

6. 为了避免地址冲突，高位作为组索引那么临近的几块地址很可能映射到同一个组中，产生地址冲突。

7. 可以提高内存到缓存的载入效率：缓存中数据块大小与内存中的页面大小相同，载入时按页载入即可。

8. (1)  $T = 1 + 3\% \times 110 = 4.3$  周期

(2)  $1GB = 2^{10} \times 64KB$  缓存命中概率  $\approx 0$ ，平均延时为 110 个周期

(3) 如果缓存是在经常访问的一部分数据内，则缓存命中率会大大提高，减少平均延时；否则平均延时可能更长。

(4) 命中率为  $X$   $1 + 110(1-X) = 105$   $X = \frac{3}{110} = 5.45\%$

故命中率至少为 5.45%

9.

| 编号 | 地址位数<br>Bit | 缓存大小<br>KB | 块大小<br>Byte | 相联度 | 组数量 | 组索引位数<br>Bit | 标签位数<br>Bit | 偏移位数<br>Bit |
|----|-------------|------------|-------------|-----|-----|--------------|-------------|-------------|
| 1  | 32          | 4          | 64          | 2   | 32  | 5            | 21          | 6           |
| 2  | 32          | 4          | 64          | 8   | 8   | 3            | 23          | 6           |
| 3  | 32          | 4          | 64          | 全相联 | 1   | 0            | 26          | 6           |
| 4  | 32          | 16         | 64          | 1   | 256 | 8            | 18          | 6           |
| 5  | 32          | 16         | 128         | 2   | 64  | 6            | 19          | 7           |
| 6  | 32          | 64         | 64          | 4   | 256 | 8            | 18          | 6           |
| 7  | 32          | 64         | 64          | 16  | 64  | 6            | 20          | 6           |
| 8  | 32          | 64         | 128         | 16  | 32  | 5            | 21          | 7           |

$$10: (1) 0.22 + p_1 \times 100 < 0.52 + p_2 \times 100 \Rightarrow p_1 - p_2 < 3 \times 10^{-3}$$

故  $p_1 - p_2 < 3 \times 10^{-3}$  时，A 优于 B

$$(2) 0.22 (1 + kp_1) < 0.52 (1 + kp_2) \Rightarrow 11p_1 - 26p_2 < \frac{15}{k}$$

故  $11p_1 - 26p_2 < \frac{15}{k}$  时，A 优于 B

11. 直接映射：索引为 4 位，即 16 进制的最后一一位为块选系，共发生 5 次块替换

2 路组相联：索引为 3 位， $0x1=0001$   $0x5=0101$  尾数为 5 的有 5 个， $5-2=3$ ，故共发生 3 次块替换

4 路组相联：索引为 2 位， $0x1=0001$   $0x5=0101$  尾数为 5 的有 5 个 尾数为 1 的有 2 个， $7-4=3$ ，故共发生 3 次块替换

8 路组相联：索引为 1 位， $0x1=0001$   $0x5=0101$  尾数为 5 的有 5 个 尾数为 1 的有 2 个， $7-8=1$ ，故不发生块替换

12. 数组为 96 位 int 型数组，4 个数字一块，块大小为 256+128 Byte，总实行 100 次遍历

缓存 A：由于数组依次读取，索引位有 3 位，故 0~31, 32~63 和 64~95 的索引相同。根据 LRU 替换策略，每次载入新的块都会发生块替换。由于是 4 位数字一个块，故缺失率为  $\frac{1}{4}$

缓存 B：索引位有 4 位，0~31, 64~95 位数的索引相同，32~63 位不发生缺失，故缺失率为  $\frac{1}{8}$

```
13. for (int j=0; j < 128; ++j) {
    for (int i=0; i < 64; ++i) {
        A[i][j] = A[j][i] + 1;
    }
}
```

14. 块大小为 32 Byte, 含 8 个 int 型数据, 4KB 大小的缓存有  $2^7$  个块。优化前的取址情况是在物理地址上每隔 15 个块取一个块, 取 64 次, 遍历后推演优化后的取址是全相邻顺序取址。

(1) 优化前: 组空引有 7 位, 每次重新载入块都要发生替换, 且非相邻取址, 故缓存缺失  $2^{13}$  次。

优化后: 相邻取址, 每个块载入时只有块中第一个数据发生缺失, 故缓存缺失  $2^{10}$  次。

(2) 优化前: 一次遍历取 64 个块, 可在下次遍历时复用, 故缓存缺失  $2^{10}$  次。

优化后: 与 (1) 相同, 缓存缺失  $2^{10}$  次。

(3) 优化前: 除了最后一小部分外要求几乎全覆盖, 需 32KB - 480B。

优化后: 理论上一个块的缓存即可, 即 32B。

15.

|     | input 数组 |      |      |      | output 数组 |      |      |      |
|-----|----------|------|------|------|-----------|------|------|------|
|     | 列 0      | 列 1  | 列 2  | 列 3  | 列 0       | 列 1  | 列 2  | 列 3  |
| 行 0 | miss     | miss | hit  | miss | miss      | miss | miss | miss |
| 行 1 | miss     | hit  | miss | hit  | miss      | miss | miss | miss |
| 行 2 | miss     | miss | hit  | miss | miss      | miss | miss | miss |
| 行 3 | miss     | hit  | miss | hit  | miss      | miss | miss | miss |