

5.

方案A:

分支成本对CPI的增加值如下:

命中率: 10% 命中但预测错误: 9% 预测正确: 81%

$$CPI_{\text{增加}} = 15\% (10\% \times 3 + 9\% \times 4 + 81\% \times 0) = 0.099$$

$$\therefore CPI = 1 + 0.099 = 1.099$$

方案B: $CPI = 15\% \times 2 + 1 = 1.3$

$$\therefore \text{加速比} = \frac{CPI_B}{CPI_A} = 1.18, \text{性能提升} = \frac{CPI_B - CPI_A}{CPI_B} \approx 15\%$$

12. 考虑如下代码片段:

li a0, 0

li a4, 10000

addi a1, a0, 0

Loop: addi a3, a0, 2

rem a2, a1, a3

0xe44: bne a2, a0, Rem2 // B1

Code A

Rem2: addi a3, a0, 5

rem a2, a1, a3

0xe84: bne a2, a0, End // B2

code B

End: addi a1, a1, 1

0xec0: bne a1, a4, Loop // B3

1) 写出功能一致的C代码

```
#include <stdio.h>
```

```
int main()
```

```
{
    int a0 = 0;
```

```
    int a4 = 10000;
```

```
    int a1 = a0;
```

```
    int a2, a3;
```

```
    for (a1 = 0; a1 < a4; a1++) {
```

```
        a3 = a0 + 2;
```

```
        a2 = a1 % a3;
```

```
        if (a2 == a0) {
```

```
            #Code A
```

```
        a3 = a0 + 5;
```

```
        a2 = a1 % a3;
```

```
        if (a2 == a0) {
```

```
            #code B
```

```
        }
```

```
    }
    return 0;
}
```

2) 在无分支预测时,上述代码中三条 bne 指令跳转的比例分别是多少?

B1: 由 $a2 = a1 \% a3$ 可知, B1 发生跳转的比例约为 $\frac{1}{2}$

B2: 由 $a3 = a1 \% a3 = a1 \% 5$ 可知, B2 跳转的比例约为 $\frac{4}{5}$

B3: 由于几乎每次都进入 for 循环体内,直到 $a1$ 增至 10000, 因此 B3 跳转的比例约为 1.

3) 引入一个静态分支预测器, 对向前跳转总是给出跳转预测, 向后总是不跳转, 则上述三条 bne 指令预测准确率分别是多少?

① B1: bne a2, a0, Rem2

准确率 = $\frac{1}{2}$

② B2: bne a2, a0, End

准确率 = $\frac{1}{5}$

③ B3: B3 为向前跳转, 准确率 = 99.99% $\approx 100\%$

13.1) B1 的 PC 值: 0xe49 = 110100001000

B2 的 PC 值: 0xe84 = 110110000100

B3 的 PC 值: 0xec0 = 110110000000

$K+2=7, K=5$, 要保证跳转正确 K 最小值为 5.

2) 对于 $N=1$ 的情况, B3 预测错误率会增加一次. N 的最小值应为 2.

3) B1: 准确率仍为 $\frac{1}{2}$ (50%)

B2: 准确率仍为 $\frac{4}{5}$ (80%)

B3: 准确率仍约为 99.99%

14. 循环中分支的规律如下:

第一次, $a_1=0$, B1 不跳转, B2 不跳转, B3 跳转

若令跳转 = 1, 不跳转 = 0, 有如下规律:

$a_1=0 \Rightarrow B_1=0, B_2=0, B_3=1$

$a_1=1 \Rightarrow B_1=1, B_2=1, B_3=1$

$a_1=2 \Rightarrow B_1=0, B_2=1, B_3=1$

$a_1=3 \Rightarrow B_1=1, B_2=1, B_3=1$

$a_1=4 \Rightarrow B_1=0, B_2=1, B_3=1$

$a_1=5 \Rightarrow B_1=1, B_2=0, B_3=1$

\vdots

∴ 要使 B_1 预测 100% 准确, 只有一种情况, $H=1$

要使 B_2 预测准确, 其跳车 5 次为一个循环, $H=4$

要使 B_3 预测 100% 准确, 一共循环 100000 次, 其间没有重复的规律, $H=9999$.

15.

根据汇编代码, 执行顺序为 $B_1 \rightarrow B_2 \rightarrow B_3 \rightarrow B_1 \dots$

要全部预测准确, 需有 9990 位

∴ M 最小值为 9990.

16. 外循环针对 i 的分支总是被忽略.

对内层循环, 首先经历 Q 次跳车, 接着不进入, 1 次不跳车, 以此规律循环重复 P 次

方案 A: 每个内部循环出错两次, 准确率 = $\frac{Q-1}{Q+1}$

方案 B: $\because H=Q$, 由于初始值为 0, \therefore 会错误 Q 次, 此后正确

$$\therefore \text{准确率} = \frac{P \times (Q+1) - Q}{P(Q+1)} = \frac{P + (P-1)Q}{P(Q+1)}$$

$$\therefore \text{方案 A 优于 B: } \frac{Q-1}{Q+1} > \frac{P + (P-1)Q}{P(Q+1)}$$

$$\therefore PQ - P > PQ + P - Q, Q > 2P$$

∴ 当 $Q > 2P$ 时方案 A 优于方案 B.

17. 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) a1初值为n, 则B2会跳转n次, 最后一次不跳转; B1跳转情况与数n有关
当 $n=8$, $p[i] = \{1, 0, 1, 0, 1, \dots\}$ 时, 对于B1, 分支在跳转与不跳转中
跳, 一共有4次错误

对于B2, 初始会错误2次, 最后错误1次, 共错误3次,

全部错误为7次

2) GHR记录 B2的跳转情况

对于B1, 第一次GHR=0, PHT=00, 正确一次, 此后错4次

对于B2, 开始时一共错误4次, 最后错一次, 共错误5次

\therefore 一次错误7次

3) 对于B1:

GHR	PHT	
① 00	00	正确
② 01	00	错误
③ 11	00	正确
④ 01	01	错误
⑤ 11	00	正确
⑥ 01	10	正确
⑦ 11	00	正确
⑧ 01	11	正确

01情况错误2次, 其余正确

对于B2, 同理有

	GHR	PHT	
①	00	00	错误
②	11	00	错误
③	10	00	错误
④	11	01	错误
⑤	10	01	错误
⑥	11	10	正确
⑦	10	10	正确
⑧	11	11	正确 错误

错误 6 次

共发生 8 次错误

4) 对于类似 B1 的跳变频率情况, 引入 GHR 会使正确率提升, 对于循环迭代则会增加较大的初始化成本, 正确率反而下降。当 n 非常大时, 这种影响会减小, 故 (3) 中预测器表现最好。

5) 当 PC 随机取值时, 跳 B1 跳转之间没有相关性, 所以不应使用 GHR; (1) 中的预测器表现最好。

18. 可能是由于异常指令处于流水线的不同阶段造成乱序产生。

流水线可以再增加一个提交阶段, 例如使用 ROB 顺序精确处理异常, 并在处理完成后恢复程序顺序执行, 来确保对乱序异常的顺序处理。

20.1)

	周期				操作码	目标	源1	源2
	Decode	Issue	WB	Committed				
I ₁	0	1	2	3	fld	T ₀	a ₀	-
I ₂	1	3	13	14	fmul.d	T ₁	T ₀	f ₀
I ₃	2	14	16	17	fadd.d	T ₂	T ₁	f ₀
I ₄	3	4	5	18	addi	a ₀	a ₀	-
I ₅	4	5	6	19	fld	T ₃	a ₀	-
I ₆	5	8	18	20	fmul.d	T ₄	T ₃	T ₃
I ₇	6	19	21	22	fadd.d	T ₅	T ₄	T ₂

2)若 ROB 仅容纳2条指令,表如下:

	周期				操作码	目标	源1	源2
	Decode	Issue	WB	committed				
I ₁	0	1	2	3	fld	T ₀	a ₀	-
I ₂	1	3	13	14	fmul.d	T ₁	T ₀	T ₀
I ₃	4	14	16	17	fadd.d	T ₂	T ₁	f ₀
I ₄	15	16	17	18	addi	a ₀	a ₀	-
I ₅	18	19	20	21	fld	T ₃	a ₀	-
I ₆	19	20	30	31	fmul	T ₄	T ₃	T ₃
I ₇	22	31	33	34	fadd.d	T ₅	T ₄	T ₂