

$$3.5 \text{ 方案 A: } CPI = 0.85 + 0.15 \times (0.1 \times 4 + 0.9 \times (0.1 \times 5 + 0.9)) = 1.099$$

$$\text{方案 B: } CPI = 0.85 + 0.15 \times 3 = 1.3$$

$$\text{则方案 A 比方案 B 快 } \frac{1.3 - 1.099}{1.099} \times 100\% = 18.29\%$$

3.12 (1) for ($i=0$; $i<10000$; $i++$) {

 if ($i \% 2 == 0$) {
 /* Code A */

}

 if ($i \% 5 == 0$) {
 /* Code B */

(2) B1 指令等价为 $i \% 2 != 0$, 跳转比例为 50%, (向后)

B2 指令等价为 $i \% 5 != 0$, 跳转比例为 80%, (向后)

B3 指令为 $\text{for } (i=0; i < 10000; i++)$ 的中间一条, 跳转比例为 99.99%, (向前)

(3) 准确率分别为: B1: 50%, B2: 20%, B3: 99.99%

3.13 (1) $0xe44 = (111001000100)_2$ $0xe84 = (111010000100)_2$

$0xec0 = (111011000000)_2$

可以发现 $k=5$, 即后 7 位, 三个地址各不相同, 因此, k 最小值为 5

(2) 当 $N \geq 2$ 时, 程序稳态后 B1 对应的计数器值始终在 0, 1 间跳转, 则 B1 预测准确率始终为 50%; B3 对应的计数器值始终为 9999 周期为 $2^N - 1$, 1 周期为 $2^N - 2$, 预测准确率始终为 99.99%;

$N=2$ 时 B2 预测准确率为 80%;

则 N 的最小值为 2, 预测准确率为 B1: 50%, B2: 80%, B3: 99.99%

3.14 程序稳态时 B1 分支历史为 1010..., B2 分支历史为 1111011110...,

B3 分支历史表为 01...1101...11

则最少需要 $H=9999$ 位历史才能完全准确预测

3.15 全局分支历史为 0011110111110111010111111111...

M 的最小值为 29970, 此时 B3 欲写入 0 时分支历史表中能保存到 B3 前 9999 次历史(1)

3.16 方案 A 使用 1 bit 计数器, 外循环 P 次的预测准确率为 $\frac{Q-2}{Q}$, 即预测错误 2 P 次

方案 B 使用 Q 位局部分支历史表, 外循环 P 次的预测准确率为, 除了前 $2(Q-1)$ 次预测错误(未初始化外, 后续所有预测)完全正确

则当A的预测准确率优于B时, $2P < 2(Q-1) \Rightarrow P < Q-1$

3.17 Loop: (W a4, o(a3)) (1) B1与B2的跳转情况为: 首是首, 非首

addi a3, a3, 1 B1 = 01010101 错误预测4次

addi a1, a1, -1 B2 = 1111110 错误预测1次(稳态)

B1: beqz a4, B2 (非稳态时若B2表项预测器初值为0, 则)

addi a2, a2, 1 错误预测3次

B2: bnez a1, Loop (2) 全局分支历史索引表次为 0111011101110110

初值 a1=n, a2=0, a3=p 以下假设讨论的均为稳态情况

开始运行时 0预测器值 11 (预测器值 11)

11 (预测器值 10)

则接下来一次循环 16 次预测均为跳转, -共发生 5 次错误预测

(3) 2位全局分支历史表为: 10(0)、00(1)、01(1)、11(1)、11(0)、10(1)、01(1)、11(1)、
11(0)、10(1)、01(1)、11(1)、11(0)、10(1)、01(1)、11(0)

开始运行时 00 预测器值为 11, 01 预测器值为 11, 10 预测器值为 11,

11 预测器值为 00

则预测为 1110011001100110, -共发生 4 次错误预测

(4) 全局分支历史表位数低时与单独使用局部预测器相比没有太大优势, 位数高则能提高一定预测准确率, n很大时 (3) 预测器效果最好

(5) 此时程序不再有“稳态情况”, 三种分支预测的准确率要依 $P[]$ 的数据规律而定, 对(1)中的预测方法准确率几乎不变, 而当 n 较大, $P[]$ 变化较不规则时, 分支历史预测的准确率会有稳定的上升

3.18 指令引发的异常种类不同，从而会发生在五级流水的不同阶段。如非法指令异常在ID触发，内存访问异常可能在IF、EX、MEM等阶段触发，因而后执行的指令有可能先触发异常。首先异常发生时处理器会停止执行指令，并保留此时的上下文信息，同时借助ROB，在指令写回后增加提交状态，确保执行且异常处理完成的指令才可提交并修改寄存器，以达到精确异常。

		Decode (ROB enqueue)	Issue	WB	Committed	Opcode	rd	rs1	rs2
3.20	I1: fild f1, S(a0) (1)	I1 0	1	2	3	fild	T0	a0	-
	I2: fmul.d f2, f1, f0	I2 1	3	13	14	fmul.d	T1	T0	f0
	I3: fadd.d f3, f2, f0	I3 2	14	16	17	fadd.d	T2	T1	f0
	I4: addi a0, a0, 8	I4 3	4	6	7	addi	T3	a0	-
	I5: fild f1, S(a0)	I5 4	6	7	8	fild	T4	T3	-
	I6: fmul.d f2, f1, f1	I6 5	13	23	24	fmul.d	T5	T4	T4
	I7: fadd.d f2, f2, f3	I7 6	24	26	27	fadd.d	T6	T5	T2
	RAW冲突: I1, I2, I2, I3, I4, I5, I5, I6, I6, I7								

		Decode (ROB enqueue)	Issue	WB	Committed	Opcode	rd	rs1	rs2
(2)	I1	I1 0	1	2	3	fild	T0	a0	-
	I2	I2 1	3	13	14	fmul.d	T1	T0	f0
	I3	I3 4	14	16	17	fadd.d	T2	T1	f0
	I4	I4 15	16	18	19	addi	T3	a0	-
	I5	I5 18	19	20	21	fild	T4	T3	-
	I6	I6 20	21	31	32	fmul.d	T5	T4	T4
	I7	I7 22	32	34	35	fadd.d	T6	T5	T2