**问答题**

**计算机系统结构设计和分析中最经常使用的三条基本原则是什么？**

**（1）大概率事件优先原则：对于大概率事件(最常见的事件)，赋予它优先的处理权和资源使用权，以获得全局的最优结果。**

**（2）Amdahl定律：加快某部件执行速度所获得的系统性能加速比，受限于该部件在系统中所占的重要性。**

**1. 名词解释：**

**SIMD：单指令多数据流计算机**

**MIMD：多指令多数据流计算机**

**CISC：复杂指令系统计算机，它是指按照进一步增强原有指令的功能以及设置更为复杂的新指令取代原先有软件子程序完成的功能，实现软件功能的硬化途径设计成CPU的计算机。**

**RISC：精简指令系统计算机，它是指按照通过减少指令总数和简化指令功能来降低硬件设计的复杂度，来提高指令执行速度的途径设计成CPU的计算机**

**VLSI超大规模集成电路；**

**MPP大规模并行处理器；**

**RISC精简指令系统计算机；**

**DMA直接存储器访问**

**WAR相关即Write After Read,写入数据一方试图在读数据一方读出数据之前，就进行写入，这样就可能造成读数据方读出的值是写入数据一方新写入的值（实际要读出的是原来的值），从而发生错误。**

**MIPS每秒执行百万条指令数，是用来描述计算机整体性能的。**

**2. 试以实例说明计算机系统结构、计算机组成与计算机实现之间的相互关系。**

**计算机系统结构作为一门学科，主要是研究软件，硬件功能分配和对软件、硬件界面的确定，即哪些功能由软件完成，哪些功能由硬件完成。**

**计算机系统结构，计算机组成和计算机实现是三个不同的概念。计算机系统结构是计算机系统的软硬件的界面；计算机组成是计算机系统**

**结构的逻辑实现；计算机实现是计算机组成的物理实现。**

**3．计算机系统结构的层次结构由高到低分别为**

**应用语言机器级，**

**高级语言机器级，**

**汇编语言机器级，**

**操作系统机器级，**

**传统机器语言机器级，**

**微程序机器级**

**4.1 常见的计算机系统结构分类法有哪两种？分类的依据是什么？**

**Flynn 分类法：按照指令流和数据流的多倍性进行分类。**

**冯氏分类法：用系统的最大并行度对计算机进行分类。**

**4.2 计算机系统Flynn 分类法，把计算机系统分成**

**单指令流单数据流(SISD)、**

**单指令流多数据流(SIMD)、**

**多指令单数据流、**

**多指令多数据流 四大类。**

**5．透明指的是**

**客观存在的事物或属性从某个角度看不到，它带来的好处是简化某级的设计，带来的不利是无法控制。**

**6.1 数据结构和数据表示之间是什么关系？**

**数据表示是能由硬件直接识别和引用的数据类型。**

**数据结构反映各种数据元素或信息单元之间的结构关系。**

**数据结构要通过软件映象变换成机器所具有的各种数据表示实现，所以数据表示是数据结构的组成元素。（2分）**

**不同的数据表示可为数据结构的实现提供不同的支持，表现在实现效率和方便性不同。数据表示和数据结构是软件、硬件的交界面。（2分）**

**6.2 确定和引入数据表示的基本原则是什么？**

**除基本数据表示不可少外，高级数据表示的引入遵循以下原则：   
（1）看系统的效率有否提高，是否养活了实现时间和存储空间。   
（2）看引入这种数据表示后，其通用性和利用率是否高。**

**7．计算机组成指的是计算机系统结构的逻辑实现，包括机器级内的数据流和控制流的组成及逻辑设计等。**

**计算机实现指的是计算机组成的物理实现。**

**8. 软件和硬件在什么意义上是等效的?在什么意义上是不等效的?**

**逻辑上等效，**

**性能、价格、实现难易程度上不一样。**

**9．说明翻译和解释的区别和联系.**

**区别：翻译是整个程序转换，**

**解释是低级机器的一串语句仿真高级机器的一条语句。**

**联系：都是高级机器程序在低级机器上执行的必须步骤。**

**10．计算机系统结构也称计算机体系结构，指的是传统机器级的系统结构。它是软件和硬件/固件的交界面，是机器语言汇编语言程序设计者或编译程序设计者看到的机器物理系统的抽象。**

**11．引入数据表示的两条基本原则是：**

**一看系统的效率有否提高；**

**二看数据表示的通用性和利用率是否高。**

**12. 指令集结构设计所涉及的内容有哪些？**

**指令集功能设计**

**主要有RISC和CISC两种技术发展方向**

**寻址方式的设计**

**设置寻址方式可以通过对基准程序进行测试设计，查看各种寻址方式的使用频率，根据使用频率设置必要地寻址方式**

**寻址方式的表示**

**可以将寻址方式编码于操作码中，也可以将寻址方式作为一个单独的字段来表示**

**操作数表示和操作数类型**

**可选择浮点型数据类型、整型数据类型、字符型、十进制数据类型等**

**指令格式的设计**

**有变长编码格式、定长编码格式和混合编码格式3种**

**13．何谓指令格式的优化?简要列举包括操作码和地址码两部分的指令格式优化可采用的各种途径和思路。**

**指令格式的优化指如何用最短位数表示指令的操作信息和地址信息，使程序中指令的平均字长最短。**

**①操作码的优化   
    采用Huffman编码和扩展操作码编码。   
②对地址码的优化：   
    采用多种寻址方式;   
    采用0、1、2、3等多种地址制;**

**在同种地址制内再采用多种地址形式，如寄存器-寄存器型、寄存器-主存型、主存-主存型等;   
    在维持指令字在存储器内按整数边界存储的前提下，使用多种不同的指令字长度。**

**14．CISC的中文意义是复杂指令计算机，RISC的中文意义是精简指令计算机。**

**15．简要比较CISC机器和RISC机器各自的结构特点，它们分别存在哪些不足和问题?为什么说今后的发展应是CISC和RISC的结合?**

**CISC结构特点：机器指令系统庞大复杂。  
RISC结构特点：机器指令系统简单，规模小，复杂度低。  
    CISC的问题：   
    (1)指令系统庞大，一般200条以上；   
    (2)指令操作繁杂，执行速度很低；   
    (3)难以优化生成高效机器语言程序，编译也太长，太复杂；   
    (4)由于指令系统庞大，指令的使用频度不高，降低系统性能价格比，增加设计人员负担。**

**RISC的问题；   
    (1)由于指令少，在原CISC上一条指令完成的功能现在需多条RISC指令才能完成，加重汇编语言程序设计负担，增加了机器语言程序长度，加大指令信息流量。   
    (2)对浮点运算和虚拟存储支持不很强。   
    (3)RISC编译程序比CISC难写。   
    由于RISC和CISC各有优缺点,在设计时，应向着两者结合，取长补短方向发展。**

**16、简要比较CISC和RISC机器各自的结构特点，它们分别存在哪些不足和问题，为什么说今后的发展应该是CISC和RISC的结合？**

|  |  |  |
| --- | --- | --- |
| **结构特性** | **CISC** | **RISC** |
| **指令系统规模与指令格式** | **指令系统庞大，指令格式可变** | **指令系统小，指令格式固定，大部分以寄存器为基础** |
| **寻址方式** | **12-24种** | **限制在3-5种** |
| **通用寄存器与高速缓存设计** | **通用寄存器8-24个，基本上使用合一的指令与数据高速缓存** | **基本上使用分开的数据与指令高速缓存，通用寄存器个数多（32-192）** |
| **时钟频率与CPI** | **钟频较低，CPI一般为2-15** | **钟频较高，CPI<1.5** |
| **CPU控制** | **大多数使用控制存储器（ROM）实现微指令控制，但现在也有使用硬连线控制** | **大多数不用控制存储器而用硬连线控制** |

**通过上表可以看出，CISC具有指令数量多，单条指令执行周期长等缺点；而RISC则具有不便于用户程序的开发等缺点，因此二者结合才是未来的发展方向。**

**17、请解释流水线技术时间和空间并行性。**

**空间并行性：设置多个独立的操作部件**

**时间并行性：分时使用同一个部件的不同部分**

**18. 计算机系统设计人员的技术挑战主要来自哪几个方面？**

**主要来自 系统结构、**

**设计工具、**

**制造工艺、**

**软件、应用和经济等多个方面。**

**19. 从执行程序的角度看，并行性等级从低到高可分为哪几级？**

**从执行程序的角度看，并行性等级从低到高可分为：**

**（1）指令内部并行：单条指令中各微操作之间的并行。**

**（2）指令级并行（Instruction Level Parallelism，ILP）：并行执行两条或两条以上的指令。**

**（3）线程级并行（Thread Level Parallelism，TLP）：并行执行两个或两个以上的线程，通常是以一**

**个进程内派生的多个线程为调度单位。**

**（4）任务级或过程级并行：并行执行两个或两个以上的过程或任务（程序段），以子程序或进程为**

**调度单元。**

**（5）作业或程序级并行：并行执行两个或两个以上的作业或程序。**

**20．为 提高流水线效率 可采用哪两种主要途径来克服速度瓶颈？**

**瓶颈段再细分（2分）**

**瓶颈段并联（3分）**

**(1) 细分瓶颈段；**

**(2) 重复设置瓶颈段。**

**21．简要解释提高计算机系统并行性的三个技术途径。**

**（1）时间重叠：引入时间因素，是让多个处理过程在时间上相互错开，轮流使用同一套设备的各个部分，以加快硬件周转来赢得速度。**

**（2）资源重复，是引入空间因素，通过重复设置硬件资源来提高性能。**

**（3）资源共享，是用软件方法让多个用户按一定时间顺序轮流使用同一套资源来提高其利用率，相应也就提高了系统的性能。**

**22. GCC的 –g 编译参数有什么含义?**

**答：在可执行程序中包含标准调试信息，生成调试信息，**[**GNU**](https://www.baidu.com/s?wd=GNU&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1dWuym3ujw-mvnLnHKbmW7B0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHRLPWnsrH61) **调试器可利用该信息。**

**23. objdump的 –S 参数有什么含义?**

**答：objdump命令是用于查看目标文件或者可执行的目标文件的构成。-S尽可能反汇编出源代码，尤其当编译的时候指定了-g这种调试参数时，效果比较明显。隐含了-d参数。**

**24. 程序中的条件语句编译为了哪些机器指令?**

**答：首先会根据cmp指令进行比较，然后根据状态寄存器的值，进行相应**

**跳转jl**

**25. 程序中的循环语句编译为了哪些机器指令?**

**答：会被编译成jmp跳转指令：**

**26. GDB的file，run，next，break，print，list，disassemble命令各有什么含义?**

**答：file：装入想要调试的可执行文件；**

**run：执行当前被调试的程序；**

**next：执行一行源代码但不进入函数内部；**

**break：在代码里设置断点，程序执行到这里时被挂起；**

**print：查看变量的值；**

**list：列出产生执行文件的源代码部分；**

**disassemble：查看源码的汇编。**

**27．“一次重叠”解释时，第K+1条指令需等K条指令执行后才能形成，称此时发生了“指令相关”。若第K条指令的结果数地址与第K+1条指令的源数地址一样时，称发生了“先写后读相关”。**

**28. 云计算的基本原理是通过使计算分布在大量的分布式计算机上，而非本地计算机或远程服务器中，企业数据中心的运行将更与互联网相似。这使得企业能够将资源切换到需要的应用上，根据需求访问计算机和存储系统。**

**问：当今互联网用户的需求是什么？**

**29. (1)Linux系统命令time给出的用户时间，系统时间，实际时间分别是什么含义；**

**Real时间是指挂钟时间，也就是命令开始执行到结束的时间。**

**这个短时间包括其他进程所占用的时间片，和进程被阻塞时所花费的时间；**

**User时间是指进程花费在用户模式中的CPU时间，**

**这是唯一真正用于执行进程所花费的时间，**

**其他进程和花费阻塞状态中的时间没有计算在内；**

**Sys 时间是指花费在内核模式中的CPU时间，**

**代表在内核中执系统调用所花费的时间，**

**这也是真正由进程使用的CPU时间。**

**(2)clock()和gettimeofday()函数获取的CPU时间和总时间是什么含义；**

**clock函数返回的是cpu时间，并不是秒数，真正的一秒钟可能包含若干个CPU时间，这个值通常是由宏CLOCKS\_PER\_SEC来定义，表示一秒中有CLOCKS\_PER\_SEC这么多个cpu时间，不同的编译器可能不同，我所用的虚拟机的编译器的CLOCKS\_PER\_SEC等于1000000，即1000000个时钟滴答数ticks为标准1秒。**

**gettimeofday()函数得到命令执行开始到结束的时间，这个总时间包括其他进程所占用的时间片，和进程被阻塞时所花费的时间。**

**(3)实验中各种方法获得的计时相关数值分别是什么精度，相互之间有什么关系？**

**答：Linux系统命令time给出的用户时间，系统时间，实际时间的精度：分、秒,秒保留3位小数，即精确到毫秒。**

**clock()函数获取的CPU时间可以根据代码设置精确，我精确到了毫秒。**

**gettimeofday()函数获取的总时间也可以根据代码设置精确，我精确到了毫秒。**

**clock()函数获取的CPU时间对应Linux系统命令time获取的用户时间；**

**gettimeofday()函数获取的总时间对应Linux系统命令time获取的实际时间。**

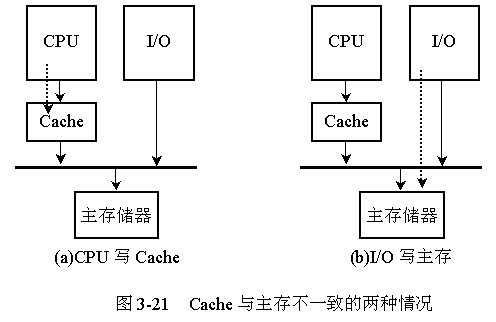
**30、造成Cache的不一致性的因素有哪些？**

**造成Cache与主存的不一致的原因：**

**(1) 由于CPU写Cache，没有立即写主存（写回法）**

**(2) 由于IO处理机或IO设备写主存**

**说明：Cache的内容是主存内容的一部分，是主存的一个小的副本，内容应该与主存保持一致。由于（1）CPU改写了Cache，没有立即写主存；（2）I/O处理机或I/O设备改写了主存，而Cache的内容没有更新，造成Cache的内容与主存内容不一致，**

****

**31、请列举Cache的更新算法，并比较其优缺点。**

**(1)写直达法，写通过法，WT(Write-through)**

**CPU的数据写入Cache时，同时也写入主存**

**(2) 写回法，抵触修改法，WB(Write-Back)**

**CPU的数据只写入Cache，不写入主存，仅当替换时，才把修改过的Cache块写回主存**

**写回法与写直达法的优缺点比较：**

**(1)可靠性，写直达法优于写回法。**

**写直达法能够始终保证Cache是主存的副本。**

**如果Cache发生错误，可以从主存得到纠正。**

**(2)与主存的通信量，写回法少于写直达法。**

**对于写回法：**

**大多数操作只需要写Cache，不需要写主存；**

**当发生块失效时，可能要写一个块到主存；**

**即使是读操作，也可能要写一个块到主存。**

**对于写直达法：**

**每次写操作，必须写、且只写一个字到主存。**

**实际上：**

**写直达法的写次数很多、每次只写一个字；**

**写回法是的写次数很少、每次要写一个块**

**(3)控制的复杂性, 写直达法比写回法简单。**

**对于写回法：**

**要为每块设置一个修改位，而且要对修改位进行管理；**

**为了保证Cache的正确性，通常要采用比较复杂的校验方式或校正方式。**

**对于写直达法：**

**不需要设置修改位；**

**只需要采用简单的奇偶校验即可。由于Cache始终是主存的副本，Cache一旦有错误可以从主存得到纠正。**

**(4)硬件实现的代价, 写回法要比写直达法好。**

**对于写直达法：**

**为了缩短写Cache流水段的时间，通常要设置一个小容量的高速寄存器堆（后行写数缓冲站），每个存储单元要有数据、地址和控制状态等3部分组成。**

**每次写主存时，首先把写主存的数据和地址写到高速寄存器堆中。**

**每次读主存时，要首先判断所读数据是否在这个高速寄存器堆中。**

**写回法不需要设置高速缓冲寄存器堆。**

**32. 解决流水线结构冲突的方法有哪些？**

**（1）流水化功能单元；（2）资源重复；（3）暂停流水线。**

**33. 根据指令对寄存器的读写顺序，可将数据冲突分为哪三种类型？**

**（1）写后读冲突**

**（2）写后写冲突**

**（3）读后写冲突**