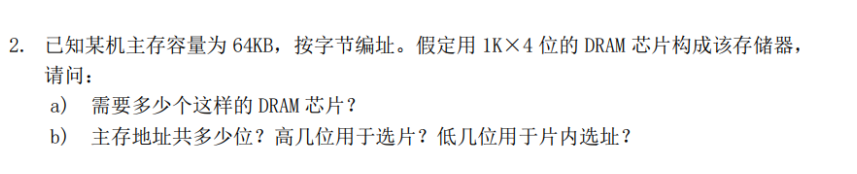
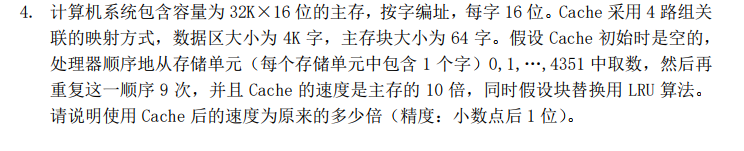
作业2

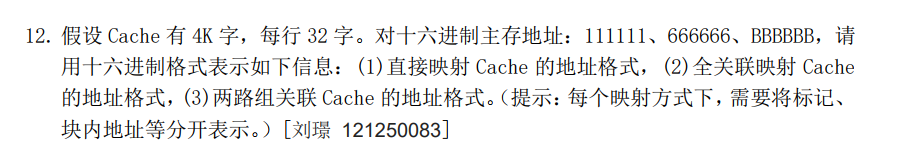


b)因为是位扩展，所以其实每一个字节，也就是每一个寻址单元是两块拼在一起的，所以说应该实际上是64块“板子”，6位寻块，10位块内寻址。



Cache是按照块来搬，但是hit不是根据块来hit

Hit是指每一个字在cache中的概率。每次搬进去一个块，但是不止一个字。所以只有每个块的第一个字的时候才会不能成功hit，之后的字都可以成功hit（所以一个块如果要搬进去，只有检查第一个字的时候没有hit要搬进去，之后三个字都会hit）

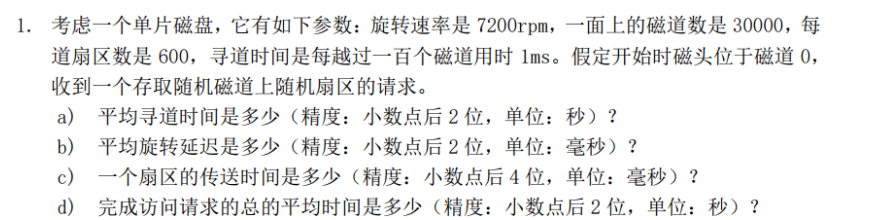
关联映射：tag+块内地址

直接映射：tag+行号位数+块内地址

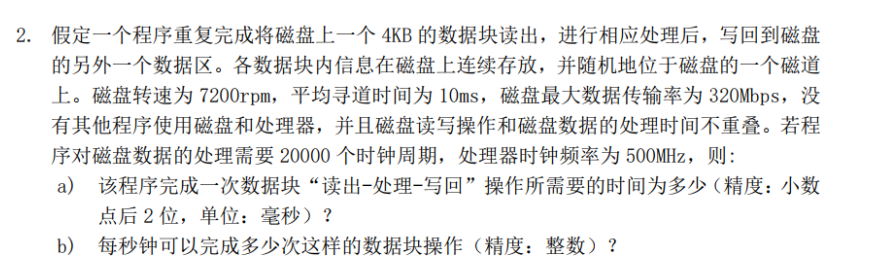
组关联映射：tag+组号位数+块内地址

注意：行号位数和组号位数同时只存在一个！不是先有几位是行号再有几位是组号：组关联映射本质上是一种特殊的直接映射

作业3

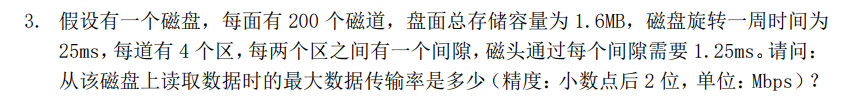


平均寻道时间：1.凡是涉及平均的基本都有“一半”这个概念：也就是越过一半磁道所用的时间：这里就是29999/2条磁道；2.初始时磁头在磁道0，要寻道的一共也就29999条



重要的隐含条件：

1. 信息连续存放：说明在存储的过程中不需要考虑跨越磁道的问题，磁盘数据传输保持在最大速率
2. 磁盘数据传输的最大速率就是不跨越磁道传输数据，数据都在一个磁道上的时候（因为这个时候没有横跨磁道的多余时间）



最大数据传输率就是最大数据传输率：这个传输率就是数值上的，跟怎么传输没有关系（比如说是读取半个磁道还是一个磁道，跟这些都没有关系），单纯的数值意义上的

其实就是磁盘的数据传输率会不断变化（数据密度，换磁道，过间隙等等），那个最大值，哪怕这个最大值是在某一瞬间产生，它也是最大数据传输率

所以最大数据传输率产生在读取一个扇区时，此时还没有通过间隙



海明码的生成规则：

只有一位1的很简单：二进制串对应数值标志着是Ci出错了

两位或者三位的：对应数值一共有8个，从最小的0011到最大的1100分别按顺序对应着Di。每一位1标志着在Ci中出现了Di。

比如：1011=D7，那么就说明在C1,C2,C3,C4的生成中都有D7

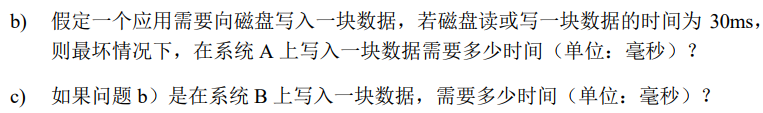
注意：C4C3C2C1的排列和D8D7D6D5D4D3D2D1的排列都是逆序！！！！！！

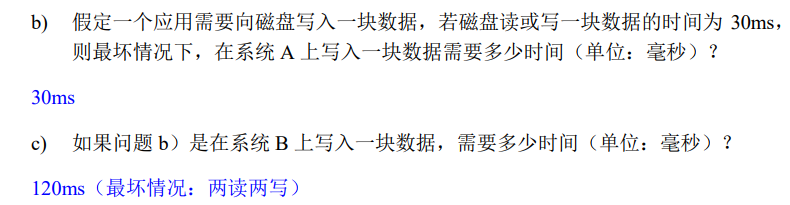
**校验码和数字码的下标都是逆序！！！！！！**

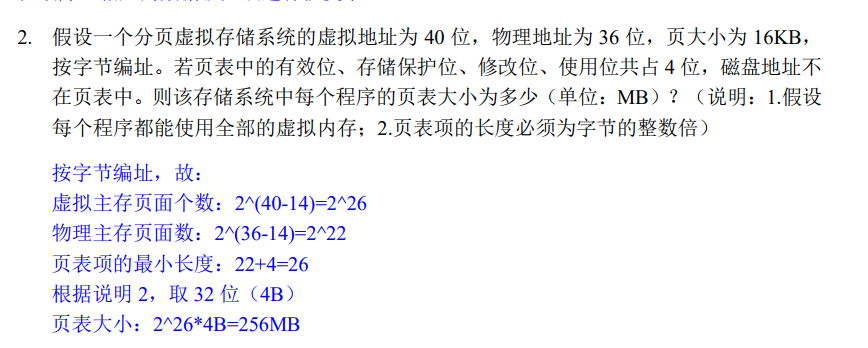
作业4

RAID里面涉及读写：就是那个两读两写还是几读几写还是直接写这种：

特别注意RAID4和RAID5都是两读两写，但是RAID6好像是三读三写（？两个校验码和一个数据本身）







注意：

1. 页表中存储的“地址”是物理页框号，并不是实际的物理地址
2. 要注意题目中的特别条件（比如“页表项的长度必须为字节的整数倍”—这是一个很明显很奇怪的限定，说明要做一些约等—不然平白无故给你这个条件干嘛！

1.在分页式当中，可以同时存在页和块这两种东西：

块是内存中比较基本的存储单位（比字节高级一点），分页是为了更便于管理而分的更大的内存块

Cache映射一般使用块；TLB，页表一般使用页，这两种不冲突

2.如何判断是虚拟地址还是物理地址？根据给定地址的位数：16位是虚拟地址；12位是物理地址；这里给出的是067AH说明是虚拟地址

3.步骤：

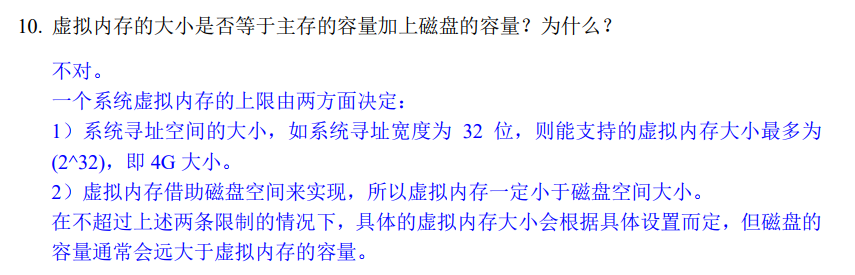
拿到虚页，看一下虚页在不在TLB里面，不在就看在不在也表里面，得到对应的实页的页号

根据实页的页号和虚页页内解析的offset组合形成内存地址

访问cache，看看这个地址的元素在不在里面（检验tag），如果在根据块内偏移获得指定的元素

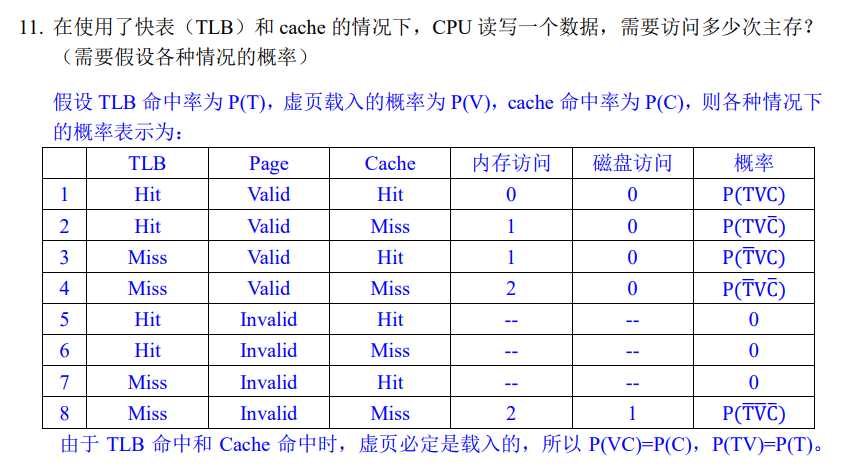
拿到属于某一个虚拟地址的元素：是一个字节，最终是要落到一个字节上；在这个过程中既有分块又有分页，但最终还是落到具体地址上（完整内存地址）：虚页到实页号的转换+offset获取实际地址的规则成为了跳板

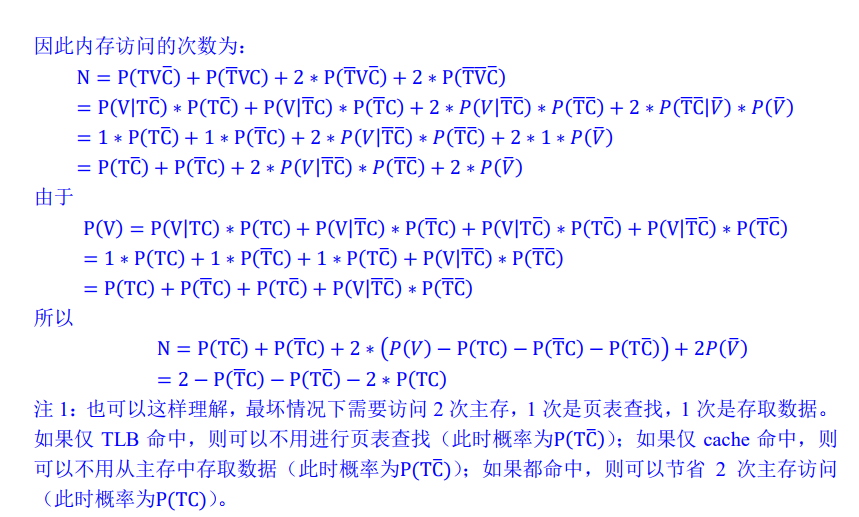
等到一个数据依托虚页被加载到内存里面以后，分页式的任务就完成了，这个时候就不要再考虑任何和页有关的东西了，考虑如何从一个内存地址拿到元素？--cache映射



虚拟内存的大小由两个因素影响：

1. 寻址空间的大小：这标志着内存可以访问多少量级的地址（比如4G）
2. 磁盘空间的大小，虚拟内存的大小肯定远小于磁盘空间（很简单啊，磁盘自己还要存储东西呢，都给虚拟内存了磁盘用啥？）





采用特殊情况排除法：

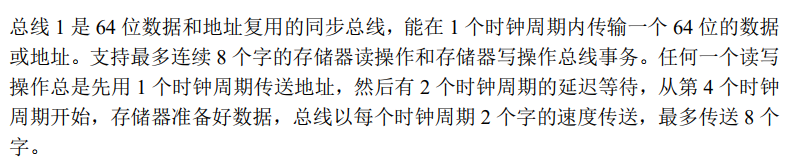
也就是说，最坏情况是访问内存两次，一次访问页表，一次存取数据（也就是页没有加载到内存当中去的情况），但是如果页表命中/TLB命中就会减少这个次数，所以考虑在两次的情况下做减法。

（有点类似于cache的命中：一定过cache，再考虑进入内存的情况；这里也是，假设最坏情况，再从这个上面减掉）

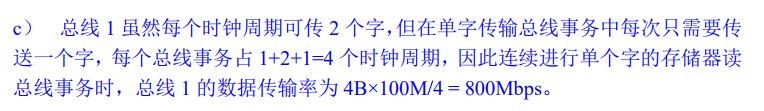
总线的传输可以多不可以少，但是时钟周期是不变的

比如一个总线是64位，字长是32位，一个周期传送数据的时候是传送2个字（总线的位数），但是如果传送1个字，也是一个时钟周期。为什么？

因为这一个时钟周期是指总线从存储器到cpu传送所需要的时间（这个时间跟数据位数无关），传两位只是因为这样传输率最高，但是也可以传一位，但时间不变







数据和地址一起传送，在写的时候会比较快（因为地址和第一个数据一起来，地址解析完毕可以直接把第一个数据写进去；但是如果是地址和数据分开传的话，地址解析完毕这个时候再开始发数据，数据发送完毕再写进去），但是在读的时候速率没有变化

但是地址和数据一起传的时候第一个数据的周期会比较长：因为要先解析地址再写入数据，第一次和第二次数据传输之间会有比较长的空闲时间用来处理这个事情

在这个题里就是这个“两个周期”的等待延迟。

