B. 随机情况：估计如果块是随机映射到磁盘扇区上时读该文件所需要的时间(以 ms 为单位)。

* 6.25 下面的表给出了一些不同的高速缓存的参数。对于每个高速缓存，填写出表中缺失的字段。记住 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。

高速缓存	m	C	B	E	S	t	s	b
1.	32	1024	4	4	64	24	6	2
2.	32	1024	4	256	1	30	0	2
3.	32	1024	8	1	128	22	7	3
4.	32	1024	8	128	1	29	0	3
5.	32	1024	32	1	32	22	5	5
6.	32	1024	32	4	8	24	3	5

6.26 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

** 6.29 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(而不是 4 字节字)。
- 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0	字节1	字节2	字节3
0	00	1	40	41	42	43
	83	1	FE	97	CC	D0
1	00	1	44	45	46	47
	83	0	—	—	—	—
2	00	1	48	49	4A	4B
	40	0	—	—	—	—
3	FF	1	9A	C0	03	FF
	00	0	—	—	—	—

A. 下面的图给出了一个地址的格式(每个小框表示一位)。指出用来确定下列信息的字段(在图中标号出来):

CO　高速缓存块偏移

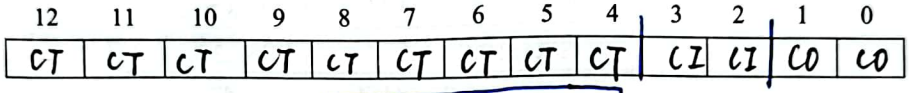
CI　高速缓存组索引

CT　高速缓存标记

E=2, B=4, S=4

b=2　s=2

CO　CI



B. 对于下面每个内存访问，当它们是按照列出来的顺序执行时，指出是高速缓存命中还是不命中。如果可以从高速缓存中的信息推断出来，请也给出读出的值。

操作	地址	命中?	读出的值(或者未知)
读	0x834	否	未知
写	0x836	命中	未知
读	0xFFD	命中	C0

(第一次缓存了)

6.30 假设我们有一个具有如下属性的系统：

FF　31

- 内存是字节寻址的。
- 内存访问是对 1 字节字的(而不是 4 字节字)。
- 地址宽 13 位。

E=4　b=2　S=3

- 高速缓存是四路组相联的($E=4$)，块大小为 4 字节($B=4$)，有 8 个组($S=8$)。

考虑下面的高速缓存状态。所有的地址、标记和值都以十六进制表示。每组有 4 行，索引列包含组索引。标记列包含每一行的标记值。V 列包含每一行的有效位。字节 0～3 列包含每一行的数据，标号从左向右，字节 0 在左边。

4 路组相联高速缓存

索引	标记	V	字节 0～3	标记	V	字节 0～3	标记	V	字节 0～3	标记	V	字节 0～3
0	F0	1	ED 32 0A A2	8A	1	BF 80 1D FC	14	1	EF 09 86 2A	BC	0	25 44 6F 1A
1	BC	0	03 3E CD 38	A0	0	16 7B ED 5A	BC	1	8E 4C DF 18	E4	1	FB B7 12 02
2	BC	1	54 9E 1E FA	B6	1	DC 81 B2 14	00	0	B6 1F 7B 44	74	0	10 F5 B8 2E
3	BE	0	2F 7E 3D A8	C0	1	27 95 A4 74	C4	0	07 11 6B D8	BC	0	C7 B7 AF C2
4	7E	1	32 21 1C 2C	8A	1	22 C2 DC 34	BC	1	BA DD 37 D8	DC	0	E7 A2 39 BA
5	98	0	A9 76 2B EE	54	0	BC 91 D5 92	98	1	80 BA 9B F6	BC	1	48 16 81 0A
6	38	0	5D 4D F7 DA	BC	1	69 C2 8C 74	8A	1	A8 CE 7F DA	38	1	FA 93 EB 48
7	8A	1	04 2A 32 6A	9E	0	B1 86 56 0E	CC	1	96 30 47 F2	BC	1	F8 1D 42 30

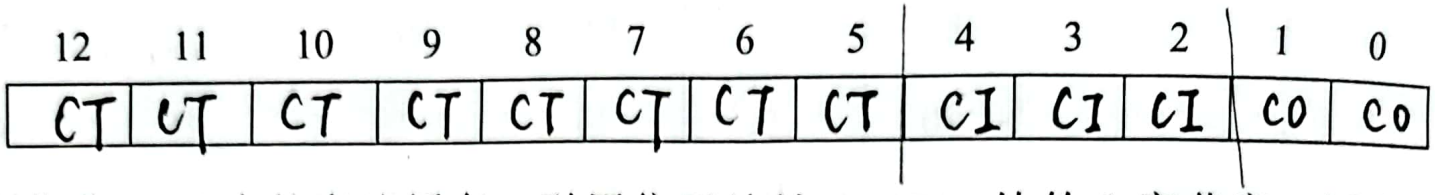
A. 这个高速缓存的大小(C)是多少字节？ C = BXSxE = 128 byte

B. 下面的图给出了一个地址的格式(每个小框表示一位)。指出用来确定下列信息的字段(在图中标号出来):

CO　高速缓存块偏移

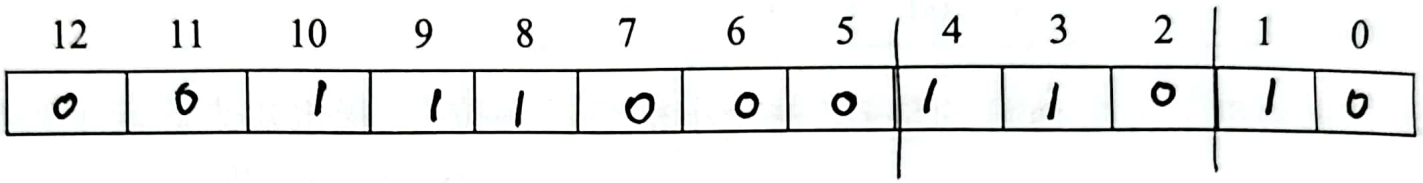
CI　　高速缓存组索引

CT　　高速缓存标记



** 6.31 假设程序使用作业 6.30 中的高速缓存，引用位于地址 0x071A 处的 1 字节字。用十六进制表示出它所访问的高速缓存条目，以及返回的高速缓存字节值。指明是否发生了高速缓存不命中。如果有高速缓存不命中，对于“返回的高速缓存字节”输入“—”。提示：注意那些有效位！

A. 地址格式(每个小框表示一位)：



B. 内存引用：

参数	值
高速缓存块偏移（CO）	0x 2
高速缓存组索引（CI）	0x 6
高速缓存标记（CT）	0x 38
高速缓存命中？（是/否）	否
返回的高速缓存字节	0x —

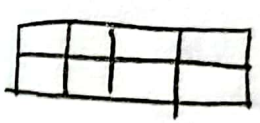
** 6.34 考虑下面的矩阵转置函数：

```
1   typedef int array[4][4];
2
3   void transpose2(array dst, array src)
4   {
5       int i, j;
6
7       for (i = 0; i < 4; i++) {
8           for (j = 0; j < 4; j++) {
9               dst[j][i] = src[i][j];
10          }
11      }
12  }
```

假设这段代码运行在一台具有如下属性的机器上：

- sizeof(int)==4。

4×4×4 = 64　/数组



- 数组 src 从地址 0 开始，而数组 dst 从地址 64 开始(十进制)。
- 只有一个 L1 数据高速缓存，它是直接映射、直写、写分配的，块大小为 16 字节。
- 这个高速缓存总共有 32 个数据字节，初始为空。
- 对 src 和 dst 数组的访问分别是读和写不命中的唯一来源。

行0 s0 d0 s0 d2 s0 d0 d2 s2 d0 s2 d2 s2 d
行1 d1 d3 s1 d1 s1 d3 d1 d3 s3 d1

对于每个 row 和 col，指明对 src[row][col]和 dst[row][col]的访问是命中(h)还是不命中(m)。

例如，读 src[0][0]会不命中，而写 dst[0][0]也会不命中。

读 →　写 ↓

dst数组

	列0	列1	列2	列3
行0	m	m	m	m
行1	m	m	m	m
行2	m	m	m	m
行3	m	m	m	m

src数组

	列0	列1	列2	列3
行0	m	m	h	m
行1	m	h	m	h
行2	m	m	h	m
行3	m	h	m	h

对于情况3，更大的块大小会帮助降低不命中率吗？为什么能或者为什么不能？

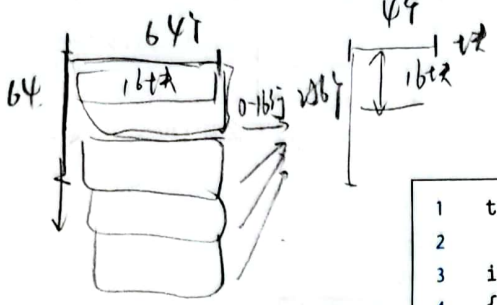
**6.37 这道题也是测试你分析C语言代码的高速缓存行为的能力。假设我们在下列条件下执行图6-47中的3个求和函数：

- sizeof(int)==4。
- 机器有4KB直接映射的高速缓存，块大小为16字节。
- 在两个循环中，代码只对数组数据进行内存访问。循环索引和值sum都存放在寄存器中。
- 数组a从内存地址0x08000000处开始存储。

对于$N=64$和$N=60$两种情况，在表中填写它们大概的高速缓存不命中率。

N=64

A: 64×64×4 = $2^6 \times 2^6 \times 2^2 = 2^{14}$



函数	N=64	N=60
sumA	25%	25%
sumB	100%	25%
sumC	50%	25%

B. 一直miss 没有冲突

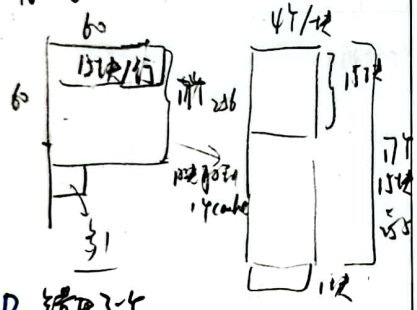
× √ √ √
× √ √ √

C.

ij	i+1 j	i j+1	i+1 j+1
×	×	√	√

先按列读，但读完列又会读读行2次，⇒ × × √ √

N=60



B. 错开3个 缓存不是全相连 ⇒ 只有conflict miss

```
1   typedef int array_t[N][N];
2
3   int sumA(array_t a)
4   {
5       int i, j;
6       int sum = 0;
7       for (i = 0; i < N; i++)
8           for (j = 0; j < N; j++) {
9               sum += a[i][j];
10          }
11      return sum;
12  }
13
14  int sumB(array_t a)
15  {
16      int i, j;
17      int sum = 0;
18      for (j = 0; j < N; j++)
19          for (i = 0; i < N; i++) {
20              sum += a[i][j];
21          }
22      return sum;
23  }
24
25  int sumC(array_t a)
26  {
27      int i, j;
28      int sum = 0;
29      for (j = 0; j < N; j+=2)
30          for (i = 0; i < N; i+=2) {
31              sum += (a[i][j] + a[i+1][j]
32                      + a[i][j+1] + a[i+1][j+1]);
33          }
34      return sum;
35  }
```

图 6-47 作业 6.37 中引用的函数

• 6.38 3M 决定在白纸上印黄方格，做成 Post-It 小贴纸。在打印过程中，他们需要设置方格中每个点的 CMYK(蓝色，红色，黄色，黑色)值。3M 雇佣你判定下面算法在一个具有 2048 字节、直接映射、块大小为 32 字节的数据高速缓存上的效率。有如下定义：

2^{11} $2^6 = 64$块

```
1   struct point_color {
2       int c;
3       int m;
4       int y;
5       int k;
6   };
7
8   struct point_color square[16][16];
9   int i, j;
```

有如下假设：

- sizeof(int)==4。
- square起始于内存地址0。
- 高速缓存初始为空。
- 唯一的内存访问是对于square数组中的元素。变量i和j存放在寄存器中。

确定下列代码的高速缓存性能：

```
1       for (i = 0; i < 16; i++){
2           for (j = 0; j < 16; j++) {
3               square[i][j].c = 0;
4               square[i][j].m = 0;
5               square[i][j].y = 1;
6               square[i][j].k = 0;
7           }
8       }
```

32B

c m y k	c m y k
16B	16B
X ✓ ✓ ✓	✓ ✓ ✓ ✓

每个组相连有c不命中

A. 写总数是多少？ 16×16×4 = 1024

B. 在高速缓存中不命中的写总数是多少？ 128

C. 不命中率是多少？ 12.5%