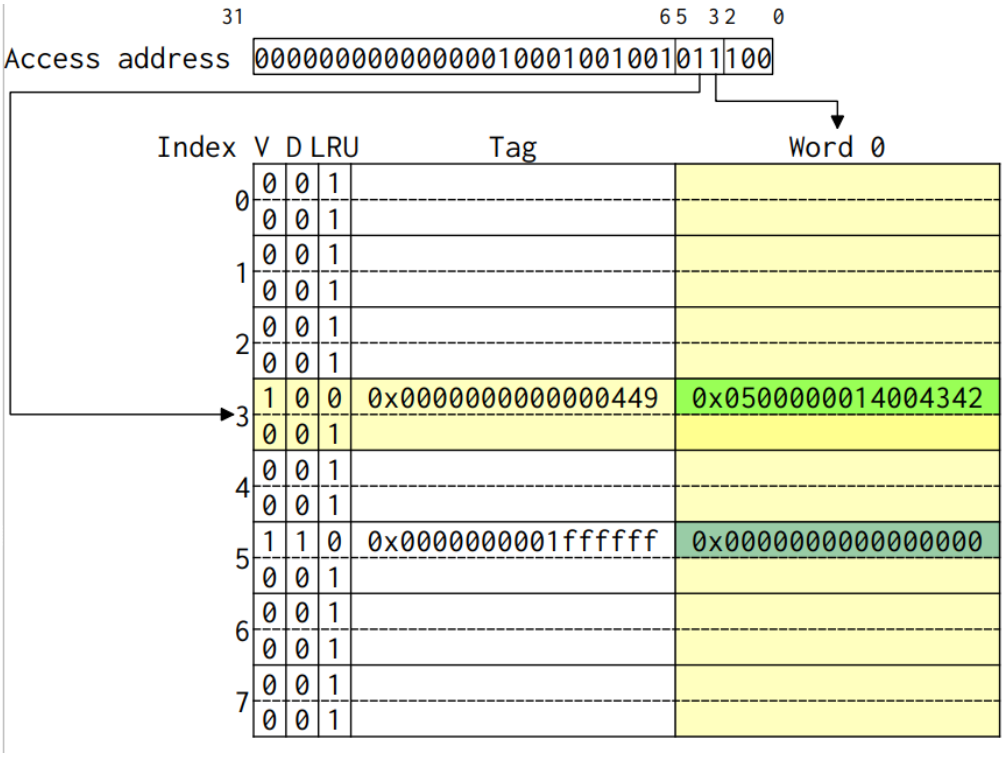
Computer Architecture HW6

Q1

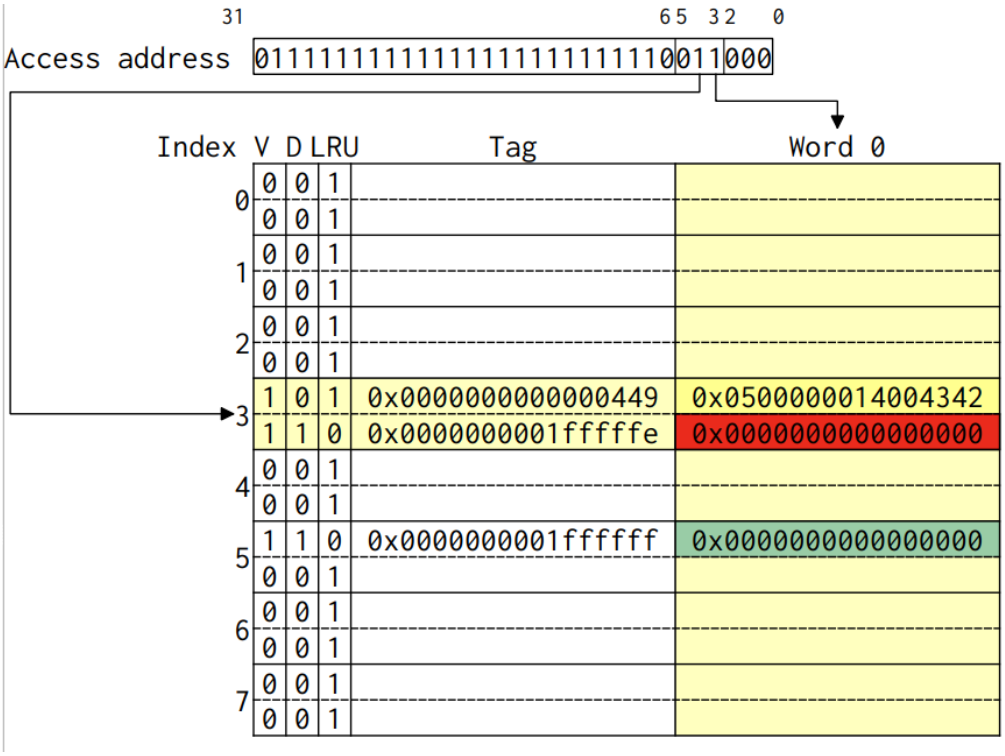
(a)

Instruction: `100f8:  00112023  sw x1 0 x2`

Before:



After:



Explanation:

With write allocate, we fetch the block on a write miss.

After the write miss at index 3, the valid bit (V) and dirty

bit (D) will be 1, and will change the LRU bit of occupied way to 1, the LRU bit of inserted way to 0, representing that the block is the most recently used.

(b)

Instruction: `100b4: 00912223 sw x9 4 x2`

Before:

31 … 65 32 0

Access address 0000000000000001000100100100 1100

Index	V	D	LRU	Tag	Word 0
0	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4142414442414241
1	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4143424142414443
2	1	0	0	0x0000000000000449	0x4142410043424142
	1	1	1	0x0000000001ffffff	0x0000000000000000
3	1	0	1	0x0000000001fffffe	0x000000000001019c
	1	0	0	0x0000000000000449	0x0500000014004342
4	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000001fffffe	0x0000000100000000
5	1	0	0	0x0000000001fffffe	0x0000000000000002
	1	1	1	0x0000000001ffffff	0x0001023800000000
6	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		
7	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		

After:

31 … 65 32 0

Access address 0111111111111111111111110011000

Index	V	D	LRU	Tag	Word 0
0	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4142414442414241
1	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4143424142414443
2	1	0	0	0x0000000000000449	0x4142410043424142
	1	1	1	0x0000000001ffffff	0x0000000000000000
3	1	1	0	0x0000000001fffffe	0x000000000001019c
	1	0	1	0x0000000000000449	0x0500000014004342
4	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000001fffffe	0x0000000100000000
5	1	0	0	0x0000000001fffffe	0x0000000000000002
	1	1	1	0x0000000001ffffff	0x0001023800000000
6	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		
7	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		

Explanation:

There is a write hit at index 3, change the way-1 dirty bit to 1, the LRU bit to 0 to represent that the block is the most recently used.(way-2 to 1)

(c)

Instruction: `100b8: 00112023 sw x1 0 x2`

Before:

31 … 6 5 3 2 0

Access address 0111111111111111111111110 011 000

Index	V	D	LRU	Tag	Word 0
0	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4142414442414241
1	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4143424142414443
2	1	0	0	0x0000000000000449	0x4142410043424142
	1	1	1	0x0000000001ffffff	0x0000000000000000
3	1	1	0	0x0000000001fffffe	0x000000000001019c
	1	0	1	0x0000000000000449	0x0500000014004342
4	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000001fffffe	0x0000000100000000
5	1	0	0	0x0000000001fffffe	0x0000000000000002
	1	1	1	0x0000000001ffffff	0x0001023800000000
6	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		
7	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		

After:

31 … 6 5 3 2 0

Access address 0111111111111111111111110 010 100

Index	V	D	LRU	Tag	Word 0
0	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4142414442414241
1	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4143424142414443
2	1	0	1	0x0000000000000449	0x4142410043424142
	1	1	0	0x0000000001fffffe	0x0000000000000000
3	1	1	0	0x0000000001fffffe	0x000000000001019c
	1	0	1	0x0000000000000449	0x0500000014004342
4	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000001fffffe	0x0000000100000000
5	1	0	0	0x0000000001fffffe	0x0000000000000002
	1	1	1	0x0000000001ffffff	0x0001023800000000
6	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		
7	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		

Explanation:

With write allocate, we fetch the block on a write miss. After the write miss at index 2, the valid bit (V) and dirty bit (D) will be 1, change the LRU bit to 0 to represent that the block is the most recently used.

Since the block to be replaced is dirty, we have a write back here.

Q2

Original:

L1 Data Cache | L1 Instr. Cache

Cache configuration:

Preset:

2^N Lines:	3	Repl. policy:	LRU
2^N Ways:	1	Wr. hit:	Write-back
2^N Words/Line:	0	Wr. miss:	Write allocate

Plot configuration:

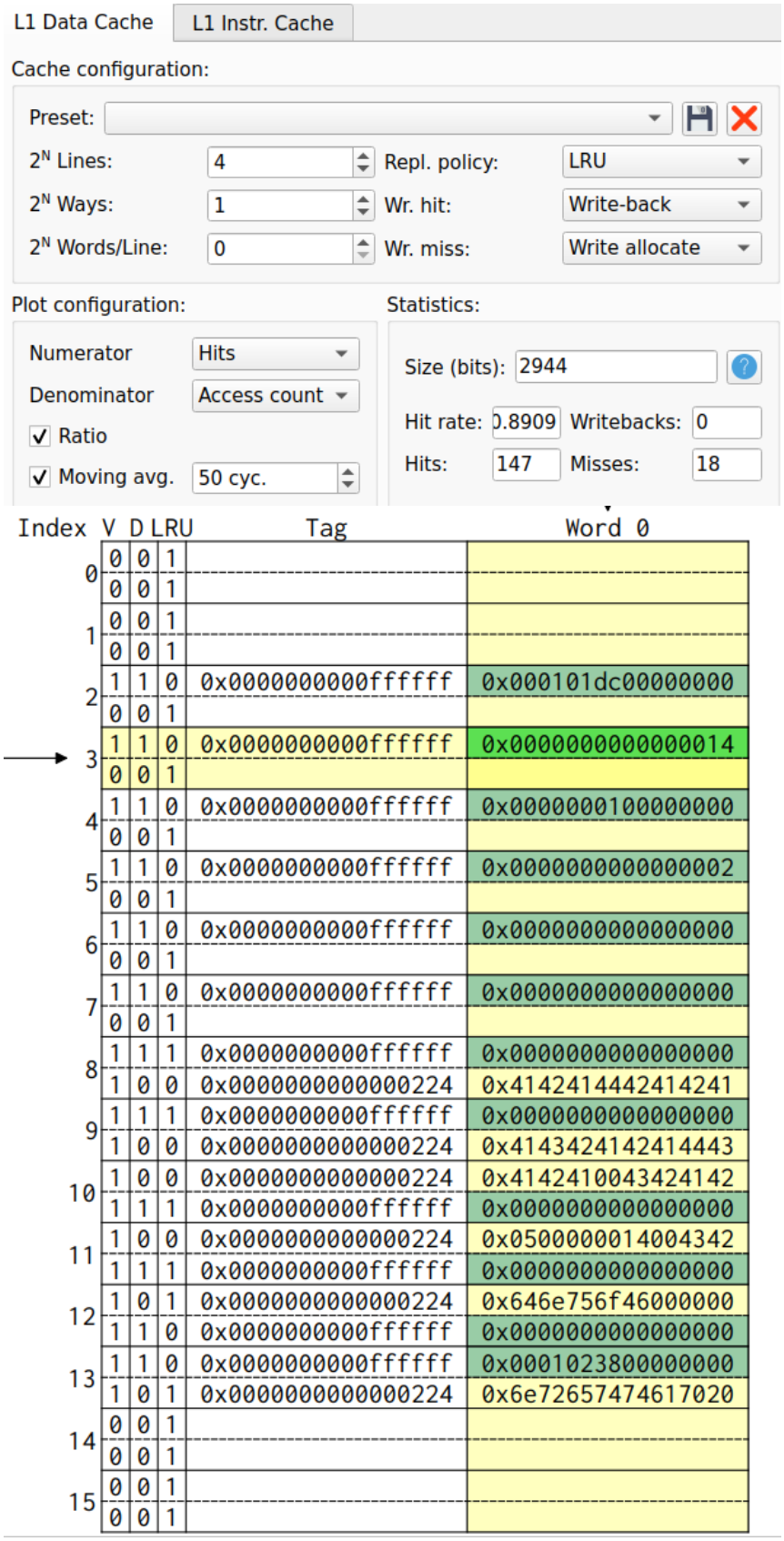
Numerator	Hits
Denominator	Access count
☑ Ratio	
☑ Moving avg.	50 cyc.

Statistics:

Size (bits):	1488		
Hit rate:	0.8606	Writebacks:	6
Hits:	142	Misses:	23

Index	V	D	LRU	Tag	Word 0
0	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4142414442414241
1	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000000000449	0x4143424142414443
2	1	0	1	0x0000000000000449	0x4142410043424142
	1	1	0	0x0000000001fffffe	0x000101dc00000000
→ 3	1	1	0	0x0000000001fffffe	0x0000000000000014
	1	0	1	0x0000000000000449	0x0500000014004342
4	1	1	1	0x0000000001ffffff	0x0000000000000000
	1	0	0	0x0000000001fffffe	0x0000000100000000
5	1	0	0	0x0000000001fffffe	0x0000000000000002
	1	1	1	0x0000000001ffffff	0x0001023800000000
6	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		
7	1	1	0	0x0000000001fffffe	0x0000000000000000
	0	0	1		

My cache:



Hit rate: 0.8606 -> 0.8909

Number of sets: 8 -> 16

Since the number of sets doubled, the chance of conflict & capacity miss get lower.

Q3

(a)

	Tag(bits)	Index(bits)	Block offset(bits)	Total size(bits)
Cache 1	16	4	2	784
Cache 2	16	2	4	580
Cache 3	17	2	3	656
Cache 4	18	0	4	588

Word address: 20 bits
Cache 1:
16 blocks => Index = 4
Tag = 20 - 4 = 16
Total size = 16 * (32 + 16 + 1) = 784
Cache 2:
4 blocks => Index = 2
Tag = 20 - 2 - 2 = 16
Total size = 4 * (4 * 32 + 16 + 1) = 580
Cache 3:
8 blocks, 2-way => 4 sets => Index = 2
Tag = 20 - 2 - 1 = 17
Total size = 8 * (2 * 32 + 17 + 1) = 656
Cache 4:
4 blocks, Fully associative => Index = 0
Tag = 20 - 1 * 2 = 18
Total size = 4 * (4 * 32 + 18 + 1) = 588

(b)

Word-address references:
16, 17, 18, 19, 20, 48, 49, 17, 48, 49, 17, 5, 6, 7

I.

Cache 2:

16	17	18	19	20	48
compulsory	hit	hit	hit	compulsory	confl
49	17	48	49	17	5
hit	capacity	capacity	hit	capacity	confl
6	7				
hit	hit				

Cache 3:

16	17	18	19	20	48
compulsory	hit	compulsory	hit	compulsory	compul
49	17	48	49	17	5
hit	hit	hit	hit	hit	compul
6	7				
compulsory	hit				

Cache 4:

16	17	18	19	20	48
compulsory	hit	hit	hit	compulsory	compulsory
49	17	48	49	17	5
hit	hit	hit	hit	hit	compulsory
6	7				
hit	hit				

II.

Cache 2:

Block	word 1	word 2	word 3	word 4
0	16	17	18	19
1	5	6	7	8
2				
3				

Cache 3:

	block 1	block 1	block 2	block 2
set	word 1	word 2	word 1	word 2
0	16	17	48	49
1	18	19		
2	20	21	4	5
3	6	7		

Q4

(a)

A = 1101100
h1: 1010
h2: 1000
h4: 1100
H = 010
=> bit2(p2) is in error
=> A’ = 1001100

(b)

B = 10110101
h1: 1100
h2: 0110
h4: 1010
h8: 1
H = 0001
h_n = odd
=> bit8(p8) is in error
=> B’ = 10110100

(c)

C = 10001011
h1: 1011
h2: 0001
h4: 0101
h8: 1
H = 1101
h_n = even
=> double error occurred

Q5

(a)

Cache 1

miss penalty: 5 + 4 = 9
instruction miss: 70000 * 0.04 * 9 = 25200
data miss: 70000 * 1/3 * 0.03 * 9 = 6300
cycles spent on cache misses: 31500

Cache 2

miss penalty: 5 + 2 = 7
instruction miss: 70000 * 0.04 * 7 = 19600
data miss: 70000 * 1/3 * 0.03 * 7 = 4900
cycles spent on cache misses: 24500

(b)

Cache 3: CPI = 1.7
instruction miss: 70000 * 0.04 * 6 = 16800
data miss: 70000 * 1/3 * 0.03 * 6 = 4200
cycles spent on cache misses: 21000

Cache 1

CPI: (70000 * 1.7 - 21000 + 31500) / 70000 = 1.85

Cache 2

CPI: (70000 * 1.7 - 21000 + 24500) / 70000 = 1.75

Q6

Virtual address	Physical address	Cache Hit/Miss	TLB Hit/Miss
0x954a16c2	0x3b9416c2	Miss	Miss
0x6542c746	0x0ae6c746	Miss	Miss
0x954a1647	0x3b941647	Miss	Hit
0x6542c412	0x0ae6c412	Miss	Hit
0x2b74c4d3	0x14acc4d3	Miss	Miss
0x6542c46a	0x0ae6c46a	Hit	Hit
0x954a16dd	0x3b9416dd	Hit	Hit
0x6542c417	0x0ae6c417	Miss	Hit
0x2b74c723	0x14acc723	Miss	Hit