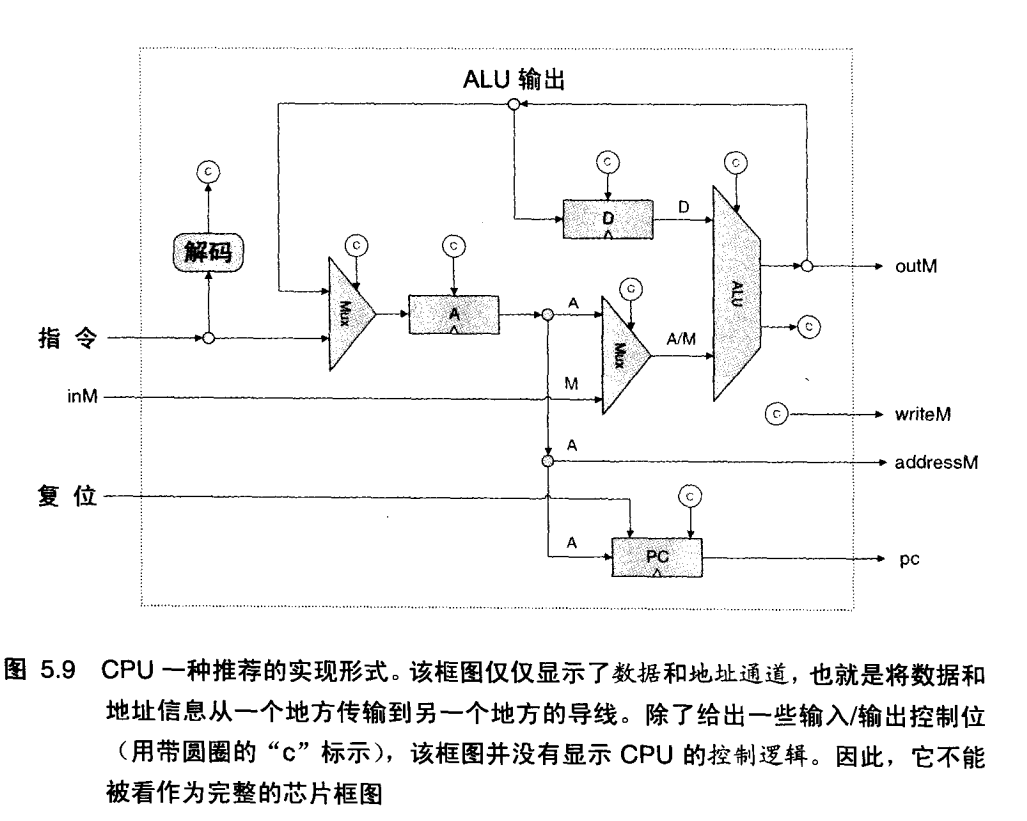


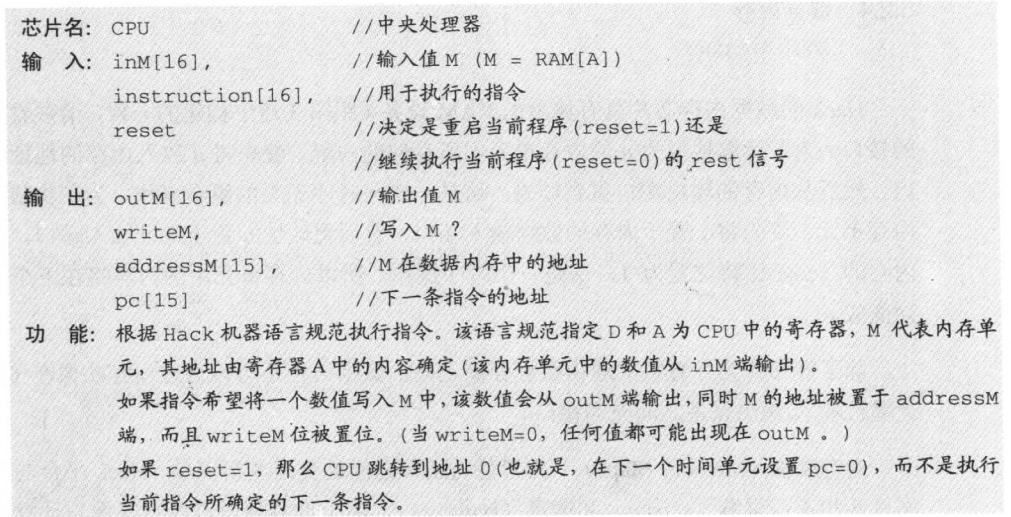
### 数据内存规定



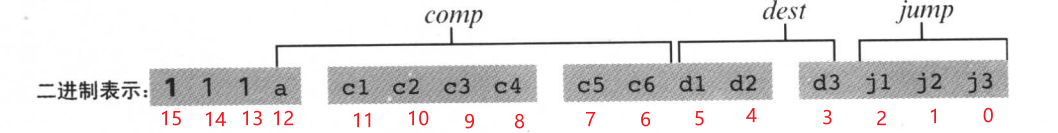
## CPU



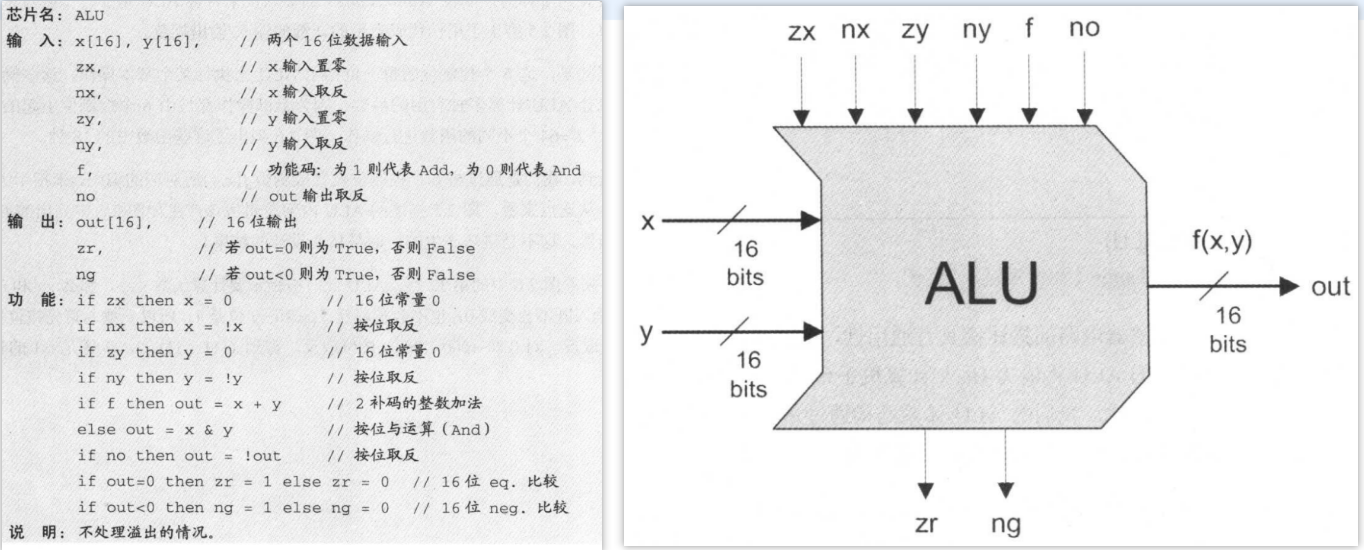




### 指令

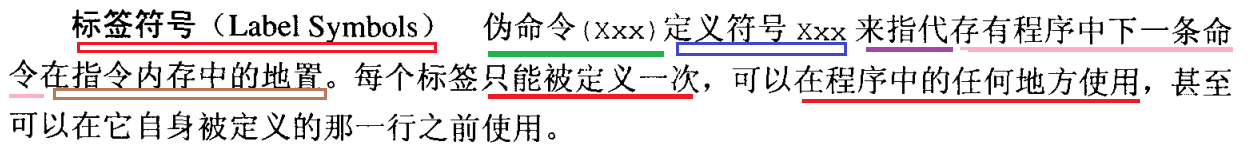


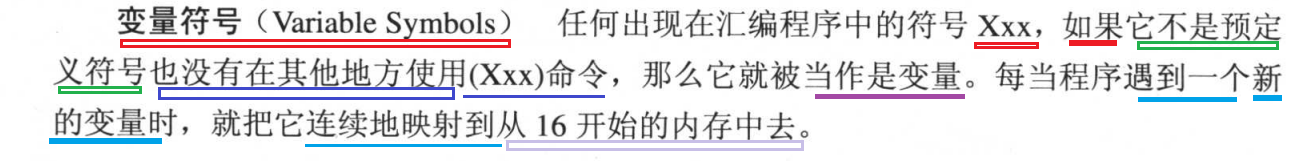
### ALU



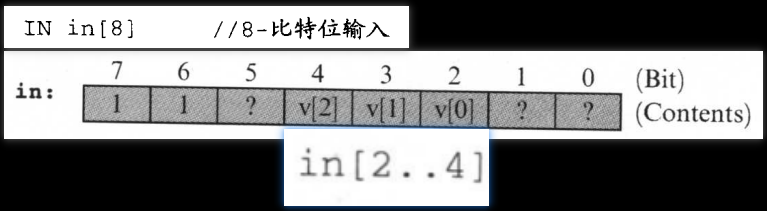
ALU：就是实现D 与 M/A的运算。

# 第六章 汇编编译器





# 附录

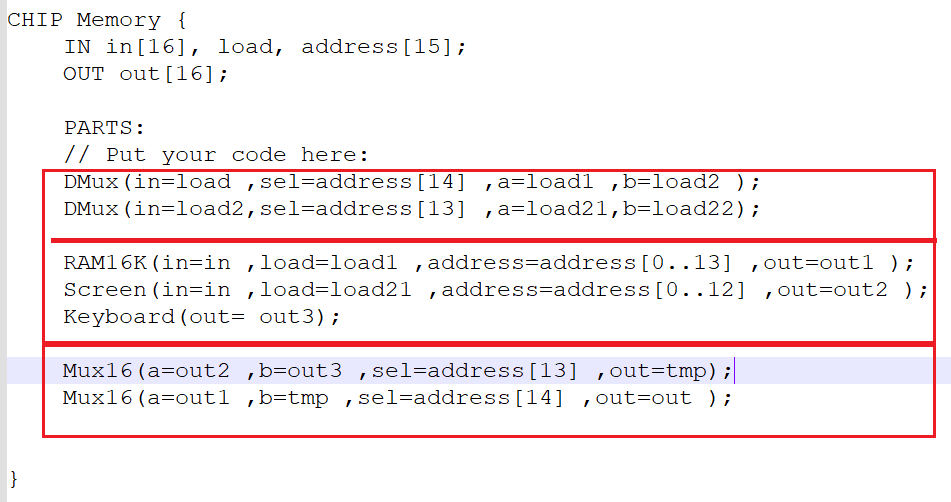




# 项目

## 5计算机体系结构

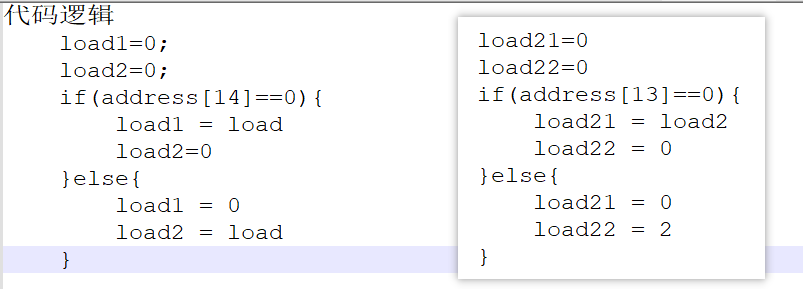
### 内存

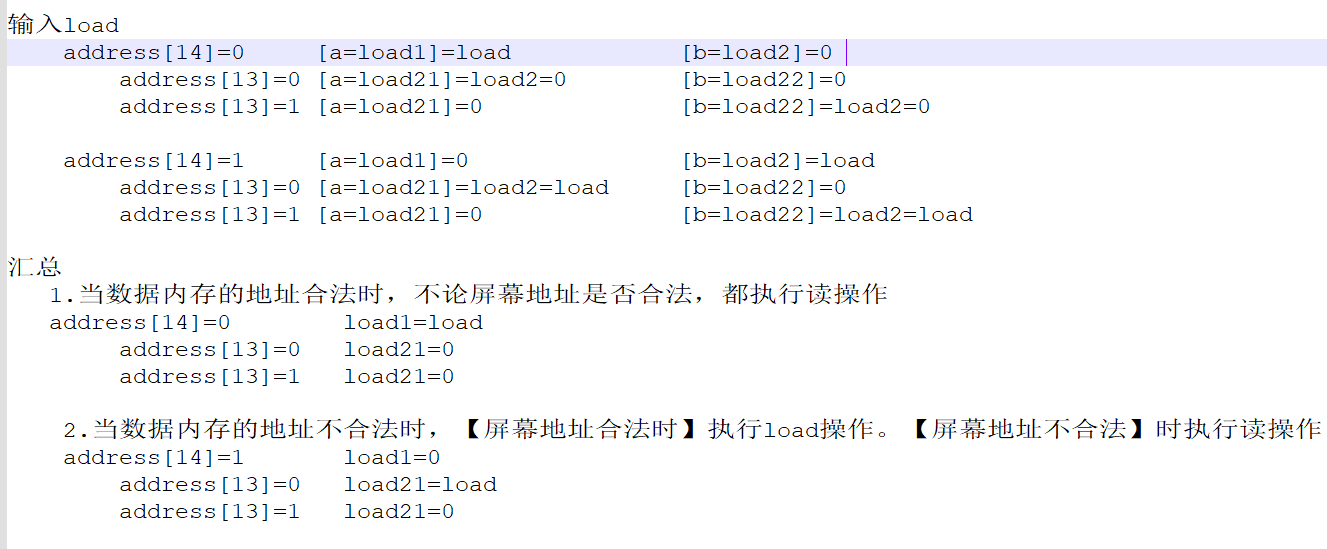


**代码解释1**

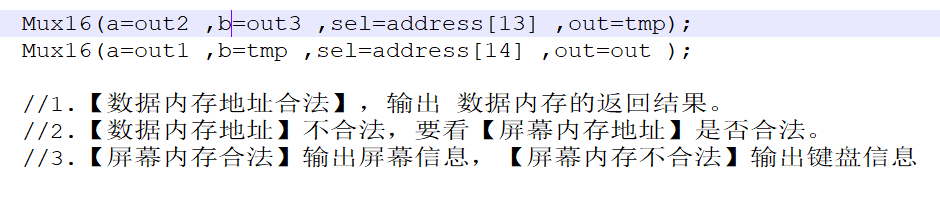
DMux(in=load ,sel=address[14] ,a=load1 ,b=load2 );

DMux(in=load2,sel=address[13] ,a=load21,b=load22);





**代码解释2**



### CPU

