第四次作业习题课

第一题

3.2 [10] <1.8、3.1、3.2>思考一下延迟数目到底意味着什么——它们表示一个给定函数生成其输出结果所需要的时钟周期数，没有别的意思。如果整个流水线在每个功能单元的延迟周期中停顿，那么至少要保证任何一对“背靠背”指令（生成结果的指令后面紧跟着使用结果的指令）正确执行。但并非所有指令对具有这种“生成者/使用者”的关系。有时，两条相邻指令之间没有任何关系。如果流水线检测到真正的数据相关，而且只会因为这些真数据相关而停顿，而不会仅仅因为有某个功能单元繁忙就盲目停顿，那表 3-22 代码序列中的循环体需要多少个时钟周期？在代码中需要容纳所述延迟的时候插入<stall>。（提示：延迟为+2 的指令需要在代码序列中插入两个<stall>时钟周期。可以这样来考虑：一条需要一个时钟周期的指令的延迟为 1+0，也就是不需要额外的等待状态。那么延迟 1+1 就意味着 1 个停顿周期，延迟 1+N有 N个额外停顿周期。

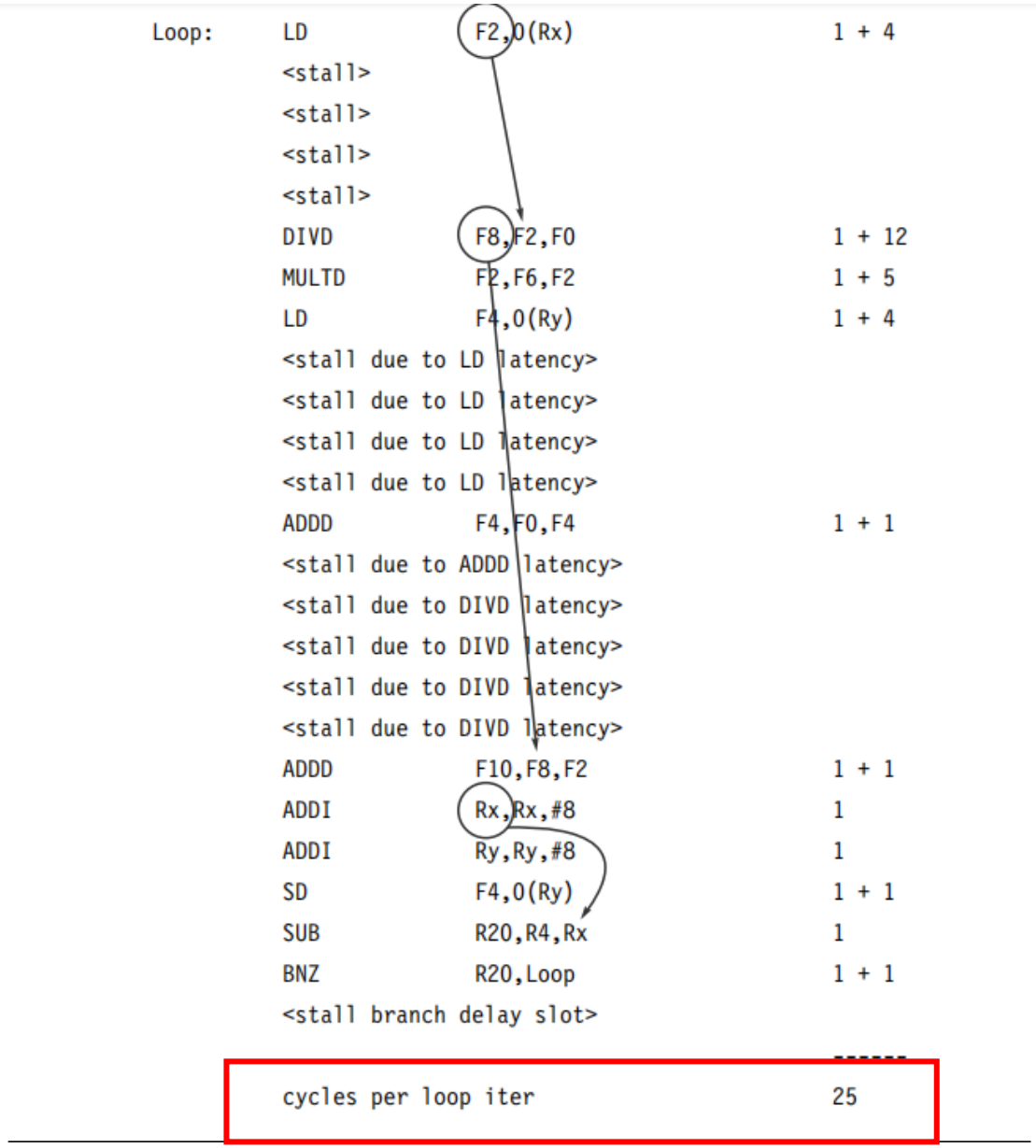
表3-22 练习3.1至练习3.6的代码与延迟

			超过一个时钟周期的延迟	
Loop:	LD	F2,0(RX)	存储器LD	+4
I0:	DIVD	F8,F2,F0	存储器SD	+1
I1:	MULTD	F2,F6,F2	整数ADD，SUB	+0
I2:	LD	F4,0(Ry)	分支	+1
I3:	ADDD	F4,F0,F4	ADDD	+1
I4:	ADDD	F10,F8,F2	MULTD	+5
I5:	ADDI	Rx,Rx,#8	DIVD	+12
I6:	ADDI	Ry,Ry,#8		
I7:	SD	F4,0(Ry)		
I8:	SUB	R20,R4,Rx		
I9:	BNZ	R20,Loop		

解题思路：

- 找出代码中的所有真数据相关
- 若需要停顿，则插入一定数目的stall

第一题



```
Loop:   LD          F2,0(Rx)                1 + 4
        <stall>
        <stall>
        <stall>
        <stall>
        DIVD        F8,F2,F0                1 + 12
        MULTD       F2,F6,F2                1 + 5
        LD          F4,0(Ry)                1 + 4
        <stall due to LD latency>
        <stall due to LD latency>
        <stall due to LD latency>
        <stall due to LD latency>
        ADDD        F4,F0,F4                1 + 1
        <stall due to ADDD latency>
        <stall due to DIVD latency>
        <stall due to DIVD latency>
        <stall due to DIVD latency>
        <stall due to DIVD latency>
        ADDD        F10,F8,F2               1 + 1
        ADDI        Rx,Rx,#8                1
        ADDI        Ry,Ry,#8                1
        SD          F4,0(Ry)                1 + 1
        SUB         R20,R4,Rx               1
        BNZ         R20,Loop                1 + 1
        <stall branch delay slot>
                                            ------
        cycles per loop iter                25
```

存在真数据相关的指令

LD F2,0(Rx) -> DIVD F8,F2,F0 +4

DIVD F8,F2,F0 -> ADDD F10,F8,F2 +12

MULTD F2,F6,F2 -> ADD,F10,F8,F2 +5

LD F4,0(Ry) -> ADDD F4,F0,F4 +4

ADDI Rx,Rx,#8-> SUB R20,R4,Rx +0 X

ADDI Ry,Ry,#8 -> SD F4,0(Ry) +0 X

SUB R20,R4,Rx ->BNZ R20,Loop +0 X

第二题

3.14 [25/25/25] <3.2、3.7>在这个练习中，我们研究如何利用软件技术从一个常见的向量循环中提取指令级并行（ILP）。下面的循环是所谓的 DAXPY 循环（双精度 a*X* 加 *Y*），它是高斯消元法的核心运算。下面的代码实现 DAXPY 运算 *Y*=a*X*+*Y*，向量长度为 100。最初，R1 被设置为数组 *X* 的基地址，R2 被设置为 *Y* 的基地址：

```
      DADDIU   R4,R1,#800  ; R1 = upper bound for X
foo:  L.D      F2,0(R1)    ; (F2) = X(i)
      MUL.D    F4,F2,F0    ; (F4) = a*X(i)
      L.D      F6,0(R2)    ; (F6) = Y(i)
      ADD.D    F6,F4,F6    ; (F6) = a*X(i) + Y(i)
      S.D      F6,0(R2)    ; Y(i) = a*X(i) + Y(i)
      DADDIU   R1,R1,#8    ; increment X index
      DADDIU   R2,R2,#8    ; increment Y index
      DSLTU    R3,R1,R4    ; test: continue loop?
      BNEZ     R3,foo      ; loop if needed
```

假定功能单元的延迟如下表所示。假定在 ID 阶段解决一个延迟为 1 周期的分支。假定结果被完全旁路。

产生结果的指令	使用结果的指令	延迟（单位：时钟周期）
浮点乘	浮点ALU运算	6
浮点加	浮点ALU运算	4
浮点乘	浮点存储	5
浮点加	浮点存储	4
整数运算和所有载入	任何指令	2

第二题

a.假定一个单发射流水线。说明在编译器未进行调度以及对浮点运算和分支延迟进行调度之后，该循环是什么样的，包括所有停顿或空闲时钟周期。在未调度和已调度情况下，结果向量Y中每个元素的执行时间为多少个时钟?为使处理器硬件独自匹配调度编译器所实现的性能改进，时钟频率应当为多少?(忽略加快时钟速度会对存储器系统性能产生的影响)

Clock cycle	Unscheduled code		Scheduled code	
1	DADDIU	R4,R1,#800	DADDIU	R4,R1,#800
2	L.D	F2,0(R1)	L.D	F2,0(R1)
3	stall		L.D	F6,0(R2)
4	MUL.D	F4,F2,F0	MUL.D	F4,F2,F0
5	L.D	F6,0(R2)	DADDIU	R1,R1,#8
6	stall		DADDIU	R2,R2,#8
	stall		DSLTU	R3,R1,R4
	stall		stall	
	stall		stall	
7	ADD.D	F6,F4,F6	ADD.D	F6,F4,F6
8	stall		stall	
9	stall		stall	
10	stall		BNEZ	R3,foo
11	S.D	F6,0(R2)	S.D	F6,-8(R2)
12	DADDIU	R1,R1,#8		
13	DADDIU	R2,R2,#8		
14	DSLTU	R3,R1,R4		
15	stall			
16	BNEZ	R3,foo		
17	stall			

产生结果的指令	使用结果的指令	延迟（单位：时钟周期）
浮点乘	浮点ALU运算	6
浮点加	浮点ALU运算	4
浮点乘	浮点存储	5
浮点加	浮点存储	4
整数运算和所有载入	任何指令	2

- ✓ 结果向量Y中每个元素的执行分别需要16和10个时钟(DADDIU R4,R1,#800不属于循环内语句)
- ✓ 编译调度后快了1.6倍，为了保持性能相同，则时钟频率应该增大为原来的1.6倍才行

第二题

b.假定一个单发射流水线。根据需要对循环进行任意次展开，使调度中不存在任何停顿，消除循环开销指令。必须将此循环展开多少次?给出指令调度。结果中每个元素的执行时间为多少?

- ✓ 需要展开3次才能使调度中不存在任何停顿
- ✓ 每个元素的执行时间为19/3个时钟周期

Clock cycle	Scheduled code	
1	DADDIU	R4,R1,#800
2	L.D	F2,0(R1)
3	L.D	F6,0(R2)
4	MUL.D	F4,F2,F0

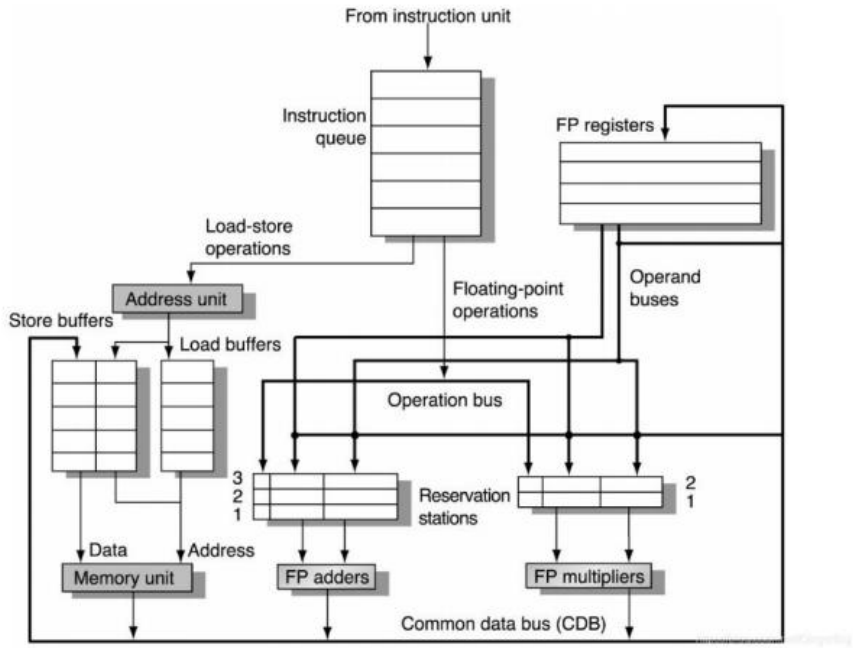
Figure S.19 The code must be unrolled three times to eliminate stalls after scheduling.

5	L.D	F2,8(R1)
6	L.D	F10,8(R2)
7	MUL.D	F8,F2,F0
8	L.D	F2,8(R1)
9	L.D	F14,8(R2)
10	MUL.D	F12,F2,F0
11	ADD.D	F6,F4,F6
12	DADDIU	R1,R1,#24
13	ADD.D	F10,F8,F10
14	DADDIU	R2,R2,#24
15	DSLTU	R3,R1,R4
16	ADD.D	F14,F12,F14
17	S.D	F6,-24(R2)
18	S.D	F10,-16(R2)
19	BNEZ	R3,foo
20	S.D	F14,-8(R2)

8改成16

Figure S.19 *Continued*

第三题



如图所示,假设浮点加法执行需要 2 个周期,浮点乘法需要 3 个周期,功能单元完全流水化。采用 Tomasulo 算法运行下列指令,写出第 7 个周期 Reservation Stations 和 Register result status 的状态。

ADD.D F4,F0,F8

MULT.D F2,F0,F4

ADD.D F4,F4,F8

MULT.D F8,F4,F2

第三题

Clk1

指令	流出	执行	写结果
ADD.D F4,F0,F8	1		
MULT.D F2,F0,F4			
ADD.D F4,F4,F8			
MULT.D F8,F4,F2			

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	yes	ADD.D	R[F0]	R[F8]		
Add2						
Add3						
Mult1						
Mult2						

第一条加法指令发射,F4的**Qi=Add1**

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi			Add1			
Value						

第三题

Clk2

指令	流出	执行	写结果
ADD.D F4,F0,F8	1	2~	
MULT.D F2,F0,F4	2		
ADD.D F4,F4,F8			
MULT.D F8,F4,F2			

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	yes	ADD.D	R[F0]	R[F8]		
Add2						
Add3						
Mult1	yes	MULT.D	R[F0]			Add1
Mult2						

第一条加法指令开始执行
第二条乘法指令发射，由于F4由上一条指令产生，所以**Qk=Add1**,因为要写F2，所以F2的**Qi=Mult1**

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi		Mult1	Add1			
Value						

第三题

Clk3

指令	流出	执行	写结果
ADD.D F4,F0,F8	1	2~3	
MULT.D F2,F0,F4	2		
ADD.D F4,F4,F8	3		
MULT.D F8,F4,F2			

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	yes	ADD.D	R[F0]	R[F8]		
Add2	yes	ADD.D		R[F8]	Add1	
Add3						
Mult1	yes	MULT.D	R[F0]			Add1
Mult2						

第一条加法指令继续执行
第二条乘法指令阻塞
第三条加法指令需要等待第一条指令的F4，所以**Qj=Add1**,因为要写F4，所以F4的**Qi=Add2**

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi		Mult1	Add2			
Value						

第三题

Clk4

指令	流出	执行	写结果
ADD.D F4,F0,F8	1	2~3	4
MULT.D F2,F0,F4	2		
ADD.D F4,F4,F8	3		
MULT.D F8,F4,F2	4		

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	no					
Add2	yes	ADD.D	M1	R[F8]		
Add3						
Mult1	yes	MULT.D	R[F0]	M1		
Mult2	yes	MULT.D			Add2	Mult1

第一条加法指令完成，所有等待F4的指令都得到结果**M1**
第四条乘法指令发射，由于等待F4和F2，所以**Qj=Add2**,**Qk=Mult1**，因为要写F8，所以F8的**Qi=Mult2**

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi		Mult1	Add2		Mult2	
Value						

第三题

Clk5

指令	流出	执行	写结果
ADD.D F4,F0,F8	1	2~3	4
MULT.D F2,F0,F4	2	5~	
ADD.D F4,F4,F8	3	5~	
MULT.D F8,F4,F2	4		

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	no					
Add2	yes	ADD.D	M1	R[F8]		
Add3						
Mult1	yes	MULT.D	R[F0]	M1		
Mult2	yes	MULT.D			Add2	Mult1

第二、三条指令开始执行
第四条指令阻塞

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi		Mult1	Add2		Mult2	
Value						

第三题

Clk6

指令	流出	执行	写结果
ADD.D F4,F0,F8	1	2~3	4
MULT.D F2,F0,F4	2	5~	
ADD.D F4,F4,F8	3	5~6	
MULT.D F8,F4,F2	4		

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	no					
Add2	yes	ADD.D	M1	R[F8]		
Add3						
Mult1	yes	MULT.D	R[F0]	M1		
Mult2	yes	MULT.D			Add2	Mult1

第二、三条指令继续执行
第四条指令继续阻塞

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi		Mult1	Add2		Mult2	
Value						

第三题

Clk7

指令	流出	执行	写结果
ADD.D F4,F0,F8	1	2~3	4
MULT.D F2,F0,F4	2	5~7	
ADD.D F4,F4,F8	3	5~6	7
MULT.D F8,F4,F2	4		

Reservation stations

名称	busy	op	Vj	Vk	Qj	Qk
Add1	no					
Add2	no					
Add3						
Mult1	yes	MULT.D	R[F0]	M1		
Mult2	yes	MULT.D	M2			Mult1

第三条指令写结果M2，等待F4的第四条指令**Vj=M2**,F4的**Value=M2**

Register result status

Register	F0	F2	F4	F6	F8	F10
Qi		Mult1			Mult2	
Value			M2			

第四题

3.19 [10/5] <3.9>考虑分支目标缓冲区，正确条件分支预测、错误预测和缓存缺失的代价分别为 0、2 和 2 个时钟周期。考虑一种区分条件与无条件分支的分支目标缓冲区设计，而条件分支存储目标地址，对于无条件分支则存储目标指令。

a. [10] <3.9>当缓冲区中发现无条件分支时，代价为多少个时钟周期？

b. [10] <3.9>判断对于无条件分支进行分支折合所获得的改进。假定命中率为 90%，无条件分支频率为 5%，缓冲区缺失的代价为两个时钟周期。这样可以获得多少改进？对于这一改进来说，必须达到多高的命中率才能提供性能增益？

a.存储无条件分支的目标指令相当于跳过了这条无条件分支指令。如果在指令获取中有一个BTB命中，并且目标指令是可用的，那么该指令将直接进入译码阶段，以代替分支指令。因此代价是-1个周期。

b.如果BTB只存储无条件分支的目标地址，则CPI的改进为5% × （90% × 0+10% × 2）=0.01。如果BTB改为存储目标指令，则CPI项将变成5% × （90%×(-1)+10%×2）=-0.035，负号表示它降低了总体CPI值。
设命中率为h，那么若需要提供性能增益，则必须有

$$5\% \times (h \times (-1) + (1-h) \times 2) <= 0$$

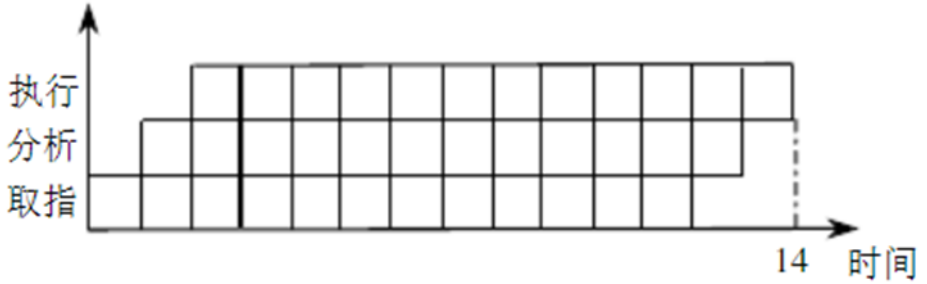
解得h>=2/3

第五题

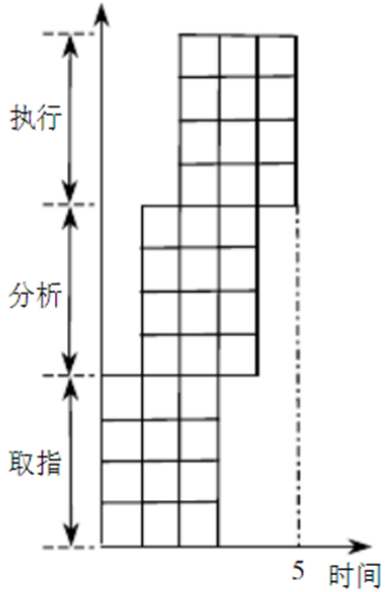
设指令流水线由取指令、分析指令和执行指令 3 个部件构成，每个部件经过的时间为Δ*t*,连续流入 12 条指令。分别画出标量流水处理机以及 ILP 均为 4 的超标量处理机、超长指令字处理机的时空图，并分别计算它们相对于标量流水处理机的加速比。

- 标量流水处理机，T1=14Δt
- 超标量处理机，T2=5Δt
- 超长指令字处理机，T3=5Δt

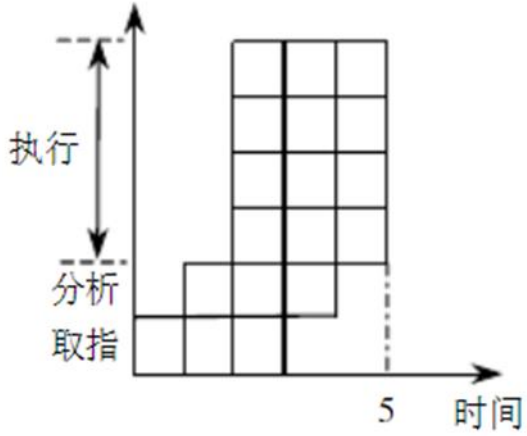
✓ 超标量处理机相对于标量流水处理机的加速比为14/5=2.8
✓ 超长指令字处理机相对于标量流水处理机的加速比为14/5=2.8



标量流水处理机时空图



超标量处理机时空图



超长指令字处理机时空图