节选自清华计算机考研912真题（2016-2020年）计算机组成原理部分，按照一般教材对知识点的分类形式对这部分考题做了简单的分类，并配有本人相应解答，仅供参考，如有疑问，可以在文章下方留言。

**运算器**

**判断(1\*5')**

浮点数float y,若y\*y仍然在浮点数表示范围内，则y大于0 (2020年)

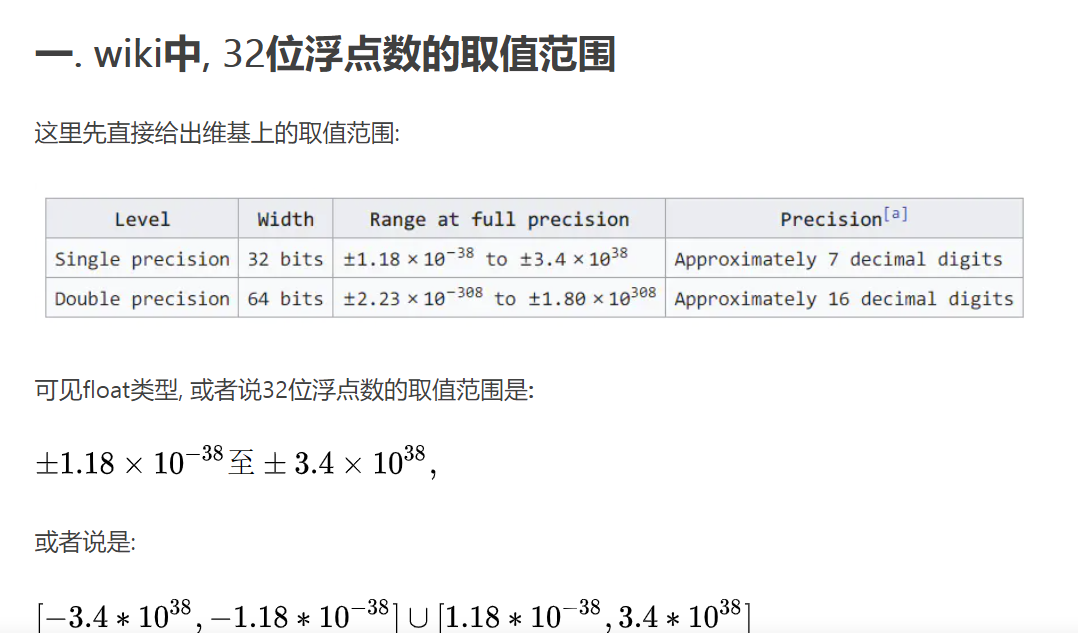
错误❌。

容易举出反例：令y=-1.0，y\*y=1.0在浮点数的表示范围内，此时y<0。

浮点数的表示范围很大的，几乎相当于负无穷到正无穷了，

所以很容易举出一个 负数\*负数 的例子

不晓得这题老师在考什么，怕别人不知道浮点数可以表示负数？？



整数Int x<0,则必有-x>0 (2020年)

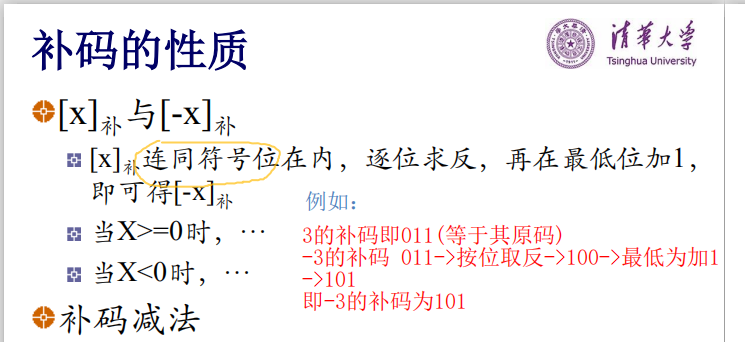
错误❌。

int 整数的取值范围为 [ -2^31 , 2^31 - 1 ]

最小负数即为 -2^31 ，其只有补码形式，即为0x8000 0000

求其负数为：将所有位的补码按位取反，最后再加上1，得到的仍然为补码 0x8000 0000，即其还是 -2^31

所以说对于 int a=0x80000000，-a依然是0x80000000。



假设x类型是C语言中的int，若x>0，则x\*x>0 (2019年)

错误❌。

当x\*x 超过int上限表示的范围，即溢出了，那么就会得到不可预期的结果。

这个结果和可能为0，可能为负数，可能为正数，要看编译器

我在codeblocks本机上测了好几个溢出的数输出都为0

C 语言若 int x,y 若 x>y，则-x<-y (2018年)

错误❌。

反例：

-2^31 + 1 > -2^31

但是 -（-2^31） = -2^31 不变，仍然等于其本身

-( -2^31 + 1 ) 是一个正数，其显然大于一个负数

所以存在 x>y，则-x> -y

好几道题都是在考计算机数据表示范围以及最小负数的特殊性

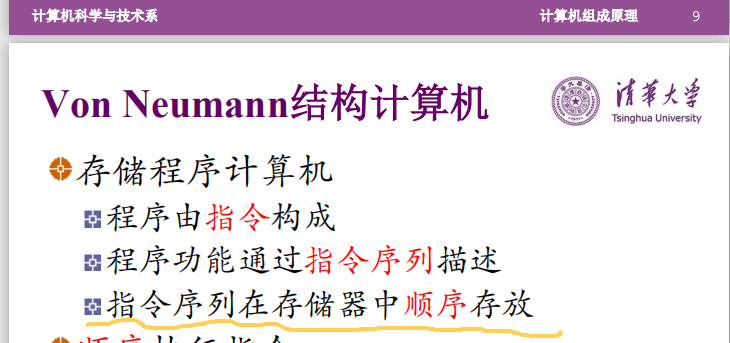
因为计算机里的数都是用补码表示的，且最小负数最特殊，和我们普通数学里的数的规律特点不一样

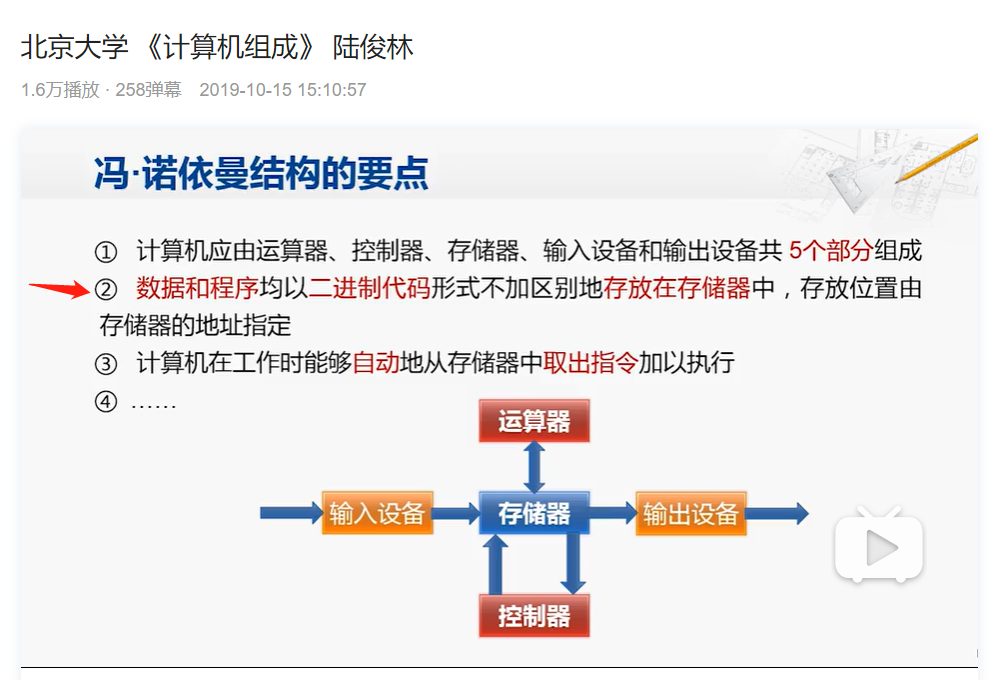
所以往往普通数学中成立的结论，这里会不成立

冯诺依曼结构体系中把程序也当做数据放在内存中 (2019年)

正确✔。

程序就是指令序列，要放在存储器中





**填空题(2+2+3+3=10)**

十进制整数+1234的32位的补码是：\_\_\_\_\_\_\_（16进制，小端机表示） (2019年)

答：0xD2040000

正数的补码就是其原码

先通过除2取余得到其二进制

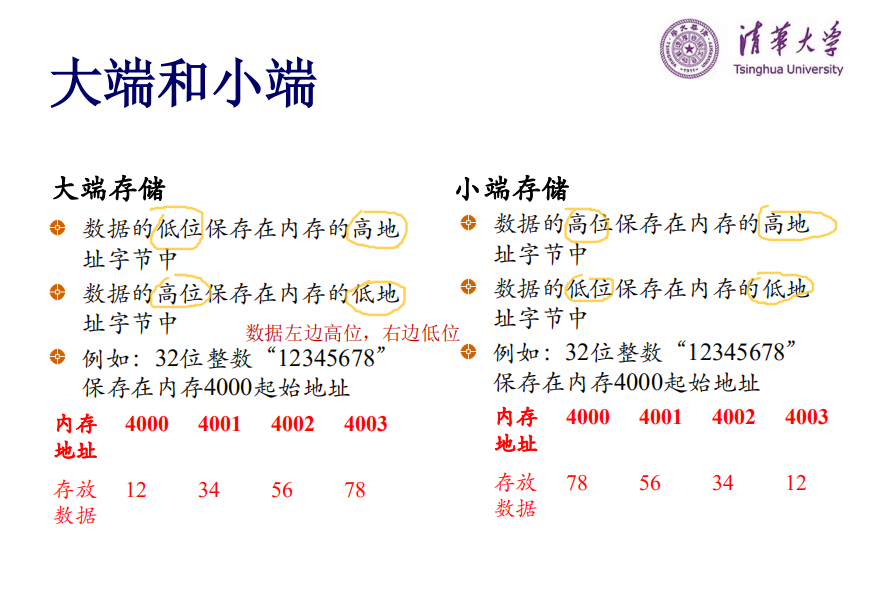
(1234)D = 0000 0000 0000 0000 0000 0100 1101 0010 = (0 0 0 0 0 4 D 2)H = 0x000004D2

而小端表示是数据高位放在内存高地址字节中

那么0x000004D2按小端存储就是 0xD2040000 （两位一组前后倒过来）

如果其按大端存储就是 0x000004D2

**tips**：按小端存储，就是把算出来的数据两位一组前后倒过来，大端存储不变



-2017 的 32 位补码表示\_\_(16 进制或 2 进制)。 (2018年)

-2017的原码：1000 0000 0000 0000 0000 0111 1110 0001

除符号位外,按位逐位取反 ： 1111 1111 1111 1111 1111 1000 0001 1110

最低为加1： 1111 1111 1111 1111 1111 1000 0001 1111 = 0xFFFFF81F

即-2017的32位的补码为：0xFFFFF81F

十进制单精度浮点数-27.625在IEEE754浮点标准下表示：\_\_\_\_\_\_\_（16进制） (2019年)

27 = 11011

0.625 = 0.101

-27.625 = -11011.101

规格化：-1.1011101 × 2^4

(-1)^1 × （1+.1011101）× 2^(131-127)

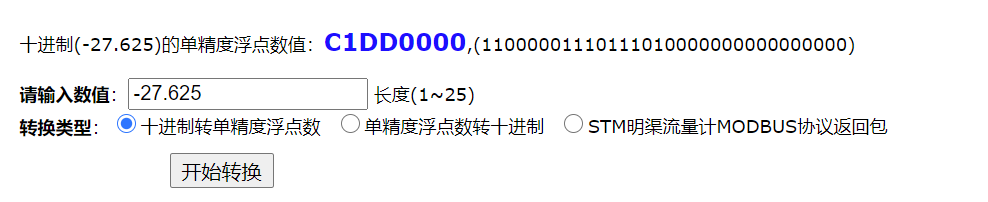
符号位 = 1 ，阶码 = 131 = 10000011 ，尾数 = 1011101

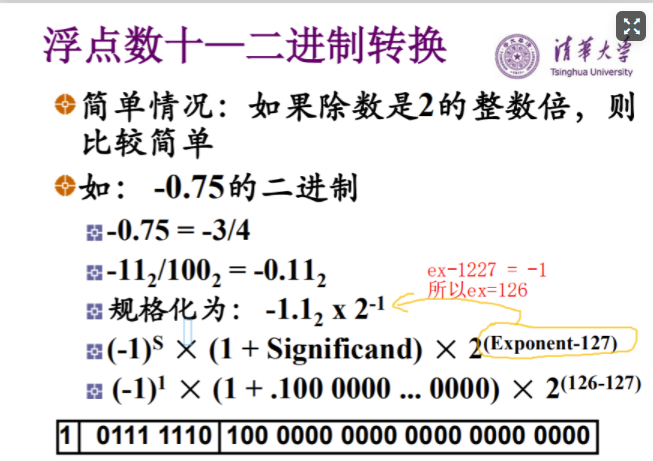
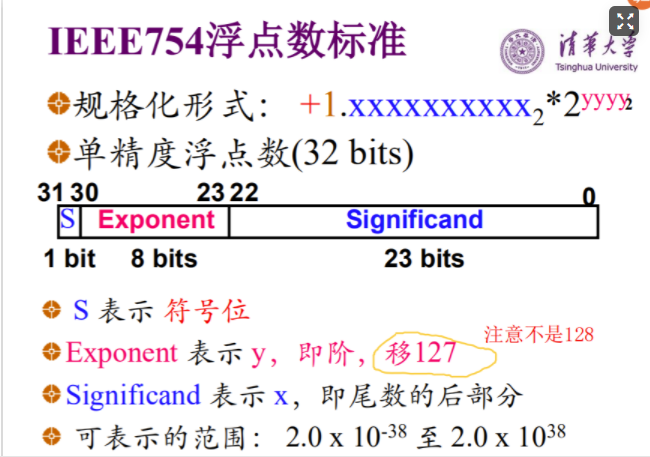
IEEE754浮点标准下表示：

|  |  |  |
| --- | --- | --- |
| 符号位 (1位) | 阶码（8位） | 尾数（23位） |
| 1 | 10000011 | 1011101+后跟16个0 |

1 10000011 1011101+16个0 = 0x**C1DD0000**

验算如下：





-2017 的 IEEE 单精度浮点表示\_\_。 (2018年)

-2017 = -1111110 0001

规格化：-1.111110 0001 × 2^10

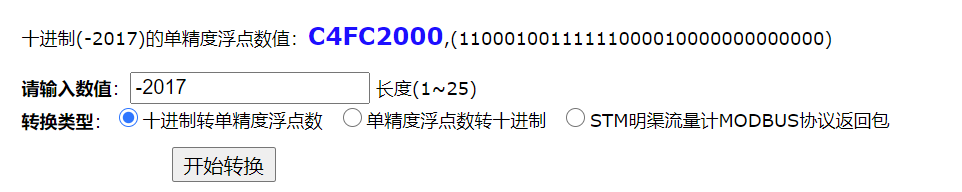
10 = 137 -127

符号位 = 1 ，阶码 = 137 = 10001001，尾数 = 111110 0001

IEEE754浮点标准下表示：

1 10001001 1111100001+后跟13个0 = 0x**C4FC2000**

验算如下：



|  |  |  |
| --- | --- | --- |
| 符号位 (1位) | 阶码（8位） | 尾数（23位） |
| 1 | 10001001 | 1111100001+后跟13个0 |

若海明码 **P1P2** D1 **P3** D2 D3 **P4** 为**01** 0 **1** 1 0 **0**，则该海明码有（）位错误（0 位，一位，二位），正确的 D1D2D3 为（）。(2017年)

**P1 P2** D1 **P3** D2 D3 **P4**

**0 1** 0 **1** 1 0 **0**

**--------------------------------------------------**

S1 = p1 ⊕ D2 ⊕ D1 = 1

S2 = p2 ⊕ D3 ⊕ D1 = 1

S3 = p1 ⊕ D3 ⊕ D2 = 0

S4 = p4 ⊕ p3 ⊕ p2 ⊕ p1 ⊕ D3 ⊕ D2 ⊕ D1 = 1

S4 S3 S2 S1 = 1011

S4 = 1，故有一位错，出错的为位S1和S2的交集即D1

将收到的海明码中的D1反转过来就可以了，原D1D2D2 = 010 -> 110

正确的D1D2D3应为 110



**选择**

IEEE规格化单精度浮点数能表示的最小正数是（） (2017年)

最小正数为

最大正数为



**控制器**

**判断(1×5')**

提高流水线的段数可提高cpu的频率，并提高单位时间内执行指令的速度 (2020年)

错误❌。

CPU频率，就是CPU的时钟频率，简单说是CPU运算时的工作的频率（1秒内发生的同步脉冲数）的简称

cpu的频率是由时钟的震荡快慢决定的，提高流水线段数不能提高cpu频率。

段数增加，可能意味着指令间冲突的概率也要增加，也会增加各段间暂缓数据时间，所以并不一定会提高单位时间内执行指令的速度。

CPI越短，程序执行能力越快 (2020年)

错误❌。

程序执行的时间由指令数量、时钟周期长度和每条指令的周期数目（CPI）等因素共同决定。单单比较CPI，无法判断程序执行的快慢。

CPI减少，执行相同程序的时间也减少 (2019年)

错误❌。

流水线的CPI就比多周期CPU的CPI小，但是相对来说流水线执行速度更快。

程序的运行时间不是仅仅靠CPI决定，而是由指令数量、时钟周期长度和每条指令的周期数目（CPI）等因素共同决定。

这种单单比较CPI，肯定错。

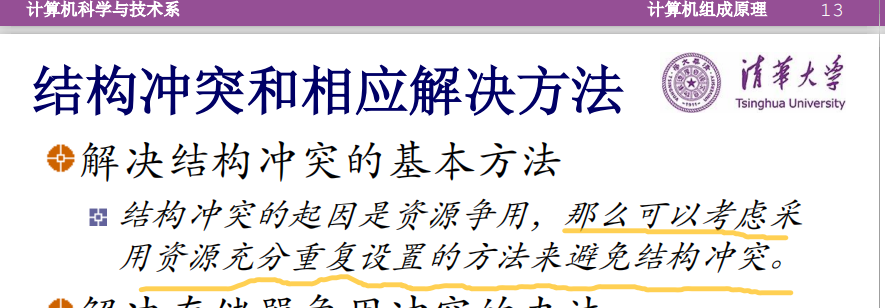
提高 cpu 主频可以加快程序执行速度 (2018年)

错误❌。

如上，只比较一个因素，肯定错。

MIPS五级流水线设计中，使用充分设置功能单元的方法可以改善结构冲突 (2019年)

正确✔。



指令由 和 组成.。 (2017年)

指令由操作码和操作数地址构成



**填空题(2+2+3+3=10)**

MIPS五级流水线中，解决数据冲突的方法，给出3个. (2019年)

答：主要是以下4种方法

使用旁路技术

静态调度 — 编译器调度方法

动态调度 — 指令顺序发射，乱序执行，指令乱序流出

暂停流水线

处理机 逻辑电路进行算术运算， 逻辑电路用于数据暂存， 逻辑电路用于分支选择。 (2018年)

组合、时序、组合

**选择(2\*5=10')**

下列表述中正确的是 (2020年)

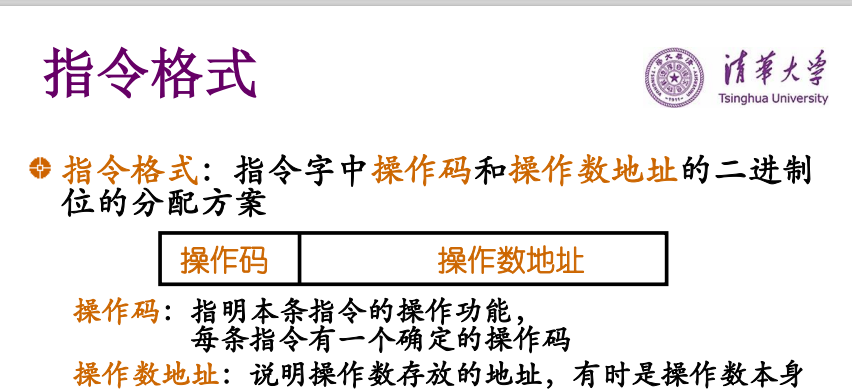
A. 指令必须给出指令操作码

B.指令必须给出指令操作数

C. 指令的长度必须一样长

D.我忘了

选A，不一定给出指令操作数，一般是给出操作数地址



解决流水线的数据冲突不可以采用的是 (2020年)

C.分支预测

D.静态调度

选 C，分支预测是用来解决控制冲突的，不是用来解决数据冲突的。



以下哪种不可以解决数据冲突 (2018年)

A.暂停流水线

B.分支预测

C.调整指令顺序

D.数据旁路

选 B，分支预测是用来解决控制冲突的，不是用来解决数据冲突的。

调整指令顺序是静态编译方法

MIPS五级流水中，有哪个数据冲突（ ） (2019年)

A.RAR B.RAW C.WAR D.WAW

选B，MIPS不会发生WAR和WAW冲突

RAR估计是乱凑的

以下关于五段流水线的处理机说法错误的是 (2018年)

A.多个处理器不会发生结构冲突

B.每个周期执行一个功能

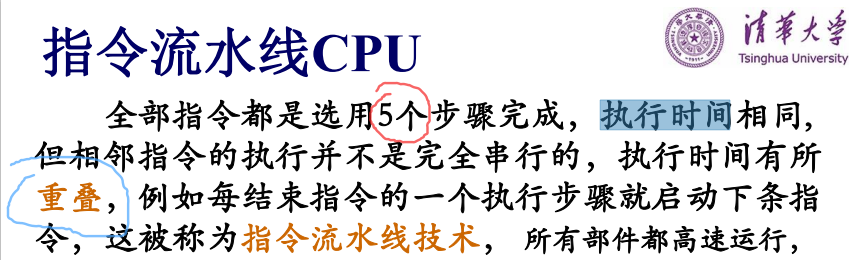
C.可以采用微程序或者硬连线设计

D.不同的指令执行时间相同

选A，A是错的。

结构冲突是因为资源冲突，而资源冲突是存储器这些。

多个处理器就是多条流水线，每一条流水线都可能产生结构冲突。



以下哪个不是响应异常的处理 (2018年)

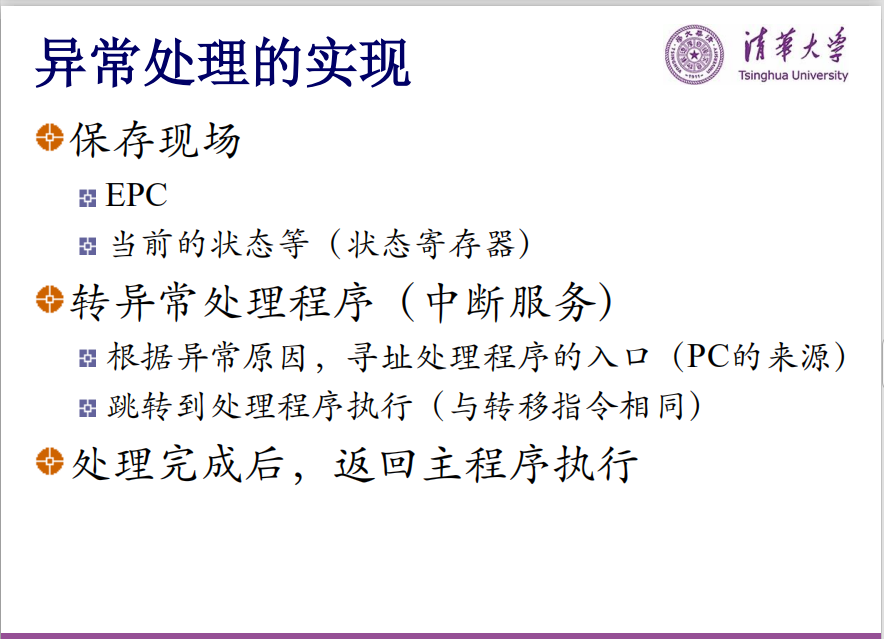
A.保存 pc

B.保存通用寄存器

C.保存异常原因

D.恢复 pc

选B ，不是保存通用寄存器，是保存状态寄存器，通用寄存器也不需要保存



计算机运行的最小单位（） (2017年)

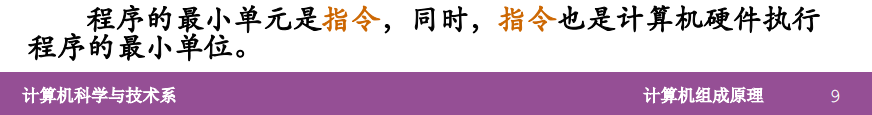
 a）？

b）微指令

c）指令

d）？

选C，指令是计算机运行的最小单位



微指令的概念：一条机器指令对应一个微程序，这个微程序是由若干条微指令构成的。一条机器指令的功能是若干条微指令组成的序列来实现的。简而言之，一条机器指令所完成的操作划分成若干条微指令来完成，由微指令进行解释和执行。

**解答** (2020年)

假设寄存器输入延迟为10ps,寄存器输出延迟为10ps,各阶段的延迟如下(其中ID段算入了通用寄存器的读取事件，其余未算入)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| IF | ID | EX | MEM | WB |
| 250ps | 180ps | 150ps | 300ps | 200ps |

实现以下3条指令：

addu rd,rs,rt

lw rt,rs,imm

j target

1. 按照单周期、多周期、流水线设计，最短的时钟周期为？请给出你的计算过程

2. 按照单周期、多周期、流水线设计，最短的指令延迟为？

解答：

addu rd,rs,rt

* R[rd] **←** R[rs] op R[rt]

lw rt,rs,imm

* Addr **←** R[rs] + SignExt(imm) 计算地址
* R[rt] **←** MEM[Addr] 读数据

j target

* PC[31：0] **←** PC[31：28] || traget[25:0]||00

1.单周期的时钟周期就是最长指令执行时间

其中LW指令执行时间最长

lw的执行时间为：取指250 + 译码180 + 执行150 + 访存300 + 写回200 =  1080ps

单周期的最短时钟周期为 1080ps。

多周期最短时钟周期是各阶段中执行时间最长的一段的时间

而各阶段中访存的时间最长，为：访存300 + 锁存器输入延迟10 + 锁存器输出延迟10 = 320ps

流水线也最短时钟周期也是各阶段中执行时间最长的一段的时间，为320ps

2.

对单周期而言，最短指令延迟是其时钟周期：1080ps

对多周期而言，J跳转指令只需2个周期，由第1问已知多周期的时钟周期为320ps，故其最短指令延迟为 2×320 = 640ps

* 多周期最短指令延迟为640ps。

对流水线而言，指令的执行时间都是相同的，等于 流水线段数 ✖ 时钟周期，故最短指令延迟为 5×320 = 1600ps

* 流水线最短指令延迟为 1600ps。

计算题(5) (2019年)

MIPS 处理器pc输入延迟2ns，寄存器堆输出延迟2ns，内存延迟 10ns，ALU 延迟 6ns，寄存器堆输入延迟1ns，流水线寄存器以及多周期锁存器输出延迟为 2ns，要有计算过程。

addu rs rt rd

subu rs rt rd

ori rs rt imm

lw rs rt imm

sw rs rt imm

beq rs rt imm

j target

1.按照单周期设计，指令内存与数据内存分开，计算指令延迟？

2.按照多周期设计，指令内存和数据内存在同⼀个内存模块，最长和最短的指令延迟分别是指哪条指令，分别计算对应的延迟，以及所需时钟周期数

3.按照五级流水线设计，指令内存和数据内存不在同⼀个内存模块，处理器能达到的最大主频是多少？

解答：1.单周期

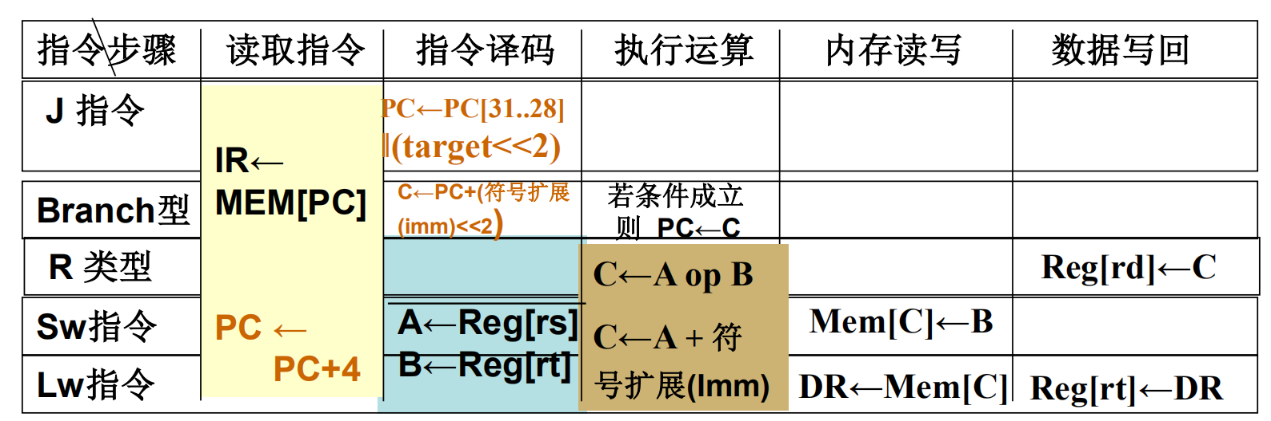
指令延迟是以最长指令为准的，LW指令时间最长，故单周期指令延迟为 31ns。

2.多周期，每段放入锁存器。

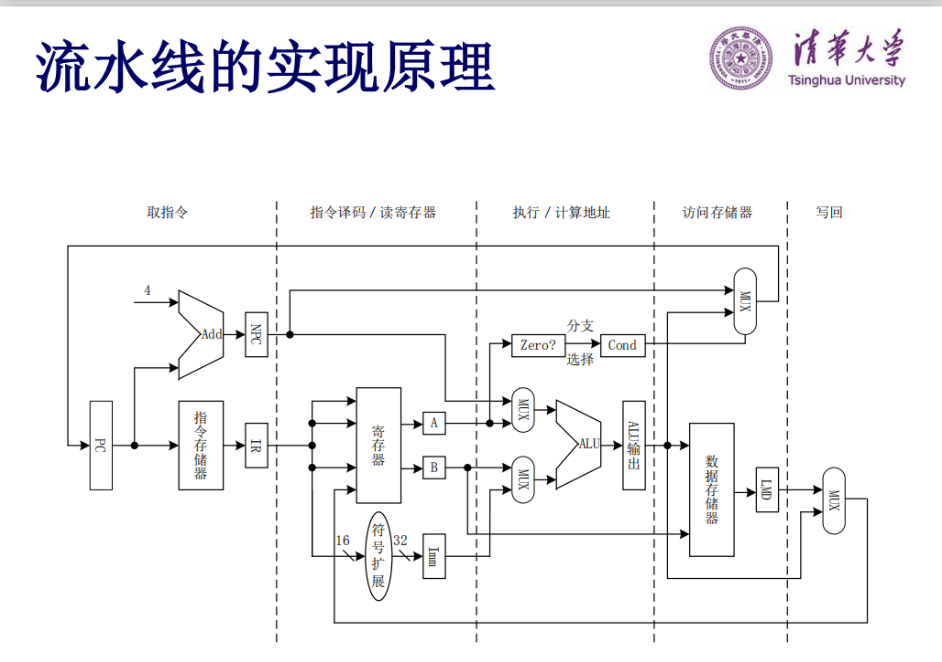
多周期的时钟周期是延迟最长的那段决定的，为MEM段，延迟为12ns

最长指令延迟是Lw指令，所需时钟周期数为5，故最长指令延迟为 12×5 = 60ns

最短指令延迟为 J指令，所需时钟周期数为2，故短长指令延迟为 12×2 = 24ns



3. 五级流水线的时钟周期为五段中最长的延迟即12ns



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **指令** | **IM取指** | **ID译码** | **EXE执行** | **MEM访存** | **WB写回** | **PC** | **总延迟** |
| addu rs rt rd | 读指令内存10 | 读寄存器2 | ALU6 |  | 写回寄存器1 | PC输入延迟2 | 21 |
| subu rs rt rd | 10 | 2 | 6 |  | 1 | 2 | 21 |
| ori rs rt imm | 10 | 2 | 6 |  | 1 | 2 | 21 |
| lw rs rt imm | 10 | 2 | 6 | 读数据内存10 | 1 | 2 | 31 |
| sw rs rt imm | 10 | 2 | 6 | 10 |  | 2 | 30 |
| beq rs rt imm | 10 | 2 | 6 |  |  | 2 | 20 |
| j target | 10 |  |  |  |  | 2 | 12 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **指令** | **IF** | **ID** | **EXE** | **MEM** | **WB** | **时钟周期数** |
| addu | 读指令内存10+锁存器2 | 读寄存器2+锁存器2 | ALU6+锁存器2 |  | 写回寄存器1 | 4 |
| subu | 10+2 | 2+2 | 6+2 |  | 1 | 4 |
| ori | 10+2 | 2+2 | 6+2 |  | 1 | 4 |
| **lw** | 10+2 | 2+2 | 6+2 | 读数据内存10+锁存器2 | 1 | 5 |
| sw | 10+2 | 2+2 | 6+2 | 10 |  | 4 |
| beq | 10+2 | 6+2 | ALU6 + (若满足条件写回)2 |  |  | 3 |
| **j** | 10+2 | 0+写回PC2 |  |  |  | 2 |

解答题（2018年)

五段流水线，每段 10ns，每个寄存器 5ns，执行下面一组指令，所需时间至少是多少？

LW R1 (0)R2

SUB R3 R1 R4

ADD R5 R1 R6

OR R7 R1 R6

ADD R9 R1 R10

解答：

时钟周期 = 延迟最长段的时间 = WB段延迟 = 10ns内存时间+5ns的锁存器 = 15ns

第一条和第二条指令之间有数据冲突，即使旁路后，也必须加一个气泡。

总的时钟周期个数 = 5 + 4 + 1 = 10，所需时间 = 10×15 = 150ns

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| LW指令1 | IF | ID | EX | ME | WB |  |  |  |  |  |
| SUB指令2 |  | IF | ID | 气泡 | EX | ME | WB |  |  |  |
| ADD指令3 |  |  | IF | 🎈 | ID | EX | EX | EX |  |  |
| OR指令4 |  |  |  | 🎈 | IF | ID | EX | EX | EX |  |
| ADD指令5 |  |  |  |  |  | IF | ID | EX | EX | EX |

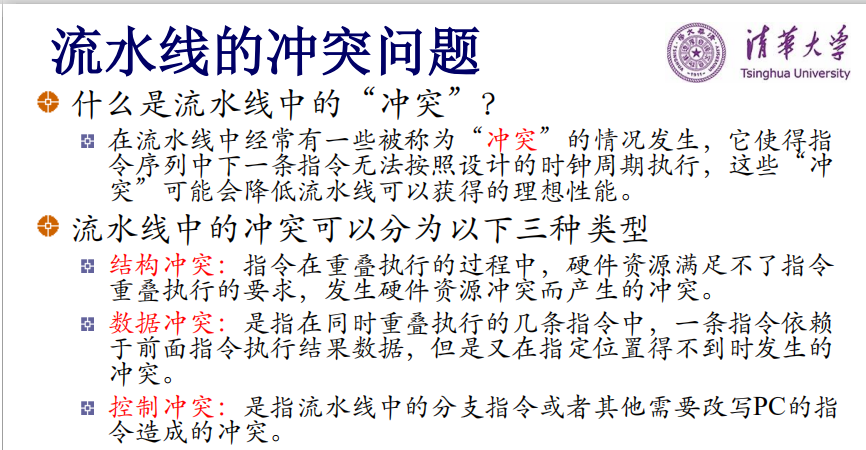
指令流水线可能发生的冲突分类，以及原因。 (2017年)

答：指令流水线可能发生的冲突分类三类：

结构冲突：指令在重叠执行过程中，需要用到相同的资源，但是硬件资源满足不了重叠执行的要求，从而产生所谓的结果冲突。

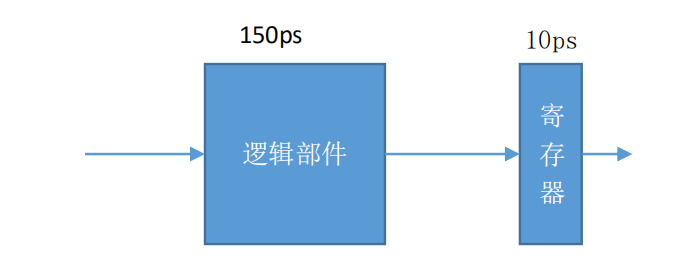
数据冲突：指令在重叠执行过程中，前面的指令还没有把结果计算出来，后面的指令就需要用了。即一条指令依赖于前面指令执行结果数据，但是又在自己该需要的时候得不到时发生的冲突。

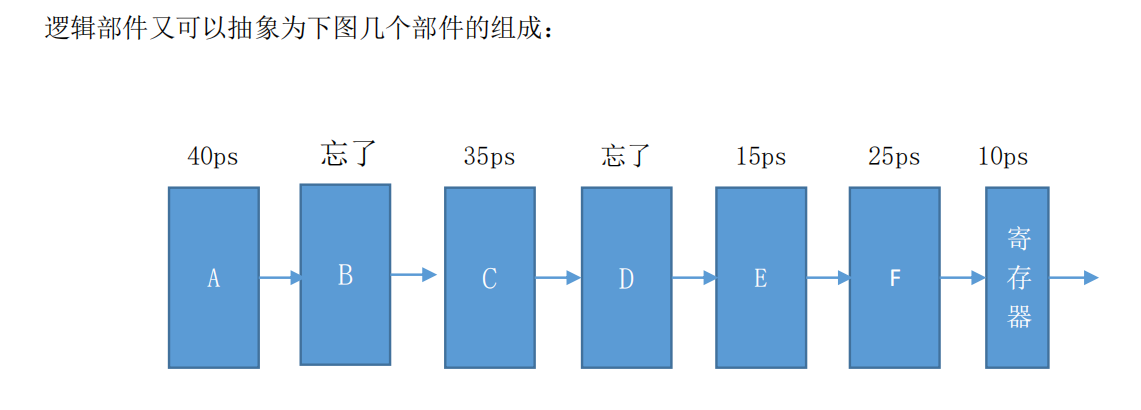
控制冲突：流水线的分支指令或者其他需要改写PC的指令造成的冲突。



**流水线大题**（2016年）

任何指令的执行部件可以抽象为一个逻辑组件和寄存器的结构，逻辑组件的延迟为 150ps，寄存器为 10ps，示意图如下





1） 要形成一个 4 级流水，应该将三个寄存器安插在那些位置？问该四级流水的延迟和最大吞吐率

答：因 40 + B + 35 + D +15 + 15 = 150ps，这里不妨假设 B = 20ps ，D = 15ps

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 40ps | 20ps | 35ps | 15ps | 15ps | 25ps | 10ps |
| **A** | **B** | **C** | **D** | **E** | **F** | **寄存器** |

4级流水线，各段延迟时间都应该比较平均，所以把3个流水线分别安插在A，B，D的后面

经过划分后，分为了 A | B | CD | EF 4段 ，对应的各段延迟为 50 | 30 | 60 | 50

四级流水线的时钟周期以最长的一段延迟时间计算，故时钟周期为：60ps

流水线延迟为：60×4 = 240ps

吞吐率：单位时间执行的指令条数

流水线的最大吞吐率为 ，其中 △t 为一段执行的时间。

1s = 10^12ps

最大吞吐率 1/ (60\*10^-12) = 16.7×10^9 条/s = 16.7GIPS

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 40ps | 10ps | 20ps | 10ps | 35ps | 15ps | 10ps | 15ps | 25ps | 10ps |
| **A** | **寄存器** | **B** | **寄存器** | **C** | **D** | **寄存器** | **E** | **F** | **寄存器** |

2） 为达到最大的吞吐率应该设计成几级流水？寄存器应该安插在哪些位置？问该流水的延迟和最大吞吐率

答：若增加流水级，时钟周期下降，则表明存在更优解；否则，只会增加指令延迟，成为无效改进。

因而最优流水级取决于最大的部件延迟。

可以看到，A的延迟最高，成为性能瓶颈。

以A为准，可划分为五级流水（A|B|C|DE**|**F或A|B|C|D**|**EF）。

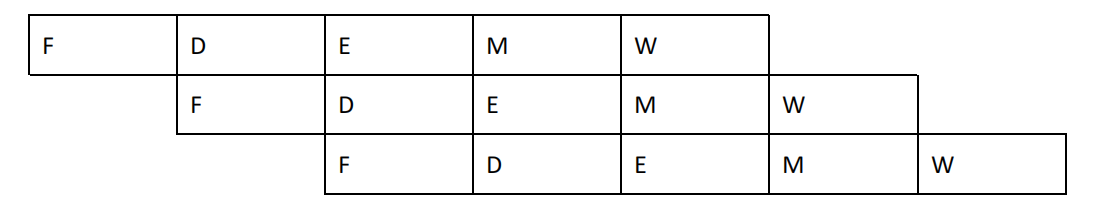
时钟周期为50ps

流水线延迟为：50×5 = 250ps

最大吞吐率 1/ (50\*10^-12) = 20×10^9 条/s = 20GIPS

参考：《深入理解计算机》-练习题-4.28，与此题类似。

如果将上面的部件形成五级流水，分为取指（F），分析（D），执行（E），访存（M），写回（W）五个阶段，每个阶段占一个时钟周期，%edx, %edy %edz %edv 为寄存器



3） 以下三个指令按指令流水进行，为了获得最大吞吐率应进行哪些操作？三条指令一共用了多少个时钟周期？（每条指令所需要的上一条结果的数据都要等到上一条运算的结果才能进行）

MOV 100, %edx

MOV 200, %edy

ADD %edx, %edy(具体最后一条实在想不起来)

存储器

解答：会产生数据冲突，为了获得最大的吞吐率应该使用旁路技术，将mov指令在ALU算出来%edx和%edy的值，直接给到ADD指令ALU的两个操作数。使用旁路技术解决冲突后，三条指令所用时间为 5 + 1 + 1 = 7个时钟周期

4） 以下四个指令按指令流水进行，为了获得最大吞吐率应进行哪些操作？四条指令一共用了多少个时钟周期？（每条指令所需要的上一条结果的数据都要等到上一条运算的结果才能进行）（程序都想不起来了囧，就记得前两条是把数字移入寄存器，第三条的执行需要前两条数据，最后一条需要第三条的数据）

解答：根据题目描述，存在数据冲突。采用旁路技术解决之。本题可能存在**必须进行暂停的数据冲突，此时需要暂停一个周期。**

故四条指令共用时钟周期 = 5 + 1 + 1 +1 + 1(暂停了一个周期) = 9个时钟周期

**存储器**

**判断(1\*5')**

缓存原理利用了程序的局部性 (2020年)

正确 ✔

缓存缺失的类型包括，写3个 (2019年)

答：必然缺失、容量缺失、冲突缺失、无效缺失



选择：全相联，4路组相联，2路组相联中缓存命中率最高的是 (2020年)

答：全相联命中率最高，但是成本也最高。

命中率从高到底分别是 ：全相连=n路组相联 > 4 路组相连 > 2 路组相连 > 直接映射



高速缓存器的几种映射方式\_\_、\_\_、\_\_。 (2018年)

答：全相连映射、直接映射、组相连映射

**填空** 给出一个指令序列，cache容量是1024B，循环100次，每次访问3和3+1024地址的内容，计算直接映射和二路组相连的缓存命中率 (2020年)

题目不全

以下说法正确的是 (2018年)

A.缓存越大程序执行速度越快

B.TLB 也是一种缓存数据和指令的缓存器

C. 指令和数据采用不同的缓存可以提高流水线速度

D.

选C

A错，A太绝对了，当缓存增加到一定程度，缓存容量再增加速度提升越来越小，但是用于映射的延迟会越来越大，因此整体速度反而下降。

B错，TLB也称为快表，相当于页表的Cache，其中存储了当前最可能被访问到的页表项，其内容是部分页表项的一个副本，作用是增加虚拟地址到物理地址的转换效率，TLB 缺失后仍然可以通过查询页表获得虚拟地址对应的物理地址。

**解答题** (2017年)

30 位虚拟地址，28 位物理地址，一级页表，页大小 16KB，访问 5ns；Cache 采用直接映射，大小64KB，块大小 4B，访问 5ns；主存访问 40ns。

1）虚拟页表脏(dirt)位 1 位，有效位 1 位，问页表大小

2）cache 标记位，索引位，块内地址各多少位

3）一次 cache 命中访问时间，cache 失效访问时间，命中率为 90%平均访问时间

4）系统进程切换时以下操作是否需要，原因

a）清除 cache 有效位

b）将已经调入页表清空

5）注意到页表访问和 cache 访问时间相同，可否通过修改 cache 映射方式，使 cache 和页表一同访问，可以的话做出相应设计，并计算 cache90%命中率的时候的平均访问时间。

解答：

1）

页大小为16KB，占14位

页表项数 = 总页数 = 2^30/2^14 = 2^16 = 64K

页表项位数 = 14位页大小 + 1位脏位 + 1位有效位 = 16位 = 2B

页表大小 = 页表项大小 × 页表项数目 = 2B × 64K= 128KB

2）

cache总块数 = 64KB/4B = 16K = 2^14，故索引位数为：14位，

块大小4B = 2^2B，故块内地址为：2位，

标记位数为：28 - 14 -2 = 12位

3）

这里把虚拟地址转为物理地址访问页表简称为访问页表

一次 cache 命中访问时间：访问页表(5ns) + 访问cache(5ns) = 10ns

cache 失效访问时间：访问页表(5ns) + 访问cache(5ns) + 访问主存(40ns)= 50ns

命中率为 90%平均访问时间：10×0.9 + 50 × 0.1 = 14ns

4）

a）清除 cache 有效位：不需要，cache在物理上对应

b）将已经调入页表清空：需要，每一个进程都有自己的页表（地址映射关系）

5）

将Cache由原来的和物理地址映射改为和虚拟地址进行映射，这样可以直接通过虚拟地址访问cache里的缓存数据。

Cache设计如下：

   cache总块数 = 64KB/4B = 16K = 2^14，故索引位数为：14位，

块大小4B = 2^2B，故块内地址为：2位，

标记位数为：30(虚拟地址位数) - 14 -2 = 14位

* 改进后一次cache命中访问时间：= 访问Cache(5ns) =5ns
* 改进后cache失效访问时间：= 访问Cache(5ns) + 主存(40ns) =45ns
* cache90%命中率的时候的平均访问时间：= 90%命中 + 10%失效= 9ns

对于传统机械硬盘，读100MB数据，顺序读取时间小于随机读取时间 (2019年)

正确 ✔

涉及到寻道延迟（机械运动），故顺序读取比较高效。

RAID6 坏两个磁盘也可以工作 (2018年)

正确 ✔

RAID6每个条带有两个冗余块，可以允许两个磁盘错误。

下列哪一项没有容错能力 (2019年)

A. RAID0         B. RAID1         C. RAID5             D. RAID6

选A

RAID0没有容错能力，只是单纯的并行访问。

**填空：**100MB的数据，RAID1为,RAID5为 (2020年)

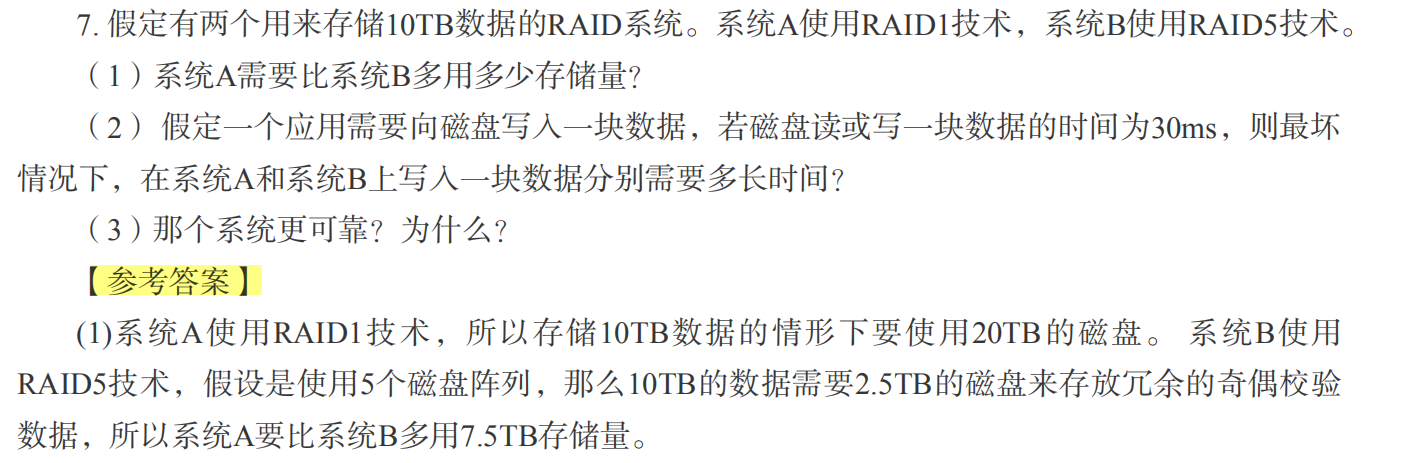
答：

使用RAID1在100MB数据的情形下，需使用200MB的磁盘。

使用RAID5技术，假设是5个磁盘阵列，100MB需要100/4 = 25MB存放冗余的奇偶校验数据。

所需存储量：25 +100=125MB。

下面来自【THU 《计算机组成原理》复习题参考解答】



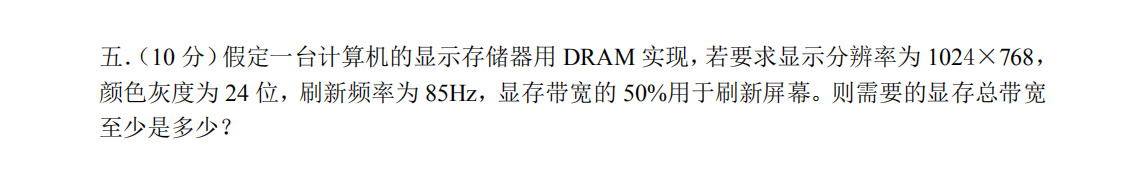
**填空(5\*1')**

一台计算机显示器的分辨率为800\*60，使用RGB颜色，每个颜色使用1个字节表达，帧率为50HZ, 显示器的总带宽的80%用于刷新屏幕，则需要的显存带宽至少为\_\_\_\_\_\_(2020年)

答：

800 \* 60 \* 3 \* 8 \* 50 /0.8 = 72Mb/s = 9MB/s

类似的题来自 【THU 计算机组成原理-2010-期末】

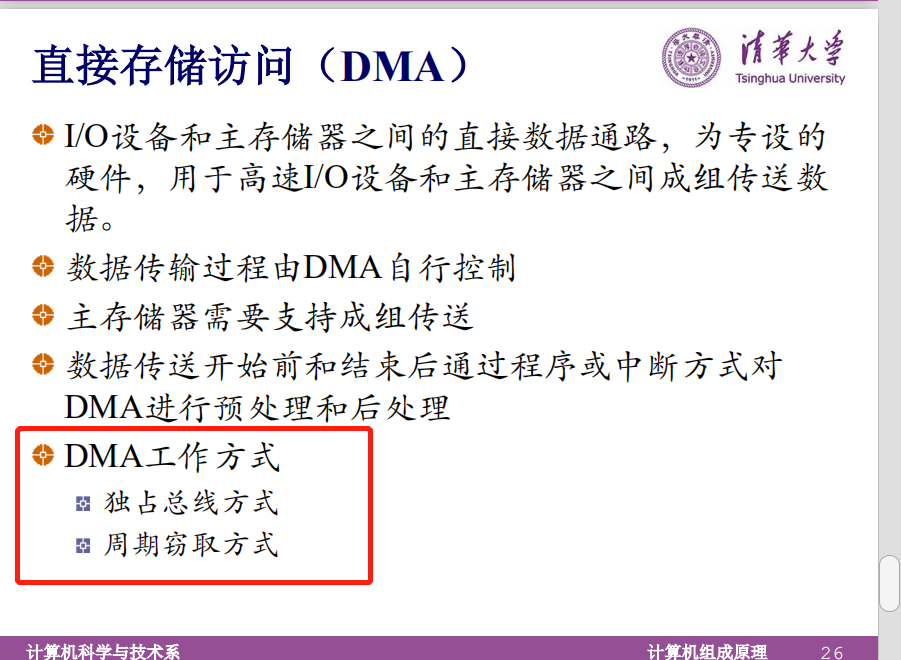


参考答案：



DMA使用总线的方式为（ ）和（ ）。 (2017年)

答：独占总线方式、周期窃取方式



**选择：**MIPS中断中不是由硬件负责的是（ ） (2020年)

A. 开中断

B.保存通用寄存器

C.保存异常原因

D.关中断

选B

下列关于静态存储器和动态存储器的描述正确的是（ ） (2019年)

A.静态存储器使用触发器，需要定期刷新

B.静态存储器使用电容，不需要定期刷新

C.动态存储器使用触发器，不需要定期刷新

D.动态存储器使用电容，需要定期刷新

选 D

正确说法：

静态存储器使用触发器，不需要定期刷新

动态存储器使用电容，需要定期刷新

下列哪个是对的 (2019年)

A．虚拟内存空间比实际的地址空间大

B．虚拟内存空间比实际的地址空间小

C．虚拟内存空间连续存放，实际内存一定连续存放

D．虚拟内存空间不连续存放，实际内存有可能连续存放

A 不一定，有可能一样大。

选D

虚拟内存空间不连续存放，只是在物理的非连续内存分配（段、页机制）的基础上增加调换功能。



**输入输出**

下面总线说法哪个正确（ ） (2019年)

A.并行总线速度大于串行

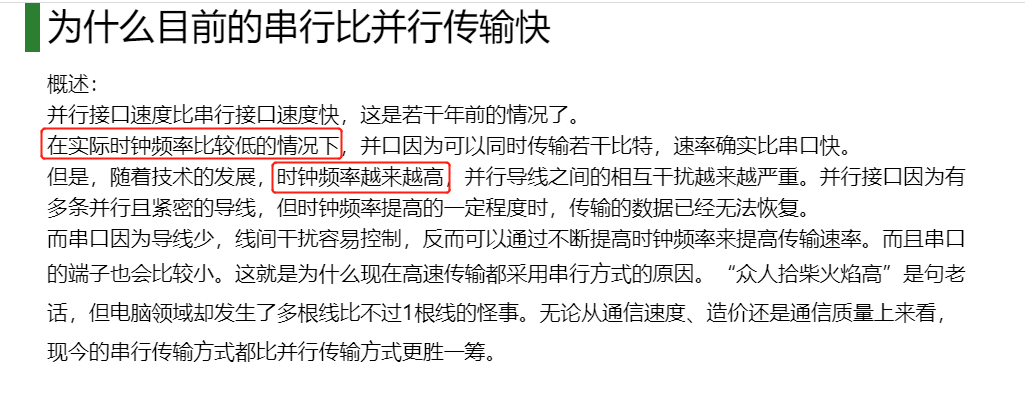
B.异步总线速度大于同步

C.单总线速度大于双总线

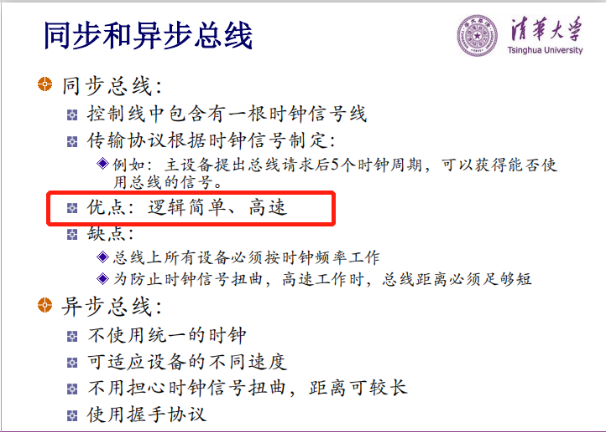
D.以上说法均错误

选D

A应该错，应该说“在相同频率下并行总线速度大于串行”更准确。



B错误，同步总线速度大于异步



C错误，双总线速度大于单总线

