

计算机体系结构Homework_3

38 + 60
= 98

Problem 1

Table 2的更新如下:

PC	Branch	Outcome	Prediction	Entry	Update
454	2	T	T	4	
543	3	NT	NT	6	
777	1	NT	NT	3	
543	3	NT	NT	7	
777	1	NT	T	3	NT with one misprediction
454	2	T	T	5	
777	1	NT	T	2	T with one misprediction
454	2	T	T	5	
543	3	T	NT	6	NT with one misprediction

由上表可知, 失败率为

$$3/9 * 100\% \approx 33.33\%$$

Table 3的更新如下:

PC	Branch	Outcome	Prediction	Entry	Update
454	0	T	T	0	
543	1	NT	T	4	T with one misprediction
777	1	NT	T	5	T with one misprediction
543	1	NT	NT	7	
777	1	NT	NT	7	
454	0	T	T	0	
777	1	NT	NT	7	
454	0	T	T	0	
543	1	T	NT	7	NT with one misprediction

由上表可知, 失败率为

$$3/9 * 100\% \approx 33.33\%$$

Problem 2

(a)

Op	dest	j	k	Issue	Read Oper	Exec Comp	Write Result
LD	F6	34	R2	1	2	3	4
LD	F2	45	R3	5	6	7	8
MULD	F0	F5	F2	6	9	19	20
MULD	F7	F2	F6	7	9	19	20
ADDD	F6	F8	F7	8	21	23	24

(b)

Time	Name	Busy	Op	F _i	F _j	F _k	Q _j	Q _k	R _j	R _k
	Integer	No								
10	Mult1	Yes	Mult	F0	F5	F2			Yes	Yes
10	Mult2	Yes	Mult	F7	F2	F6			Yes	Yes
	Add	Yes	Add	F6	F8	F7		Mult2	Yes	No

Problem 3

固定2个周期的分支惩罚的处理器器的CPI为：

$$1 + 2 * 20\% = 1.4$$

使用分支预测缓存的处理器器的CPI为：

①缓存命中但是预测错误，惩罚周期为4，概率为：

$$20\% * 85\% * (1 - 90\%) = 1.7\%$$

②缓存未命中，惩罚周期为3，概率为：

$$20\% * (1 - 85\%) = 3\%$$

所以使用分支缓存的处理器器的CPI为：

$$1 + 1.7\% * 4 + 3\% * 3 = 1.158$$

因此使用分支预测缓存的处理器器的速度是固定2个惩罚周期处理器器的速度的

$$1.4 / 1.158 \approx 1.209 \text{倍}.$$

Problem 4

(a)

由于不正确的预测和缓冲区遗漏的惩罚都为 2 个周期，因此可以推出在缓存中的预测正确率为 100%，因此当非条件分支指令出现在缓存中时，惩罚为 0 个周期。相对于在缓存中找到条件分支的目标地址，然后通过地址寻找指令，少了一次寻址，因此惩罚周期为-1.

(b)

对于没有使用分支折叠的情况，平均惩罚周期为：

$$10\% * (1 - 80\%) * 2 = 0.04$$

对于使用分支折叠的情况，平均惩罚周期为：

$$10\% * [80\% * (-1) + (1 - 80\%) * 2] = -0.04$$

因此使用分支折叠可以减少的平均惩罚周期数为：

$$0.04 - (-0.04) = 0.08$$

Problem 5

Iteration	Instructions	Issue	Executes	Memory access	Write CDB	comment
1	L.D F2,0(R1)	1		2	3	发送的第一条指令
1	MUL.D F4,F2,F0	2	4		19	等待F2; 3-4: RS; 5-18: 执行Mult
1	L.D F6,0(R2)	3		4	5	4: Loadbuf
1	ADD.D F6,F4,F6	4	20		30	等待F4; 5-20: RS; 21-29: 执行Add
1	S.D F6,0(R2)	5		31		等待F6; 6-31: Storebuf
1	DADDIU R1,R1,#8	6	7		8	
1	DADDIU R2,R2,#8	7	8		9	
1	DSLTU R3,R1,R4	8	9		10	
1	BNEZ R3,foo	9	11			等待R3
2	L.D F2,0(R1)	10		12	13	等待BNEZ; 11-12: Loadbuf
2	MUL.D F4,F2,F0	11	19		34	等待F2; 13-19: RS; 20-33: 执行Mult
2	L.D F6,0(R2)	12		13	14	13:Loadbuf
2	ADD.D F6,F4,F6	13	35		45	等待F4; 14-35: RS; 36-44: Add
2	S.D F6,0(R2)	14		46		等待F6; 15-46: Storebuf
2	DADDIU R1,R1,#8	15	16		17	
2	DADDIU R2,R2,#8	16	17		18	
2	DSLTU R3,R1,R4	17	18		20	
2	BNEZ R3,foo	18	20			等待R3

