数组。每个组包含 E 个高速缓存行(cache line)。每个行是由一个 $B=2^b$ 字节的数据块(block)组成的，一个有效位(valid bit)指明这个行是否包含有意义的信息，还有 $t=m-(b+s)$ 个标记位(tag bit)(是当前块的内存地址的位的一个子集)，它们唯一地标识存储在这个高速缓存行中的块。

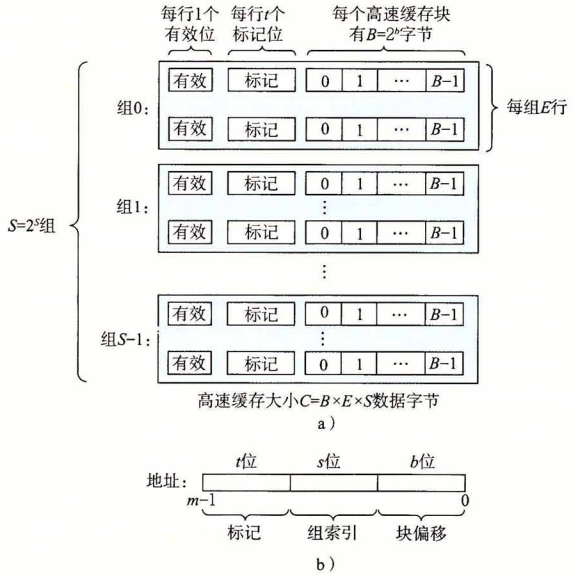

图 6-25　高速缓存(S，E，B，m)的通用组织。a)高速缓存是一个高速缓存组的数组。每个组包含一个或多个行，每个行包含一个有效位，一些标记位，以及一个数据块；b)高速缓存的结构将 m 个地址位划分成了 t 个标记位、s 个组索引位和 b 个块偏移位

一般而言，高速缓存的结构可以用元组(S，E，B，m)来描述。高速缓存的大小(或容量)C 指的是所有块的大小的和。标记位和有效位不包括在内。因此，$C=S\times E\times B$。

当一条加载指令指示 CPU 从主存地址 A 中读一个字时，它将地址 A 发送到高速缓存。如果高速缓存正保存着地址 A 处那个字的副本，它就立即将那个字发回给 CPU。那么高速缓存如何知道它是否包含地址 A 处那个字的副本的呢？高速缓存的结构使得它能通过简单地检查地址位，找到所请求的字，类似于使用极其简单的哈希函数的哈希表。下面介绍它是如何工作的：

参数 S 和 B 将 m 个地址位分为了三个字段，如图 6-25b 所示。A 中 s 个*组索引位*是一个到 S 个组的数组的索引。第一个组是组 0，第二个组是组 1，依此类推。组索引位被解释为一个无符号整数，它告诉我们这个字必须存储在哪个组中。一旦我们知道了这个字必须放在哪个组中，A 中的 t 个标记位就告诉我们这个组中的哪一行包含这个字(如果有的话)。当且仅当设置了有效位并且该行的标记位与地址 A 中的标记位相匹配时，组中的这一行才包含这个字。一旦我们在由组索引标识的组中定位了由标号所标识的行，那么 b 个块偏移位给出了在 B 个字节的数据块中的字偏移。

你可能已经注意到了，对高速缓存的描述使用了很多符号。图 6-26 对这些符号做了个小结，供你参考。