HW04

PB19071405 王昊元

2022 年 04 月 26 日

1. a. 未进行调度的情况下和已调度的情况下的指令顺序及停顿如下：

时钟周期数	未调度	已调度
1	DADDIU R4,R1,#800	DADDIU R4,R1,#800
2	L.D F2,0(R1)	L.D F2,0(R1)
3	Stall	L.D F6,0(R2)
4	MUL.D F4,F2,F0	MUL.D F4,F2,F0
5	L.D F6,0(R2)	DADDIU R1,R1,#8
6	Stall	DADDIU R2,R2,#8
7	Stall	DSLTU R3,R1,R4
8	Stall	Stall
9	Stall	Stall
10	ADD.D F6,F4,F6	ADD.D F6,F4,F6
11	Stall	Stall
12	Stall	Stall
13	Stall	BNEZ R3,foo
14	S.D F6,0(R2)	S.D F6,0(R2)
15	DADDIU R1,R1,#8	
16	DADDIU R2,R2,#8	
17	DSLTU R3,R1,R4	
18	Stall	
19	BNEZ R3,foo	
20	Stall	

未调度时，结果向量 Y 中每个元素的执行时间，也就是循环的时钟周期为 19，调度后为 13。为使处理器硬件独自匹配调度编译器所实现的性能改进，时钟频率应当为原来的 $\frac{19}{13} = 1.46$ 倍。

b. 展开 3 次可消除循环开销，指令调度结果如下：

时钟周期数	指令
1	DADDIU R4,R1,#800
2	L.D F2,0(R1)
3	L.D F6,0(R2)
4	MUL.D F4,F2,F0
5	L.D F2,8(R1)
6	L.D F10,8(R2)
7	MUL.D F8,F2,F0
8	L.D F2,16(R1)
9	L.D F14,16(R2)
10	MUL.D F12,F2,F0
11	ADD.D F6,F4,F6
12	DADDIU R1,R1,#24
13	ADD.D F10,F8,F10
14	ADD.D R2,R2,#24
15	DSLTU R3,R1,R4
16	ADD.D F14,F12,F14
17	S.D F6,-24(R2)
18	S.D F10,-16(R2)
19	BNEZ R3,foo
20	S.D F14,-8(R2)

由上表可知，处理 Y 中的 3 个元素的执行时间为 19 个周期，即每个元素的执行时间为 $\frac{19}{3}$ 个周期。

2. 如下图所示：

迭代	指令	发射	执行/存储器访问	写 CDB	注释
1	L.D F2,0(R1)	1	2	3	
1	MUL.D F4,F2,F0	2	4	19(=4+15)	等待 F2 写回
1	L.D F6,0(R2)	3	4	5	
1	ADD.D F6,F4,F6	4	20	30(=20+10)	等待 F4 写回
1	S.D F6,0(R2)	5	31		等待 F6 写回
1	DADDIU R1,R1,#8	6	7	8(=7+1)	
1	DADDIU R2,R2,#8	7	8	9(=8+1)	
1	DSLTU R3,R1,R4	8	9	10(=9+1)	
1	BNEZ R3,foo	9	11		等待 R3 写回
2	L.D F2,0(R1)	10	12	13	等待跳转结果
2	MUL.D F4,F2,F0	11	19	34(=19+15)	等待乘法器空闲
2	L.D F6,0(R2)	12	13	14	
2	ADD.D F6,F4,F6	13	35	45(=35+10)	等待 F4 写回
2	S.D F6,0(R2)	14	46		等待 F6 写回
2	DADDIU R1,R1,#8	15	16	17(=16+1)	
2	DADDIU R2,R2,#8	16	17	18(=17+1)	
2	DSLTU R3,R1,R4	17	18	19(=18+1)	
2	BNEZ R3,foo	18	20		等待 R3 写回
3	L.D F2,0(R1)	19	20	21	
3	MUL.D F4,F2,F0	20	34	49(=34+15)	等待乘法器空闲
3	L.D F6,0(R2)	21	22	23	
3	ADD.D F6,F4,F6	22	50	60(=50+10)	等待 F4 写回
3	S.D F6,0(R2)	23	61		等待 F6 写回
3	DADDIU R1,R1,#8	24	25	26(=25+1)	
3	DADDIU R2,R2,#8	25	26	27(=26+1)	
3	DSLTU R3,R1,R4	26	27	28(=27+1)	
3	BNEZ R3,foo	27	29		等待 R3 写回

PS: 因为 EX 和 MEM 在同一时钟周期完成，所以表中执行指令的周期和访问存储器的周期合并。

- 第 1 次迭代周期为 31(=31-1+1) 周期
- 第 2 次迭代周期为 37(=46-10+1) 周期
- 第 3 次迭代周期为 43(=61-19+1) 周期

3. (1)

$$
\begin{aligned}
\text{CPI} &= \text{没有分支的 CPI} + \text{分支预测的 CPI} \\
&= 1 \times 85\% + \text{平均一个分支分支预测的 CPI} \times 15\% \\
&= 0.85 + \text{命中的 CPI} + \text{未命中的 CPI} \times 0.15 \\
&= 0.85 + (90\% \times (90\% \times 1 + 10\% \times 4) + 10\% \times 3) \times 0.15 \\
&= 1.0705
\end{aligned}
$$

(2)

$$\begin{aligned}\text{CPI'} &= \text{没有分支的 CPI} + \text{分支预测的 CPI} \\ &= 1 \times 85\% + 2 \times 15\% \\ &= 1.15\end{aligned}$$

采用分支目标缓冲执行速度更快。

4. 因为只考虑无条件转移指令，所以当缓冲命中时，则等同于预测正确的情况。

$$\text{CPI} = \text{没有分支的 CPI} + \text{分支预测的 CPI}$$

由

$$\text{CPI}_1 = 1 \times 95\% + \text{CPI}_{\text{jump}} \times 5\% = 1.1$$

可知 $\text{CPI}_{\text{jump}} = 3$

则有

$$\text{CPI}_2 = 1 \times 95\% + (90\% \times 1 + 10\% \times 3) \times 5\% = 1.01$$