Cache Simulator: Block Set Associative and Least Recently Used

Theoni Anne N. Lim
De La Salle University, theoni_lim@dlsu.edu.ph

Vincent Alvin S. Marquez
De La Salle University, vincent_marquez@dlsu.edu.ph

Joaquin Lorenzo T. Sitoy
De La Salle University, joaquin_sitoy@dlsu.edu.ph

Stanley Yale Z. Zeng
De La Salle University, stanley_yale_zeng@dlsu.edu.ph

Links:
Github: https://github.com/stanstan50/Cache_Simulation.git
Website: https://cache-simulation-jmww.onrender.com
Video Demo: https://drive.google.com/file/d/1TIl1ePo6A03tlWIeCH0q6T8NpMwVLgr_/view?usp=sharing

I. Introduction

The Cache Simulator demonstrates how a block set associative cache mapping using the Least Recently Used algorithm can be used to store a sequence of data. The web application uses the node.js framework and was deployed using Render.

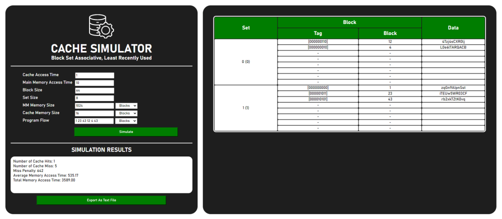
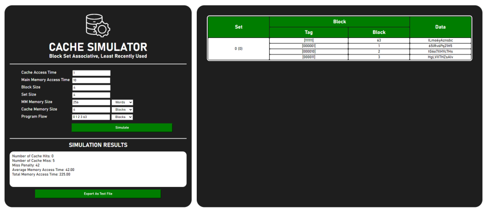
The front end was coded using standard HTML, CSS, and Javascript for user interaction. Inputs are manually typed in, but incrementors are also available. Dropdown selection boxes are available to switch between a block input (integers) or an address/word input (hex). A sequence of space-separated blocks or words are to be inputted in the Program Flow. These represent the data from main memory to be accessed sequentially. Pressing the "Simulate" button will output the cache snapshot to the right and the "Export As Text File" will export a text file of the time calculations and corresponding inputs.

The back end has a main simulate() function that initiates the cache accessing simulation. Functions like getUserInput() formats the inputs from the HTML-side into something more digestible for other functions while displayCache() feeds the cache snapshot (represented as a 2D array) back to the front end for display. Many helper functions such as parseProgramFlow(), hexadeciToDeci(), and createSetRow() are used for formatting, data conversion, and data creation. Inputs are error-checked before sending to the simulate() function.

II. Test Cases and Results

Inputs are shown on the left while the output/cache snapshot is shown on the right. Below the input box are the access time calculations. Indices for each test case are to the left of the screenshot column.

Table 1: Valid-Normal Input/Output

0		9	

Test case 0:

CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time: 1
Main Memory Access Time: 10
Block Size: 64
Set Size: 8
MM Memory Size: 1024 Blocks
Cache Memory Size: 16 Blocks
Program Flow: 1 23 43 12 4 43 Blocks

SIMULATION RESULTS

Number of Cache Hits: 1
Number of Cache Miss: 5
Miss Penalty: 642
Average Memory Access Time: 535.17
Total Memory Access Time: 3589.00

Set	Block		Data
	Tag	Block	
0 (0)	[000000110]	12	4Tzj4sCXR0lj
	[000000010]	4	L0e6ITARQACB
	-	-	
	-	-	
	-	-	
	-	-	
	-	-	
	-	-	
1 (1)	[000000000]	1	zqGn9AlpnSst
	[000001011]	23	iTEUw5WR03CF
	[000010101]	43	rb2xkTZtK0vq
	-	-	
	-	-	
	-	-	
	-	-	
	-	-	

Test case 9:

CACHE SIMULATOR
Block Set Associative, Least Recently Used

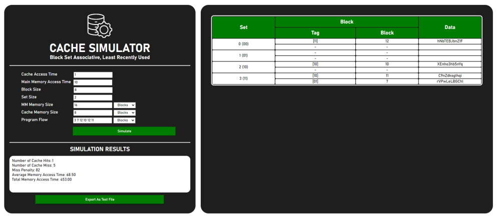
Cache Access Time: 1
Main Memory Access Time: 10
Block Size: 4
Set Size: 4
MM Memory Size: 256 Words
Cache Memory Size: 4 Blocks
Program Flow: 0 1 2 3 63 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 5
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 225.00

Set	Block		Data
	Tag	Block	
0 (0)	[111111]	63	lLmo6yAznsbc
	[000001]	1	65lRv6Pq21H5
	[000010]	2	tG6o7XHYc7Hs
	[000011]	3	HgLVXTHZsAlv

1



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	8	
Set Size	2	
MM Memory Size	16	Blocks
Cache Memory Size	8	Blocks
Program Flow	3 7 12 10 12 11	Blocks

Simulate

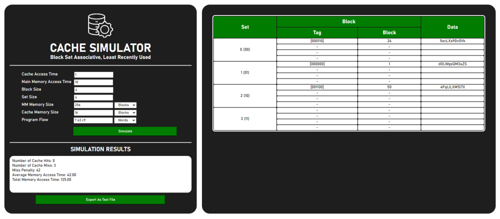
SIMULATION RESULTS

Number of Cache Hits: 1
Number of Cache Miss: 5
Miss Penalty: 82
Average Memory Access Time: 68.50
Total Memory Access Time: 453.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	[11]	12	hNbTEBJbnZ1F
	-	-	
1 (01)	-	-	
	-	-	
2 (10)	[10]	10	XEnho3hb5nYq
	-	-	
3 (11)	[10]	11	C9nZdksglhqi
	[01]	7	rVPwLeLBGChl

2



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	256	Blocks
Cache Memory Size	16	Blocks
Program Flow	7 63 c9	Words

Simulate

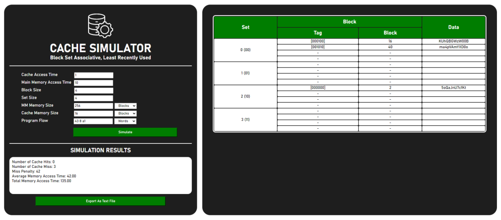
SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 3
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 135.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	[000110]	24	5ocLXs90v0Vk
	-	-	
	-	-	
	-	-	
1 (01)	[000000]	1	dOLWqsQM3uZS
	-	-	
	-	-	
	-	-	
2 (10)	[001100]	50	4FqIJLXM5I7X
	-	-	
	-	-	
	-	-	
3 (11)	-	-	
	-	-	
	-	-	
	-	-	

3



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	256	Blocks
Cache Memory Size	16	Blocks
Program Flow	43 8 a1	Words

Simulate

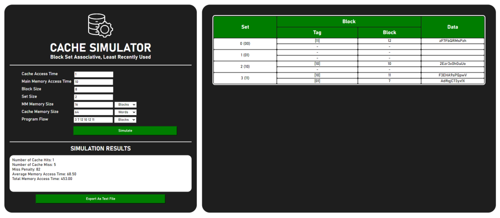
SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 3
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 135.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	[000100]	16	KUhQBGWzW00B
	[001010]	40	ma4pVAmYXD0o
	-	-	
	-	-	
1 (01)	-	-	
	-	-	
	-	-	
	-	-	
2 (10)	[000000]	2	5oQaJnU7cfKt
	-	-	
	-	-	
	-	-	
3 (11)	-	-	
	-	-	
	-	-	
	-	-	

4



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	8	
Set Size	2	
MM Memory Size	16	Blocks
Cache Memory Size	64	Words
Program Flow	3 7 12 10 12 11	Blocks

Simulate

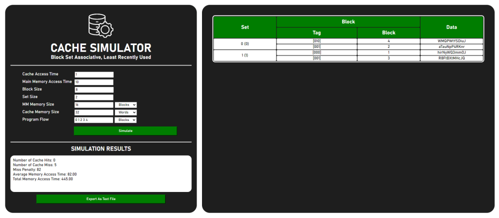
SIMULATION RESULTS

Number of Cache Hits: 1
Number of Cache Miss: 5
Miss Penalty: 82
Average Memory Access Time: 68.50
Total Memory Access Time: 453.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	[11]	12	zF7FbQRMsPzh
	-	-	
1 (01)	-	-	
	-	-	
2 (10)	[10]	10	2Ezr3x0hGuUo
	-	-	
3 (11)	[10]	11	F3EHA9sPQpwV
	[01]	7	AdRqjC73yxfX

5



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	8	
Set Size	2	
MM Memory Size	16	Blocks
Cache Memory Size	32	Words
Program Flow	0 1 2 3 4	Blocks

Simulate

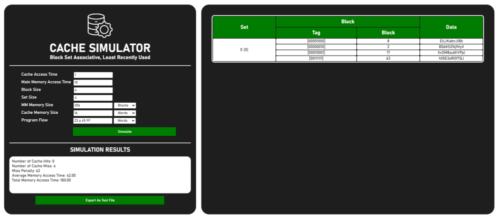
SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 5
Miss Penalty: 82
Average Memory Access Time: 82.00
Total Memory Access Time: 445.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (0)	[010]	4	WMQPWtYSDiuJ
	[001]	2	aTauNpP4RKnr
1 (1)	[000]	1	hirNyWQ3mmOJ
	[001]	3	R8FtBXtMHcJQ

6



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	256	Blocks
Cache Memory Size	16	Words
Program Flow	23 a 45 FF	Words

Simulate

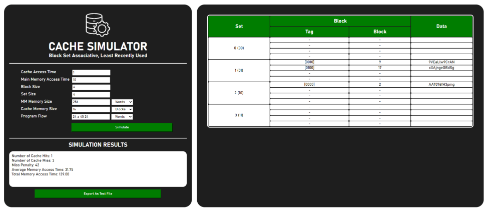
SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 4
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 180.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (0)	[00001000]	8	EILIKabnJl8b
	[00000010]	2	B06K5ZNjfHyX
	[00010001]	17	XvDM8auWVPpl
	[00111111]	63	hfGE3sROf7QJ

10



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	256	Words
Cache Memory Size	16	Blocks
Program Flow	24 a 45 24	Words

Simulate

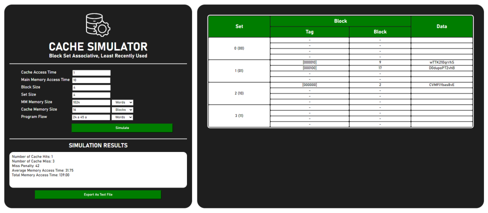
SIMULATION RESULTS

Number of Cache Hits: 1
Number of Cache Miss: 3
Miss Penalty: 42
Average Memory Access Time: 31.75
Total Memory Access Time: 139.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	-	-	
	-	-	
	-	-	
	-	-	
1 (01)	[0010]	9	9VEaLlw9CrAN
	[0100]	17	cXAjngeGBdSg
	-	-	
	-	-	
2 (10)	[0000]	2	AAT076fH3pmg
	-	-	
	-	-	
	-	-	
3 (11)	-	-	
	-	-	
	-	-	
	-	-	

11



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	1024	Words
Cache Memory Size	16	Blocks
Program Flow	24 a 45 a	Words

Simulate

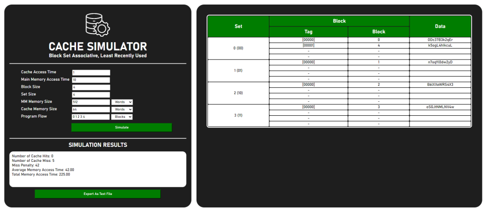
SIMULATION RESULTS

Number of Cache Hits: 1
Number of Cache Miss: 3
Miss Penalty: 42
Average Memory Access Time: 31.75
Total Memory Access Time: 139.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	-	-	
	-	-	
	-	-	
	-	-	
1 (01)	[000010]	9	wTTK210qrrhS
	[000100]	17	D0dupoPT2vhB
	-	-	
	-	-	
2 (10)	[000000]	2	CVMFlYbas8vE
	-	-	
	-	-	
	-	-	
3 (11)	-	-	
	-	-	
	-	-	
	-	-	

12



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	512	Words
Cache Memory Size	64	Words
Program Flow	0 1 2 3 4	Blocks

Simulate

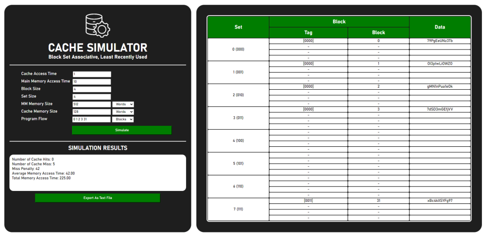
SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 5
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 225.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	[00000]	0	ODc37B3k2qEr
	[00001]	4	k5ogL4hlkcuL
	-	-	
	-	-	
1 (01)	[00000]	1	n7oqYl0dw2yD
	-	-	
	-	-	
	-	-	
2 (10)	[00000]	2	B6lXXeWRS4X3
	-	-	
	-	-	
	-	-	
3 (11)	[00000]	3	oSGJtNMLNV4w
	-	-	
	-	-	
	-	-	

13



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	512	Words
Cache Memory Size	128	Words
Program Flow	0 1 2 3 31	Blocks

Simulate

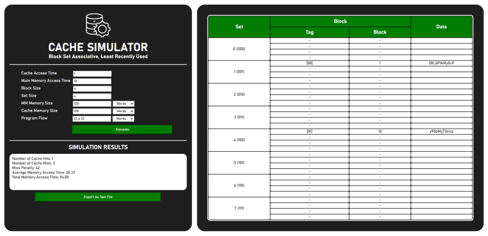
SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 5
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 225.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (000)	[0000]	0	79PgEeUNz3Tb
	-	-	
	-	-	
	-	-	
1 (001)	[0000]	1	Ol3pIwLiOWZO
	-	-	
	-	-	
	-	-	
2 (010)	[0000]	2	gMN1nPua1eOk
	-	-	
	-	-	
	-	-	
3 (011)	[0000]	3	7dSD3mGEfjVV
	-	-	
	-	-	
	-	-	
4 (100)	-	-	
	-	-	
	-	-	
	-	-	
5 (101)	-	-	
	-	-	
	-	-	
	-	-	
6 (110)	-	-	
	-	-	
	-	-	
	-	-	
7 (111)	[0011]	31	xBc4kXSYFg97
	-	-	
	-	-	
	-	-	

14



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	128	Words
Cache Memory Size	128	Words
Program Flow	23 a 23	Words

Simulate

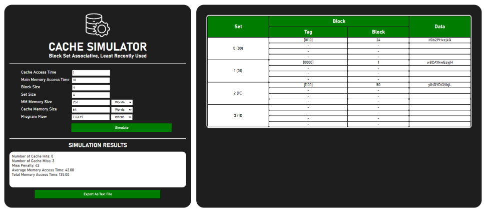
SIMULATION RESULTS

Number of Cache Hits: 1
Number of Cache Miss: 2
Miss Penalty: 42
Average Memory Access Time: 28.33
Total Memory Access Time: 94.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (000)	-	-	
	-	-	
	-	-	
	-	-	
1 (001)	[00]	1	28L6PtkMy0cP
	-	-	
	-	-	
	-	-	
2 (010)	-	-	
	-	-	
	-	-	
	-	-	
3 (011)	-	-	
	-	-	
	-	-	
	-	-	
4 (100)	[01]	12	y90bMy73olzz
	-	-	
	-	-	
	-	-	
5 (101)	-	-	
	-	-	
	-	-	
	-	-	
6 (110)	-	-	
	-	-	
	-	-	
	-	-	
7 (111)	-	-	
	-	-	
	-	-	
	-	-	

15



CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1	
Main Memory Access Time	10	
Block Size	4	
Set Size	4	
MM Memory Size	256	Words
Cache Memory Size	64	Words
Program Flow	7 63 c9	Words

Simulate

SIMULATION RESULTS

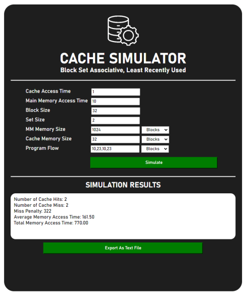
Number of Cache Hits: 0
Number of Cache Miss: 3
Miss Penalty: 42
Average Memory Access Time: 42.00
Total Memory Access Time: 135.00

Export As Text File

Set	Block		Data
	Tag	Block	
0 (00)	[0110]	24	if0b2PHxzjkQ
	-	-	
	-	-	
	-	-	
1 (01)	[0000]	1	w8CAYkwEsyjH
	-	-	
	-	-	
	-	-	
2 (10)	[1100]	50	ylNDYDt3VbjL
	-	-	
	-	-	
	-	-	
3 (11)	-	-	
	-	-	
	-	-	
	-	-	

Table 2: Valid-Special Input/Output

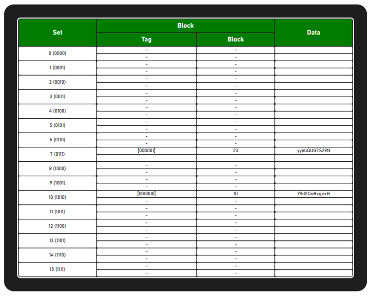
0


CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1
Main Memory Access Time	10
Block Size	32
Set Size	2
MM Memory Size	1024 Blocks
Cache Memory Size	32 Blocks
Program Flow	10,23,10,23 Blocks

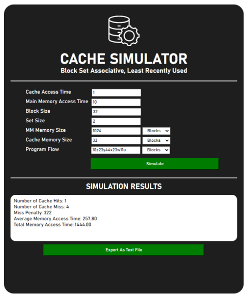
SIMULATION RESULTS

```
Number of Cache Hits: 2
Number of Cache Miss: 2
Miss Penalty: 322
Average Memory Access Time: 161.50
Total Memory Access Time: 770.00
```



Set	Block		Data
	Tag	Block	
0 (0000)	-	-	
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	[000001]	23	yy6bQUG7S29N
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	[000000]	10	Y9d2UoRvgexH
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

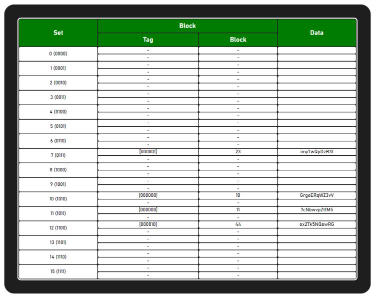
1


CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1
Main Memory Access Time	10
Block Size	32
Set Size	2
MM Memory Size	1024 Blocks
Cache Memory Size	32 Blocks
Program Flow	10z23y44x23w11u Blocks

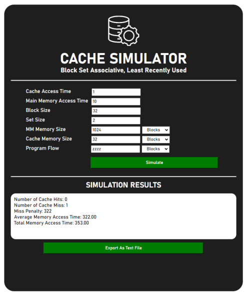
SIMULATION RESULTS

```
Number of Cache Hits: 1
Number of Cache Miss: 4
Miss Penalty: 322
Average Memory Access Time: 257.80
Total Memory Access Time: 1444.00
```



Set	Block		Data
	Tag	Block	
0 (0000)	-	-	
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	[000001]	23	imy7wQpDzR3f
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	[000000]	10	GrgoERqWZ3vV
	-	-	
11 (1011)	[000000]	11	7cNbwvpZtfM5
	-	-	
12 (1100)	[000010]	44	ox2Tk5NQswRG
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

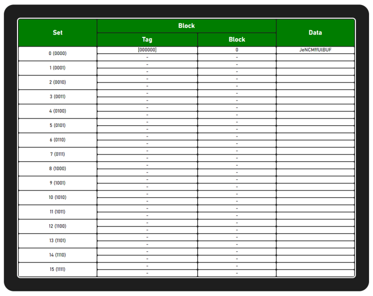
2


CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1
Main Memory Access Time	10
Block Size	32
Set Size	2
MM Memory Size	1024 Blocks
Cache Memory Size	32 Blocks
Program Flow	zzzz Blocks

SIMULATION RESULTS

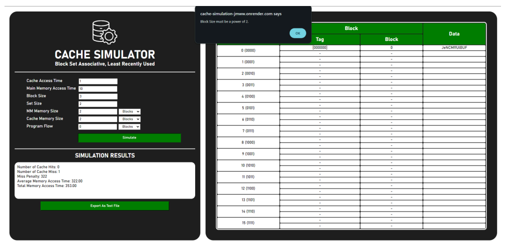
```
Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00
```



Set	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUlBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

Table 3: Invalid Input

0


cache-simulation-jmww.onrender.com says
Block Size must be a power of 2.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

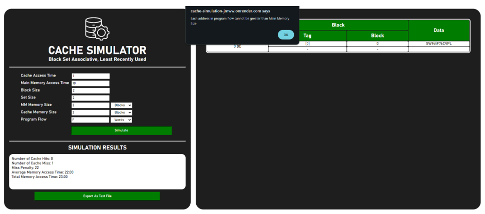
Cache Access Time	1
Main Memory Access Time	10
Block Size	3
Set Size	2
MM Memory Size	2 Blocks
Cache Memory Size	2 Blocks
Program Flow	0 Blocks

SIMULATION RESULTS

```
Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00
```

	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUlBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

6


cache-simulation-jmww.onrender.com says
Each address in program flow cannot be greater than Main Memory Size

CACHE SIMULATOR
Block Set Associative, Least Recently Used

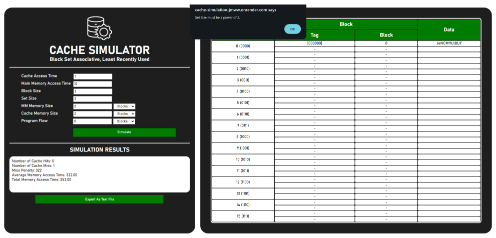
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	2 Blocks
Cache Memory Size	2 Blocks
Program Flow	F Words

SIMULATION RESULTS

```
Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 22
Average Memory Access Time: 22.00
Total Memory Access Time: 23.00
```

	Block		Data
	Tag	Block	
0 (0)	[0]	0	SW9dtF76CVPL
	-	-	

1


cache-simulation-jmww.onrender.com says
Set Size must be a power of 2.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

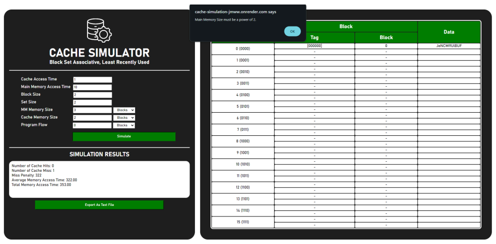
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	3
MM Memory Size	2 Blocks
Cache Memory Size	2 Blocks
Program Flow	0 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00

	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUIBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

2


cache-simulation-jmww.onrender.com says
Main Memory Size must be a power of 2.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

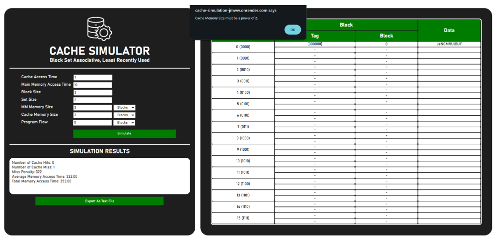
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	3 Blocks
Cache Memory Size	2 Blocks
Program Flow	0 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00

	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUIBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

3


cache-simulation-jmww.onrender.com says
Cache Memory Size must be a power of 2.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

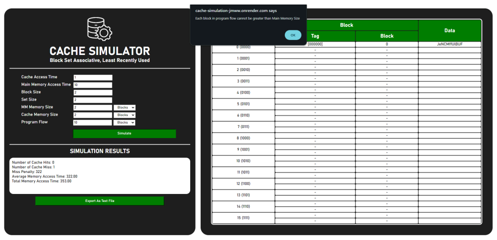
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	2 Blocks
Cache Memory Size	3 Blocks
Program Flow	0 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00

	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUIBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

4


cache-simulation-jmww.onrender.com says
Each block in program flow cannot be greater than Main Memory Size

CACHE SIMULATOR
Block Set Associative, Least Recently Used

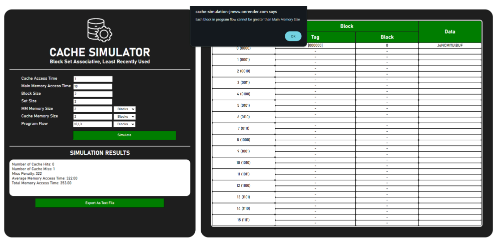
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	2 Blocks
Cache Memory Size	2 Blocks
Program Flow	10 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00

	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUIBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

5


cache-simulation-jmww.onrender.com says
Each block in program flow cannot be greater than Main Memory Size

CACHE SIMULATOR
Block Set Associative, Least Recently Used

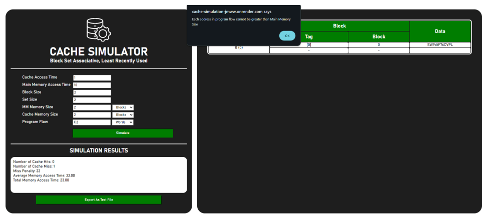
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	2 Blocks
Cache Memory Size	2 Blocks
Program Flow	10,1,3 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 322
Average Memory Access Time: 322.00
Total Memory Access Time: 353.00

	Block		Data
	Tag	Block	
0 (0000)	[000000]	0	JeNCMffUIBUF
	-	-	
1 (0001)	-	-	
	-	-	
2 (0010)	-	-	
	-	-	
3 (0011)	-	-	
	-	-	
4 (0100)	-	-	
	-	-	
5 (0101)	-	-	
	-	-	
6 (0110)	-	-	
	-	-	
7 (0111)	-	-	
	-	-	
8 (1000)	-	-	
	-	-	
9 (1001)	-	-	
	-	-	
10 (1010)	-	-	
	-	-	
11 (1011)	-	-	
	-	-	
12 (1100)	-	-	
	-	-	
13 (1101)	-	-	
	-	-	
14 (1110)	-	-	
	-	-	
15 (1111)	-	-	
	-	-	

7


cache-simulation-jmww.onrender.com says
Each address in program flow cannot be greater than Main Memory Size

CACHE SIMULATOR
Block Set Associative, Least Recently Used

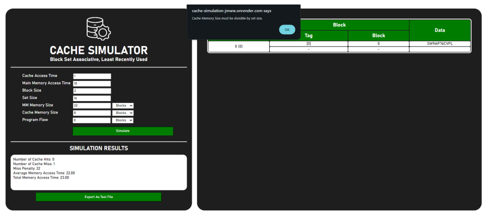
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	2 Blocks
Cache Memory Size	2 Blocks
Program Flow	F,2 Words

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 22
Average Memory Access Time: 22.00
Total Memory Access Time: 23.00

	Block		Data
	Tag	Block	
0 (0)	[0]	0	SW9dtF76CVPL
	-	-	

8


cache-simulation-jmww.onrender.com says
Cache Memory Size must be divisible by set size.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

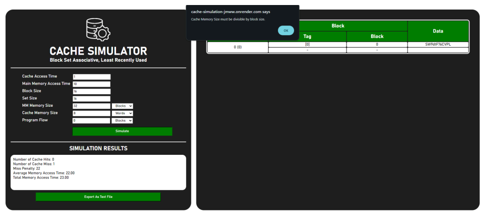
Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	16
MM Memory Size	32 Blocks
Cache Memory Size	8 Blocks
Program Flow	0 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 22
Average Memory Access Time: 22.00
Total Memory Access Time: 23.00

	Block		Data
	Tag	Block	
0 (0)	[0]	0	SW9dtF76CVPL
	-	-	

9


cache-simulation-jmww.onrender.com says
Cache Memory Size must be divisible by block size.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

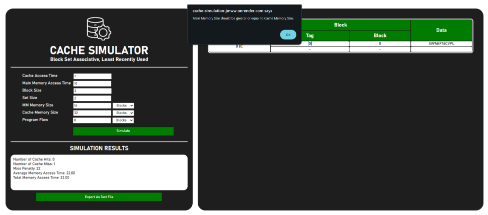
Cache Access Time	1
Main Memory Access Time	10
Block Size	16
Set Size	16
MM Memory Size	32 Blocks
Cache Memory Size	8 Words
Program Flow	0 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 22
Average Memory Access Time: 22.00
Total Memory Access Time: 23.00

	Block		Data
	Tag	Block	
0 (0)	[0]	0	SW9dtF76CVPL
	-	-	

10


cache-simulation-jmww.onrender.com says
Main Memory Size should be greater or equal to Cache Memory Size.

CACHE SIMULATOR
Block Set Associative, Least Recently Used

Cache Access Time	1
Main Memory Access Time	10
Block Size	2
Set Size	2
MM Memory Size	16 Blocks
Cache Memory Size	32 Blocks
Program Flow	0 Blocks

SIMULATION RESULTS

Number of Cache Hits: 0
Number of Cache Miss: 1
Miss Penalty: 22
Average Memory Access Time: 22.00
Total Memory Access Time: 23.00

	Block		Data
	Tag	Block	
0 (0)	[0]	0	SW9dtF76CVPL
	-	-	

III. Analysis and Discussion

To verify the results of the simulator, manual computations were written to clarify the specifications of the block set associative, LRU mapping.

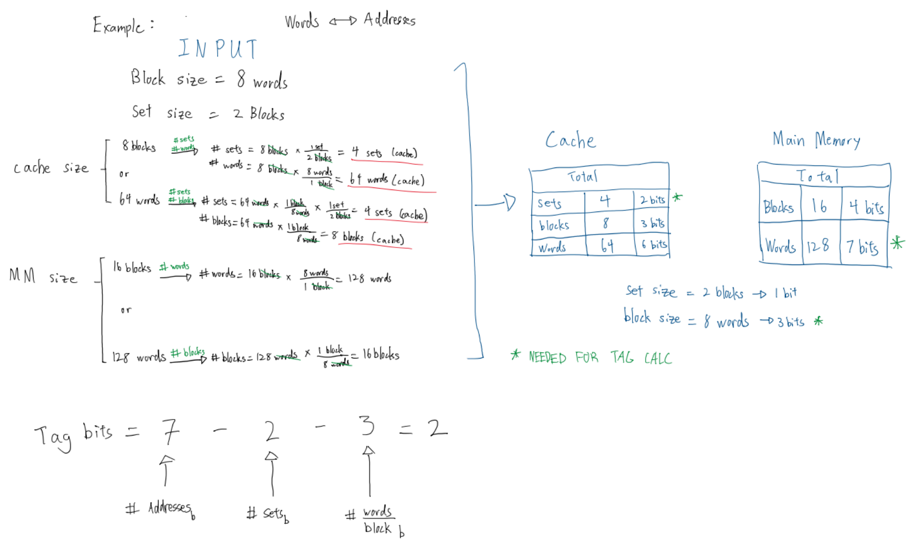

Example:
Words ↔ Addresses
INPUT
Block size = 8 words
Set size = 2 Blocks
cache size
8 blocks → # sets = 8 blocks × 1 set / 2 blocks = 4 sets (cache)
# words = 8 blocks × 8 words / 1 block = 64 words (cache)
or
64 words → # sets = 64 words × 1 block / 8 words × 1 set / 2 blocks = 4 sets (cache)
# blocks = 64 words × 1 block / 8 words = 8 blocks (cache)
MM size
16 blocks → # words = 16 blocks × 8 words / 1 block = 128 words
or
128 words → # blocks = 128 words × 1 block / 8 words = 16 blocks
Cache
Total
Sets 4 2 bits *
blocks 8 3 bits
Words 64 6 bits
Main Memory
Total
Blocks 16 4 bits
Words 128 7 bits *
Set size = 2 blocks → 1 bit
block size = 8 words → 3 bits *
* NEEDED FOR TAG CALC
Tag bits = 7 - 2 - 3 = 2
# Addresses_b
# sets_b
# words/block_b

Figure 1: Manual Computation of Test Case 1 of Valid/Normal Input

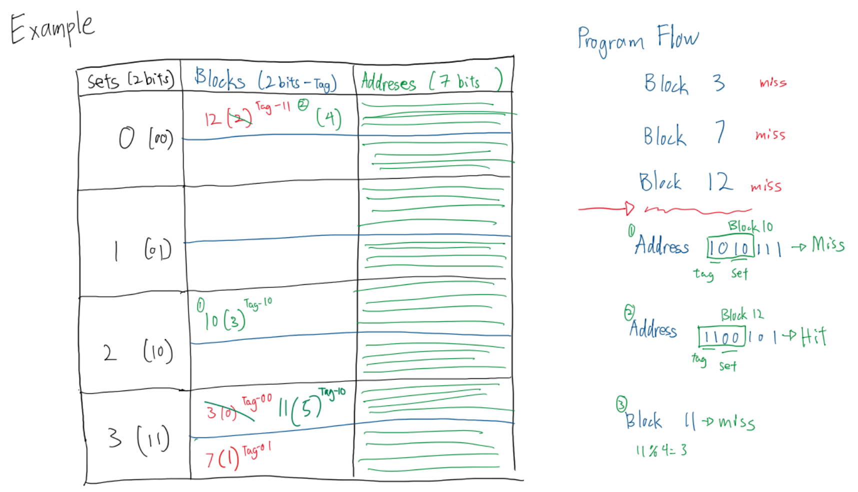
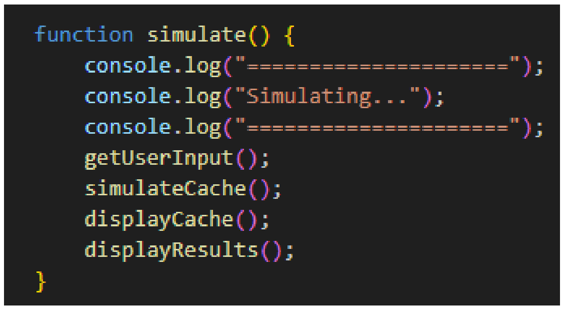


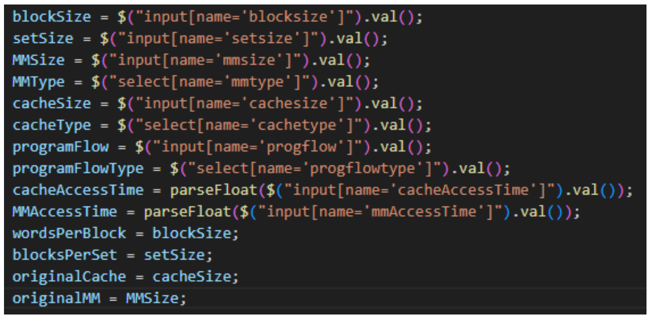
Figure 2: Manual Computation of Test Case 1 of Valid/Normal Input (cont.)

The figures above show hand calculations of Test Case 1 from Table 1 and in the video demonstration, this was the second test case to be shown. As seen from the example, blocks 3, 7, 12, and 10 were misses, resulting in them being stored in their respective sets using the modulo operation, with the age in parentheses. The second block 12 initiated a hit, updating the age in Set 0, Block 0. Block 11 resulted in a miss. The modulo operation allocates it to Set 3, but since it's already full, the least recently used value (block 3), gets replaced. This manual computation reflects the snapshot shown in the simulator. For a detailed analysis of the process, the main program flow is as follows:

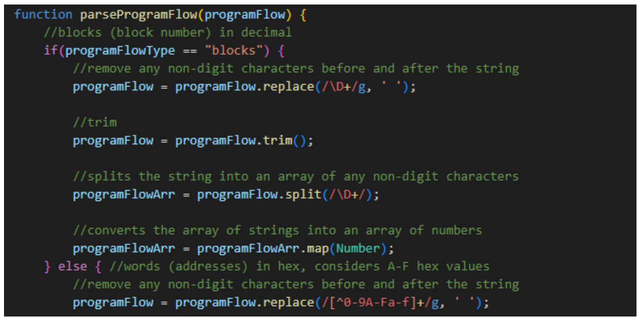


```
function simulate() {
    console.log("====================");
    console.log("Simulating...");
    console.log("====================");
    getUserInput();
    simulateCache();
    displayCache();
    displayResults();
}
```

Figure 3: Main Program Flow



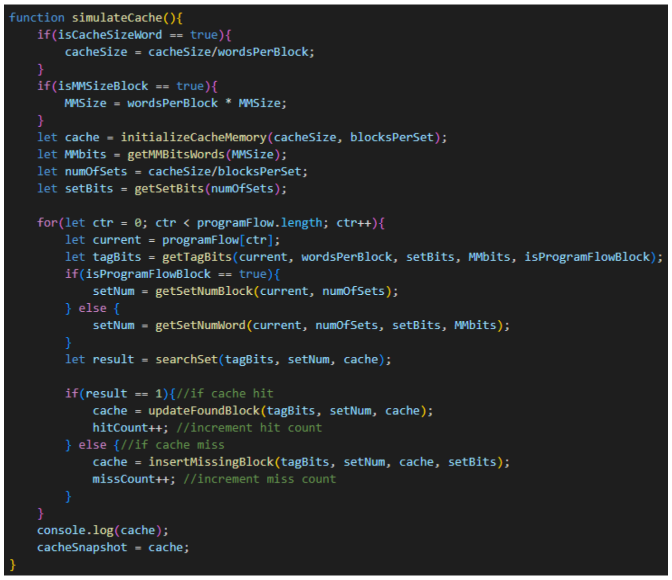
```
blockSize = $("input[name='blocksize']").val();
setSize = $("input[name='setsize']").val();
MMSize = $("input[name='mmsize']").val();
MMType = $("select[name='mmtype']").val();
cacheSize = $("input[name='cachesize']").val();
cacheType = $("select[name='cachetype']").val();
programFlow = $("input[name='progflow']").val();
programFlowType = $("select[name='progflowtype']").val();
cacheAccessTime = parseFloat($("input[name='cacheAccessTime']").val());
MMAccessTime = parseFloat($("input[name='mmAccessTime']").val());
wordsPerBlock = blockSize;
blocksPerSet = setSize;
originalCache = cacheSize;
originalMM = MMSize;
```



```
function parseProgramFlow(programFlow) {
    //blocks (block number) in decimal
    if(programFlowType == "blocks") {
        //remove any non-digit characters before and after the string
        programFlow = programFlow.replace(/\D+/g, ' ');

        //trim
        programFlow = programFlow.trim();

        //splits the string into an array of any non-digit characters
        programFlowArr = programFlow.split(/\D+/);

        //converts the array of strings into an array of numbers
        programFlowArr = programFlowArr.map(Number);
    } else { //words (addresses) in hex, considers A-F hex values
        //remove any non-digit characters before and after the string
        programFlow = programFlow.replace(/[^0-9A-Fa-f]+/g, ' ');
```

Figure 4: Code Snippets of getUserInput() (left) and parseProgramFlow() (right)

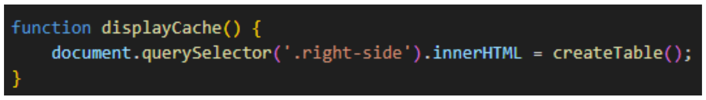
The function, getUserInput(), uses jquery to extract the text from the website's input fields. The function, parseProgramFlow() is then called to parse the input from the program flow field, using any unrelated characters such as special characters or non-hexadecimal letters as separators between the actual data. If no valid blocks or hexadecimal characters are present, the program flow will default to 0 as an input. This ensures that the program doesn't crash despite invalid inputs.



```
function simulateCache(){
    if(isCacheSizeWord == true){
        cacheSize = cacheSize/wordsPerBlock;
    }
    if(isMMSizeBlock == true){
        MMSize = wordsPerBlock * MMSize;
    }
    let cache = initializeCacheMemory(cacheSize, blocksPerSet);
    let MMbits = getMMBitsWords(MMSize);
    let numOfSets = cacheSize/blocksPerSet;
    let setBits = getSetBits(numOfSets);

    for(let ctr = 0; ctr < programFlow.length; ctr++){
        let current = programFlow[ctr];
        let tagBits = getTagBits(current, wordsPerBlock, setBits, MMbits, isProgramFlowBlock);
        if(isProgramFlowBlock == true){
            setNum = getSetNumBlock(current, numOfSets);
        } else {
            setNum = getSetNumWord(current, numOfSets, setBits, MMbits);
        }
        let result = searchSet(tagBits, setNum, cache);

        if(result == 1){//if cache hit
            cache = updateFoundBlock(tagBits, setNum, cache);
            hitCount++; //increment hit count
        } else {//if cache miss
            cache = insertMissingBlock(tagBits, setNum, cache, setBits);
            missCount++; //increment miss count
        }
    }
    console.log(cache);
    cacheSnapshot = cache;
}
```

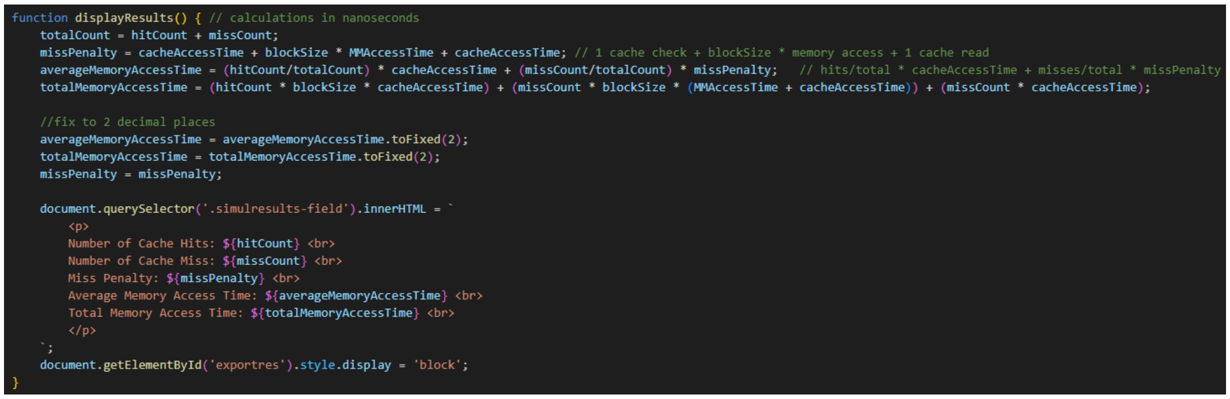
Figure 5: Code Snippet of simulateCache()

Function simulateCache() is then called which performs the main operations and computations to simulate cache accessing and obtain the final cache snapshot. The function first checks whether the cache and main memory are in either blocks or words. It then converts the Cache Memory Size into cacheSize (cache size in blocks) and MM Memory Size into MMSize (main memory size in words) for the simplicity of using a single variable. Variable cacheSize will be used for creating the 2D array representation of the cache memory while MMSize will be used to compute the number of bits needed to represent an address. A for-loop was implemented to iterate over each element in the formatted program flow array. For each element, the program extracts the bits needed to search for an address in the cache, checking for a cache hit or miss. Should a hit occur, the program updates the block's age for the LRU algorithm and increments the hit counter. For a miss, the program inserts the missing block into the cache and similarly updates the age and miss counter. The final representation is saved in the cacheSnapshot variable. This whole process is to simulate, step-by-step, the cache operations in order to ensure the program is accurate to real world situations.



```
function displayCache() {
    document.querySelector('.right-side').innerHTML = createTable();
}
```

Figure 6: Code Snippet of displayCache()



```javascript
function displayResults() { // calculations in nanoseconds
    totalCount = hitCount + missCount;
    missPenalty = cacheAccessTime + blockSize * MMAccessTime + cacheAccessTime; // 1 cache check + blockSize * memory access + 1 cache read
    averageMemoryAccessTime = (hitCount/totalCount) * cacheAccessTime + (missCount/totalCount) * missPenalty;   // hits/total * cacheAccessTime + misses/total * missPenalty
    totalMemoryAccessTime = (hitCount * blockSize * cacheAccessTime) + (missCount * blockSize * (MMAccessTime + cacheAccessTime)) + (missCount * cacheAccessTime);

    //fix to 2 decimal places
    averageMemoryAccessTime = averageMemoryAccessTime.toFixed(2);
    totalMemoryAccessTime = totalMemoryAccessTime.toFixed(2);
    missPenalty = missPenalty;

    document.querySelector('.simulresults-field').innerHTML = `
        <p>
        Number of Cache Hits: ${hitCount} <br>
        Number of Cache Miss: ${missCount} <br>
        Miss Penalty: ${missPenalty} <br>
        Average Memory Access Time: ${averageMemoryAccessTime} <br>
        Total Memory Access Time: ${totalMemoryAccessTime} <br>
        </p>
    `;
    document.getElementById('exportres').style.display = 'block';
}
```

Figure 7: Code Snippet of displayResults()

Function displayCache() is then called to update the final snapshot on the right hand side of the front page. Function displayResults() computes the memory access time, taking into account the hit count and miss penalty. The formulas are as shown in the figure above. The final product of displayResults is shown at the bottom of the front page, below the inputs, and can be downloaded as a text file.

IV. Conclusion

When accessing data, it is common to have data that is more frequently used than others. Thus, it is efficient to have storage devices such as caches closer to the processor for faster access. Different cache mapping algorithms are used to determine how to store the data and simulators are useful for visualizing the resulting cache structure. Compared to other mappings such as direct mapping or fully associative, block set associative provides a balance between quick access and data overlap prevention. Test cases in the documentation show how the simulator can be used, illustrating the type of input needed and corresponding output, including error-handling. This project provided valuable insight on the inner workings of cache memory.