LRU Algorithm

Counter implementation of LRU requires 2 bits per line

Each LRU bit represents the absolute rank for the line allocated with the LRU bit.

00b’ represents the most recently Used way and 11b’ represents the least recently used way

Initialize all counter bits according to the position (i.e way 0 gets 00b’ and get 3 gets 11b’)

LRUAlgorithm

If cache hit occurs

Increment the LRU bits of the lines that have smaller counter value than the line that was just accessed.

Reset the LRU bits for the line that was just accessed to 00b’

else if cache miss occurs

If compulsory miss (means there is at least one line with valid bit = 0)

Read cache line from memory and fill the empty line.

Else if conflict miss occurs (means all line has valid bit = 1)

Read cache line from memory and replace the line that have LRU bits = 11b’.

Increment the LRU bits of the lines that have smaller counter value than the line that was just filled.

Reset the LRU bits for the line that was just filled to 00b’

Endif

Endif

Endalgorithm