Microprocessors Project
Milestone 1

Prof: Cherif Salama
TA: Jailan Salah

Done by:
1)Hussien Mohamed 25-7532
2)Karim Elbawab 25-10613
3)Mohamed Khaled 25-0880
4)Moaaz ElNashar 25-0996
5)Mostafa Adel 25-9450

Structure:

The cache class represents the real cache, with all of its properties
represented in the class's method.

The line class represents each line in the cache.

The word class represents each word (in bytes) in the line.

The simulator class used to simulate any assembly program, and calculates
the hit ratio and the AMAT.

Constructors:

Cache:- The cache constructor consists of the geometry needed to construct the the cache itself, using the S,L,M attributes. Also, The constructor consists of boolean variables allocate and through to determine the read and write policies of the cache, the cycle, miss, and hit integers are used to calculate the AMAT, array of integers memory representing the memory, and finally an array of integers of lines which is representing the lines of the cache.

Line:- The line constructor consists of boolean variables holdsData, dirty, occupied which are used to determine if a line is holding data, dirty, or occupied. Finally, It consists of an array of words constructing the line itself.

Word:- The word constructor consists of two integers value, and address.

Simulator:- The simulator consists of a hashmap insts representing my instructions, where the key is the pc of an instruction and the value is a string (The instruction itself), array of integers of size 7 where every index represent a register from R0 to R7, two arraylists of type cache one for the i-caches and the other for the d-caches, and integers pc, memoryCycle, and cyclesCounter.

Functionality:

Cache:- First, we compute the number of sets needed.Then, we divide the pc by the length of the word, then we get the set needed for putting in it this pc.
This brief explanation is used in all of the other methods for reading, and getting data and instructions, Also it's used in the replace and write procedures of data and instructions

Simulator:- The simulator compiles my assembly code, where it checks the syntax and it add the instructions read during the compilation to the hashmap of instructions mentioned above.
After the compilation is done, The simulator starts to simulate the assembly instructions saved in my hashmap and the methods of the cache class is used to start accessing, reading, and writing data and instructions in the cache.

Tasks:

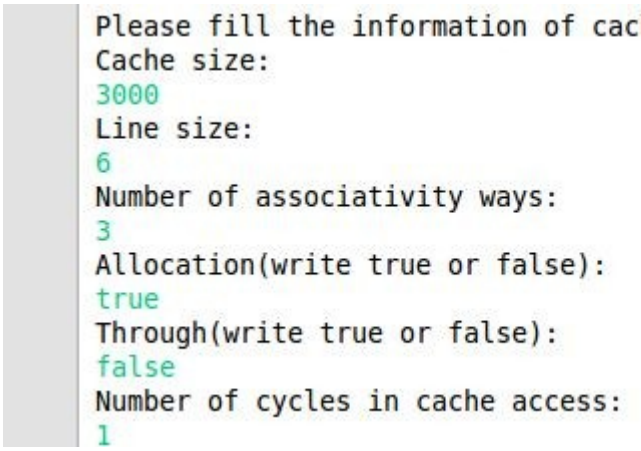
The Cache is done by Hussien.
The simulator is divided into two parts, the isValid method is done by Karim and Mostafa, the simulateInstruction method is done by Mohamed and Moaaz.

User Guide:

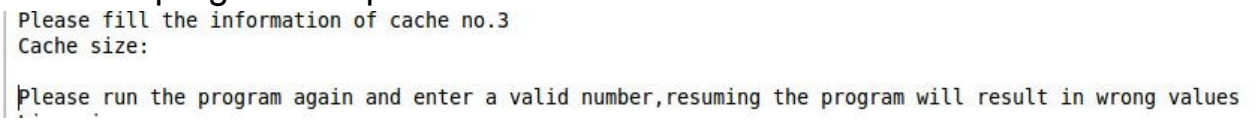
Before You start the program as a user you should provide the program by some parameters like the cache properties and memory properties and also your assembly code.

The program will start to ask you how many caches do you want as follows:

Then as in this example the program will ask the user to add the paramters of each cache (size, line size, allocate or not , etc ...) as in the images:



The program also tells you if you insert any inappropriate value that you should run the program again because in this case the program will put the default value.



At the end, the program will show you the result of the program (The hit value of each cache and the AMAT of the full code)

```
Here is the hit rate for all Cache levels :
Hit Rate of L1 : 0.45454545454545453
Hit Rate of L2 : 0.0
Hit Rate of L3 : 0.3333333333333333
AMAT of the program : 11.636363636363637
```

Tested Programs and Discussion :

Program 1 :

1
ADDI R1,R1,5
JALR R2,R1
ADDI R3,R3,3
JMP R3,3
MUL R1,R1,R1
NAND R3,R3,R1
RET R2
SW R3,R1,4
ADDI R3,R3,5
MUL R3,R3,R3
ADDI R4,R4,8
ADDI R5,R5,9
SUB R6,R5,R4

First Memory hierarchy : 1 Cache
:Geometry(1kB,2bytes,1),write through,write allocate,
L1 hit time = 3 cycles. Memory access penalty = 10 cycles.

Output : L1 hit rate = 0 %,AMAT = 13 cycles.

In this memory hierarchy,every line holds just one instruction so every instruction is an instruction miss since no instruction where repeated twice. There were only one memory access so obviously there was no hit in either instruction nor memory so hit ratio = 0 %.And AMAT = 3 + 100% * 10 = 13.

Second Memory hierarchy : 2 caches, 1st Cache :Geometry(2kB,4bytes,3),write through,write allocate,
L1 hit time = 3 cycles.
2nd Cache = Geometry(4kB,8bytes,1),write back,write around,
L2 hit time = 6 cycle
Memory access penalty = 10 cycles.

Output : L1 hit rate = 27.27 %,L2 hit rate = 50 %.
AMAT = 10.27 cycles.

The program haven't changed and no memory access instruction added. But,Line size is larger in L1 and even larger in L2 so more instructions will be cached every time which will improve the hit rate and consequently improves AMAT.

Program 2:

```
1
ADDI R1,R1,5
ADDI R4,R4,2
ADDI R2,R2,1
ADDI R3,R3,2
BEQ R1,R2,4
ADDI R5,R5,2
ADDI R2,R2,1
MUL R3,R3,R4
JMP R4,-7
SW R3,R5,3
LW R7,R5,3
```

First Memory hierarchy : 1 Cache :Geometry(1kB,2bytes,1),write through,write allocate,

L1 hit time = 3 cycles. Memory access penalty = 10 cycles.

Output : L1 hit rate = 58.6 %,AMAT = 7.13 cycles.

Although The memory hierarchy haven't changed than first program but the hit ratio was significantly improved since there was a loop and instructions were repeated and cached.

Second Memory hierarchy : 2 caches, 1st Cache :Geometry(2kB,4bytes,3),write through,write allocate,
L1 hit time = 3 cycles.
2nd Cache = Geometry(4kB,8bytes,1),write back,write around,
L2 hit time = 6 cycle
Memory access penalty = 10 cycles.

Output : L1 hit rate = 44.8 %,L2 hit rate = 75 %.
AMAT = 8.51 cycles.

Presence of second cache level decreases the hit rate of L1 than first one and since L2 hit time is much slower than L1, So The AMAT increased slightly.

Program 3:

```
1
ADDI R1,R1,2
ADDI R2,R2,3
NAND R3,R1,R2
SUB R4,R1,R3
SW R1,R4,8
SW R2,R4,9
SW R3,R4,10
LW R5,R4,8
LW R6,R4,9
LW R7,R4,10
```

First Memory hierarchy : 1 Cache
:Geometry(1kB,2bytes,1),write through,write allocate,
L1 hit time = 3 cycles. Memory access penalty = 10 cycles.

Output : L1 hit rate = 18.75 %,AMAT = 11.25 cycles.

There were no loops here and we didn't store more than instruction in time so all instructions were misses whoever there were some hit on data due to consecutive load and store operations.

Second Memory hierarchy : 2 caches, 1st Cache
:Geometry(2kB,4bytes,3),write through,write allocate,
L1 hit time = 3 cycles.
2nd Cache = Geometry(4kB,8bytes,1),write back,write around,
L2 hit time = 6 cycle
Memory access penalty = 10 cycles.

Output : L1 hit rate = 37.5 %,L2 hit rate = 50 %.
AMAT = 9.25 cycles.

Having larger line size improved hit rate for both caches in either accesses or just the instruction flow which helped in optimizing the AMAT.