



**THE AMERICAN
UNIVERSITY IN CAIRO**
الجامعة الأمريكية بالقاهرة

**CSCE2303 - Computer Organization & Assembly
Language Programming**

Project II
Memory Hierarchy Simulator

Fall 2022

Submitted To: Dr. Nourhan Zayed

Submitted By:

Laila El Saeed - 900191891
Mohamed Hashish - 900201220
Sherif Wessa - 900203171

Implementation:

- Project is a memory cache simulator that traces the behaviour of the memory cache given the the cache information including: memory cache size (S), line size (L), and number of cycles to access memory (clk)
- Input is divided into 2 text files:
 - First one is: “input.txt” which includes: memory cache size (S), line size (L), and number of cycles to access memory (clk)
 - Second one is: “Memory Addresses.txt” for user to input sequence of 20 memory addresses in bytes
- The program traces and stores no. of accesses and no. of hits/misses, it displays the cache index alongside the valid bit and tag for each index. The hit/miss ratio is calculated as well as Average Memory Access Time (AMAT) value using the value of 100 clock cycles provided in the project description.

Design Decisions and Assumption:

- We decided to create a cache as a struct which contains the tag with a bool called valid bit.
- User enters cache data as well as memory addresses in the right format as seen in the .txt files

Bugs or Issues in simulator:

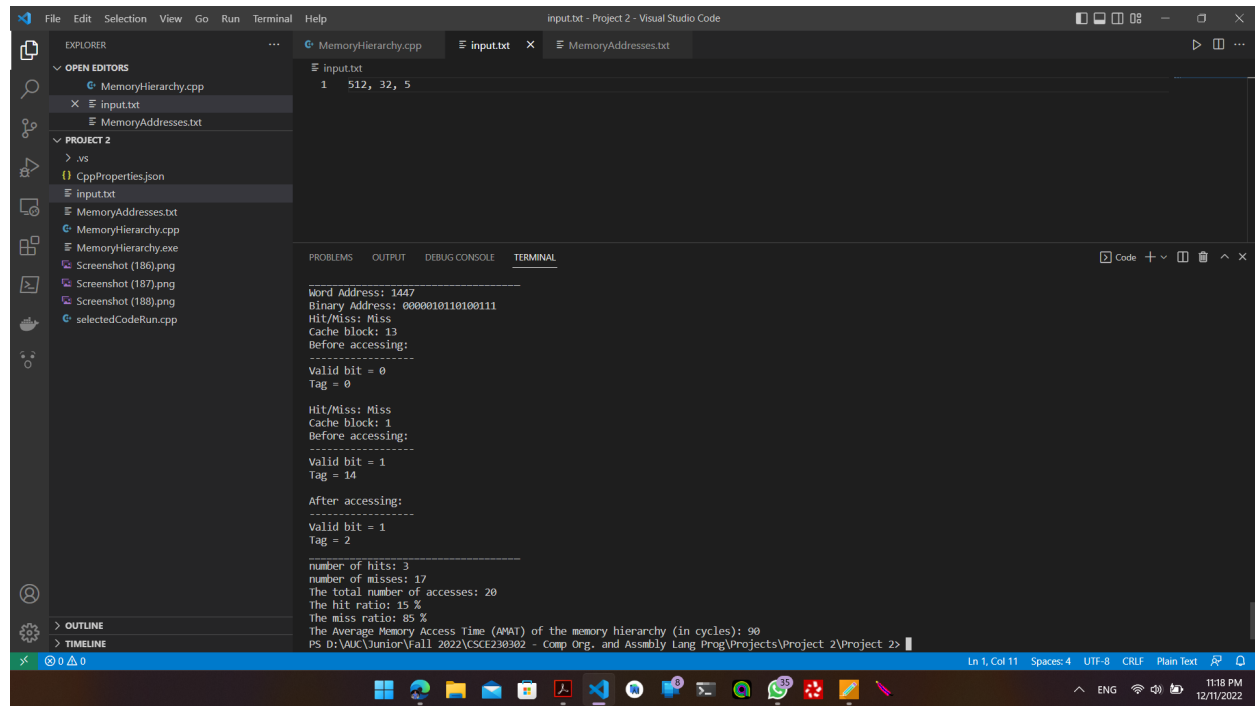
- While using the 2 test cases, no bugs/issues were found

Bonus:

- We created a GUI app which implements the same code using Qt Framework, with some minor modifications to adapt the new style/framework

User guide with screenshots:

Example 1: $S=512$, $L=32$, $clk=5$



The screenshot shows the Visual Studio Code interface with a project named "input.txt - Project 2". The Explorer panel on the left shows the project structure, including files like MemoryHierarchy.cpp, input.txt, and MemoryAddresses.txt. The main editor displays the input.txt file with the content "1 512, 32, 5". The Output panel at the bottom shows the execution results of the program.

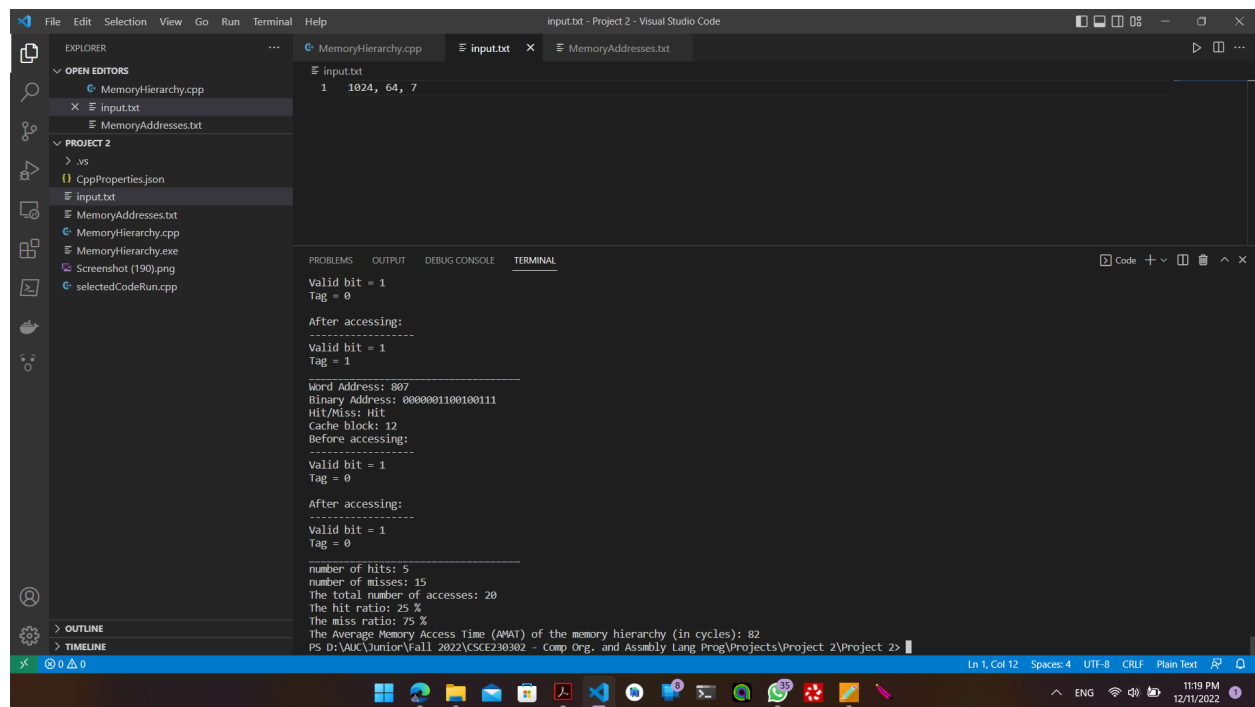
```
Word Address: 1447
Binary Address: 0000010110100111
Hit/Miss: Miss
Cache block: 13
Before accessing:
Valid bit = 0
Tag = 0

Hit/Miss: Miss
Cache block: 1
Before accessing:
Valid bit = 1
Tag = 14

After accessing:
Valid bit = 1
Tag = 2

number of hits: 3
number of misses: 17
The total number of accesses: 20
The hit ratio: 15 %
The miss ratio: 85 %
The Average Memory Access Time (AMAT) of the memory hierarchy (in cycles): 90
PS D:\VAUC\Junior\Fall1 2022\CSCE230302 - Comp Org. and Assembly Lang Prog\Projects\Project 2\Project 2>
```

Example 2: S=1024, L=64, clk=7



```
File Edit Selection View Go Run Terminal Help
input.txt - Project 2 - Visual Studio Code

EXPLORER
OPEN EDITORS
  MemoryHierarchy.cpp
  input.txt
  MemoryAddresses.txt
PROJECT 2
  .vs
  CppProperties.json
  input.txt
  MemoryAddresses.txt
  MemoryHierarchy.cpp
  MemoryHierarchy.exe
  Screenshot (190).png
  selectedCodeRun.cpp

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Valid bit = 1
Tag = 0

After accessing:
Valid bit = 1
Tag = 1

Word Address: 807
Binary Address: 0000001100100111
Hit/Miss: Hit
Cache block: 12
Before accessing:
Valid bit = 1
Tag = 0

After accessing:
Valid bit = 1
Tag = 0

number of hits: 5
number of misses: 15
The total number of accesses: 20
The hit ratio: 25 %
The miss ratio: 75 %
The Average Memory Access Time (AMAT) of the memory hierarchy (in cycles): 82
PS D:\AUC\Junior\Fall 2022\CSC230302 - Comp Org. and Assmly Lang Prog\Projects\Project 2\Project 2> |
```

List of sequences simulated: (at least provide two 20 access sequences)

- Sequence of memory addresses used in test case 1:

16
28
357
204
820
1389
216
313
7224
2208
1177
1168
410
108
8900
3709
5004
2022
1447
1079

- Sequences of memory addresses used in test case 2:

32
42
420
204
820
1389
216
300
7224
700
1177
1168
410
455
8900
600
73
2022
1447
807

