Documentation

Name – Taral Jain
Roll Number – 2019392

- **The code uses in high-level language Java, designed to implement the Cache Memory that allows loading and searching using Direct Mapping, Fully Associative Mapping, K-way Set Associative Mapping.**

➢ Assumptions

1) All the inputs have to be given **Manually** by the user.

2) In different mappings, the inputs are:

- **Direct Mapping**: N (Total number of words in memory)
  W (Number of words in a Block/Block Size)
  CL (Number of Cache Lines)
- **Fully Associative**: N (Total number of words in memory)
  W (Number of words in a Block/ Block Size)
  CL (Number of Cache Lines)
  Operation (READ, WRITE, DISPLAY CACHE)
  Address (To READ or WRITE)
- **K-way set Associative:** N (Total number of words in memory)
  W (Number of words in a Block/ Block Size)
  CL (Number of Cache Lines)
  K (Number of cache lines in a Set)
  Operation (READ, WRITE, DISPLAY CACHE)
  Address (To READ or WRITE)

3) The number of bits in Address is assumed to be exactly equal to **“$\log_2(N)$”**

4) Main Memory is present ONLY in **Direct Mapping**

5)- In case of miss- "Address not found"

- In case of replacement - Address by which the block will be replaced
- In Direct Mapping – Direct block replacement from Main Memory (**NO POLICY**)

6) For Block replacement, **LRU** (Least Recently Used) policy is used.

➢ Explanation of Code

Direct Mapping – The cache contains multiple sets with a single cache line per set. Based on the address of the memory block, it can only occupy a single cache line.

- The line is determined by the index bits derived from the address of the memory block
- The memory block is placed in the line identified and the tag is stored in the tag field associated with the line
- If the cache line is previously occupied, then the new data replaces the memory block in the cache
- To **search** a particular block, we compare tags. If the tag matches, then there is a cache hit and the word is displayed on screen. Otherwise, cache miss occurs and memory block is fetched from Main Memory.

Fully Associative Mapping – The cache contains a single cache set with multiple cache lines. A memory block can occupy any of the cache lines.

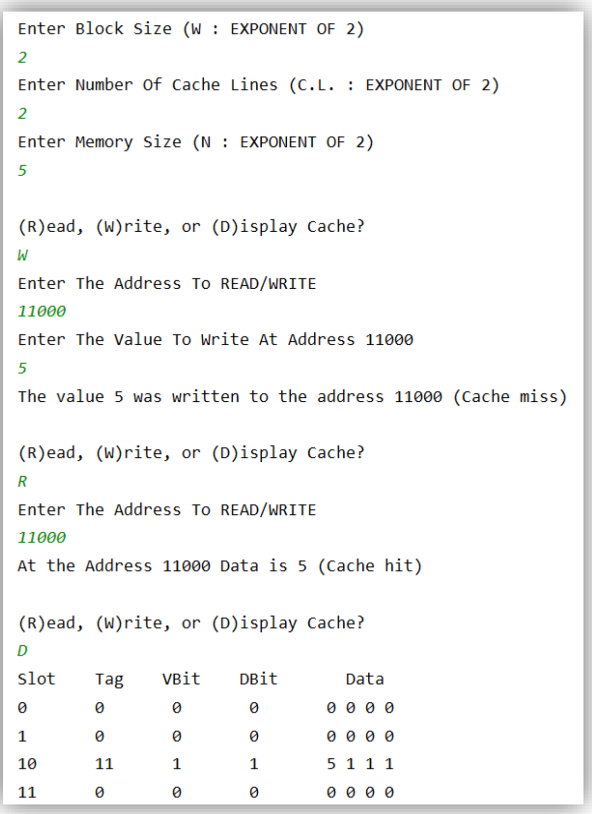
- The cache line is selected by Tag of the line. If the Tag is 0, the new memory block can be placed in the cache line, else it has to be placed in another cache line with Tag 0.
- If the cache is full then a block is **removed** using **LRU policy** and new memory block is then placed in that cache line.
- To Search, we compare Tag of the memory address with Tag associated with all the cache lines using loop traversal. If match occurs, the block is present in the cache and is a **Cache Hit**, we print the Word on screen. Else, **Cache Miss** and block has to be loaded from Main memory.

K-way Set Associative Mapping – It lies in mediocrity of Direct Mapping and Fully Associative Mapping.

- The cache is fragmented as ‘CL/K’ sets and each set contain ‘K’ Cache Lines.
- A memory block is first mapped onto a set and then placed into **any** cache line of the set if and only if the particular set has **empty Cache Line(s)**.
- The set number is determined by some particular bits of address.
- The block is placed in empty cache line in the determined set, and Tag is associated with the Cache Line. If all Cache Lines in the set are occupied, then We use **LRU policy** to remove some block and then store new block into that removed Cache Line.
- To Search, we compare Tag with Tag of all Cache Lines in determined set. If the Tag matches, **Cache Hit** occurs and we print the particular Word in that Cache Line. Else, **Cache Miss** occurs and the block is loaded from Main Memory.

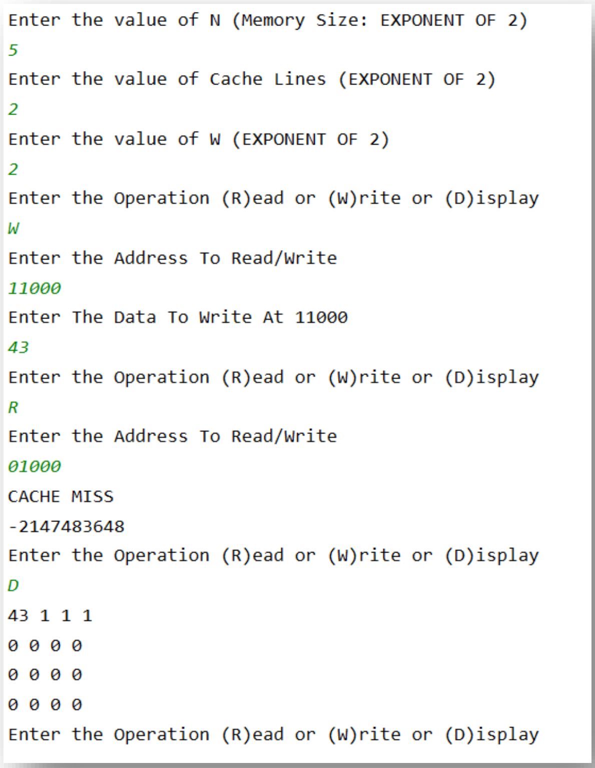
➢ Sample I/O of Code

Direct Mapping



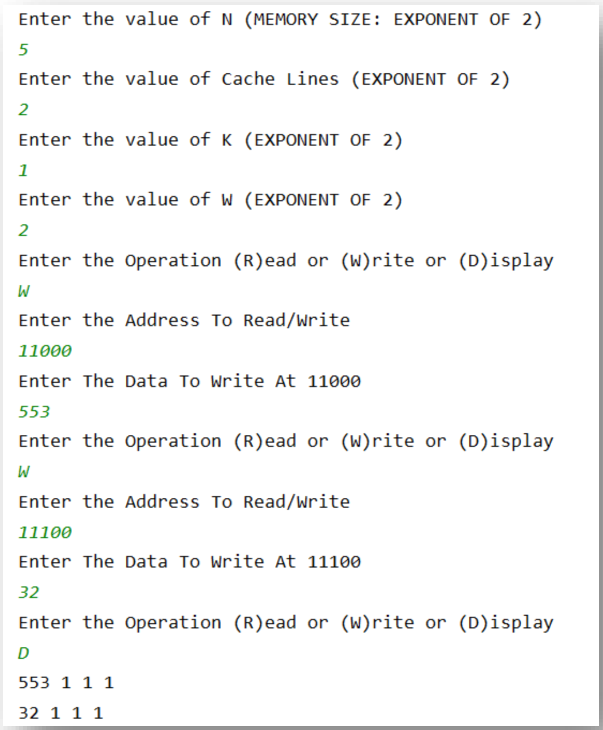
```
Enter Block Size (W : EXPONENT OF 2)
2
Enter Number Of Cache Lines (C.L. : EXPONENT OF 2)
2
Enter Memory Size (N : EXPONENT OF 2)
5

(R)ead, (W)rite, or (D)isplay Cache?
W
Enter The Address To READ/WRITE
11000
Enter The Value To Write At Address 11000
5
The value 5 was written to the address 11000 (Cache miss)

(R)ead, (W)rite, or (D)isplay Cache?
R
Enter The Address To READ/WRITE
11000
At the Address 11000 Data is 5 (Cache hit)

(R)ead, (W)rite, or (D)isplay Cache?
D
Slot    Tag    VBit    DBit       Data
0       0      0       0       0 0 0 0
1       0      0       0       0 0 0 0
10      11     1       1       5 1 1 1
11      0      0       0       0 0 0 0
```

Fully Associative Mapping



```
Enter the value of N (Memory Size: EXPONENT OF 2)
5
Enter the value of Cache Lines (EXPONENT OF 2)
2
Enter the value of W (EXPONENT OF 2)
2
Enter the Operation (R)ead or (W)rite or (D)isplay
W
Enter the Address To Read/Write
11000
Enter The Data To Write At 11000
43
Enter the Operation (R)ead or (W)rite or (D)isplay
R
Enter the Address To Read/Write
01000
CACHE MISS
-2147483648
Enter the Operation (R)ead or (W)rite or (D)isplay
D
43 1 1 1
0 0 0 0
0 0 0 0
0 0 0 0
Enter the Operation (R)ead or (W)rite or (D)isplay
```

K-way Set Associative Mapping



```
Enter the value of N (MEMORY SIZE: EXPONENT OF 2)
5
Enter the value of Cache Lines (EXPONENT OF 2)
2
Enter the value of K (EXPONENT OF 2)
1
Enter the value of W (EXPONENT OF 2)
2
Enter the Operation (R)ead or (W)rite or (D)isplay
W
Enter the Address To Read/Write
11000
Enter The Data To Write At 11000
553
Enter the Operation (R)ead or (W)rite or (D)isplay
W
Enter the Address To Read/Write
11100
Enter The Data To Write At 11100
32
Enter the Operation (R)ead or (W)rite or (D)isplay
D
553 1 1 1
32 1 1 1
```