

CS152 Computer Architecture and  
Engineering  
**Cache Implementations**

*Last Updated:*  
2/15/2008 4:31 PM

<http://inst.eecs.berkeley.edu/~cs152/sp08>

## Direct-mapped Cache

The following diagram shows how a direct-mapped cache is organized. To read a word from the cache, the input address is set by the processor. Then the index portion of the address is decoded to access the proper row in the tag memory array and in the data memory array. The selected tag is compared to the tag portion of the input address to determine if the access is a hit or not. At the same time, the corresponding cache block is read and the proper line is selected through a MUX.

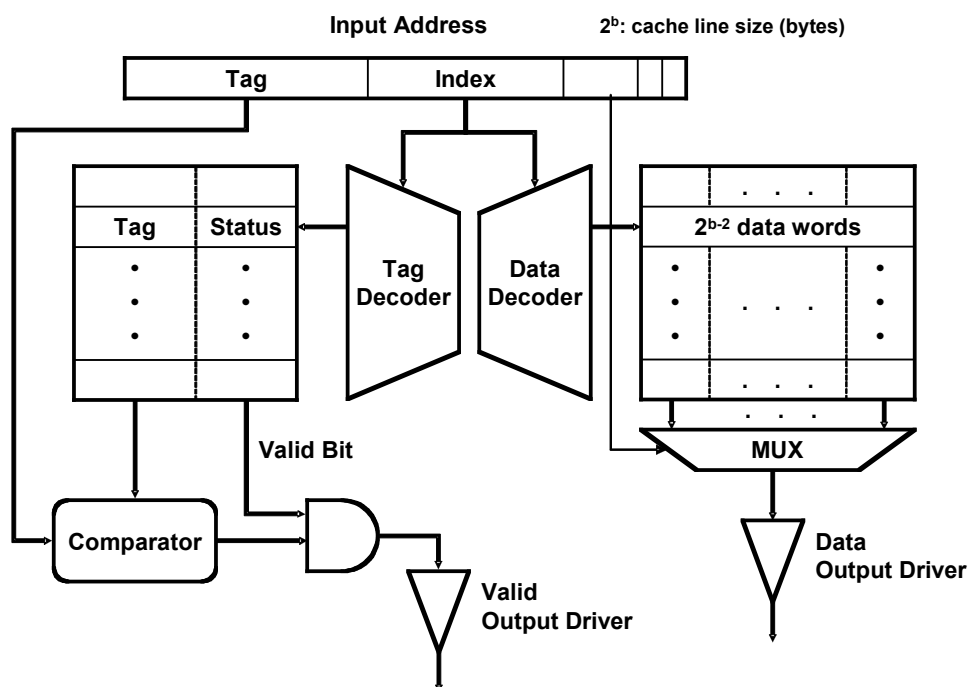


Figure H2-A: A direct-mapped cache implementation

In the tag and data array, each row corresponds to a line in the cache. For example, a row in the tag memory array contains one tag and two status bits (valid and dirty) for the cache line. For direct-mapped caches, a row in the data array holds one cache line.

## Four-way Set-associative Cache

The implementation of a 4-way set-associative cache is shown in the following diagram. (An  $n$ -way set-associative cache can be implemented in a similar manner.) The index part of the input address is used to find the proper row in the data memory array and the tag memory array. In this case, however, each row (set) corresponds to four cache lines (four ways). A row in the data memory holds four cache lines (for 32-bytes cache lines, 128 bytes), and a row in the tag memory array contains four tags and status bits for those tags (2 bits per cache line). The tag memory and the data memory are accessed in parallel, but the output data driver is enabled only if there is a cache hit.

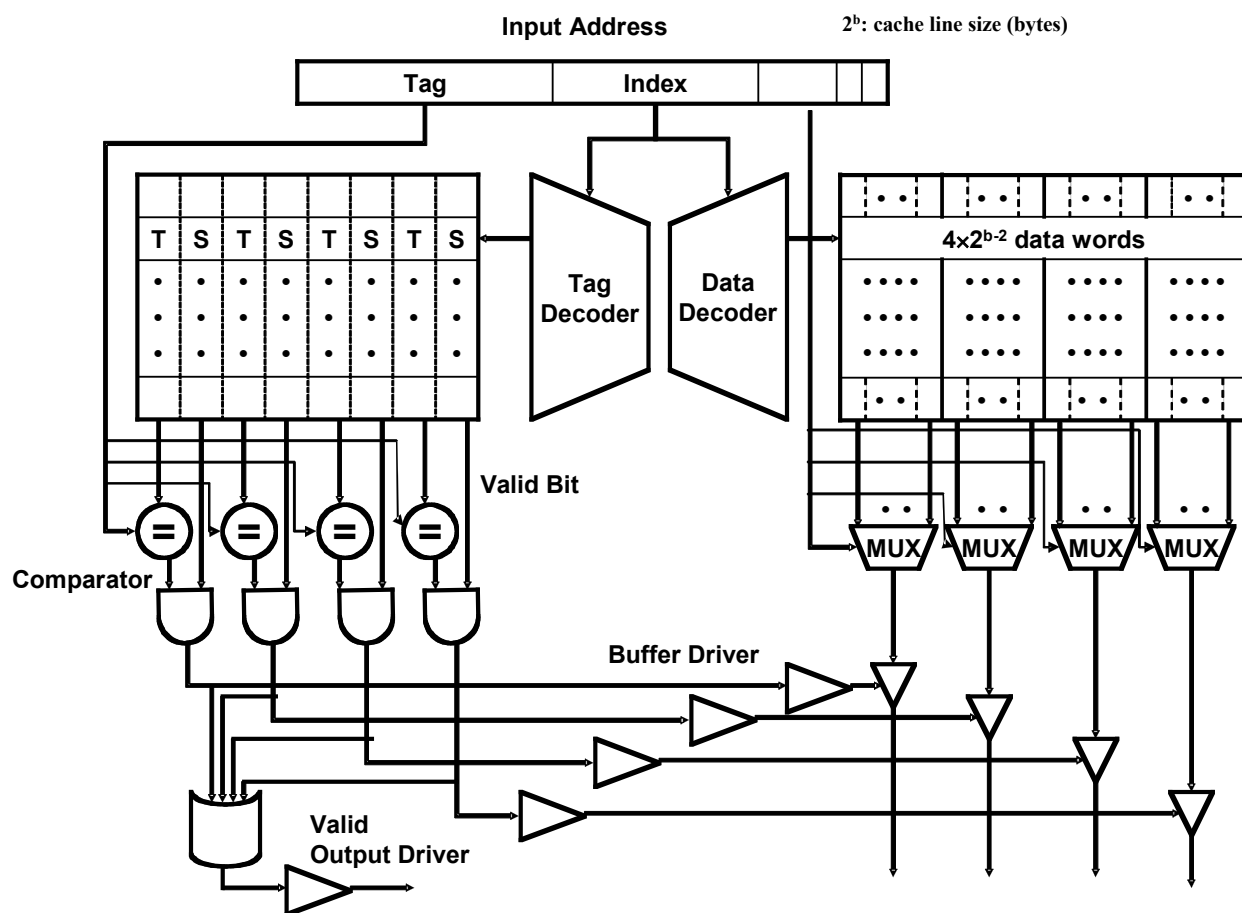


Figure H2-B: A 4-way set-associative cache implementation