

Homework : 存储

本此作业统一以 $K = 10^3$, $M=10^6$, $G=10^9$ 为计量单位。以后如果遇到类似的题目请提前查看/询问/确定这件事。

请直接用 Markdown 题目源文件填充答案，最后统一提交 PDF 格式，比如使用 Typora 导出。

T2

下面的表给出了一些不同的高速缓存的参数。你的任务是填写出表中缺失的字段。其中 m 是物理地址的位数，C 是高速缓存大小（数据字节数），B 是以字节为单位的块大小，E 是相联度，S 是高速缓存组数，t 是标记位数，s 是组索引位数，而 b 是块偏移位数。

m	C	B	E	S	t	s	b
32		4	4		24	6	
32	1024	32	2				
32	2048			256	21	8	3

T3

假设我们有一个具有如下属性的系统:

- 内存是字节寻址的。
- 内存访问是对 1 字节字的（比如访问一个 char）。
- 地址宽 12 位。
- 高速缓存是两路组相联的(E=2)，块大小为 4 字节(B=4)，有 4 个组(S=4)。

高速缓存的内容如下，所有的地址、标记和值都以十六进制表示:

组索引	标记	有效位	字节1	字节2	字节3	字节4
0	00	1	40	41	42	43
	83	1	FE	97	CC	D0
1	00	1	44	45	46	47
	83	0	54	55	56	57
2	00	1	48	49	4A	4B
	40	0	21	22	23	24
3	FF	1	9A	D0	03	EE
	00	0	A1	A2	A3	A4

对于下面每个内存访问，当他们顺序执行时，指出高速缓存是否命中，如果命中且操作前的数可从已有信息判断，请给出。

操作	地址	命中	值 (或未知)
读	0x834	miss	/
写	0x836	hit	/
读	0xFFFF	hit	EE

T4

仔细阅读下面的程序，根据条件回答下列各题：

- 地址宽度为 10
- 数组的起始地址为 0b0001000000 (即二进制表示)
- Block size = 4 Byte, Set = 4, 两路组相连 (B = 4, S = 4, E = 2)
- 替换算法为 LRU (最近最少使用)

```
#define LENGTH 8
void clear44(char array[LENGTH][LENGTH]) {
    int i, j;
    for (i = 0; i < 4; i++)
        for (j = 0; j < 4; j++)
            array[i][j] = 0;
}
```

4.1.1

以上程序会发生几次缓存miss？

一个cache block可以存储4个char，那么每次的array[i][0]都会miss，但是会把array[i]这行的四个char都载入cache，得到三个hit。\$\$ 一共miss四次 \$\$ **4.1.2**

如果 LENGTH = 16，那么会发生几次缓存miss？

array[i][0]会miss，之后的都会hit \$\$ 一共miss四次 \$\$ **4.1.3**

如果 LENGTH = 17，那么会发生几次缓存miss？

array[0][0] miss，然后array[0][1]~array[0][3] 会hit

array[1][0] miss, array[1][1]~array[1][2] 会hit, array[1][3] miss，载入 array[1][3]~array[1][6]。

array[2][0] miss, 从内存读入 array[1][15]~array[2][1]，所以 array[2][1] 会hit，

array[2][2] miss，载入 array[2][2]~array[2][5]。

array[3][0] miss, 从内存读入 array[2][14]~array[3][0]，array[3][1] miss \$\$ 一共miss七次 \$\$

4.1.4

请画出在 LENGTH = 17 时，程序执行结束时 set0 和 set1 的高速缓存状态，假设一开始全空。

可以用 `Array[0][0] ~ Array[0][3]` 的形式表示 Data 段落，有效位为 0 的行留空，每个 Set 内的顺序无所谓

SetID	Tag	Data
0	110	<code>array[1][15]~array[2][1]</code>
0	101	<code>array[0][16]~array[1][2]</code>
1	111	<code>array[3][1]~array[3][4]</code>
1	110	<code>array[2][2]~array[2][5]</code>

4.2

修改条件为

- 地址宽度为 10
- 数组的起始地址为 0b0010000000 (即二进制表示)
- Cache 的容量为 16 Byte, Block size = 4 Byte, 全相联
- 替换算法为 LRU

4.2.0

Tag 的位数是多少？

8位

4.2.1

原始程序会发生几次缓存miss？

\$\$ 一共 miss 四次 \$\$ **4.2.2**

如果 LENGTH = 16，那么会发生几次缓存miss？

\$\$ 一共 miss 四次 \$\$ **4.2.3**

如果 LENGTH = 17，那么会发生几次缓存miss？

\$\$ 一共 miss 七次 \$\$ **4.2.4**

请画出在 LENGTH = 17 时，程序执行结束时的高速缓存状态，假设一开始全空。

可以用 `Array[0][0] ~ Array[0][3]` 的形式表示 Data 段落，有效位为 0 的行留空

SetID	Tag	Data
0	00101001	<code>array[2][2]~array[2][5]</code>
1	00101100	<code>array[2][14]~array[3][0]</code>
2	00101101	<code>array[3][1]~array[3][4]</code>
3	00101000	<code>array[1][15]~array[2][1]</code>