5、(1) 2^{64} byte，2^{52} 页，

(2) 2^{48} byte，2^{36} 页

(3) 因为虚拟地址64位但实际上一个进程不会用到如此多页，且每个进程实际所需分页表数不同，故使用多级页表可以根据实际进程创建页表占用空间，节省不必要开销

6、假设高位2位，中位2位

0000	若用高位索引，
0001	则连续的块将被
0010	映射到相同组中，
0011	使得无法很好地
0100	利用缓存的空间
0101	局部性，因为很多相
0110	邻地块争夺竞争
0111	

若用中位索引，则连续的块会映射到不同的组，使得充分最大地利用空间局部性

7. 虚拟内存子系页偏移位数，表示实际物理页的大小

选择将段将不同路对应到不同的物理页？

8、(1) $3\% \times 110 + 97\% \times 1 = 4.27$

(2) 随机访问，则访问速为 $\frac{64K}{1G} \times 1 + (1 - \frac{64K}{1G}) \times 110 \approx 110$。

(3) 程序的空间局部性越高，处理器性能越好。

(4) $\lambda \times 1 + (1-\lambda) \times 110 < 105$

解得 $\lambda > 4.59\%$

命中率大于 4.59%

9.

9. 根据给出的不同缓存配置，补全下表中缺失的字段。

编号	地址位数 Bit	缓存大小 KB	块大小 Byte	相联度	组数量	组索引位数 Bit	标签位数 Bit	偏移位数 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	20	7

10.

(1) $0.22 + p_1 \times 100 < 0.52 + 100 p_2$

$\Rightarrow p_1 - p_2 < 0.003$

(2) $0.22 + (k-1) \times p_1 \times 0.22 < 0.52 + (k-1) p_2 \times 0.52$

$\Rightarrow 0.22 p_1 - 0.52 p_2 < \frac{0.3}{k-1}$

11.

(1) 直接映射：

若 0x1001 → p_1

则 0x1005 → $p_1 + 4$

0x1021 → p_1 替换

0x1045 → p_1+4 替换

0x1305 → p_1+4 替换

0x2005 → p_1+4 替换

0x4005 → p_1+4 替换

} 5次

(2) 2路：

0x1001 → p_1

0x1005 → p_1+4

0x1021 → p_1

0x104f → p_1+4

0x13ff → p_1+4 替换

0x20ff → p_1+4 替换 } 3次

0xffff → p_1+4 替换

(3) 4路：

0x1001 → p_1

0x1005 → p_1

0x1021 → p_1

0x1045 → p_1

0x13ff → p_1 替换

0x2ee5 → p_1 替换 } 3次

0xffff → p_1 替换

(4) 8路：

0x1001 → p_1

0x1005 → p_1

0x1021 → p_1 无替换

⋮ ⋮

0xffff → p_1

12、

A: $256 \div 16 \div 2 = 8$ 组，16 块

int32_t 4字节

$i=0$ 时，j 从 0→63，miss $64 \times \frac{1}{4} = 16$ 次

$i=0$ 时，j 从 64→95，miss $32 \times \frac{1}{4} = 8$ 次

∴ 缺失率：$\frac{1}{4} = 25\%$

直接映射：

$256 \div 16 = 16$ 组，一块 16 字节，

每块可容纳 4 int，

同块 0~3 → 0

4~7 → 1

⋮ ⋮

60~63 → 15

64~67 → 0

⋮ ⋮

92~95 → 7

B: 直接映射

缺失率 $= \frac{24 + 16 \times 99}{100 \times 96} = 16.75\%$

13

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

14.

1) 优化前：$64 \times 128 = 2^{13}$

优化后：$64 \times 128 \times \frac{1}{8} = 2^{10}$

2) 优化前后都是：$64 \times 128 \times \frac{1}{8} = 2^{10}$

3) 优化前：$(127 \times 8 + 1) \times 32 = 32544$ B

优化后：32 B

15.

不清晰。

	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

16. (1) $512 \div 16 \div 2 = 16$,

而 $128 \times 4 \div 16 = 32$,

即 input[0][0] ~ input[0][63] 为, input[0][64] ~ input[0][127]

会分别映射在相同组不同块数, input[1]亦如此, 故

缺失率 25%, 命中率为 75%

(2) 不改, 块大小不变, 则每块仅会最少

miss 1次. 而原方案已做到

极致, 故无法提高命中率

(3) 可以, 如将块大小增加到

32 byte, 则缺失率减小

至 12.5%, 即命中率为 87.5%.