Report On Cache Simulator Lab 6

Name:Aditya Garg
Roll Number:CS22BTECH11002

November 22, 2023

1 Coding Approach

- The code has been broken down into 3 functions "dmc" for direct mapped cache , "fac" for fully associative cache and "sac" for set associative cache.
- Based on the cache.config file , the number of blocks , sets and the bits used for tag and index are calculated .
- The instructions from cache.access are stored in a vector which is later then parsed into tag and index . Offset bits are ignored as they are not stored in cache and the parsed input is stored in a vector which is later passed to the function . Based on the associativity mentioned , the appropriate function is called .
- For any given instruction , based on the index/set value present in the "meta data" vector, the tags bit are compared . If the tags bit are matched it results in a hit else it is a miss .
- For a read/write miss , based on the replacement policy and write allocate/ no-allocate the data in cache is replaced .
- For FIFO policy an array stores the insertion time at a given set/ index in the cache while LRU policy is implemented using an array which stores the time at which a way in a set was last accessed .
- The minimum value is found for FIFO replacement policy while maximum value is found for LRU policy in case of a miss .
- Finally the address , tag , set and HIT/MISS for each instruction is printed

2 Testing Approach

- I tried to generate cases having a pattern in addresses to check the replacement policy and whether tag bits were getting stored in cache or not .

- Some test cases have been added in TestCases folder along with the their output for verification.
- I tried running my code on different cache size , replacement policy and associative size to observe the output and find any issues in the code .
- The output for each testcase is written to file "output.txt" .