5. 1). 64位虚拟地址，空间大小 4G×4G，页大小 4KB.

则有 1M×4G 个页表项，页表大小 1M×4G×8B = 32×1024T

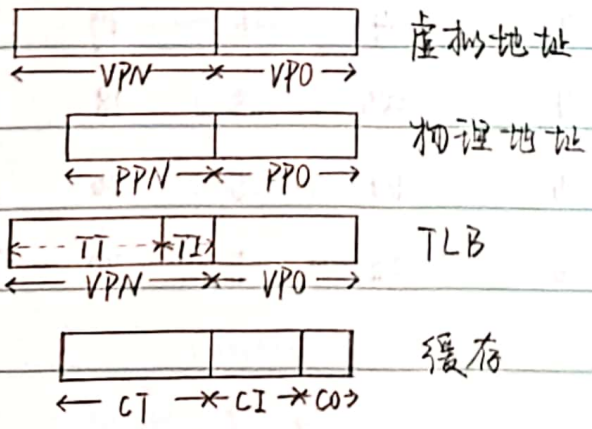
2). 页表大小 = (256T/4K)×8B = 512G

3). 因为页表需要覆盖全部虚拟地址空间，则不分级页表缺一不可；但分级页表只需全部一级页表，二级页表可在需要时创建，并且二级页表可以不在内存中而进行动态的调入，所以能够降低存储开销.

5/~~30~~ 16

6. 如果用高位做索引，则存储器中连续的若干块就会被映射到同一个组，如此缓存中的其他组将被闲置，降低效率，同时会增加冲突失效.

7.



TLB 和 Cache 并行访问原理：虚拟地址中的 VPN 用来查找 TLB，VPO 用来查找缓存，通过 TLB 将 VPN 映射到 PPN，PPN 作为缓存标记 CT，而将 VPO 分为索引 CI 和块位移 CO

故要求 VPO 和 CI + CO 的位数相同

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

2). $64K/1G = 2^{-14}$

$2^{-14} \times 1 + (1-2^{-14}) \times 110 = 110$

3). 时间局部性原理指出，程序中的某条指令一旦执行，不久后可能再次执行，某数据被访问后，可能再次被访问。空间局部性指出，一旦程序访问了某个存储单元，相邻单元也将被访问。由局部性原理，程序从内存中取出所需的数据块后，将放入缓存以备下次访问。在取数时，先将相邻的一块数据载入缓存。缓存的应用，加速了访存的平均延时，并且缓存的命中率越高，整体延时越低。

4). $x\% \times 1 + (1-x\%) \times 110 < 105$

$\Rightarrow$ 平均命中率需 $> \frac{5}{109} \approx 4.59\%$

9.

地址位数(Bit)	缓存大小 KB	块大小 Byte	相联度	组数量	组索引 Bit	标签 Bit	块偏移 Bit
32	4	64	2	32	5	21	6
32	4	64	8	8	3	23	6
32	4	64	全相联	1	0	26	6
32	16	64	1	256	8	18	6
32	16	128	2	64	6	19	7
32	64	64	4	256	8	18	6
32	64	64	16	64	6	20	6
32	64	128	16	32	5	20	7

10. 1). $0.22p_1 + 100(1-p_1) < 0.52p_2 + 100(1-p_2)$

$\Rightarrow p_1 > 0.997p_2$

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

$\Rightarrow (0.52p_2 - 0.22p_1)(k-1) < 0.3k$

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

11. 1). 块地址 直接映射 块替换 2路组相联 块替换

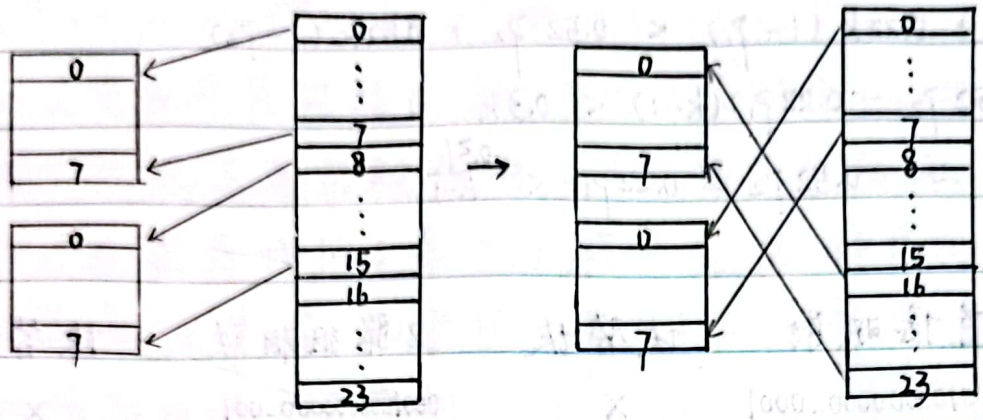
0x1001	000|00000000_0001	X	000|0000000000_001	X
0x1005	000|00000000_0101	X	000|0000000000_101	X
0x1021	000|0000001 0_0001	√	000|0000001 00_001	√
0x1045	000|00000100_0101	√	000|00000|000_101	√
0x1305	000|0011 0000_0101	√	000|0011 00000_101	√
0x2ee5	001|0111 0111 0_0101	√	001|0111 0111 00_101	√
0xffe5	1111 1111 1110 1111 0000_0101	√	1111 1111|0000 0_101	√

4路组相联		8路组相联	
000|0000000000_01	X	000|0000000000_1	X
000|0000000001_01	√	000|0000000010_1	√
000|0000001000_01	√	000|0000001 0000_1	√
~~000|0000001000_01~~	~~√~~	000|00000|0001 0_1	√
000|00000|0001_01	√	000|00|00000 10_1	√
000|0011 00000|_01	√	001|0111 0111 00 10_1	√
001|0111 0111 00|_01	√	1111 1111 1111 0000 0_1	√
1111 1111 0000 0|_01	√		

分别替换 5, 5, 6, 6 次

12、 A：2路组相联

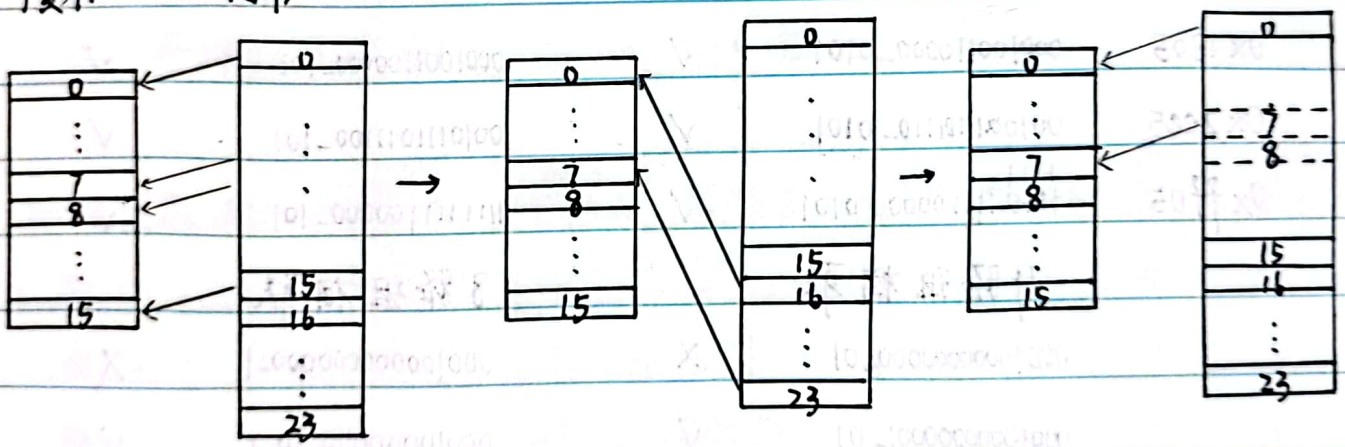
缓存(16块) 内存(全部24块，96个数)



缺失率 25%.

B：直接映射

缓存 内存



缺失率 (25%×1 + 16.7%×99)/100 = 16.75%.

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

14. 1). 优化前，每一次读都 miss，写入 hit，缓存缺失 64×128 = 8192 次
优化后，每一次调入连续的 8 个数，只在第一次读时 miss，后 8 次写和 7 次读都 hit，缓存缺失 1024 次。

2). 8192 ; 1024

3). ~~128×3×4 = 3×2^9~~ ~~8192×4 = 2^15~~ 1.5 KB，缓存容量 32 KB，优化前
优化后，每次调入的数据不会再次被访问，因此缓存容量只需 32 B。

15.

	input				output			
	列0	列1	列2	列3	列0	列1	列2	列3
行0	miss	miss	miss	miss	miss	miss	miss	miss
行1	hit	hit	hit	hit	miss	miss	miss	miss
行2	hit	hit	hit	hit	miss	miss	miss	miss
行3	hit	hit	hit	hit	miss	miss	miss	miss

16. 1). 75 %

2). 不可以，因为这里发生的都是强制失效

3). 可以，这里对数据的访问是连续的，更大的块可以一次调入更多连续数据，提高命中率。