第一章 计算机系统结构导论

一. 填空题：

1. 运算器（ALU）、控制器（CU）、存储器（MEM）、输入/输出设备（IN/OUT）；
2. 硬件、软件；
3. 全硬件方法、硬件软件结合方法、全软件方法；
4. 高级语言；
5. 系统结构、计算机组成、计算机实现；
6. 高；
7. 使用频率 、占总执行时间的比例；
8. 串行、并行处理机。

二. 选择题：

1.A 2.C 3.D 4.D 5.A 6.D 7.A 8.B 9.A

三. 问答题：

1. 答：冯·诺依曼型计算机系统结构的基本特点可以概括为以下几点：

（1）字长固定的存储器、顺序线性编址的一维结构；

（2）指令由操作码和地址码组成；

（3）指令在存储器中按顺序存放；

（4）指令和数据被同等对待；

（5）以运算器、控制器为中心；

（6）指令、数据均以二进制表示。

同时，其局限性在于：

（1）由于冯·诺依曼型结构以数值计算为主，因而对自然语言、图像、图形和符号处理的能力较差，不能满足上述领域的应用需求。

（2）由于程序算法从整体上来说是顺序型的，从而限制了并行操作的发挥，使计算机的运算速度不能在现有基础上取得根本性的突破。

（3）在该结构上发展起来的软件系统越来越复杂，正确性无法保证，软件生产率低下。

（4）该结构的硬件投资较大，可靠性差，在体系结构发展上受到限制。

（5）使用该结构的计算机应用人员需要既懂专业知识，又具备编程技巧。

2. 答：计算机系统由硬件和软件组成，二者是不可分割的整体。

硬件是计算机系统中的实际装置，是系统的基础和核心，一般由 CPU、MEM、I/O 接口、BUS 和外部设备等组成，它以机器语言（即指令系统）提供给程序员使用。软件指操作系统、汇编程序、编译程序、文本编辑程序、调试程序、数据库管理系统、文字处理系统、诊断程序以及各种应用程序等，基本上已与硬件的实现无关。

3. 答：系统结构是计算机系统的软、硬件界面，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。它们各自有不同的内容，但又有紧密的关系。系统结构设计不要对组成、实现有过多和不合理限制；组成设计应在系统结构指导下，以目前能实现的技术为基础；实现应在组成的逻辑结构指导下，以目前器件技术为基础，以优化性能价格比为目标。

4. 答：系统结构设计的步骤可以简要概括为如下内容：

（1）需求分析。对系统的应用环境（实时处理、分时处理、网络、远程处理、事务处理、科学计算、容错、高保密性等），所用语言的种类和特性，对操作系统的要求，所用外部设备的特性进行技术分析、经济分析和市场分析。

（2）需求说明。包括设计准则（造价、可靠性、可行性、可扩性、兼容性、速度、安全性、灵活性），功能说明，所用芯片说明，引入新结构成功的把握性以及程序设计方便性等。

（3）概念设计。根据上述已确定的准则进行软、硬件功能分析，确定机器级界面。

（4）具体设计。对机器级界面的各个方面进行详细、具体的定义，包括数据表示、指令系统、寻址方式、存储结构、中断系统、I/O 方式、总线结构等。该阶段可提出几种方案供选择。

（5）设计优化和评价。可反复进行，以期获得尽可能高的性能价格比。

5. 答：计算机系统设计的主要任务有下列三个方面：

（1）确定用户对计算机系统的功能、价格和性能要求；

（2）实现软、硬件的平衡；

（3）确保系统结构设计应符合今后发展的方向。

四. 计算题：

1.**解：**根据 Amdahl 定律可知：

（1）程序 A 最大可获得 $\frac{1}{0.6+\frac{0.4}{2}}=1.25$ 的加速比；

（2）程序 B 最大可获得$\frac{1}{0.01+\frac{0.99}{2}}=1.98$ 的加速比。

2.**解：**设向量化百分比为 x，则根据 Amdahl 定律可得$2=\frac{1}{(1-x)+\frac{x}{10}}$

解得 x = $\frac{5}{9}$ ，即当向量化百分比达到 56%时，才能使加速比为 2。

第二章 处理器及其相关技术

一. 填空题

9. 控制器、运算器、寄存器；

10. 微处理器；

11. 操作码、地址码、操作码；

12. RISC、CISC；

13. 星形总线、环形总线、Mesh 总线；

14. 高并发；

15. GPU；

16. 静态调度、动态调度、动静态调度；

17. 负载可预测应用、负载不可预测应用；

18. 硬件；

19. 共享内存并行系统。

二. 选择题

1.A 2.B 3.C 4.A 5.B 6.D 7.A 8.B 9.C 10.D

三. 问答题

1.答：RISC 和 CISC 是 CPU 的两种架构。RISC 克服了 CISC 系统庞杂的缺陷，其指令系统只包含使用频率很高的少量指令，并提供一些必要的指令以支持操作系统和高级语言。RISC-V 指令集是基于 RISC 原理建立的开放指令集架构，完全开源，设计简单，采用模块化设计，易于移植 UNIX 系统，可以根据具体场景进行选择。

2.答：中央处理器（CPU）由控制器、运算器和寄存器组成，三个部件的功能如下：

（1）控制器。其任务是负责从存储器中取出指令，确定指令的类型，并对指令进行译码，控制整个计算机系统一步一步地完成各种操作。

（2）运算器。为计算机提供计算与逻辑功能，控制器把数据带给 ALU 后，就根

据指令完成算术运算或逻辑判断及逻辑运算。

（3）寄存器。它是处理器内部的存储单元，主要存储当前指令信息和将要执行的下一条指令的地址等。

3.答：CPU 并行运算有三种实现方式：指令级并行、线程级并行以及数据并行。各自特点如下：

（1）指令级并行是一种指令层面的并行技术，目的是使 CPU 能够在同一时刻并行执行多条指令，其基本原理是：当指令之间不相关时，它们可以重叠起来并行执行。

（2）线程级并行是多处理器支持多个线程同时并行执行。实现 CPU 的线程级并行的基础是 CPU 具有多个物理内核，能够在每个内核上执行一个独立线程。

（3）数据并行是指把需要处理的数据划分成若干个部分分配到不同的处理部件上，并在每个处理部件上运行相同的处理程序对所分派的数据进行处理，即遵循 SIMD（单指令多数据流）模型。

4.答：GPU 的存储层次可由上至下分为三级，速度依次降低：

（1）寄存器：位于 GPU 上的每个 SM 中，访问速度最快，用于存放线程执行时所需要的变量，具有线程私有的性质。

（2）共享内存：位于每个 SM 中，访问速度仅次于寄存器，由一个 SM 中的所有线程共享。共享内存主要存放频繁修改的变量，可以为局部多线程建立通信。

（3）全局内存：位于 GPU 片外存储体，即显存。全局内存容量最大，但访问延迟高、速度较慢。所有 GPU 需要处理的数据都必须先传入全局内存，最后在程序执行完毕后从全局内存传给主机的内存。

5.答：CUDA 程序的一般执行流程为：

（1）调用 cudaMalloc()函数，在显存中开辟空间用于存储需要由 GPU 计算的数据。

（2）调用 cudaMemcpy()函数，将数据由主机内存复制到 GPU 显存中。

（3）启动核函数，进行相关高并行度计算。

（4）调用 cudaMemcpy()函数，将数据由显存取回内存中。

（5）调用 cudaFree()函数，释放显存占用空间。

四. 计算题

1.解：请具体计算步骤如下图所示：

步骤①：0.40 0.30 0.15 0.05 0.05 0.03 0.01 0.01

步骤②：0.40 0.30 0.15 0.05 0.05 0.03 **0.02**

步骤③：0.40 0.30 0.15 0.05 0.05 **0.05**

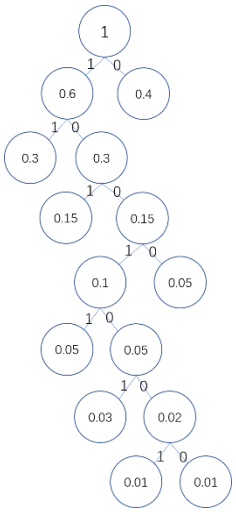
步骤④：0.40 0.30 0.15 **0.1** 0.05

步骤⑤：0.40 0.30 0.15 **0.15**

步骤⑥：0.40 0.30 **0.30**

步骤⑦：**0.60** 0.40

步骤⑧：**1.00**



最后得到哈夫曼编码如下表所示：

	P1	P2	P3	P4	P5	P6	P7	P8
编码	0	1000	101	100101	10011	11	1001001	1001000
码长	1	4	3	6	5	2	7	7

最短编码长度为：

H= 0.40*1+0.05*4+0.15*3+0.03*6+0.05*5+0.30*2+0.01*7+0.01*7 = 2.22

五. 编程题：

1. 答：方法 a 实现代码如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
int main(){
     int i,n;
     float a[100],b[100],result;
     //initializations
     n=100;
     result=0.0;
     for(i=0;i<n;i++){
          a[i]=rand()%5;
          b[i]=rand()%9;
     }
    #pragma omp parallel for
     for(i=0;i<n;i++){
    #pragma omp critical
          result=result+(a[i]*b[i]);
     }

     printf("%f\n",result);
     return 0;
}
```

结果输出：

```
687.000000

Process returned 0 (0x0)   execution time : 0.013 s
Press any key to continue.
```

方法 b 实现代码如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <stdio.h>
#include <omp.h>
//reduction
int main(){
     int i,n,chunk;
     float a[100],b[100],result;
     //initializations
     n=100;
     chunk=10;
     result=0.0;
     for(i=0;i<n;i++){
          a[i]=rand()%5;
          b[i]=rand()%9;
     }
    #pragma omp parallel for
default(shared)private(i)schedule(static,chunk)reduction(+:result)
```

```
        for(i=0;i<n;i++)
            result=result+(a[i]*b[i]);

        printf("%f\n",result);
        return 0;
    }
```

结果输出：

```
687.000000

Process returned 0 (0x0)   execution time : 0.083 s
Press any key to continue.
```

2.答：实现代码如下：

```
    #include <stdio.h>
    __global__ void helloFromGPU(void){
        printf("hello world from GPU!\n");
    }
    int main(void){
        printf("hello world from CPU!\n");

        helloFromGPU <<<1,10>>>();
        cudaDeviceReset();
        return 0;
    }
```

第三章 存储系统结构

一. 填空题

20. 加基址、地址映像；

21. 地址映像、地址变换；

22. 随机算法（RAND）、先进先出算法（FIFO）、近期最少使用算法（LRU）、优化替换算法（OPT）；

23. 并行存储器；

24. 相同；

25. Cache；

26. 全相联方式、页表；

27. 存储区域、访问方式；

28. 磁盘阵列。

二. 选择题

1.A 2.B 3.C 4.C 5.A 6.D

三. 问答题

1.答：程序定位，即如何把程序的逻辑地址变换成实际的主存物理地址，分为加基址方式和地址映像方式。加基址方式分为静态定位和动态定位两种，地址映像方式有段式管理，页式管理和段页式管理三种。

2.答：目前常见的几种 Cache 优化策略是：减少不命中代价、降低不命中率、采用并行技术、减少 Cache 命中时间。

3.答:常见的地址映像方式有全相联映像、直接映像、组相联映像和段相联映像，各自特点如下：

（1）全相联映像是指任何虚页可映像到实存的任何页面位置，地址变换方法分为页表法和目录表法两种。

（2）直接映像的定义是每个虚页只能映像到实存的一个特定页面，优点是地址变换时，只需将相应部分拼接，缺点是实页冲突概率高、利用率低。

（3）组相联映像是各组之间是直接映像，组内各页间是全相联映像。它是全相联映像和直接映像两种映像方式的结合。

（4）段相联是全相联的一个特例，即段间是全相联映像，段内各页间是直接映像。

4.答：常见的替换算法主要有下列 4 种：

（1）随机算法。用硬件或软件随机产生被替换的页号。由于没有利用实存使用的“历史”信息，不能反映程序的局部性，故命中率很低。

（2）先进先出算法。先进入实存的页先被替换。该算法虽然利用了实存使用的“历史”信息，让最早进入实存的页被替换出去，但没有正确反映程序的局部性。因为最早进来的页，也许是目前或最近一段时间内要经常用到的页，所以该算法不能反映实际使用需求。

（3）近期最少使用算法。把近期最久未访问的页替换出去。这种“近期”是指过去了的近期，该算法是根据过去的近期使用状况预测未来近期中哪一页可能不被使用，并将其替换出去，故能比较准确地反映程序的局部性，命中率有所提高。

（4）优化替换算法。预测各页今后使用时刻，选择其中时间间隔最长的页替换出去。

5.答：Cache 的主要作用是弥补主存和 CPU 之间的速度差距，因此它的管理部件是用硬件实现的，并对程序员透明。虚拟存储器的主要作用是弥补主存和辅存之间的容量差距，因此它的管理部件基本上靠软件，适当结合硬件来实现，并且虚拟地址空间可被应用程序员感觉到和加以利用，实现对系统程序员也不是透明的。

四. 计算题

1.**解：**先求命中率 h:$\mathrm{h} = \frac{2420}{2420+80} = 0.968$

平均访问时间$T_a = 0.968 * 40 + (1 - 0.968) * 240 = 46.4(\mathrm{ns})$

其中$r = \frac{240}{4} = 6$

则 Cache/主存系统的效率 $\mathrm{e} = \frac{1}{r+(1-r)*0.968} = 86.2\%$

2.**解：**Cache 容量为 16KB（2^{14}个字节）,而 Cache 块的大小为 16B (2^4个字节)，因此 Cache 可分为 16KB/16B=1K =2^{10}=1024 块，于是块地址占 10 个二进制位，块内地址占 4 个二进制位。由于采用直接映像的方式，所以主存的后 14 位就是要装入的到 Cache 中的位置。主存地址 1234E8F8 是十六进制数，它的一个位对应 4 个二进制位，将各个位分别化为二进制数便把这个数化成了二进制数，1234E8F8 对应的二进制数为:10010001101001110100011111000，得 Cache 的地址为:10100011111000。

Cache 地址结构：

块号	块内地址
10	4

内存 地址结构：

?	块号	块内地址
?	10	4

第四章 流水线结构

一、术语解释

单功能性流水线: 只能完成一种功能的流水线。在计算机中要实现多个功能，都采用多个单功能流水线。

多功能流水线：同一个流水线可有多种连接方式来实现多种功能。

静态流水线：同一时间内，流水线的各段只能按同一种连接方式工作。

动态流水线：在同一时间内，流水线的各段可按不同运算的连接方式工作。

线性流水线：流水线各段串行连接，没有反馈回路。

非线性流水线: 流水线中除了串行连接通路，还有反馈回路。在流水过程中，有些段要反复多次使用，它常用于递归或组成多功能流水线。

超标量处理机：超标量处理机中使用了多指令流水线，每个时钟周期同时发射多条指令并产生多个结果。

超流水线处理机：超流水线处理机是通过各部分硬件的充分重叠工作来提高处理机性能。一台并行度 ILP 为 n 的超流水线处理机，在一个时钟周期内能发射 n 条指令，每隔 $1/n$ 个时钟周期发射一条指令。

控制相关：当流水线遇到转移指令或其他改变 PC 值的指令而造成断流时，会引起控制相关。

数据相关：在一个程序中，存在必须等前一条指令执行完才能获得需要的数据，然后执行下一条指令的情况，那么这两条指令数据相关。

先行控制方式：在执行部件执行第 k 条指令的同时，指令控制部件对其后继的 k+1，k+2…条指令进行预取和预处理，为执行部件执行新的指令做好必要和充分的前期准备。这样，就能使指令分析部件和指令执行部件连续、流畅地工作。流程中指令分析部件和执行部件之间有等待的时间间隔Δt，但它们各自的流程却是连续的。这种方式称为先行控制方式。

写后读相关：假设指令 j 是在指令 i 后面执行的指令，RAW 表示指令 i 将数据写入寄存器后，指令 j 才能从这个寄存器读取数据。如果指令 j 在指令 i 写入寄存器前尝试读出该寄存器的内容，将得到不正确的数据。

写后写相关：假设指令 j 是在指令 i 后面执行的指令，WAW 表示指令 i 将数据写入寄存器后，指令 j 才能将数据写入这个寄存器。如果指令 j 在指令 i 之前写该寄存器，将使得该寄存器的值不是最新值。

二、问答题

1. 答：一条指令的解释过程可以通过多个独立的部件完成。当一条指令的分析子过程在指令分析器中结束，并将结果送入执行部件去实现执行子过程时，指令分析器不必等本指令在执行部件中完成后再对下一条指令进行分析子过程，而是同时进行。

流水线有如下特点：（1）可以划分为若干互有联系的子过程（功能段）。每个功能段由专用功能部件实现。（2）实现功能段所需的时间应尽可能相等，避免因不等产生处理瓶颈，造成流水线“断流”。（3）形成流水线处理，需要一段准备时间，称为“通过时间”。只有在此之后流水过程才能够稳定。（4）指令流不能顺序执行时，会使流水过程中断；再形成流水过程，则需经过一段时间。不应经常“断流”，否则效率不会很高。（5）流水线技术适用于大量重复的程序过程，只有输入端能连续地提供服务，流水线效率才能够得到充分发挥。

2. 答：不是越多越好，增加流水线的段数，流水线的吞吐率和加速比都会提高。由于在每段的输出端都必须设置一个锁存器（或称缓冲寄存器），因此段数增多时各锁存器的延迟时间累加值也将增加，甚至有可能超出流水线各段的延迟时间的累加值。同时，增加锁存器也增加了流水线硬件价格。所以，在设计流水线时，要综合各方面的因素，根据最佳性能价格比的要求，选择流水线的最佳段数。选取时可以参考流水的加速比、吞吐量和效率等参数。

3. 答：指令流水线中的某个过程段，有时会因为阻塞，无法在规定的时间内完成它的任务，致使流水线断流。造成流水线断流的阻塞有三种，即结构阻塞、数据阻塞、控制阻塞。解决结构阻塞可以使相关的指令暂停一个或多个时钟周期执行。解决数据阻塞的方式指令暂停执行或者设置相关的专用数据通如。解决控制阻塞的方式是分支预测技术和预先取值成功和不成功两个控制流方向上的目标指令

三、填空题

1. 10、2.4、7

2. 2n+1、n+2

3. 先行读数栈、先行操作栈和后行写数栈

四、选择题

1. A

2. C

3. A

五、计算题

1.

(1) 100ns，对齐最长一步耗时

(2) 写后读相关，下表相比未发生相关，延迟 200ns

	1	2	3	4	5	6	7
ADD	取指	译 码 取数	运算	写回			
SUB		取值			译 码 取数	运算	
							写回

(3) 添加数据旁路，第一条指令运算完后，无需写回，第二条指令就可译码取数，还需延迟 100ns

2. 指令如下：(中间结果寄存器 R,源操作数寄存器 A,最后结果 F)

I1: R1 ← A1+A2

I2: R2←A3+A4

I3: R3←A5+A6

I4: R4←A7+A8

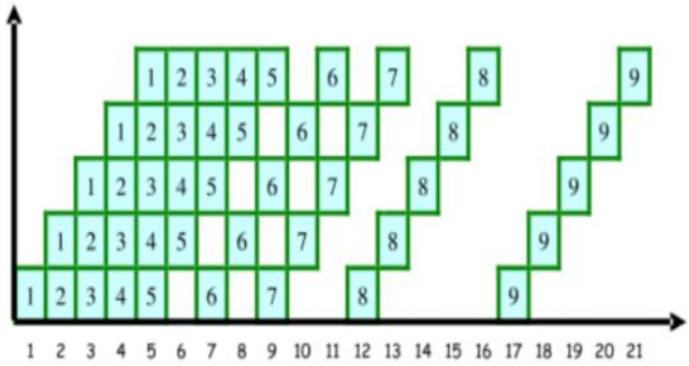
I5: R5←A9+A10

I6: R6←R1+R2

I7: R7←R3+R4

I8:　R8←R5+R6

I9：　F←R7+R8

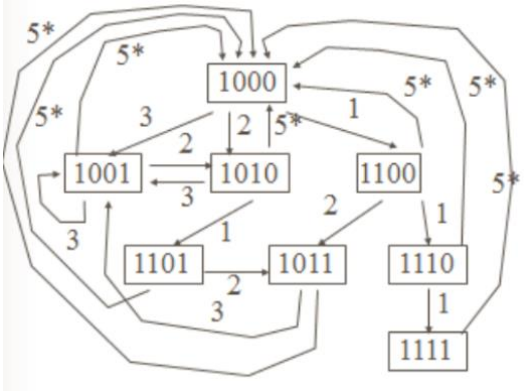


吞吐率　9/(21*t)　加速比　(9*5)/21=2.1429　效率　(9*5)/(21*5)=3/7=0.43

3.

(1) 禁止向量(4),初始冲突向量(1000)

(2) 如图所示



(3) 如表格所示

1,5	6/2	2,1,2,5	10/4
1,1,5	7/3	2,3,5	10/3
1,1,1,5	2	3,5	8/2
1,2,5	8/3	3	3
1,2,3,5	11/4	3,2,5	10/3
2,5	7/3	3,2,1,5	11/4
2,1,5	8/3	2,3	5/2

(4) (1,1,1,5)　最小等待为 2

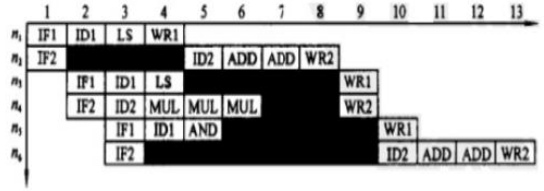
4.

(1) n1 和 n2 关于寄存器 R0 的写读数据相关　　n3 和 n6 关于寄存器 R2 的写读数据相关

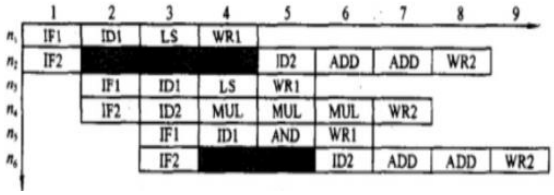
n4 和 n5 关于寄存器 R4 的读写数据相关　　n3 和 n6 关于寄存器 R2 的写写数据相关

(2)

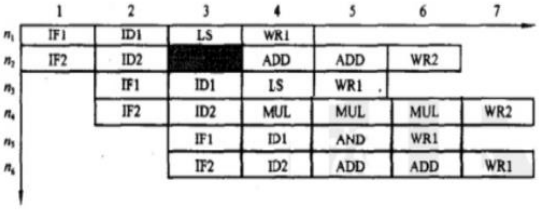
顺序发射顺序完成



顺序发射乱序完成



采用数据通路的顺序发射乱序完成



第五章 并行处理机与多处理机系统

一、术语解释

互连网络：互连网络是一种由开关元件按照一定的拓扑结构和控制方式构成的网络，用于计算机系统内部多个处理机或多个功能部件之间的相互连接。

结点度：与结点相连接的边（即链路或通道）数称为结点度。在单向通道情况下，进入结点的通道数称为入度（In Degree），而从结点出来的通道数则称为出度（Out Degree）。结点度为入度和出度之和，它反映了结点所需要的 I/O 端口数，也反映了结点的价格。

网络直径：它是网络中任意两个结点之间距离的最大值。这是网络通信性能的一个指标。从通信的角度来看，网络直径应当尽可能小。

等分带宽：当某一网络被分成相等的两半时，沿分界面的最小边数（通道）称为通道等分宽度。一个网络可以有多个等分平面，最小等分平面是指具有最小连线数的等分平面。网络的等分带宽指每秒在最小等分平面上通过所有连线的最大位数或字节数。

静态互连网络：静态互连网络是指各结点间有专用的链路且在运用中不能改变的网络。在此类网络中，每一个开关元件固定与一个结点相连，建立该结点与邻近结点间的连接通路，直接实现两个结点之间的通信。静态互连网络适用于构造通信模式可预测的计算机或者用静态连接实现通信的计算机。

动态互连网络：动态互连网络可以实现多种用途和通用目的，它能根据程序要求实现所有通信模式，使用开关或仲裁器以提供动态连接特性。动态互连网络的价格和所采用的链路、开关、仲裁器的成本有关，其性能涉及网络带宽、数据传输速率、网络时延和所用的通信模式。常见的动态互连网络有总线互连、交叉开关互连等。

静态一致性校验：其基本思想是：只让该进程的独用信息（指令和操作数据）和共享只读信息进入本处理机的 Cache，而不准共享可写（即可修改）信息进入 Cache，让其只留在主存中。

动态一致性校验：基本思想是，在若干 Cache 中使同一个信息（指令、数据）始终保持动态一致。一种方法是广播法。当每个处理机每次写 Cache 时，不仅写入自己的 Cache 和共享的主存，而且还把信息送到所有其他 Cache，如果其他 Cache

有与自己 Cache 相同的目标行，则也进行改写。另一种方法是目录法。在快速 RAM 中构建一个目录表，按条目标识信息的状态。

二、问答题

1. 并行性有不同的等级。从执行角度看，并行性等级可从低到高划分为：

(1) 指令内部并行，即指令内部的微操作之间的并行

(2) 指令间并行，即并行执行两条或多条指令

(3) 任务级或过程级并行，即并行执行两个或多个过程或任务（程序段）

(4) 作业或程序级并行，即在多个作业或程序间的并行

2. 计算机系统提高并行性的措施很多，就其思想而言，可归纳为下列三种技术途径:

(1) 时间重叠,多个处理过程在时间上相互错开，轮流重叠使用同一套硬件的各个部件，以加快部件的周转而提高速度。指令流水线是最典型的时间重叠。

(2) 资源重复, 重复设置多个硬件部件以提高计算机系统的性能，例如多处理机。

(3) 资源共享，利用软件方法，使多个用户分时使用同一个部件或设备，典型如分时系统。

3. 多处理机要实现任务一级的并行，不能再象 SIMD 计算机那样只能对多数据流执行同一指令操作。因此，在结构上，它的多个处理机要用多个指令部件分别控制，并且要有复杂的互连网络实现机间通信；在算法上，不限于数组向量处理，要挖掘和实现更多通用算法中隐含的并行性；在系统管理上，要更多依靠软件手段有效地解决资源管理，特别是处理机管理以及进程调度等问题。**根本原因**是因为：多处理机属于多指令流多数据流(MIMD)计算机，而并行处理机属于单指令流多数据流(SIMD)计算机，它们的差别归结底来源于并行性级别的不同。

三、填空题

1. 4、12、2^{12}、8!、阻塞

2. SIMD、MISD

3. 静态优先级算法、固定时间片算法、动态优先级算法、先来先服务算法

4. 紧密耦合多处理机、松散耦合多处理机

5. 共享主存多处理机、分布主存多处理机

6. 23016745

四、选择题

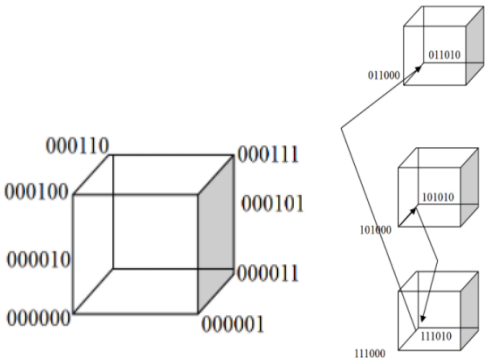
1. D

2. B

五、综合题

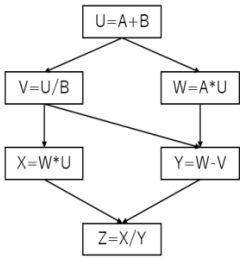
1. 4、4、11、9

2. 101000→101010→111010→011010



3.

(1) 如图所示



(2) 改写的程序如下

```
10 U=A+B
       FORK 30
20 W=A*U
      JOIN 2
      GOTO 40
30 V=U/B
      JOIN 2
40 FORK 60
50 X = W*U
```

JOIN 2

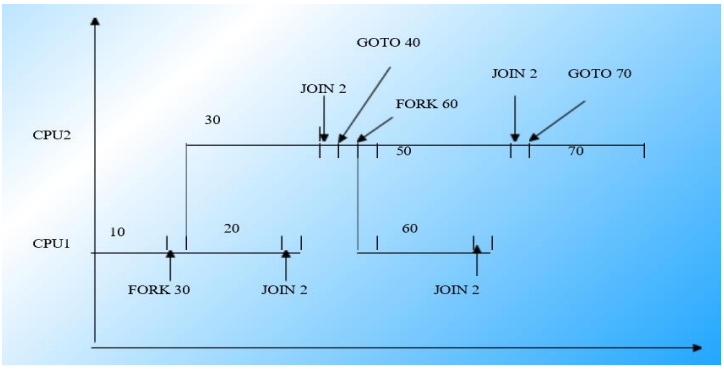
GOTO 70

60 Y=W-V

JOIN 2

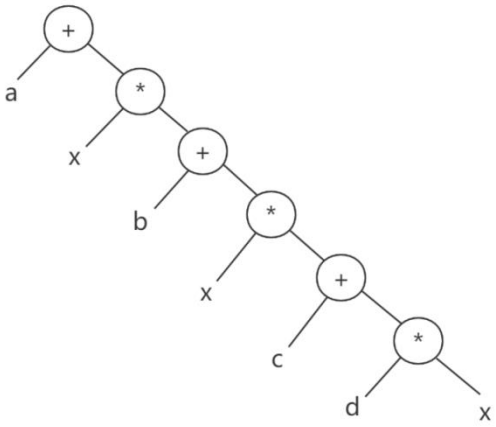
70 Z= X/Y

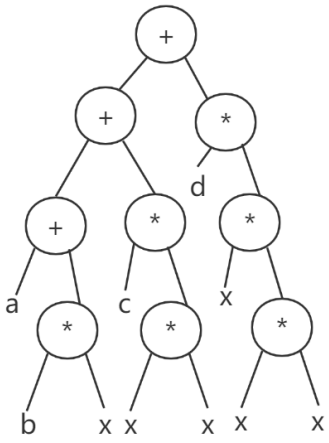
(3) 如图所示



4.

(1) 分别如图所示





(2) 顺序 6 级运算，减少树高可采用 4 级运算

Tp=4　P=3　Sp=6/4　Ep=Sp/p=1/2

第六章 集群、网格和云计算

一、术语解释

高可用集群: 这类集群致力于提供高度可靠的服务——利用集群系统的容错性对外提供 7×24 小时的不间断服务，如高可用的文件服务器、数据库服务等关键应用。

负载均衡集群: 这类集群可以使任务在集群中尽可能平均地分摊到不同的计算节点处理，充分利用集群的处理能力，提高对任务的处理效率。

高性能计算集群： 这种集群上运行的是专门开发的并行应用程序（例如 MPI，Hadoop，Spark 等），它可以把一个问题的计算任务分配到多个计算节点上，利用这些计算节点的资源来完成任务，从而完成单机不能胜任的工作。

infiniBand: InfiniBand 支持 RDMA 网络协议。由于这是一种新的网络技术，因此需要支持该技术的网卡和交换机。InfiniBand 的基本带宽是 2.5Gb/s，这是 InfiniBand 1.x。InfiniBand 是全双工的，因此在两个方向上的理论最大带宽都是 2.5Gb/s，总计 5Gb/s

OpenStack： penStack 是由 NASA（美国国家航空航天局）和 Rackspace 合作研发的项目，且通过 Apache 许可证授权开放源码。OpenStack 既是一个社区，也是一个项目和一个开源软件。OpenStack 是一个可以管理拥有大量计算、存储和网络资源等的数据中心的云计算平台操作系统，通过仪表板为管理员提供所有的管理控制功能，通过 Web 界面为用户提供云计算资源等服务。开发者可以通过 API 访问云计算资源和创建云应用，可以为公有云、私有云等不同规模的云提供可扩展的、灵活的云计算。

SaaS: SaaS 平台供应商将某些特定应用软件功能封装成服务。统一部署在自己的服务器上，客户可以根据实际需求，通过互联网向其订购所需的应用软件服务，按订购的服务数量和时长向其支付费用，并通过互联网获得其提供的服务。

PaaS: PaaS 对资源的抽象层次更进一步，它提供用户应用程序的运行环境。PaaS 通过全球互联网为开发、测试和管理软件应用程序提供按需开发环境。客户不需要管理或控制底层的云基础设施，包括网络、服务器、操作系统、存储设备等，但可以控制部署的应用程序，也可以控制运行应用程序的托管环境配置。

网格: 网格技术是通过高速网络连接，统一管理各类不同物理位置的资源（超级计算机、大型数据库、存储设备、各种仪器设备、知识库等），配置系统软件、工具和应用环境，使之成为一个互相协调的先进计算设施。

计算节点: 计算节点是集群系统中数量最多的节点，用来完成用户提交的计算任务。集群的性能取决于所有计算节点的性能及其发挥情况。因此，计算节点需要有强大的性能。计算节点的性能不仅取决于计算性能，还取决于存储性能和通信性能，是一个系统整体的综合表现。计算性能涉及所配置 CPU 的核数、主频及相应的加速部件等。

管理节点: 管理节点的主要功能是通过各种软件对集群系统进行安装、维护、运行状态监控、资源管理和作业管理等。例如，作业管理就是将用户提交的计算任务按照预设的调度方法通过管理节点调度到计算节点上进行计算，并对作业的运行情况进行监控和管理。对于小型集群，可以将这些功能软件安装在一台计算机上，但是考虑到系统的性能和安全，一般会把这些软件安装在单独的服务器上，特别是管理软件和作业调度软件，一般都安装在独立的服务器上，甚至还需要进行热备份，这些就是集群中的管理节点和作业调度节点。

二、问答题

1. 有如下特点：

(1) 系统开发周期短。

(2) 用户投资风险小。

(3) 系统价格低。

(4) 节约系统资源。

(5)系统扩展性好。

(6) 用户编程方便。

2. 容器和虚拟机之间的主要区别在于虚拟化层的位置和操作系统资源的使用方式。虚拟机需要安装操作系统，系统会将虚拟硬件、内核以及用户空间打包在新虚拟机中，利用“虚拟机管理程序”运行在物理设备上。容器不需要安装操作系统，而是通过目录和命名空间的隔离直接在宿主系统上运行。容器可以看成是按需装好一组特定应用的虚拟机，直接利用了宿主机的内核，抽象层比虚拟机更少，更

加轻量化，启动速度更快。虚拟机已经是比较成熟的技术了，而容器技术作为下一代虚拟化技术，代表着未来的发展方向。

三、填空题

1. 资源池层、管理中间件层
2. 计算机、储存系统和网络
3. 节点机的维护、存储系统的维护和网络设备的维护
4. 存储、网络、cpu 等计算资源
5. 私有云、公共云、混合云
6. 大量、多样、高速

四、综合题

1. 略
2. 略