1.7 将计算机系统中某一功能的处理速度加快10倍，但该功能的处理时间仅为整个系统运行时间的40%，则采用此增强功能方法后，能使整个系统的性能提高多少？

**解** 由题可知： 可改进比例 = 40% = 0.4 部件加速比 = 10

根据Amdahl定律可知：



采用此增强功能方法后，能使整个系统的性能提高到原来的1.5625倍。

1.8 计算机系统中有三个部件可以改进，这三个部件的部件加速比为：

部件加速比1=30； 部件加速比2=20； 部件加速比3=10

1. 如果部件1和部件2的可改进比例均为30%，那么当部件3的可改进比例为多少时，系统加速比才可以达到10？
2. 如果三个部件的可改进比例分别为30%、30%和20%，三个部件同时改进，那么系统中不可加速部分的执行时间在总执行时间中占的比例是多少？

解：（1）在多个部件可改进情况下，Amdahl定理的扩展：



已知S1＝30，S2＝20，S3＝10，Sn＝10，F1＝0.3，F2＝0.3，得：



得F3＝0.36，即部件3的可改进比例为36%。

（2）设系统改进前的执行时间为T，则3个部件改进前的执行时间为：（0.3+0.3+0.2）T = 0.8T，不可改进部分的执行时间为0.2T。

已知3个部件改进后的加速比分别为S1＝30，S2＝20，S3＝10，因此3个部件改进后的执行时间为：



改进后整个系统的执行时间为：Tn = 0.045T+0.2T = 0.245T

那么系统中不可改进部分的执行时间在总执行时间中占的比例是：



3.4 设一条指令的执行过程分成取指令、分析指令和执行指令三个阶段，每个阶段所需的时间分别为△t、△t和2△t 。分别求出下列各种情况下，连续执行N条指令所需的时间。

（1）顺序执行方式；

（2）只有“取指令”与“执行指令”重叠；

（3）“取指令”、“分析指令”与“执行指令”重叠。

解：（1）每条指令的执行时间为：△t＋△t＋2△t＝4△t

连续执行N条指令所需的时间为：4N△t

（2）连续执行N条指令所需的时间为：4△t＋3（N-1）△t＝（3N＋1）△t

（3）连续执行N条指令所需的时间为：4△t＋2（N-1）△t＝（2N＋2）△t

4.2 简述Tomasulo算法的基本思想。

答：核心思想是：① 记录和检测指令相关，操作数一旦就绪就立即执行，把发生RAW冲突的可能性减小到最少；② 通过寄存器换名来消除WAR冲突和WAW冲突。寄存器换名是通过保留站来实现，它保存等待流出和正在流出指令所需要的操作数。

基本思想：只要操作数有效，就将其取到保留站，避免指令流出时才到寄存器中取数据，这就使得即将执行的指令从相应的保留站中取得操作数，而不是从寄存器中。指令的执行结果也是直接送到等待数据的其它保留站中去。因而，对于连续的寄存器写，只有最后一个才真正更新寄存器中的内容。一条指令流出时，存放操作数的寄存器名被换成为对应于该寄存器保留站的名称（编号）。

* Consider a program with the given characteristics
  + Instruction count (I-Count) = 106 instructions
  + 30% of instructions are loads and stores
  + D-cache miss rate is 5% and I-cache miss rate is 1%
  + Miss penalty is 100 clock cycles for instruction and data caches
  + Compute combined misses per instruction and memory stall cycles
* Combined misses per instruction in I-Cache and D-Cache
  + 1% + 30% × 5% = 0.025 combined misses per instruction
  + Equal to 25 misses per 1000 instructions
* Memory stall cycles
  + 0.025 × 100 (miss penalty) = 2.5 stall cycles per instruction
  + Total memory stall cycles = 106 × 2.5 = 2,500,000

**CPI with Memory Stalls**

* A processor has CPI of 1.5 without any memory stalls
  + Cache miss rate is 2% for instruction and 5% for data
  + 20% of instructions are loads and stores
  + Cache miss penalty is 100 clock cycles for I-cache and D-cache
* What is the impact on the CPI?
* Answer:

Mem Stalls per Instruction =0.02×100 + 0.2×0.05×100 = 3

CPIMemoryStalls =1.5 + 3 = 4.5 cycles per instruction

CPIMemoryStalls / CPIPerfectCache =4.5 / 1.5 = 3

Processor is 3 times slower due to memory stall cycles

CPINoCache =1.5 + (1 + 0.2) × 100 = 121.5 (a lot worse)

3.13有一个流水线由4段组成，其中每当流经第3段时，总要在该段循环一次，然后才能流到第4段。如果每段经过一次所需要的时间都是，问：

1. 当在流水线的输入端连续地每时间输入任务时，该流水线会发生什么情况？
2. 此流水线的最大吞吐率为多少？如果每输入一个任务，连续处理10个任务时的实际吞吐率和效率是多少？
3. 当每段时间不变时，如何提高该流水线的吞吐率？仍连续处理10个任务时，其吞吐率提高多少？

解：（1）会发生流水线阻塞情况。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 第1个任务 | S1 | S2 | S3 | S3 | S4 |  |  |  |  |  |  |
| 第2个任务 |  | S1 | S2 | stall | S3 | S3 | S4 |  |  |  |  |
| 第3个任务 |  |  | S1 | stall | S2 | stall | S3 | S3 | S4 |  |  |
| 第4个任务 |  |  |  |  | S1 | stall | S2 | stall | S3 | S3 | S4 |

（2）





（3）重复设置部件





吞吐率提高倍数＝＝1.64

3.17 假设各种分支指令数占所有指令数的百分比如下：

|  |  |
| --- | --- |
| 条件分支 | 20%（其中的60%是分支成功的） |
| 跳转和调用 | 5% |

现有一条段数为4的流水线，无条件分支在第二个时钟周期结束时就被解析出来，而条件分支要到第三个时钟周期结束时才能够被解析出来。第一个流水段是完全独立于指令类型的，即所有类型的指令都必须经过第一个流水段的处理。请问在没有任何控制相关的情况下，该流水线相对于存在上述控制相关情况下的加速比是多少？

解：没有控制相关时流水线的平均CPI＝1

存在控制相关时：由于无条件分支在第二个时钟周期结束时就被解析出来，而条件分支

要到第3个时钟周期结束时才能被解析出来。所以：

（1）若使用排空流水线的策略，则对于条件分支，有两个额外的stall，对无条件分支，有一个额外的stall：

CPI = 1+20%\*2+5%\*1 = 1.45

加速比S=CPI/1 = 1.45

（2） 若使用预测分支成功策略，则对于不成功的条件分支，有两个额外的stall，对无条件分支和成功的条件分支，有一个额外的stall 1：

CPI = 1+20%\*(60%\*1+40%\*2) +5%\*1 = 1.33

加速比S=CPI/1 = 1.33

（3）若使用预测分支失败策略，则对于成功的条件分支，有两个额外的stall；对无条件分支，有一个额外的stall；对不成功的条件分支，其目标地址已经由PC 值给出，不必等待，所以无延迟：

CPI = 1+20%\*(60%\*2 + 40%\*0) +5%\*1 = 1.29

加速比S=CPI/1 = 1.29

4.4 假设有一条长流水线，仅仅对条件转移指令使用分支目标缓冲。假设分支预测错误的开销为4个时钟周期，缓冲不命中的开销为3个时钟周期。假设：命中率为90%，预测精度为90%，分支频率为15%，没有分支的基本CPI为1。

1. 求程序执行的CPI。
2. 相对于采用固定的2个时钟周期延迟的分支处理，哪种方法程序执行速度更快？

解：（1）程序执行的CPI = 没有分支的基本CPI（1） + 分支带来的额外开销

分支带来的额外开销是指在分支指令中，缓冲命中但预测错误带来的开销与缓冲没有命中带来的开销之和。

分支带来的额外开销= 15% \* (90%命中×10%预测错误×4 + 10％没命中×3)= 0.099

所以，程序执行的CPI ＝ 1 ＋ 0.099 = 1.099

（2）采用固定的2 个时钟周期延迟的分支处理CPI = 1 + 15%×2 = 1.3

由（1）（2）可知分支目标缓冲方法执行速度快。

4.9 设指令流水线由取指令、分析指令和执行指令3个部件构成，每个部件经过的时间为△t，连续流入12条指令。分别画出标量流水处理机以及ILP均为4的超标量处理机、超长指令字处理机、超流水处理机的时空图，并分别计算它们相对于标量流水处理机的加速比。

解：标量流水处理机的时空图：



执行完12条指令需T1＝14△t。

超标量流水处理机与超长指令字处理机的时空图：



超标量流水处理机中，每一个时钟周期同时启动4条指令。执行完12条指令需T2＝5△t，相对于标量流水处理机的加速比为：



超长指令字处理机中，每4条指令组成一条长指令，共形成3条长指令。执行完12条指令需T3＝5△t，相对于标量流水处理机的加速比为：



超流水处理机的时空图：



超流水处理机中，每1/4个时钟周期启动一条指令。执行完12条指令需T4＝5.75△t，相对于标量流水处理机的加速比为：



 4、（10分）假定我们有一台计算机，如果所有的cache访问都命中的话，它的CPI是2.0。唯一的数据访问指令是store和load，它们占指令总数的40％，不命中损失是25个时钟周期，不命中率是2％。如果所有的指令访问cache都命中的话，那么机器的速度是存在cache不命中时的多少倍？

首先计算所有cache访问都命中时计算机的性能：  
    CPU执行时间＝（CPU时钟周期＋内存停机周期）×时钟周期时长  
           　　＝（IC×CPI＋0）×时钟周期时长  
           　　＝ IC×2.0×时钟周期时长  
    现在计算考虑cache不命中在内的真实计算机性能，我们先计算内存停机周期：  
    内存停机周期＝IC×每条指令访问内存的次数×不命中率×不命中损失  
           　　＝IC×（1＋0.4）×0.02×25  
           　　＝IC×0.7  
    其中（1＋0.4）代表每条指令访问一次内存,而占指令总数40%的store和load访问两次内存,所以平均每条指令访问访问(1+0.4)次内存。这样总的性能是：  
    CPU执行时间=（IC×2.0＋IC×0.7）×时钟周期时长  
           　　= IC×2.7×时钟周期时长  
    性能提高的比是执行时间之比的倒数：  
    cache不命中考虑在内的CPU执行时间/cache访问全部命中的CPU执行时间为：  
    2.7×IC×时钟周期时长/2.0×IC×时钟周期时长＝1.35  
    cache访问全部命中时的速度是有cache不命中时机器速度的1.35倍。

   5、（10分）假设某台机器访问存储器都是cache命中，那么它的CPI等于2。还假设只有Load和Store指令才能访问存储器数据，这两种指令的数目占整个程序的40%。如果访问存储器时出现cache缺失，则一次缺失需要花费25个时钟周期。问这台机器在所有指令都cache命中情况比有2%缺失情况快几倍？

根据题意，在程序的执行过程中平均每条指令需要一次取指令和0.4次访问数据。因此，在有2% cache缺失时，由于cache缺失带来的额外开销为：

s_02

    这台机器在所有指令都cache命中时，CPU执行时间为：

s_03

    而在有2%缺失情况时，CPU执行时间为：

s_04

    所以这台机器在所有指令都cache命中情况比有2%缺失情况快  
　　 s_05

1、（12分）在一台单流水线处理机上执行下面的程序。每条指令都要经过“取指令”、“译码”、“执行”和“写结果”4个流水段，每个流水段的延迟时间都是5ns。执行部件的输出端有直接数据通路与它的输入端相连接，执行部件产生的条件码也直接送入控制器。

K1： MOVE R1, #4 ；R1←向量长度4

K2：LOOP：MOVE R2, A(R1) ；R2←A向量的一个元素

K3： ADD R0, R2 ；R0←(R0)＋(R2)

K4： DNE R1, LOOP ；R1←(R1)－1,若(R1)≠0转向LOOP

K5： MOVE SUN，R0 ；SUN←(R0)，保存结果

(1)列出指令之间的所有数据相关，包括读写、写读和写写数据相关。

(2)采用预测转移不成功的静态分支预测技术，画出指令流水线的时空图(可用指令序号表示)，并计算流水线的吞吐率、加速比和效率。

(3)采用预测转移成功的静态分支预测技术，计算指令流水线的吞吐率、加速比和效率。

**1、解:**

(1)指令K1与指令K2之间关于R1的写读数据相关（第1次循环）

指令K1与指令K4之间关于R1的写读数据相关（第1次循环）

指令K1与指令K4之间关于R1的写写数据相关（第1次循环）

指令K2与指令K3之间关于R2的写读数据相关（每次循环内）

指令K2与指令K4之间关于R1的读写数据相关（第次循环内）

指令K3与指令K5之间关于R0的写读数据相关（最后一次循环）

另外，相邻循环体之间的数据相关还有：

指令K4与下一循环的指令K2之间关于R1的写读数据相关

指令K3与下一循环的指令K2之间关于R2的读写数据相关

指令K2与下一循环的指令K2之间关于R2的写写数据相关

指令K3与下一循环的指令K3之间关于R0的写读数据相关

指令K3与下一循环的指令K3之间关于R0的读写数据相关

指令K3与下一循环的指令K3之间关于R0的写写数据相关

指令K4与下一循环的指令K4之间关于R1的写读数据相关

指令K4与下一循环的指令K4之间关于R1的读写数据相关

指令K4与下一循环的指令K4之间关于R1的写写数据相关

(2)采用预测转移不成功的静态分支预测技术

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** |
| 写结果 |  |  |  | **K1** | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** | **K5** |
| 执行 |  |  | **K1** | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** | **K5** |  |
| 译码 |  | **K1** | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** | **K5** |  |  |
| 取指令 | **K1** | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** |  |  | **K2** | **K3** | **K4** | **K5** |  |  |  |



(3)采用预测转移成功的静态分支预测技术

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** |
| 写结果 |  |  |  | **K1** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** |  |  | **K5** |
| 执行 |  |  | **K1** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** |  |  | **K5** |  |
| 译码 |  | **K1** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** |  |  | **K5** |  |  |
| 取指令 | **K1** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** | **K2** | **K3** | **K4** |  |  | **K5** |  |  |  |



4、（12分）某RISC处理机的工作主频为500MHz，有一个Cache和一个主存储器，Cache的存取周期为2ns，主存储器的存取周期为20ns，Cache的命中率为99%，有20%的LOAD/STORE指令，并假设处理机速度的瓶颈完全在存储系统。

(1)求理想情况(Cache的命中率为100%)下的CPI。

(2)计算该RISC处理机的实际MIPS速率。

(3)如果处理机的工作主频提高到1GHz，计算实际CPI和MIPS速率。

4、**解答：**

(1)

CPI=(2ns+2ns×20%)×500MHz=1.2

(2)

MIPS=1/(2ns×99%+20ns×1%+2ns×20%×99%+20ns×20%×1%)

=1/2.616ns=382.3

(3)

CPI=(2ns×99%+20ns×1%+2ns×20%×99%+20ns×20%×1%)×1GHz

=2.616ns×1GHz=2.616

MIPS=1/2.616ns=382.3

5.10 假设对指令Cache的访问占全部访问的75%；而对数据Cache的访问占全部访问的25%。Cache的命中时间为1个时钟周期，失效开销为50 个时钟周期，在混合Cache中一次load或store操作访问Cache的命中时间都要增加一个时钟周期，32KB的指令Cache的失效率为0.39%，32KB的数据Cache的失效率为4.82%，64KB的混合Cache的失效率为1.35%。又假设采用写直达策略，且有一个写缓冲器，并且忽略写缓冲器引起的等待。试问指令Cache和数据Cache容量均为32KB的分离Cache和容量为64KB的混合Cache相比，哪种Cache的失效率更低？两种情况下平均访存时间各是多少？

解：（1）根据题意，约75%的访存为取指令。

因此，分离Cache的总体失效率为：（75%×0.15%）＋（25%×3.77%）＝1.055%；

容量为128KB的混合Cache的失效率略低一些，只有0.95%。

（2）平均访存时间公式可以分为指令访问和数据访问两部分：

平均访存时间＝指令所占的百分比×（读命中时间＋读失效率×失效开销）＋ 数据所占的百分比×（数据命中时间＋数据失效率×失效开销）

所以，两种结构的平均访存时间分别为：

分离Cache的平均访存时间＝75%×（1＋0.15%×50）＋25%×（1＋3.77%×50）

＝（75%×1.075）＋（25%×2.885）＝1.5275

混合Cache的平均访存时间＝75%×（1＋0.95%×50）＋25%×（1＋1＋0.95%×50）

＝（75%×1.475）＋（25%×2.475）＝1.725

因此，尽管分离Cache的实际失效率比混合Cache的高，但其平均访存时间反而较低。分离Cache提供了两个端口，消除了结构相关。

5.11 给定以下的假设，试计算直接映象Cache和两路组相联Cache的平均访问时间以及CPU的性能。由计算结果能得出什么结论？

1. 理想Cache情况下的CPI为2.0，时钟周期为2ns，平均每条指令访存1.2次；
2. 两者Cache容量均为64KB，块大小都是32字节；
3. 组相联Cache中的多路选择器使CPU的时钟周期增加了10％；
4. 这两种Cache的失效开销都是80ns；
5. 命中时间为1个时钟周期；
6. 64KB直接映象Cache的失效率为1.4％，64KB两路组相联Cache的失效率为1.0％。

解： 平均访问时间＝命中时间＋失效率×失效开销

平均访问时间1-路=2.0+1.4% \*80=3.12ns

平均访问时间2-路=2.0\*(1+10%)+1.0% \*80=3.0ns

两路组相联的平均访问时间比较低

CPUtime=（CPU执行+存储等待周期）\*时钟周期

CPU time=IC（CPI执行+总失效次数/指令总数\*失效开销） \*时钟周期

=IC（（CPI执行\*时钟周期）+（每条指令的访存次数\*失效率\*失效开销\*时钟周期））

CPU time 1-way=IC(2.0\*2+1.2\*0.014\*80)＝5.344IC

CPU time 2-way=IC(2.2\*2+1.2\*0.01\*80)＝5.36IC

相对性能比：5.36/5.344=1.003

直接映象cache的访问速度比两路组相联cache要快1.04倍，而两路组相联Cache的平均性能比直接映象cache要高1.003倍。因此这里选择两路组相联。

二、（10分）判断题：

1、对计算机系统中经常使用的基本单元功能，宜于用软件来实现，这样可降低系统的成本。( **F** )

2、由于RISC简化了指令系统，因此，RISC上的目标程序比CISC上的目标程序要短一些，程序执行的时间就会少一些。（**F** 　）

3、流水线调度是看如何调度各任务进入流水线的时间，使单功能线性流水线有高的吞吐率和效率。（ **T**　）

4、无论采用什么方法，只要消除流水线的瓶颈段，就能提高流水线的吞吐率和效率。( **F**　)

5、在满足Cache与主存的一致性方面，写回比写直达法好。（ **F**　）

6、在多处理机上，各个任务的执行时间不同时，在个处理机总的运行时间均衡的前提下，取不均匀分配，让各处理机所分配的任务数要么尽量的多，要么尽量的少，这样，才可使总的运行时间减少。（　**F**　）

7、Cache组相联映象的块冲突概率比直接映象的高。（ **F** ）

8、要使线性流水线的实际吞吐率接近于理想的最大吞吐率，应将子过程数分得越多越好。（**F**）

9、在系列机内可以将单总线改为双总线，以减少公共总线的使用冲突。(  **F** )

1. （×）由于流水线的最大加速比等于流水线深度，所以增加流水段数总可以增大流水线加速比。
2. （√）流水线深度受限于流水线的延迟和额外开销。
3. （√）编译器可以通过重新排列代码的顺序来消除相关引起的暂停。
4. （√）多级存储层次是利用程序局部性原理来设计的。
5. （√）“Cache－主存”层次：弥补主存速度的不足。
6. （√）“主存－辅存”层次： 弥补主存容量的不足。
7. （√）写调块策略是用于写操作失效时的策略。
8. （√）写合并是提高写缓冲利用率的技术。
9. （√）相联度越高，冲突失效就越少。
10. （×）强制性失效和容量失效也受相联度的影响。
11. （×）容量失效却随着容量的增加而增加。
12. （√）2:1的Cache经验规则说明容量为N 的直接映象Cache的失效率约等于大小为N/2 的两路组相联Cache的失效率。
13. （√）一些降低失效率的方法会增加命中时间或失效开销。
14. （×）具有越低失效率的计算机系统性能越高。
15. （×）具有越低平均访存时间的系统性能越高。
16. （×）具有越低失效率的存储系统性能越高。
17. （√）具有越低平均访存时间的存储系统性能越高。
18. （×）Victim Cache是位于CPU和Cache间的又一级Cache。
19. （×）伪相联cache取直接映象及组相联两者的优点，命中时间小，失效开销低。
20. （√）伪相联cache具有快速命中与慢速命中两种命中时间。
21. （×）预取必须和正常访存操作并行才有意义。
22. （√）预取必须和正常指令的执行并行才有意义。
23. （√）数据对存储位置的限制比指令的少，因此更便于编译器优化。
24. （√）Cache中的写缓冲器导致对存储器访问的复杂化。
25. （√）Cache命中时间往往会直接影响到处理器的时钟频率。
26. （√）采用容量小、结构简单的Cache会减小cache的命中时间。
27. （√）写操作流水化会减小cache的命中时间。
28. （√）组相联或直接映象Cache中才可能存在冲突失效。
29. （√）TLB是页表转换查找缓冲器。
30. （√）TLB中的内容是页表部分内容的一个副本。
31. （×）程序的时间局部性指程序即将用到的信息很可能与目前正在使用的信息在空间上相邻或者临近。
32. （×）程序的空间局部性指程序即将用到的信息很可能就是目前正在使用的信息。
33. （√）Amdahl定律揭示的性能递减规则说明如果仅仅对计算机中的一部分做性能改进，则改进越多，系统获得的效果越小。
34. （×）Amdahl定律中“可改进比例”指可改进部分在改进系统计算时间中所占的比例。
35. （√）Amdahl定律中“部件加速比”指可改进部分改进以后性能的提高。
36. （√）执行时间不是唯一的性能指标，但它是最普遍的性能表示形式。
37. （×）RISC结构的机器性能一定要比CISC结构的机器性能高。
38. （×）平均每条指令的执行周期数（CPI）与程序无关。
39. （√）CPU性能公式中指令条数（IC）与指令集格式和编译器有关。
40. （√）CPU的组织在一定程度上会影响CPU所能达到的频率。
41. （√）解释执行比翻译执行花的时间多，但存储空间占用较少。
42. （×）计算机体系结构设计这不必关心指令集具体实现。

一、选择题：

1、在计算机系统结构来看，机器语言程序员看到的机器属性是（  **C**  ）。

A)计算机软件所要完成的功能 B)计算机硬件的全部组成

C)编程要用到的硬件知识 D)计算机各部件的硬件实现

2、对汇编语言程序员透明的是（**A** ）。

A)I/O方式中的DMA访问方式 B)浮点运算

C)程序性中断 D)存取操作数

4、在提高CPU性能的问题上，从系统结构角度，可以（**C** ）。P10

A)提高时钟频率 B)减少程序指令条数

C)减少每条指令的时钟周期数 D)减少程序指令条数和减少每条指令的时钟周期数

5、能实现指令、程序、任务级并行的计算机系统属于（ **D**）。

A)SISD B)SIMD　 C)MISD　 D)MIMD

6、计算机系统结构不包括（**A** ）。P4

A)主存速度 B)机器工作状态 C)信息保护 D)数据表示

10、推出系列机的新机器，不能更改的是（**A** ）。

A)原有指令的寻址方式和操作码 B)系统的总线的组成

C)数据通路宽度 D)存储芯片的集成度

11、在流水机器中，全局性相关是指（ **D**）。

A)先写后读相关 B)先读后写相关 C)指令相关 D)由转移指令引起的相关

12、下列说法不正确的是（ **D**）。

A)线性流水线是单功能流水线 B)动态流水线是多功能流水线

C)静态流水线是多功能流水线 D)动态流水线只能是单功能流水线

14、在系统结构设计中，提高软件功能实现的比例会（ **C**）。

A)提高解题速度 B)减少需要的存储容量 C)提高系统的灵活性 D)提高系统的性能价格比