第一章

1.2 试用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。

答：如在设计主存系统时，确定主存容量、编址方式、寻址范围等属于计算机系统结构。确定主存周期、逻辑上是否采用并行主存、逻辑设计等属于计算机组成。选择存储芯片类型、微组装技术、线路设计等属于计算机实现。

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。一种体系结构可以有多种组成。一种组成可以有多种实现。

1.4 计算机系统“从中间开始设计”方法中的“中间”指的是什么地方？这样设计的好处是什么？

答： “中间”是指层次结构中的软硬件的交界面，目前一般是在传统机器语言级与操作系统机器级之间。

好处：软件和硬件并行设计可以缩短设计周期，设计过程中可以交流协调，是一种交互式的很好的设计方法。

1.7 某台主频为 400MHz 的计算机执行标准测试程序，程序中指令类型、执行数量和平均时钟周期数如下：

指令类型	指令执行数量	平均时钟周期数
整数	45000	1
数据传送	75000	2
浮点	8000	4
分支	1500	2

求该计算机的有效 CPI、MIPS 和程序执行时间。

解：（1）CPI ＝(45000×1＋75000×2＋8000×4＋1500×2) / 129500＝1.776

（2）MIPS 速率＝f/ CPI ＝400/1.776 ＝225.225MIPS

（3）程序执行时间＝ (45000×1＋75000×2＋8000×4＋1500×2) / 400=575s

1.8 已知 4 个程序在 3 台计算机上的执行时间如下表所示：

程序	执行时间（秒）		
	计算机 A	计算机 B	计算机 C
程序 1	1	10	20
程序 2	1000	100	20
程序 3	500	1000	50
程序 4	100	800	100

假设 4 个程序都执行 100 000 000 条指令，计算着 3 台计算机中每台机器上每个程序的 MIPS 速率，分别计算它们的算术平均值、几何平均值和调和平均值。

解：在这里用到的公式有：

MIPS速率 $MIPS=IC/(T*10^6)$ 其中：IC为指令条数，T为执行时间。

算术平均值 $S_m=\frac{1}{n}\sum_{i=1}^{n}T_i$ 其中：Ti是执行第i个测试程序的执行时间，n是测试程序组中程序的个数。

几何平均值 $G_m=\sqrt[n]{\prod_{i=1}^{n}R_i}=\sqrt[n]{\prod_{i=1}^{n}\frac{1}{T_i}}$ 其中：Ri是由n个程序组成的工作负荷中执行第i个程序的速度，Ri=1/Ti。

调和平均值 $H_m=\frac{n}{\sum_{i=1}^{n}\frac{1}{R_i}}=\frac{n}{\sum_{i=1}^{n}T_i}$ 其中：Ri是由n个程序组成的工作负荷中执行第i个程序的速度，Ri=1/Ti。
Ti是第i个程序的执行时间。

因此：

程序	MIPS 速率（百万条指令/秒）		
	A	B	C
程序 1	100	10	5
程序 2	0.1	1	5
程序 3	0.2	0.1	2
程序 4	1	0.125	1

故：

程序	MIPS 速率		
	A	B	C
算术平均值	25.3	2.81	3.25
几何平均值	1.19	0.59	2.66
调和平均值	0.25	0.20	2.1

1.9 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：

$$\text{系统加速比}=\frac{1}{(1-0.4)+\frac{0.4}{10}}=1.5625$$

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。

1.10 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：

$$S_n=\frac{1}{(1-\sum F_i)+\sum\frac{F_i}{S_i}}$$

已知 S_1＝30，S_2＝20，S_3＝10，S_n＝10，F_1＝0.3，F_2＝0.3，得：

$$10=\frac{1}{1-(0.3+0.3+F_3)+(0.3/30+0.3/20+F_3/10)}$$

得 F_3＝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 S_1＝30，S_2＝20，S_3＝10，因此 3 个部件改进后的执行时间为：

$$T_n'=\frac{0.3T}{30}+\frac{0.3T}{20}+\frac{0.2T}{10}=0.045T$$

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：

$$\frac{0.2T}{0.245T}=0.82$$

1.11 假设浮点数指令 FP 指令的比例为 30%，其中浮点数平方根 FPSQR 占全部指令的比例为 4%，FP 操作的 CPI 为 5，FPSQR 操作的 CPI 为 20，其他指令的平均 CPI 为 1.25.现有两种改进方案，第

一种是把 FPSQR 操作的 CPI 减至 3，第二种是把所有的 FP 操作的 CPI 减至 3，试比较两种方案对系统性能的提高程度。

解：改进之前，系统指令平均始终周期 CPI 为 CPI=∑(CPIi*(Ii/IC))=(5*30%)+(1.25*70%)=2.375.

如果采用 A 方案：FPSQR 操作的 CPI 减至 3，则整个系统的平均时钟周期数为：

CPIA=CPI-(CPIFPSQR-CPI'FPSQR)*4%=2.375-(20-3)*4%=1.695

如果采用 B 方案：把所有的 FP 操作的 CPI 减至 3，则整个系统的平均时钟周期数为：

CPIB=CPI-(CPIFP-CPI'FP)*4%=2.375-(5-3)*30%=1.775

从降低整个系统的指令平均时钟周期数的程度来看，方案 A 要优于 B。

另外，分别计算两种方案的加速比：（SA=改进秦 CPU 的执行时间/A 的 CPU 执行时间=(IC*时钟周期*CPI)/(IC*时钟周期*CPIA)=CPI/CPIA）

SA=2.375/1.695=1.4

SB=2.375/1.775=1.34

由此也可知，方案 A 优于方案 B。

第二章

2.2 区别不同指令集结构的主要因素是什么？根据这个主要因素可将指令集结构分为哪 3 类？

答：区别不同指令集结构的主要因素是 CPU 中用来存储操作数的存储单元。据此可将指令系统结构分为堆栈结构、累加器结构和通用寄存器结构。

2.5 指令集结构设计所涉及的内容有哪些？

答： (1) 指令集功能设计：主要有 RISC 和 CISC 两种技术发展方向； (2) 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统计，察看各种寻址方式的使用频率，根据适用频率设置必要的寻址方式。 (3) 操作数表示和操作数类型：主要的操作数类型和操作数表示的选择有：浮点数据类型、整型数据类型、字符型、十进制数据类型等等。 (4) 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域来表示。 (5) 指令集格式的设计：有变长编码格式、固定长度编码格式和混合型编码格式 3 种。

2.8 指令中表示操作数类型的方法有哪几种？

答：操作数类型有两种表示方法：（1）操作数的类型由操作码的编码指定，这是最常见的一种方法；（2）数据可以附上由硬件解释的标记，由这些标记指定操作数的类型，从而选择适当的运算。

2.9 表示寻址方式的主要方法有哪些？简述这些方法的优缺点。

答：表示寻址方式有两种常用的方法：（1）将寻址方式编于操作码中，由操作码在描述指令的同时也描述了相应的寻址方式。这种方式译码快，但操作码和寻址方式的结合不仅增加了指令的条数，导致了指令的多样性，而且增加了 CPU 对指令译码的难度。（2）为每个操作数设置一个地址描述符，由该地址描述符表示相应操作数的寻址方式。这种方式译码较慢，但操作码和寻址独立，易于指令扩展。

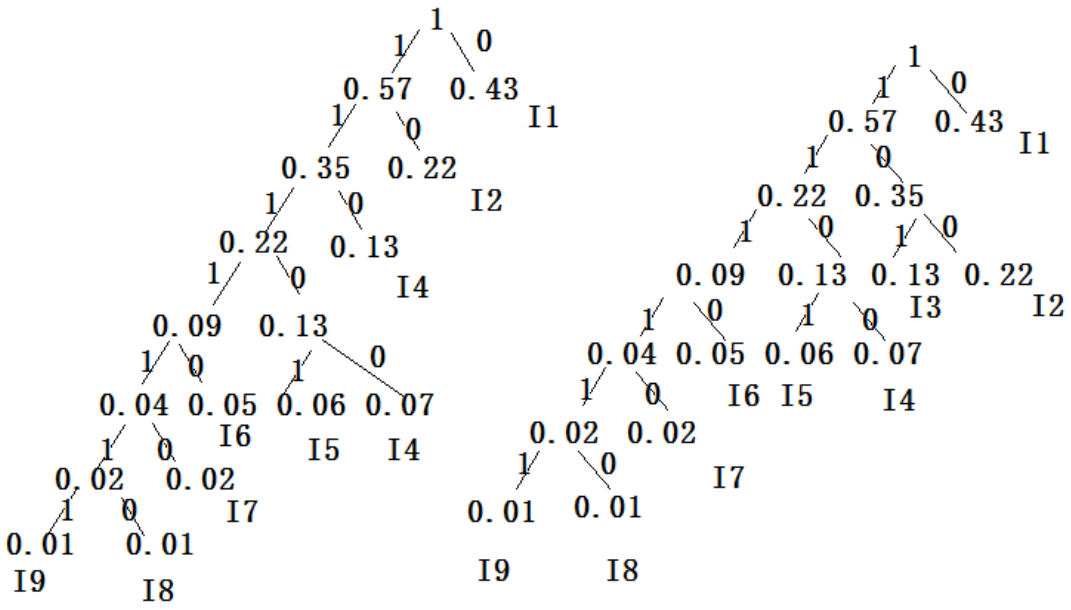
2.11 某台处理机的各条指令使用频度如下表所示：

指令	使用频度	指令	使用频度	指令	使用频度
ADD	43%	JOM	6%	CIL	2%
SUB	13%	STO	5%	CLA	22%
JMP	7%	SHR	1%	STP	1%

请分别设计这 9 条指令操作码的哈弗曼编码、3/3/3 扩展编码和 2/7 扩展编码，并计算这 3 种编码的平均码长。

解：

根据给出的九条指令的使用频度和哈弗曼生成算法的结构的不用构造了两种不同的哈夫曼树。



（左边为 A，右边为 B）

各编码如下：

指令	Ii	Pi	哈弗曼 A	哈弗曼 B	3/3/3	2/7
ADD	I1	0.43	0	0	00	00
CLA	I2	0.22	10	100	01	01
SUB	I3	0.13	110	101	10	1000
JMP	I4	0.07	11100	1100	1100	1001
JOM	I5	0.06	11101	1101	1101	1010
STO	I6	0.05	11110	1110	1110	1011
CIL	I7	0.02	111110	11110	111100	1100
SHR	I8	0.01	1111110	111110	111101	1101
STP	I9	0.01	1111111	111111	111110	1110

由表可知，三种编码的平均码长为：(公式：$L=\sum Pi*Li$)

哈弗曼编码：2.42 位

3/3/3 编码：2.52 位

2/7 编码：2.70 位

平均码长：2*43%+2*22%+4*（1-43%-22%）=2.7

2. 12 某机指令字长 16 位。设有单地址指令和双地址指令两类。若每个地址字段为 6 位.且双地址指令有 X 条。问单地址指令最多可以有多少条?

解：

双地址指令结构为：（4 位操作码）（6 位地址码）（6 位地址码）
单地址指令结构为：（10 位操作码）（6 位地址码）
因此，每少一条双地址指令，则多 2^6 条单地址指令，
双地址指令最多是 2^(16-6-6)=2^4=16 条 ，
所以单地址指令最多有（16-X）*2^6 条。

2. 13 若某机要求：三地址指令 4 条，单地址指令 255 条，零地址指令 16 条。设指令字长为 12 位．每个地址码长为 3 位。问能否以扩展操作码为其编码?如果其中单地址指令为 254 条呢?说明其理由。

解：

（1）不能用扩展码为其编码。

指令字长 12 位，每个地址码占 3 位，三地址指令最多是 2^(12-3-3-3)=8 条，现三地址指令需 4 条,

所以可有 4 条编码作为扩展码，而单地址指令最多为 4×2^3×2^3=2^8=256 条，现要求单地址指令 255 条，

所以可有一条编码作扩展码

因此零地址指令最多为 1×2^3＝8 条

不满足题目要求，故不可能以扩展码为其编码。

（2）若单地址指令 254 条，可以用扩展码为其编码。

依据（1）中推导，单地址指令中可用 2 条编码作为扩展码，零地址指令为 2×2^3＝16 条，满足题目要求

第三章

3.2 简述流水线技术的特点。

答：流水技术有以下特点：

（1） 流水线把一个处理过程分解为若干个子过程，每个子过程由一个专门的功能部件来实现。因此，流水线实际上是把一个大的处理功能部件分解为多个独立的功能部件，并依靠它们的并行工作来提高吞吐率。

（2） 流水线中各段的时间应尽可能相等，否则将引起流水线堵塞和断流。

（3） 流水线每一个功能部件的前面都要有一个缓冲寄存器，称为流水寄存器。

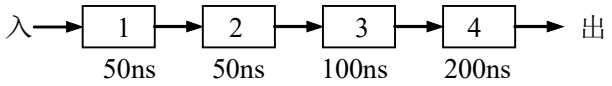
（4）流水技术适合于大量重复的时序过程，只有在输入端不断地提供任务，才能充分发挥流水线的效率。

（5） 流水线需要有通过时间和排空时间。在这两个时间段中，流水线都不是满负荷工作。

3.3 简述通过软件（编译器）来减少分支延迟的 3 种静态方法及它们的共同特点

（1）预测分支失败：沿失败的分支继续处理指令，就好象什么都没发生似的。当确定分支是失败时，
说明预测正确，流水线正常流动；当确定分支是成功时，流水线就把在分支指令之后取出的指
令转化为空操作，并按分支目标地址重新取指令执行。
（2）预测分支成功：当流水线 ID 段检测到分支指令后，一旦计算出了分支目标地址，就开始从该目
标地址取指令执行。
（3）延迟分支：主要思想是从逻辑上“延长”分支指令的执行时间。把延迟分支看成是由原来的分支
指令和若干个延迟槽构成。不管分支是否成功，都要按顺序执行延迟槽中的指令。
3 种方法的共同特点：它们对分支的处理方法在程序的执行过程中始终是不变的。它们要么总是
预测分成功，要么总是预测分支失败。

3.6 有一指令流水线如下所示



（1） 求连续输入 10 条指令，该流水线的实际吞吐率和效率；

（2） 该流水线的“瓶颈”在哪一段？请采取两种不同的措施消除此“瓶颈”。对于你所给出的两种新的流水线，连续输入 10 条指令时，其实际吞吐率和效率各是多少？

解：（1）

$$T_{pipeline} = \sum_{i=1}^{m} \Delta t_i + (n-1)\Delta t_{max}$$

$$= (50 + 50 + 100 + 200) + 9 \times 200$$

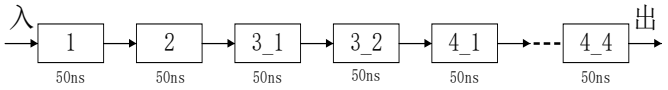
$$= 2200(ns)$$

$$TP = n/T_{pipeline} = 1/220(ns^{-1})$$

$$E = TP \cdot \frac{\sum_{i=1}^{m} \Delta t_i}{m} = TP \cdot \frac{400}{4} = \frac{5}{11} \approx 45.45\%$$

（2）瓶颈在 3、4 段。

变成八级流水线（细分）



$$T_{pipeline} = \sum_{i=1}^{m} \Delta t_i + (n-1)\Delta t_{max}$$

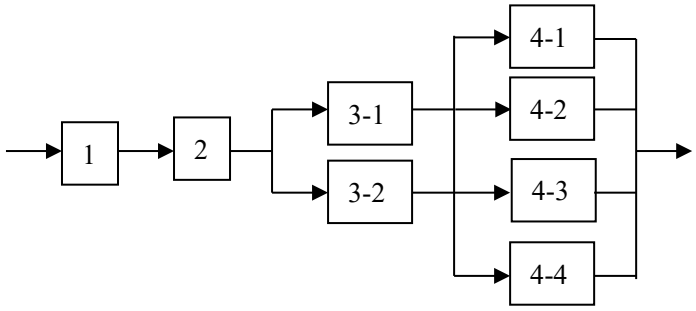
$$= 50 \times 8 + 9 \times 50$$

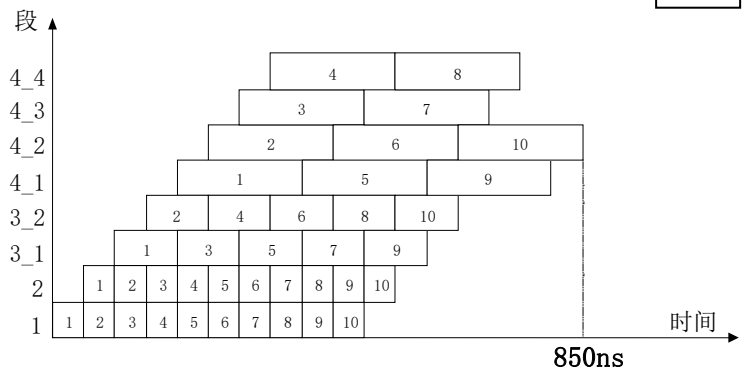
$$= 850(ns)$$

$$TP = n/T_{pipeline} = 1/85(ns^{-1})$$

$$E = TP \cdot \frac{\sum_{i=1}^{m} \Delta ti}{m} = TP \cdot \frac{400}{8} = \frac{10}{17} \approx 58.82\%$$

重复设置部件





$$TP = n/T_{pipeline} = 1/85(ns^{-1})$$

$$E = 400 \times 10/850 \times 8 = 10/17 \approx 58.82\%$$

3.7 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 Δt，问：

（1） 当在流水线的输入端连续地每 Δt 时间输入任务时，该流水线会发生什么情况？

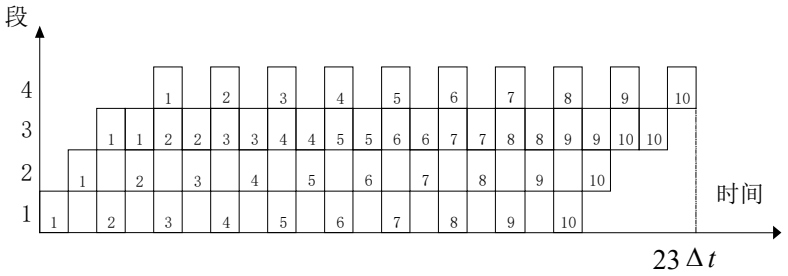
（2） 此流水线的最大吞吐率为多少？如果每 $2\Delta t$ 输入一个任务，连续处理 10 个任务时的实际吞吐率和效率是多少？

（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）



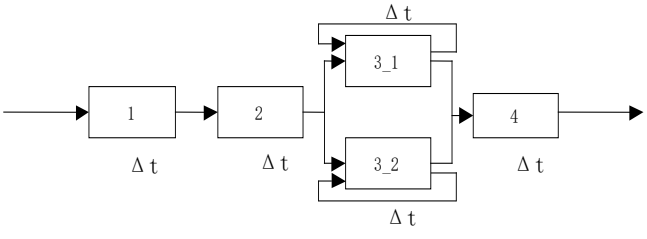
$$TP_{max} = \frac{1}{2\Delta t}$$

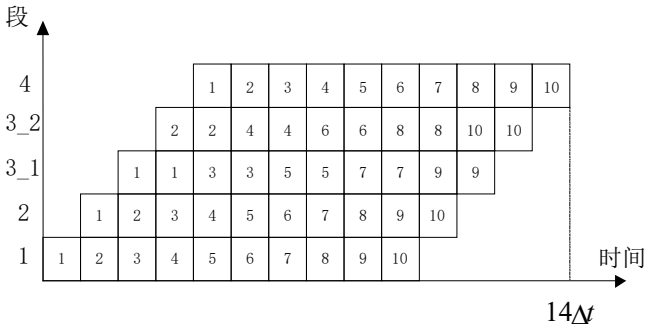
$$T_{pipeline} = 23\Delta t$$

$$Tp = {}^{n}\!/_{T_{pipeline}} = {}^{10}\!/_{23\Delta t}$$

$$\Delta E = TP \cdot {}^{5\Delta t}\!/_{4} = {}^{50}\!/_{92} \approx 54.35\%$$

（3）重复设置部件



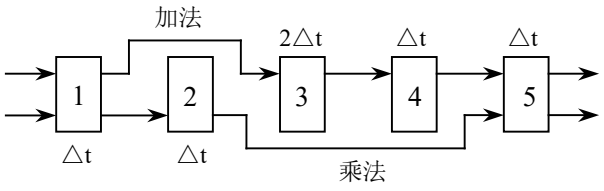


$$TP = n/T_{pipeline} = 10/14 \cdot \Delta t = 5/7 \cdot \Delta t$$

$$\text{吞吐率提高倍数} = \frac{5/7\Delta t}{10/23\Delta t} = 1.64$$

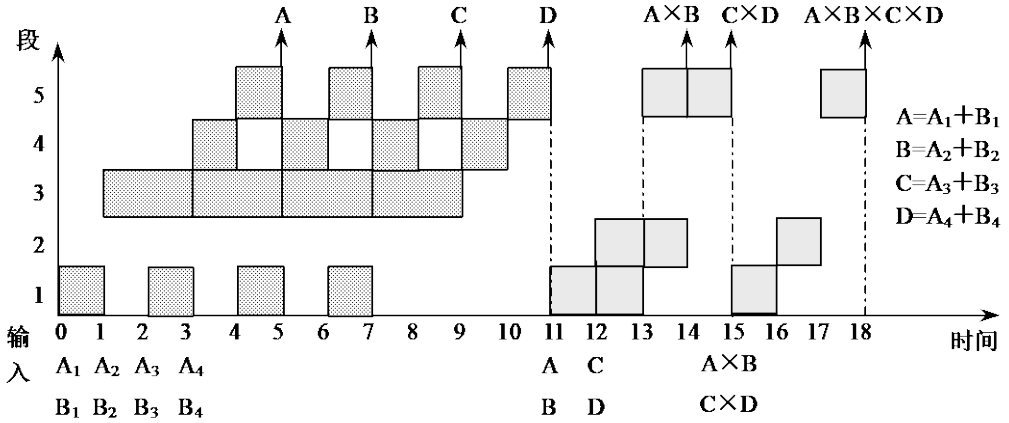
3.8 有一条静态多功能流水线由 5 段组成，加法用 1、3、4、5 段，乘法用 1、2、5 段，第 3 段的时间为 2△t，其余各段的时间均为△t，而且流水线的输出可以直接返回输入端或

暂存于相应的流水寄存器中。现要在该流水线上计算 $\prod_{i=1}^{4}(A_i + B_i)$ ，画出其时空图，并计算其吞吐率、加速比和效率。



解：首先，应选择适合于流水线工作的算法。对于本题，应先计算 A_1+B_1、A_2+B_2、A_3+B_3 和 A_4+B_4；再计算$(A_1+B_1) \times (A_2+B_2)$和$(A_3+B_3) \times (A_4+B_4)$；然后求总的结果。

其次，画出完成该计算的时空图，如图所示，图中阴影部分表示该段在工作。



由图可见，它在 18 个Δt 时间中，给出了 7 个结果。所以吞吐率为：

$$TP=\frac{7}{18\Delta t}$$

如果不用流水线，由于一次求积需 3Δt，一次求和需 5Δt，则产生上述 7 个结果共需（4×5+3×3）Δt =29Δt。所以加速比为：

$$S=\frac{29\Delta t}{18\Delta t}=1.61$$

该流水线的效率可由阴影区的面积和 5 个段总时空区的面积的比值求得：

$$E=\frac{4\times5+3\times3}{5\times18}=0.322$$

3.9 在一个 5 段的流水线处理机上需经 9Δt 才能完成一个任务，各段执行时间均为 Δt，任务处理过程对各段使用时间的预约表如下表所示。

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
S_1	√								√
S_2		√	√						
S_3				√			√	√	
S_4				√	√				
S_5						√	√		

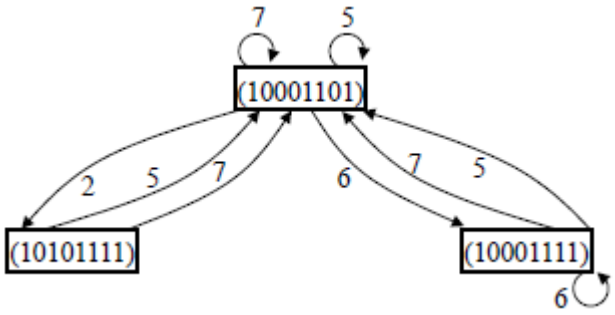
（1）画出流水线的状态有向图，并由状态图得出流水线的最优调度策略和最大吞吐率。

（2）按最优调度策略输入 6 个任务，求流水线的实际吞吐率、加速比和效率

解：1、禁止表F = (8, 4, 3, 1)，

由禁止表可得出初始冲突向量：$C0$= (10001101)，然后运算

状态有向图：



最优调度策略是(2, 5)，流水线的最大吞吐率就是最优调度策略的最大吞吐率，有TPmax=1/3.5Δ t

（2）按最优调度策略输入 6 个任务，流水线的实际吞吐率和加速比分别为：

$$TP=\frac{6}{(2+5+2+5+2+9)\Delta t}=\frac{6}{25\Delta t}$$

$$S=\frac{6\times 9\Delta t}{25\Delta t}=2.16$$

流水线的时空图如图 3.27 所示。

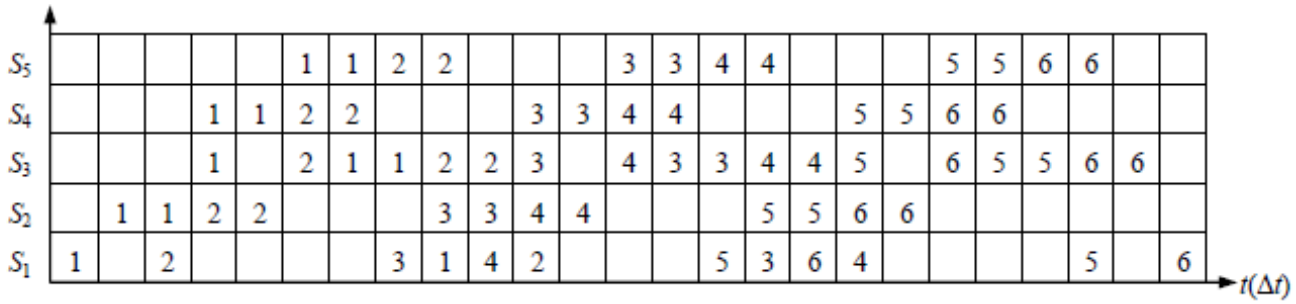


图 3.27　时空图

由图 3.27 所示时空图可见，6 个任务在流水线中确实不发生段争用冲突，6 个任务的执行时间为 25Δt。由时空图可计算出流水线的效率为：

E=6*11/5*125=0.528

3.10

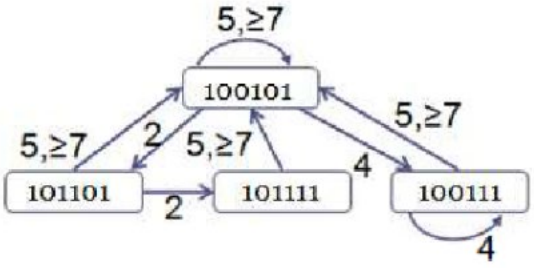
- （1）有一个5段流水线，各段执行时间均为，其预约表如下所示：

功能段 \ 时间	1	2	3	4	5	6	7
S1	√						√
S2		√			√		
S3			√	√			
S4				√			√
S5					√	√	

- （1）画出流水线任务调度的状态转移图。
- （2）分别求出允许不等时间间隔调度和等时间间隔调度的两种最优调度策略，计算这两种调度策略的流水线最大吞吐率。
- （3）若连续输入10个任务，分别求采用这两种调度策略的流水线实际吞吐率和加速比

(1) 禁止表 F = {1,3,6}　　初始冲突向量C_0 = (100101)

状态转移图如下所示



(2)

	允许不等时间间隔	等时间间隔
调度策略	(2,2,5)	(4)
最大吞吐率	1/(3△t)	1/(4△t)

(3)

	允许不等时间间隔	等时间间隔
实际吞吐率	10/(34△t)	10/(43△t)
加速比	70/34 = 2.06	70/43 = 1.63

3.11 在 MIPS 流水线上运行如下代码序列：

```
LOOP:   LW      R1，0（R2）
        DADDIU  R1，R1，#1
        SW      R1，  0（R2）
        DADDIU  R2，R2，#4
        DSUB    R4，R3，R2
        BNEZ    R4，LOOP
```

其中：R3 的初值是 R2+396。假设：在整个代码序列的运行过程中，所有的存储器访问都是命中的，并且在一个时钟周期中对同一个寄存器的读操作和写操作可以通过寄存器文件“定向”。问：

（1） 在没有任何其它定向（或旁路）硬件的支持下，请画出该指令序列执行的流水线时空图。假设采用排空流水线的策略处理分支指令，且所有的存储器访问都命中 Cache，那么执行上述循环需要多少个时钟周期？

（2） 假设该流水线有正常的定向路径，请画出该指令序列执行的流水线时空图。假设采用预测分支失败的策略处理分支指令，且所有的存储器访问都命中 Cache，那么执行上述循环需要多少个时钟周期？

（3） 假设该流水线有正常的定向路径和一个单周期延迟分支，请对该循环中的指令进行调度，你可以重新组织指令的顺序，也可以修改指令的操作数，但是注意不能增加指令的条数。请画出该指令序列执行的流水线时空图，并计算执行上述循环所需要的时钟周期数。

解：

寄存器读写可以定向，无其他旁路硬件支持。排空流水线。

```
指令          1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
LW          IF  ID  EX  M   WB
DADDIU        IF  S   S   ID  EX  M   WB
SW                    IF  S   S   ID  EX  M   WB
DADDIU                        IF  ID  EX  M   WB
DSUB                              IF  S   S   ID  EX  M   WB
BNEZ                                      IF  S   S   ID  EX  M   WB
LW                                                IF  S   S   IF  ID  EX  M   WB
```

第 i 次迭代（i＝0..98）开始周期：1＋（i×17）

总的时钟周期数：（98×17）＋18＝1684

有正常定向路径，预测分支失败。

```
指令          1   2   3   4   5   6   7   8   9  10  11   1  13  14  15
LW          IF  ID  EX  M   WB
DADDIU        IF  ID  S   EX  M   WB
SW                IF  S   ID  EX  M   WB
DADDIU                IF  ID  EX  M   WB
DSUB                      IF  ID  EX  M   WB
BNEZ                          IF  ID  EX  M   WB
LW                                IF  miss miss IF  ID  EX  M   WB
```

第 i 次迭代（i＝0..98）开始周期：1＋（i×10）

总的时钟周期数：（98×10）＋11＝991

有正常定向路径。单周期延迟分支。

```
LOOP:  LW      R1，0(R2)
       DADDIU  R2，R2，#4
       DADDIU  R1，R1，#1
       DSUB    R4，R3，R2
       BNEZ    R4，LOOP
       SW      R1，-4(R2)
```

第 i 次迭代（i ＝0..98）开始周期：1＋（i ×6 ）

总的时钟周期数：（98×6）＋10＝598

```
指令          1   2   3   4   5   6   7   8   9  10  11
LW          IF  ID  EX  M   WB
DADDIU        IF  ID  EX  M   WB
DADDIU            IF  ID  EX  M   WB
DSUB                  IF  ID  EX  M   WB
BNEZ                      IF  ID  EX  M   WB
SW                            IF  ID  EX  M   WB
LW                                IF  ID  EX  M   WB
```

3.12 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

第五章

5.2 对于正确地执行程序来说，必须保持的最关键的两个属性是?
数据流和异常行为。
保持异常行为是指：无论怎么改变指令的执行顺序，都不能改变程序中异常的发生情况。

即原来程序中是怎么发生的，改变执行顺序后还是怎么发生。

弱化为：指令执行顺序的改变不能导致程序中发生新的异常。

数据流：指数据值从其产生者指令到其消费者指令的实际流动。

5.3 记分牌算法中，记分牌中记录的信息由哪 3 部分构成?

- 指令状态表：记录正在执行的各条指令已经进入到了哪一段。
- 功能部件状态表：记录各个功能部件的状态。每个功能部件有一项，每一项由以下 9 个字段组成：
  - Busy：忙标志，指出功能部件是否忙。初值为“no”；
  - Op：该功能部件正在执行或将要执行的操作；

- Fi：目的寄存器编号；
- Fj，Fk：源寄存器编号；
- Qj，Qk：指出向源寄存器 Fj、Fk 写数据的功能部件；
- Rj，Rk：标志位，为“yes”表示 Fj，Fk 中的操作数就绪且还未被取走。否则就被置为“no”。

- 结果寄存器状态表 Result：每个寄存器在该表中有一项，用于指出哪个功能部件（编号）将把结果写入该寄存器。
  - 如果当前正在运行的指令都不以它为目的寄存器，则其相应项置为“no”。
  - 结果寄存器状态表 Result 各项的初值为“no”（全 0）

5.4 简述 Tomasulo 算法的基本思想。

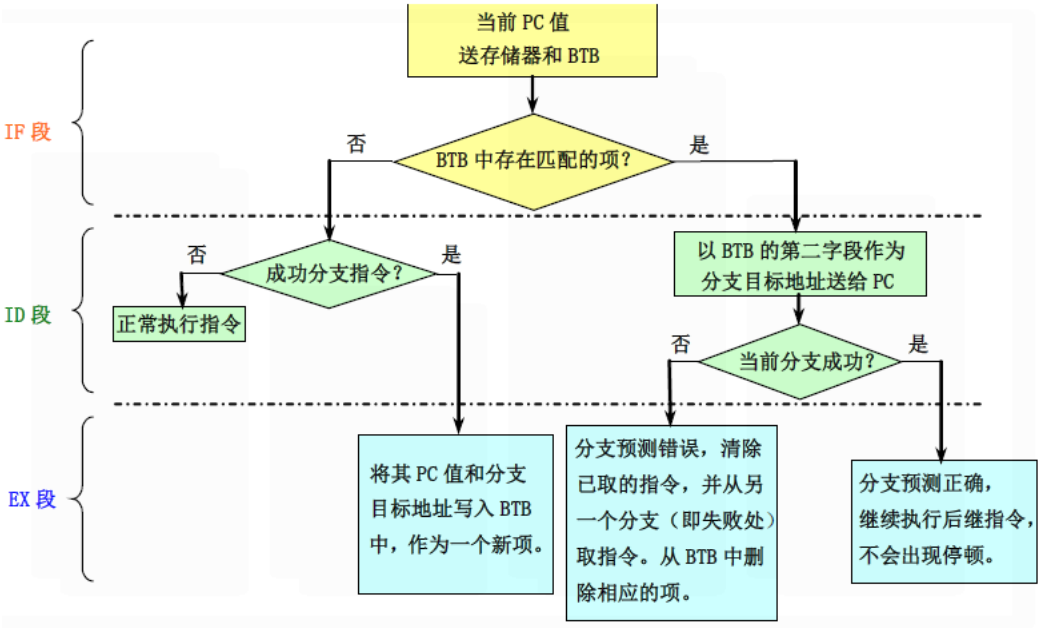
答：核心思想是：① 记录和检测指令相关，操作数一旦就绪就立即执行，把发生 RAW 冲突的可能性减小到最少；② 通过寄存器换名来消除 WAR 冲突和 WAW 冲突。寄存器换名是通过保留站来实现，它保存等待流出和正在流出指令所需要的操作数。

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）。

5.5 采用动态分支预测技术的目的是什么,需要解决哪些关键问题?

- 采用动态分支预测技术的目的
  - 预测分支是否成功
  - 尽快找到分支目标地址（或指令）（避免控制相关造成流水线停顿）
- 需要解决的关键问题
  - 如何记录分支的历史信息，要记录哪些信息?
  - 如何根据这些信息来预测分支的去向，甚至提前取出分支目标处的指令?

5.6 采用动态分支目标缓冲器 BTB 后,在流水线的三个阶段所进行的相关操作有哪些?



5.7 基于硬件的前瞻执行结合了哪 3 种思想?

- 动态分支预测。用来选择后续执行的指令。
- 在控制相关的结果尚未出来之前，前瞻地执行后续指令。
- 用动态调度对基本块的各种组合进行跨基本块的调度

第七章

7.2 简述“Cache—主存”层次与“主存—辅存”层次的区别。

答：

存储层次 比较项目	“Cache—主存”层次	“主存—辅存”层次
目的	为了弥补主存速度的不足	为了弥补主存容量的不足
存储管理的实现	全部由专用硬件实现	主要由软件实现
访问速度的比值（第一级比第二级）	几比一	几万比一
典型的块（页）大小	几十个字节	几百到几千个字节
CPU 对第二级的访问方式	可直接访问	均通过第一级

不命中时 CPU 是否切换	不切换	切换到其它进程

7.3 替换算法有哪几种,他们的优缺点是什么

（1）随机法。简单、易于用硬件实现，但这种方法没有考虑 Cache 块过去被使用的情况，反映不了程序的局部性，所以其失效率比 LRU 的高。
（2）先进先出法。容易实现。它虽然利用了同一组中各块进入 Cache 的顺序这一“历史”信息，但还是不能正确地反映程序的局部性。
（3）最近最少使用法 LRU。失效率最低。但是 LRU 比较复杂，硬件实现比较困难。

7.5 通过编译器对程序优化来改进 Cache 性能的方法有哪几种？简述其基本思想。

答：（1）数组合并。通过提高空间局部性来减少失效次数。有些程序同时用相同的索引来访问若干个数组的同一维，这些访问可能会相互干扰，导致冲突失效，可以将这些相互独立的数组合并成一个复合数组，使得一个 Cache 块中能包含全部所需元素。（2）内外循环交换。循环嵌套时，程序没有按数据在存储器中的顺序访问。只要简单地交换内外循环，就能使程序按数据在存储器中的存储顺序进行访问。（3）循环融合。有些程序含有几部分独立的程序段，它们用相同的循环访问同样的数组，对相同的数据作不同的运算。通过将它们融合成一个单一循环，能使读入 Cache 的数据被替换出去之前得到反复的使用。（4）分块。通过改进时间局部性来减少失效。分块不是对数组的整行或整列进行访问，而是对子矩阵或块进行操作。

7.8 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。

7.10 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 $_{1\text{-路}}$=2.0+1.4% *80=3.12ns

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

两路组相联的平均访问时间比较低

CPU$_{time}$=（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 $_{time\ 1\text{-}way}$=IC(2.0*2+1.2*0.014*80)＝5.344IC

CPU $_{time\ 2\text{-}way}$=IC(2.2*2+1.2*0.01*80)＝5.36IC

相对性能比：$\dfrac{\text{CPU}_{\text{time}-2\text{way}}}{\text{CPU}_{\text{time}-1\text{way}}}=5.36/5.344=1.003$

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。

7.11 在伪相联中，假设在直接映象位置没有发现匹配，而在另一个位置才找到数据（伪命中）时，不对这两个位置的数据进行交换。这时只需要 1 个额外的周期。假设失效开销为 50 个时钟周期，2KB 直接映象 Cache 的失效率为 9.8%，2 路组相联的失效率为 7.6%；128KB 直接映象 Cache 的失效率为 1.0%，2 路组相联的失效率为 0.7%。

（1） 推导出平均访存时间的公式。

（2） 利用（1）中得到的公式，对于 2KBCache 和 128KBCache，计算伪相联的平均访存时间。

解：

不管作了何种改进，失效开销相同。不管是否交换内容，在同一“伪相联”组中的两块都是用同一个索引得到的，因此失效率相同，即：失效率$_{伪相联}$＝失效率 $_{2\text{路}}$。

伪相联 cache 的命中时间等于直接映象 cache 的命中时间加上伪相联查找过

程中的命中时间*该命中所需的额外开销。

命中时间$_{伪相联}$＝命中时间$_{1路}$＋伪命中率$_{伪相联}$×1

交换或不交换内容，伪相联的命中率都是由于在第一次失效时，将地址取反，再在第二次查找带来的。

因此 伪命中率$_{伪相联}$＝命中率$_{2路}$－命中率$_{1路}$＝（1－失效率$_{2路}$）－（1－失效率$_{1路}$）

＝失效率$_{1路}$－失效率$_{2路}$。交换内容需要增加伪相联的额外开销。

平均访存时间$_{伪相联}$＝命中时间$_{1路}$＋（失效率$_{1路}$－失效率$_{2路}$）×1

＋失效率$_{2路}$×失效开销$_{1路}$

将题设中的数据带入计算，得到：

平均访存时间$_{2Kb}$=1+(0.098-0.076)*1+(0.076 *50) =4.822

平均访存时间 $_{128Kb}$=1+(0.010-0.007)*1+(0.007 *50) =1.353

显然是 128KB 的伪相联 Cache 要快一些。

7.13

题解 P154【题 7.58】某个程序共访问存储器 1 000 000 次，该程序在某个系统中运行，系统中 cache 的不命中率为 7%，其中，强制性不命中和容量不命中各占 25%，冲突不命中占 50%。问：

（1）当允许对该 cache 所做的唯一改变是提高相联度时，此时期望能够消除的最大不命中次数是多少？

（2）当允许同时提高 cache 的容量大小和相联，此时期望能够消除的最大不命中次数是多少？

解：（1）提高 cache 的相联度，可以降低冲突不命中的次数，但不会影响强制性不命中和容量不命中的次数。已知 cache 的不命中率为 7%，程序共访问存储器 1 000 000 次，所以总的不命中次数为 70 000，其中 50%为冲突不命中的次数。

因此，提高 cache 的相联度能够消除的最大不命中次数是：70 000 x 50% = 35 000 。

（2）当同时提高 cache 的容量大小和相联度时，可以消除容量不命中和冲突不命中的次数。而这两种不命中占总不命中次数的 75%，所以，能够消除的最大不命中次数是：70 000 X 75% = 52 500 。

7.14 假设一台计算机具有以下特性：

（1） 95%的访存在 Cache 中命中；

（2） 块大小为两个字，且失效时整个块被调入；

（3） CPU 发出访存请求的速率为 10^9 字/s；

（4） 25%的访存为写访问；

（5） 存储器的最大流量为 10^9 字/s（包括读和写）；

（6） 主存每次只能读或写一个字；

（7） 在任何时候，Cache 中有 30%的块被修改过；

（8） 写失效时，Cache 采用按写分配法。

现欲给该计算机增添一台外设，为此首先想知道主存的频带已用了多少。试对于以下两种情况计算主存频带的平均使用比例。

（1） 写直达 Cache；

（2） 写回法 Cache。

解：采用按写分配

（1）写直达 cache 访问命中，有两种情况：

读命中，不访问主存；

写命中，更新 cache 和主存，访问主存一次。

访问失效，有两种情况：

读失效，将主存中的块调入 cache 中，访问主存两次；

写失效，将要写的块调入 cache，访问主存两次，再将修改的数据写入 cache 和主存，访问主存一次，共三次。上述分析如下表所示。

访问命中	访问类型	频率	访存次数
Y	读	95%*75%=71.3%	0
Y	写	95%*25%=23.8%	1
N	读	5%*75%=3.8%	2
N	写	5%*25%=1.3%	3

一次访存请求最后真正的平均访存次数=(71.3%*0)+(23.8%*1)+(3.8%*2)+(1.3%*3)＝0.35

已用带宽=0.35×10^9/10^9=35.0%

（2）写回法 cache 访问命中,有两种情况：

读命中，不访问主存；

写命中，不访问主存。采用写回法，只有当修改的 cache 块被换出时，才写入主存；

访问失效,有一个块将被换出，这也有两种情况：

如果被替换的块没有修改过，将主存中的块调入 cache 块中，访问主存两次；

如果被替换的块修改过，则首先将修改的块写入主存，需要访问主存两次；然后将主存中的块调入 cache 块中，需要访问主存两次，共四次访问主存。

访问命中	块为脏	频率	访存次数
Y	N	95%*70%=66.5%	0
Y	Y	95%*30%=28.5%	0
N	N	5%*70%=3.5%	2
N	Y	5%*30%=1.5%	4

所以：

一次访存请求最后真正的平均访存次数=66.5%*0＋28.5%*0+3.5%*2+1.5%*4=0.13

已用带宽＝0.13×10^9/10^9＝13%

第八章

8.2 RAID 有哪些分级？各有何特点？

答：(1)RAID0。亦称数据分块，即把数据分布在多个盘上，实际上是非冗余阵列，无冗余信息。(2)RAID1。亦称镜像盘，使用双备份磁盘。每当数据写入一个磁盘时，将该数据也写到另一个冗余盘，这样形成信息的两份复制品。如果一个磁盘失效，系统可以到镜像盘中获得所需要的信息。镜像是最昂贵的解决方法。特点是系统可靠性很高，但效率很低。(3)RAID2。位交叉式海明编码阵列。即数据以位或字节交叉的方式存于各盘，采用海明编码。原理上比较优越，但冗余信息的开销太大，因此未被广泛应用。(4)RAID3。位交叉奇偶校验盘阵列，是单盘容错并行传输的阵列。即数据以位或字节交叉的方式存于各盘，冗余的奇偶校验信息存储在一台专用盘上。(5)RAID4。专用奇偶校验独立存取盘阵列。即数据以块(块大小可变)交叉的方式存于各盘，冗余的奇偶校验信息存在一台专用盘上。

(6)RAID5。块交叉分布式奇偶校验盘阵列，是旋转奇偶校验独立存取的阵列。即数据以块交叉的方式存于各盘，但无专用的校验盘，而是把冗余的奇偶校验信息均匀地分布在所有磁盘上。(7)RAID6。双维奇偶校验独立存取盘阵列。即数据以块(块大小可变)交叉的方式存于各盘，冗余的检、纠错信息均匀地分布在所有磁盘上。并且，每次写入数据都要访问一个数据盘和两个校验盘，可容忍双盘出错。

8.3 同步总线和异步总线各有什么优缺点？

答：(1) 同步总线。同步总线上所有设备通过统一的总线系统时钟进行同步。同步总线成本低，因为它不需要设备之间互相确定时序的逻辑。但是其缺点是总线操作必须以相同的速度运行。 (2) 异步总线。异步总线上的设备之间没有统一的系统时钟，设备自己内部定时。设备之间的信息传送用总线发送器和接收器控制。异步总线容易适应更广泛的设备类型，扩充总线时不用担心时钟时序和时钟同步问题。但在传输时，异步总线需要额外的同步开销。

8.4 试比较三种通道的优缺点及适用场合。

答：（1）字节多路通道。一种简单的共享通道，主要为多台低速或中速的外围设备服务。（2）数组多路通道。适于为高速设备服务。（3）选择通道。为多台高速外围设备（如磁盘存储器等）服务的。

8.5 在有 Cache 的计算机系统中，进行 I/O 操作时，会产生哪些数据不一致问题？如何克服？

答：（1）存储器中可能不是 CPU 产生的最新数据 ，所以 I/O 系统从存储器中取出来的是陈旧数据。

（2）I/O 系统与存储器交换数据之后，在 Cache 中，被 CPU 使用的可能就会是陈旧数据。

第一个问题可以用写直达 Cache 解决。

第二个问题操作系统可以保证 I/O 操作的数据不在 cache 中。如果不能，就作废 Cache 中相应的数据。

8.8 有 8 台外设的数据传输率如表所示，设计一种通道，Ts=2us，Td=2us。

（1）如果按字节多路通道设计，通道的最大流量是多少？若希望从 8 台外设中至少选择 4 台外设同时连接到该通道上，而且尽量多连接传输速率高的外设，那么，应选择哪些外设连接到该通道上？

（2）如果按数组多路通道设计，且通道一次传送定长数据块的大小 k＝512B，该通道的最大流量是多少？从 8 台外设中能选择哪些外设连接到该通道上？

设备号	1	2	3	4	5	6	7	8
数据传输速率（KB/s）	500	240	100	75	50	40	14	10

（知识点：通道处理机、字节多路通道、选择通道、数组多路通道）

解：

（1） $f_{maxbete}=1/(Ts+Td)=250KB/s$

根据 $f_{maxbete} \geq \sum_{i=1}^{p} f_i$ 所以应选择 3，4，5，7，8 同时链接到通道上，因为

$\sum_{i=1}^{5} f_i$ =100+75+50+14+10=249KB/s<250KB/s

(2) $f_{maxblock}$ =k/(Ts+kTd)=512/(2+512×2)=499KB/s

根据 $f_{maxblock} \geq f_i$ 因此，除外设 1 外，其他外设都可以同时连接通道上。

8.10 通道型 I/O 系统由一个字节多路通道 A(其中包括两个子通道 A1 和 A2)，两个数组多路通道 B1 和 B2 及一个选择通道 C 构成，各通道所接设备和设备的数据传送速率如表所示。

(1)分别求出各通道应具有多大设计流量才不会丢失信息；

(2)设 I/O 系统流量占主存流量的 1/2 时才算流量平衡，则主存流量应达到多少?

通道号		所接设备的数据传送速率(KB/s)
字节多路通道	子通道 A1	50 35 20 20 50 35 20 20
	子通道 A2	50 35 20 20 50 35 20 20
数组多路通道 B1		500 400 350 250
数组多路通道 B2		500 400 350 250
选择通道 C		500 400 350 250

解：

(1)要不丢失信息，各通道需要达到的流量：字节多路通道子通道 A1:0.25KB/S;字节多路通道子通道 A2:0.25KB/S;数组多路通道 B1:500KB/s;数组多路通道 B2:500KB/s;选择通道 C:500KB/s。

(2)主存流量应达到 4MB/S。

剖析:

(1)设备要求字节多路通道或其子通道的实际最大流量，是该通道所接各设备的字节传送速率之和;

设备要求数组多路通道或选择通道的实际最大流量，是该通道所接各设备的字节传送速率中的最大者。

(2)I/O 系统中，各种通道和子通道可以并行工作，因此，I/O 系统的最大流量应等于各通道最大流量之和。