

T5.

1) $\frac{2^{64}}{2^{12}} = 2^{52}$ 个页表条目

$2^{52} \times 8 = 2^{55}$ Byte.

2) $\frac{2^{48}}{2^{12}} = 2^{36}$, $2^{36} \times 8 = 2^{39}$ Byte

3) 多级页表将整个虚拟地址划分为多个层级, 每个层级都只需要相对较小的页表空间

T6.

x[i] 0 00 0000
x[u] 0 01 0000

采用中间位索引时, 相邻的数组会存在不同的组中; 采用高位索引时, 相邻的数组会存在同一个组, 按顺序访问时易发生抖动

T7.

① 位数相同可以简化地址转换, 减少了映射计算

② 位数相同可以共享一些逻辑电路和运算单元, 简化硬件设计

③ 保持地址映射的一致性

T8

$$1) 1 \times 97\% + 110 \times 3\% = 427 \text{ 周期}$$

$$2) \frac{64KB}{1GB} \times 1 + (1 - \frac{64KB}{1GB}) \times 110 = 110 \text{ 周期}$$

3) 局部性是指访问当前地址意味着将来很有可能访问相邻地址, 因此以块为单位存入 cache 中能有较高命中率, 减少运行周期。而若完全随机访问则使 cache 失去意义。

$$4) a \times 1 + (1-a) \times 110 > 105 \Rightarrow a > 0.0459$$

T9

编号	地址位数 Bit	缓存大小 KB	块大小 Byte	相邻度	组数 32 Bit	标签位数 Bit	偏移位数 Bit
1	32	4	64	32	5	21	6
2	32	4	64	8	3	33	6
3	32	4	64	全相邻	64	0	26
4	32	16	64	1	256	8	18
5	32	16	128	2	64	6	19
6	32	64	64	4	256	8	18
7	32	64	64	16	64	6	20
8	32	64	128	16	32	5	20

T10

$$1) \bar{t}_A = 0.22ns \times (1-P_1) + 100.22ns \times P_1$$

$$\bar{t}_B = 0.52ns \times (1-P_2) + 100.52ns \times P_2$$

$$\text{令 } \bar{t}_A < \bar{t}_B \Rightarrow P_1 - P_2 < 1.3 \times 10^{-3}$$

$$2) \bar{t}_A = 0.22(1-P_1) + 0.22kP_1$$

$$\bar{t}_B = 0.52(1-P_2) + 0.52kP_2$$

$$\bar{t}_A < \bar{t}_B \Rightarrow 11P_1 - 26P_2 < \frac{15}{k-1}$$

T11

$$64 = 2^6, 16 = 2^4$$

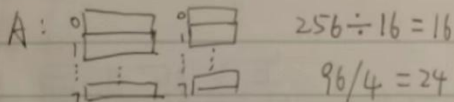
直接映射: $1, 5, 33 \Rightarrow 1, 69 \Rightarrow 5, 773 \Rightarrow 5, 5, 5$ 替换5次

2路组相联: 替换3次


4路: $1, 1, 1, 1, 1, 1, 1, 1$ 替换3次

8路: $1, 1, 1, 1, 1, 1, 1, 1$ 替换0次

T12




$$\text{缺失率} = \frac{1}{4} = 25\%$$


B:  $\frac{24 \times \frac{1}{4} + 99 \times (16 \times \frac{1}{4})}{100 \times 24} = 16.75\%$


T13

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

T14

1) 优化前: $32/4 = 8, 4KB/32B = 2^7$  $A[0][0], A[0][1] \dots A[3][63]$
 $A[1][0]$
 $A[2][0]$
缺失次数 $64 \times 128 = 8192$

优化后:  缺失次数: $64 \times 128 \times \frac{1}{8} = 1024$

2)  优化前: $0 \sim 2^8-1$ 2^7-1 路, 2^7-1 路. 缺失次数 $\frac{1}{8} \times 64 \times 128 = 1024$

优化后: $\frac{1}{8} \times 64 \times 128 = 1024$

3) 优化前: $8 \times 128 \times 32B = 32KB$

优化后: $8 \times 128 \times 32B = 32KB$

2 T15

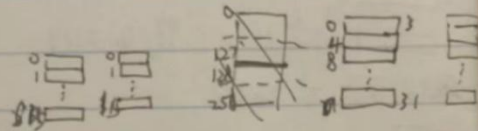
	input				output			
	列0	列1	列2	列3	列0	列1	列2	列3
行0	miss	h	h	h	miss	m	m	m
行1	m	h	h	h	m	m	m	m
行2	m	h	h	h	m	m	m	m
行3	m	h	h	h	m	m	m	m

$$32/16 = 2 \quad 16/4 = 4$$

1 T16

$$1) 512/16 = 2^5 \quad 16/4 = 4$$

命中率 75%



2) 不可以, 增加总大小即增加函数, 并不会增加命中率

3) 可以, 提高空间局部性。