(1)

fld f2,0(a0)	1	2-5	6	7
fdiv.d f8,f0,f2	8	9-19	20	21
fmul.d f2,f6,f2	22	23-27	28	29
fld f4,0(a1)	23	24-27	28	29
faddd f4,f0,f4	30	31-33	34	35
faddd f10,f8,f2	31	32-34	35	36
fsd f10,0(a0)	37	38 39	40	41
fsd f4,0(a1)	38	39 40	41	42
addi a0,a0,8	42	43	44	45
addi a1,a1,8	43	44	45	46
sub x20,x4,a0	46	47	48	49
bnz x20,Loop	50	51 52	53	54

3+49个周期.

(2)

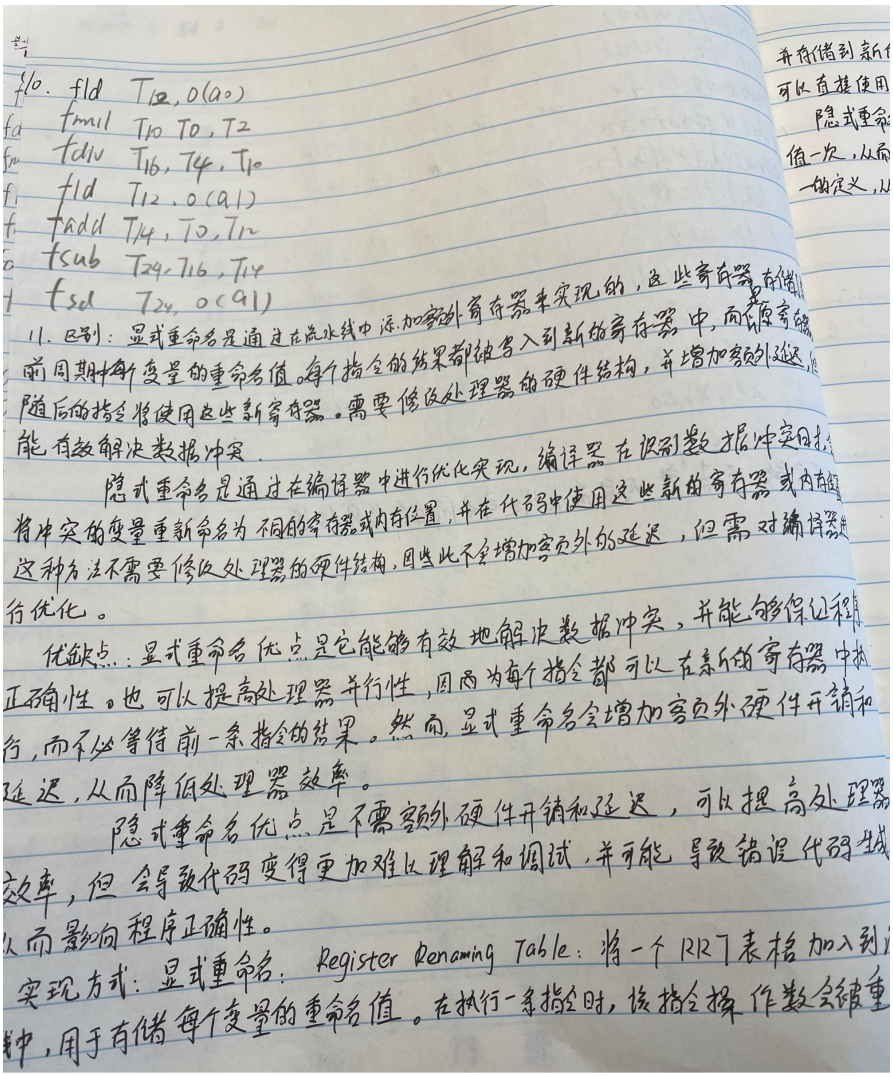
fld f2,0(a0)	1	2-5	6	7
fdiv.d f8,f0,f2	1	8-18	19	20
fmul.d f2,f6,f2	2	8-12	13	14
fld f4,0(a1)	2	3-6	7	8
faddd f4,f0,f4	3	9-11	12	13
faddd f10,f8,f2	3	21-23	24	25
fsd f10,0(a0)	4	26,27	28	29
fsd f4,0(a1)	4	14,15	16	17
addi a0,a0,8	5	30	31	32
addi a1,a1,8	5	18	19	20
sub x20,x4,a0	6	33	34	35
bnz x20,Loop	6	36	37	38

8个周期.

(3)

fld f2,0(a0)
fld f4,0(a1)
faddd f4,f0,f4
fdiv.d f8,f0,f2
fmuld f2,f6,f2
faddd 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

不会做. 不清楚 两段流水线之间如何进行信息传递.



fld T_{12}, 0(a_0)
fmul T_{10} T_0, T_2
fdiv T_{16}, T_4, T_{10}
fld T_{12}, 0(a_1)
fadd T_{14}, T_0, T_{12}
fsub T_{24}, T_{16}, T_{14}
fsd T_{24}, 0(a_1)

11. 区别：显式重命名是通过在流水线中添加额外寄存器来实现的，这些寄存器存储前一周期指令变量的重命名值。每个指令的结果都被写入到新的寄存器中，而后续指令随后的指令将使用这些新寄存器。需要修改处理器的硬件结构，并增加额外的延迟，能有效解决数据冲突。

隐式重命名是通过在编译器中进行优化实现，编译器在识别数据冲突时将冲突的变量重新命名为不同的寄存器或内存位置，并在代码中使用这些新的寄存器或内存这种方法不需要修改处理器的硬件结构，因此不会增加额外的延迟，但需对编译器行优化。

优缺点：显式重命名优点是它能够有效地解决数据冲突，并能够保证程序正确性。也可以提高处理器并行性，因为每个指令都可以在新的寄存器中执行，而不必等待前一条指令的结果。然而，显式重命名会增加额外硬件开销和延迟，从而降低处理器效率。

隐式重命名优点是不需额外硬件开销和延迟，可以提高处理器效率，但会导致代码变得更加难以理解和调试，并可能导致错误代码生成从而影响程序正确性。

实现方式：显式重命名：Register Renaming Table：将一个RRT表格加入到中，用于存储每个变量的重命名值。在执行一条指令时，该指令操作数会被重

并存储到新的寄存器中，同时该指令结果也会被存储到新寄存器中。这样，后续指令可以直接使用这些新的寄存器，从而避免了数据冲突。

隐式重命名：Static Single Assignment (SSA)：通过在编译期间将每个变量赋值一次，从而实现了变量的唯一定义。在执行一条指令时，该指令的操作数可以直接使用其的定义，从而避免数据冲突。