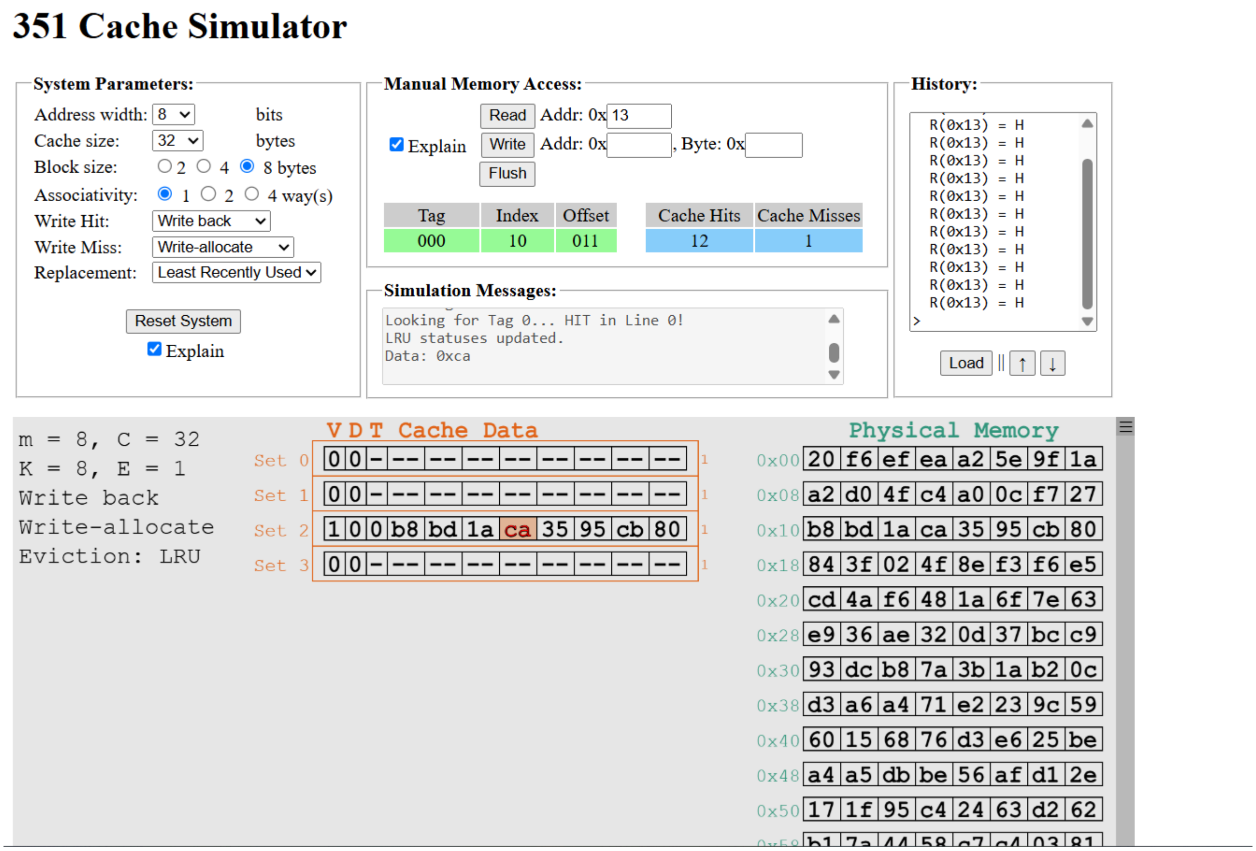
Practical 4: Memory Cache

1. **Execute a Read memory request to address 0x13**



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

☑ Explain

Addr: 0x 13

Addr: 0x , Byte: 0x

Tag	Index	Offset
000	10	011

Cache Hits	Cache Misses
12	1

Simulation Messages:

```
Looking for Tag 0... HIT in Line 0!
LRU statuses updated.
Data: 0xca
```

History:

```
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
>
```

```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

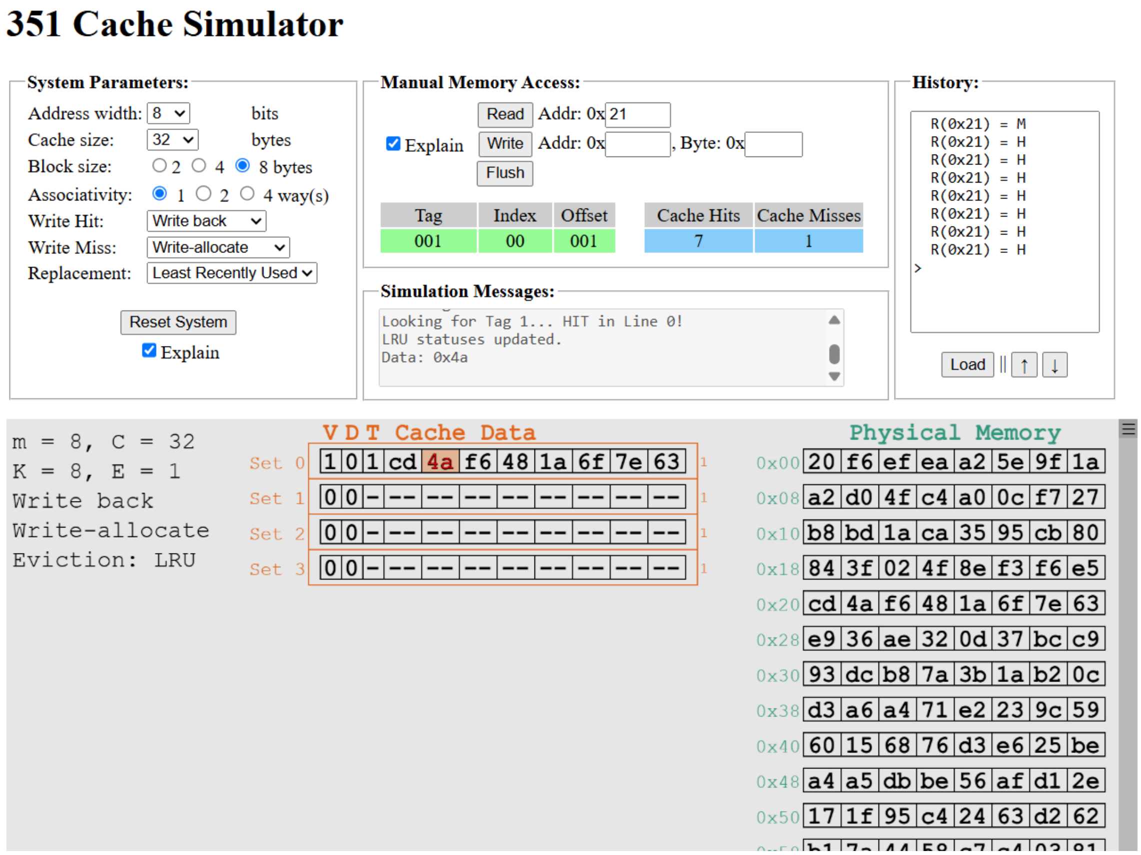
	V	D	T								
Set 0	0	0	-	--	--	--	--	--	--	--	--
Set 1	0	0	-	--	--	--	--	--	--	--	--
Set 2	1	0	0	b8	bd	1a	ca	35	95	cb	80
Set 3	0	0	-	--	--	--	--	--	--	--	--

Physical Memory

0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62
0x58	b1	7a	44	58	c7	c4	03	81

- Address breakdown: 0001 00 11 (tag=0001, index=00, offset=11)
- First access (R(0x13)):
    + Cache miss (block not in cache)
    + Loads block (addresses 0x10-0x13) into Set 0
    + Data at 0x13 is 0xca (from physical memory)
    + Miss count increments to 1
- Subsequent accesses:
    + All hits (same block already in cache)
    + Hit count reaches 12 (some hits not shown in history)
- Cache state:
    + Set 0 contains tag 0000 (partial view shown)
    + LRU tracking working correctly
    ○ Set 0 contains tag 0000 (partial view shown)
    ○ LRU tracking working correctly

2. Execute a Read memory request to address 0x21



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Read Addr: 0x21
☑ Explain Write Addr: 0x , Byte: 0x
Flush

Tag	Index	Offset	Cache Hits	Cache Misses
001	00	001	7	1

Simulation Messages:

```
Looking for Tag 1... HIT in Line 0!
LRU statuses updated.
Data: 0x4a
```

History:

```
R(0x21) = M
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
>
```

```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

	V	D	T	Cache Data								
Set 0	1	0	1	cd	4a	f6	48	1a	6f	7e	63	1
Set 1	0	0	-	--	--	--	--	--	--	--	--	1
Set 2	0	0	-	--	--	--	--	--	--	--	--	1
Set 3	0	0	-	--	--	--	--	--	--	--	--	1

Physical Memory

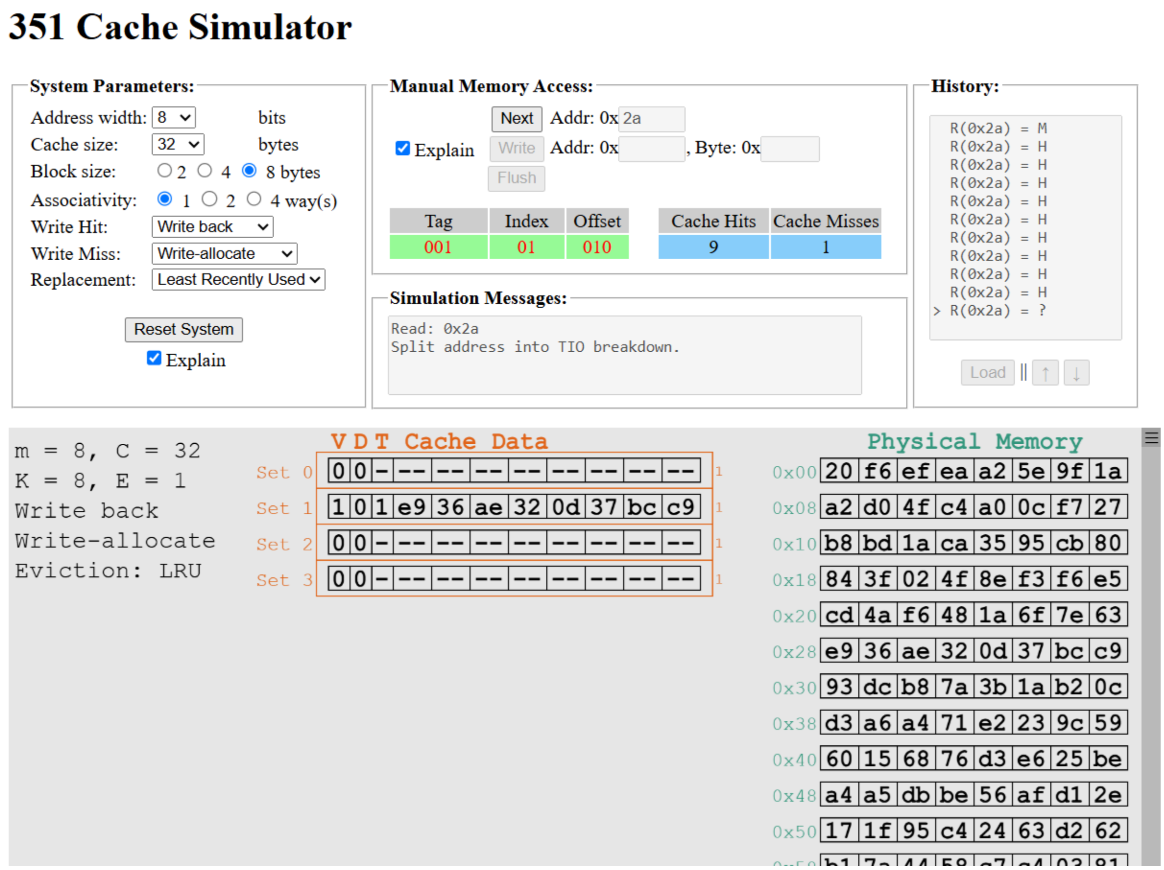
0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62
0x58	b1	7a	44	58	a7	c4	03	81

- Access: Read from address 0x21
- Address breakdown: 001 00 001 (tag=001, index=00, offset=001)
- First access (R(0x21)):
  + Cache miss (M in history)
  + Loads 8-byte block (0x20-0x27) into Set 0
  + Data at 0x21 is 0x4a
- Subsequent accesses:

+ All hits (H in history)
+ Total hits: 12, misses:

- Cache behavior:
  + Direct mapped (1-way) means only one block per set
  + No conflict issues in this access pattern

3. Execute a Read memory request to address 0x2a



351 Cache Simulator

System Parameters:

Address width: 8 bits

Cache size: 32 bytes

Block size: ○ 2 ○ 4 ● 8 bytes

Associativity: ● 1 ○ 2 ○ 4 way(s)

Write Hit: Write back

Write Miss: Write-allocate

Replacement: Least Recently Used

Manual Memory Access:

Addr: 0x 2a

Addr: 0x , Byte: 0x

Tag	Index	Offset
001	01	010

Cache Hits	Cache Misses
9	1

History:

```
  R(0x2a) = M
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
> R(0x2a) = ?
```

Simulation Messages:

```
Read: 0x2a
Split address into TIO breakdown.
```

```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

	V	D	T								
Set 0	0	0	-	--	--	--	--	--	--	--	--
Set 1	1	0	1	e9	36	ae	32	0d	37	bc	c9
Set 2	0	0	-	--	--	--	--	--	--	--	--
Set 3	0	0	-	--	--	--	--	--	--	--	--

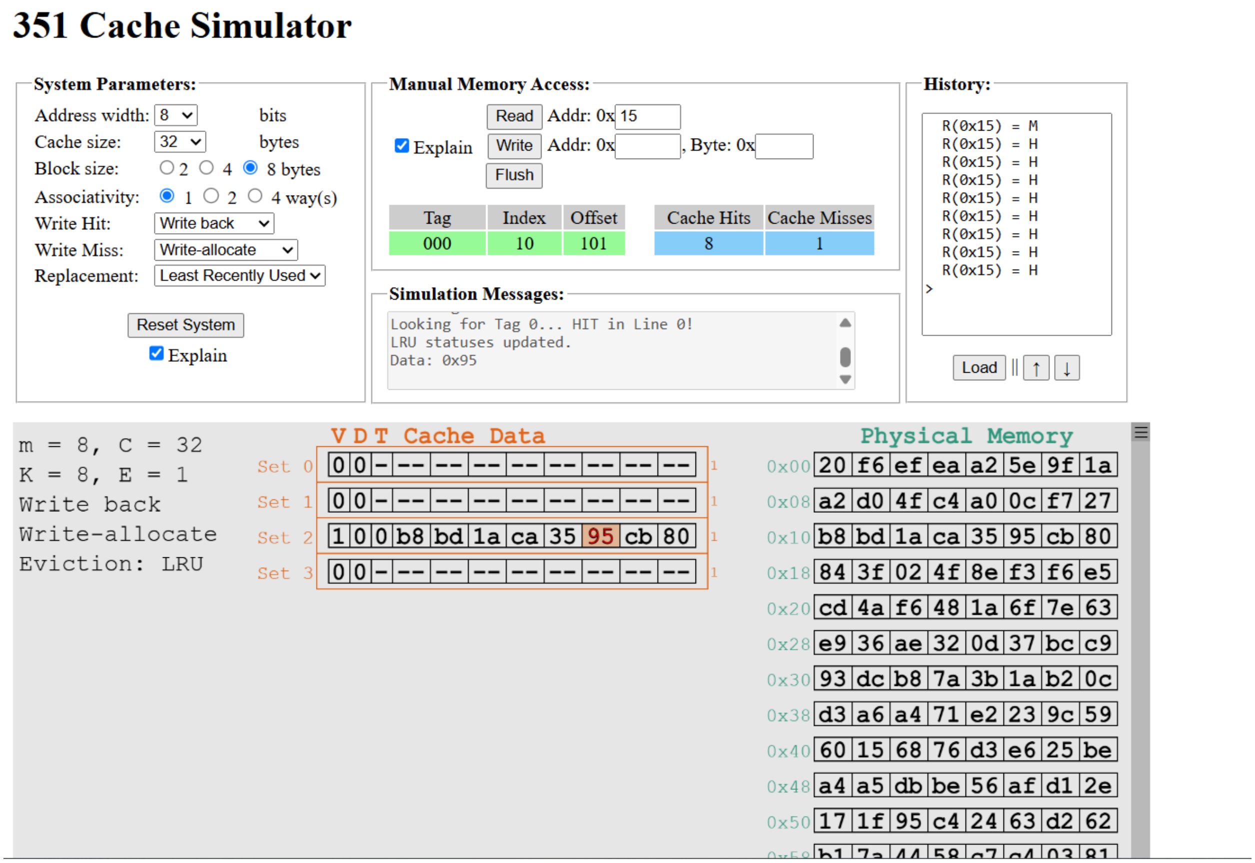
Physical Memory

0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62
0x58	b1	7a	44	58	c7	c4	03	81

- Address breakdown: 001 01 010 (tag=001, index=01, offset=010)
- First access (R(0x2a)):
  - Cache miss (not shown but implied by later hits)
  - Loads block (0x28-0x2f) into Set 1

- Subsequent accesses:
  - All hits (H in history)
  - Total hits: 9, misses: 1

4. Execute a Read memory request to address 0x15:



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

Reset System
☑ Explain

Manual Memory Access:

Read Addr: 0x 15
☑ Explain Write Addr: 0x , Byte: 0x
Flush

Tag	Index	Offset		Cache Hits	Cache Misses
000	10	101		8	1

Simulation Messages:

```
Looking for Tag 0... HIT in Line 0!
LRU statuses updated.
Data: 0x95
```

History:

```
R(0x15) = M
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
>
```

Load || ↑ ↓

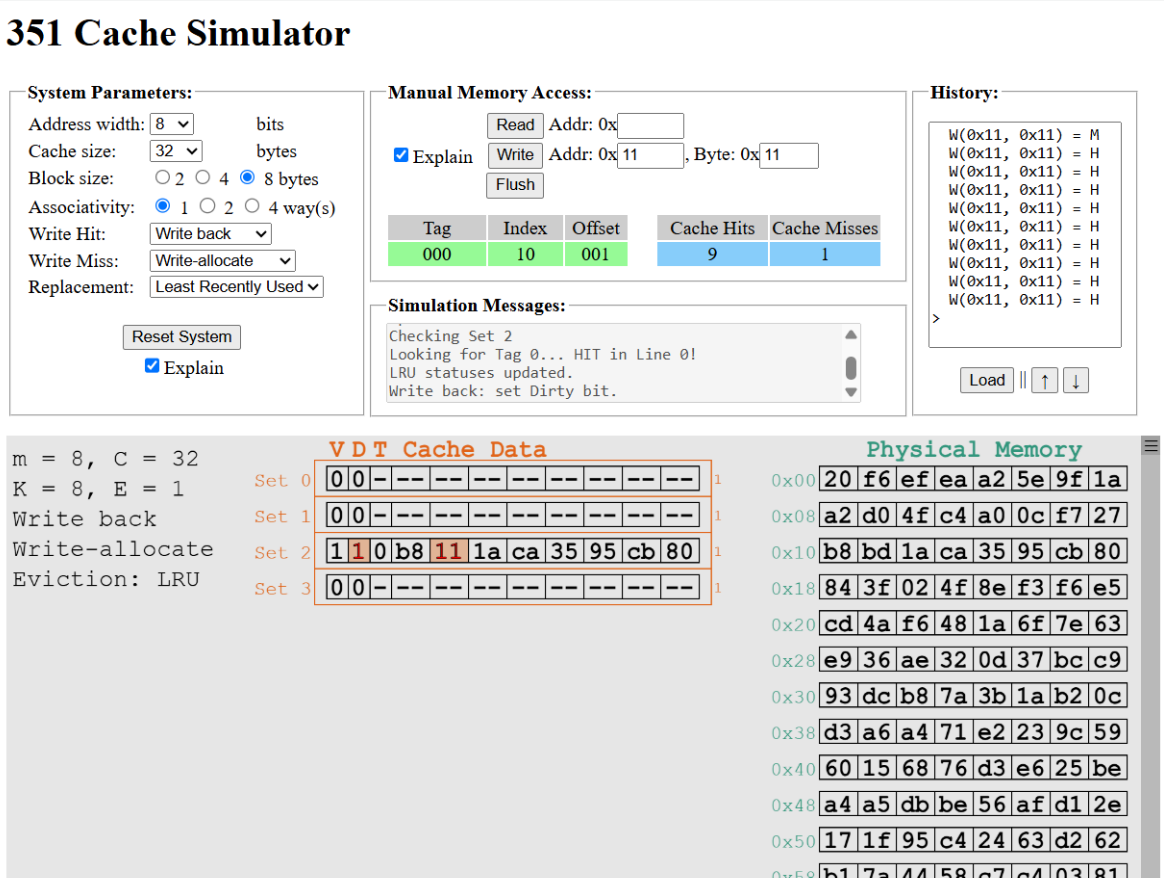
```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

```
       V D T Cache Data
Set 0  0 0 - -- -- -- -- -- -- -- --  1
Set 1  0 0 - -- -- -- -- -- -- -- --  1
Set 2  1 0 0 b8 bd 1a ca 35 95 cb 80  1
Set 3  0 0 - -- -- -- -- -- -- -- --  1
```

```
     Physical Memory
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
```

- Address breakdown: 000 10 101 (tag=000, index=10, offset=101)
- First access (R(0x15)):
  + Cache miss (M in history)
  + Loads block (0x10-0x17) into Set 2
- Total hits: 8, misses: 1

5. Execute a Write memory request to address 0x11:



351 Cache Simulator

System Parameters:

Address width:	8	bits
Cache size:	32	bytes
Block size:	○ 2 ○ 4 ● 8	bytes
Associativity:	● 1 ○ 2 ○ 4	way(s)
Write Hit:	Write back	
Write Miss:	Write-allocate	
Replacement:	Least Recently Used	

☑ Explain

Manual Memory Access:

Addr: 0x

☑ Explain Addr: 0x 11 , Byte: 0x 11

Tag	Index	Offset	Cache Hits	Cache Misses
000	10	001	9	1

Simulation Messages:

```
Checking Set 2
Looking for Tag 0... HIT in Line 0!
LRU statuses updated.
Write back: set Dirty bit.
```

History:

```
W(0x11, 0x11) = M
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
>
```

```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

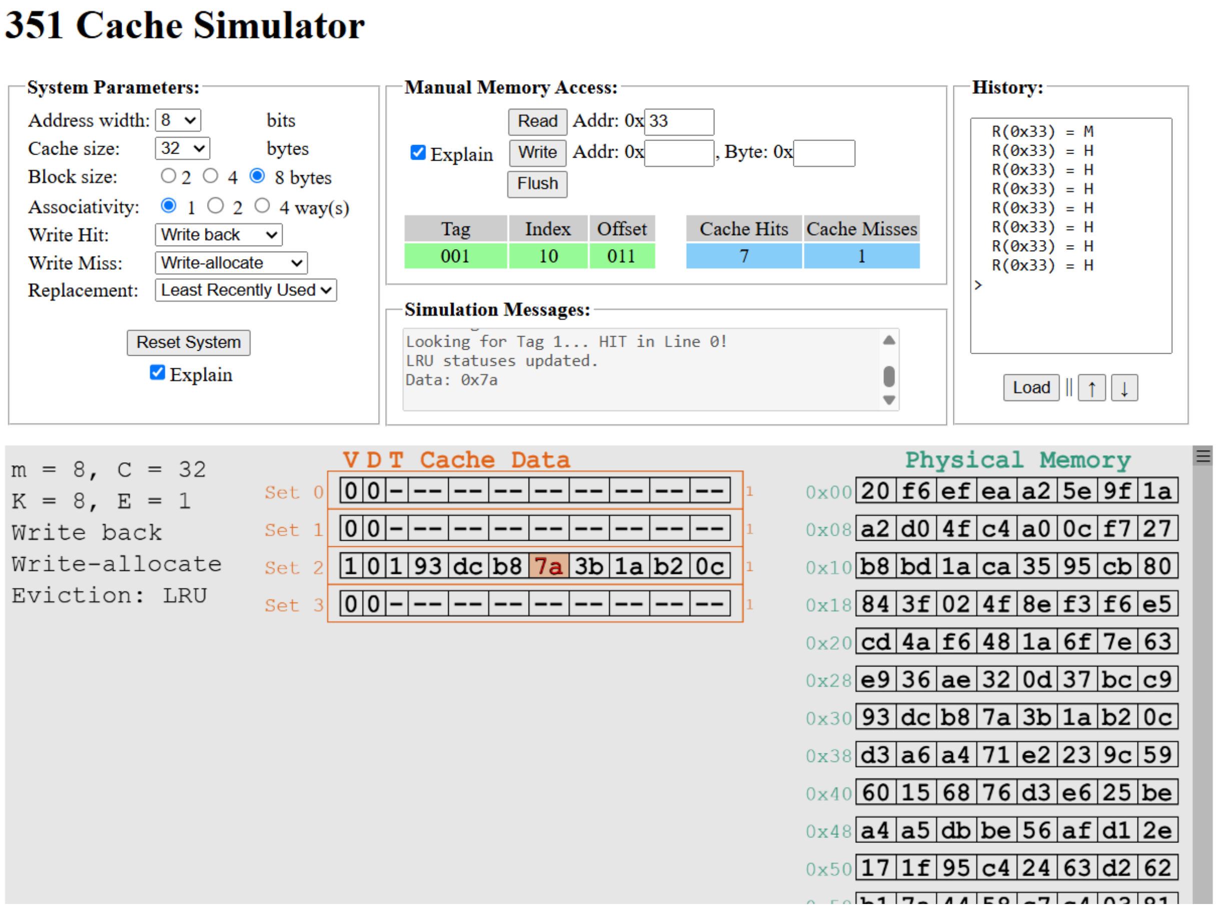
	V	D	T									
Set 0	0	0	–	--	--	--	--	--	--	--	--	1
Set 1	0	0	–	--	--	--	--	--	--	--	--	1
Set 2	1	1	0	b8	11	1a	ca	35	95	cb	80	1
Set 3	0	0	–	--	--	--	--	--	--	--	--	1

Physical Memory

0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62
0x58	b1	7a	44	58	c7	c4	03	81

- Address breakdown: 000 10 001 (tag=000, index=10, offset=001)
- Behavior:
  - Write-allocate policy means it loads block first
  - Write-back policy sets dirty bit
  - Cache hit (block already in Set 2 from previous access)
  - Updates data at offset 001 within the block
  - Dirty bit set, will write back to memory when evicted

6. **Execute a Read memory request to address 0x33:**



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x 33
☑ Explain Addr: 0x , Byte: 0x

Tag	Index	Offset
001	10	011

Cache Hits	Cache Misses
7	1

Simulation Messages:

```
Looking for Tag 1... HIT in Line 0!
LRU statuses updated.
Data: 0x7a
```

History:

```
R(0x33) = M
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
>
```

```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

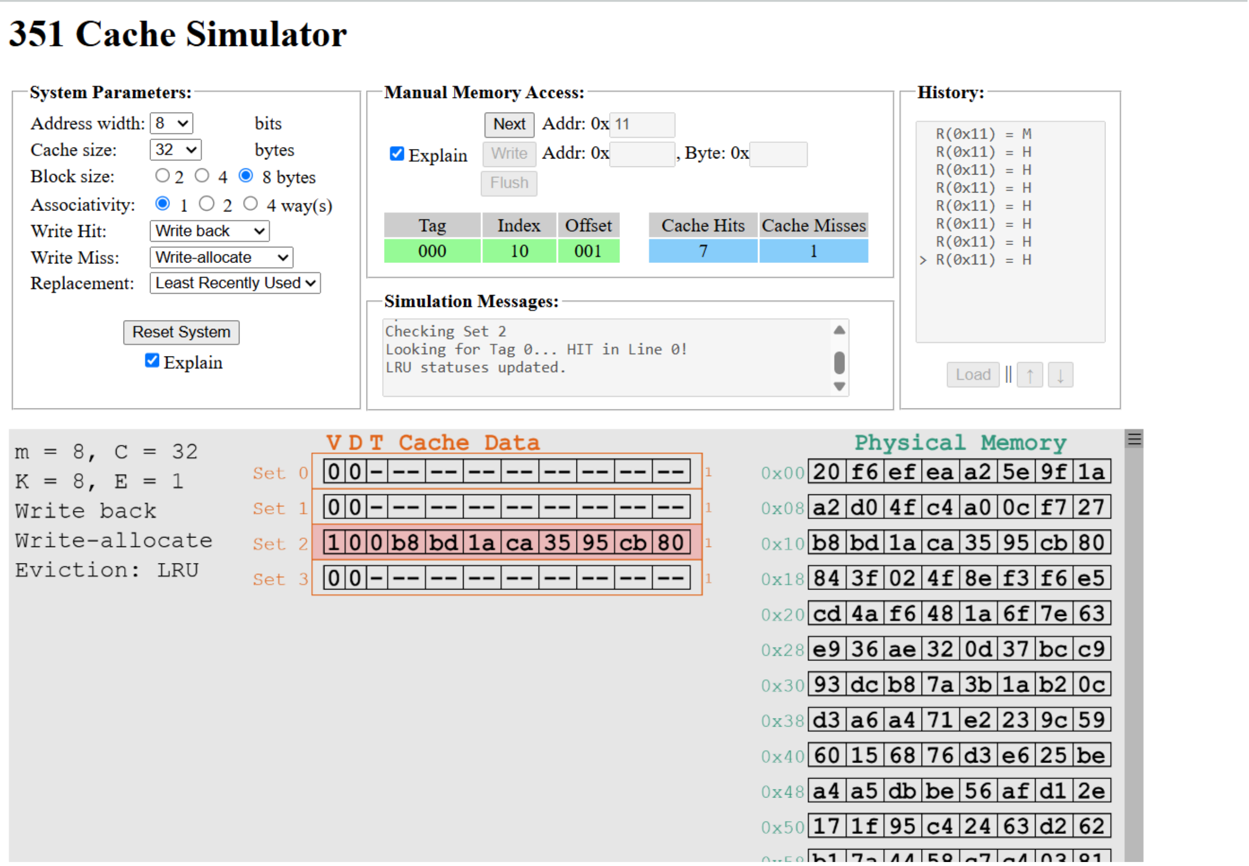
	V	D	T									
Set 0	0	0	-	--	--	--	--	--	--	--	--	1
Set 1	0	0	-	--	--	--	--	--	--	--	--	1
Set 2	1	0	1	93	dc	b8	7a	3b	1a	b2	0c	1
Set 3	0	0	-	--	--	--	--	--	--	--	--	1

Physical Memory

0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62

- Address breakdown: 001 10 011 (tag=001, index=10, offset=011)
- First access (R(0x33)):
  - Loads block (0x30-0x37) into Set 2
  - Evicts previous block from Set 2 (if dirty, would write back)
  - Data at 0x33 is 0x7a
- Cache contents shown:
  - Set 2: Valid=1, Dirty=0, Tag=1

7. **Execute a Read Memory request to address 0x11**



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x 11
☑ Explain Addr: 0x , Byte: 0x

Tag	Index	Offset	Cache Hits	Cache Misses
000	10	001	7	1

History:

```
  R(0x11) = M
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
> R(0x11) = H
```

Simulation Messages:

```
Checking Set 2
Looking for Tag 0... HIT in Line 0!
LRU statuses updated.
```

```
m = 8, C = 32
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

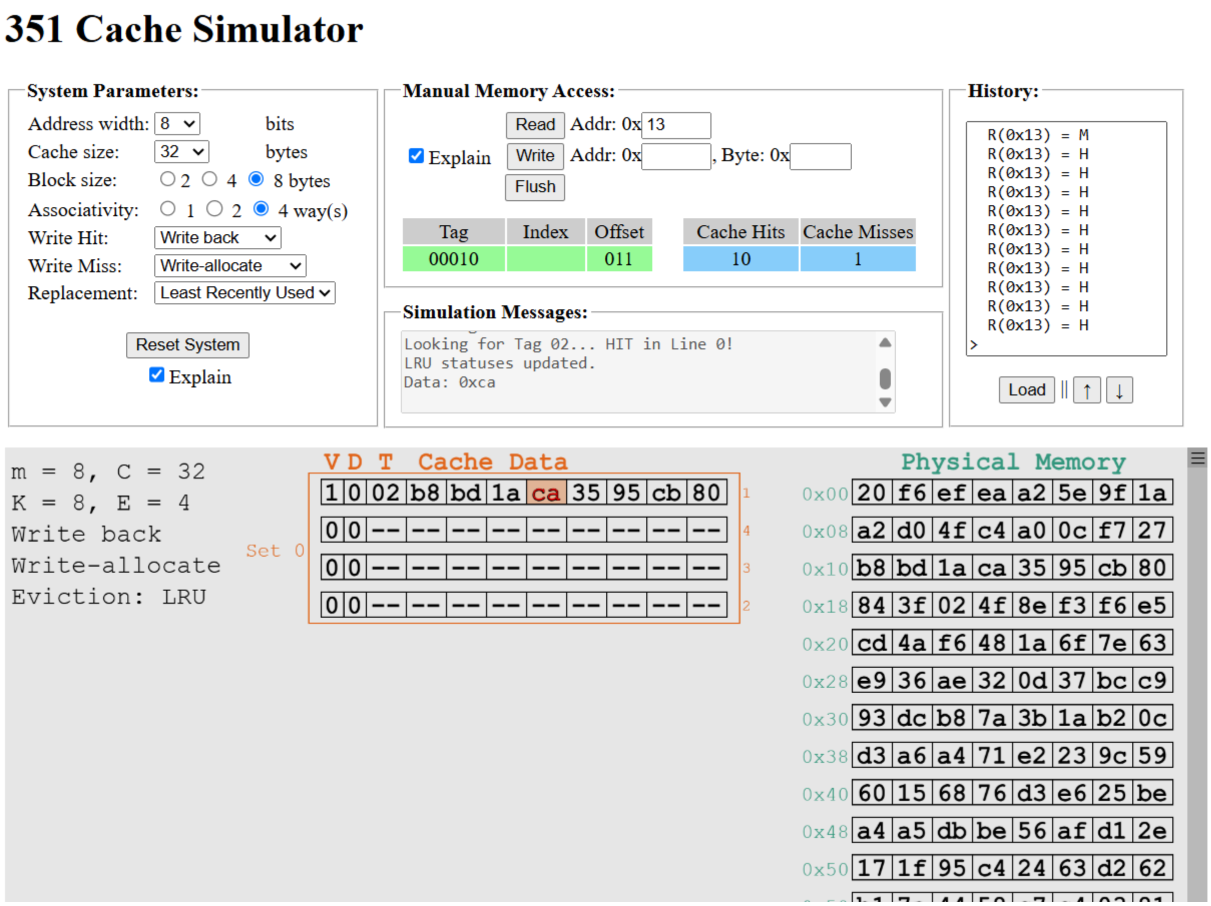
```
        V D T  Cache Data
Set 0 |0|0|-|--|--|--|--|--|--|--|--| 1
Set 1 |0|0|-|--|--|--|--|--|--|--|--| 1
Set 2 |1|0|0|b8|bd|1a|ca|35|95|cb|80| 1
Set 3 |0|0|-|--|--|--|--|--|--|--|--| 1
```

```
          Physical Memory
0x00 |20|f6|ef|ea|a2|5e|9f|1a|
0x08 |a2|d0|4f|c4|a0|0c|f7|27|
0x10 |b8|bd|1a|ca|35|95|cb|80|
0x18 |84|3f|02|4f|8e|f3|f6|e5|
0x20 |cd|4a|f6|48|1a|6f|7e|63|
0x28 |e9|36|ae|32|0d|37|bc|c9|
0x30 |93|dc|b8|7a|3b|1a|b2|0c|
0x38 |d3|a6|a4|71|e2|23|9c|59|
0x40 |60|15|68|76|d3|e6|25|be|
0x48 |a4|a5|db|be|56|af|d1|2e|
0x50 |17|1f|95|c4|24|63|d2|62|
0x58 |b1|7a|44|58|c7|c4|03|81|
```

- Access: Read from address 0x11
- Address breakdown: 000 10 001 (tag=000, index=10, offset=001)
- Behavior:
  - Cache hit (block now in Set 2)
  - Total hits: 7, misses: 1
  - Shows LRU updating even on hits

8. Modification of cache configuration:

– Change Associativity: 4 ways



351 Cache Simulator

System Parameters:

- Address width: 8 bits
- Cache size: 32 bytes
- Block size: ○ 2 ○ 4 ● 8 bytes
- Associativity: ○ 1 ○ 2 ● 4 way(s)
- Write Hit: Write back
- Write Miss: Write-allocate
- Replacement: Least Recently Used

Reset System

☑ Explain

Manual Memory Access:

Read Addr: 0x 13

☑ Explain Write Addr: 0x , Byte: 0x

Flush

Tag	Index	Offset	Cache Hits	Cache Misses
00010		011	10	1

Simulation Messages:

```
Looking for Tag 02... HIT in Line 0!
LRU statuses updated.
Data: 0xca
```

History:

```
R(0x13) = M
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
>
```

Load || ↑ ↓

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

Set 0

V	D	T	Cache Data								LRU
1	0	02	b8	bd	1a	ca	35	95	cb	80	1
0	0	--	--	--	--	--	--	--	--	--	4
0	0	--	--	--	--	--	--	--	--	--	3
0	0	--	--	--	--	--	--	--	--	--	2

Physical Memory

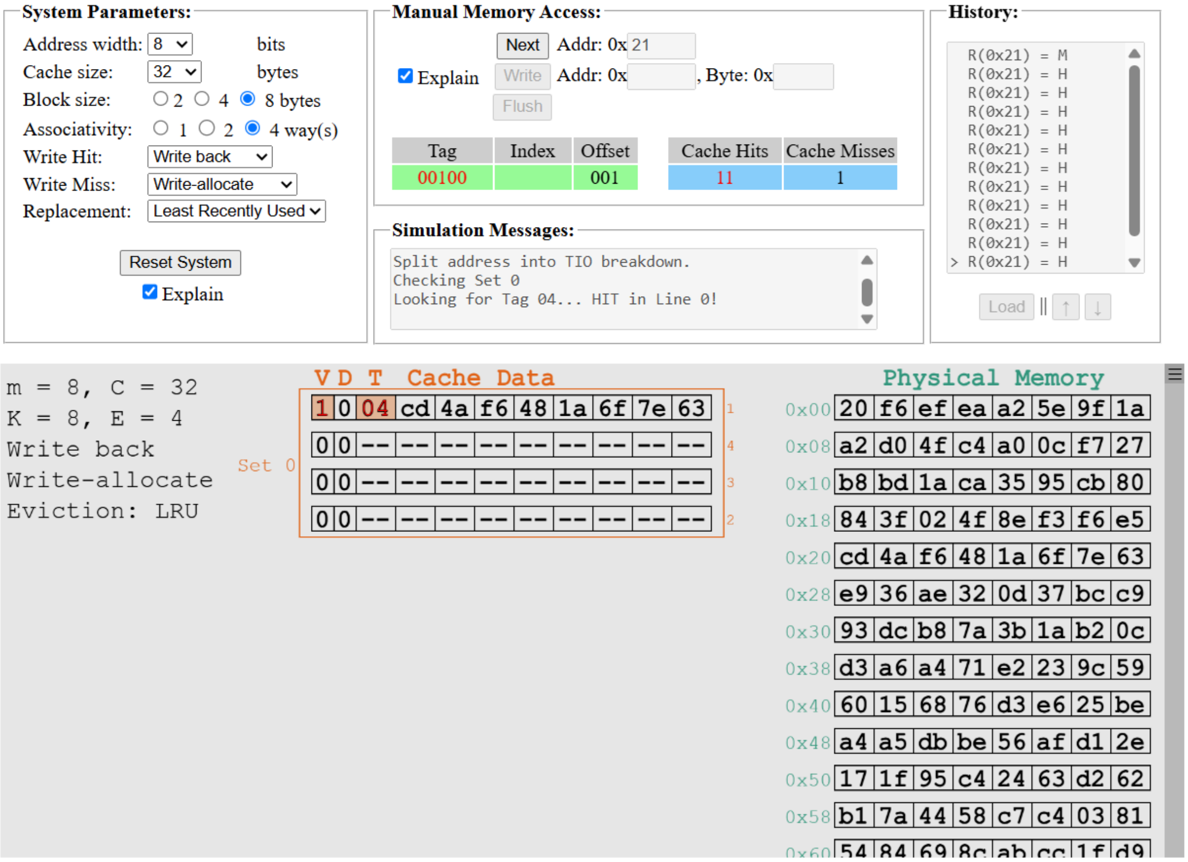
```
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
```

- Cache Hits and Misses: 10 accesses (1 miss + 9 hits)
- Hit rate: Both show similar hit rates (~90%) for this access pattern
- Block Loading Analysis
  - Current 8B-block/4-way configuration
- First access (R(0x13)=M):
  - Address 0x13 breakdown: 00010 011 (Tag=00010, Offset=011)
  - Entire cache is one set (fully associative)
  - Miss loads the 8-byte block from memory (0x10-0x17: b8 bd 1a ca 35 95 cb 80)
  - Stored in one of the 4 available cache lines
- Subsequent accesses: All hits because the block remains in cache
  - LRU status updates on each access

- Compared to original 4B-block/2-way:
  - Spatial locality less effective with smaller block
  - Current configuration benefits from larger blocks capturing more nearby data

- Write Policy Analysis: Both configurations use:

    + Write back: On writes, only cache is updated (sets dirty bit)
    + Write allocate: On write misses, block is loaded into cache first
    + The policies work identically in both configurations, just with different block sizes.
- Advantages of Current Configuration
    + Better spatial locality: 8-byte blocks capture more data per miss
    + Higher associativity (4-way): Reduces conflict misses (though in this case, it's effectively fully associative)
    + Simpler management: Only one set to track in this small cache
- Disadvantages:
    + Higher miss penalty: Loading 8 bytes takes longer than 4 bytes
    + Potential waste: If programs don't use spatial locality well, larger blocks may bring in unused data



System Parameters:

Address width:	8	bits
Cache size:	32	bytes
Block size:	○ 2 ○ 4 ● 8 bytes	
Associativity:	○ 1 ○ 2 ● 4 way(s)	
Write Hit:	Write back	
Write Miss:	Write-allocate	
Replacement:	Least Recently Used	

☑ Explain

Manual Memory Access:

Addr: 0x 21

☑ Explain

Addr: 0x , Byte: 0x

Tag	Index	Offset	Cache Hits	Cache Misses
00100		001	11	1

Simulation Messages:

```
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 04... HIT in Line 0!
```

History:

```
  R(0x21) = M
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
  R(0x21) = H
> R(0x21) = H
```

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

```
        V D T  Cache Data
        1 0 04 cd 4a f6 48 1a 6f 7e 63  1
        0 0 -- -- -- -- -- -- -- -- --  4
Set 0   0 0 -- -- -- -- -- -- -- -- --  3
        0 0 -- -- -- -- -- -- -- -- --  2
```

```
Physical Memory
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
0x60 54 84 69 8c ab cc 1f d9
```

Performance Overview

- Cache Hits: 11
- Misses: 1

➔ Hit Rate = 11 / (11 + 1) = 91.67% (efficient)

Address width (m): 8 bits → address range: 0x00 to 0xFF

- Cache size (C): 32 bytes
- Block size (B): 8 bytes
- Associativity (E): 4-way set associative (4 lines per set)
- Number of sets (S): C / (B × E) = 32 / (8 × 4) = 1 set (Set 0)
- Write policy:
  - Write Hit: Write-back
  - Write Miss: Write-allocate
- Replacement policy: Least Recently Used (LRU)
- Block from **address 0x20** maps to **set 0**, **tag = 0x04**:
- TIO Breakdown of 0x21 (0010 0001):
  - Tag = 0x04
  - Index = 0 (only 1 set)
  - Offset = 1 → 2nd byte in block

➔ Thus, block [0x20 – 0x27] is currently cached, which is why repeated accesses to 0x21 result in cache hits.

Write Policy Analysis

- Write-back:

+ On a write hit, data is only written to the cache, and the line is marked dirty.

+ On eviction, if a line is dirty, it must be written back to memory.

Write-allocate:

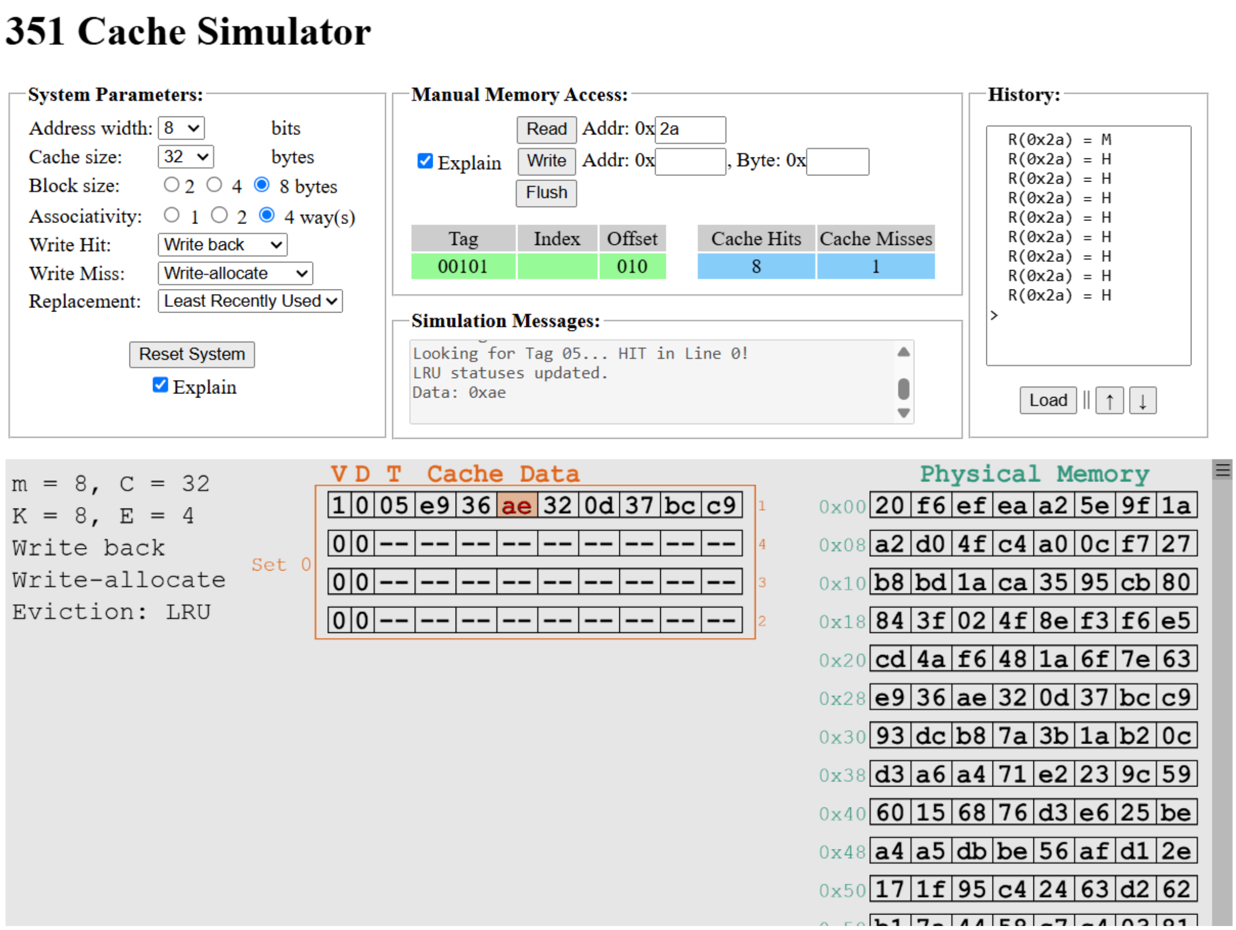
+ On a **write miss**, the block is loaded into the cache first (allocated), then written to.

- Replacement Policy: LRU

+ LRU ensures that the **least recently used** block is evicted when all lines in the set are occupied.

+ Since this is a 4-way associative cache and only one block is loaded so far, no eviction has happened yet.

Metric	**Current Config (4-way)**
Cache Hits	11
Cache Misses	1
Hit Rate	91.67%
Associativity	4-way (Set 0 only)
Write Policy	Write-back + Write-allocate
Replacement Policy	LRU
Advantage vs Direct-Mapped	Better hit rate, fewer evictions



351 Cache Simulator

System Parameters:
Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ○ 2 ● 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x 2a
☑ Explain

Addr: 0x , Byte: 0x

Tag	Index	Offset	Cache Hits	Cache Misses
00101		010	8	1

History:

```
R(0x2a) = M
R(0x2a) = H
R(0x2a) = H
R(0x2a) = H
R(0x2a) = H
R(0x2a) = H
R(0x2a) = H
R(0x2a) = H
R(0x2a) = H
>
```

Simulation Messages:

```
Looking for Tag 05... HIT in Line 0!
LRU statuses updated.
Data: 0xae
```

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

```
       V D T  Cache Data
Set 0  1|0|05|e9|36|ae|32|0d|37|bc|c9  1
       0|0|--|--|--|--|--|--|--|--|--  4
       0|0|--|--|--|--|--|--|--|--|--  3
       0|0|--|--|--|--|--|--|--|--|--  2
```

```
     Physical Memory
0x00 20|f6|ef|ea|a2|5e|9f|1a
0x08 a2|d0|4f|c4|a0|0c|f7|27
0x10 b8|bd|1a|ca|35|95|cb|80
0x18 84|3f|02|4f|8e|f3|f6|e5
0x20 cd|4a|f6|48|1a|6f|7e|63
0x28 e9|36|ae|32|0d|37|bc|c9
0x30 93|dc|b8|7a|3b|1a|b2|0c
0x38 d3|a6|a4|71|e2|23|9c|59
0x40 60|15|68|76|d3|e6|25|be
0x48 a4|a5|db|be|56|af|d1|2e
0x50 17|1f|95|c4|24|63|d2|62
```

Address Breakdown: 0x2A

- 0x2A = binary: 00101010
- Cache uses:
  - Offset: 3 bits → bits [2:0] → 010 → offset = 2 (3rd byte of block)
  - Index: Since there's only 1 set → 0 bits used → always set 0
  - Tag: Remaining upper bits → bits [7:3] → 00101 → tag = 05

Cache Access Process

- Cache checks Set 0

- Looks for Tag = 05
- Set 0 initially empty (from fresh system or flushed state)
- MISS occurs because no matching tag

🔹 What Happens on Miss

- Write-allocate & write-back apply only on writes, but this is a read, so:
- Cache loads block from main memory starting at aligned address:
    - Aligned block address of 0x2A = 0x28 (block base aligned to 8 bytes)

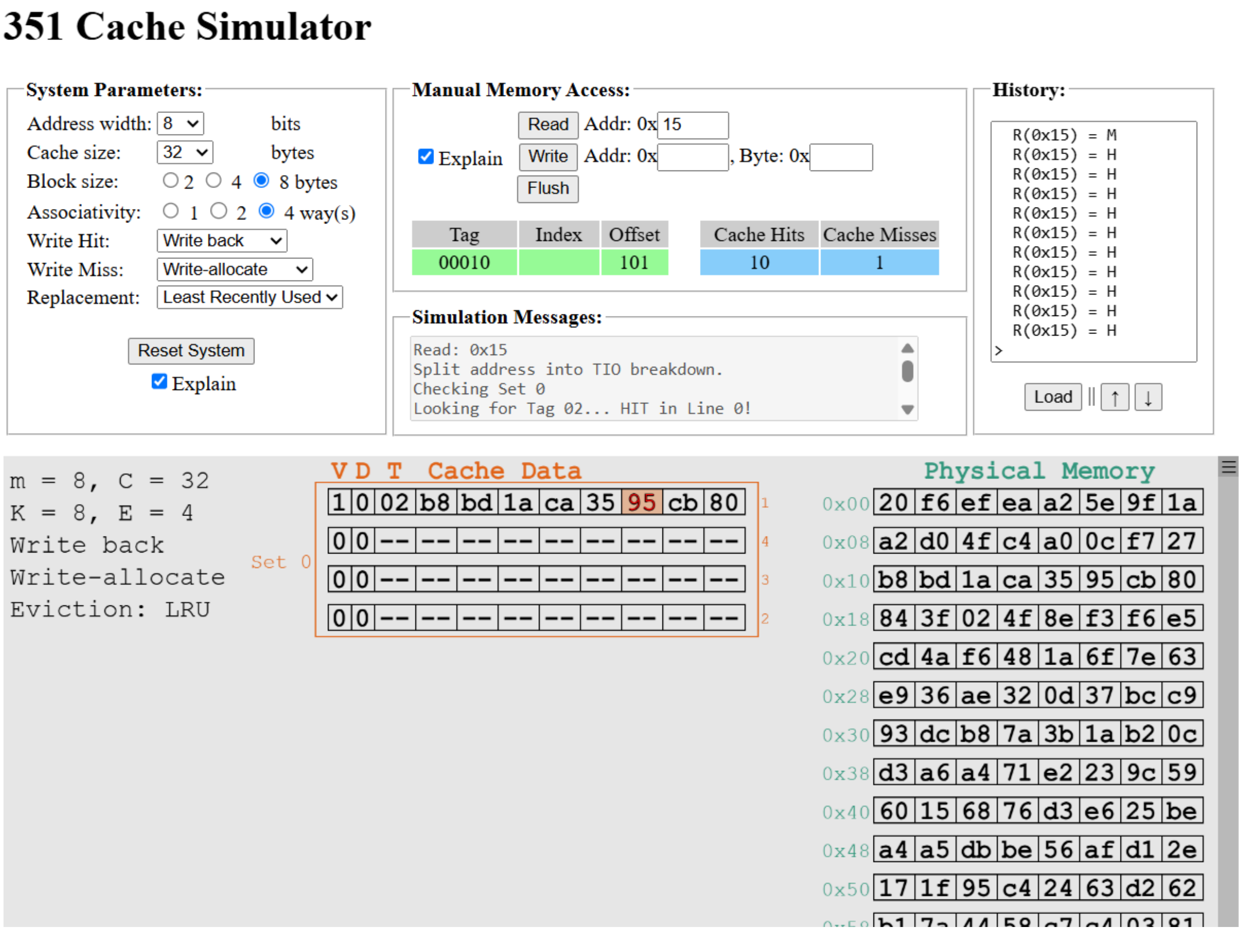
Cache stores it in line 0 of Set 0 (others still empty)

Cache line metadata:

- Valid bit = 1
- Dirty bit = 0 (nothing modified yet)
- Tag = 05

Detail	Value
Block loaded	From address 0x28
Data loaded	e9 36 ae 32 0d 37 bc c9
Stored in	Cache set 0, line 0
Initial access	MISS (data not cached yet)
Later accesses	All HITS
Tag stored	05
Byte returned	ae (at offset 2)

Cache Hits	8
Cache Misses	1



351 Cache Simulator

System Parameters:

- Address width: 8 bits
- Cache size: 32 bytes
- Block size: ○ 2 ○ 4 ● 8 bytes
- Associativity: ○ 1 ○ 2 ● 4 way(s)
- Write Hit: Write back
- Write Miss: Write-allocate
- Replacement: Least Recently Used

Reset System

☑ Explain

Manual Memory Access:

Read Addr: 0x 15

☑ Explain Write Addr: 0x , Byte: 0x

Flush

Tag	Index	Offset
00010		101

Cache Hits	Cache Misses
10	1

Simulation Messages:

```
Read: 0x15
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 02... HIT in Line 0!
```

History:

```
R(0x15) = M
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
R(0x15) = H
>
```

Load || ↑ ↓

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

```
       V D T  Cache Data
       1 0 02 b8 bd 1a ca 35 95 cb 80   1
Set 0  0 0 -- -- -- -- -- -- -- -- --   4
       0 0 -- -- -- -- -- -- -- -- --   3
       0 0 -- -- -- -- -- -- -- -- --   2
```

```
Physical Memory
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
```

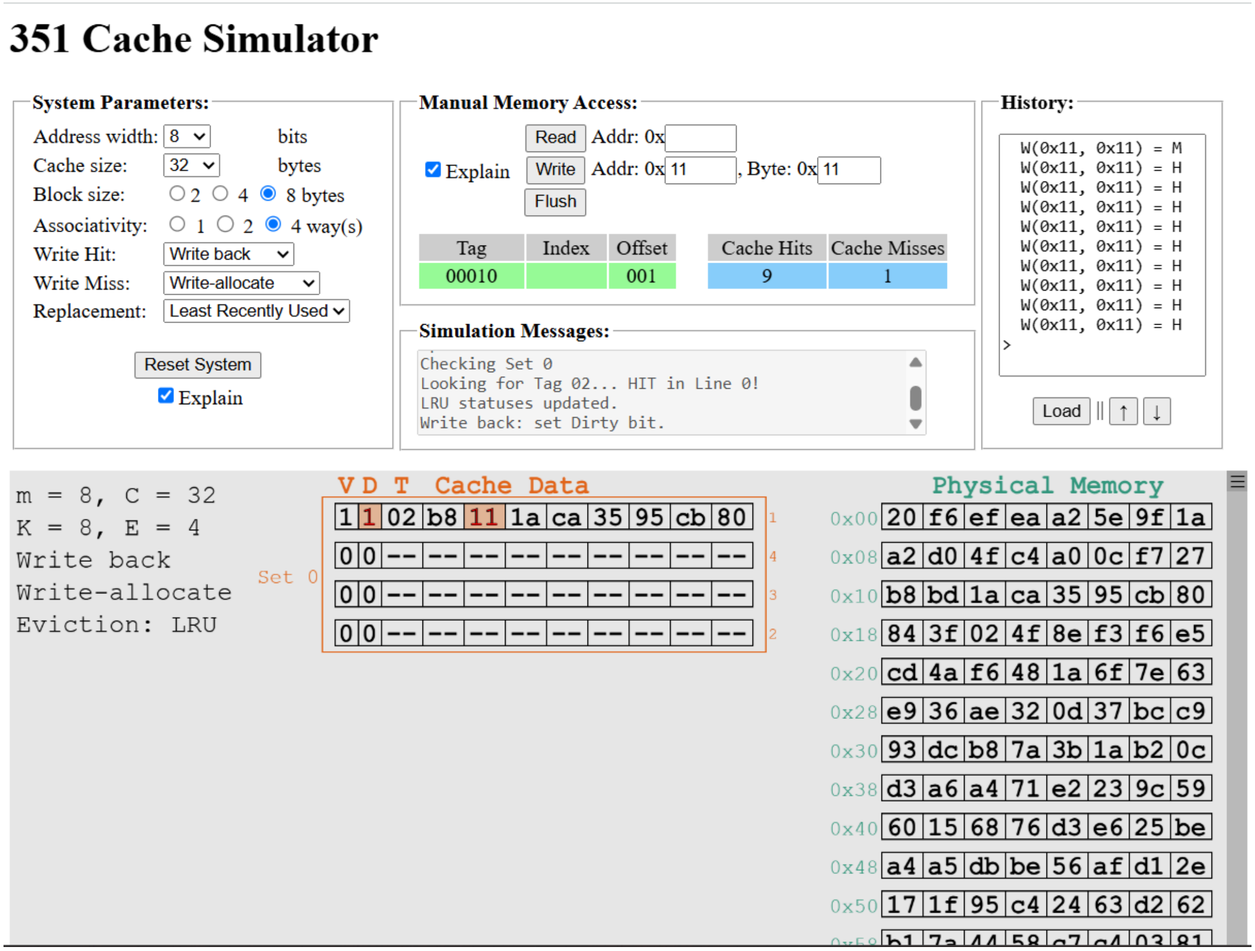
Address Breakdown: 0x15

- 0x15 = binary: 00010101
- Breakdown:
    - Offset = 101 → offset = 5 (6th byte in block)
    - Index = always 0 (1 set)
    - Tag = 00010 → tag = 02

Cache Access Process:

- Cache checks Set 0
- Looks for Tag = 02
- Set 0 contains only 1 block (from earlier access)
- Tag does not match → MISS

Detail	Value
Block loaded	From address 0x10
Data loaded	b8 bd 1a ca 35 95 cb 80
Stored in	Cache set 0, line 0
Initial access	MISS
Later accesses	All HITS
Tag stored	02
Byte returned	95 (at offset 5)
Cache Hits	10
Cache Misses	1



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ○ 2 ● 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x

☑ Explain

Addr: 0x 11, Byte: 0x 11

Tag	Index	Offset	Cache Hits	Cache Misses
00010		001	9	1

Simulation Messages:

```
Checking Set 0
Looking for Tag 02... HIT in Line 0!
LRU statuses updated.
Write back: set Dirty bit.
```

History:

```
W(0x11, 0x11) = M
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
W(0x11, 0x11) = H
>
```

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

```
       V D T  Cache Data
       1 1 02 b8 11 1a ca 35 95 cb 80  1
Set 0  0 0 -- -- -- -- -- -- -- -- --  4
       0 0 -- -- -- -- -- -- -- -- --  3
       0 0 -- -- -- -- -- -- -- -- --  2
```

```
        Physical Memory
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
```

With 9 cache hits and 1 cache miss, the current configuration shows a very high cache hit rate:

- Total accesses: 9 (hits)+1 (misses)=10
- Hit Rate: (9/10)∗100%=90%
- Miss Rate: (1/10)∗100%=10%

This indicates good performance for the sequence of memory accesses that were simulated.

Block Placement Analysis

From the cache data visualization:

- **Set 0** contains valid data (highlighted blocks show 11, 02, b8, 11, 1a, ca, 35, 95, cb, 80)
- The cache is organized as 4 sets (since 32 bytes ÷ 8 bytes per block = 4 blocks total, with 4-way associativity = 1 set)
- Only one cache line is currently populated, indicating most accesses hit the same set

Write Policy Analysis

Write-Back + Write-Allocate Policy:

- Write-Back: Modified data stays in cache until eviction (reduces memory traffic)
- Write-Allocate: On write miss, block is loaded into cache first
- The "Write back: set Dirty bit" message in simulation indicates this policy is active
- LRU replacement ensures least recently used blocks are evicted first

Performance Comparison Considerations

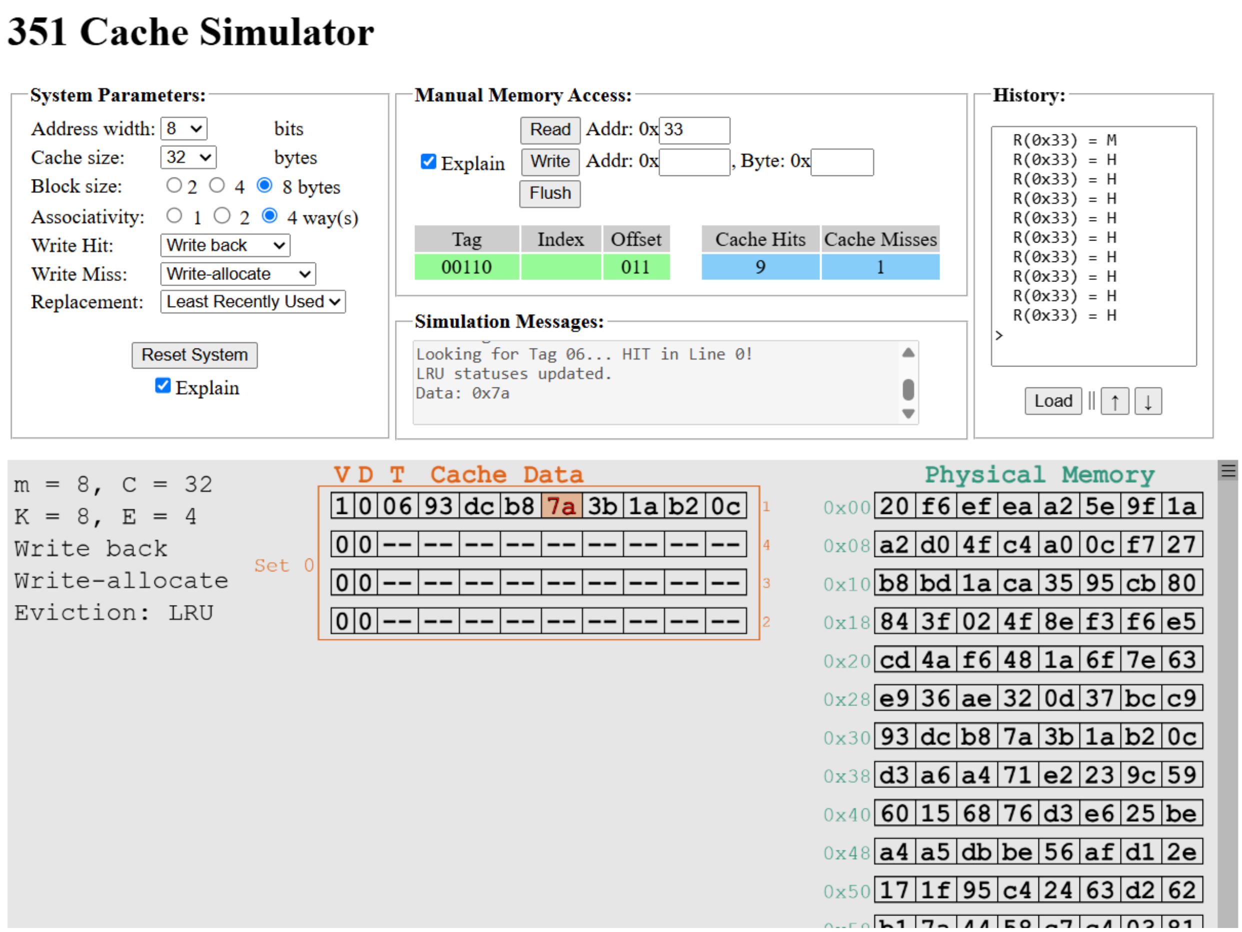
Without seeing the "first configuration" you're comparing to, this setup shows:

Strengths:

- High hit rate (90%) suggests good temporal locality
- 4-way associativity reduces conflict misses
- Write-back policy minimizes memory writes

Potential Issues:

- Small cache size (32 bytes) limits capacity
- Single populated set suggests potential mapping conflicts



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ○ 2 ● 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

Reset System
☑ Explain

Manual Memory Access:

Read Addr: 0x 33
☑ Explain Write Addr: 0x , Byte: 0x
Flush

Tag	Index	Offset
00110		011

Cache Hits	Cache Misses
9	1

Simulation Messages:

```
Looking for Tag 06... HIT in Line 0!
LRU statuses updated.
Data: 0x7a
```

History:

```
R(0x33) = M
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
R(0x33) = H
>
```

Load || ↑ ↓

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

	V	D	T	Cache Data								
Set 0	1	0	06	93	dc	b8	7a	3b	1a	b2	0c	1
	0	0	--	--	--	--	--	--	--	--	--	4
	0	0	--	--	--	--	--	--	--	--	--	3
	0	0	--	--	--	--	--	--	--	--	--	2

	Physical Memory							
0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62

Detailed Cache Configuration Analysis

System Parameters

- Address Width: 8 bits (256 possible addresses: 0x00-0xFF)
- Cache Size: 32 bytes total capacity
- Block Size: 8 bytes per cache line
- Associativity: 4-way set associative
- Total Blocks: 4 blocks (32 bytes ÷ 8 bytes per block)
- Number of Sets: 1 set (4 blocks ÷ 4 ways = 1 set)

Memory Access Breakdown

Current Access: Address 0x33

- Binary: 00110011
- Tag: 00110 (5 bits)
- Index: 0 (0 bits - since only 1 set)

- Offset: 011 (3 bits for 8-byte blocks)

Performance Metrics

- Cache Hits: 9
- Cache Misses: 1
- Hit Rate: 90% (9/10 total accesses)
- Miss Rate: 10%

Cache State Analysis

Cache Data Structure:

- Set 0, Way 0: Valid bit=1, Tag=0, Data=10 06 93 dc b8 7a 3b 1a
- Ways 1-3: Currently empty (Valid bit=0, shown as dashes)
- Highlighted block: Contains the data for the current access

Current Operation:

- Simulation message: "Looking for Tag 06... HIT in Line 0!"
- LRU status updated after the hit
- Data retrieved: 0x7a (shown in simulation)

Write Policy Configuration

- Write Hit Policy: Write-back
  - Modified data stays in cache until eviction
  - Reduces memory bus traffic
  - Uses dirty bit to track modifications
- Write Miss Policy: Write-allocate
  - On write miss, block is first loaded into cache
  - Then the write operation proceeds
- Replacement Policy: Least Recently Used (LRU)
  - Tracks access order for the 4 ways
  - Evicts least recently accessed block when needed

Access History Pattern

Recent accesses to address 0x33:

- First access: Miss (M) - block loaded into cache
- Subsequent 8 accesses: All hits (H)
- Demonstrates excellent temporal locality

Physical Memory Context

The physical memory shows the complete memory space with:

- Data at address 0x30: 93 dc b8 7a 3b 1a b2 0c
- This corresponds to the cached block data
- Address 0x33 contains value 7a (highlighted in red)

Cache Efficiency Analysis

Strengths:

- High hit rate (90%) indicates good locality of reference
- 4-way associativity reduces conflict misses effectively
- Write-back policy minimizes memory writes
- Single set means no index conflicts

Limitations:

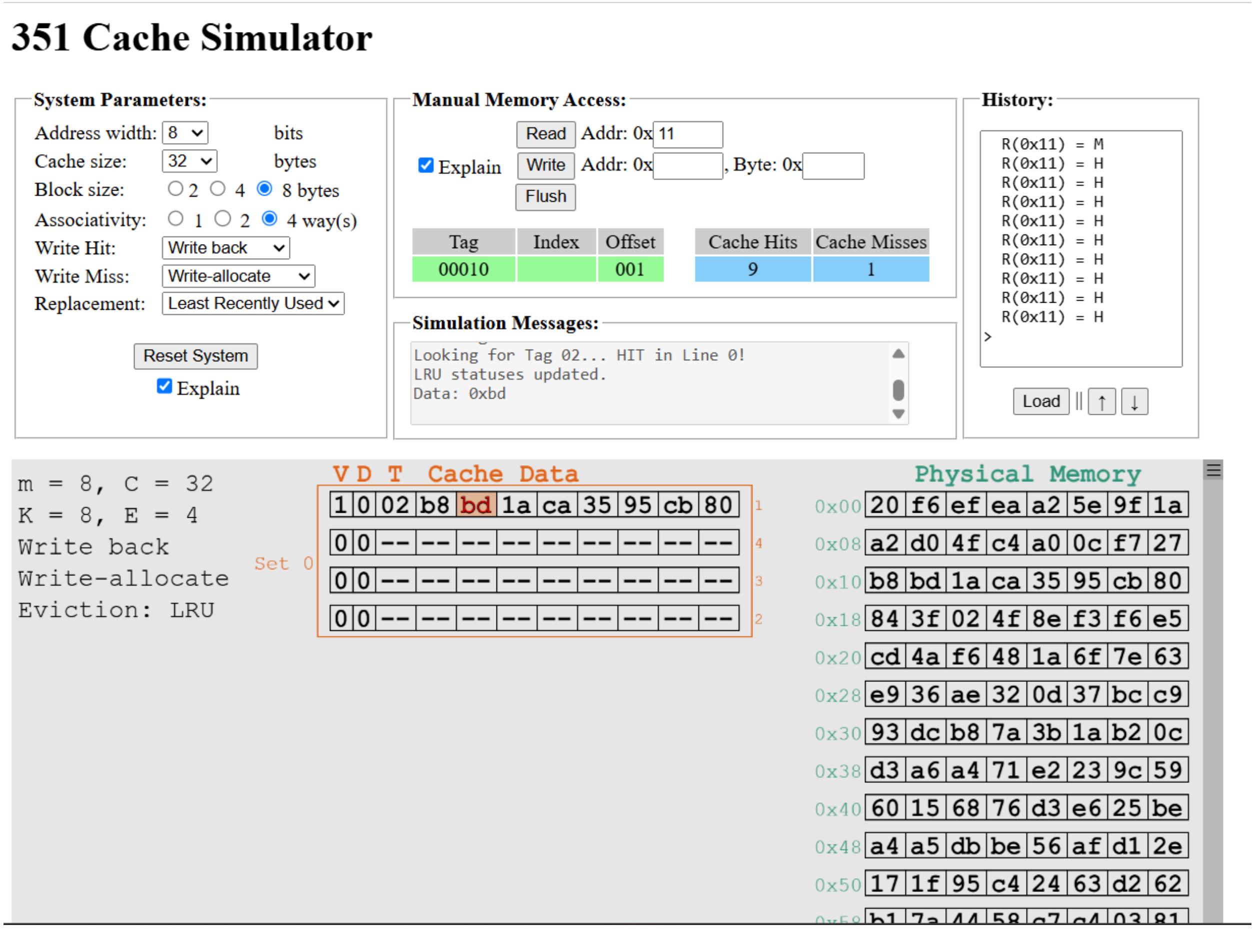
- Small 32-byte capacity limits the working set size
- All blocks compete for the same set (no spatial distribution)
- High associativity overhead for such a small cache

Address Mapping Strategy

With 8-bit addresses and 1 set:

- Tag bits: 5 bits (identifies which block)
- Index bits: 0 bits (only one set)
- Offset bits: 3 bits (8-byte block addressing)
- Every address maps to the same set, making associativity crucial

This configuration demonstrates a highly associative, small capacity cache optimized for temporal locality in constrained memory environments.



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ○ 2 ● 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

☑ Explain

Addr: 0x 11
Addr: 0x , Byte: 0x

Tag	Index	Offset
00010		001

Cache Hits	Cache Misses
9	1

History:

```
R(0x11) = M
R(0x11) = H
R(0x11) = H
R(0x11) = H
R(0x11) = H
R(0x11) = H
R(0x11) = H
R(0x11) = H
R(0x11) = H
R(0x11) = H
>
```

Simulation Messages:

```
Looking for Tag 02... HIT in Line 0!
LRU statuses updated.
Data: 0xbd
```

```
m = 8, C = 32
K = 8, E = 4
Write back
Write-allocate
Eviction: LRU
```

Set 0

V	D	T	Cache Data								
1	0	02	b8	bd	1a	ca	35	95	cb	80	1
0	0	--	--	--	--	--	--	--	--	--	4
0	0	--	--	--	--	--	--	--	--	--	3
0	0	--	--	--	--	--	--	--	--	--	2

Physical Memory

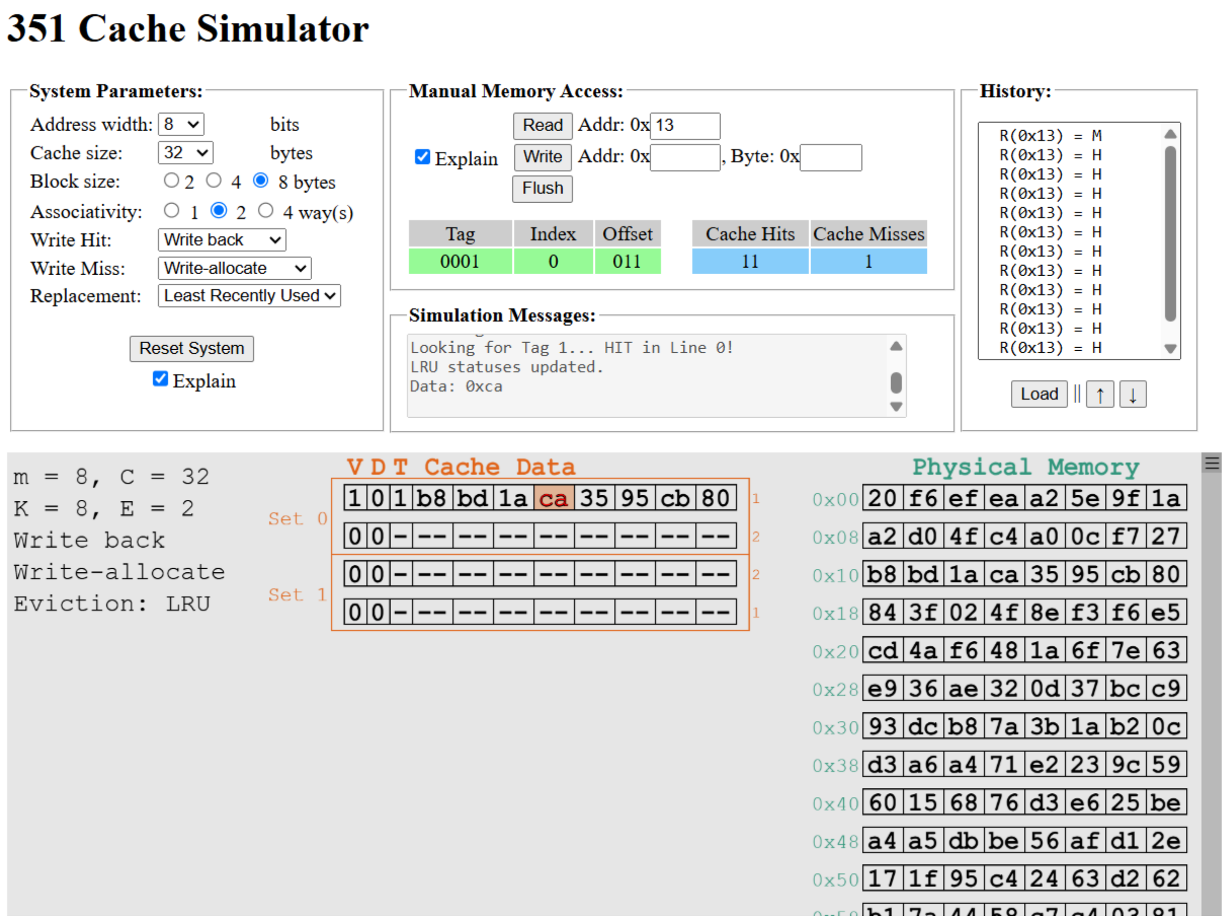
0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62
0x58	b1	7a	44	58	c7	c4	03	81

- Address width (m) = 8 bits ⇒ Address range: 0x00 to 0xFF
- Cache size (C) = 32 bytes
- Block size (B) = 8 bytes
- Associativity (E) = 4-way set associative
- Number of sets (S) = C / (B × E) = 32 / (8 × 4) = 1 set
- Write Hit policy = Write-back
- Write Miss policy = Write-allocate
- Eviction Policy = LRU

So we have only 1 set (Set 0) and 4 cache lines, meaning all data maps to a single set with LRU for eviction.

9. Modification of cache configuration:

– Change Associativity: 2 ways



351 Cache Simulator

System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ● 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x 13
☑ Explain
Addr: 0x , Byte: 0x

Tag	Index	Offset	Cache Hits	Cache Misses
0001	0	011	11	1

Simulation Messages:

```
Looking for Tag 1... HIT in Line 0!
LRU statuses updated.
Data: 0xca
```

History:

```
R(0x13) = M
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
R(0x13) = H
```

```
m = 8, C = 32
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

```
        V D T Cache Data
Set 0   1 0 1 b8 bd 1a ca 35 95 cb 80   1
        0 0 - -- -- -- -- -- -- -- --   2
Set 1   0 0 - -- -- -- -- -- -- -- --   2
        0 0 - -- -- -- -- -- -- -- --   1
```

```
Physical Memory
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
```

Parameter	Value
Address width (m)	8 bits (0x00–0xFF)
Cache size (C)	32 bytes
Block size (B)	8 bytes
Associativity (E)	2-way set associative

Number of sets (S)	2
Write hit policy	Write-back
Write miss policy	Write-allocate
Replacement policy	LRU

So, for any 8-bit address:

- Bits [7:4] = Tag (4 bits)
- Bit [3] = Index (1 bit)
- Bits [2:0] = Offset (3 bits)

From the interface:

- Address accessed repeatedly: 0x13 = 0001 0011 (binary)
- Cache Hits: 9
- Cache Misses: 1
- Total Reads: 10
- First access to 0x13 was a miss, followed by 9 hits

Write Hit = Write-back:
→ Data is written only to cache initially. The dirty bit (D) tracks modifications.
→ No write to main memory unless block is evicted while dirty.

Write Miss = Write-allocate:
→ On a write miss, the block is first brought into the cache, then modified.

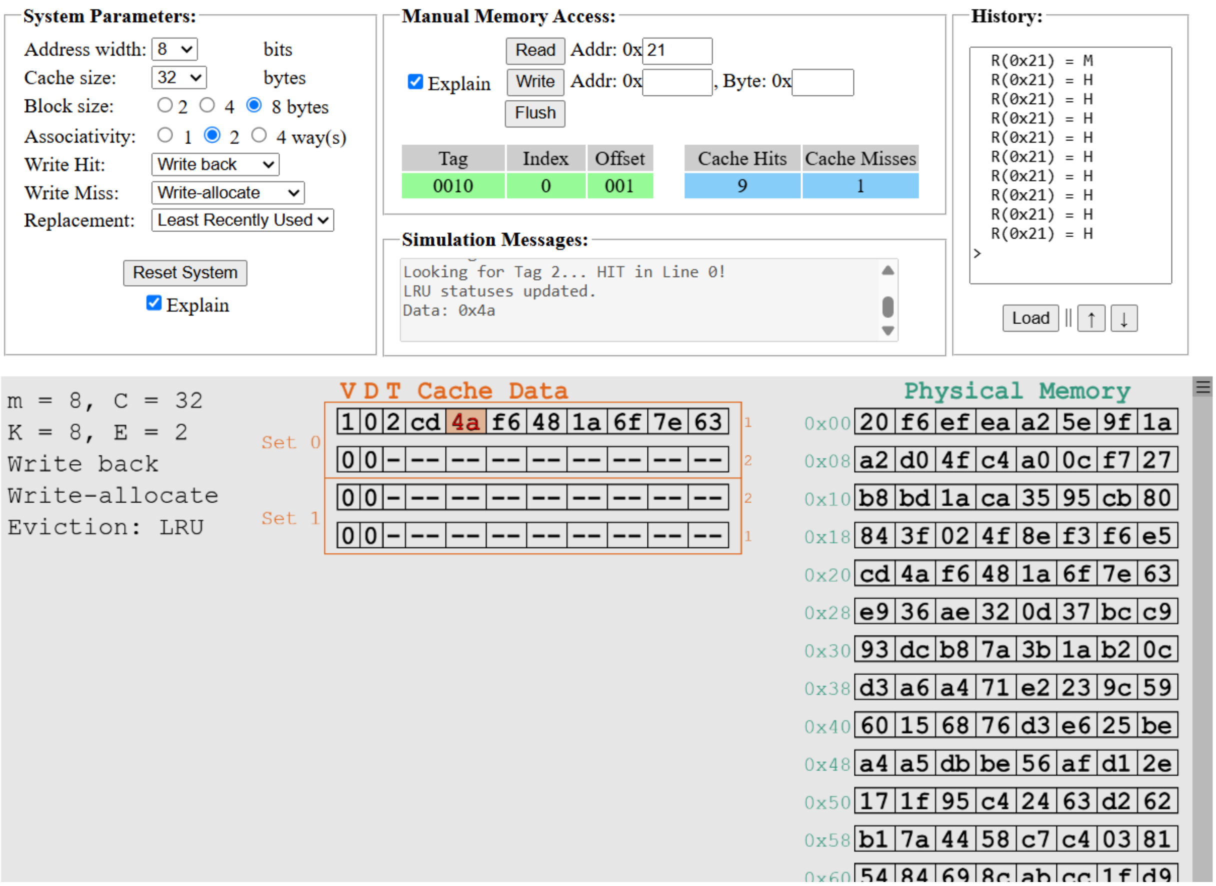
Current Scenario:
→ Only reads, no writes issued
→ Dirty bit remains 0 (clean)

Eviction & LRU Policy

- Cache uses Least Recently Used (LRU) for eviction within each set
- No eviction occurred (only one line loaded in Set 0)
- Line 0 in Set 0 is being repeatedly accessed (and stays MRU)

Conclusion

- Cache structure is behaving correctly: set-indexing, block placement, and LRU tracking all operate as expected.
- The write-back and write-allocate policies are configured, but not exercised in this read-only scenario.
- Repeated accesses to 0x13 hit in cache due to temporal locality.
- Efficient use of cache with minimal overhead (1 miss only).



System Parameters:

- Address width: 8 bits
- Cache size: 32 bytes
- Block size: ○ 2 ○ 4 ● 8 bytes
- Associativity: ○ 1 ● 2 ○ 4 way(s)
- Write Hit: Write back
- Write Miss: Write-allocate
- Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

☑ Explain

Addr: 0x 21

Addr: 0x , Byte: 0x

Tag	Index	Offset
0010	0	001

Cache Hits	Cache Misses
9	1

History:

```
R(0x21) = M
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
R(0x21) = H
>
```

Simulation Messages:

```
Looking for Tag 2... HIT in Line 0!
LRU statuses updated.
Data: 0x4a
```

```
m = 8, C = 32
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

```
       V D T Cache Data
Set 0  1|0|2|cd|4a|f6|48|1a|6f|7e|63  1
       0|0|-|--|--|--|--|--|--|--|--  2
Set 1  0|0|-|--|--|--|--|--|--|--|--  2
       0|0|-|--|--|--|--|--|--|--|--  1
```

```
Physical Memory
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
0x60 54 84 69 8c ab cc 1f d9
```

Parameter	Value
Address width (m)	8 bits (0x00 to 0xFF)
Cache size (C)	32 bytes
Block size (B)	8 bytes
Associativity (E)	2-way set associative
Number of sets (S)	2
Write policy (hit)	Write-back
Write policy (miss)	Write-allocate
Replacement policy	LRU

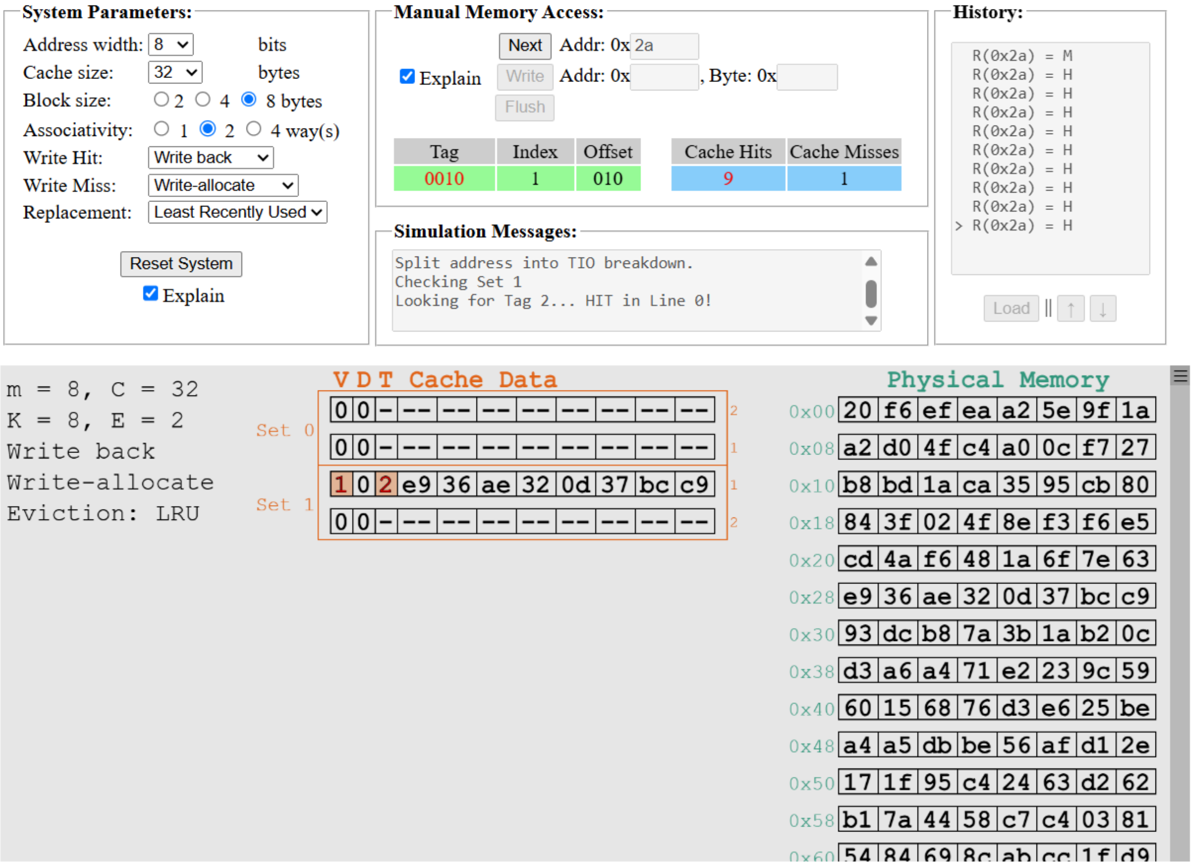
Field	**Bits**	**Purpose**
Tag	[7:4]	Identifies block
Index	[3]	Selects cache set
Offset	[2:0]	Byte in block

Write Policy Analysis

- Write-back (Hit): Writes are done in cache only. Main memory updated only when dirty block is evicted.
- Write-allocate (Miss): On a miss during write, the block is loaded into cache first, then written.
- This case: All read operations, so:
  - No block marked dirty (D = 0)
  - No main memory writes

LRU (Least Recently Used) Analysis

- Only one line in Set 0 is used (Line 0)
- LRU status updated correctly per simulation message
- No eviction needed yet



System Parameters:

Address width: 8 bits

Cache size: 32 bytes

Block size: ○ 2 ○ 4 ● 8 bytes

Associativity: ○ 1 ● 2 ○ 4 way(s)

Write Hit: Write back

Write Miss: Write-allocate

Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x 2a

☑ Explain

Addr: 0x , Byte: 0x

Tag	Index	Offset
0010	1	010

Cache Hits	Cache Misses
9	1

Simulation Messages:

```
Split address into TIO breakdown.
Checking Set 1
Looking for Tag 2... HIT in Line 0!
```

History:

```
  R(0x2a) = M
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
  R(0x2a) = H
> R(0x2a) = H
```

```
m = 8, C = 32
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

```
Set 0  0 0 - -- -- -- -- -- -- -- --  2
       0 0 - -- -- -- -- -- -- -- --  1
Set 1  1 0 2 e9 36 ae 32 0d 37 bc c9  1
       0 0 - -- -- -- -- -- -- -- --  2
```

Physical Memory

```
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
0x60 54 84 69 8c ab cc 1f d9
```

Cache Structure Overview

System Parameters (top-left):

- Address Width (m): 8 bits → 256-byte memory address space
- Cache Size (C): 32 bytes
- Block Size: 8 bytes → each block holds 8 bytes
- Associativity (E): 2-way set associative
- Number of Sets (S):
  Number of sets = Cache size / (Block Size×Associativity)=32/(8×2)=2
  → 2 sets: Set 0 and Set 1
- Write Policy:

    - Write Hit: Write-back
    - Write Miss: Write-allocate
- Eviction: LRU (Least Recently Used)

Address 0x2A Breakdown

Using T-I-O (Tag-Index-Offset):

- Offset bits (b): $\log_2$(Block Size) = $\log_2(8)$ = 3 bits
- Index bits (s): $\log_2$(Number of Sets) = $\log_2(2)$ = 1 bit
- Tag bits (t): 8 - 3 - 1 = 4 bits

Address 0x2A = 00101010
Split as:

- Tag: 0010
- Index: 1
- Offset: 010

➢ Block 0x28 – 0x2F is loaded into Set 1

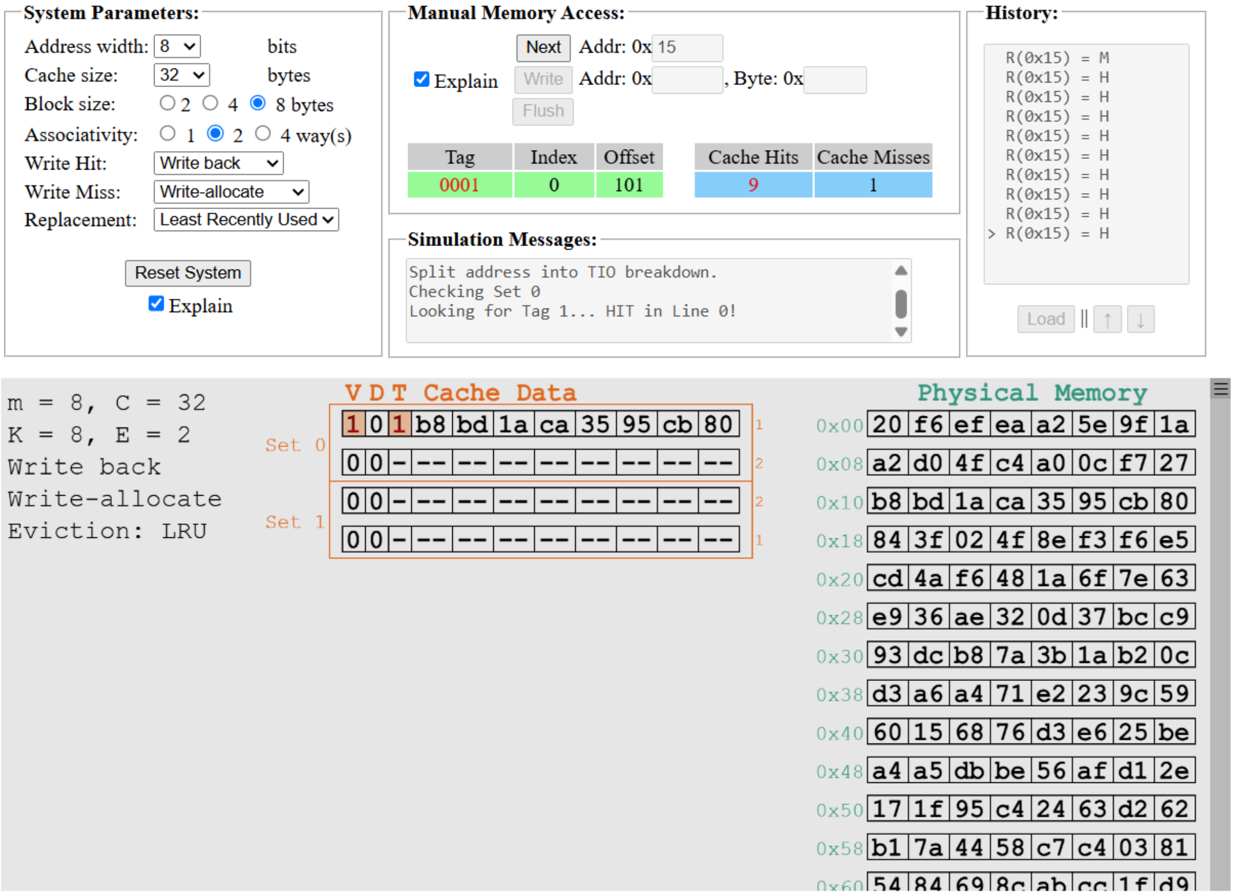
In V D T Cache Data (Set 1):

- The valid line has:
    - V = 1 (valid)
    - D = 0 (not dirty — since only reads so far)
    - Tag = 2 (0010) → matches 0x2A
    - Data = e9 36 ae 32 0d 37 bc c9 ← matches Physical Memory at 0x28

. Write Policy Analysis

- Write Hit: Write-back → Modified blocks are not written immediately to memory.
- Write Miss: Write-allocate → On a write miss, the block is loaded into cache first, then written to.
- Eviction: LRU → If the set is full (2 blocks), the Least Recently Used block is replaced.

In this test, no writes occurred, so the dirty bit remains 0.



System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ● 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

Reset System
☑ Explain

Manual Memory Access:

Next Addr: 0x 15
☑ Explain Write Addr: 0x , Byte: 0x
Flush

Tag	Index	Offset
0001	0	101

Cache Hits	Cache Misses
9	1

Simulation Messages:

```
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 1... HIT in Line 0!
```

History:

```
  R(0x15) = M
  R(0x15) = H
  R(0x15) = H
  R(0x15) = H
  R(0x15) = H
  R(0x15) = H
  R(0x15) = H
  R(0x15) = H
  R(0x15) = H
> R(0x15) = H
```

Load || ↑ ↓

```
m = 8, C = 32
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

```
         V D T Cache Data
Set 0   |1|0|1|b8|bd|1a|ca|35|95|cb|80| 1
        |0|0|-|--|--|--|--|--|--|--|--| 2
Set 1   |0|0|-|--|--|--|--|--|--|--|--| 2
        |0|0|-|--|--|--|--|--|--|--|--| 1
```

```
Physical Memory
0x00|20|f6|ef|ea|a2|5e|9f|1a|
0x08|a2|d0|4f|c4|a0|0c|f7|27|
0x10|b8|bd|1a|ca|35|95|cb|80|
0x18|84|3f|02|4f|8e|f3|f6|e5|
0x20|cd|4a|f6|48|1a|6f|7e|63|
0x28|e9|36|ae|32|0d|37|bc|c9|
0x30|93|dc|b8|7a|3b|1a|b2|0c|
0x38|d3|a6|a4|71|e2|23|9c|59|
0x40|60|15|68|76|d3|e6|25|be|
0x48|a4|a5|db|be|56|af|d1|2e|
0x50|17|1f|95|c4|24|63|d2|62|
0x58|b1|7a|44|58|c7|c4|03|81|
0x60|54|84|69|8c|ab|cc|1f|d9|
```

Cache Structure Overview

System Parameters:

- Address Width (m): 8 bits → 256 bytes of addressable memory

- Cache Size (C): 32 bytes
- Block Size: 8 bytes
- Associativity (E): 2-way set associative
- Number of Sets (S):Cache size / (Block Size×Associativity)=32/(8×2)=2 ⇒Set 0 and Set 1
- Write Hit: Write-back
- Write Miss: Write-allocate
- Eviction: LRU

Address Breakdown

We analyze address 0x15 → binary: 0001 0101

- Offset bits (b): 3 bits → Block size = 8 bytes
- Index bits (s): 1 bit → 2 sets
- Tag bits (t): 8 - 3 - 1 = 4 bits

Split of 0x15 = 0001 0101:

- Tag: 0001 → 1 in hex
- Index: 0
- Offset: 101

Therefore, block [0x10 – 0x17] is being accessed, and it maps to Set 0.

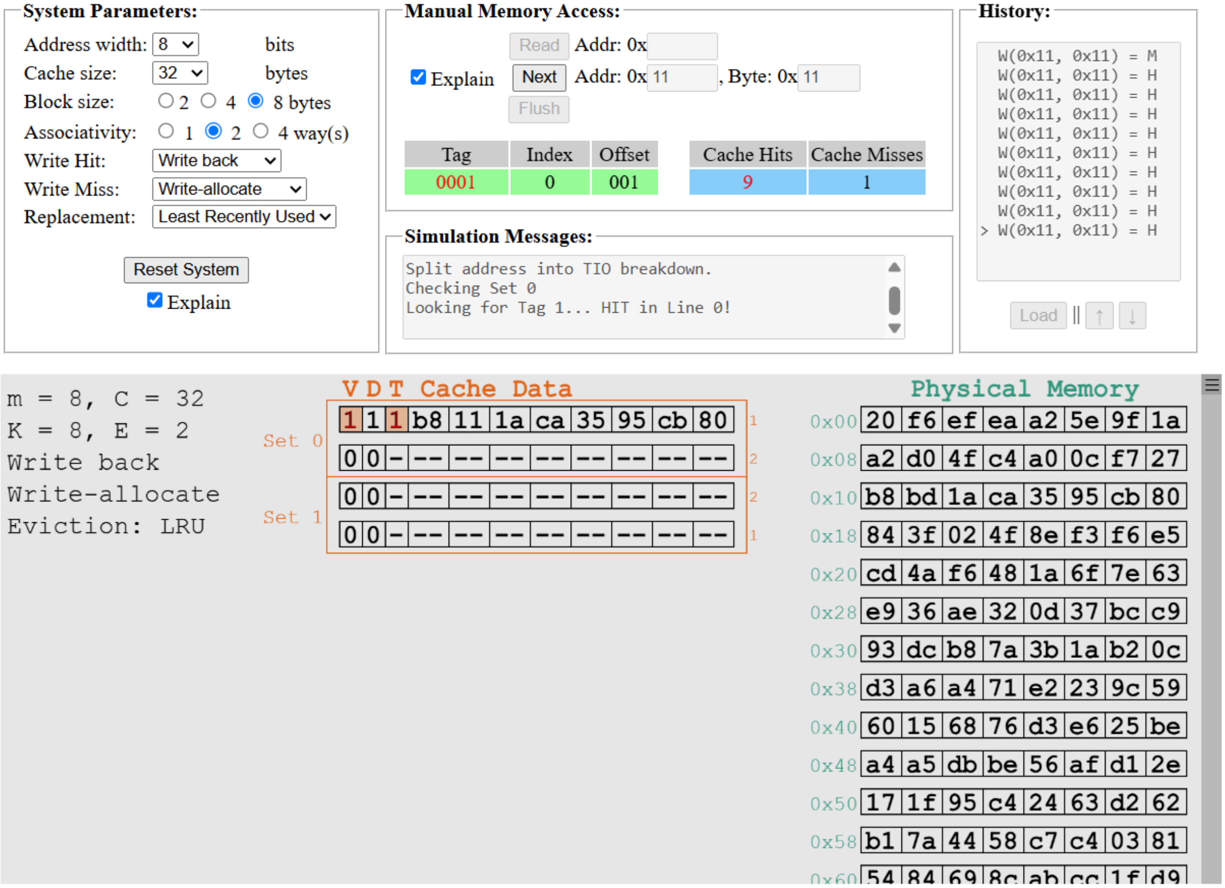
Write Policy Analysis

- Write Hit = Write-back
  → Modifications to cached data are not immediately reflected in main memory.

→ Dirty bit (D) becomes 1 only if a write occurs.

- Write Miss = Write-allocate
  → On a miss during write, the block is first loaded into cache, then written.

In this case, only reads are performed — no change to D (still 0).



System Parameters:

Address width: 8 bits
Cache size: 32 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ● 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x

☑ Explain

Addr: 0x 11, Byte: 0x 11

Tag	Index	Offset	Cache Hits	Cache Misses
0001	0	001	9	1

Simulation Messages:

```
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 1... HIT in Line 0!
```

History:

```
  W(0x11, 0x11) = M
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
  W(0x11, 0x11) = H
> W(0x11, 0x11) = H
```

```
m = 8, C = 32
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

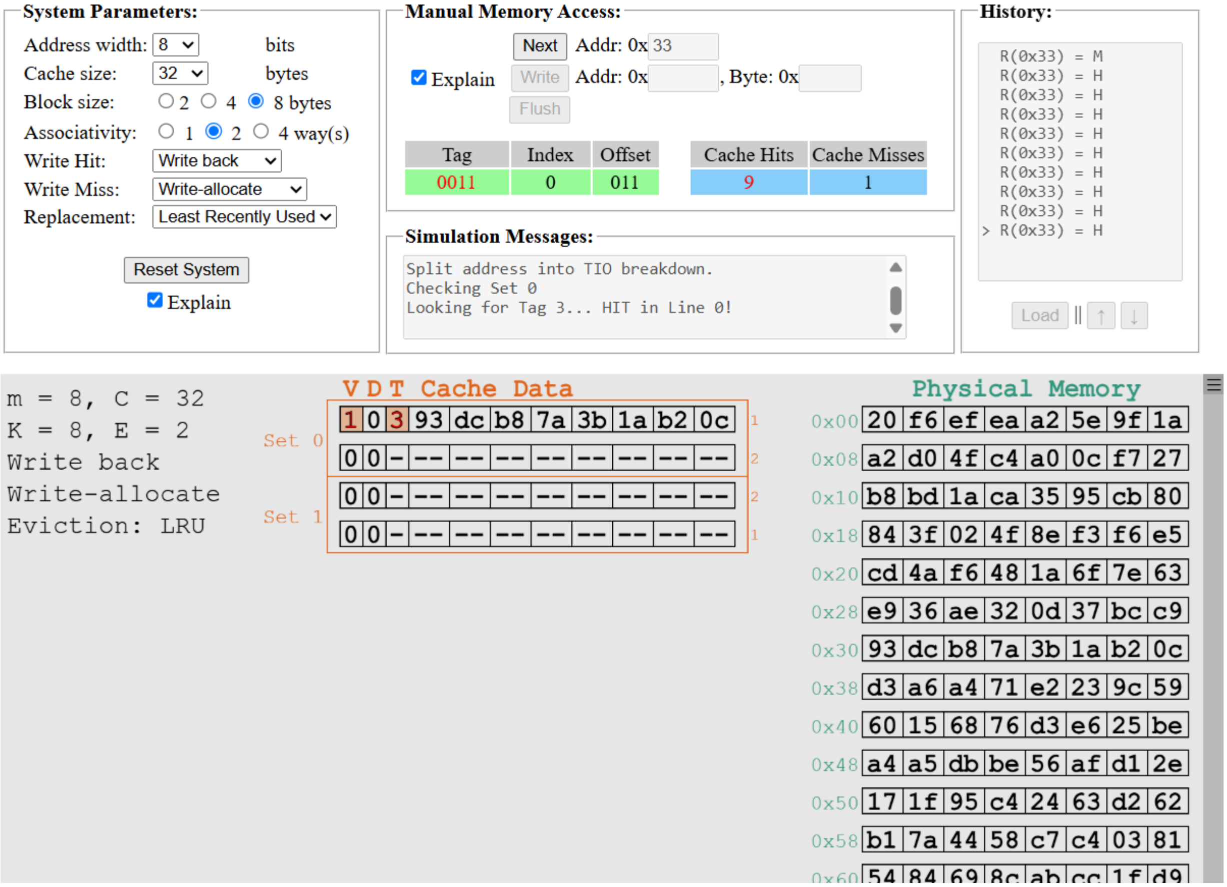
```
Set 0  |1|1|1|b8|11|1a|ca|35|95|cb|80|  1
       |0|0|-|--|--|--|--|--|--|--|--|  2
Set 1  |0|0|-|--|--|--|--|--|--|--|--|  2
       |0|0|-|--|--|--|--|--|--|--|--|  1
```

Physical Memory

```
0x00 20 f6 ef ea a2 5e 9f 1a
0x08 a2 d0 4f c4 a0 0c f7 27
0x10 b8 bd 1a ca 35 95 cb 80
0x18 84 3f 02 4f 8e f3 f6 e5
0x20 cd 4a f6 48 1a 6f 7e 63
0x28 e9 36 ae 32 0d 37 bc c9
0x30 93 dc b8 7a 3b 1a b2 0c
0x38 d3 a6 a4 71 e2 23 9c 59
0x40 60 15 68 76 d3 e6 25 be
0x48 a4 a5 db be 56 af d1 2e
0x50 17 1f 95 c4 24 63 d2 62
0x58 b1 7a 44 58 c7 c4 03 81
0x60 54 84 69 8c ab cc 1f d9
```

Aspect	**Details**
Address Accessed	0x11
Access Type	Write (W)
Cache Hit/Miss	1 Miss (first access), 9 Hits (subsequent writes)

Block Loaded	0x10–0x17 → Loaded into Set 0, Line 0
Tag in Cache	Tag = 0001 (hex = 1)
Set Index	0
Offset	1 (second byte in block)
Modified Byte	0x11 written at offset 1 (was bd, now 11)
Valid Bit (V)	1 (block is valid)
Dirty Bit (D)	1 (block has been modified)
Write Policy	Write-back (memory not yet updated), Write-allocate (block loaded)
Eviction Policy	LRU (but no eviction yet occurred)



System Parameters:

Address width: 8 bits

Cache size: 32 bytes

Block size: ○ 2 ○ 4 ● 8 bytes

Associativity: ○ 1 ● 2 ○ 4 way(s)

Write Hit: Write back

Write Miss: Write-allocate

Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

Addr: 0x 33

☑ Explain

Addr: 0x , Byte: 0x

Tag	Index	Offset
0011	0	011

Cache Hits	Cache Misses
9	1

History:

```
  R(0x33) = M
  R(0x33) = H
  R(0x33) = H
  R(0x33) = H
  R(0x33) = H
  R(0x33) = H
  R(0x33) = H
  R(0x33) = H
  R(0x33) = H
> R(0x33) = H
```

Simulation Messages:

```
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 3... HIT in Line 0!
```

```
m = 8, C = 32
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

V D T Cache Data

	V	D	T									
Set 0	1	0	3	93	dc	b8	7a	3b	1a	b2	0c	1
	0	0	-	--	--	--	--	--	--	--	--	2
Set 1	0	0	-	--	--	--	--	--	--	--	--	2
	0	0	-	--	--	--	--	--	--	--	--	1

Physical Memory

0x00	20	f6	ef	ea	a2	5e	9f	1a
0x08	a2	d0	4f	c4	a0	0c	f7	27
0x10	b8	bd	1a	ca	35	95	cb	80
0x18	84	3f	02	4f	8e	f3	f6	e5
0x20	cd	4a	f6	48	1a	6f	7e	63
0x28	e9	36	ae	32	0d	37	bc	c9
0x30	93	dc	b8	7a	3b	1a	b2	0c
0x38	d3	a6	a4	71	e2	23	9c	59
0x40	60	15	68	76	d3	e6	25	be
0x48	a4	a5	db	be	56	af	d1	2e
0x50	17	1f	95	c4	24	63	d2	62
0x58	b1	7a	44	58	c7	c4	03	81
0x60	54	84	69	8c	ab	cc	1f	d9

Parameter	**Value**
Address width	8 bits
Cache size (C)	32 bytes
Block size (B)	8 bytes
Associativity	2-way set associative
Number of sets (S)	2 sets (C / (B * E) = 32 / (8×2) = 2)
Write Hit Policy	Write-back
Write Miss Policy	Write-allocate
Replacement Policy	LRU (Least Recently Used)

Address Breakdown

We analyze address 0x15 → binary: 0001 0101

- Offset bits (b): 3 bits → Block size = 8 bytes
- Index bits (s): 1 bit → 2 sets
- Tag bits (t): 8 - 3 - 1 = 4 bits

Split of 0x15 = 0001 0101:

- Tag: 0001 → 1 in hex
- Index: 0
- Offset: 101

Therefore, block [0x10 – 0x17] is being accessed, and it maps to Set 0.

Cache Contents

In the VDT Cache Data:

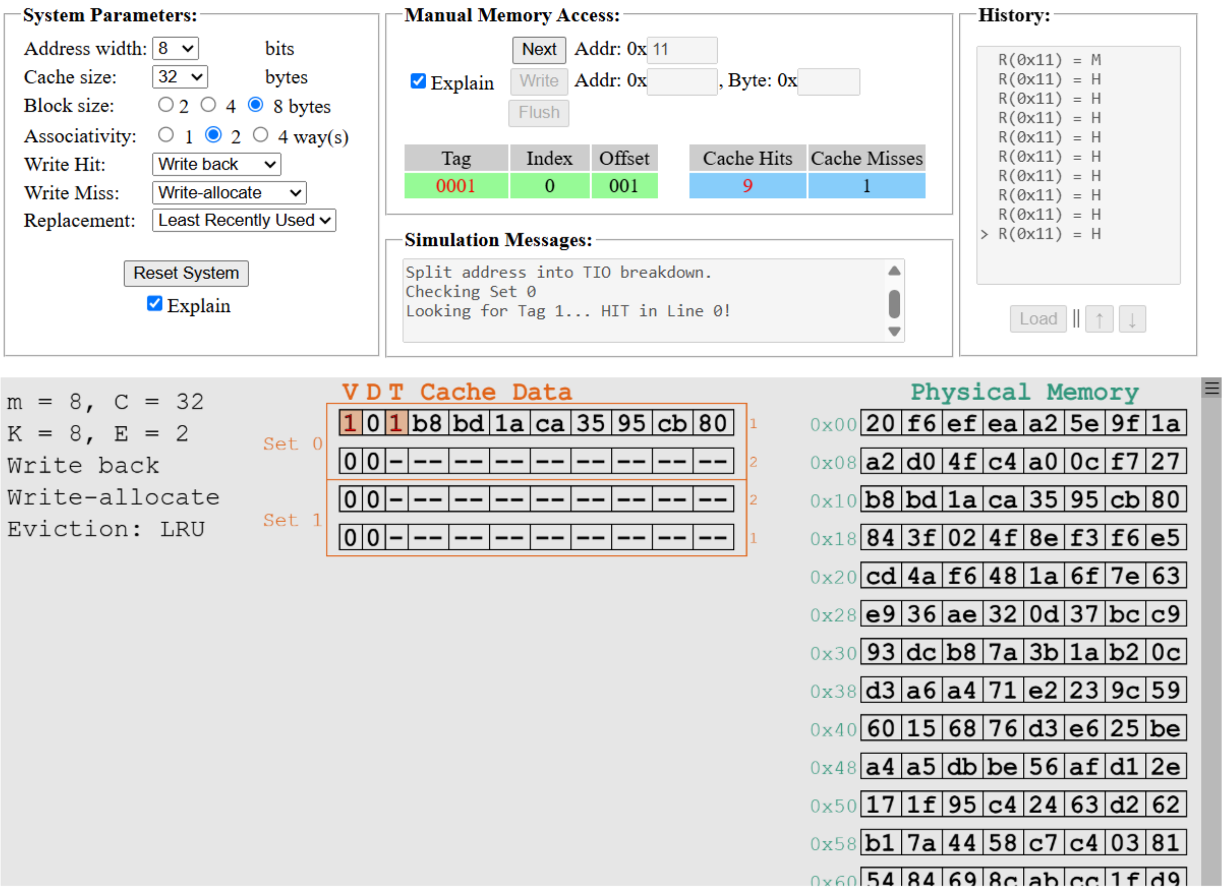
- Set 0, Line 0 contains:
  - V = 1 (valid)
  - D = 0 (clean)
  - Tag = 1
  - Data = b8 bd 1a ca 35 95 cb 80
    → Matches memory from 0x10–0x17

Block 0x10–0x17 loaded into Set 0, Line 0

Write Policy Analysis

- Write Hit = Write-back
  → Modifications to cached data are not immediately reflected in main memory.
  → Dirty bit (D) becomes 1 only if a write occurs.
- Write Miss = Write-allocate
  → On a miss during write, the block is first loaded into cache, then written.

In this case, only reads are performed — no change to D (still 0).



System Parameters:

Address width: 8 bits

Cache size: 32 bytes

Block size: ○ 2 ○ 4 ● 8 bytes

Associativity: ○ 1 ● 2 ○ 4 way(s)

Write Hit: Write back

Write Miss: Write-allocate

Replacement: Least Recently Used

Reset System

☑ Explain

Manual Memory Access:

Next Addr: 0x 11

☑ Explain Write Addr: 0x , Byte: 0x

Flush

Tag	Index	Offset	Cache Hits	Cache Misses
0001	0	001	9	1

Simulation Messages:

```
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 1... HIT in Line 0!
```

History:

```
  R(0x11) = M
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
  R(0x11) = H
> R(0x11) = H
```

Load || ↑ ↓

```
m = 8, C = 32                 V D T  Cache Data                           Physical Memory
K = 8, E = 2          Set 0   1|0|1|b8|bd|1a|ca|35|95|cb|80  1      0x00 20|f6|ef|ea|a2|5e|9f|1a
Write back                    0|0|-|--|--|--|--|--|--|--|--  2      0x08 a2|d0|4f|c4|a0|0c|f7|27
Write-allocate                0|0|-|--|--|--|--|--|--|--|--  2      0x10 b8|bd|1a|ca|35|95|cb|80
Eviction: LRU         Set 1   0|0|-|--|--|--|--|--|--|--|--  1      0x18 84|3f|02|4f|8e|f3|f6|e5
                                                                     0x20 cd|4a|f6|48|1a|6f|7e|63
                                                                     0x28 e9|36|ae|32|0d|37|bc|c9
                                                                     0x30 93|dc|b8|7a|3b|1a|b2|0c
                                                                     0x38 d3|a6|a4|71|e2|23|9c|59
                                                                     0x40 60|15|68|76|d3|e6|25|be
                                                                     0x48 a4|a5|db|be|56|af|d1|2e
                                                                     0x50 17|1f|95|c4|24|63|d2|62
                                                                     0x58 b1|7a|44|58|c7|c4|03|81
                                                                     0x60 54|84|69|8c|ab|cc|1f|d9
```

Parameter	Value
Address width	8 bits
Cache size (C)	32 bytes
Block size (B)	8 bytes
Associativity	2-way set associative
Number of sets (S)	2 sets (C / (B×E) = 32 / (8×2) = 2)
Write Hit Policy	Write-back
Write Miss Policy	Write-allocate
Replacement Policy	LRU

Aspect	**Detail**
Cache Hits	9
Cache Misses	1
Block Loaded	Block from address 0x10–0x17
Loaded Into	Set 0, Line 0
Write Hit Policy	Write-back (memory updated only on eviction)
Write Miss Policy	Write-allocate (block fetched into cache first)
Replacement Policy	LRU (will evict least recently used block when needed)