Matthew Kozak

EGRE 426, Lab 5

11/21/2019

Implementation

The program for this lab assignment was chosen to be written in C, as VHDL was recommended. The very first thing implemented was command line argument configuration in order to specify the parameters necessary to chance the cache for testing, discussed later. The program specified how many parameters it needs at runtime. Next, file intake was managed. The program will check if the address file specified by a ‘.txt.’ is actually present in the directory of the executable. If it is not, it will inform the user. To finalize the initial error checking, the code checks the associativity and the block size to confirm that both are equal to a power of 2. If either of these are not a power of 2, the program will let the user know what needs to be resolved.

The initializations of the code begins with determining the bits necessary for an offset. This is done by taking the natural logarithm of the block size input by the user, then dividing that by the natural logarithm of 2. The result of this rounded up provides an integer representation of the number of bits required to represent the offset required. The same formula is used to determine the bits for the index. However, when calculating this value, the program will also remove a number of bits based on the number of bits needed for the associativity. For example:

* For ‘addresses.txt’, there are 10000 elements, which requires 12 bits to be represented
* If an associativity of 2 is selected, this reduces the result by 1
* If an associativity of 4 is selected, this reduces the result by 2, etc.

This result (‘b\_min’ in the code) is then used to produce a mask called ‘shift\_temp’. This mask is used against the address read in to determine the index.

After these values are calculated, the arrays to hold the tag, valid bit, count (for LRU), and the index (e.g. data for Step 1 cases) are generated, and cleared to be all zeros.