

9.10.11

9. 指令序列.

执行周期 (S表暂停, 后接暂停-周期数)
S1即表示在下一周期发射新指令

Loop: fld	f2.0(a0)	4
fdivd	f8, f0, f2	(S4) 11
fmul.d	f2, f6, f2	(S1) 5
fld	f4.0(a1)	(S1) 4
fadd.d	f4, f0, f4	(S4) 3
fadd.d	f10, f8, f2	(S1) 3
fsd	f10.0(a0)	(S3) 2
addi	a0, a0, 8	(S2) 1
addi	a1, a1, 8	(S1) 1
sub	x20, x4, a0	(S1) 1
bnz	x20, Loop	(S1) 2

$$\text{总周期 } N = 4 + 11 + 5 + 3 + 4 + 2 + 1 + 1 + 1 + 2 = 31 \text{ 个周期}$$

(2) 若采用双发射.

执行周期

4

$$\text{总周期 } N = 4 + 11 + 5 + 2 + 1 + 1 + 2$$

(S4) 11

$$= 26 \text{ 个周期}$$

(S1) 5

4

(S1) 3

3

(S3) 2

(S2) 1

(S1) 1

(S1) 2

	Tsd		
(3) Loop.	fld	f2, 0(a0)	4
	fld	f4, 0(a1)	4
	fdiv.d	f8, f0, f2	(s1) 11
	fadd.d	f4, f0, f4	(s1) 3
	fsd	f4, f0 , 0(a1)	(s2) 2
	addi	a1, a1, 8	(s2) 1
	fadd.d	f10, f8, f2	(s6) 3
	fsd	f10, 0(a0)	(s3) 2
	fadd.d addi	a0, a0, 8	(s2) 1
	sub	x20, x4, a0	(s1) 1
	bnz	x20, Loop	(s1) 2

$$\begin{aligned}
 \text{总周期 } N &= 4 + (11 - 4 + 1) + 3 + 2 + 1 + 1 + 2 \\
 &= 21 \text{ 个周期}
 \end{aligned}$$

原 f_k 对应 T_k		
10. fld	$T_9, 0(a0)$	(f_4 改为 T_9)
fmul.d	$T_{10}, \overset{T_0}{\cancel{f_0}}, T_{12}$	(f_2 改为 T_{10}) 原 f_0, f_2 对应 T_0, T_2
fdiv	T_{11}, T_9, T_{10}	(f_8 改为 T_{11} , f_4, f_2 对应 T_9, T_{10})
fld	$T_{12}, 0(a1)$	(f_4 改为 T_{12})
fadd.d	T_{13}, T_0, T_{12}	(f_6 改为 T_{13} , f_0, f_4 对应 T_0, T_{12})
fsub.d	T_{14}, T_{11}, T_{13}	(f_8 改为 T_{14} , 原 f_8, f_6 对应 T_{11}, T_{13})
fsd	$T_{14}, 0(a1)$	(f_8 对应 T_{14})

11. 显式重命名	隐式重命名
区别: 需要显式地标识每个寄存器, 并为每个操作数分配唯一寄存器名	不需要显式地标识每个寄存器 由处理器在运行时自动分配寄存器
优点: 运行时不需要执行寄存器重命名的开销 寄存器名已被分配好了	动态避免数据数冲突, 减轻编程负担
缺点: 需显式标识寄存器名, 工作量较大	在运行时执行寄存器重命名, 有额外开销
实现: 使用编译器来为每个操作数分配 唯一的寄存器名	使用硬件来维护寄存器分配表 并在运行时动态分配寄存器