

9

(1)

$$4+11+\cancel{5}+4+\cancel{-1}+$$

$$4+11+5+3+1+\cancel{2}+1+0+1+1+2$$

$$\approx 31$$

(2)

~~$4+\cancel{11}+5+3+2+1+2=28$~~

$4+11+8+3+2+1+2=27$

~~$4+\cancel{11}+\cancel{5}+3+2+2+1+2+1+3=22$~~

(3) Loop:  $f(d, f_2, 0(a_0))$  ④ ~~$fdiv.d f_4, 0(a_1)$~~  ④ ~~$fadd.d f_4, f_0, f_4$~~  ③ ~~$fdiv.d f_8, f_0, f_2$~~  ⑦ ~~$addi a_0, a_0, 8$~~  ① ~~$addi a_1, a_1, 8$~~  ① ~~$sub x_{20}, x_4, a_0$~~  ① ~~$fsub.d f_4, 0(a_1)$~~  ② ~~$fmlhd.d f_2, f_6, f_2$~~  ⑤ ~~$fadd.d f_{10}, f_8, f_2$~~  ③ ~~$fsub.d f_{10}, 0(a_0)$~~  ② ~~$bnez x_{20}, \text{Loop}$~~  ①

$$4+11+5+3+2+2-1=26$$

10. Loop:  $f1d$   $f4, 0$   
RT

$f1c$

$a_1 f_2 f_4 f_6 f_8 \text{ to } T_{17} \sim T_{63}$

$T_9 \bar{T}_{10} T_{11} T_{12} T_{13} T_{14}$

$f1d T_{15}, 0(T_9)$

$fmod. d T_{16}, \bar{T}_{14}, T_{10}$

$fdiv. d T_{17}, T_{15}, T_{16}$

$f1d \underline{T_{18}, 0(T_7)}$

RT:

$a_0 a_1 f_0 f_2 f_4 f_6 f_8 8, 00 f1c$

$\bar{T}_9 T_9 \bar{T}_{10} T_{11} \bar{T}_{13} \bar{T}_{14} \bar{T}_{15} 8, 00 \bar{T}_{18} \sim T_{63}$

Loop:  $f1d \bar{T}_{16}, 0(T_9)$

$fmod. d T_{17}, T_{11}, T_{12} 8, 00$  end

$fdiv. d \bar{T}_{18}, \bar{T}_{16}, T_{17} 8, 00$   $\bar{T}_{17}$  but

$f1d T_{19}, 0(\bar{T}_{10}) 8, 00$   $\bar{T}_{19}$  but

$fadd. d T_{20}, T_{11}, T_{19} 8, 00$   $\bar{T}_{19}$  but

$fsub. d \bar{T}_{21}, T_{18}, T_{20} 8, 00$   $\bar{T}_{20}$  end

$f9d T_{22}, 0(\bar{T}_{10}) 8, 00$   $\bar{T}_{22}$  end

11 区别：显式重命名，确保物理寄存器数大于工规定义的寄存器数目，其中可存放任意指令结果

隐式：物理寄存器数与工规规定保持一致，但其中仅存放已经最终写回的指令结果。处于推测状态的指令值由一些其他结构保护

显式重命名是指编译器在编译时将虚拟寄存器直接映射到物理寄存器，需要硬件支持大量的物理寄存器，因此需要更多的芯片面积和功耗，但是由于指令中使用~~虚~~<sup>物理</sup>拟寄存器。这种方式不需要所以可以减少进行重命名的操作，这种方法不需要修改指令的寄存器操作数，可以更好地利用有限的物理寄存器，因此需要较少的芯片面积和功耗。但是，在运行时需要频繁地进行重命名操作，可能会导致额外的延迟和开销。

现代处理器会采用隐式重命名方式，并配合一些优化策略提高执行效率