

**Name: Shyamal Patel, Yousef Awimrin
and Christian Perez
ECE 366- Project 4- Cache
Report PDF**

PDF Report components:

Part A) (10 pts) Project reflections

1. List out your group members and contributions.

- Our group members are Shyamal Patel, Yousef Awimrin and Christian Perez

2. What are some of the features that you find useful in Python?

- Some features that we found useful in python was the use of list pop() and insert(). List pop() is used for removing return values from a list of a given index, and this is used throughout our project. Insert() is another function in python that is really helpful. It inserts an element at a given index in a list, and again that's something we need to do throughout this project. These functions are especially useful because they help make the LRU cache.

3. What would you advise other students about the projects for this course in the future?

- We would advise that we have a strong foundation for the previous projects. If something doesn't make sense it is important to get it cleared up (especially in projects 1 and 2). A good understanding of the previous projects will help a lot in this project.

Part B) (30 pts) Cache research with MARS

Cache configurations comparison: an illustrative table or chart – use MARS to collect cache hit / miss results by running the given test program (you can also change the parameters – the number marked by # in the program) with various cache configurations – try as many cases as possible. With your table / chart, provide some analysis on the question of “In general, what kind of cache configuration works the best, given the same total data size?”

	Version #1		Version #2		Version #3	
	Hit	Miss	Hit	Miss	Hit	Miss
For DM	65	24	55	9	52	11
For FA	67	22	54	10	51	12
For SA	67	22	54	10	51	12
For SA	67	22	54	10	51	12

A. How does block size affect cache performance?

- If we increase the block size, the cache performance would be reduced. Since the number of blocks that the cache can hold is reduced. By logic, this means that because there are less blocks, there would be fewer sets. When there are fewer sets that means collisions and misses are more likely to occur.

B. How does associativity (# of ways in a set) affect cache performance?

- Associativity are slower

- Adding more ways, but less sets did not affect the hit and miss for configuration 4.

Part C) (60 pts) Cache Simulator Results

Simulator showcase:

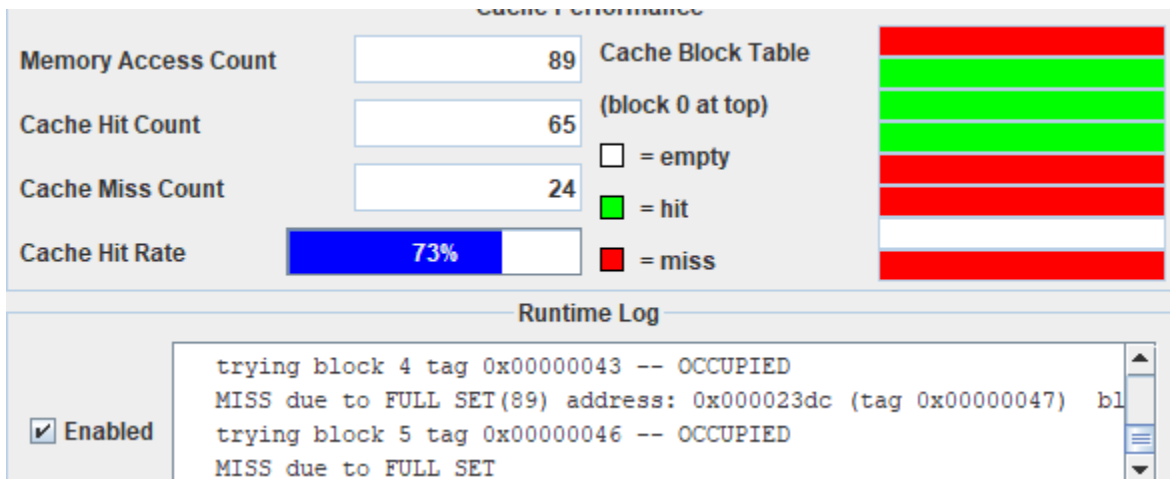
- A. Show the results of the test program (in appendix) for each cache configuration. Your result should confirm with MARS'.

cache configuration 1:

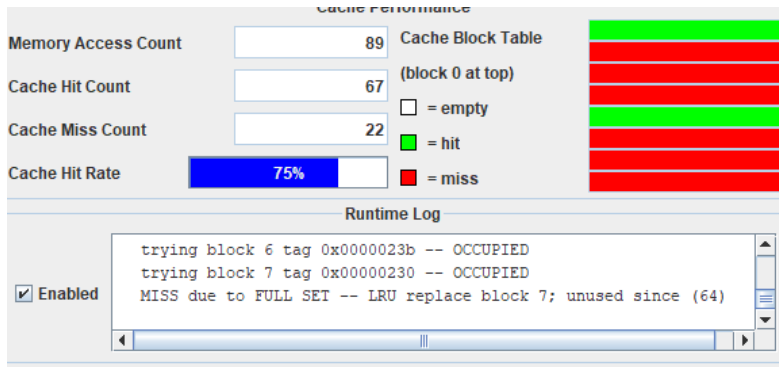
```
89) Memory Access Location(Hex): 0x23dc
Memory Access Location(Bin): 00000000000000000010001111011100
BreakDown:
  -Tag: 0000000000000000001000111 (0x47)
  -Set: 5 (0x5)
  -IN-blk offset:12 (0xc)

Cache Miss, Set is Occupied. Writing data into cache
Hit: 65, Miss: 24

Final HitRate is: 73%
```



cache configuration 2:



```
(89) Memory Access Location(Hex): 0x23dc
Memory Access Location(Bin): 0000000000000000010001111011100
BreakDown:
    -Tag: 000000000000000001000111101 (0x23d)
    -Set: 0 (0x0)
    -IN-blk offset:12 (0xc)
Block 0 -- OCCUPIED.
Block 1 -- OCCUPIED.
Block 2 -- OCCUPIED.
Block 3 -- OCCUPIED.
Block 4 -- OCCUPIED.
Block 5 -- OCCUPIED.
Block 6 -- OCCUPIED.

Trying block 7 tag 0x230 -- OCCUPIED.
All blocks are occupied, implementing LRU policy...
replacing block 7 since it is the least recently used block
Hit: 67, Miss: 22
```

```
Block 6 -- OCCUPIED.

Trying block 7 tag 0x230 -- OCCUPIED.
All blocks are occupied, implementing LRU policy...
replacing block 7 since it is the least recently used
Hit: 67, Miss: 22

Final HitRate is: 75%
```

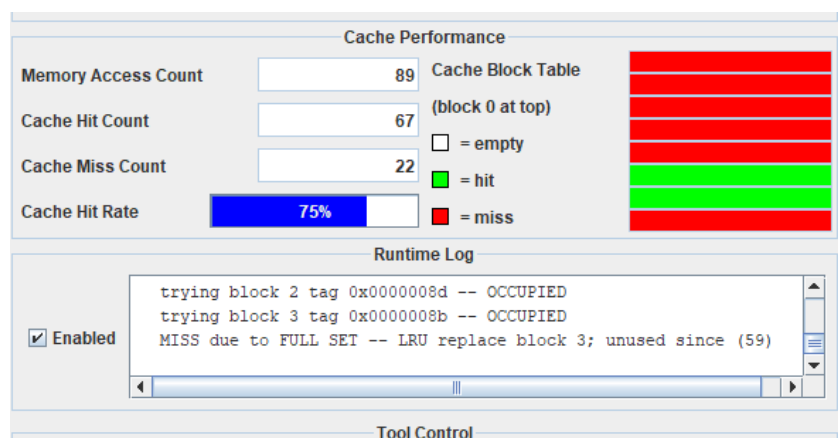
Part D) (extra credit 10 pts): Implement cache configuration 3 and 4 and include them in the report content above.

cache configuration 3:

```
(89) Memory Access Location(Hex): 0x23dc
Memory Access Location(Bin): 000000000000000010001111011100
BreakDown:
-Tag: 000000000000000010001111 (0x8f)
-Set: 1 (0x1)
-IN-blk offset:12 (0xc)

Trying Set 1 way 0 Tag:(0x8b) -- Occupied.

Trying Set 1 way 1 tag 0x8b -- OCCUPIED.
All blocks are occupied, implementing LRU policy...
replacing Set 1 way -1 since it is the least recently used block
Hit: 67, Miss: 22
```



```
All blocks are occupied, impl
replacing Set 1 way -1 since
Hit: 67, Miss: 22

Final HitRate is: 75%
```

cache configuration 4:

Hit: 67, Miss: 21

Trying Set 1 way 2 Tag:(0x11d) -- OCCUPIED.

Hit: 67, Miss: 21

All blocks are occupied, implementing LRU policy...
replacing Set 1 way 2 since it is the least recently used block

Hit: 67, Miss: 22

Memory Access Count	<input type="text" value="89"/>	Cache Block Table (block 0 at top) <input type="checkbox"/> = empty <input checked="" type="checkbox"/> = hit
Cache Hit Count	<input type="text" value="67"/>	
Cache Miss Count	<input type="text" value="22"/>	
Cache Hit Rate	<input type="text" value="75%"/>	

Final HitRate is: 75%

Appendix:

Version 1: test case

20080002
20090034
ad282000
11200005
2129fffc
00084022
2108ffef
01094026
1000fff9
20082174
20092000
212a003c
200e0003
8d2b0000
8d2c0000
018b682a
11a00001
000c5820
21290004
21ceffff
15c0fff9
ad0b0000
2108002c
2129fff8
012a682a
15a0fff2

Data Cache Simulation Tool, Version 1.2

Simulate and illustrate data cache performance


Cache Organization

Placement Policy: Number of blocks:

Block Replacement Policy: Cache block size (words):

Set size (blocks): Cache size (bytes):

Cache Performance

Memory Access Count: Cache Block Table (block 0 at top): 

Cache Hit Count: ☐ = empty

Cache Miss Count: ☒ = hit

Cache Hit Rate: ☐ = miss

Runtime Log

☒ Enabled

```
trying block 4 tag 0x00000043 -- OCCUPIED
MISS due to FULL SET(89) address: 0x0000023dc (tag 0x00000047) b1
trying block 5 tag 0x00000046 -- OCCUPIED
MISS due to FULL SET
```

Tool Control

```
Special: 0 0%
-----
Hit:65
Miss:24
.
```

Version 2: My own test case

34080002
20090060
ad282000
2129fffc
11200004
01084020
00084022
2108fffd
1000ff9
20082070
200a2060
20092000
200e0003
8d2b0000
21290004
8d2c0000
018b682a
11a00001
000c5820
21ceffff
15c0fff9
ad0b0000
21080004
012a682a
15a0fff3

Data Cache Simulation Tool, Version 1.2

Simulate and illustrate data cache performance

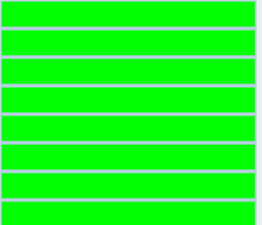
Cache Organization

Placement Policy: Number of blocks:

Block Replacement Policy: Cache block size (words):

Set size (blocks): Cache size (bytes):

Cache Performance

Memory Access Count: Cache Block Table: 

Cache Hit Count: (block 0 at top)

Cache Miss Count: ☐ = empty

Cache Hit Rate: ☐ = hit

☐ = miss

Runtime Log

☒ Enabled

```
trying block 6 tag 0x00000040 -- HIT
(64) address: 0x0000208c (tag 0x00000041) block range: 0-0
trying block 0 tag 0x00000041 -- HIT
```

Tool Control

Hit:55
Miss:9

Version 3: project 2 default code

20080003
3c091234
3529fedc
200b0001
200c2020
200f0015
0100682a
ad880000
01007020
000e57c3
01ca7026
01ca7022
01cb7020
11a00002
000e4020
08000011
000e4022
218c0004
216b0001
016f802a
1600fff1
200c2020
200b0014
20100000
20110000
20140000
200f0000
11600014
8d880000
216bffff
01096826
200e0020
11c00006
31aa0001
000d6842
21ceffff
1540fffb
21ef0001
15c0fff9
218e0060
adcf0000
01f0902a
16400003
000f8020
000c8820
0008a020
218c0004
0800001a
200d0000
ac142000
ac102004
Ac112008

Data Cache Simulation Tool, Version 1.2

Simulate and illustrate data cache performance

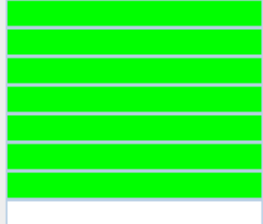
Cache Organization

Placement Policy: Number of blocks:

Block Replacement Policy: Cache block size (words):

Set size (blocks): Cache size (bytes):

Cache Performance

Memory Access Count: Cache Block Table: 

Cache Hit Count: (block 0 at top)

Cache Miss Count:

Cache Hit Rate:

☐ = empty
☒ = hit
☐ = miss

Runtime Log

☐ Enabled

Tool Control

```
Cache Hit, Valid bit is 1. Accessing  
Hit: 52, Miss: 11
```