Piyush Sharma
2019316

CACHE PROJECT

PROBLEM STATEMENT

Write a program that allows loading into cache and searching cache using:

1. Direct Mapping
2. Fully Associative Mapping
3. K- Way Set Associative Mapping

SOLUTION

In this project, I have designed an algorithimic implementation of the cache memory using all three techniques.
Programming Language: Java

ASSUMPTIONS

The following are the assumptions:

1. The size of Main Memory, Cache Memory and Block Size should be taken in Bytes
   Eg: 128 Bytes, 1024 Bytes
2. The size of Main Memory, Cache Memory and Block Size is entered in power of 2
   Eg; 64 Bytes, 256 Bytes
3. The number of lines in Cache is entered in power of 2
   Eg; 4, 8, 16, 32
4. The size of each word is 2 Bytes
   1 word = 2 bytes = 16 bits
5. The Random Cache Replacement Algorithm is applied for replacement during Fully and K- Way Set Assocoative Mapping
   WikiPedia : Random Replacement :
   https://en.wikipedia.org/wiki/Cache_replacement_policies#Random_replacement_(RR)

INPUT DATA

The following are the inputs taken from the user in the project:

1. Integer : N : Size of main memory in bytes
2. Integer : B : Size of each block in cache memory
3. Integer : CL: Number of lines in cache memory
4. Integer : K : (Only for K- Way Set Associative Technique)
5. Integer : T : Total number of operations to be performed
6. String : PA : Physical Address to be read/written

Piyush Sharma
2019316

OUTPUT

The following are part of the output of the project:

1. Information about Cache and Main Memory - size and blocks, etc

```
************************** DISPLAY INFORMATION STARTS **************************

MAIN MEMORY
Size is 32 Bytes
No. of blocks are 16
No. of total words is 16
Size of each block is 2 Bytes
No. of words in each block are 1
Size of each word is 2 Bytes

CACHE MEMORY
Size is 16 Bytes
No. of blocks are 8 Bytes
No. of total words is 8
Size of each block is 2 Bytes
No. of words in each block are 1
Size of each word is 2 Bytes

**************************** DISPLAY INFORMATION ENDS ****************************
```

2. Displaying Cache after every operation

```
***************************** CACHE MEMORY START *****************************

1111000011001100
1001101101110001
1101111111011000
1111001001010001
1111000011111100
1001000011111100
0111101000000101
1001001100000010

***************************** CACHE MEMORY ENDS *****************************
```

3. Categorizing the given PA into Tag , Block Offset, Block No, etc

```
The given Physical Address provides these bits :
Tag = 7   Block Offset = 2
In the Cache Memory

TAG : 1010101
BLOCK OFFSET : 11
```

4. Reading/Writing information from/onto the Cache Memory

```
THE GIVEN PHYSICAL ADDRESS CORRESPONDS TO THE WORD NUMBER 343 OF THE MAIN MEMORY
THIS BLOCK NUMBER 85 OF MAIN MEMORY IS TO BE ADDED
THIS BLOCK WILL BE ADDED IN THE LINE NUMBER 0 OF CACHE MEMORY

***** CACHE MISS *****
 This block is not available in the Cache Memory

Do you want to write some information in this word ? (Y/N)
y

Input valid 2 Byte = 16 bit information for this particular word
101011111111111

***** DATA SUCCESSFULLY WRITTEN *****

The value stored in Cache Memory for the given Physcial Address is 101011111111111
```

5. Displaying Cache Hit, Miss and Ratio

```
TOTAL NO. OF CACHE HIT = 2
TOTAL NO. OF CACHE MISS = 3
CACHE HIT RATIO  = 0.4

************************* THANK YOU FOR USING TO THE DIRECT CACHE IMPLEMENTATION ****
```

GETTING STARTED

Please follow these instructions to use the Cache Memory Implementation Project:

1. Download and save the .java file on your system
2. Execute the program using an IDE
3. Follow the on-screen instructions.
4. The final result is displayed on the IDE Console

ERRORS HANDLED

The following are the errors that the Direct Mapped Cache Project:

1. `ERROR: PLEASE ENTER VALID INPUT FOR MAPPING TECHNIQUE`
   If invalid mapping technique is selected, then this error is displayed and the program is terminated.

2. `ERROR: MAIN MEMORY SIZE MUST BE LARGER THAN CACHE SIZE`
   If while inputting the initial values, the size of cache memory is more than main memory, then it displays this error and exits the program.

3. `ERROR: NUMBER OF CACHE LINES CAN NOT BE A ODD NUMBER`
4. `ERROR: VALUE OF 'K' CAN NOT BE A ODD NUMBER`
   If while inputting the initial values, the number of cache lines or ‘K’ value is an odd number, eg, 5, 17, 33, then it displays this error and exits the program.

5. `ERROR: SIZE MAIN MEMORY CAN NOT BE A ODD NUMBER`
6. `ERROR: SIZE OF INDIVIDUAL BLOCK CAN NOT BE A ODD NUMBER`
   If while inputting the initial values, the size of main memory or the block size is an odd number eg, 65, 1023, then it displays this error and exits the program

7. `ERROR: INPUT " + n + " MUST BE A POWER OF 2`
   Check if all the given values are in power of 2 or not.

8. `ERROR: INVALID RESPONSE : OVERWRITING STEP ABORTED`
   If while overwriting data, an invalid reponse is entered. Then it aborts the overwrtiting process and moves on to the next function.

9. `ERROR: BITS OF PHYSICAL ADDRESS NOT VALID`
   If while entering the physical address, the no. of bits are not as required. Then this error message is loaded and the program terminates.

SUPPORT

If you have any issues regarding this project, feel free to contact me.
Name: Piyush Sharma
E-Mail: piyush19316@iiitd.ac.in

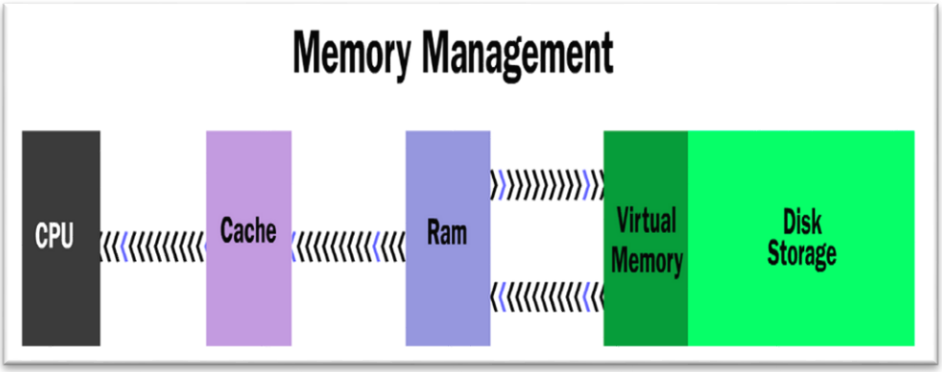
Piyush Sharma
2019316

UNDERSTANDING - CACHE PROJECT

The main objective behind this whole project is to design an implementation of a cache memory and highlight the technqiues of mapping. So, to thoroughly understand the project, it is important to learn/revise some theory related to the project.

CACHE MEMORY

Cache is a memory that is embedded onto the CPU chip. It acts an hardware storage for the data between RAM and CPU Chip. Cache is extremely faster than RAM, ROM and its speed is comparable to the speed of the CPU. And hence, It is embedded onto the CPU Chip. The average storage capacity of Cache is around 8 MB.



CACHE MAPPING

Cache has limited storage as compared to the RAM/ Main Memory and hence, only a limited number of blocks of main memory can be placed in the lines of the Cache Memory. Cache Mapping is this process of bringing the contents of the main memory in a systematic order and also make sure that the CPU can refer this content.

There are three types of cache mapping technique:

1. Direct Mapping
2. Fully Associative Mapping
3. K- Way Set Associative Mapping

DIRECT MAPPING

- A particular block of main memory can map to only one particular line of the cache
- This relation is given by : (Block No. in main memory) modulo (Total lines in cache)
- This technique is easy to implement and fast also.
- It uses a lot of useless calls
- It does not use the empty lines in Cache Memory to accommodate incoming blocks and hence, gives rise to conflict misses.

Division of Physical Address in Direct Mapping

FULLY ASSOCIATIVE MAPPING

- A particular block of main memory can map to any line of the cache
- Hence, no conflict miss and no empty lines left remaining.
- This technique is harder to implement than the direct mapping and also applies one of the Replacement Algorithm of Cache Mapping
- It has a lot of bits for the Tag
- Since, it compares the PA with each tag. Hence, increases the number of comparsions.

Division of Physical Address in Fully Associative Mapping

K WAY SET ASSOCIATIVE MAPPING

- It is a mixture of Direct and Fully Associative Mapping
- The cache lines is initally divided into sets based on a relation similar to Direct Mapping while inside each line, it follows Fully Associative Mapping.
- No. Of Sets = (Total No. Of lines in Cache) / K
- The relation is given by : (Block No. in main memory) modulo (No. Of Sets)
- This technique is harder to implement than the direct mapping and also applies one of the Replacement Algorithm of Cache Mapping
- It decreases the number of comparsions
- It uses the empty spaces more efficently than direct mapping

Division of Physical Address in K-way Set Associative Mapping

Piyush Sharma
2019316

RESOURCES

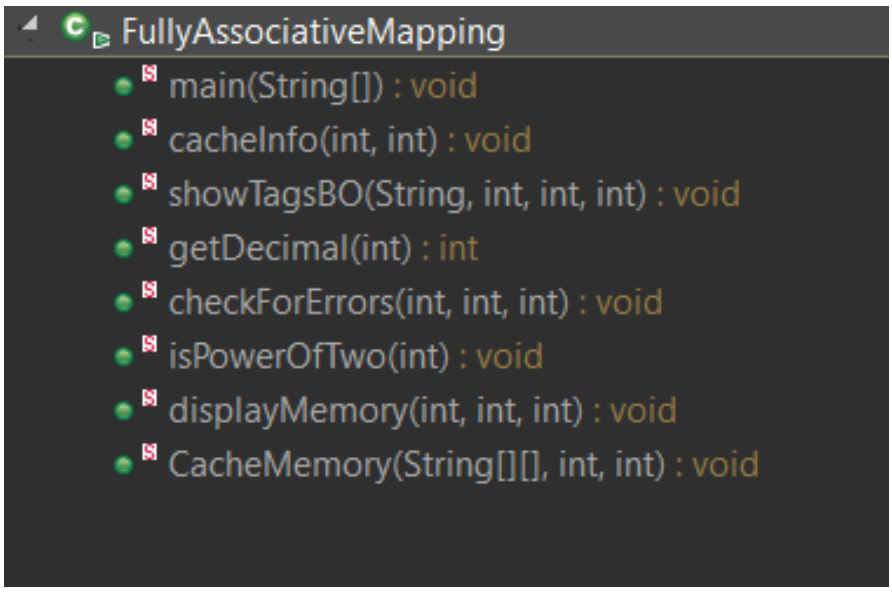
1. YouTube – Computer Organisation and Archieture by Gate Smashers
   https://www.youtube.com/playlist?list=PLxCzCOWd7aiHMonh3G6QNKq53C6oNXGrX
2. GeeksForGeeks – Cache Memory and Mapping
   https://www.geeksforgeeks.org/cache-memory-in-computer-organization/
3. WikiPedia – CPU Cache
   https://en.wikipedia.org/wiki/CPU_cache#Direct-mapped_cache
4. GateVidyalaya - Computer Organisation and Archieture
   https://www.gatevidyalay.com/computer-organization-architecture/

Piyush Sharma
2019316

UNDERSTANDING – THE CODE

LIST OF FUNCTIONS

Herby, follows a list of all the functions used in the Java program for the assembler:



FUNCTIONS AND THEIR EXPLAINATION

Hereby, follows the list of functions used in the program with a quick explaination about them.
The list is sorted in the order of the appearance of the functions in the code.

```
public static void main(String[] args) throws IOException
```
The main function selects the type of mapping to be implemented and calls the necessary programs to it

```
public static void direct() throws IOException
public static void direct() throws IOException
public static void direct() throws IOException
```
This is the function for Mapping technique. It does a lot of things:
1. Inputs all the necessary values
2. Displays all important information
3. Checks whether the given PA is a cache hit or miss
4. Writes and over writes data
5. Performs the Random Replacement Algorithm
6. Displays the final output

```
public static void cacheInfo( int cacheHit, int cacheMiss )
```
Displays the total no. of hits, misses and hit ratio

```
public static void showTagsBO(String paArray, int
bitsOfBlocksInEachLine, int bitsOfLines, int bitsOfWords)
```
Converts, categorises and displays the tags, block no, block offset for thr given physical address.

```
public static int getDecimal(int binary)
```
Converts the given binary number into its equivalent decimal number

```
public static void checkForErrors(int cl, int b, int n)
```
Checks whether the main memory size is larger than the cache size and checks if the inputted values are even or not. Cause if they are odd, then it is an error.

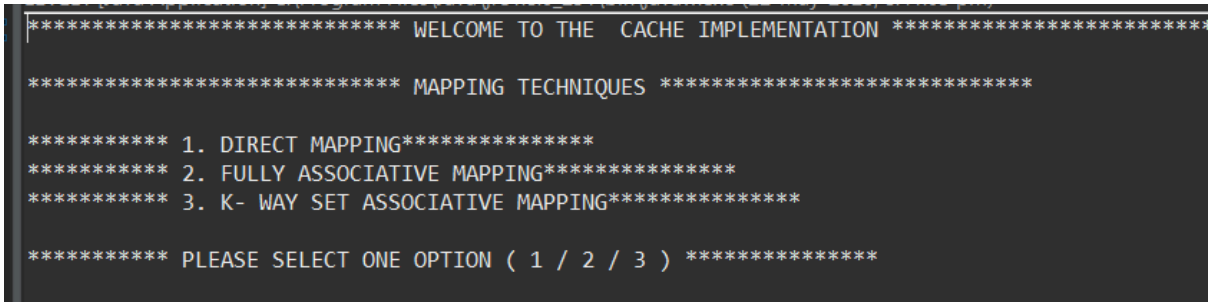
```
public static void isPowerOfTwo(int n)
```
Checks whether the given number is a power of 2 or not. Because if it is not a power of 2, then it is an error.

```
public static void displayMemory(int n, int cl, int b)
```
Displays valid information about the Main Memory and Cache Memory like No. Of Blocks, Size, No. Of Total Words, etc.

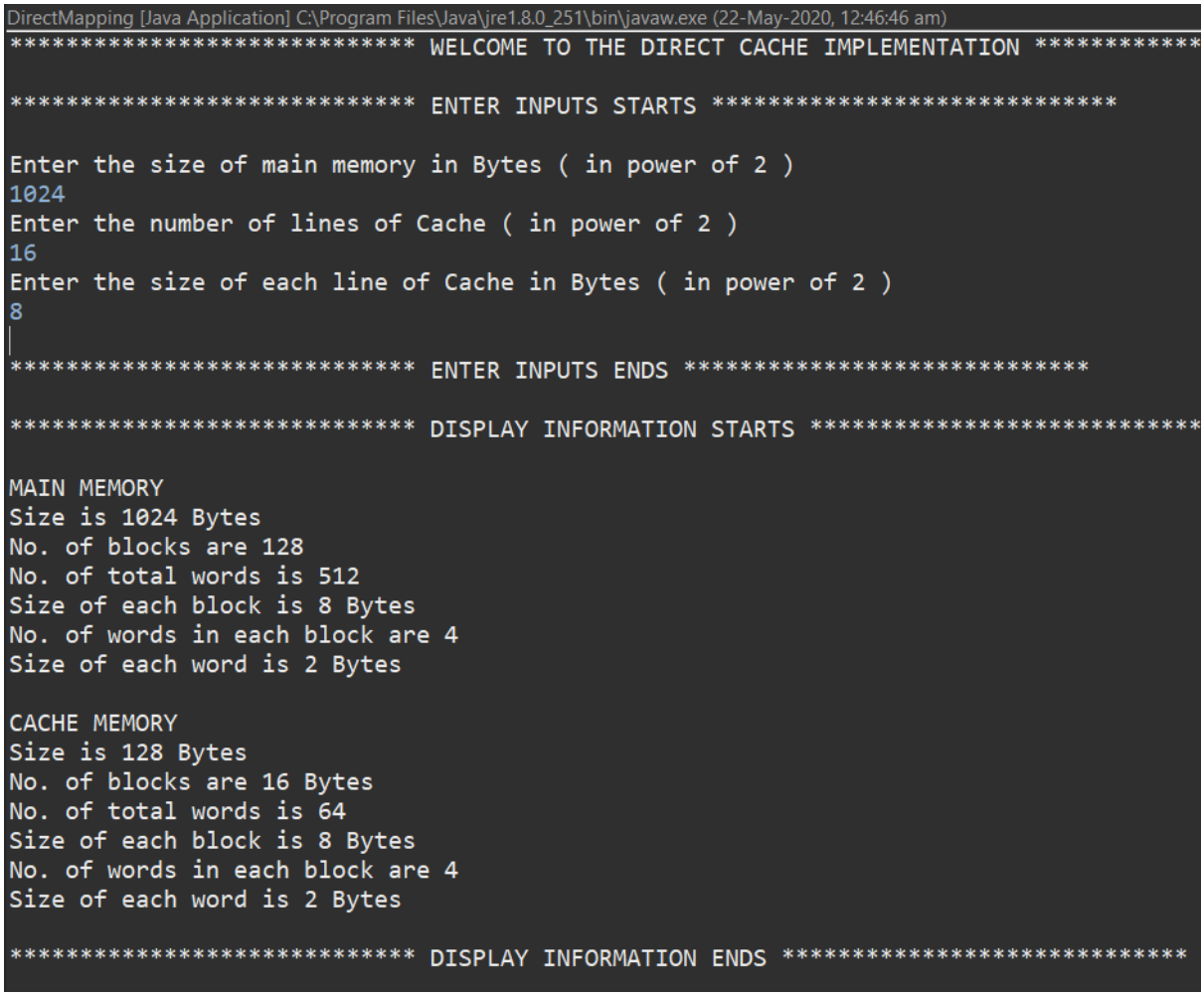
```
public static void CacheMemory(String[][] cache, int cl, int b)
```
Outputs the current cache memory as a 2 Dimensional real cache

RUNNING THE PROJECT

This is the welcome screen of the project and loads up when the program is executed.



The user enters the asked values and a table of information is presented that contains important info about the main memory and cache memory.



This is the initial look of the cache memory when it has no data/blocks in it.
The user enters the number of query he/she wants to perform.

```
**************************** CACHE MEMORY START ****************************

null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null

**************************** CACHE MEMORY ENDS *****************************

Enter the no. of operations you want to perform
```

This user performs only 1 query. The valid physical address is also entered and it is matched to its corresponding word and block in main memory and cache memory.
The tag, block offset are also calculated and displayed along with the notification whether it was a Cache Hit or a Cache Miss.

```
Enter the no. of operations you want to perform
1
**************************** OPERATION STARTS ****************************

Please Enter the Physical Address of 10 bits
1000011011

The given Physical Address provides these bits :
Tag  = 3   Line Number  = 5   Block Offset = 2
In the Cache Memory

TAG : 100
LINE NUMBER : 00110
BLOCK OFFSET : 11

THE GIVEN PHYSICAL ADDRESS CORRESPONDS TO THE WORD NUMBER 269 OF THE MAIN MEMORY
THIS BLOCK NUMBER 67 OF MAIN MEMORY IS TO BE ADDED
THIS BLOCK WILL BE ADDED IN THE LINE NUMBER 3 OF CACHE MEMORY

***** CACHE MISS ***** because this block is not available in the Cache Memory

Do you want to write some information in this word ? (Yes/No)
Yes
```

The user writes/ overwrites the data and then, the new modified cache is displayed along with a successful / unsuccessful writing process message.

```
Input valid 2 Byte = 16 bit information for this particular word
1000111100001111

***** DATA SUCCESSFULLY WRITTEN *****

The value stored in Cache Memory for the given Physcial Address is 1000111100001111


**************************** CACHE MEMORY START ****************************

null null null null
null null null null
null null null null
null 1000111100001111 null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null
null null null null

**************************** CACHE MEMORY ENDS ****************************
```

This is the final ending scene of the project and displays the count of hit and misses and also calculates the hit ratio and terminates the program successfully.

```
**************************** CACHE MEMORY ENDS ****************************


TOTAL NO. OF CACHE HIT = 0
TOTAL NO. OF CACHE MISS = 1
CACHE HIT RATIO  = 0.0

************************* THANK YOU FOR USING TO THE DIRECT CACHE IMPLEMENTATION ********
```

```
************************** ALL OPERATIONS PERFORMED SUCCESSFULLY **************************

************************** THANK YOU FOR USING THE CACHE IMPLEMENTATION ***************************
```