计算机系统结构习题课

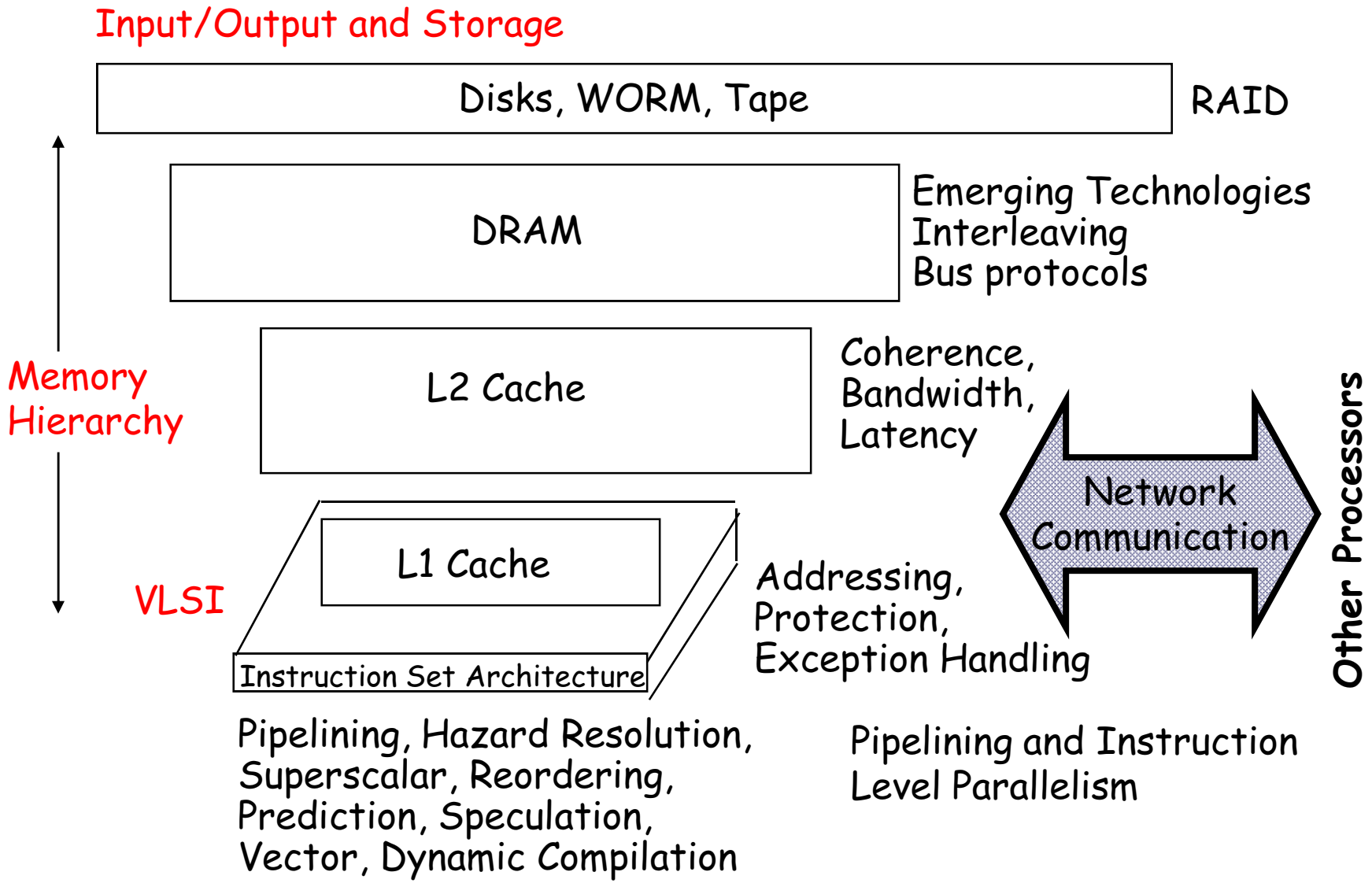
姚杰

2013/5/16

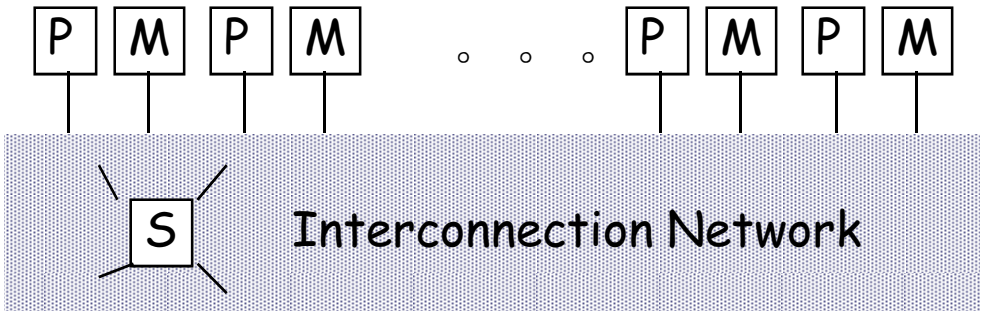
说明

- 本PPT不拷贝
- 在自己的作业本上修改、记录
- 答疑时间12周星期5（5月4日）上午1-2节东九楼A203

Computer Architecture Topics



Computer Architecture Topics



Processor-Memory-Switch

Multiprocessors
Networks and Interconnections

Shared Memory,
Message Passing,
Data Parallelism

Network Interfaces

Topologies,
Routing,
Bandwidth,
Latency,
Reliability

- 1.7 对于一台400MHz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。

$$CPI = \sum (IC_i \times CPI_i) / IC$$

$$CPI = \frac{45000 \times 1 + 75000 \times 2 + 8000 \times 4 + 1500 \times 2}{129500} = 1.776$$

$$MIPS\ 速率 = \frac{f}{CPI \times 10^6} = \frac{400 \times 10^6}{1.776 \times 10^6} = 225.225\ MIPS$$

$$程序执行时间 = \frac{(45000 \times 1 + 75000 \times 2 + 8000 \times 4 + 1500 \times 2)}{400 \times 10^6} = 575\ \mu s$$

- 1.10 计算机系统有三个部件可以改进，这三个部件的加速比如下：
  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的可改进比例为30%，那么当部件3的可改进比例为多少时，系统的加速比才可以达到10？
- (2) 如果三个部件的可改进比例为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？

$$T_e = T_o\left[(1-f_e)+\frac{f_e}{S_e}\right] \qquad S=\frac{1}{(1-f_e)+\frac{f_e}{S_e}}$$

$$S=\frac{1}{(1-\sum_i f_i)+\sum_i \frac{f_i}{S_i}}$$

$$S = \left\{ [1-(f_1+f_2+f_3)] + \frac{f_1}{S_1} + \frac{f_2}{S_2} + \frac{f_3}{S_3} \right\}^{-1}$$

$$10 = \left\{ [1-(0.3+0.3+f_3)] + \frac{0.3}{30} + \frac{0.3}{20} + \frac{f_3}{30} \right\}^{-1}$$

$$f_3 = \frac{65}{180} = 0.36$$

$$p = \frac{[1-(0.3+0.3+0.2)]T}{\frac{0.3T}{30} + \frac{0.3T}{20} + \frac{0.2T}{10} + 0.2T}$$

$$= \frac{0.2}{\frac{0.3}{30} + \frac{0.3}{20} + \frac{0.2}{10} + 0.2}$$

$$= \frac{0.2}{\frac{0.6}{60} + \frac{0.9}{60} + \frac{1.2}{60} + \frac{12}{60}}$$

$$= \frac{12}{14.7} = 0.82$$

- 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：把FPSQR操作的CPI减至3

  第二种：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。

解法1：

使用差分形式的CPI公式，不需要求原始CPI，直接比较CPI增量的大小即可

方案1： $\Delta CPI_1 = (3 - 20) \times 4\% = -0.68$

方案2： $\Delta CPI_2 = (3 - 5) \times 30\% = -0.6$

结论： 方案1导致的CPI降幅更大，性能更好

- 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：把FPSQR操作的CPI减至3

  第二种：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。

解法2：

利用原始CPI的唯一性，先使用已知条件求出原始CPI，再求出除去FPSQR指令外其他指令的平均CPI，最后比较改进后的CPI大小。

原始CPI = 5 × 30% + 1.25 × (1 - 30%) = 2.375

设除FPSQR外其余指令的平均CPI为X

则 2.375 = 20 × 4% + (1 - 4%)X ，解出X = 1.640625

方案1： $CPI_1 = 3 \times 4\% + 1.640625 \times (1 - 4\%) = 1.695$

方案2： $CPI_2 = 3 \times 30\% + 1.25 \times (1 - 30\%) = 1.775$

结论： 方案1导致的新CPI更小，性能更好

- 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：把FPSQR操作的CPI减至3

  第二种：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。

解法3：

用Amdahl公式求。记指令总条数=M，时钟周期长度=CYCLE。

原始总时间T_{old} = 0.3M × 5 × CYCLE + 0.7M × 1.25 × CYCLE = M × 2.375 × CYCLE

T_{FP} = 0.3M × 5 × CYCLE = M × 1.5 × CYCLE，所占比例为1.5/2.375 ≈ 63%

T_{FPSQR} = 0.04M × 20 × CYCLE = M × 0.8 × CYCLE，所占比例为0.8/2.375 ≈ 34%

方案1： Se = 20/3，Fe ≈ 34%，Sn_1 = 1 / [(1 - Fe) + Fe / Se] ≈ 1.4

方案2： Se = 5/3，Fe ≈ 63%，Sn_2 = 1 / [(1 - Fe) + Fe / Se] ≈ 1.3

结论： 方案1导致加速比更大，性能更好

- 2.11 某台处理机的各条指令使用频度如下所示。

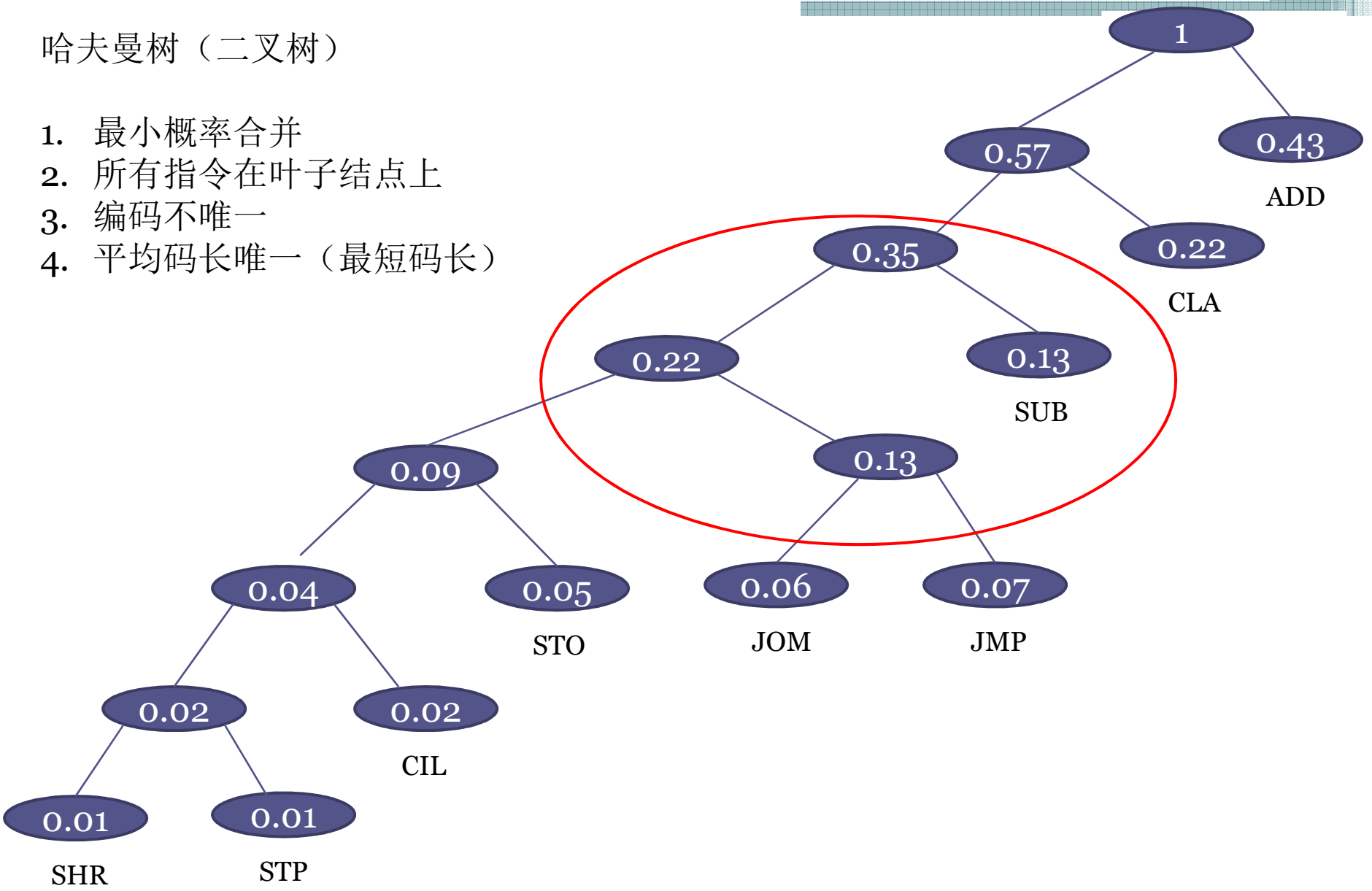
指令	使用频度	指令	使用频度	指令	使用频度
ADD	43%	JOM	6%	CIL	2%
SUB	13%	STO	5%	CLA	22%
JMP	7%	SHR	1%	STP	1%

- 请分别设计这9条指令操作码的哈夫曼编码、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。

- 可变长编码：最好的编码格式，可用最少的二进制表示目标代码。哈夫曼编码，开始主要用于电报报文，比较适合在这种串行传输环境中解码。（可写C程序对一段字符串进行字符频率统计，然后进行哈夫曼编码和解码。检验哈夫曼编码在降低目标代码长度上的效果。）
- 固定长度编码：大部分RISC指令系统采用这种编码格式，降低译码复杂度，提高译码速度。（ASCII）
- 混合型编码：提供若干种固定指令字长，既能减少目标代码长度，又能降低译码复杂度。（GB2312扩展编码）

哈夫曼树（二叉树）

1. 最小概率合并
2. 所有指令在叶子结点上
3. 编码不唯一
4. 平均码长唯一（最短码长）



指令	使用频度	哈夫曼编码1	哈夫曼编码2	3-3-3编码	2-7编码
ADD	0.43	0	0	00	00
CLA	0.22	10	100	01	01
SUB	0.13	110	101	10	1000
JMP	0.07	11100	1100	1100	1001
JOM	0.06	11101	1101	1101	1010
STO	0.05	11110	1110	1110	1011
CIL	0.02	111110	11110	1110	1011
SHR	0.01	1111110	111110	111101	1101
STP	0.01	1111111	111111	111110	1110

哈夫曼编码的平均码长为：2.42位。
3-3-3扩展编码的平均码长为：2.52位。
2-7扩展编码的平均码长为：2.70位。

- 2.14 （补充题）模拟以下MIPS程序的单条指令运行方式，在表中用16进制编码记录每一步产生的结果。

```
        .data
n:      .word 3                 ;n和x是偏移地址
x:      .double 0.5

        .text
        LD      R1, n(R0)       ;R1装入双字3(64位)
        L.D     F0, x(R0)       ;F0装入双精度浮点数0.5(64位)
        DADDI   R2, R0, 1       ; R2 ← 1
        MTC1    R2, F11         ;把通用寄存器R2中的低32位传送到浮点寄存器F11的低32位
        CVT.D.L F2, F11         ;把F11中的数据转换成双精度浮点数，送给F2。
loop:   MUL.D   F2, F2, F0      ; F2 ← F2*F0
        DADDI   R1, R1, -1      ; decrement R1 by 1
        BNEZ    R1, loop        ; if R1≠0 continue
        HALT                    ; 此条不填表
```

- 提示：MIPS浮点数的格式是IEEE754

IEEE754

$$N = m \times r_m^e$$

- 为便于软件的移植，浮点数的表示格式应该有统一标准（定义）。1985年IEEE（Institute of Electrical and Electronics Engineers）提出了IEEE754标准。该标准规定基数为2，阶码E用移码表示，尾数M用原码表示，根据原码的规格化方法，最高数字位总是1，该标准将这个1缺省存储，使得尾数表示范围比实际存储的一位。

双精度浮点数

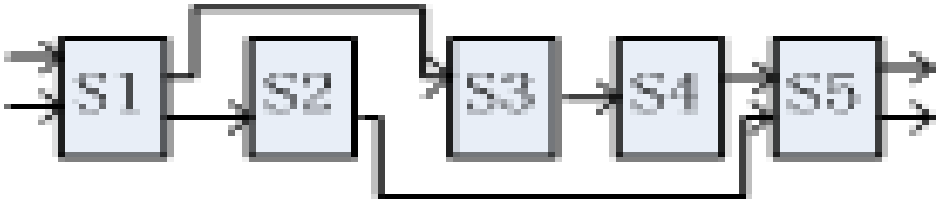
类型	数符	阶码	尾数	总位数	指数偏移
短实数	1位	8位	23位	32位	127
长实数	1位	11位	52位	64位	1023
临时实数	1位	15位	64位	80位	16383

- 0.5的二进制表示：0.1=1.0*(10)^-1
  尾数：(1).000…0
  阶码：-1+1023=0x3fe
  0x3fe0000000000000
- 1的二进制表示：1.0=1.0*(10)^0
  尾数(1).000…0
  阶码：0+1023=0x3ff
  0x3ff0000000000000

序号	结果寄存器名称	结果值（16进制）
1	R1	0000000000000003
2	F0	3fe0000000000000
3	R2	0000000000000001
4	F11	0000000000000001
5	F2	3ff0000000000000
6	F2	3fe0000000000000
7	R1	0000000000000002
序号	结果寄存器名称	结果值（16进制）
8	无	无
9	F2	3fd0000000000000
10	R1	0000000000000001
11	无	无
12	F2	3fc0000000000000
13	R1	0000000000000000
14	无	无

习题3.8

- （1）有一条动态多功能流水线由5个功能部件组成，如下：

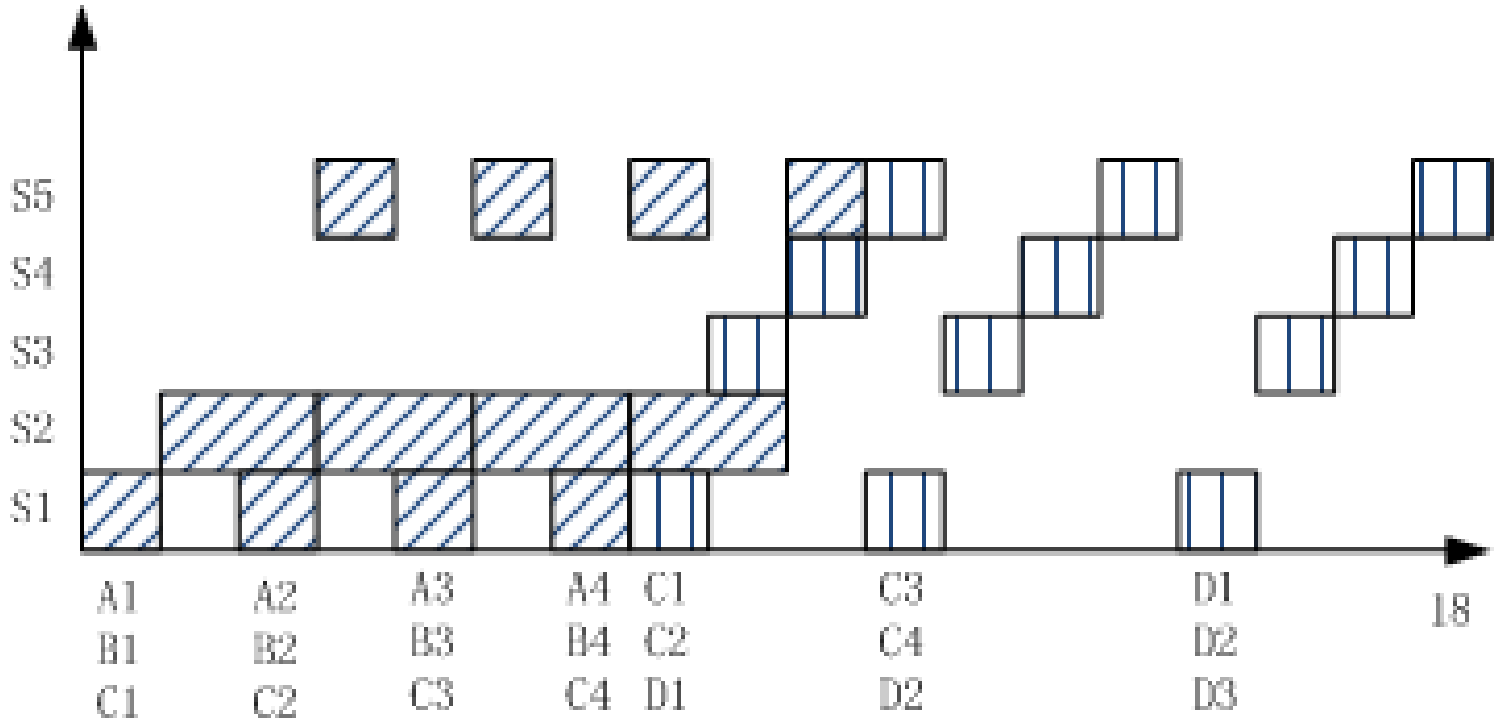


- 其中1、2、5段组成乘法流水线，1、3、4、5组成加法流水线，第二段的时间为2△t，其余各个功能段时间均为△t，假设该流水线的输出结果可以直接返回输入端，而且设置有足够的缓冲寄存器，若以最快的方式用该流水线计算：

$$\sum_{i=1}^{4}(A_i * B_i)$$

- 计算实际的吞吐率、加速比和效率。

解答：



吞吐量TP = 7 / (18 △t) = 0.39/ △t

加速比S = 28 △t /18 △t =1.56

效率E = 28/(18×5) = 0.31

习题3.10

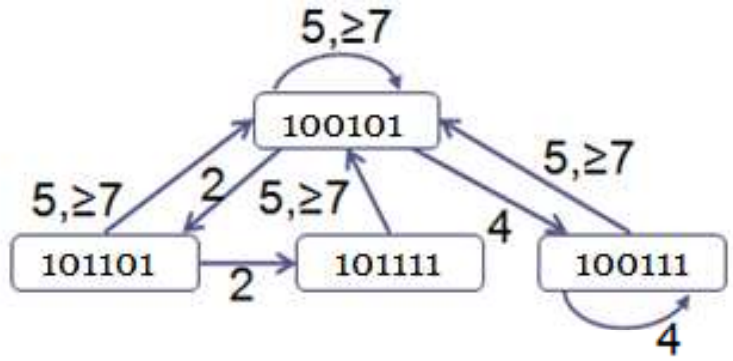
- （1）有一个5段流水线，各段执行时间均为，其预约表如下所示：

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

习题3.11

- 在改进的DLX流水线上运行如下代码序列：
- LOOP: LW R1, 0(R2)
- ADDI R1, R1, #1
- SW 0(R2), R1
- ADDI R2, R2, #4
- SUB R4, R3, R2
- BNZ R4, LOOP
- 其中，R3的初始值是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“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可以命中Cache，那么执行上述循环需要多少个时钟周期？
- （2）假设该DLX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可以命中Cache，那么执行上述循环需要多少个时钟周期？
- （3）假设该DLX流水线有正常的定向路径，请对该循环中的指令进行调度。注意可以重新组织指令的顺序，也可以修改指令的操作数，但是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需要的时钟周期数？

					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
LOOP:	LW	R1	0(R2)		IF	ID	EX	MEM	WB														
	ADDI	R1	R1	#1		IF	ID	S	S	EX	MEM	WB											
	SW	0(R2)	R1				IF	S	S	ID	S	S	EX	MEM	WB								
	ADDI	R2	R2	#4				S	S	IF	S	S	ID	EX	MEM	WB							
	SUB	R4	R3	R2							S	S	IF	ID	S	S	EX	MEM	WB				
	BNZ	R4	LOOP											IF	S	S	ID	S	S	EX	MEM	WB	
																	IF	S	S	S(IF)	IF		

需要进行396/4=99次循环，由于每次分支都清空流水线。从上图可以看出每次循环需要16(15)个时钟周期，因此总共需要的时钟周期数为99×16＋1＝1585(1486)

					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
LOOP:	LW	R1	0(R2)		IF	ID	EX	MEM	WB																
	ADDI	R1	R1	#1		IF	ID	S	EX	MEM	WB														
	SW	0(R2)	R1				IF	S	ID	EX	MEM	WB													
	ADDI	R2	R2	#4				S	IF	ID	EX	MEM	WB												
	SUB	R4	R3	R2						IF	ID	EX	MEM	WB											
	BNZ	R4	LOOP								IF	ID	EX	MEM											
	LW	R1	0(R2)									IF	miss	IF											

需要进行396/4=99次循环，由于每次分支都清空流水线。从上图可以看出每次循环需要9个时钟周期，因此总共需要的时钟周期数为99×9+1＝892

习题3.11（3）

有正常定向路径。单周期延迟分支。

Loop: lw r1,0(r2)
addi r2,r2,#4
addi r1,r1,#1
sub r4,r3,r2
bnz 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

Instruction	1	2	3	4	5	6	7	8	9	10	11
lw r1,0(r2)	IF	ID	EX	M	WB						
addi r2,r2,#4		IF	ID	EX	M	WB					
addi r1,r1,#1			IF	ID	EX	M	WB				
sub r4,r3,r2				IF	ID	EX	M	WB			
bnz r4,loop					IF	ID	EX	M	WB		
sw r1,-4(r2)						IF	ID	EX	M	WB	
lw r1,0(r2)							IF	ID	EX	M	WB

习题5.8

- 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+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。
  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）知分支目标缓冲方法执行速度快。

习题5.9

假设分支目标缓冲的命中率为90%，程序中无条件转移指令的比例为5%，没有无条件转移指令的程序的CPI值为1。假设分支目标缓冲中包含分支目标指令，允许无条件转移指令进入分支目标缓冲，则程序的CPI值为多少？假设原来的CPI=1.1

（1）原来不采用分支目标缓冲器BTB情况下

实际CPI = 理想CPI+各种停顿拍数

=1 + 5%×L + 95%×0

=1.1

解出L=2

（2）现在采用分支目标缓冲器BTB情况下

实际CPI = 理想CPI+各种停顿拍数

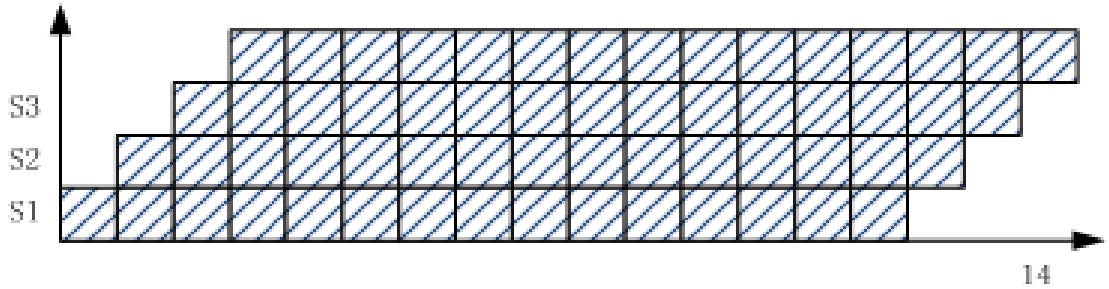
=1 + 5%×{ 90%×0+10%×2 } +95%×0

=1.01

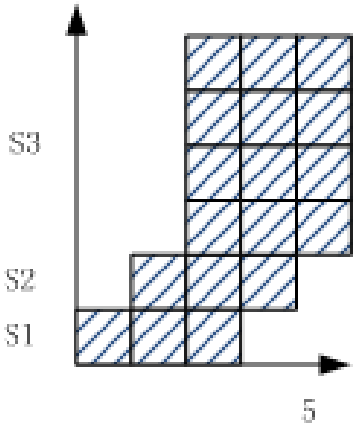
习题5.11

- 5.2 设指令流水线由取指令、分析指令、计算指令三个部分构成，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。

1. 标量流水处理机



2. 超长指令字处理机　　Tk=（k+n-1）△t=(3+12-1) △t=14 △t



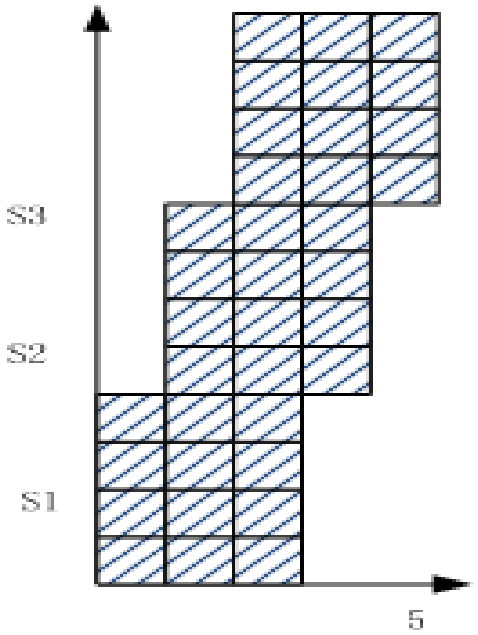
采用指令级并行技术，ILP=4,12个任务组装成3条长指令，每条含4条小指令，n=3.

Tk=（k+n-1）△t=(3+3-1) △t=5 △t

加速比S= 14 △t/5 △t=2.8

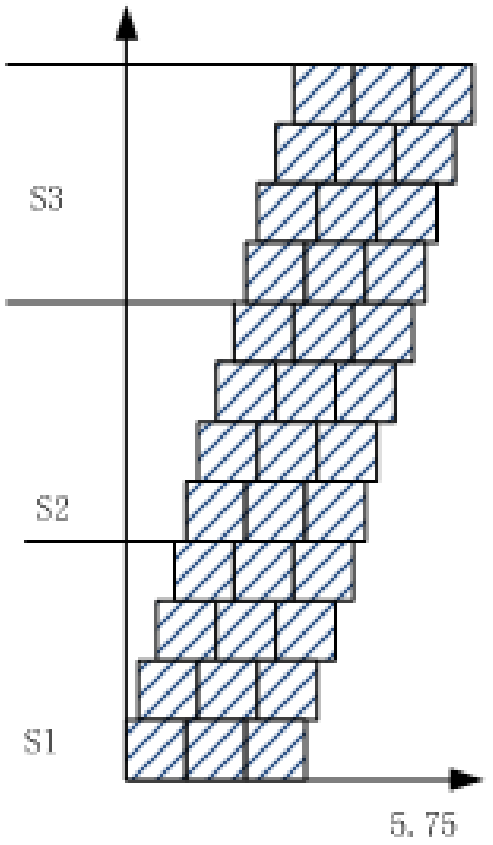
3.ILP均为4的超标量处理机

Tk=(k+n-1) △t=(3+3-1) △t =5 △t
加速比S=14 △t/5 △t=2.8



4.超流水处理机

ILP=4,12个任务在4条时钟依次错开.
025 △t的流水线上流过，等效于时钟4细分，
所以可取k=12,n=12,时钟= △t/4。
Tk=(k+n-1) △t/4=(12+12-1) △t/4=5.75 △t，
加速比S=14 △t/5 .75 △t=2.435



习题6.7

用GCD测试法判断下面的循环中是否存在循环携带的真数据相关。

```
for(i = 2; i <= 100 ; i += 2)
        a[i] = a[i-1];
```

在这个循环中a=1，b=0，c=1，d=-1，这样GCD(a,c)=1,d-b=-1，由于前者可以整除后者，但由于该循环的步长为2，而i仅和i-1有关，因此不存在循环携带的真数据相关。

习题6.8

下面这段循环完成点积运算，寄存器F2的初始值为0.试结合使用循环展开和基本指令调度技术，消除其中的所有流水线“空转”周期。假设流水线延迟如表所示，分支指令也会带来1个“空转”周期。

产生结果的指令	使用结果的指令	延迟(时钟周期数)
浮点计算	另一个浮点计算	3
浮点计算	浮点store(S.D)	2
浮点load(L.D)	浮点计算	1
浮点load(L.D)	浮点store(S.D)	0

```
loop:   L.D     F0,0(R1)
        L.D     F4,0(R2)
        MUL.D   F0,F0,F4
        ADD.D   F2,F0,F2
        DADDUI  R1,R1,#-8
        DADDUI  R2,R2,#-8
        BNE     R1,R3,loop
```

原程序周期数：
每对元素所需的时钟周期数=12，
其中空转数=5；

新程序

```
loop:   L.D  F0,16(R1)          ;F0 ←A(i+2)
        L.D F4,16(R2)           ;F4 ←B(i+2)
        L.D F6,8(R1)            ;F6 ←A(i+1)
        MUL.D  F0,F0,F4         ;F0 ←A(i+2) ×B(i+2)
        L.D F8,8(R2)            ;F8←B(i+1)
        L.D F10,0(R1)           ;F10 ←A(i)
        MUL.D  F6,F6,F8         ;F6 ←A(i+1) ×B(i+1)
        ADD.D  F2,F0,F2         ;F2 ←F2+ A(i+2) ×B(i+2)
        L.D F12,0(R2)           ;F12←B(i)
        DADDUI  R1,R1,-24       ;R1 ←R1-24
        MUL.D  F10,F10,F12      ;F10 ←A(i) ×B(i)
        ADD.D  F2,F6,F2         ;F2 ←F2+ A(i+1) ×B(i+1)
        DADDUI  R2,R2,-24       ;R2 ←R2-24
        BNE   R1,R3,loop        ;若R1 ≠R3,循环
        ADD.D  F2,F10,F2        ; F2 ←F2+ A(i) ×B(i)
```

新程序周期数：每对元素所需的时钟周期数=16/3=5.3，
其中空转数=1/3=0.3

习题7.9

- 假设在3000次访存中，第一级cache不命中110次，第二级cache不命中55次。试问：在这种情况下，该cache系统的局部不命中率和全局不命中率各是多少？

- 解：

第一级cache不命中率(全局和局部)是110/3000，即3.67%
第二级cache的局部不命中率是55/110，即50%
第二级cache的全局不命中率是55/3000，即1.83%

习题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%。

习题7.10(解答)

- 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。

习题7.11

- 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，需要1个额外的周期，而且不交换两个Cache中的数据，失效开销为50个时钟周期。试求：
- 推导出平均访存的时间公式。
- 利用（1）中得到的公式，对于2KBCache和128KBCache，重新计算伪相联的平均访存时间。请问哪一种伪相联更快？
- 假设 2KB直接映象Cache的总失效率为0.098，2路相联的总失效率为0.076；
- 128KB直接映象Cache的总失效率为0.010，2路相联的总失效率为0.007。

习题7.11(解答)

- 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。

习题7.12

假设采用理想存储器系统时的基本CPI是1.5，主存延迟是40个时钟周期；传输速率为4B/时钟周期，且cache中50%的块是修改过的。每个块中有32B，20%的指令是数据传送指令。并假设没有写缓存，在TLB不命中的情况下，需要20时钟周期，TLB不会降低cache命中率。CPU产生指令地址或cache不命中时产生的地址有0.2%没有在TLB中找到。

(1)在理想TLB情况下，计算均采用写回法16KB直接映像混合Cache、16KB两路组相联混合cache和32KB直接映像混合cache机器的实际CPI

(2)在实际TLB情况下，用(1)的结果，计算均采用写回法16KB直接映像混合Cache、16KB两路组相联混合cache和32KB直接映像混合cache机器的实际CPI

习题7.12

（1）假设TLB不命中率=0

均摊不命中开销=不命中率×{[40+32B/4B+0×20]+50%×[40+32B/4B+0×20]}

=不命中率×{48+50%×48}=不命中率×72

实际CPI1=1.5+1.2×不命中率×72=1.5+不命中率×86.4

带入3种Cache结构的不命中率得：

Cache结构	不命中率	实际CPI
16KB直接混合映像	0.029	4.0056
16KB两路混合映像	0.022	3.4008
32KB直接混合映像	0.020	3.2280

习题7.12

（2）假设TLB不命中率=0.2%

均摊不命中开销=不命中率×{[40+32B/4B+0.2%×20]+50%×[40+32B/4B+0.2%×20]}

=不命中率×{48.04+50%×48.04}=不命中率×72.06

实际CPI2=1.5+1.2×不命中率×72.06=1.5+不命中率×86.472

带入3种Cache结构的不命中率后得

Cache结构	不命中率	实际CPI
16KB直接混合映像	0.029	4.0077
16KB两路混合映像	0.022	3.4024
32KB直接混合映像	0.020	3.2294

习题7.14

- 假设一台计算机具有以下特性：
- 95%的访存在Cache中命中；
- 块大小为两个字，且失效时整个块被调入；
- CPU发出访存请求的速率为10^9字/秒；
- 25%的访存为写访问；
- 存储器的最大流量为10^9字/秒（包括读和写）；
- 主存每次只能读或写一个字；
- 在写回策略时，Cache中 有30%的块被修改过；
- 写失效时，Cache采用写分配法。
- 现欲给计算机增添一台外设，为此想先知道主存的频带已经使用了多少。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。

习题7.14(解答)

- 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%

习题7.14(解答)

- （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%

习题8.11

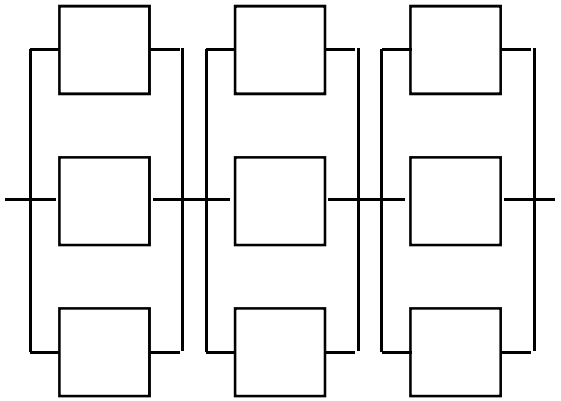
- 假设在一个计算机系统中：
- （1）每页为32KB，cache块大小为128B
- （2）对应新页的地址不在cache中，CPU不访问新页中的任何数据；
- （3）Cache中95%的被替换块将再次被读取，并引起一次不命中；
- （4）cache使用写回方法，平均60%的块被修改过；
- （5）I/O系统缓冲能够存储一个完整的cache块
- （6）访问或不命中在所有cache块中均匀分布；
- （7）在CPU和I/O之间，没有其他访问cache的干扰；
- （8）无I/O时，每100万个时钟周期内有18000次不命中；
- （9）不命中开销是40个时钟周期，如果被替换的块被修改过，则再加上30个周期用于写回主存；
- （10）假设计算机平均每200万个周期处理一页
- 试分析I/O对于性能的影响有多大

习题8.11（解）

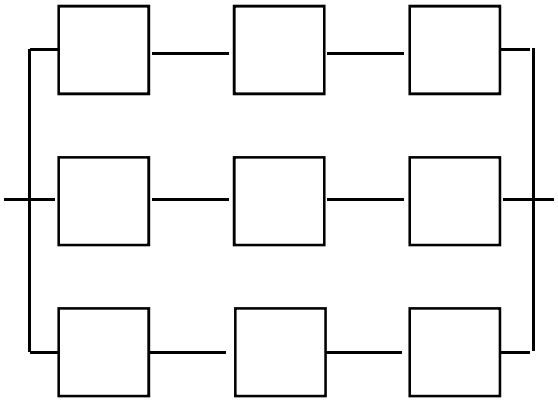
- 每个页有32KB/128=256cache块
- 因为cache和内存之间按块传输，所以I/O传输本身不产生cache缺失，但是它可能要替换cache中的有效块。如果这些被替换块中有60%是被修改过，将需要（256×60%）×30=4608个时钟周期将这些被修改过的块写回主存。
- 这些被替换出去的块中，有95%的后续需要访问，从而产生95%×256=244次不命中，将再次发生替换。由于被替换的244块中数据是从I/O直接写入cache中，因此所有块都为被修改块，需要写回主存（因为CPU不会直接访问从I/O来的新页中的数据），所以他们不会立即从主存中调入cache），需要时间是244×（40+30）=17080个时钟周期。
- 没有I/O时，每一页平均使用200万个时钟周期，cache不命中36000次，其中60%被修改过，所需的处理时间为
- （36000×40%）×40+（36000×60%）×（40+30）=2088000时钟周期
- I/O造成的额外性能损失比例为：
- （4608+17080）/（2000000+2088000）=0.53%
- 即产生大约0.53%的性能损失

8.12混联系统可靠度

假定每个部件的正常工作时间服从指数分布，它们的不可靠度都是10^{-3}，比较并串联系统(a)和串并联系统(b)的可靠性哪一个更高。



(a) 并串联系统



(b) 串并联系统

8.12混联系统可靠度(解：)

可靠度：R(t)，不可靠度：F(t)，关系式：R(t) + F(t) = 1。

在各单元相同情况下，用n代表串联倍数，m代表并联倍数。

①串并联系统：先串联、后并联

可靠度$R(t) = 1-[1-R_i^n(t)]^m$。

②并串联系统：先并联、后串联

可靠度$R(t) = [1-(1-R_i(t))^m]^n$。

公式用途：RAID 0+1是串并联系统，RAID 1+0是并串联系统

- $F= 10^{-3}$, R= 0.999

(1)并串联系统$(1-(10^{-3})^3)^3 = (0.999999999)^3 =$ 0.999999997

(2)串并联系统$1-(1-(0.999)^3)^3 = 0.99999997$

所以，(1)并串联系统可靠度更大

9.9、设函数的自变量是十进制数表示的处理机变化。现有32个处理器，编号为0、1、…、31。

(1)分别计算下列互连函数

$Cube_2(12)$ σ(8) β(9) $PM2I_{+3}(28)$ $Cube_0(\sigma(4))$ $\sigma(Cube_0(18))$

(2)用$Cube_0$和σ构成混洗交换网（每步只能用$Cube_0$和σ 一次），网络直径是多少？从5号处理机发送数据到7号处理机，最短路径要经过几步？列出经过的处理机编号

(3)采用循环移数网络构成互连网，网络直径是多少？结点度是多少？与2号处理机距离最远的是几号处理机？

解：

(1) $Cube_2(12)= Cube_2(01100B)=01000B=8$

$\sigma(8)= \sigma(01000B) = 10000B = 16$

$\beta(9) = \beta(01001B) = 11000B = 24$

$PM2I_{+3}(28) = PM2I_{+3}(11100B) = 11100B+01000B \bmod 2^5 =4$

$Cube_0(\sigma(4)) = Cube_0(\sigma(00100B)) = 01001B = 9$

$\sigma(Cube_0(18)) = \sigma(Cube_0(10010B)) = 00111B = 7$

(2) 2^5个结点的混洗交换网的直径是2n-1=2*5-1=9;

从5号处理机(00101B)发送到7号处理机(00111B)最短路径经过6步

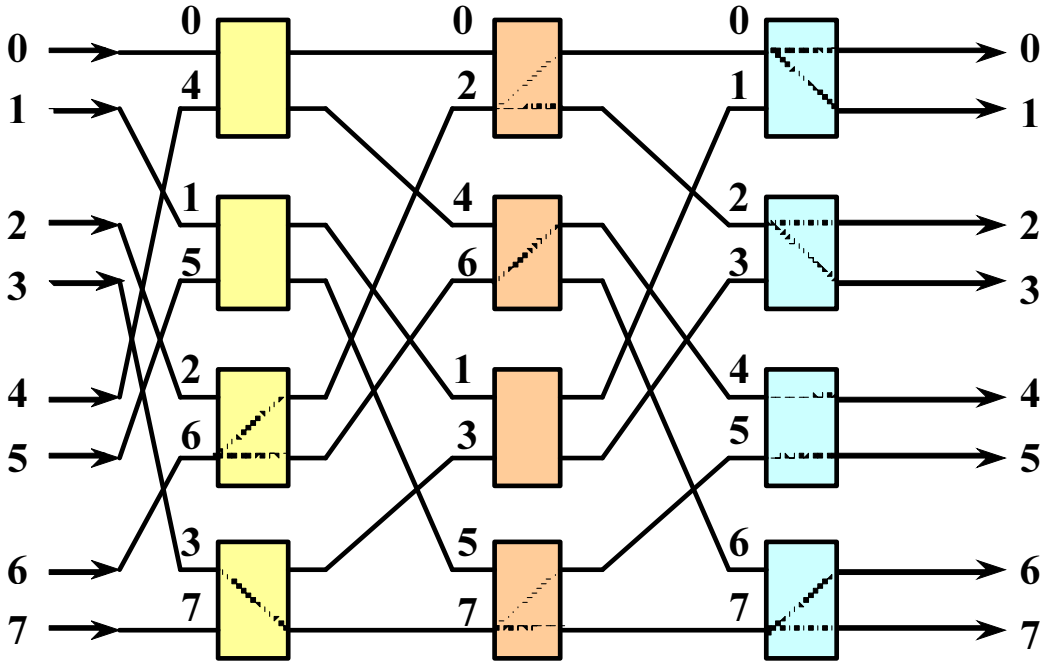
包括5步左移和1步求反：

00101B -> 01010B -> 10100B -> 01001B -> 10010B -> 10011B -> 00111B

(3) 网络直径是3；结点度是2n-1 = 2*5-1 = 9；12或16，其他对称节点也是。

习题9.13

- 用一个N=8的三级Omega网络连接8个处理机（P0～P7），8个处理机的输出端分别依次连接Omega的8个输入端0～7，8个处理机的输入端分别依次连接Omega的8个输出端0～7，如果处理机P6要把数据播送到处理机P0～P4，处理机P3要把数据播送到处理机P5～P7，那么，Omega网络能否同时为它们的播送要求实现连接，画出实现播送的Omega网络的开关状态图。



习题10.6

- 一个具有32台处理机的系统，对远程存储器访问时间是2000ns。除了通信以外，假设计算中的访问均命中局部存储器。当发出一个远程请求时，本地处理机挂起。处理机的时钟周期是10ns，假设指令基本的CPI为1.0（假设所有访存均命中cache）。对于下述两种情况：
- （1）没有远程访问；
- （2）0.5%的指令需要远程访问
- 试问前者比后者快多少？

解：远程访存时钟周期为2000/10=200

则 有远程访问情况

$CPI_2=CPI_1+p\times C=2$

因此前者比后者快2倍

习题10.9

- 采用排队锁和fetch-and-increment重新实现栅栏同步，并将它们分别与采用旋转锁实现的栅栏同步进行性能对比。

- 解：fetch-and-increment(count);
- if(count=total){ //进程全部到达
- count = 0; //重置计数器
- release=1; //释放进程
- }
- else{ //还有进程未到达
- spin(release=1);//等待信号
- }
- 当有N个处理器时，上述代码执行fetch-and-increment操作N次，当访问释放操作的时候，有N个cache未命中。当最后一个处理器到达栅栏条件后，release补置为“1”，此时有N-1个cache未命中（对于最后一个到达栅栏的处理器，当它读release的时候，将在主存中命中）。所以，共有 3N-1次总线传输操作。如果有10个处理器，则共有29次总线传输操作，总共需要2900个时钟周期。

课程结束...