

5. 考虑一个深度流水线处理器，无分支指令时其基本 CPI 为 1。对于分支指令采用两种方案，方案 A 使用一个分支目标缓存 (BTB)，缓存缺失代价为额外 3 个周期，缓存命中但预测错误的代价为额外 4 个周期，缓存命中且预测正确则无分支代价。假设这个 BTB 的命中率为 90%，预测正确率为 90%。方案 B 不使用分支预测，分支代价固定为额外 2 个周期。假设分支频率为所有指令的 15%，则处理器采用方案 A 比采用方案 B 快多少？

$$\text{方案 A: } CPI_a = 0.85 + 0.15 \left(0.1 \times 4 + 0.9 \times 0.1 \times 5 + 0.9 \times 0.9 \times 1 \right) \\ = 1.099$$

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

$$S = \frac{CPI_b}{CPI_a} = 1.183$$

∴ A 的速度是 B 速度的 1.183 倍
快了 18.3%

12. 考虑如下的代码片段:

```

li      a0,0
li      a4,10000
0x0000: addi   a1,a0,0
Loop:   addi   a3,a0,2
        rem    a2,a1,a3
0xe44:  bne    a2,a0,Rem2 //B1
        #...CodeA
Rem2:   addi   a3,a0,5
0xe80:  rem    a2,a1,a3
0xe84:  bne    a2,a0,End   //B2
        #...CodeB
End:    addi   a1,a1,1
0xec0:  bne    a1,a4,Loop  //B3

```

$a_0 = 0$
 $a_4 = 10000$
 $a_1 = 0$
 $a_3 = 2$

if ($a_1 \% a_3 \neq 0$) ; to Rem2

$a_3 = 5$
 if ($a_1 \% a_3 \neq 0$) $a_1 = a_1 + 1$

- 1) 写出与该汇编代码功能一致的 C 语言代码。
- 2) 无分支预测时, 上述代码中的三条 bne 指令发生跳转的比例分别是多少?
- 3) 引入一个静态分支预测器, 该预测器对向前跳转总是给出“跳转”预测, 对向后跳转总是给出“不跳转”预测, 则上述代码中的三条 bne 指令的预测准确率分别是多少?

int i = 0

1) while (i < 10000)
 { if (i % 2 == 0) # CodeA
 if (i % 5 == 0) # CodeB,
 i++;
 }

2) B₁: 比例 50% ($i \% 2 \neq 0$)

B₂: 比例 80% ($i \% 5 \neq 0$)

B₃: 比例 99.9% (从 1 到 10000, 共 9999 次跳转)

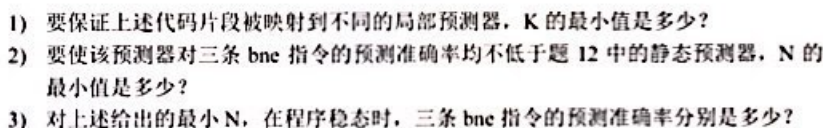
3) B₁ 总是预测跳转, 准确率 50%

B₂ 总是预测跳转, 准确率 80%

B₃ 总是预测不跳转, 准确率 0.01%

1) $0xe44 = (11101100)_2$
 $0xe84 = (11101000100)_2$
 $0xec0 = (111011000000)_2$
 $\therefore K$ 的最小值为 5 ($k \leq 4$ 无解)

13. 仍考虑题 12 中的代码片段, 现引入局部预测器, 如下图所示。该预测器使用 PC 的第 $[(K+2):3]$ 共 K 位索引一张预测器表, 该表的每个表项是一个 N -bit 的计数器, 计数器的最高位用于预测是否跳转 (1 为跳转, 0 为不跳转), 并根据实际跳转结果更新计数器的值 (跳转自增 1, 增至 2^N-1 后不再变化; 不跳转自减 1, 减至 0 后不再变化)。假设所有计数器的初始值均为 0。



数 $00101 \dots 01$ $N=1$: 修正 $\overline{1000}$

$B_2: FTTT FTTT FTTT \dots$

$$= 2: 00, 00 \quad 01, 10 \quad 11, 11 \quad 10, 11 \quad 11, 11 \quad 11, 11 \quad 10, 11 \quad 11, 11$$
[illegible]

易知 $N = 2$ 时推确并必全大于 0.01%

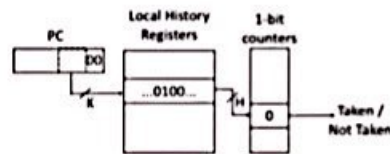
3) $\beta_1: 50\%$

B₂ : 80%

$B_2: 99.99\%$

14. 仍考虑题 12 中的代码片段，现引入局部分支历史，如下图所示。该预测器使用 PC 的第 $[K+2:3]$ 共 K 位索引一个局部分支历史表，其每个表项是一个 H 位的局部分支历史，

该 H 位的历史被进一步用于索引一张由 1 比特计数器构成的预测表（1 为跳转，0 为不跳转）。计数器会根据实际跳转结果进行更新。



假设 K 的值是够大，使得上述代码片段中的不同分支会被映射到局部分支历史表中的不同位置，则为了使得三条 `bne` 指令都能在程序稳态时被完全准确地预测， H 的最小值是多少？

$H=1$: "0" → 1, "1" → 0

$B_1: i \% 2 \neq 0$ 则跳 → 010101 ... 0101

$H=2$: "11" 可得 1, 也可得 0, X

$B_2: i \% 5 \neq 0$ 则跳 → 01110111 ... 0111

$H=3$: "111" 可得 1, 也可得 0, ✓

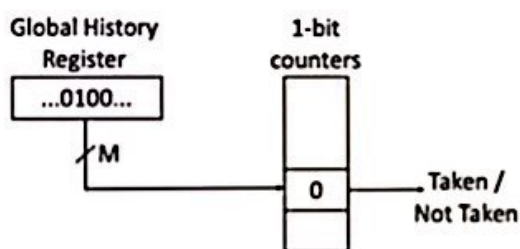
$H=4$: "0111", "1110", "1101", "1011" → 1, "1111" → 0 $H=4$ ✓

$B_2: i \neq 1000$ 则跳 → 11111111 ... 1110

H 位代表的逻辑：状态机：前 H 次正确

综上 H 最小值为 4 (至少包含最长连续预测)

15. 仍考虑题 12 中的代码片段，现引入全局分支历史，如下图所示。该预测器拥有一个 M 位的 GHR，记录了程序中任意分支的跳转历史。当一个新分支被执行时，跳转分支使得 GHR 左移 1 位并在末位写入 1，未跳转分支则使得 GHR 左移 1 位并在末位写入 0。GHR 被用于索引一张单比特计数器构成的预测表（1 为跳转，0 为不跳转）。计数器会根据实际跳转结果进行更新。



为了使得三条 bnc 指令都能在程序稳态时被完全准确地预测，M 的最小值是多少？

[illegible] B, B_2, B_3

液相 ($F=0, T=1$)

FFT
TTF
FTT
TTF
FTT
TFT
FTT
TTF
FTT
TTF
FFT
TTF
FTT
TTF
FTT
TFT

1
7
3
7
3
5
3
7
3
7
1
7
3
7
3
5

FFI TII FFI TTI FFI ICI FFI TII FFI TTI FFI TI FFI TI FFI TI FFI

FF T 7 3-7-FTT

FTT 7-3-7- FFT

TTT-3-7-3-TF7

TFJ 3-7-3-TT7

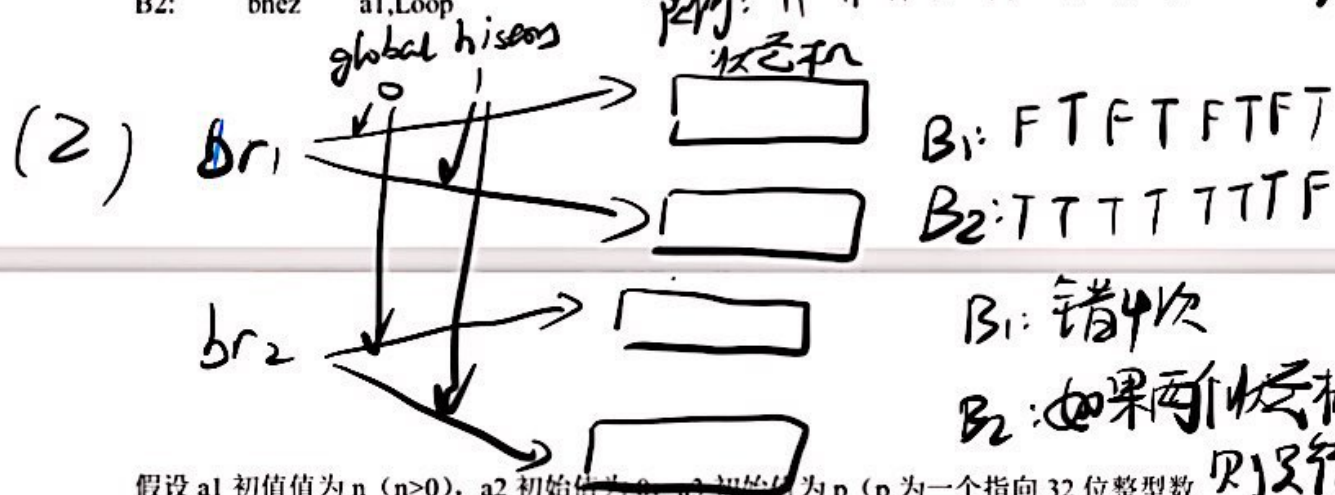
可知 $M \geq 12$

17. 考虑如下的指令序列:

```

Loop:  lw    a4,0(a3)
      addi   a3,a3,4
      addi   a1,a1,-1
B1:    beqz  a4,B2
      addi   a2,a2,1
B2:    bnez  a1,Loop
    
```

(1) $B_1: F T F T F T F T$
 预测: 00 00 01 00 01 00 01 00 01 错误4次
 $B_2: T T T T T T T F$
 预测: 11 11 11 11 11 11 11 11 错误1次



假设 $a1$ 初值为 n ($n > 0$), $a2$ 初始值为 0, $a3$ 初始值为 p (p 为一个指向 32 位整型数组首地址的指针)。

- 1) 假设处理器使用 2 位局部预测器, 分支 $B1$ 和 $B2$ 映射到不同的预测器表项。若 $n=8$ 且数组的数据模式为 $p[] = \{1, 0, 1, 0, 1, \dots\}$ 。则上述代码执行过程中一共会发生多少次错误预测?
- 2) 现引入 1 位的全局分支历史, 1) 中的其他假设不变, 则上述代码执行过程中一共会发生多少次错误预测?
- 3) 若改为 2 位的全局分支历史表, 1) 中的其他假设不变, 则上述代码执行过程中一共会发生多少次错误预测?
- 4) 比较上述结果, 分析在该情境中全局分支历史表的位数对预测准确率有怎样的影响? 当 n 非常大时, 上述哪种预测器表现最好?
- 5) 当数组 $p[]$ 的数据模式变为在 0 和 1 之间以均等概率随机取值时, 4 中的结论有什么变化?

2 位全局分支历史, 1 个分支历史表 0123 号
 $F (T) (T) (T) (T) (T) (T) (T) (T) T T T F T T F$
 错误 3 次

(4) 全局
 (5) 两种预测器表现不好

18. 解释为什么即使在顺序的 5 级 RISC 流水线中, 指令引发的异常也可能会乱序产生? 为了支持精确的异常处理, 流水线是如何做到对乱序产生的异常进行 (按程序顺序的) 顺序处理的?

① 因为每条指令执行的周期不同, 且指令与指令之间还会存在数据依赖、控制冲突等不可控因素

② 指令在 WB 阶段必须保证按照源程序的顺序进行写回。这样如果有指令产生中断或异常, 处理器可以禁止在程序顺序上位于这条指令之后的指令的写回操作。即乱序执行顺序写回。当指令被发射时, 处理器在重排序缓冲区中保留个记录, 当指令执行完成后需要写回时, 若在它之前有未写回的指令, 则该指令继续在重排序缓冲区等待。这样流水线一旦发生中断需要回滚, 没有执行写回操作的指令不会对处理器或存储系统的数据造成影响

20. 考虑一个拥有浮点单元的单发射乱序处理器，该处理器包含以下假设：

- 处理器的浮点单元包含一个 2 运算周期的加法器、一个 10 运算周期的乘法器，和一个单执行周期的浮点加载/存储单元，加法和乘法器均是完全流水化的。
- 当发生写回冲突时，更早的指令会获得优先写回权。
- 浮点指令的结果只能在写回阶段完成后被其他指令使用，整型指令的结果则可以前

馈。

- 处理器使用寄存器重命名，从 T0、T1、T2 起有不受限制的重命名寄存器可用。
- 译码级每周可以至多 1 条重命名后的指令添加到 ROB 中，指令通过 ROB 顺序提交且每周至多提交 1 条指令。指令能够被提交的最早时间是完成写回后的下一个周期。
- 忽略前端取指，指令经过译码、发射、执行和写回后即可完成执行并提交。

现考虑如下的指令序列：

I1:	fld	f1, 5(a0)	1
I2:	fmul.d	f2, f1, f0	10
I3:	fadd.d	f3, f2, f0	2
I4:	addi	a0, a0, 8	2
I5:	fld	f1, 5(a0)	10
I6:	fmul.d	f2, f1, f1	10
I7:	fadd.d	f2, f2, f3	2

$$T_3 = a0 + 8$$

- 如果 ROB 的深度是无限的，将下表补充完全。（部分结果已给出）

	周期				操作码	目标	源 1	源 2
	Decode (ROB enqueue)	Issue	WB	Committed				
I1	0	1	2	3	fld	T0	a0	—
I2	1	3	13	14	fmul.d	T1	T0	f0
I3	2	14	16	17	fadd.d	T2	T1	f0
I4	3	15	17	18	addi	T3	a0	—
I5	4	16	18	19	fld	T4	T3	—
I6	5	17	20	21	fmul.d	T5	T4	T4
I7	6	18	21	22	fadd.d	T6	T5	T2

- 如果 ROB 仅容纳 2 条指令，当一条指令提交后的下一周期该条目可以被新指令占据，重新将下表补充完全。（部分结果已给出）

	周期				操作码	目标	源 1	源 2
	Decode (ROB enqueue)	Issue	WB	Committed				
I1	✓ 0	1	2	3	fld	T0	a0	—
I2	✓ 1	3	13	14	fmul.d	T1	T0	f0
I3	✓ 4	14	16	17	fadd.d	T2	T1	f0
I4	15	16	18	19	addi	T3	a0	—
I5	18	19	20	21	fld	T0	T3	—
I6	20	21	31	32	fmul.d	T2	T0	T0
I7	22	32	34	35	fadd.d	T4	T1	T2