**4.2图4-2中基本的单周期MIPS实现仅能实现某些指令。可以在这个指令集中加入新的指令，但决定是否加入取决于给处理器的数据通路和控制通路增加的成本和复杂度。对下面的新指令而言，试回答下列3个问题:**

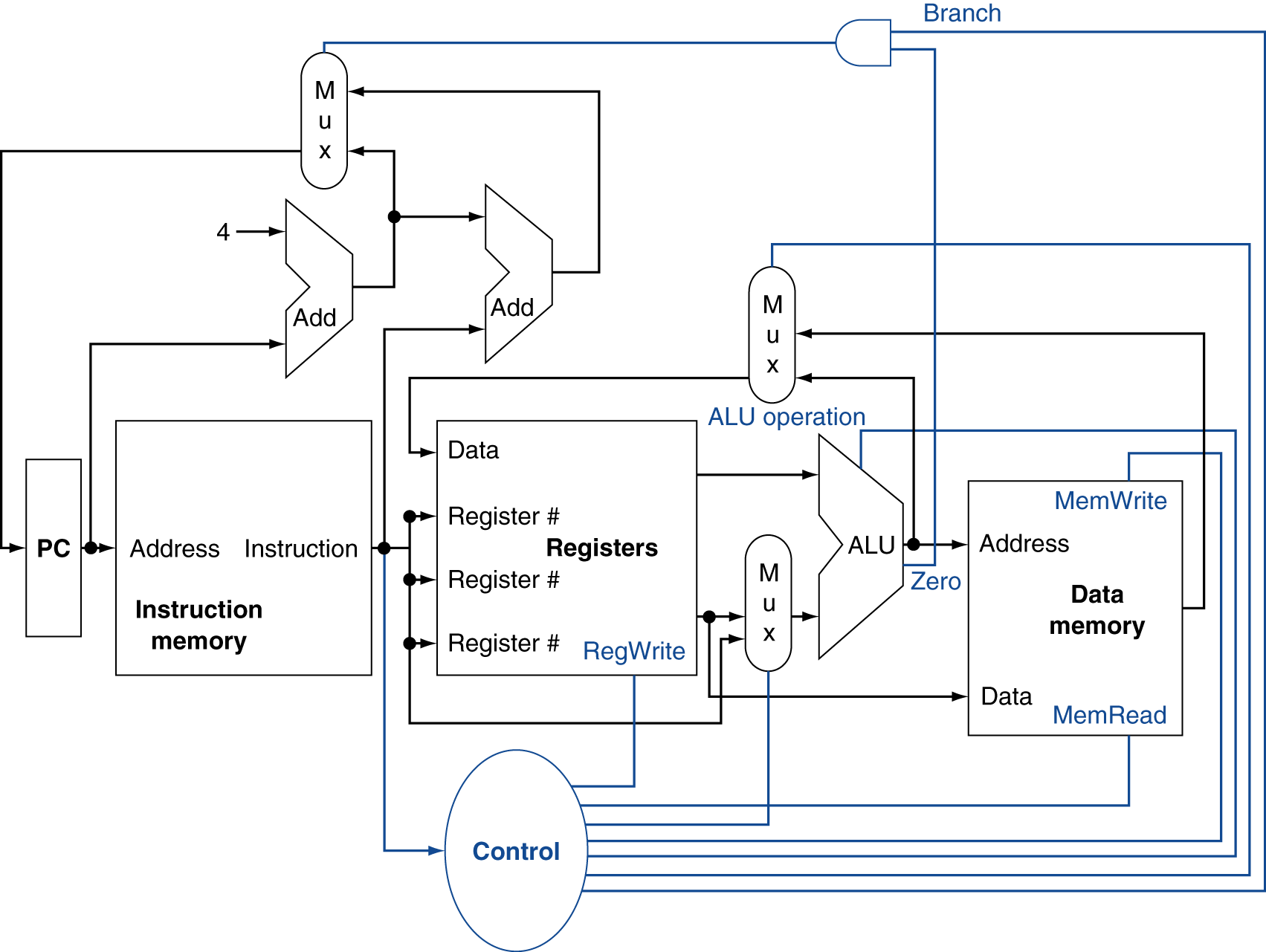
**指令: LWI Rd,Rt (Rs)**

**解释: Reg[Rd] = Mem[ Reg[Rt] +Reg[Rs]]**

**4.2.1 [10]<4.1>对上述指令而言，哪些已有的单元还可以被使用?**

**4.2.2 [10]<4.1>对 上述指令而言，还需要增加哪些功能单元?**

**4.2.3 [10]<4.1>为了支持这些指令，需要在控制单元增加哪些信号?**

****

解：**指令: LWI Rd,Rt(Rs)**

**解释: Reg[Rd] = Mem[ Reg[Rt] +Reg[Rs]]**

分析，该指令是R型指令

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Op 6 | Rs 5 | Rt 5 | Rd 5 | 0 5 | Func6 |

（1）pc，指令存储器；寄存器的读端口，控制单元；算数逻辑单元；数据存储器；寄存器的读写端口。

（2）不需要。这个指令可以使用现有的模块来实现。

（3）不需要。这个指令不用添加新的控制信号就可以实现，只需要控制逻辑的变化。

**4.3 当处理器设计者考虑改 进处理器数据通路时，往往要考虑性能与成本的折中。假设我们从图4-2的数据通路出发，其中指令存储器、加法器、多选器、ALU、寄存器堆、数据存储器和控制单元的延迟分别为400ps、100ps、30ps、 120ps 、200ps、 350ps和100ps,相应的成本分别为1000、30、10、100、200、2000和500。**

**考虑给ALU增加一个乘法，这将使ALU的延时增加300ps,同时ALU的成本增加600。这样做的结果是需要执行的指令减少了5%，主要是由于不再需要模拟MUL指令。**

**4.3.1 [10] <4.1>改进前后的时钟周期分别是多少?**

**4.3.2 [10] <4.1>改进后将获得多大的加速比?**

**4.3.3 [10] <4.1>比较改进前后的性价比。**

解：

（1）最长的访存指令的关键路径：指令存储器、寄存器堆、多选器、ALU、访存、多选器、寄存器堆

改进前的时钟周期=400+200+30+120+350+30+200=1330ps （注：回写寄存器堆未考虑不扣分，即改进前的时钟周期=400+200+30+120+350+30=1130ps）

改进后的时钟周期=1330+300=1630ps（注：回写寄存器堆未考虑不扣分，即1130ps+300=1430）

（2）加速比=改进前的CPU执行时间/改进后的CPU执行时间

=CPI\*IC\*1330/CPI\*(0.95\*IC)\*1630 =0.85

（或若未考虑回写寄存器0.83）

（3）改进前的成本=1000+30\*2+10\*3+100+200+2000+500=3890

改进后的成本=3890+600=4490

改进前后的成本=4490/3890=1.15

成本比/加速比=1.15/0.85=1.35（或未考虑回写为1.39）

结论：花费大于收益。

**4.4 本练习题中的题目假定在实现一个处理器的数据通路时，逻辑模块的延时如下**

****

**4.4.1[10]<4.3>如果处理器只需做连续取指这一件事(见图4-6)，那么时钟周期是多少?**

**4.4.2[10]<4.3>考虑一个与图4-11（P170）类似的数据通路，但是假设处理器只需处理无条件相对跳转指令，那么时钟周期是多少?**

**4.4.3[10]<4.3>重做练习题4.4.2,但这次假设只需处理****有条件相对跳转指令。**

**本练习题中剩下的三个问题是关于Shift-Left-2数据通路单元的:**

4.4.4~6

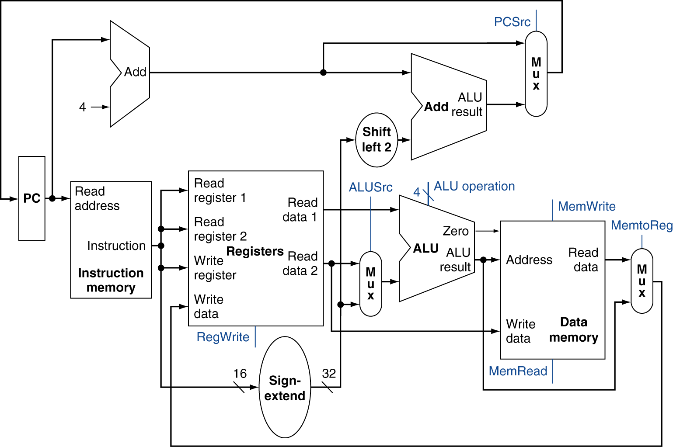
图4-17/4-24答案不同

**4.4.4 [10]<4.3>哪些类型的指令需要该单元?**

**4.4.5 [20]<4.3>对哪些类型的指令而言，该单元位于关键路径上?**

**4.4.6 [10]<4.3>假设仅需支持beq指令和add指令，讨论该****单元的延迟变化对处理器时钟周期的影响。假定其他单元的延迟不变。**

解：



（1）I-Mem比Add单元耗时长，所以时钟周期时间等于I-Mem的延时:200 ps

（2）这条指令的关键路径是通过指令存储器，Sign-extend和Shift-left-2得到偏移量，加法器计算新的PC地址, Mux选择该值而不是PC+4。所以时钟周期是

I-Mem regs shift2 add mux

200 ps + 15 ps + 10 ps + 70 ps + 20 ps = 315 ps

（3）有条件相对跳转指令与无条件相对跳转指令有相同的计算分支地址的长延时路径。除此之外，它们有一个长延时路径，通过寄存器、Mux和ALU来计算PCSrc条件。见P177图4-17（注图4-17不支持j指令，图4-24支持j指令）关键路径是两者中较长的一条，通过PCSrc的路径延迟较长。所以时钟周期是

I-Mem regs mux Alu mux

200 ps + 90 ps + 20 ps + 90 ps + 20 ps = 420 ps

（4）PC相对跳转指令 beq rs rt label（+j， jal 图4-24）

op rs rt offset

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | 5 | 5 | 6 |

（5）无条件相对跳转指令。从（3）中可知，Shift-Left-2数据通路单元不在有条件相对跳转指令的关键路径上。注意，MIPS实际上没有无条件相对跳转指令(bne zero, zero, Label扮演了这个角色，所以不需要无条件相对跳转指令操作码)，所以对于MIPS，这个问题的答案实际上是“None”。

（6）在这两条指令(BEQ和ADD)中，beq的关键路径更长，因此它决定时钟周期时间。每个ADD路径都小于或等于beq的对应路径，所以Shift-Left-2数据通路单元的延迟变化对ADD没有影响，我们的重点是该单元的延迟如何影响beq的关键路径。这个单元不在关键路径上，所以这个单元成为关键的唯一方法是增加它的延迟直到地址计算的路径1（通过Sign-extend+Shift-left-2+branch add=15+70+10=95）变得比路径2（通过registers+Mux+ALU=90+90+20=200）更长.

因此，Shift-left-2的延迟必须增加105 ps或更多，才能干扰时钟周期时间。

**4.5本练习题的问题中，假定没有流水线阻塞，各种类型所占的比例如下:**

**C:\Users\jingming\AppData\Local\Temp\1558616926(1).png**

**4.5.1 [10]<4.3>数据存储器平均用了多少时钟周期?**

**4.5.2 [10]<4.3>符号扩展电路的输入平均用了多少时钟周期?在未用到该输入的其他时间，符号扩展电路在做什么?**

解：

（1）LW和SW指令使用数据存储器，所以答案是:25% + 10% =35%

（2）符号扩展电路实际上是在每个循环中计算一个结果，但它的输出对于ADD和NOT指令来说被忽略。ADDI（提供即时ALU操作数）、BEQ（提供pc相对的偏移量）、LW、SW（提供用于寻址内存的偏移量）都需要符号扩展电路的输入，所以答案是：20% + 25% + 25% + 10% = 80%

**4.7 本练习题将讨论单周期数据通路中指令的执行细节。本练习题中的问题基于单周期处理器取入如下的指令字后的一个时钟周期：**

**10101100011000100000000000010100**

**假定数据存储器是全0且处理器的寄存器在取入以上指令字的时钟周期的开始时的内容如下：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **r0** | **r1** | **r2** | **r3** | **r4** | **r5** | **r6** | **r8** | **r12** | **r31** |
| **0** | **-1** | **2** | **-3** | **-4** | **10** | **6** | **8** | **2** | **-16** |

**（1）<4.4> 对该指令字而言，符号扩展单元和左移两位单元（图4-24的左上角）的输出是什么？**

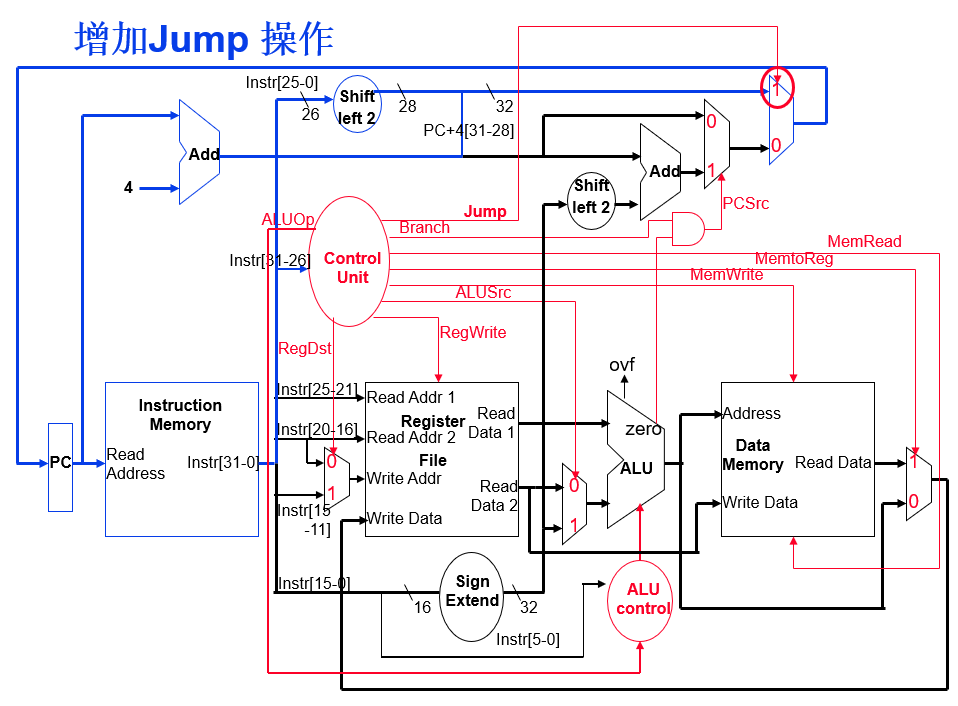
**（2）<4.4> 对该指令字而言，ALU控制单元的输入是什么？**

**（3）<4.4> 该指令执行后的新PC值是什么？在图4-24中灰线决定该新PC值的路径。**

**（4）<4.4> 对给定的指令字和寄存器堆初值，给出每个多选器数据输出的值。**

**（5）<4.4> 给出ALU和两个加法器数据输入的值。**

**（6）<4.4> 给出寄存器堆所有输入信号的值。**



答：（1）**符号扩展单元 0…0 0000 0000 0001 0100(指令的低16位)**

**左移两位单元 00 0110 0010 0000 0000 0001 0100（指令的0-25位）00（左移两位）**

（2）分析指令的01串：**101011 00011 00010 0000 0000 0001 0100**

**操作码** **rs(r3)** **rt(r2)** **16位**

**0x2B（sw指令，I型指令）**

**Sw指令做将rt存入rs+offset 所以，ALU control做加法输出ALUOP 00，输入操作码101011**

**（3）pc+4 因为不是跳转指令，所以，加4后经过两个选择器写回pc**

**（4）查阅表4-18（P178）可得 sw指令，指令存储器出来的多选器与sw无关**

**运算器多选器16位扩展。数据存储器无关。上方两个都是选0，输出pc+4.**

**(5) -3+20 pc+4 pc+4+20\*4（注意这里不是jump的那一条通路，是16位扩张那个beq）**

**RegDst 可取任何值（0或1）所以输出指令的20-16位值位00010（2）或15-11的值为0**

**ALUSrc取1选16位扩展 0000 0000 0001 0100（20）**

**MemtoReg 未知**

**PcSrc取0 Jump取0，输出pc+4**

**（6）25-21位20-16位对应的寄存器号，writeaddr 任意， writedata任意，Regwrite 0**

**4.8本练习题讨论流水线对处理器时钟周期的影响。下面给出了数据通路中不同阶段的延迟情况：**

**C:\Users\jingming\AppData\Local\Temp\1558621080(1).png**

**另外，假定处理器执行各种指令的比率如下面所示：**

**C:\Users\jingming\AppData\Local\Temp\1558621098(1).png**

**4.8.1 [5]<4.5>流水线处理器与非流水线处理器的时钟周期分别是多少?**

**4.8.2 [10]<4.5>lw指令在流水线处理器和非流水线处理器中的总延迟分别是多少?**

**4.8.3 [10]<4.5>如果可以将原流水线数据通路的一级划分为两级，每级的延迟是原级的一半，那么你会选择哪一级进行划分?划分后处理器的时钟周期为多少?**

**4.8.4 [10]<4.5>假设没有阻塞和冒险，数据存储器的利用率是多少(占总周期数的百分比)?**

**4.8.5 [10]<4.5>假设没有阻塞和冒险，寄存器堆的写寄存器端口的利用率是多少?**

**4.8.6 [30]<4.5>假设一种多周期的处理器设计，其中每条指令需要多个时钟周期完成，但上一条指令完成前不取下条指令。在这种设计中，指令仅经过其所需的阶段(例如，存储指令仅需4个时钟周期，因为其不需要WB阶段)。比较单周期设计、多周期设计和流水线设计三者的时钟周期和总执行时间。**

解：

（1）非流水线处理器的时钟周期=250+350+150+300+200=1250ps

流水线处理器的时钟周期=max{250，350，150，300，200}=350ps

（2）lw指令在流水线处理器中的总延迟=350\*5=1750ps

lw指令在非流水线处理器中的总延迟=250+350+150+300+200=1250ps

（3）选择ID级进行划分，划分后的处理器的时钟周期=max{250，350/2，150，300，200}=300ps

（4）在4种指令中，LW指令和SW用到了数据存储器

数据存储器的利用率=20%+15%=35%

（5）在4种指令中，ALU指令和LW指令用到了寄存器堆的写寄存器端口

寄存器的写寄存器端口的利用率：45%+20%=65%

（6）单周期设计的时钟周期为1250ps，多周期设计和流水线设计的时钟周期均为350ps。

在单周期设计中，每条指令需要一个时钟周期。在流水线中，没有阻塞的长时间运行的程序在每个周期中完成一条指令。一个多周期设计完成一个LW需要5个周期，一个SW需要4个周期(无WB)，一个ALU指令需要4周期(无MEM)，一个BEQ指令3周期(无WB和MEM)。

多周期设计的总执行时间/流水线设计的总执行时间=0.2\*5+0.2\*3+(0.45+0.15)\*4=4倍

单周期设计的总执行时间/流水线设计的总执行时间=1250/350=3.57倍

**4.9本练习题讨论数据相关如何影响4.5节中基本五级流水线的运行。结合下面的指令序列完成问题:**

**or r1,r2,r3**

**or r2,r1,r4**

**or rl,r1,r2**

**另外，假定每种相关旁路的周期时间如下:**

****

**4.9.1 [10] <4.5>指出指令序列中存在的相关及其类型。**

**4.9.2 [10] <4.5>假设该流水线处理器没有旁路，指出指令序列中存在的冒险并加入nop指令以消除冒险。**

**4.9.3 [10] <4.5>假设该流水线处理器中有充分的旁路。指出指令序列中存在的冒险并加入nop指令以消除冒险。**

**4.9.4 [10] <4.5>该指令序列在无旁路和有充分的旁路时，总执行时间分别是多少?后者相对于前者的加速比是多少?**

**4.9.5 [10] <4.5>如果仅有ALU至ALU的旁路(没有从MEM到EX的旁路)，如何加入nop指令以消除可能的冒险?**

**4.9.6 [10] <4.5>该指令序列在仅有ALU至ALU的旁路时，总执行时间分别是多少?与无旁路的情况相比，加速比是多少?**

**解：**

1. R1：I1 -> I2/I3 RAW I2 -> I3 WAR I1 -> I3 WAW

R2: I1 -> I2 WAR I2 -> I3 RAW

1. 因为WAW和WAR不会引发危险

OR R1,R2,R3

NOP 推迟I2避免R1的RAW

NOP

OR R2,R1,R4

NOP 推迟I3避免R2的RAW

NOP

OR R1,R1,R2

（3） 在完全旁路的情况下，ALU指令可以将一个值转发到下一条指令的EX阶段，而不会有冒险，所以不需要加入nop。（不是装载使用型时）

（4） 无旁路时：一个三指令序列需要执行7(P190图)个周期+（2）中添加的4个周期=（7+4）\*250=2750ps

充分旁路时：7\*300=2100ps 加速比：2750/2100=1.31

（5） 使用ALU-ALU的旁路, ALU指令可以转发到下一条指令，但不能转发到第三条指令。此时也无冒险，所以不需要加入nop。

OR R1,R2,R3

OR R2,R1,R4 ALU-ALU将R1从I1旁路到I2

OR R1,R1,R2 ALU-ALU将R2从I2旁路到I3

（6）无旁路时：2750ps

仅有ALU至ALU的旁路时：7\*290=2030ps 加速比：2750/2030=1.35

**4.11考虑下面的循环:**

**loop: 1w r1,0(r1)**

**add r1,r1,r2**

**1w r1,0(r1 )**

**1w r1,0(r1)**

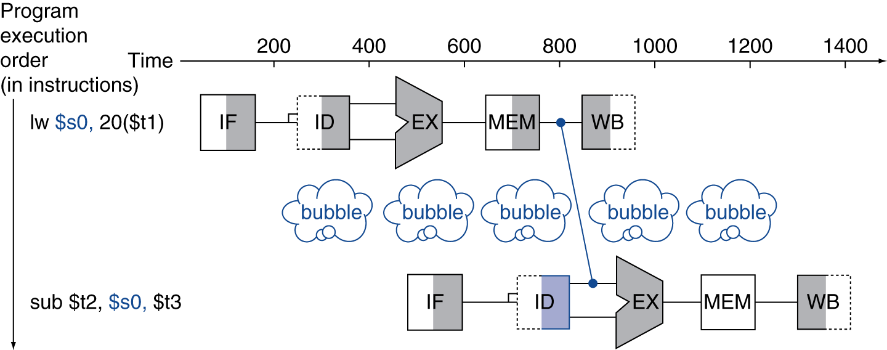
**beq r1,r0,1oop**

**假定使用了完美的分支预测(没有控制冒险导致的阻塞)，没有延迟槽，流水线支持完全旁路。另外假定循环结束前该循环迭代了很多次。**

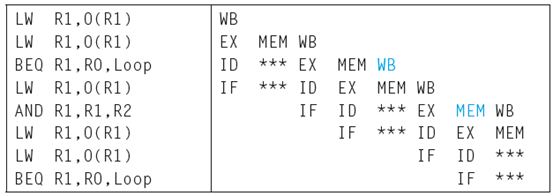
**4.11.1 [10]<4.6>画出循环第三次执行的流水线图，从取出循环的首条指令开始至取出下次循环的首条指令(不包括该次取指)结束。给出这段时间内流水线中的所有指令(不仅仅是第三次循环中的那些指令)。**

**4.11.2 [10]<4.6>在这段时间内有百分之多少五级流水线都在做有用的工作?**

解：



Lw的mem后下一tiao执行ex

（1）

R的ex后下一条执行ex

（2）在特定的时钟周期中，如果流水线阻塞，或者经过该阶段的指令在该阶段没有执行任何有用的工作，则流水线阶段就没有执行有用的工作。在4.11.1的流水线执行关系图中，如果某个阶段的名称没有显示在特定的循环中，那么该阶段阻塞，而特定的指令没有执行有用工作的阶段，则用蓝色标记。BEQ指令在MEM阶段正在做有用的工作，因为它在该阶段确定下一条指令的PC的正确值。通过分析，有：

|  |  |  |
| --- | --- | --- |
| 每次循环迭代所需周期 | 所有阶段都在做有用工作的周期数 | 百分比 |
| 8 | 0 | 0% |

注：

**add r1,r2,r2**

**sw r2, 8(r2)**

**add r1,r2,r2**

**add r1,r2,r2**

**add r1,r2,r2**

IF ID EX MEM WB

IF ID EX MEM WB

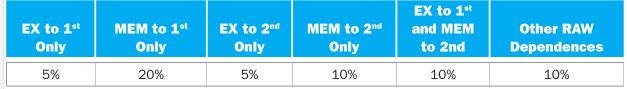
IF ID EX MEM WB

IF ID EX MEM WB

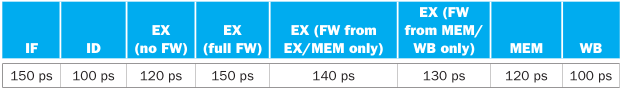
IF ID EX MEM WB

1/9 总共9个时钟周期，其中一个都在做有用的工作

**4.12本练习题讨论流水线处理器中旁路的成本/复杂度/性能折中。参考图4-45的流水线数据通路,假设指令中有部分存在RAW ( read after write,写后读)数据相关。RAW数据相关根据生成结果的流水级(EX或MEM)和使用结果的流水级(1st 意味着生成结果后的第一条指令， 2nd 意味着生成结果后的第二条指令)确认。假设在时钟周期的前半部分写寄存器，在后半部分读寄存器，这样“EX to 3rd”和“MEM to 3rd”相关不会产生数据冒险。最后假设无数据冒险时处理器的CPI为1。**

****

**假定各级流水线延迟如下。其中EX级给出了不同旁路情况下的延迟。**

****

**4.12.1 [10]<4.7>如果不使用旁路，会有百分之多少的时钟周期因为数据冒险阻塞?**

**4.12.2 [5]<4.7>如果使用完全的旁路(旁路所有可以旁路的结果)，会有百分之多少的时钟周期因为数据冒险阻塞?**

**4.12.3 [10]<4.7>假设不能提供三输入多选器(对完全的旁路是必需的)，我们必须确定从EX/MEM流水线寄存器旁路(旁路下一个周期)还是****从MEM/WB流水线寄存器旁路( 旁路2个周期)更好?哪种方法会产生更少的数据阻塞?**

**4.12.4 [10]<4.7>对给定的冒险概念和流水级延迟，完全的旁路相对于无旁路的加速比是多少?**

**4.12.5[10]<4.7>如果加入能消除所有数据冒险的时间旅行旁路，其相对于具有旁路机制的处理器的加速比是多少?假设在EX级完全旁路的基础上加入这个还没发明的时间旅行旁路的代价是增加100ps的延迟。**

**4.12.6 [20] <4.7>重做练习题4.12. 3,这次问哪种方法会产生更小的CPI。**

解：

（1）分析：RAW引起的数据冒险阻塞有如下几种情况：

1）Ex to 1st：会引起2个周期的阻塞

2）mem to 1st: 2

3）ex to 2nd： 1

4）mem to 2nd： 1

5）ex to 1st 和 mem to 2nd ：

Lw $t3 , 0($t2)

Add $s0, $t0, $t1

Sub $t2, $s0, $t3

由ex to 1st决定延迟 2

所以，不使用旁路，CPI为理想的CPI+延迟引起的时钟周期

= 1+（1）+2）+5））\*2 + （3）+4））\*1

= 1+0.35\*2 +0.15\*1

= 1.85

0.85/1.85 = 46%

即46%的时钟周期因数据冒险而阻塞

（2）在完全旁路的情况下，只有2）mem to 1st 仍需阻塞一个时钟周期（装载使用型指令）

所以，CPI =1+（2））\*1 = 1+0.2 = 1.2 0.2/1.2 = 17%

即17%的时钟周期因数据冒险而阻塞

（3）如果只从EX/MEM寄存器旁路，

1. ex to 1st : 无阻塞
2. mem to 1st: 2
3. ex to 2nd： 1
4. mem to 2nd：1
5. ex to 1st 和 mem to 2nd ：由mem to 2nd 决定 1

所以 2）\*2 + 3）+ 4）+ 5） = 0.2\*2 + 0.05 + 0.1 + 0.1 = 0.65

仅从MEM/WB转发

1. ex to 1st : 1
2. mem to 1st: 1
3. ex to 2nd： 0
4. mem to 2nd：0
5. ex to 1st 和 mem to 2nd : 1

所以 0.05 + 0.2 + 0.1 = 0.35

所以从MEM/WB流水线寄存器旁路更好，会产生更少的数据阻塞。

1. 在4.12.1和4.12.2中，我们已经计算了不旁路和完全旁路时的CPI。

cpu时间 = I\*CPI\*cc = I \* 1.85 \*CC

CC = max（IF，ID，ex全，mem，WB）=150 ps 则有：

无旁路：1.85\*150ps = 277.5ps 完全旁路：1.20\*150ps = 180ps

加速比 = 227.5/180 = 1.54

1. 完全旁路：1.20\*150ps = 180ps

时间旅行旁路（可消除所有冒险）：1\*（100ps+150ps）= 250ps

加速比：180/250 = 0.72

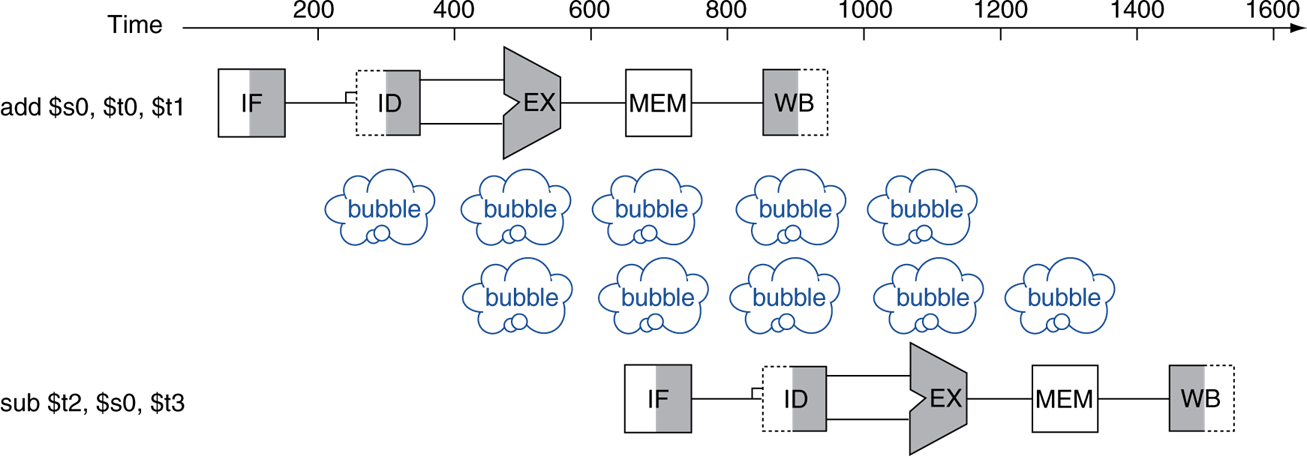
（6）EX/MEM = 1.65\*150ps = 247.5ps MEM/WB = 1.35\*150ps = 202.5ps

从MEM/WB流水线寄存器旁路更好，会产生更少的数据阻塞。

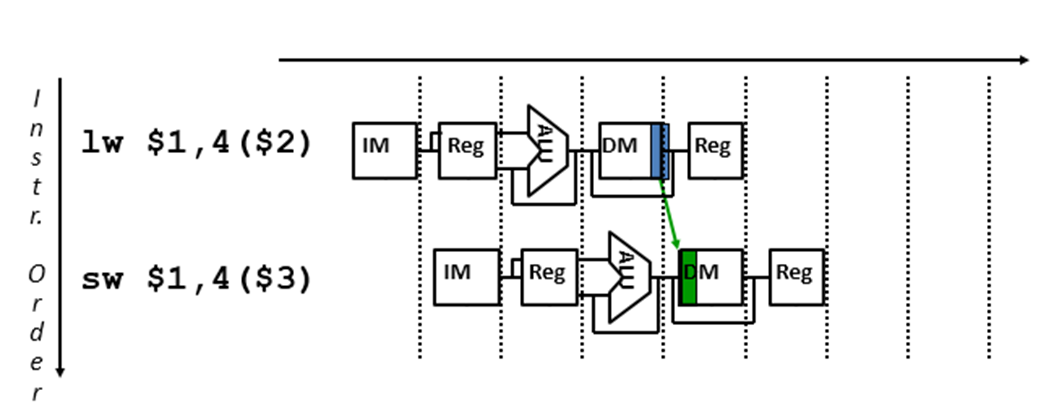
**PPT答案：**

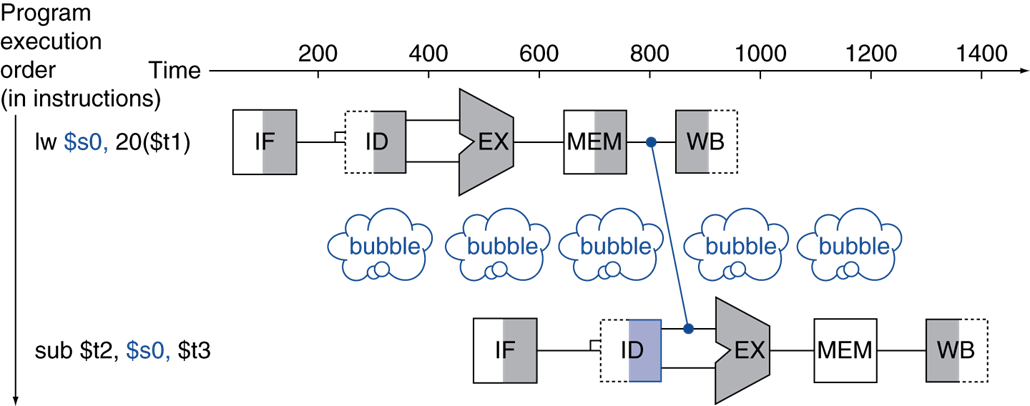
Ex to1st（2个时钟周期阻塞）

例：add $s0, $t0, $t1  
 sub $t2, $s0, $t3



Mem to 1st(一个或2个时钟周期阻塞）

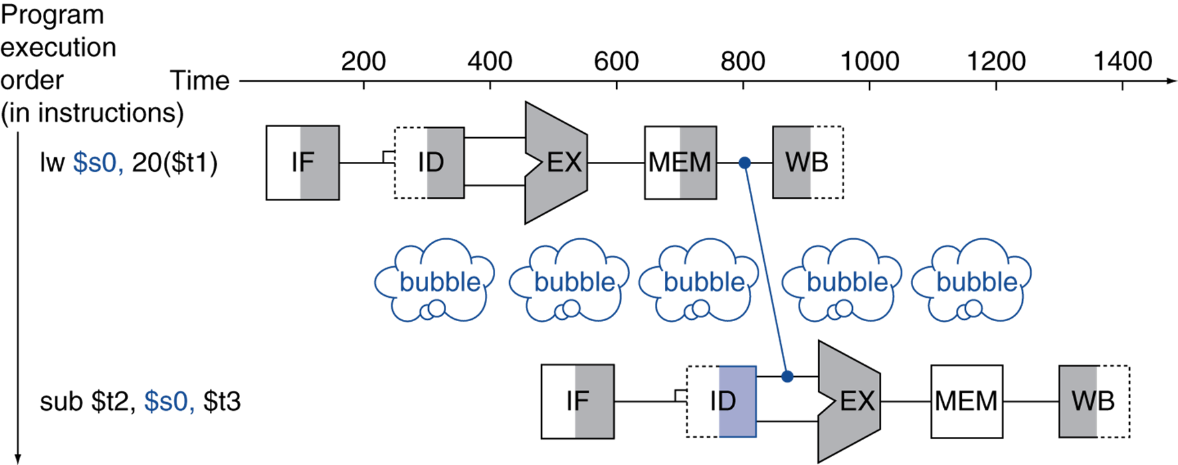




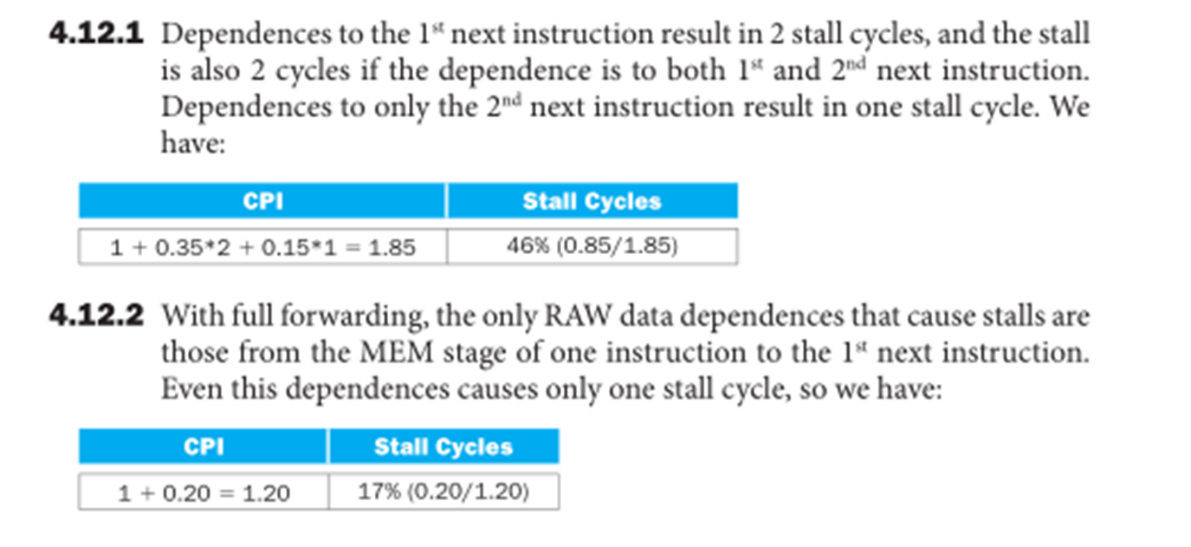
（这个图是转发了，若不转发，还要延迟一个时钟周期，即2个时钟周期）

Ex to 2nd（一个时钟周期阻塞）

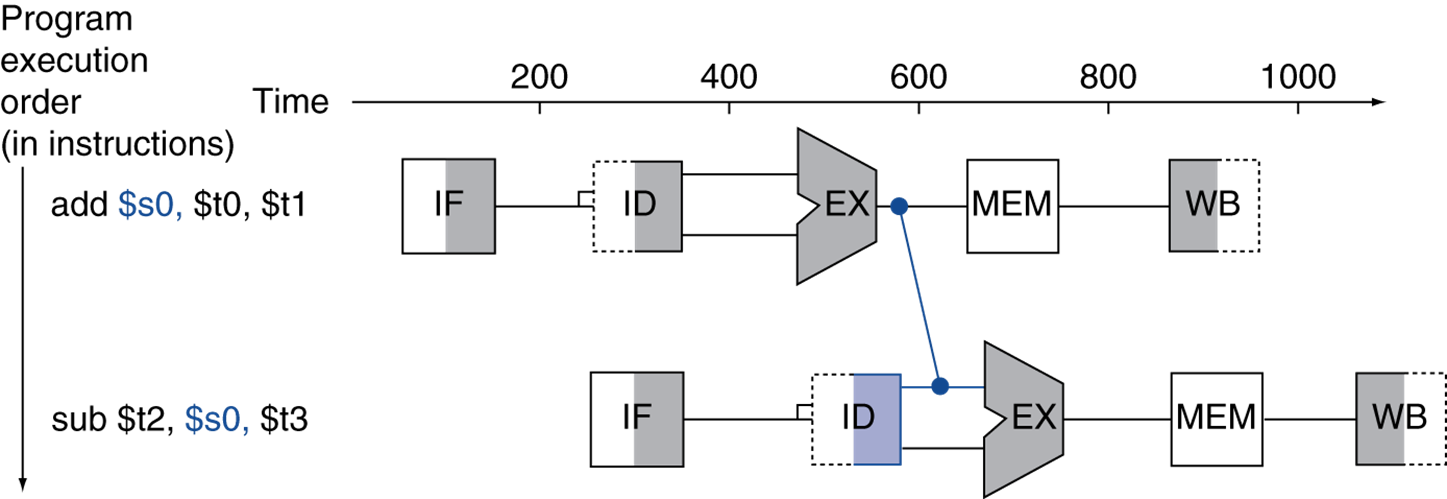
Mem to 2nd（这个图是转发了，若不转发，还要延迟一个时钟周期，即1个时钟周期）



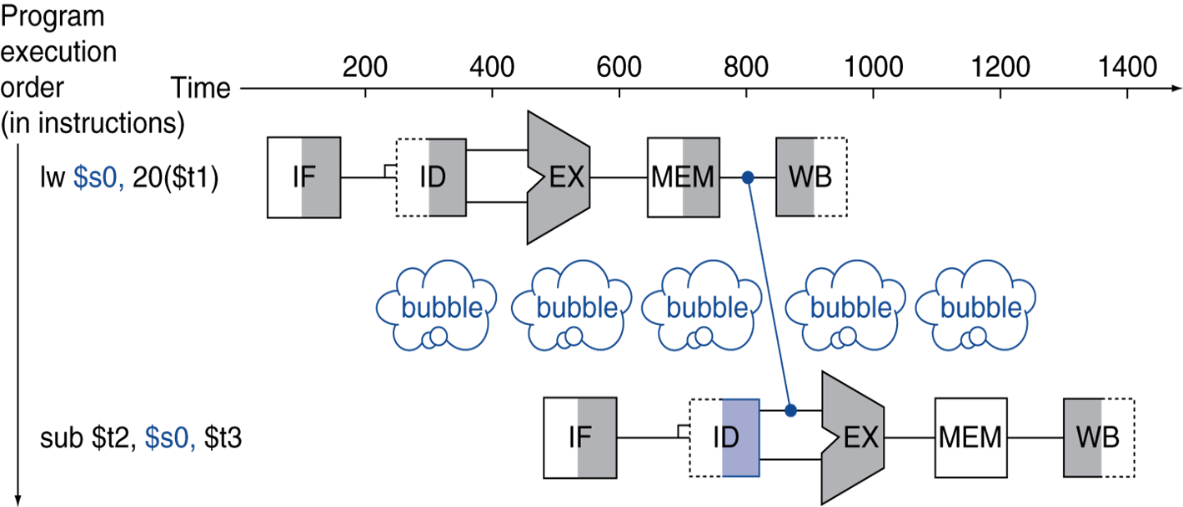
网上答案

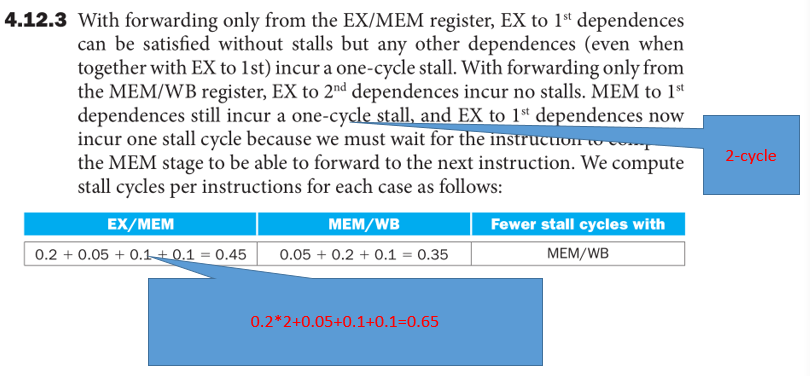


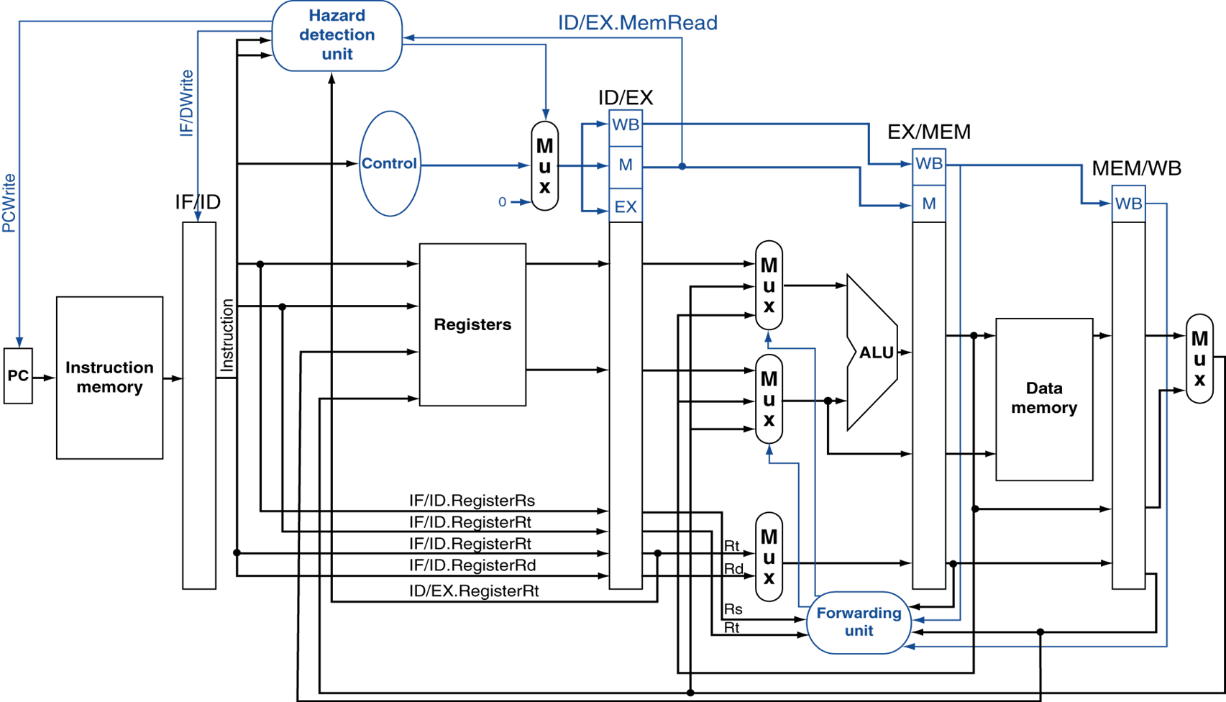
EX/Mem 转发

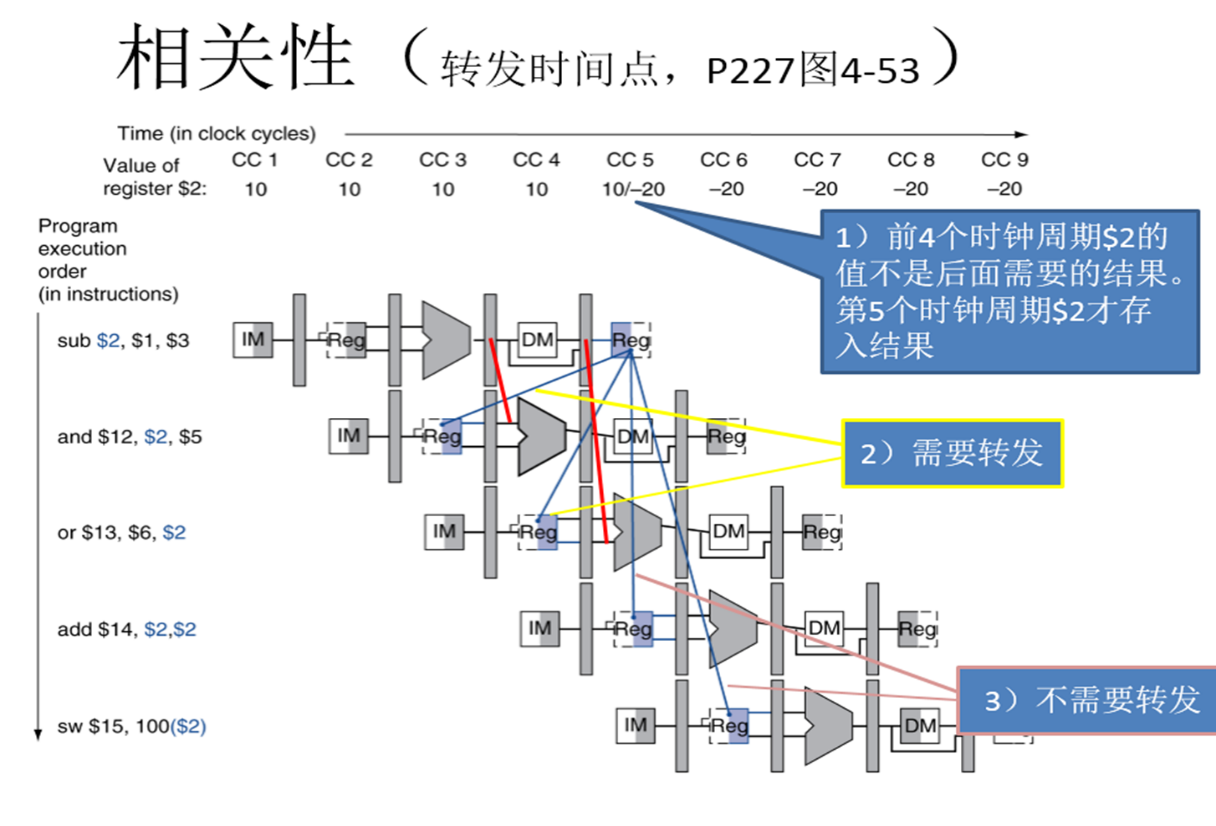


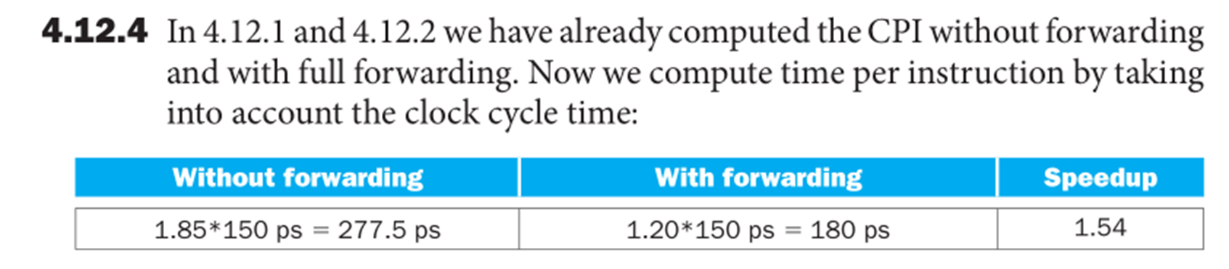
Mem/WB转发（MEMto1st，延一个时钟周期）

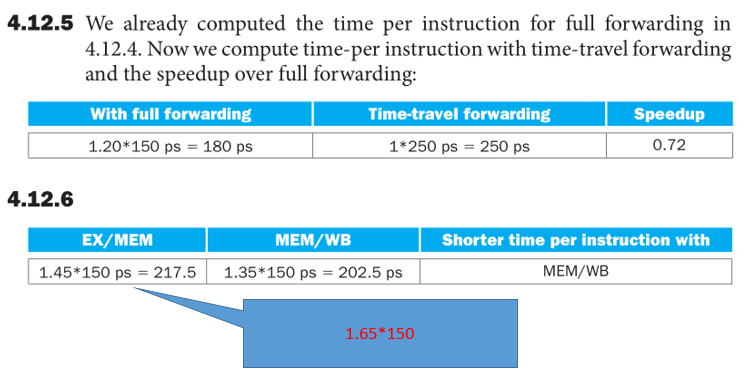




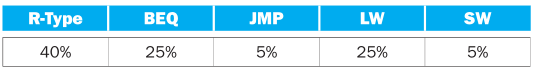




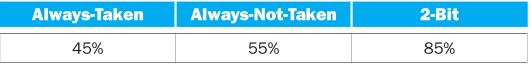




**4.15一个好的分支预测器有多重要取决于条件分支指令的频率，它与分支预测器的精度共同决定误预测分支导致的阻塞时间长短。在本练习题中，假设指令的动态执行频度如下:**

****

**假定分支预测器精度如下：**

****

**4.15.1 [10]<4.8> 误预测分支导致的阻塞将增加CPI。对****分支总发生预测器而言，误预测分支将导致CPI增加多少?假设分支方向在EX级确定，没有数据冒险且不使用延迟时间槽。4.15.2 [10]<4.8> 重做练习题4.15.1,这次改为分支总不发生预测器。**

**4.15.3 [10]<4.8>重做练习题4.15.1,这次改为2位分支预测器。**

**4.15.4 [10]<4.8>对****2位分支预测器而言，将一半分支指令用ALU指令替代(一条ALU指令替代一条分支指令)将获得的加速比是多少?假设被正确预测的分支指令和被不正确预测的分支指令被取代的概率相同。**

**4.15.5 [10]<4.8>对2位分支预测器而言，将一半分支指令用ALU指令替代(两条ALU指令替代一条分支指令)将获得的加速比是多少?假设被正确预测的分支指令和被不正确预测的分支指令被取代的概率相同。**

**4.15.6 [10]<4.8>有些分支是非常容易预测的。假设80%的分支指令都是非常容易预测的循环返回分支，那么2位分支预测器对剩下的20%分支指令的预测精度是多少?**

解：

1. 每个分支如果不能被分支总发生预测器正确预测，将导致3个阻塞周期，因此我们有:25%\*（1-45%）\*3 = 41.25%
2. 每个分支如果不能被分支总不发生预测器正确预测，将导致3个阻塞周期，因此我们有:25%\*（1-55%）\*3 = 33.75%
3. 每个分支如果不能被2位分支预测器正确预测，将导致3个阻塞周期，因此我们有:25%\*（1-85%）\*3 = 11.25%
4. 正确预测的分支的CPI是1，现在它们变成了CPI也是1的ALU指令。错误预测指令也会变成CPI为1的ALU指令，所以我们有: 1 + 3\*(1-0.85)\*0.25\*0.5 = 1.05625

代替前CPI：1 + 25%\*（1-85%）\*3 = 1.1125 加速比：1.1125/1.05625 = 1.053

1. 每条转换后的分支指令现在都需要一个额外的周期来执行，所以我们有:

1 + (1 + 3\*(1 − 0.85))\*0.25\*0.5 = 1.18125 加速比：1.1125/1.18125 = 0.94

1. 设2位分支预测器对剩下20%分支指令预测精度为y，则：

1\*0.85 = 0.8 + 0.2\*y =〉 y = 0.25