计组书面作业第二单元

1. 以下关于 RISC-V 经典五级流水线 CPU 说法错误的是（　　）。
   A. 当需要在 EXE 段暂停流水线时，IF/ID 和 ID/EXE 流水段寄存器都应该保持不变；此时 MEM 流水段在没有阻塞的情况下，应将 EXE/MEM 流水段寄存器设置为空指令对应的值
   B. 当 ID 和 EXE 两个流水段下一周期需要清空时，IF/ID 和 ID/EXE 流水段寄存器都设置为空指令对应的值
   C. 在 IF 和 MEM 两个流水段发生结构冲突时，应该优先处理 IF 流水段的请求，以防止流水线中的指令流被阻塞住
   D. 在同一时钟周期有可能同时发生数据、控制、结构冲突

C

2. 在 THINPAD 平台上所实现的流水线结构 CPU 上，使用循环计算 1 到 100 的和，所有数据都在寄存器中，可能引发若干种冲突。针对此问题，下列说法中正确的是（　　）。
   A. 使用 beq 指令判断循环是否应该结束时，可能引发控制冲突
   B. 相邻的两轮循环可能因为读写相同的寄存器造成结构冲突
   C. 分支目标缓冲（Branch Target Buffer）对减少本次实验中的冲突没有帮助
   D. 数据旁路无法完全消除该程序中数据冲突带来的性能损失

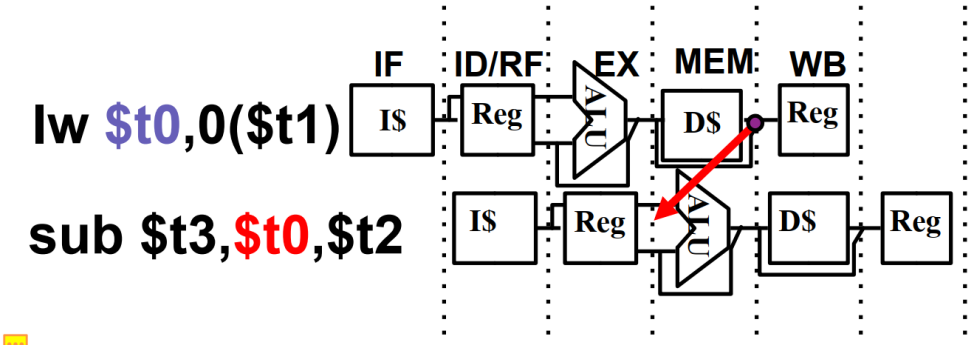
A

3. 简答并给出理由：计算机性能指标 CPI 是否是越低越好？

较低的CPI一般意味着更好的性能，因为在同等时钟频率下，较低的CPI可以在相同时间内执行更多的指令。但一味追求CPI低也会存在问题，比如：1.为了实现较低CPI，可能会修改指令集架构，如使用更简单的指令，这样每条指令的工作内容会更少，而完成一个相同的任务，需要的指令数量更多，反而可能降低了计算机的运行效率。2.为了降低CPI，可能会采取流水线设计，但如果存在分支预测错误或者冲突的情况，可能会引起流水线中断，增加延迟，反而拖慢了实际执行时间。3.较低的CPI往往会要求更复杂的电路设计，这对于硬件芯片的设计提出了更高的要求

4. 简答并给出理由：数据旁路可以解决所有数据冲突吗？

不可以，数据旁路无法解决 Load-Use Case，如课上的例子



在执行一些和数组有关的操作时往往会有类似的指令，在执行第一条 load 指令的时候，写入 `t0` 的值需要在MEM阶段后才能得到，而不是像其它需要写回寄存器的指令一样在EXE之后就可以得到，因此，如果下条指令需要使用 `t0` 寄存器内的值，就必须要等待一个周期，因此，数据旁路并不解决所有数据冲突。

5. 简答并给出理由：RISC-V 只用一套 CSR 寄存器，能否支持中断嵌套？

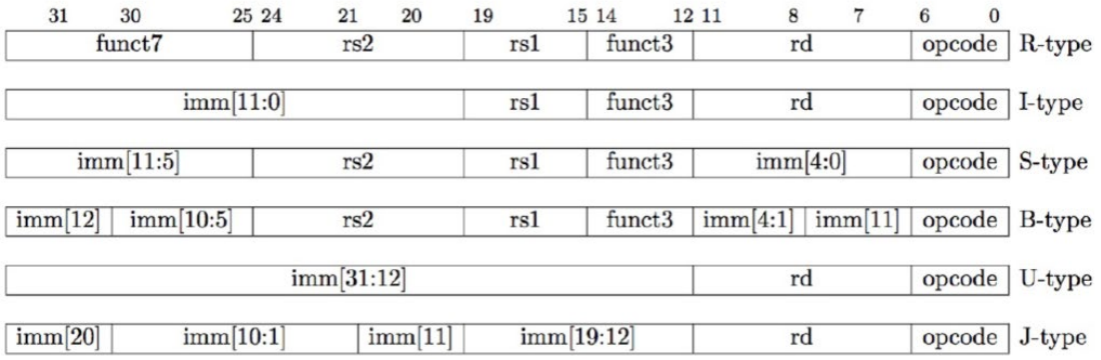
可以，可以通过中断控制器处理和控制中断请求，根据中断的优先级调整顺序。虽然只有一套CSR寄存器，但可以将所有需要保存的寄存器的值（如通用寄存器、CSR寄存器等）保存到内存或者堆栈中（把内存的一部分地址预留出来作为内核栈或者异常栈），这样就实现了对现场的保存，然后去处理下一层的中断或异常。

6. 有如下 RISC-V 代码：

```
1.      lui     a5,%hi(fibo)
2.      addi    a4,a5,%lo(fibo)
3.      li      a3,1
4.      addi    a5,a5,%lo(fibo)
5.      sw      a3,0(a4)
6.      sw      a3,4(a4)
7.      addi    a2,a5,32
8.      li      a4,1
9.      j       .L3
10. .L5:
11.     lw      a4,4(a5)
12.     lw      a3,0(a5)
13. .L3:
14.     add     a4,a4,a3
```

```
15.     sw      a4,8(a5)
16.     addi    a5,a5,4
17.     bne     a5,a2,.L5
18.     ret
```

RISC-V 指令格式如下：



imm[12|10:5]	rs2	rs1	000	imm[4:1|11]	1100011	BEQ
imm[12|10:5]	rs2	rs1	001	imm[4:1|11]	1100011	BNE
imm[12|10:5]	rs2	rs1	100	imm[4:1|11]	1100011	BLT
imm[12|10:5]	rs2	rs1	101	imm[4:1|11]	1100011	BGE
imm[12|10:5]	rs2	rs1	110	imm[4:1|11]	1100011	BLTU
imm[12|10:5]	rs2	rs1	111	imm[4:1|11]	1100011	BGEU

请回答：

(1) 请写出“bne a5,a2,.L5”对应指令的十六进制表示。(a5 和 a2 的寄存器编号分别为 15 和 12。)

(2) 请分析 bne 理论上的跳转范围。

(3) 对于上面的代码，请写出等效的 C 伪代码，并给出程序终止后 a4 的值。

(4) 如果使用五阶段流水线 CPU 设计，请指出上述代码中所有可能出现的数据冲突。

(1) rs1 = 01111，rs2=01100，imm=1 1111 1110 1100，因此对应指令的表示为0xFEC796E3

(2) 立即数有13位，因此跳转范围可以在 $2^{13}B = 8KB(\pm 4KB)$，考虑到最低位只能是零，精确来说是-4KB~+(4k-1)B

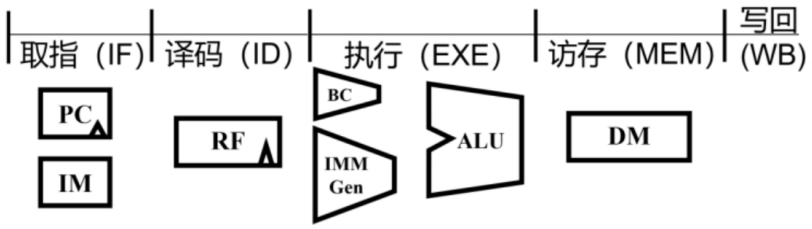
(3) a4的值为0x37（十进制下55）

```
fibo[0] = 1;
fibo[1] = 1;
for(int i=0; i<8; i++){
```

```
  fibo[i+2] = fibo[i] + fibo[i+1];
}
```

(4) 发生数据冲突的指令行数对为(1,2),(1,4),(2,5),(3,5),(3,6),(4,7),(8,14),(8,15),(14,15),(16,17),(16,11),(16,12),(11,14),(12,14)

7. 一个五级流水线 RISC-V CPU 各流水段的基本功能部件如图所示。



（1） 该 CPU 仅实现了部分数据旁路，在没有发生结构冲突的情况下，CPU 从 init 开始执行，执行 swap 代码段时仅发生了 1 次暂停。

```
vec:
  .word 1,9,2,2            # word 是 32 位
init:
  li a1, 0
  la a0, vec
  nop
  nop
swap:
  ori  a1, a1, 1
  slli t0, a1, 2
  add  t0, a0, t0
  lw   t1, 0(t0)
  lw   t2, 4(t0)


  sw   t2, 0(t0)
  sw   t1, 4(t0)
// End of swap
```

a) 请写出该 CPU **没有**实现的数据旁路，需要说明部件名称与流水段寄存器名称（若有）。如：ALU@EXE/MEM->ALU，表示保存在 EXE/MEM 流水段的 ALU 计算结果到 ALU 的数据旁路。

b) 重写该代码段，在保证代码段整体执行结果一致的情况下，避免 CPU 出现暂停。

(a) 没有实现 DM@MEM/WB→DM

(b)

```
ori a1, a1, 1
slli t0, a1, 2
lw t1, 0(t0)
lw t2, 4(t0)
sw t1, 4(t0)
sw t2, 0(t0)
```

(2) 若该 CPU 实现了所有可能的数据旁路，且该 CPU 不支持分支预测，且在跳转指令之后取 PC+4 作为 next PC，若执行跳转则清空 IF、ID，并将跳转目的地址作为 next PC。CPU 从 init 开始执行下列代码。

```
vec:
   .word 0
init:
  la a0, vec
  addi t0, x0, 0x1
  addi t1, x0, 0x0
  addi t3, x0, 0xa
  nop
count:
  add t0, t0, 0x1
  add t3, t3, -1
  add t1, t1, t0
  bne t3, x0, count
out:
  sw  t1, (a0)
  nop
  nop
```

a) 请写出：EXE 段执行上述 sw 指令的**前一个周期**，CPU 各流水段正在执行的指令（空泡用 nop 代替）。

b) CPU 在执行上述 count 时是否使用了所有可能的数据旁路？若有请写出所有未使用的数据旁路。

(a)

- IF: `nop`
- ID: `sw t1, (a0)`
- EXE: `bne t3,x0, count`
- MEM: `add t1, t1, t0`
- WB: `add t3, t3, -1`

(b)

没有，可能没有使用 ALU@EXE/MEM→ALU

（3） 有两种方案用于处理 ID 段产生的异常：

方案一：将 ID 段产生的异常信号存入 ID\EXE 流水寄存器，在下一个周期由 EXE 段的异常处理模块处理。

方案二：将 ID 段产生的异常信号直接传递给 EXE 段的异常处理模块，使得在这一周期即可开始进行处理。

在上述两个方案中，异常处理模块在接收到异常信号后，将发生异常的指令 PC 写入 MEPC，等待流水线清空，并控制 PC 寄存器的下一个取值为异常处理程序首地址。

这两个方案中有一个方案不满足精确异常定义。请指出不满足的方案，并通过汇编代码，举例说明在什么情况下，该方案无法实现精确异常。（译码异常指令用 RV.X 代替。）

方案一不满足，如果异常指令在EXE段被处理，那么它之后的指令可能已经进入了流水段并开始产生影响，这违反了精确异常的定义。例如下面的代码

```
addi t2, t1, 5
RV.X
add t3, t2, t1
```

在第一条指令中t2被重新写了，而第三条指令则需要用到t2的结果，出现数据冲突，如果用方案一，假设异常处理过程改变了 `t2` 的值或者流水线的状态，那么第三条指令执行可能会依赖于错误的信息，从而违反了精确异常的要求。