**Assignment I**

**Problem Bank 20**

* **Assignment Description:**

Cache memory is a type of high-speed volatile computer memory that provides high-speed data access to a processor and stores frequently used computer programs, applications, and data. It acts as a temporary storage area between the main memory (RAM) and the central processing unit (CPU). The purpose of cache memory is to store copies of frequently accessed data and instructions, reducing the time it takes for the CPU to retrieve information. This proximity to the CPU allows cache memory to deliver faster data access speeds compared to main memory, contributing to overall system performance optimization.

The assignment aims to provide deeper understanding of cache by analysing its behaviour using cache implementation of CPU- OS Simulator. The assignment has three parts.

* Part I deals with Cache Memory Management with Direct Mapping
* Part II deals with Cache Memory Management with Associative Mapping
* Part III deals with Cache Memory Management with Set Associative Mapping
* **Submission:** You will have to submit this documentation file and the name of the file should be GROUP-NUMBER.pdf. For Example, if your group number is 1, then the file name should be GROUP-1.pdf.

Submit the assignment by **January 06, 2024 through canvas only**. File submitted by any means outside CANVAS will not be accepted and marked. In case of any issues, please drop an email to the course lead TA Vaibhav Jain (email : vaibhav.jain@wilp.bits-pilani.ac.in)

**Caution!!!**

All submissions for graded components must be the result of your original effort. It is strictly prohibited to copy and paste verbatim from any sources, whether online or from your peers. The use of unauthorized sources or materials, as well as collusion or unauthorized collaboration to gain an unfair advantage, is also strictly prohibited. Please note that we will not distinguish between the person sharing their resources and the one receiving them for plagiarism, and the consequences will apply to both parties equally.

In cases where suspicious circumstances arise, such as identical verbatim answers or a significant overlap of unreasonable similarities in a set of submissions, will be investigated, and severe punishments will be imposed on all those found guilty of plagiarism.

**Evaluation:**

* The assignment carries 10 marks
* Grading will depend on
  + Contribution of each student in the implementation of the assignment
  + **Plagiarism or copying will result in -10 marks**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FILL IN THE DETAILS GIVEN BELOW\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Assignment Set Number:**

**Group Name:**

**Contribution Table:**

**Contribution** (This table should contain the list of all the students in the group. Clearly mention each student’s contribution towards the assignment. Mention “No Contribution” in cases applicable.)

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.** | **Name (as appears in Canvas)** | **ID NO** | **Contribution** |
| **1.** | **Maneesha Rai** | **2023da04384** | **100%** |
| **2.** | **Mithilesh Kumar Rai** | **2023da04194** | **100%** |
| **3.** | **Ayush Rai** | **2023da04026** | **100%** |
|  |  |  |  |

**Resource for Part I, II and III:**

* Use following link to login to “eLearn” portal.
  + <https://elearn.bits-pilani.ac.in>
* Click on “My Virtual Lab – CSIS”
* Using your canvas credentials login into Virtual lab
* In “BITS Pilani” Virtual lab click on “Resources”. Click on “Computer Organization and software systems” course.
  + Use resources within “LabCapsule3: Cache Memory”

**Code to be used:**

The following code written in STL Language:

program Twenty

VAR a array(5) INTEGER

VAR b array(5) INTEGER

VAR len byte

VAR i byte

VAR j byte

VAR p byte

VAR q byte

VAR x byte

VAR n byte

a(1)=80

a(2)=20

a(3)=19

a(4)=89

a(5)=30

len = 5

writeln("Numbers to sort :")

for n = 1 to len

write(a(n), " ")

next

writeln("")

writeln("Selection Sort")

for i = 1 to len

for j = i+1 to len

p = a(i)

q = a(j)

if p > q then

x = a(i)

a(i) = a(j)

a(j) = x

end if

next

next

for n = 1 to len

write(a(n), " ")

next

end

**General procedure to convert the given STL program into ALP:**

* Open CPU OS Simulator. Go to **advanced tab** and press **compiler** button
* Copy the above program in **Program Source** window
* Open **Compile** tab and press **compile** button
* In **Assembly Code,** enter **start address** and press **Load in Memory** button
* Now the assembly language program is available in CPU simulator.
* Set speed of execution to **FAST.**
* Open I/O console
* To run the program press **RUN** button.

**General Procedure to use Cache set up in CPU-OS simulator**

* After compiling and loading the assembly language code in CPU simulator, press “Cache-Pipeline” tab and select cache type as “Data”. Press “SHOW CACHE” button.
* In the newly opened cache window, choose appropriate cache Type, cache size, set blocks, replacement algorithm and write policy.

**At first, write your understanding / observation about the given program then attempt the part as mentioned below.**

**Part I: Direct Mapped Cache**

1. Execute the above program by setting block size to 2, 4, 8, 16 and 32 for cache size = 8, 16 and 32. Record the observation in the following table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Block Size | Cache size | # Hits | # Misses | % Miss Ratio | %Hit Ratio |
| 2 | 8 | 309 | 352 | 53.25% | 46.75% |
| 4 | 288 | 370 | 56.2% | 43.80% |
| 8 | 316 | 342 | 51.9% | 48.10% |
| 2 | 16 | 431 | 227 | 34.5% | 54.10% |
| 4 |  | 378 | 280 | 42.5% | 65.50% |
| 8 |  | 356 | 302 | 45.9% | 57.50% |
| 16 |  | 398 | 260 | 39.5% | 60.50% |
| 2 | 32 | 529 | 129 | 19.6% | 80.40% |
| 4 |  | 489 | 169 | 25.6% | 74.40% |
| 8 |  | 446 | 212 | 32.2% | 67.80% |
| 16 |  | 455 | 203 | 30.8% | 69.20% |
| 32 |  | 485 | 173 | 26.2% | 73.80% |

1. Plot a single graph of Cache hit ratio Vs Block size with respect to cache size = 8, 16 and 32. Comment on the graph that is obtained.

A green triangle with numbers and lines

Description automatically generated

**Part II: Associative Mapped Cache**

a) For the given program, fill up the following table for three different replacement algorithms and state which replacement algorithm is better and why?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Replacement Algorithm: Random | | | | |
| Block Size | Cache size | Miss | Hit | Hit ratio |
| 2 | 4 | 368 | 290 | 0.44 |
| 2 | 8 | 334 | 324 | 0.49 |
| 2 | 16 | 320 | 338 | 0.51 |
| 2 | 32 | 219 | 439 | 0.67 |
| 2 | 64 | 68 | 590 | 0.90 |
| Replacement Algorithm: FIFO | | | | |
| Block Size | Cache size | Miss | Hit | Hit ratio |
| 2 | 4 | 412 | 246 | 0.37 |
| 2 | 8 | 378 | 280 | 0.43 |
| 2 | 16 | 239 | 419 | 0.64 |
| 2 | 32 | 83 | 575 | 0.87 |
| 2 | 64 | 65 | 593 | 0.90 |
| Replacement Algorithm: LRU | | | | |
| Block Size | Cache size | Miss | Hit | Hit ratio |
| 2 | 4 | 409 | 249 | 0.38 |
| 2 | 8 | 370 | 288 | 0.44 |
| 2 | 16 | 193 | 465 | 0.71 |
| 2 | 32 | 88 | 570 | 0.87 |
| 2 | 64 | 69 | 589 | 0.90 |

b) Plot the graph of Cache Hit Ratio Vs Cache size with respect to different replacement algorithms. Comment on the graph that is obtained.

**Part III: Set Associative Mapped Cache**

Execute the above program by setting the following Parameters:

* Number of sets (Set Blocks): 2 way
* Cache Type: Set Associative
* Replacement: LRU/FIFO/Random

a) Fill up the following table for three different replacement algorithms and state which replacement algorithm is better and why?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Replacement Algorithm: Random | | | | |
| Block Size | Cache size | Miss | Hit | Hit ratio |
| 2 | 4 | 402 | 256 | 0.39 |
| 2 | 8 | 342 | 316 | 0.48 |
| 2 | 16 | 234 | 424 | 0.64 |
| 2 | 32 | 130 | 528 | 0.80 |
| 2 | 64 | 69 | 589 | 0.90 |
| Replacement Algorithm: FIFO | | | | |
| Block Size | Cache size | Miss | Hit | Hit ratio |
| 2 | 4 | 412 | 246 | 0.37 |
| 2 | 8 | 366 | 292 | 0.44 |
| 2 | 16 | 241 | 417 | 0.63 |
| 2 | 32 | 124 | 534 | 0.81 |
| 2 | 64 | 68 | 590 | 0.90 |
| Replacement Algorithm: LRU | | | | |
| Block Size | Cache size | Miss | Hit | Hit ratio |
| 2 | 4 | 409 | 249 | 0.38 |
| 2 | 8 | 350 | 308 | 0.47 |
| 2 | 16 | 226 | 432 | 0.66 |
| 2 | 32 | 115 | 543 | 0.83 |
| 2 | 64 | 66 | 592 | 0.90 |

1. Plot the graph of Cache Hit Ratio Vs Cache size with respect to different replacement algorithms. Comment on the graph that is obtained.

A graph of a number of people

Description automatically generated with medium confidence

c) Fill in the following table and analyse the behaviour of Set Associate Cache. Which one is better and why?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Replacement Algorithm: LRU | | | | |
| Block Size, Cache size | Set Blocks | Miss | Hit | Hit ratio |
| 2, 64 | 2 – Way | 66 (10.0%) | 592 | 0.90 |
| 2, 64 | 4 – Way | 66 (10.0%) | 592 | 0.90 |
| 2, 64 | 8 – Way | 70 (10.6%) | 588 | 0.89 |