

第九周作业

9. (1) 最终的记分牌结构为

| | Issue | Read Oper | Exec Comp | Write Result |
|--------------------|-------|-----------|-----------|--------------|
| fld f2, 0(a0) | 1 | 2 | 3 | 4 |
| fdiv.d f8, f0, f2 | 5 | 6 | 14 | 15 |
| fmul.d f2, f6, f2 | 6 | 7 | 9 | 10 |
| fld f4, 0(a1) | 7 | 8 | 9 | 10 |
| fadd.d f4, f0, f4 | 11 | 12 | 13 | |
| fadd.d f10, f8, f2 | 14 | 16 | 17 | |
| fsw f10, 0(a0) | 15 | | | 18 |
| fsw f4, 0(a1) | 16 | | | 17 |
| addi a0, a0, 8 | 17 | | 17 | |
| addi a1, a1, 8 | 18 | | 18 | |
| sub x20, x4, a0 | 19 | | 19 | |
| bnz x20, Loop | 20 | | 21 | |

∴ 一次迭代需要21个周期执行完成。

(2) 最终的记分牌结构是

| | Issue | Read Oper | Exec Comp | Write Result |
|--------------------|-------|-----------|-----------|--------------|
| fld f2, 0(a0) | 1 | 2 | 3 | 4 |
| fdiv.d f8, f0, f2 | 5 | 6 | 14 | 15 |
| fmul.d f2, f6, f2 | 5 | 6 | 8 | 9 |
| fld f4, 0(a1) | 6 | 7 | 8 | 9 |
| fadd.d f4, f0, f4 | 6 | 10 | 11 | |
| fadd.d f10, f8, f2 | 7 | 16 | 17 | |
| fsw f10, 0(a0) | 7 | | | 8 |
| fsw f4, 0(a1) | 8 | | | 12 |

| | | | |
|------|-------------|----|----|
| addi | a0, a0, 8 | 8 | 8 |
| addi | a1, a1, 8 | 9 | 9 |
| sub | x20, x4, a0 | 9 | 9 |
| bnz | x20, Loop | 10 | 11 |

∴ 一次迭代需要 17 个周期

(3) 调整后的指令序列为

```

fld  f2, 0(a0)
fld  f4, 0(a1)
fdiv.d f8, f0, f2
fmul.d f2, f6, f2
fadd.d f4, f0, f4
fadd.d f10, f8, f2
fsd   f10, 0(a0)
fsd   f4, 0(a1)
addi  a0, a0, 8
addi  a1, a1, 8
sub   x20, x4, a0
bnz   x20, Loop

```

一次迭代需要 16 个周期。

10. 解: 重命名后的指令序列为

```

fld  T9, 0(a0)
fmul.d T10, T0, T2
fdiv.d T11, T9, T2
fld  T12, 0(a1)
fadd.d T13, T0, T12
fsub.d T14, T8, T13
fsd  T15, 0(a1)

```


11. 答: 显式重命名是在每次需要重命名时, 都会为操作数分配新的物理寄存器, 并将这个新的寄存器标记为操作数的名称。物理上的寄存器堆具有的真实寄存器数目比ISA定义的寄存器数目更多。

隐式重命名是指采用“重命名表”的方式, 将虚拟寄存器重命名为物理寄存器, 并在运行时更新重命名表。物理寄存器数量和ISA寄存器数量相同。

显式重命名优点为可以在不依赖编译器的情况下, 通过硬件的方式解决指令重排问题, 对编译器的支持性要求较低。缺点为进行重命名操作, 可能会影响性能。

隐式重命名的优点为实现起来简单, 对硬件的要求也比较低, 缺点为额外的寄存器映射关系会增加程序的代码或文件的大小, 编译器的复杂度提升。

对于显式重命名, 一种可能的实现方式是使用物理寄存器表和逻辑寄存器表; 对于隐式重命名, 则是使用重命名表。