在 32 位的 MIPS 指令系统中，I 类型指令的低 16 位中通常用来存储立即数或者地址偏移量，因此该指令除去寄存器操作数之外的操作数可能出现在立即数或者地址偏移量的位置。

如果该指令是一条 R 类型的算术运算类指令，由于 R 类型指令的低 16 位通常用来存储第三个操作数的寄存器编号，因此使用 R 类型指令处理 I 类型指令时可能会出现问题。

四

这种DRAM存储芯片采用二维地址结构，行地址线数量和列地址线数量相等。如果每个芯片有1M×8个位元，则这个芯片的存储位元阵列有1000000行和8列。

改正：1024行和8192列 具体查看课程总复习

首先，由 1M×8 的 DRAM 芯片构建的存储器容量为 1MB × 8 = 8MB。因此，需要 8 个 DRAM 芯片才能构建出 8MB 容量的存储器。

存储器地址线的数量取决于容量，每增加一个字节，就需要增加一根地址线。因此，8MB 容量的存储器地址线的数量为 $2^{23}$，即 23 根。

对于按字节编址的存储器，地址的低三位被用来确定字节在一个字内的位置，所以这三位地址不能用来产生片选信号。剩下的地址位用来产生片选信号，具体哪几位地址用来产生片选信号取决于 DRAM 芯片的布局方式。例如，如果每个 DRAM 芯片都对应一个地址范围，那么可以使用最高的几位地址产生片选信号。

对于 8MB 容量的存储器，存储最高段地址的芯片对应的地址范围是 000000H$ ~ $7FFFFF\_16H$。

（1）连续刷新两行的平均时间间隔是 4ms。

刷新地址与行地址有关

五

主存地址可以分为以下几个字段：

缓存行号（cache line number）：这个字段用于指定缓存中的哪一行。由于缓存有 32 行，所以该字段的位数为 5（二进制下，2的5次方等于 32）。

块内地址（block offset）：这个字段用于指定缓存块中的哪一个字节。由于块的大小为 8 字节，所以该字段的位数为 3（二进制下，2的3次方等于 8）。

主存地址（main memory address）：这个字段用于指定主存中的哪一个字节。由于主存的容量为 2的16次方 字节，所以该字段的位数为 16。

因此，主存地址的总位数为 5 + 3 + 16 = 24。

六

平均存取时间 = 0.9 \* 20ns + 0.1 \* (60ns + 0.6 \* 60ns + 0.4 \* 12ms)  
= 20ns + 0.1 \* (120ns + 4.8ms)  
= 20ns + 12.8ns + 4.8ms  
= 4.812ms

译码器是用来寻址的逻辑部件

七

数据相关的寄存器：$s1, $t0, $t1, $t2, $t3, $t5  
相应指令序列：L3: lw $t3, 15($s1) L4: sw $t5, 72($t0)

是的，在L3和L4中存在寄存器数据冲突的现象，因为$s1和$t0都被用于存储和加载数据

不能解决上述数据冲突，因为在第三个周期，两条指令都会读取R1的值，但是由于指令执行的顺序是不确定的，所以可能会出现冲突。  
若采用暂停的方式使上述程序能够正确执行，至少需要4个周期的暂停，具体执行过程如下：  
第一个周期：执行LW指令，将R2的值加载到R1中。  
第二个周期：暂停，等待LW指令完成。  
第三个周期：暂停，等待LW指令完成。  
第四个周期：执行SW指令，将R1的值存储到R2中。  
如果不存在冲突，则执行该指令片段所需的最少周期数为3，具体执行过程如下：  
第一个周期：执行LW指令，将R2的值加载到R1中。  
第二个周期：执行SW指令，将R1的值存储到R2中。  
第三个周期：执行完成。

状态存储：GRF DM

状态转移：IFU EXT ALU CU

ROM的优点在于数据不能被改变，下次打开文件时里面的内容也就是指令仍然存在。而且在电路运行期间，ROM里面的数据也就是指令不会被修改。

RAM的优点在于可写可读可复位，支持 load 和 store 指令，并且速度快于ROM，可以相对方便快捷的对里面的值进行修改。但由于RAM采用按字存储的方式，不支持跨地址读写数据；并且一次只能读写一个地址（4字节），因此无法直接实现lb，sb等指令。

GRF是临时存储单元，是CPU的核心部件。而Logisim提供的寄存器功能与实际的CPU中相同，一切运算都必须通过寄存器完成，同时是实现寄存器模块效率最高的元件。

****得到的指令在splitter中按字段分好****

DM中一个字是一个地址

加一个模块，当地址大于0x3000时减去0x3000

新的练习

2020年的第三题

立即数寻址，基址寻址，PC偏移寻址

立即数寻址：16位立即数代表一个十六进制数，用于指定操作数的值。  
基址寻址：16位立即数代表一个偏移量，用于指定操作数的地址。  
PC偏移寻址：16位立即数代表一个偏移量，用于指定操作数的地址，该地址是当前指令的地址加上偏移量。

补充：这三种寻址方式为立即数寻址（Immediate Addressing）、直接寻址（Direct Addressing）和寄存器寻址（Register Addressing）。在这三种寻址方式中，16位的立即数分别代表不同的地址或者是操作数。立即数寻址中，16位立即数表示某个地址或操作数；在直接寻址中，16位立即数表示一个目标地址；在寄存器寻址中，16位立即数表示一个操作数。

2020

四

sll $t0,$a1,2 //mulitply k by 4 ，int类型，k是下标，乘4→左移两位

add $t0 $t0,$a0; //address of v[k] ，k的地址

lw$t1,0($t0); //load v[]

lw$s3,4($t0); //load v[k+1]

sw$s3,0($t0); //store v[k+1] into v[k]

swSt1,4($t0); //store old v[k] into v[k+1]

Jr $ra

2022 考卷

## 用不超过两条指令将数值0xF78C033C分配到寄存器$t0中

li $t0, 0xF78C033C   
nop

## 用不超过两条指令从内存地址0xF78C000C的单元读取一个字到存储器$t0当中

lw $t0, 0xF78C000C  
nop

## 假如 $a0接收的数分别是15，64，-1，该函数分别循环多少次，相应的返回值分别是多少？

循环3次，返回值分别是15，64，-1

命中率=cache完成存取的次数/cache完成存取的次数+主存完成存取的次数

关于LRU的算法：cache块的总数= a的n次方，则计数器需要n位，也就是访问多少块就是那个答案

关于缓存题看这里

主存地址 = 主存块号 + 块内地址

主存地址结构

if(直接映射) //固定放

主存地址 = 主存Tag + 行号 + 块内地址

else if(组相联映射) //定位到某一组，任选组内一块进行映射:

主存地址 = 主存Tag + 组号 + 块内地址 //组间采用直接映射，组内的块之间采用全相联映射

else //全相联映射 //随意放

主存地址 = 主存块号 + 块内地址 //位数较多

主存地址 = 主存字块标记tag + Cache地址

Cache地址结构

if(直接映射) //直接定位到某一块

Cache地址 = Cache行号 + 块内地址

else if(组相联映射) //定位到某一组，任选组内一块进行映射:

Cache地址 = Cache组号 + 块内地址 //组间采用直接映射，组内的块之间采用全相联映射

else //全相联映射 //不定位，任选一个块

Cache地址 = 块内地址 //位数较多

比如这题主存4GB（按字节编址），Cache512KB，块大小16B；主存234589ABH的过程（假设Gache空白）。

Tag=2的32次方-块大小2的4的次方=28

#直接映射

主存4GB（按字节编址），Cache512KB，块大小16B；读主存234589ABH的过程（假设Cache空白）。

主存地址：32位

块内地址=16B=2的4次方

Cache分为：512KB/16B=2的15次方

Cache行地址：2的15次方

Tag=32-15-4=13位

标记：13位 Cache行15位 块内地址：4位

#Cache块数=？KB的Cache/块大小

#Cache 组号：Cache块数/？路组相联

#Cache的总容量=tag+cache的存储容量（比如题目里提到容量为多少）具体的tag可以看上面是哪个方式

#cache 组数=cache 容量/块大小\*？多少路组

#主存每组块数=主存容量/块大小\*Cache组数

#主存组数=主存容量/块大小

#Cache的主存储容量：Cache的Tag\*块大小转换的位数据（比如16B=128位 这里就是tag\*128） 再乘Cache容量/块大小

补充：块的大小\*（块大小转换为数据（比如16B=128）+Cache的tag）

#Cache的tag计算（有效位和脏位的时候）=有效位+多少个字就有多少个脏位+主存的tag

#对于直接映射 Cache的行号= ( Cache大小 ÷ 块大小 )

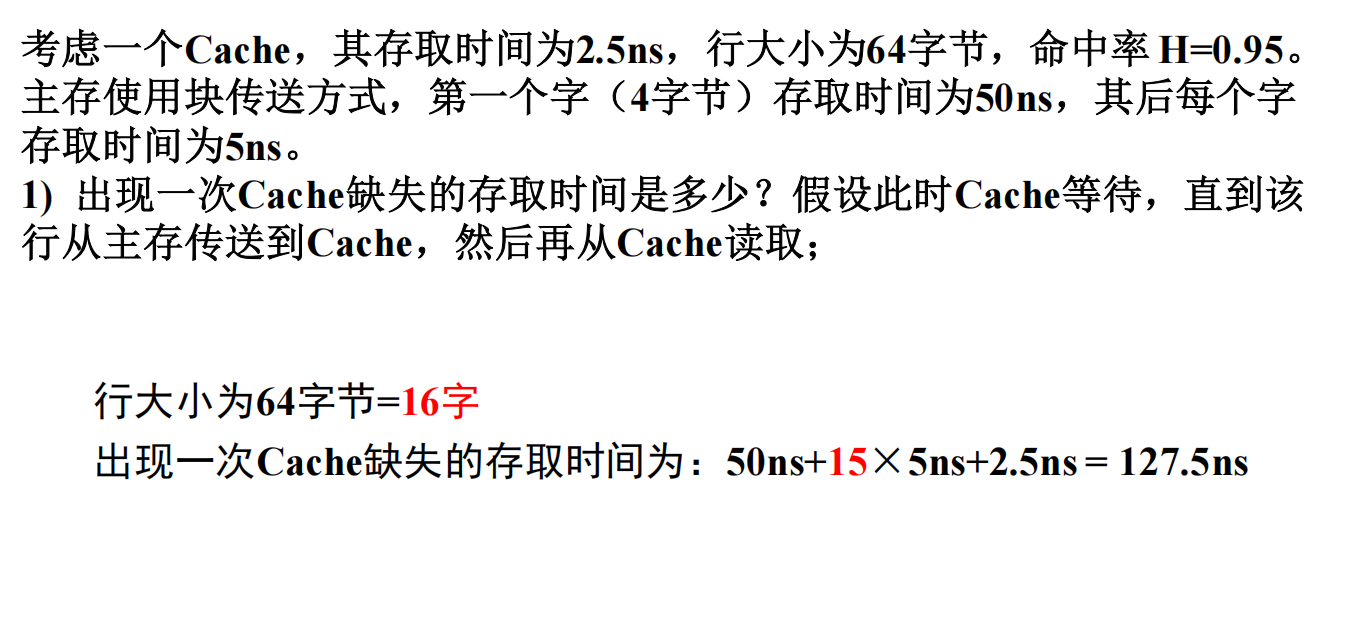


#上面的18位组内块的地址就是主存的tag

重点看习题集！！！作业的合集！我发现了套路，很多都是在作业这边出。

**#Cache**缺失的存取时间为：第一个字存取时间+块大小的字节转换为字-1\*后每个字的存取时间+cache存取时间

#一个字=4字节

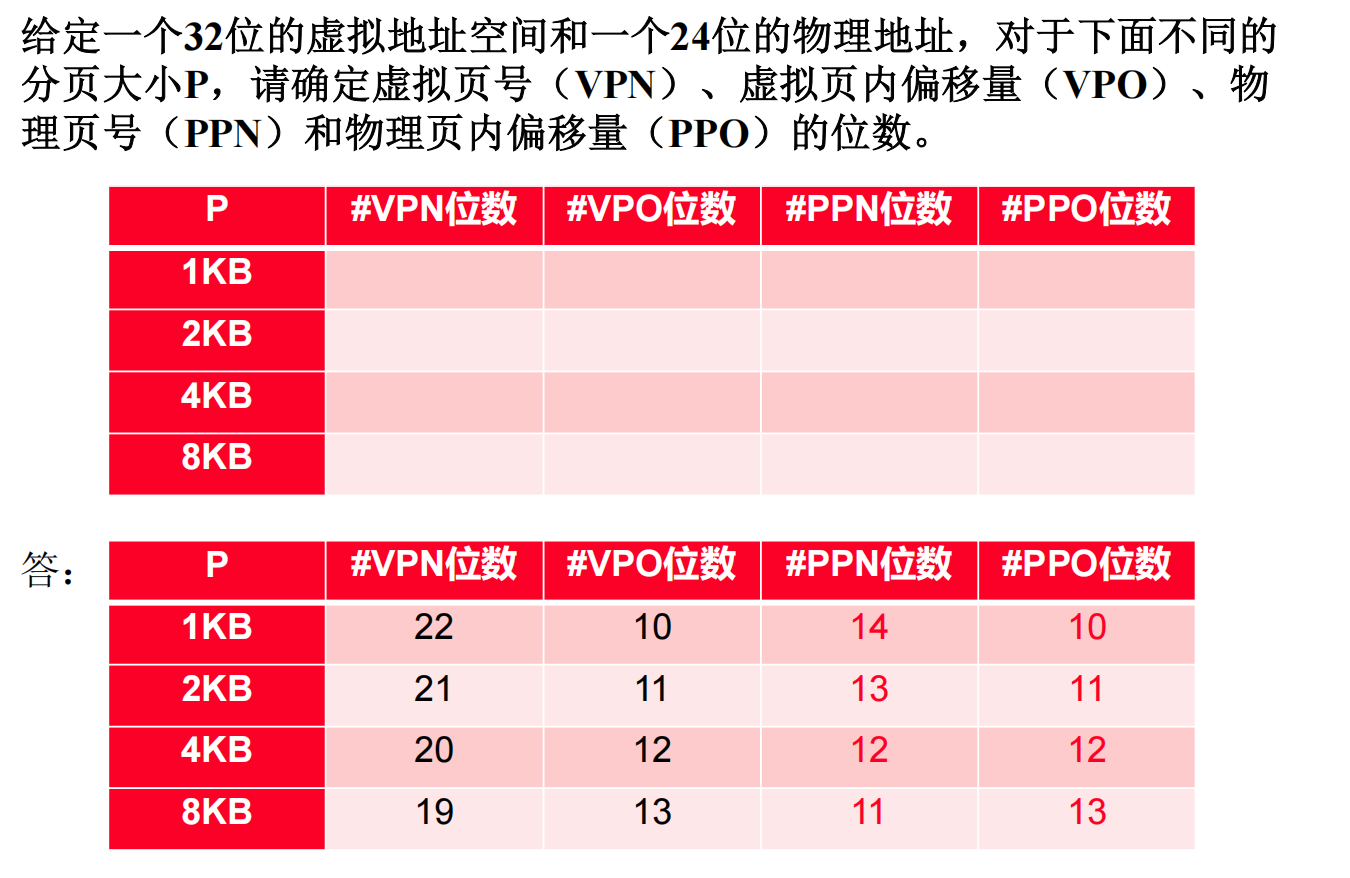


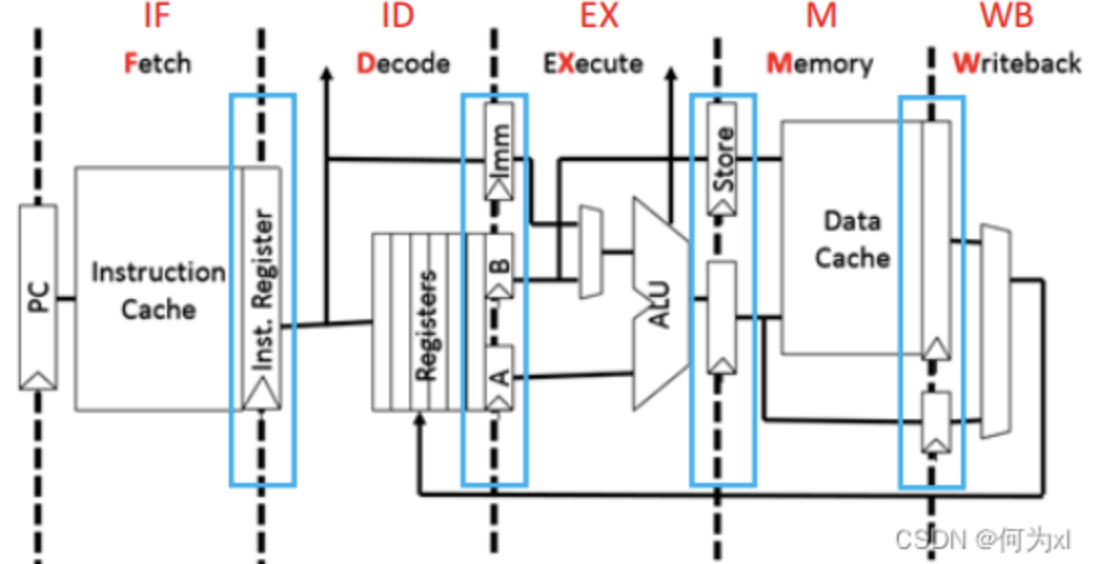
计算vpn vpo ppn ppo 位数很简单，比如32位虚拟地址，24位物理地址

答：1kb=2^10次方 所以vpn就是32-10=22 vpo就是剩下的10

同理 ppn 24位地址-1kb的10次方=14 ppo 就是10

#v代表虚拟 p代表物理





**lw指令写RF是在WB，sw指令读RF是在ID，**

**add指令 读RF是在ID，写RF是在WB**

#加速比=1/（1-命中率） +命中率\*cache访问周期/主存访问周期

#LRU就是最近最少使用

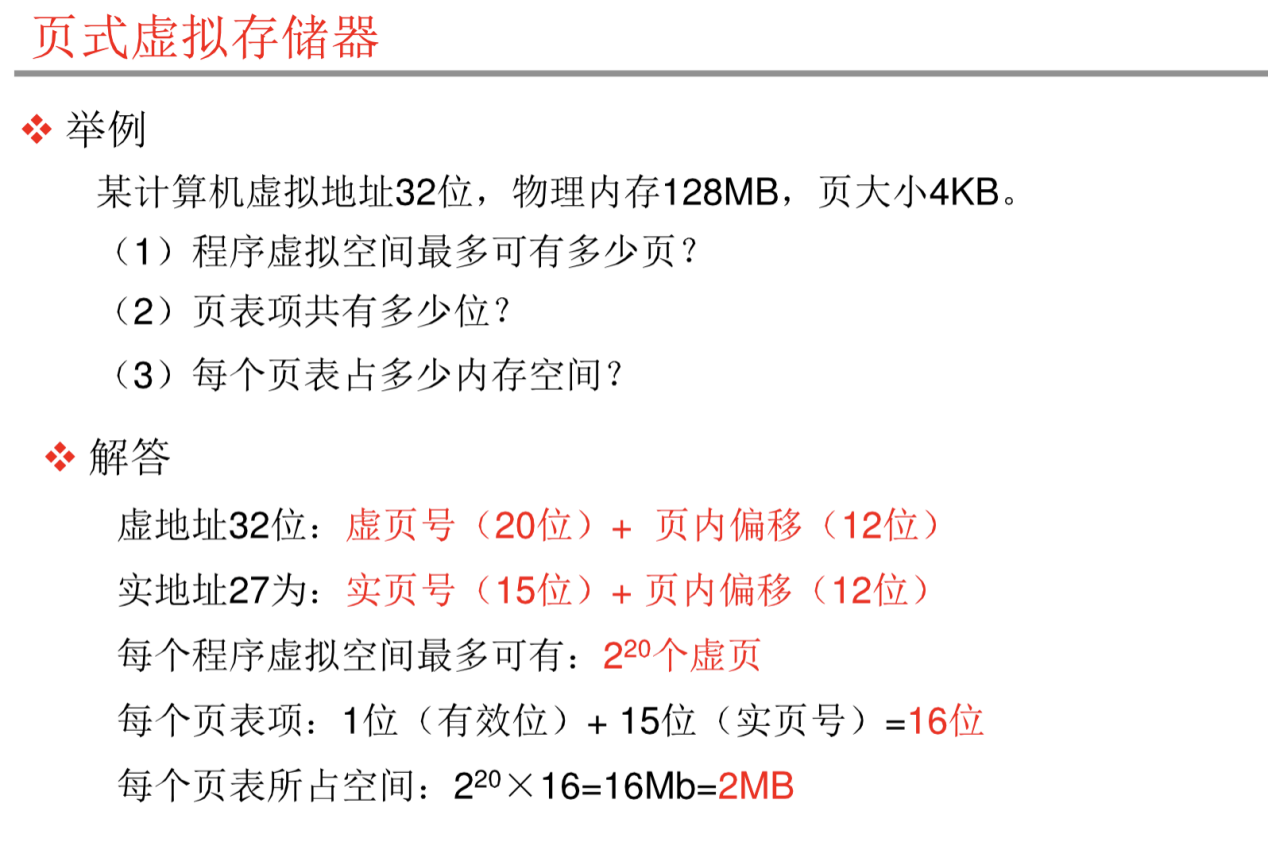
#LFU 最小使用频率

#页表项=物理空间大小/页大小=？位 （具体看要不要加有效位）

#页表所占空间=2的多少虚页号\*页表项

#异步刷新间隔=刷新周期 集中刷新间隔=刷新周期 分散刷新=刷新行数\*存储周期

#刷新计数器的位数取决于行数 所以2021的四。2答案是8



#mips32位中的立即数必然小于2^32

#具备n个并行部件的处理器相比，一个n阶流水线处理器 拥有同等水平的吞吐能力

#已知某计算机字长8位,其主存的地址码为20位=1024k x 8

#如果地址线A13与CPU断线,并搭接到高电平上,将会出现A13恒为“1”地情况.即只能正确访问地址中A13=1的RAM芯片1、3、5、7，而访问不到地址中

A13=0的RAM芯片0、2、4、6

#虚拟地址转换物理地址

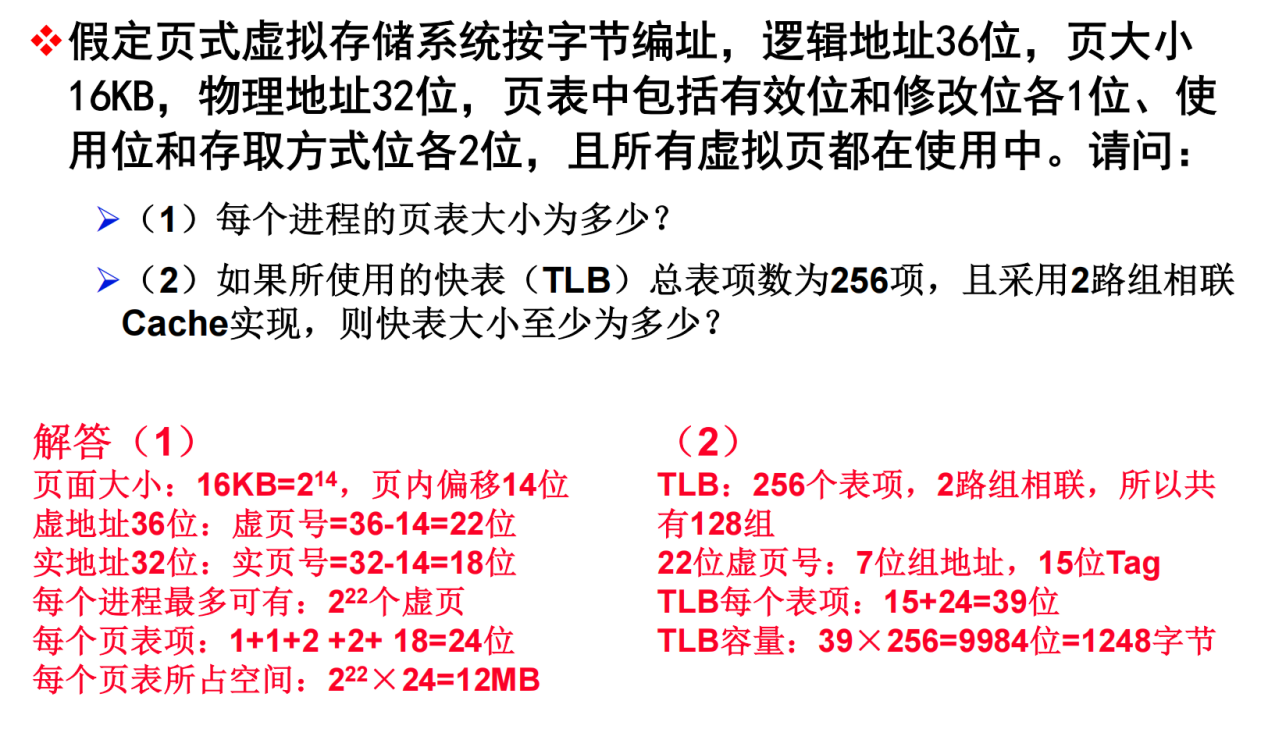
计算公式：块号 \* 页面大小 + 页偏移量 = 物理地址

页号 = 逻辑地址 / 页面大小

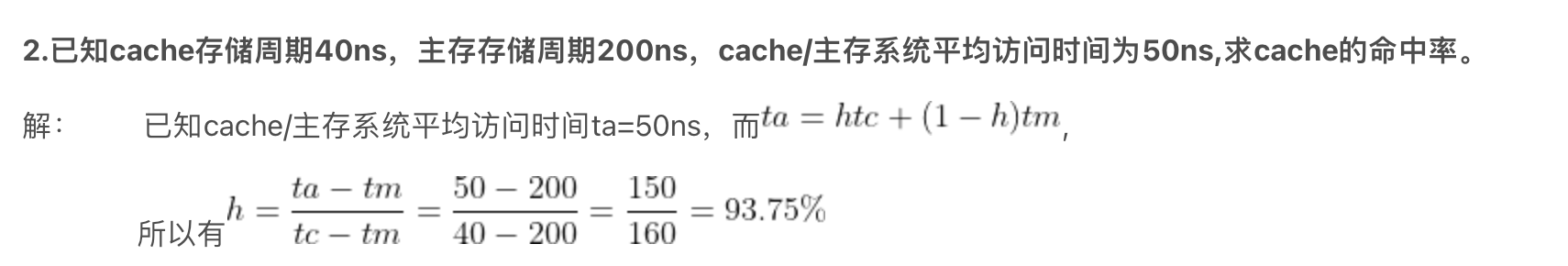
页偏移量 = 逻辑地址 % 页面大小

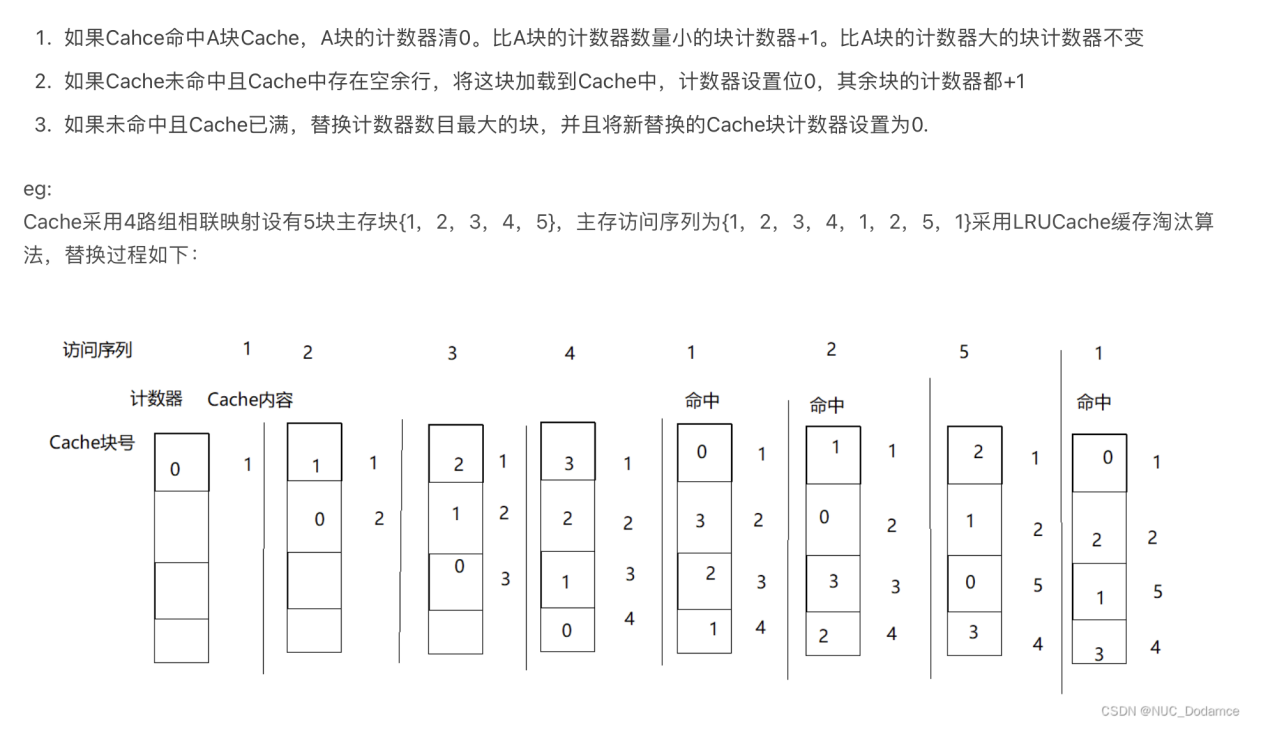
根据页表用页号找到相对应的块号

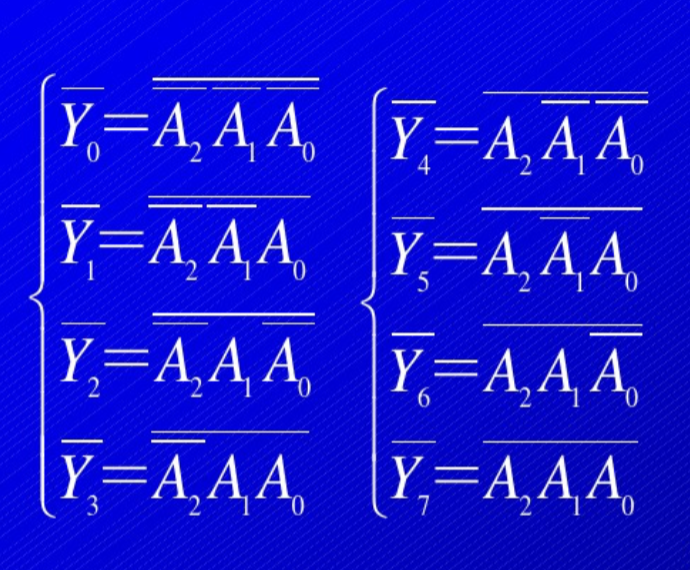
#**芯片地址空间：000H～FFF H = FFF代表着16\*16\*16代表着4K字**



#当只有ns的时候计算cache的命中率







#片选信号的逻辑表达式

#存储器带宽计算：总线宽度/存取周期

#Vcc接1 GND接0

2022

* 四
* 五
* 六的1
* 七的3
* 八的3，4
* 九的5（虚地址转换物理地址）