COD第4次作业答案

负责助教：王晶

4.16 在本题中将讨论流水线如何影响处理器的时钟周期。假设数据通路的各个流水级的延迟如下：

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300ps	200ps

同时，假设处理器执行的指令分布如下：

ALU/Logic	Jump/Branch	Load	Store
45%	20%	20%	15%

4.16.1 [5] <4.5> 在流水化和非流水的处理器中，时钟周期分别是多少？

4.16.2 [10] <4.5> 在流水化和非流水的处理器中，对于 `ld` 指令的延迟分别是多少？

4.16.3 [10] <4.5> 如果我们将数据通路中的一个流水级拆成两个新流水级，每一个新流水级的延迟是原来的一半，那么我们将拆分哪一级？新处理器的时钟周期是多少？

4.16.4 [10] <4.5> 假设没有停顿或冒险，数据存储的利用率如何？

4.16.5 [10] <4.5> 假设没有停顿或冒险，寄存器堆的写口利用率如何？

4.16.1 对于流水线处理器，时钟周期取延迟最大的一个阶段；对于非流水线，时钟周期为各阶段的延迟总 和。
那么对于本题，流水线周期为350ps，非流水线周期为250 + 350 + 150 + 300 + 200 = 1250ps

4.16.2 流水线中，ld指令进流水线到出流水线的时间为1750(350*5)ps，ld指令的绝对延迟为1250ps，这道题题意不清，两个答案都算对。非流水线中，ld指令的延迟为1250ps。

4.16.3 应该拆分延迟最大的流水级即ID阶段。新处理器的时钟周期应该为拆分后延迟最大的流水线对应的 延迟，即MEM阶段，故为300ps。

4.16.4 题中英文版的表述不同

中文版如下：**4.16.4** ［10］<4.5>假设没有停顿或冒险，数据存储的利用率如何？

英文版如下：4.16.4 [10] <§4.5> Assuming there are no stalls or hazards, what is the utilization of the data memory?

4.16.4 其中data memory这一模块的利用率即数据存储和数据读取的利用率和。但中文版问的是“数据存储”的利用率，容易让人理解为只有数据存储的利用率而忽略了数据读取。

Load指令会利用data memory来进行数据读取，占比20%；store指令会利用data memory来进行数据存储，占比15%。所以这题回答15%/35%均判正确。

4.16.5 ALU/Logic和Load指令均会用到寄存器堆的写口: ALU/Logic需要将运算结果写回寄存器堆，而Load 需要将从存储器中读取的数据写回寄存器堆，所以寄存器堆的写口利用率为45% + 20% = 65%。

4.22 [5]<4.5>对于如下的 RISC-V 的汇编片段：

```
sd    x29, 12(x16)
ld    x29, 8(x16)
sub   x17, x15, x14
beqz  x17, label
add   x15, x11, x14
sub   x15, x30, x14
```

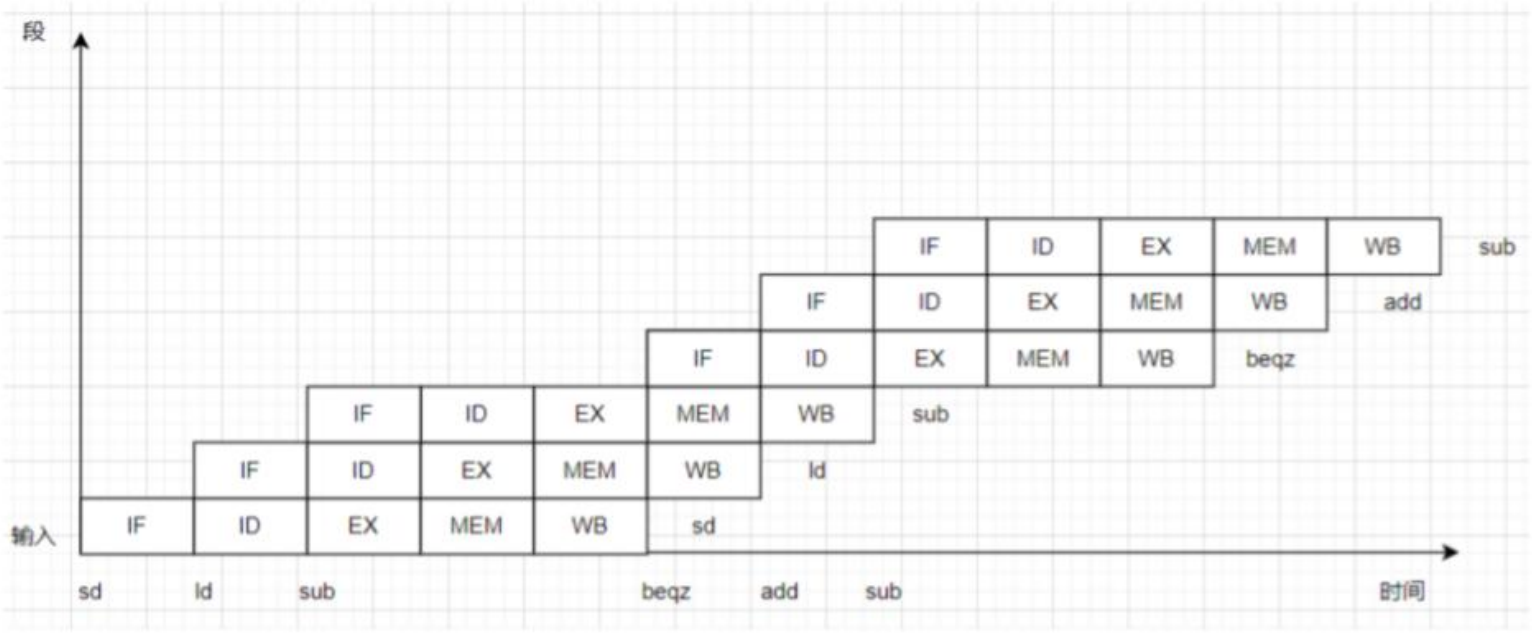
4.22.1 [5]<4.5>请画出流水线图，说明以上代码会在何处停顿。

4.22.2 [5]<4.5>是否可通过重排代码来减少因结构冒险而导致停顿的次数？

4.22.3 [5]<4.5>该结构冒险必须用硬件来解决吗？我们可以通过在代码中插入 NOP 指令来消除数据冒险，对于结构冒险是否可以相同处理？如果可以，请解释原因。否则，也请解释原因。

4.22.4 [5]<4.5>在典型程序中，大约需要为该结构冒险产生多少个周期的停顿？(使用习题 4.8 中的指令分布)。

4.22.1



`sub x17,x15,x14` 和 `beqz x17,label` 存在冲突，beqz需要 stall两个周期，等待sub的结果写回寄存器堆后才能取出正确结果。

这里我们是按照寄存器堆同步写异步读处理的，即读寄存器不需要时钟控制，但写寄存器需时钟控制。如果有同学按同步写同步读多stall一个周期算对。

如果sub到beqz有forward操作，只需stall一个周期。故此题stall 1/2/3个周期均可。

注意nop与stall不同，nop空指令，有完整的 IF/ID/EX/MEM/WB阶段，实际中被扩展为 `addi x0,x0,0`；而stall相当于一种使能信号，当stall为高电平有效时，段寄存器不能进行读写操作。但二者都可能使流水线暂停。

4.22.2 结构冒险：如果一条指令需要的硬件部件还在为之前的指令工 作，而无法为这条指令提供服务，那就导致了结构冒险。 CPU有两种结构：冯诺依曼结构(数据和指令存储在一起)和哈佛结构(数据和指令分开存储)。 所以这题的答案有2种。
按冯诺依曼结构作答：由于指令进行数据访问时都会与此时正处于IF阶段的指令冲突，导致stall，那么重排代码也不会减少这种stall的次数。倘若将sd或ld移动到sub1后面，会因为 sd和ld在MEM阶段的数据访问而导致后续指令(`add x15，x11，x14` `sub x15，x30，x14`) stall两个周期，注意，sd和ld不能插入beqz之后，否则不一定能被执行到，故无法减少stall次数。

按哈佛结构，分析同上，但是不会因为sd和ld在MEM阶段的数据访问而导致stall两个周期，故可以减少。无论回答是或否言之有理即可，只回答是或否不得分。

4.22.3 对于冯诺依曼结构，不能通过插入NOP来解决，因为即使是NOP也要在存储器中取出指令，所以无论加 在哪里也避免不了sd和ld在MEM阶段的数据访问导致的结构冒险。通过硬件解决的方案就是把冯诺依曼结构变成了哈佛结构或者增加mem的端口或者通过stall处理。
对于哈佛结构，可以通过在sub1之后加两个NOP来解决问题。

注：4.22.2和4.22.3绝大部分同学都默认使用了冯诺依曼结构，没有提前说明，之后碰到这种问题，为避免扣分最好事先说明采用的是哪种结构。

R-type/I-type (non-ld)	ld	sd	beq
52%	25%	11%	12%

上图是4.22.4中使用的指令结构，下面是4.22.4的解答：

对于冯诺伊曼结构的结构冒险，每次MEM阶段对数据进行读取或者存储都会与此时正处于ID阶段的 指令冲突。所以Load和Store指令都会产生一个stall，故大概产生36%：11%+25%个周期的停顿。
对于哈佛结构的结构冒险，情况较复杂，与指令顺序等有关，这里不考虑。

4.23 如果我们改变 load/store 指令格式，使用寄存器（不需要立即数偏移）作为访存地址，这些指令就不再需要使用 ALU（具体见习题 4.15）。这样的话，MEM 阶段和 EX 阶段就可以重叠，流水级数变为四级。

4.23.1 [10] <4.5> 流水线级数的减少会影响时钟周期吗？

4.23.2 [10] <4.5> 这样的变化会提高流水线的性能吗？

4.23.3 [10] <4.5> 这样的变化会降低流水线的性能吗？

4.15 从 4.4 节可知，ld 是 CPU 中延迟最长的一条指令。如果修改 ld 和 sd 的功能，去掉地址偏移，比如 ld 或 sd 指令的访存地址只能使用计算后存放于 rs1 中的数值，那么就不会有指令同时使用 ALU 和数据存储，这将缩短时钟周期。但是这也会增多指令数目，因为很多 ld 和 sd 指令将会被 ld/add 或者 sd/add 组合取代。

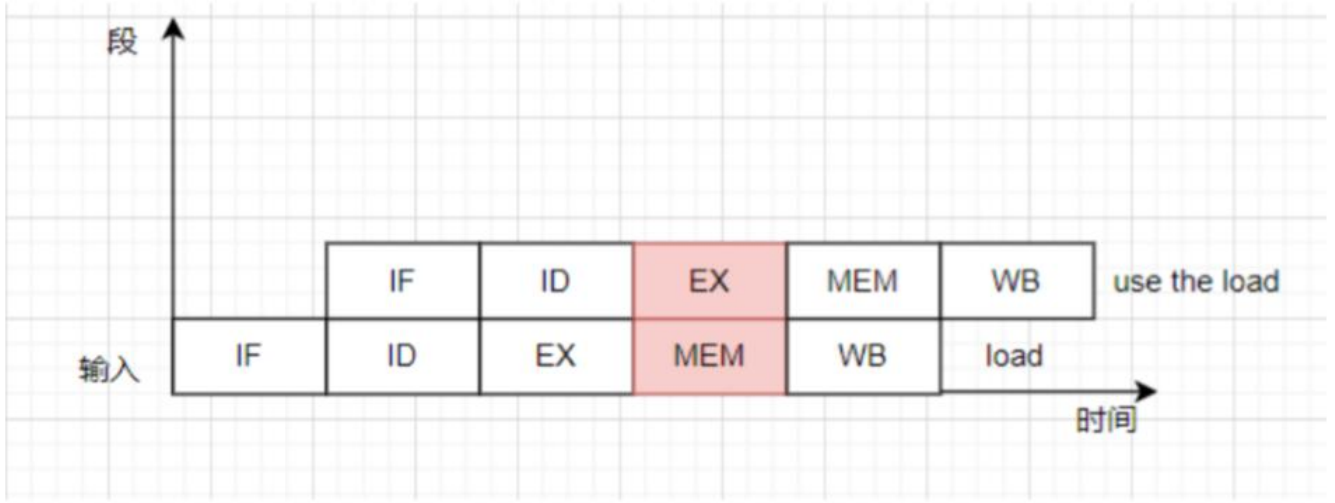
4.4节中的指令组合：

指令类型	取指令	读寄存器	ALU操作	数据存取	写寄存器	总时间
Load doubleword (ld)	200 ps	100 ps	200 ps	200 ps	100 ps	800 ps
Store doubleword (sd)	200 ps	100 ps	200 ps	200 ps		700 ps
R-format (add, sub,and, or)	200 ps	100 ps	200 ps		100 ps	600 ps
Branch (beq)	200 ps	100 ps	200 ps			500 ps

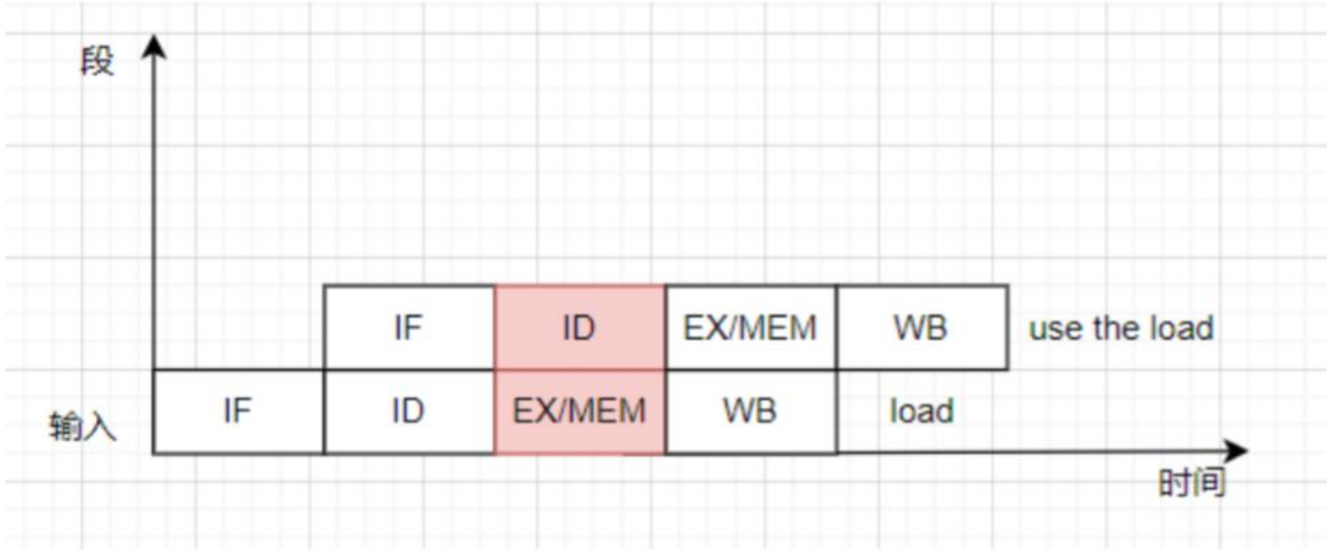
4.23.1 不会改变。原图中Instruction fetch的延迟仍为最大。

注：这道题很多同学没有找对图，但回答出不一定改变，且言之有理的均未扣分。

4.23.2 这样会优化load-use型冒险，比如load x1,0(x0);addi x2,x1,4。



优化前：需要stall两个周期。



优化后：只用stall一个周期。

故可以提高流水线性能。

4.23.3　因为很多ld和sd指令将会被ld/add或者sd/add组合取代。所以会增多指令数目。

总结：本次作业，很多同学不写原因。为避免扣分，请大家往后做题时写清楚理由，计算题要有过程，简答题和画图题要有解释说明。本次作业如果在习题课后仍有问题，可以联系对应负责助教解决。有认为批改错误的同学期末考试前均可联系助教解决。