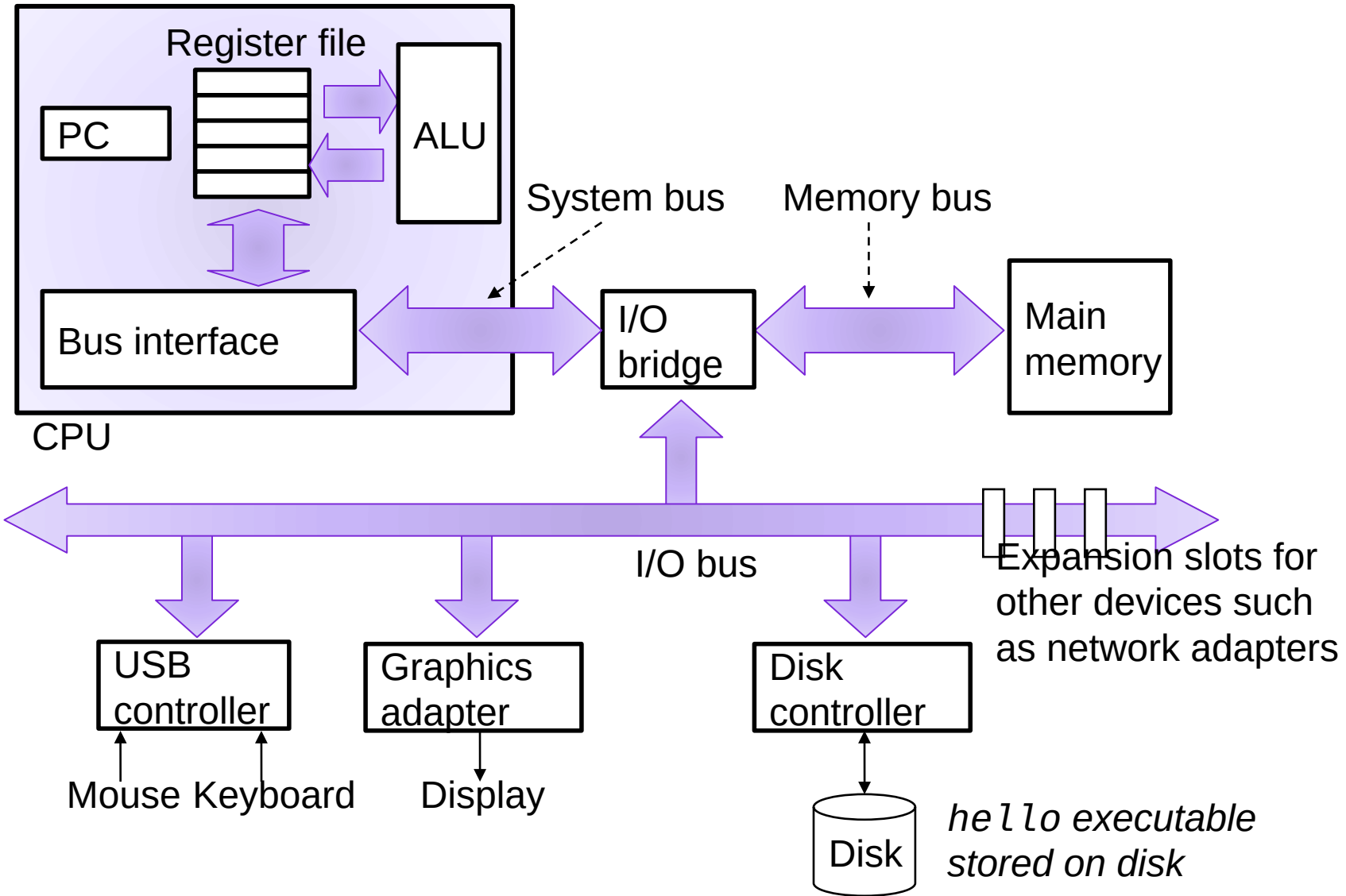
Lecture 10: Caches

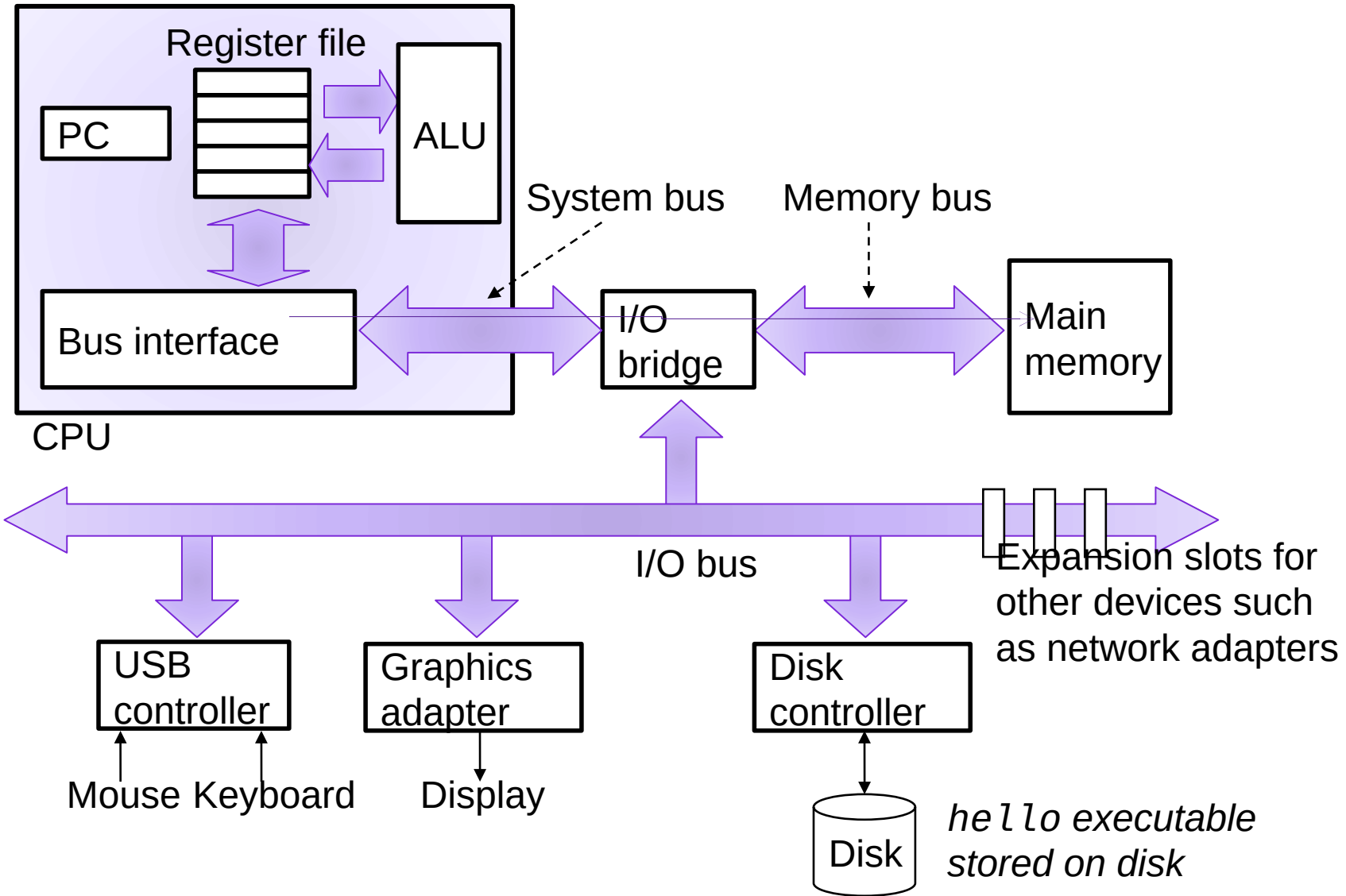
CS 105

Spring 2025

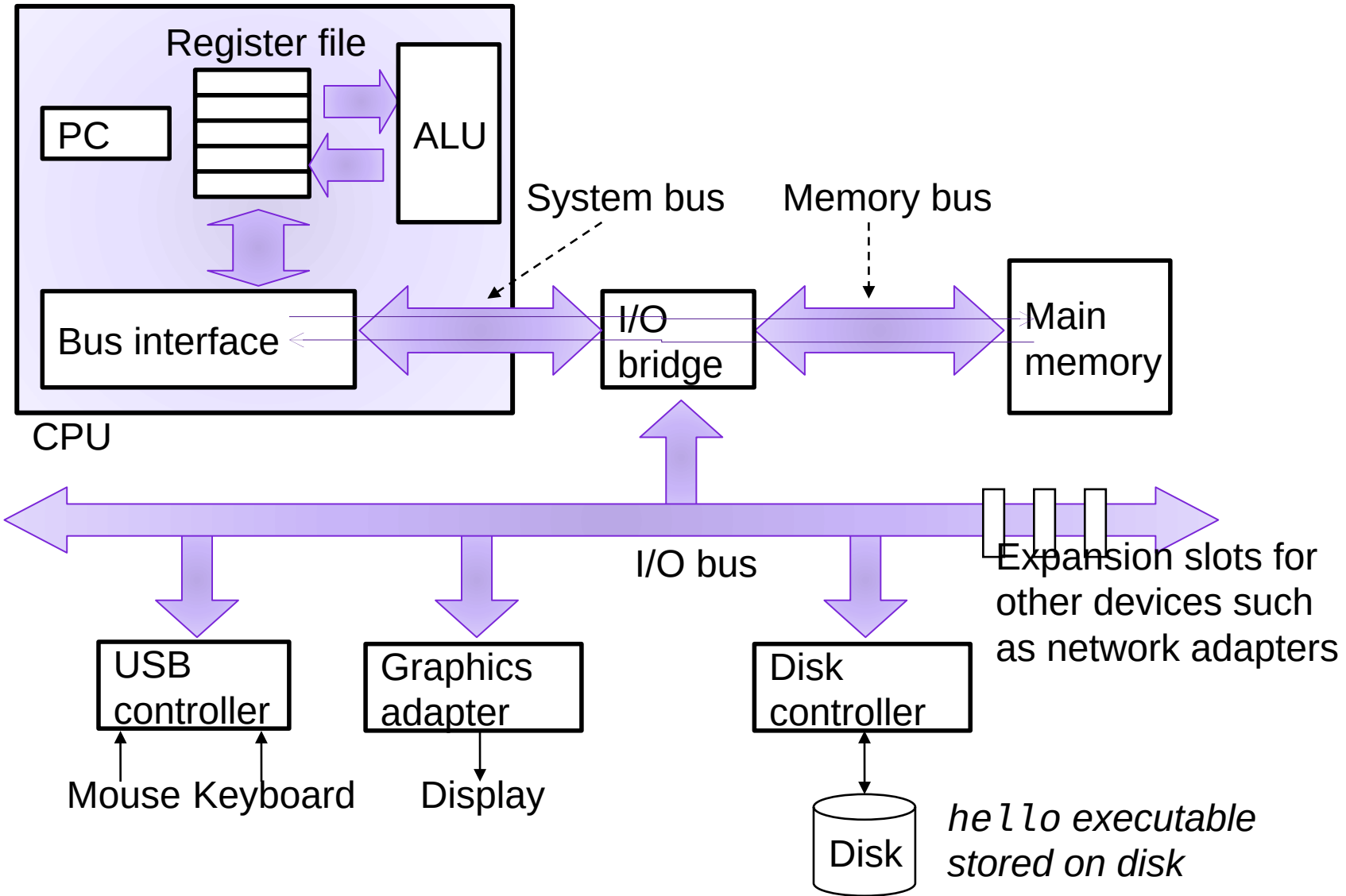
A Computer System



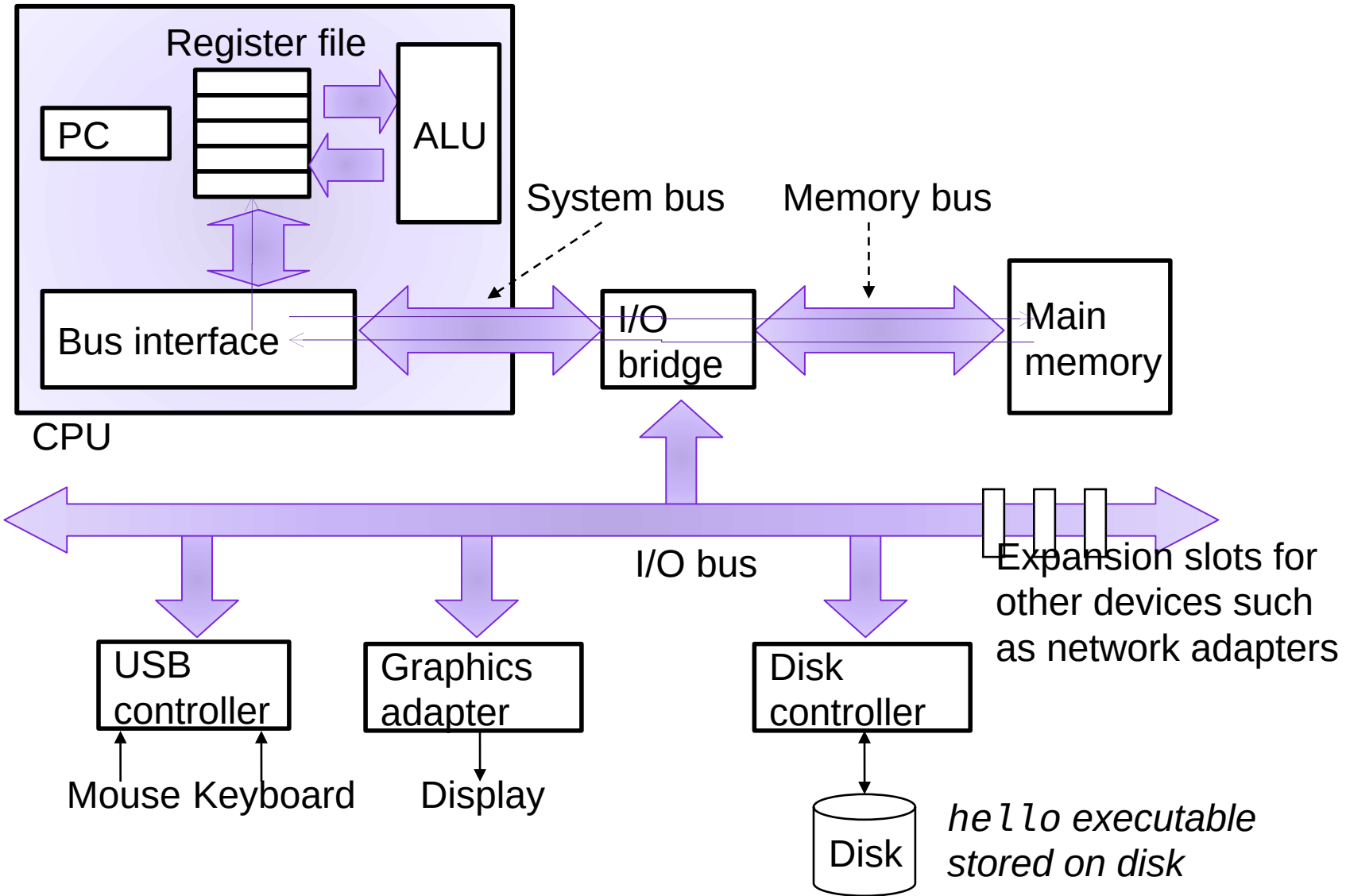
A Computer System



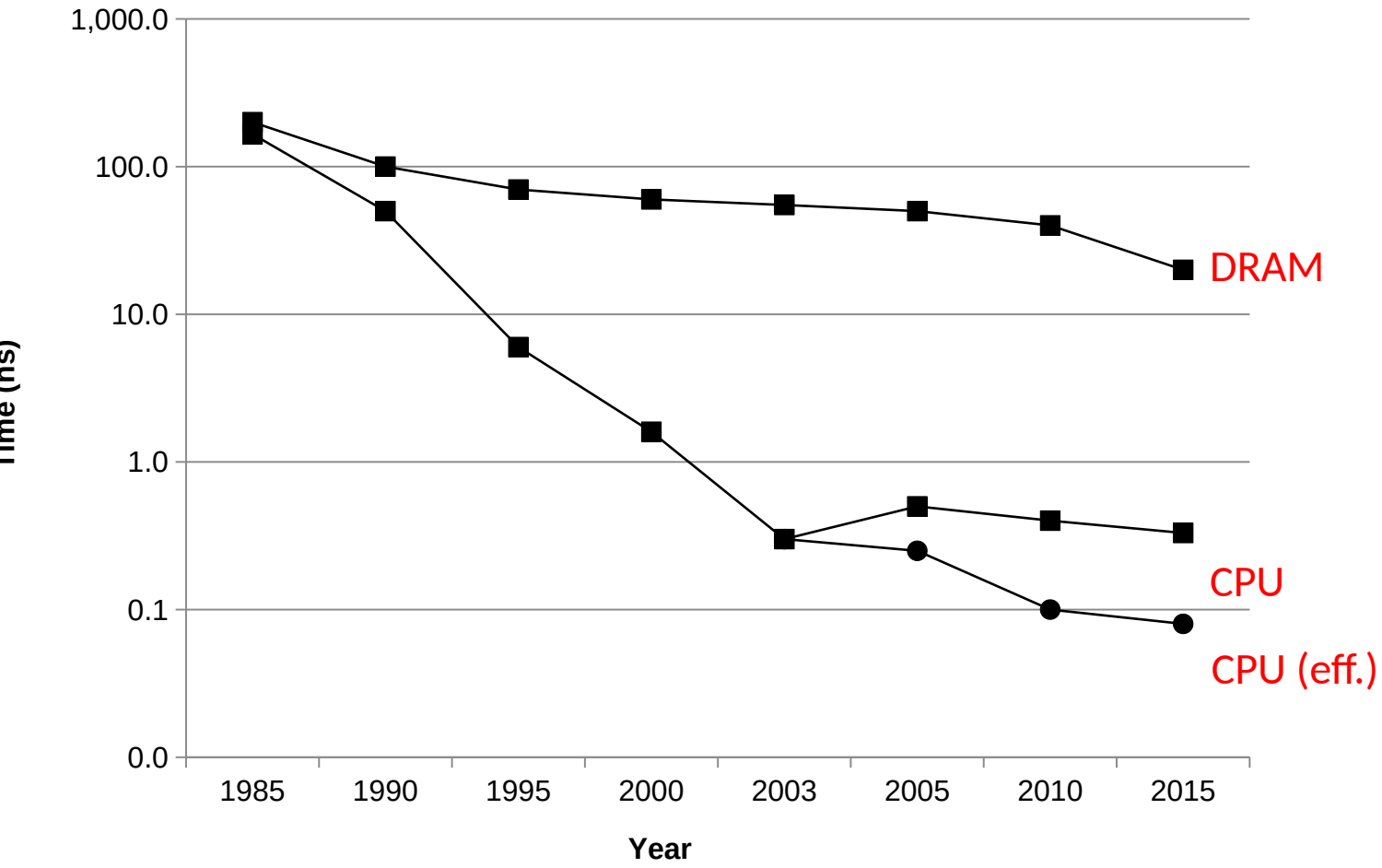
A Computer System



A Computer System

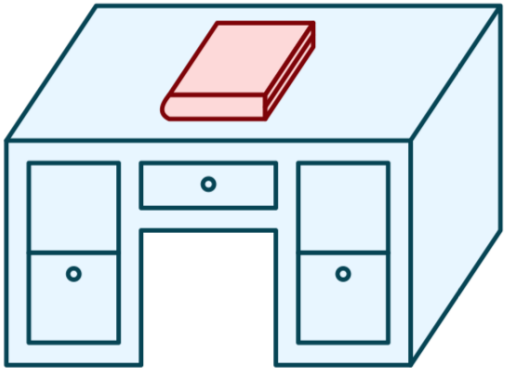


The CPU-Memory Gap

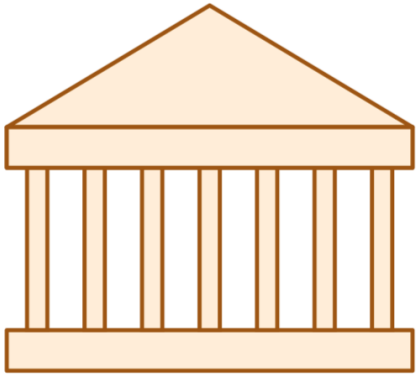


Life without caches

- You decide that you want to learn more about computer systems than is covered in this course
- The library contains all the books you could possibly want, but you don't like to study in libraries, you prefer to study in your dorm room.
- You have the following constraints:



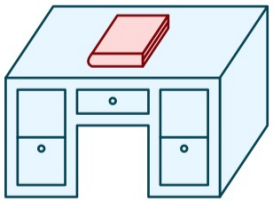
Desk
(can hold one book)



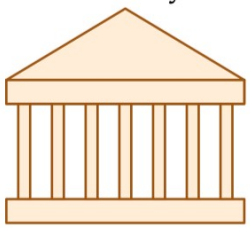
Library
(can hold many books)

Quantifying Speed (without caches)

Desk
(can hold one book)

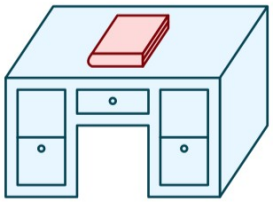


Library
(can hold many books)

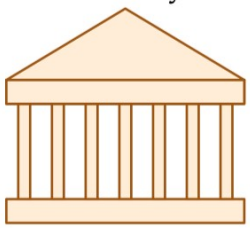


Quantifying Speed (without caches)

Desk
(can hold one book)



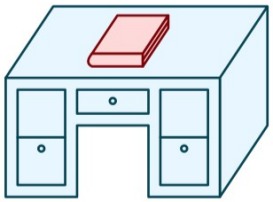
Library
(can hold many books)



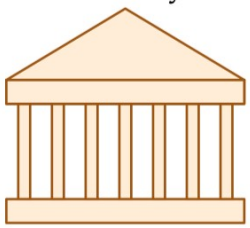
Need book 1

Quantifying Speed (without caches)

Desk
(can hold one book)



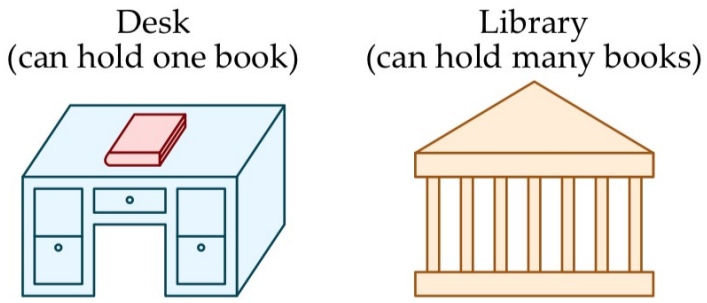
Library
(can hold many books)



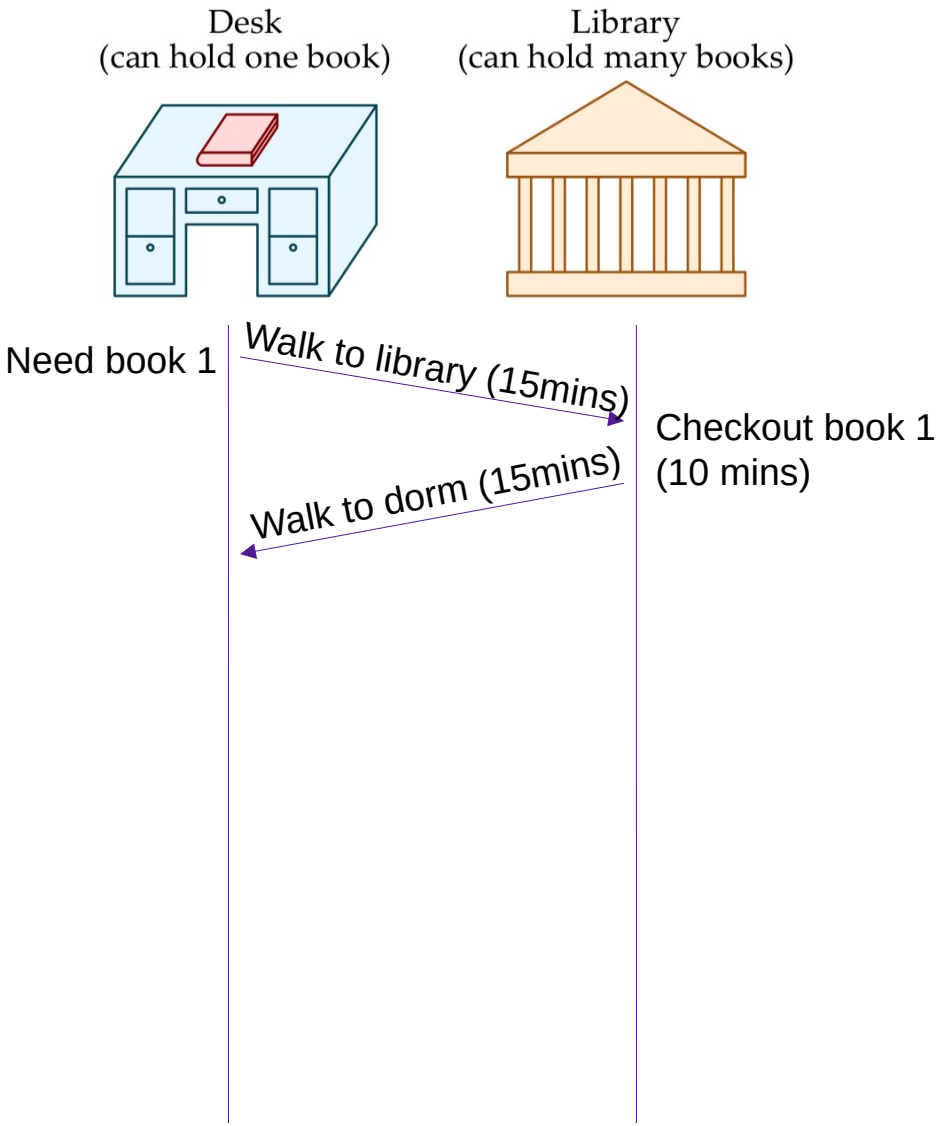
Need book 1

Walk to library (15mins)

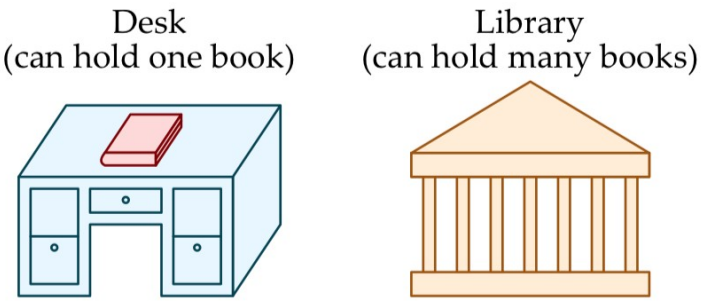
Quantifying Speed (without caches)



Quantifying Speed (without caches)



Quantifying Speed (without caches)



Need book 1

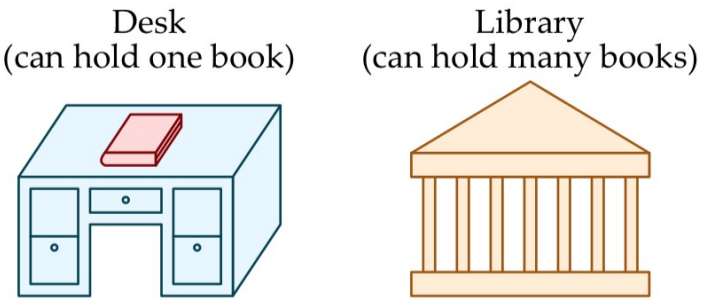
Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

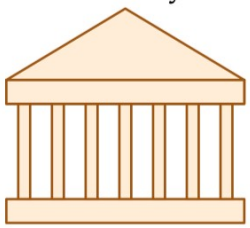
Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Quantifying Speed (without caches)

Desk
(can hold one book)



Library
(can hold many books)

Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

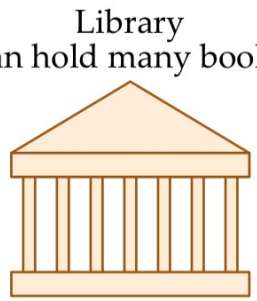
Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

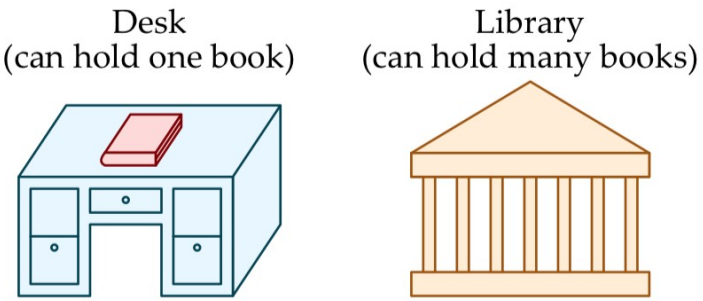
Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

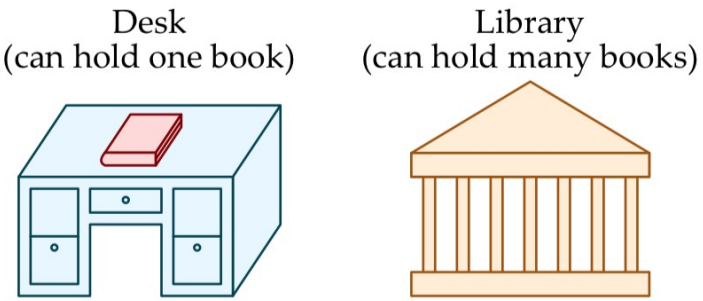
Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

Read book 2
(10 mins)
Need book 1 again!

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

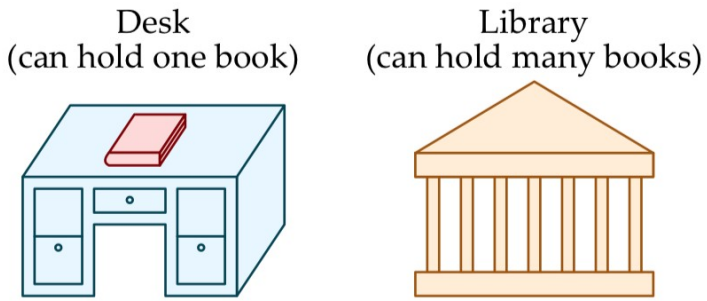
Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

Read book 2
(10 mins)
Need book 1 again!

Walk to library (15mins)

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

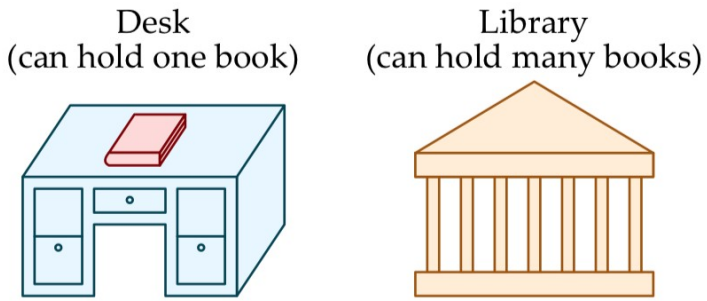
Walk to dorm (15mins)

Read book 2
(10 mins)
Need book 1 again!

Walk to library (15mins)

Return book 2
Checkout book 1
(10 mins)

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

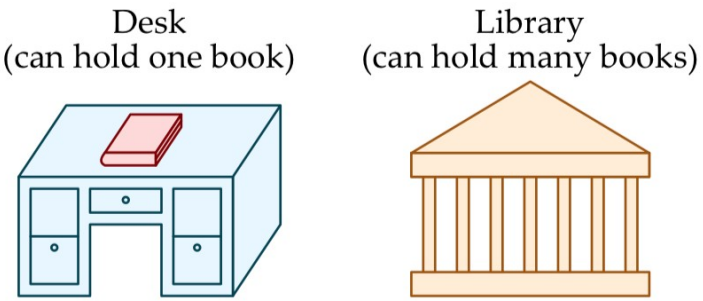
Read book 2
(10 mins)
Need book 1 again!

Walk to library (15mins)

Return book 2
Checkout book 1
(10 mins)

Walk to dorm (15mins)

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

Read book 2
(10 mins)
Need book 1 again!

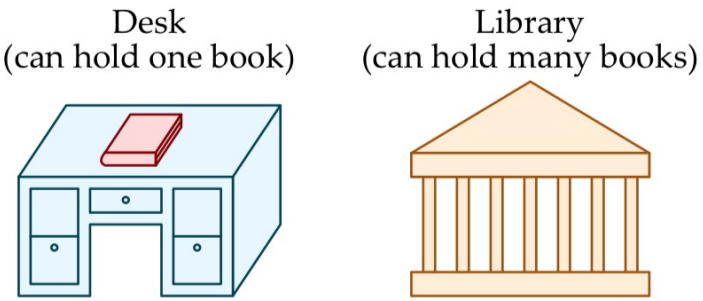
Walk to library (15mins)

Return book 2
Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

Read book 2
(10 mins)
Need book 1 again!

Walk to library (15mins)

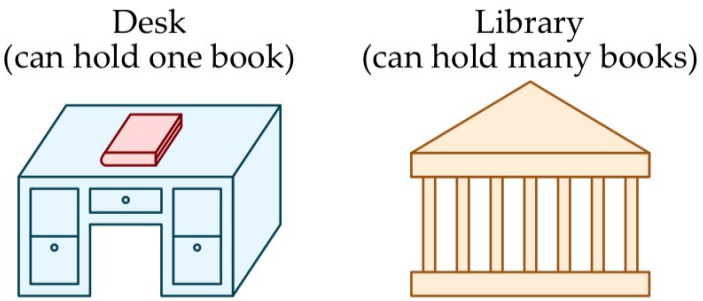
Return book 2
Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)

- Average latency to access a book: 40mins

Quantifying Speed (without caches)



Need book 1

Walk to library (15mins)

Checkout book 1
(10 mins)

Walk to dorm (15mins)

Read book 1
(10 mins)
Need book 2

Walk to library (15mins)

Return book 1
Checkout book 2
(10 mins)

Walk to dorm (15mins)

Read book 2
(10 mins)
Need book 1 again!

Walk to library (15mins)

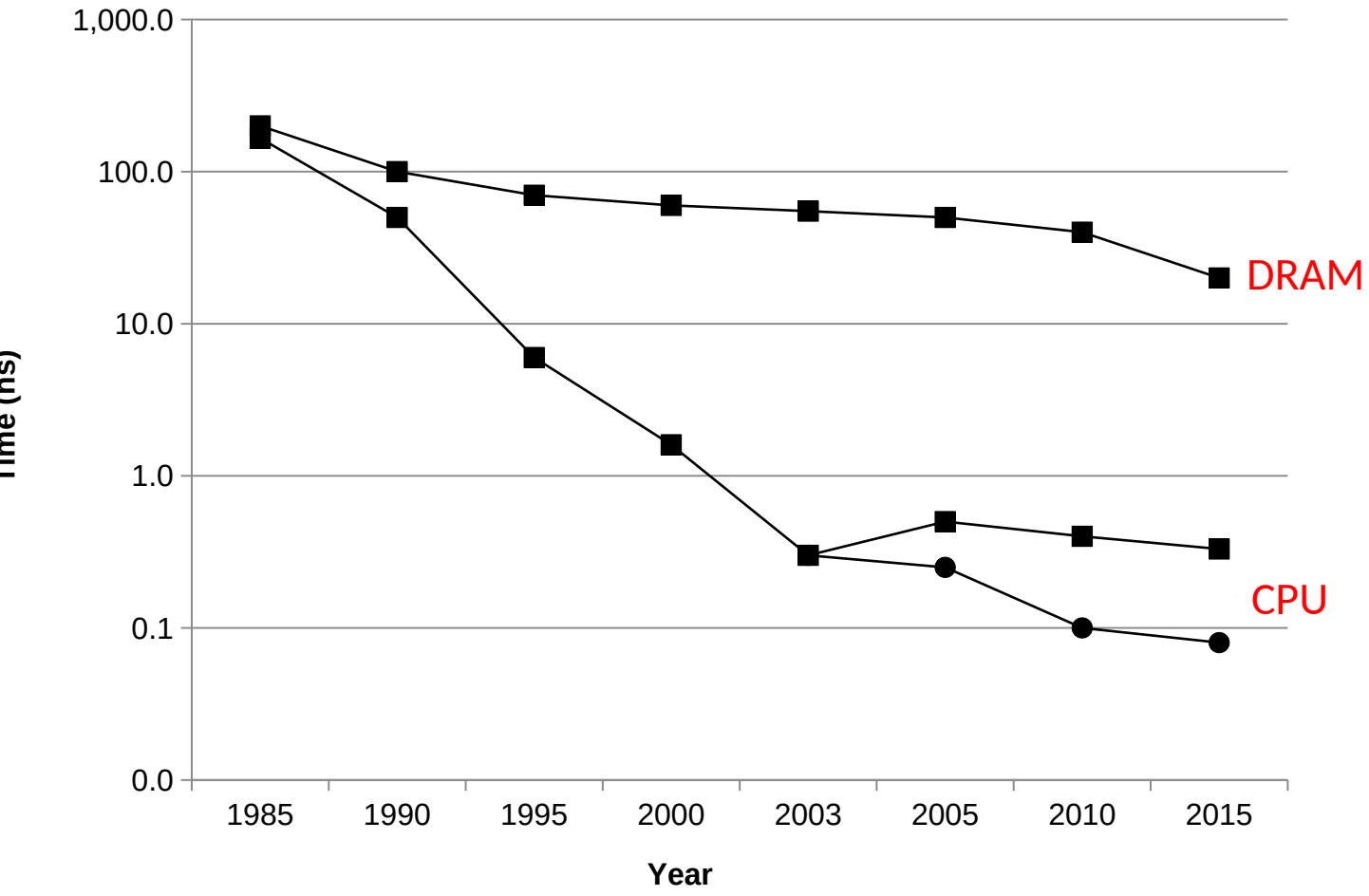
Return book 2
Checkout book 1
(10 mins)

Walk to dorm (15mins)

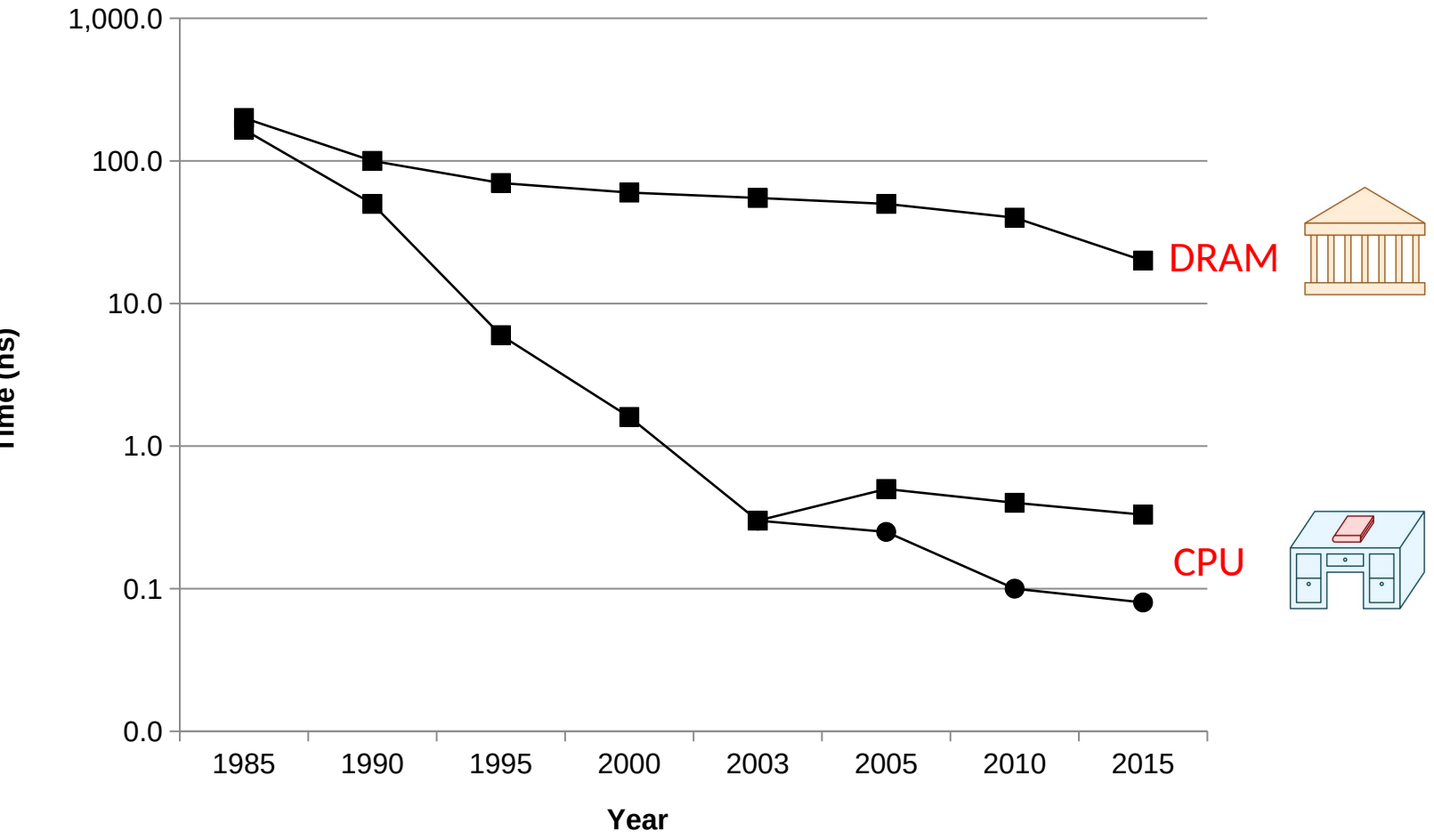
Read book 1
(10 mins)

- Average latency to access a book: 40mins
- Average throughput (incl. reading time): 1.2 books/hr

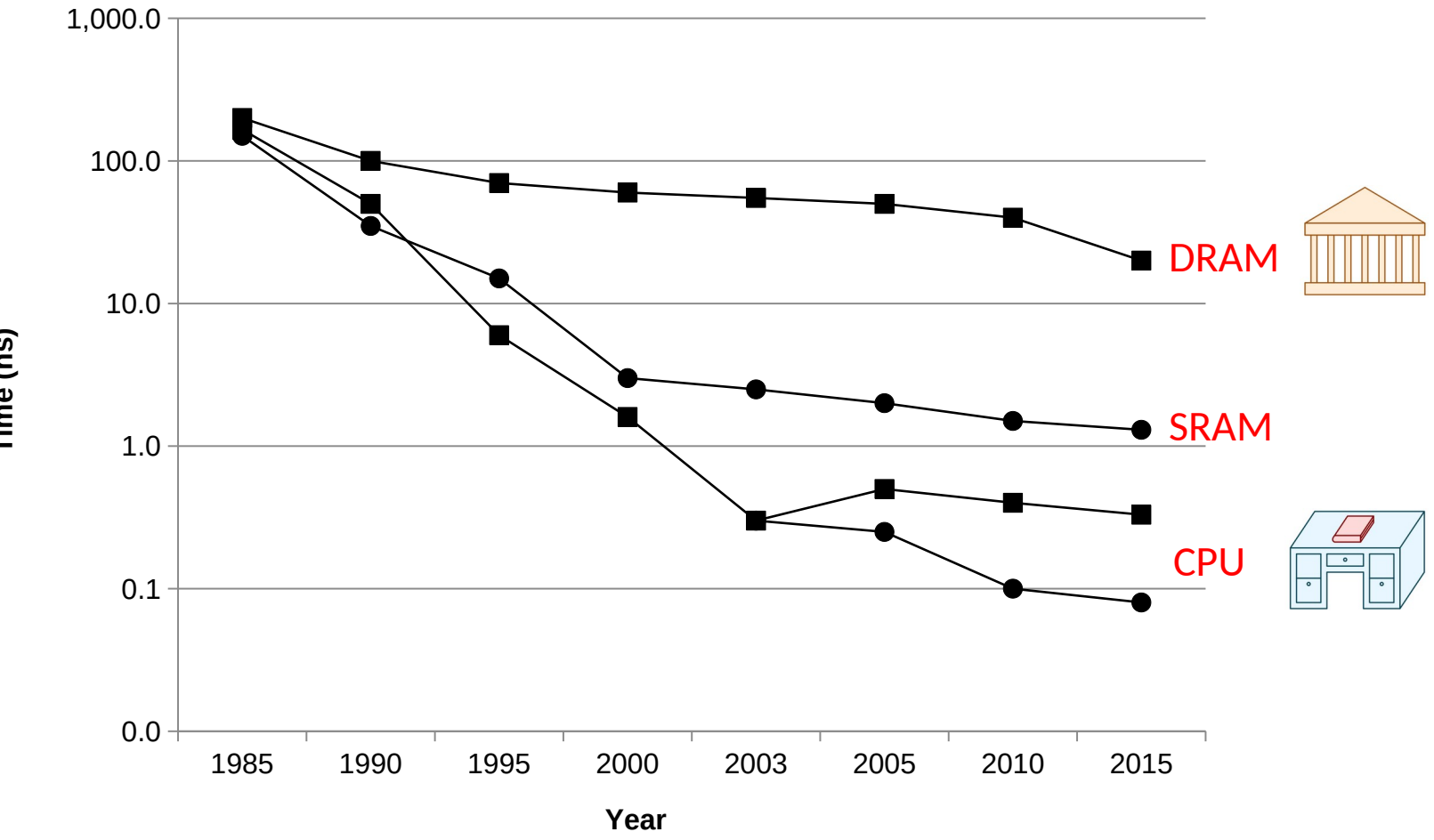
The CPU-Memory Gap



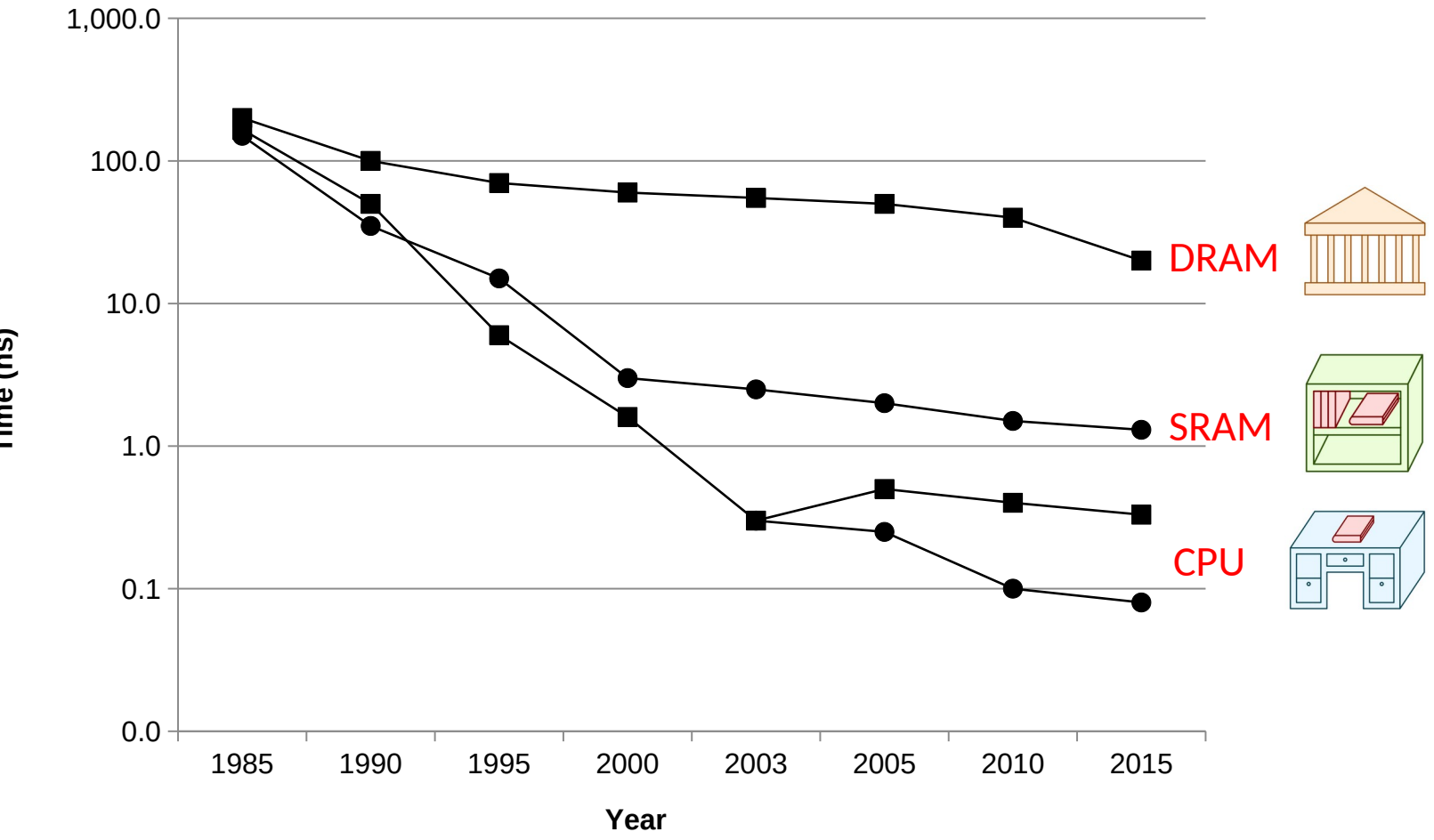
The CPU-Memory Gap



The CPU-Memory Gap

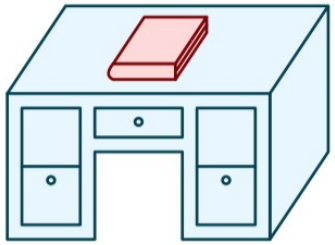


The CPU-Memory Gap

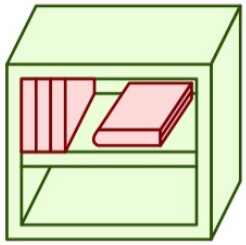


Life with caching

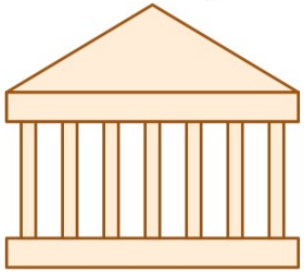
Desk
(can hold one book)



Book Shelf
(can hold a few books)

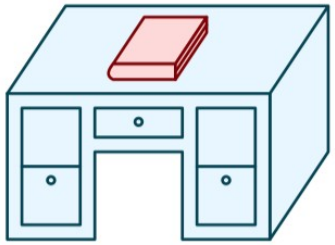


Library
(can hold many books)

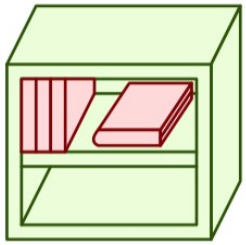


Life with caching

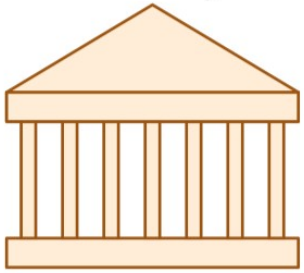
Desk
(can hold one book)



Book Shelf
(can hold a few books)



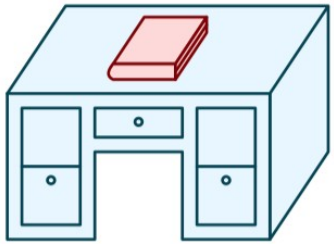
Library
(can hold many books)



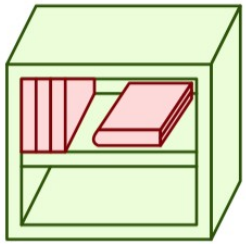
Need book 1

Life with caching

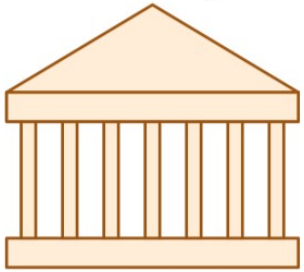
Desk
(can hold one book)



Book Shelf
(can hold a few books)



Library
(can hold many books)

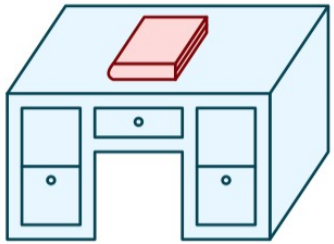


Need book 1

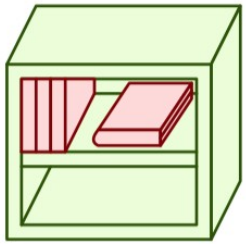
Check bookshelf (5mins)

Life with caching

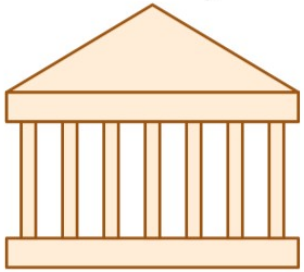
Desk
(can hold one book)



Book Shelf
(can hold a few books)



Library
(can hold many books)



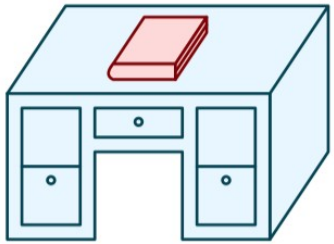
Need book 1

Check bookshelf (5mins)

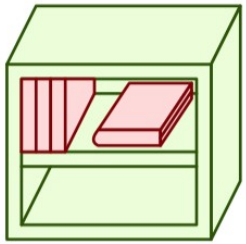
Walk to library (15mins)

Life with caching

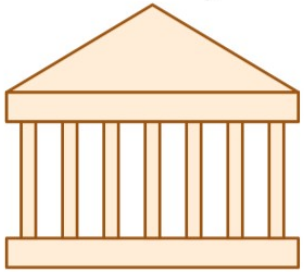
Desk
(can hold one book)



Book Shelf
(can hold a few books)



Library
(can hold many books)



Need book 1

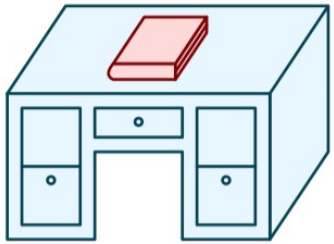
Check bookshelf (5mins)

Walk to library (15mins)

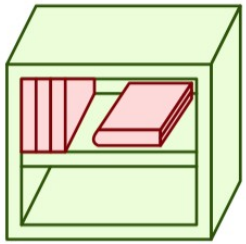
Checkout book 1 & 2
(10 mins)

Life with caching

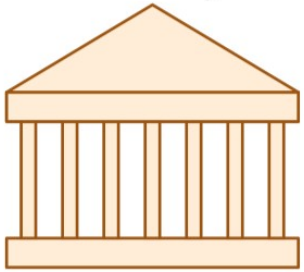
Desk
(can hold one book)



Book Shelf
(can hold a few books)



Library
(can hold many books)



Need book 1

Check bookshelf (5mins)

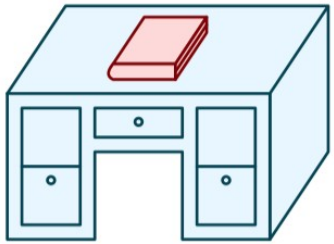
Walk to library (15mins)

Walk to dorm (15mins)

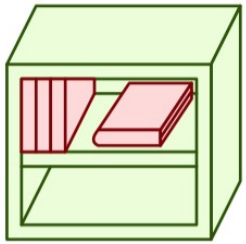
Checkout book 1 & 2
(10 mins)

Life with caching

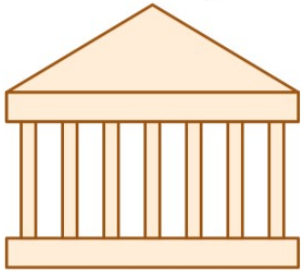
Desk
(can hold one book)

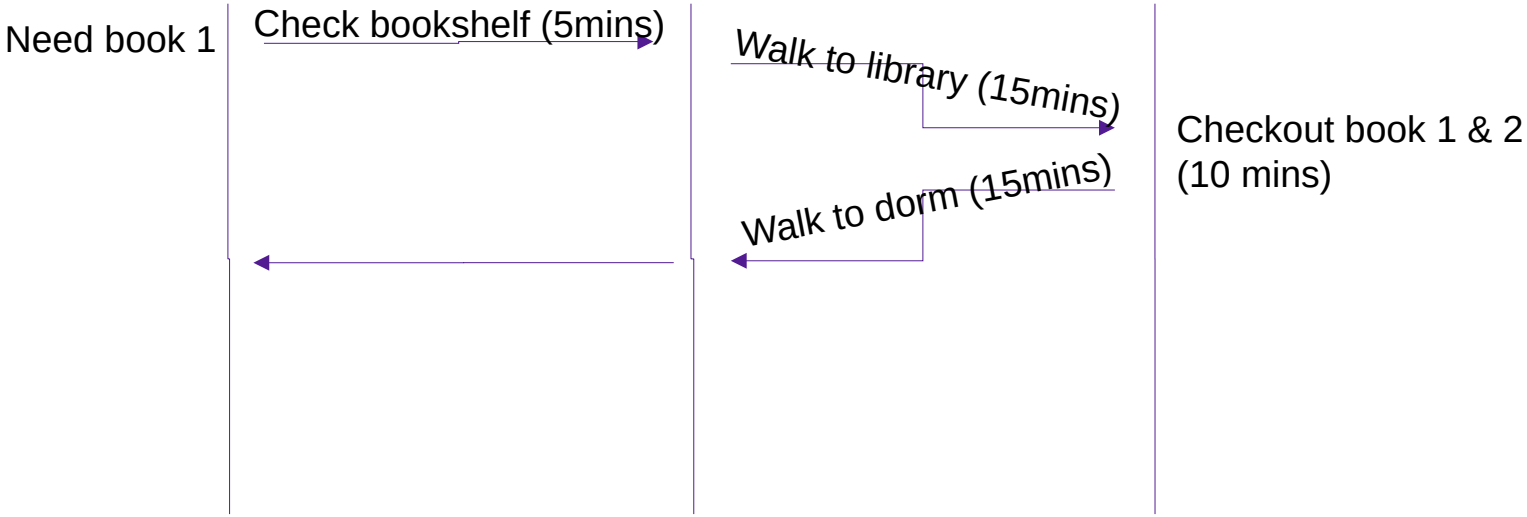


Book Shelf
(can hold a few books)

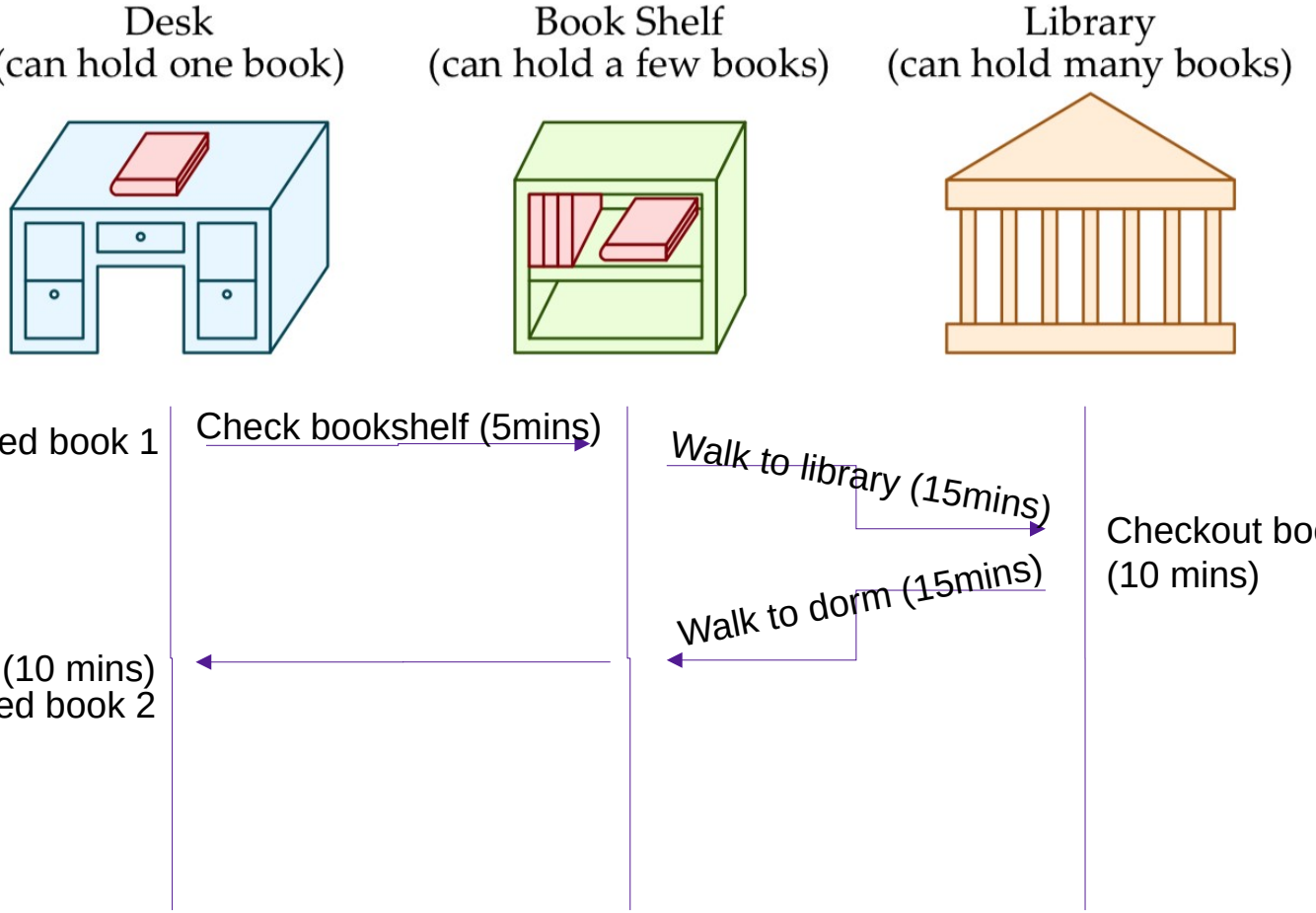


Library
(can hold many books)

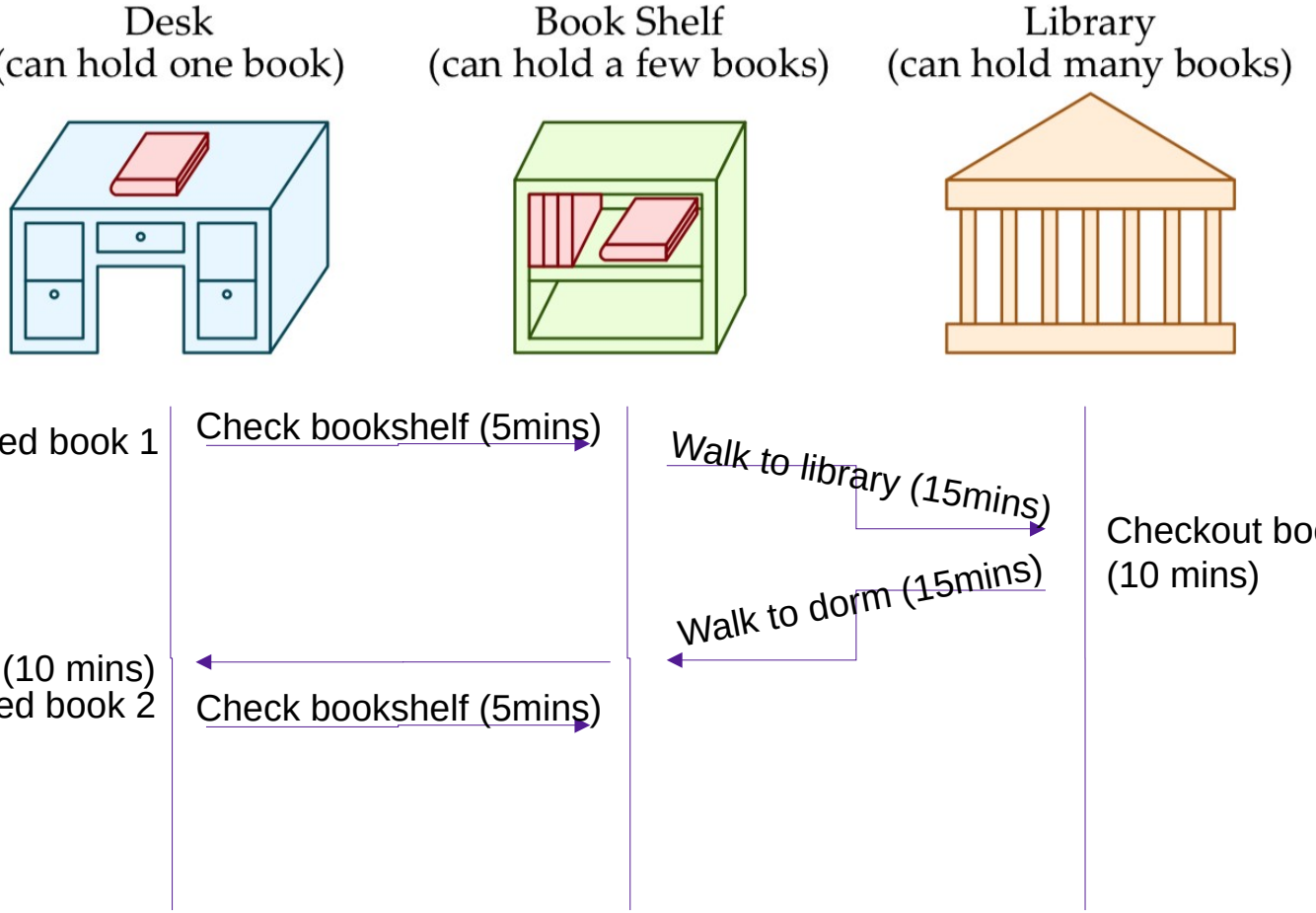




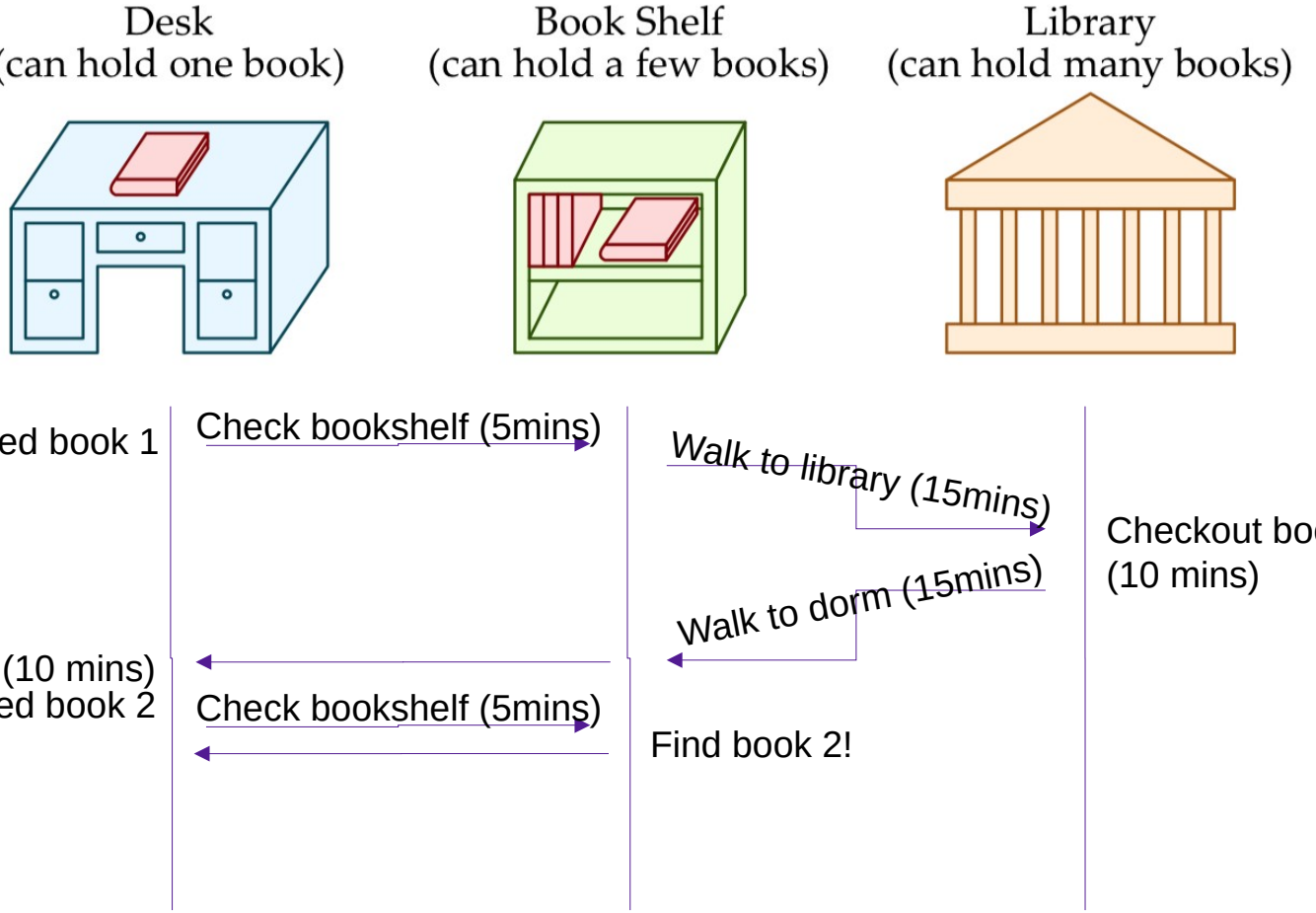
Life with caching



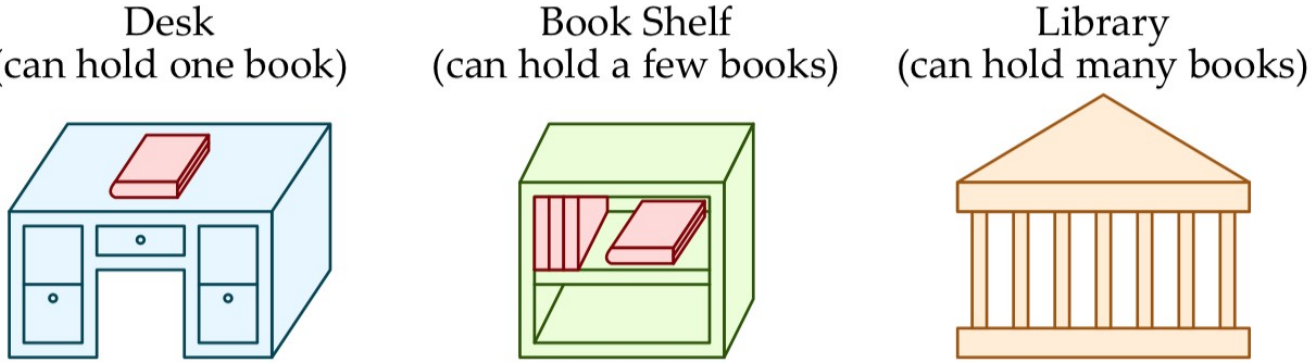
Life with caching



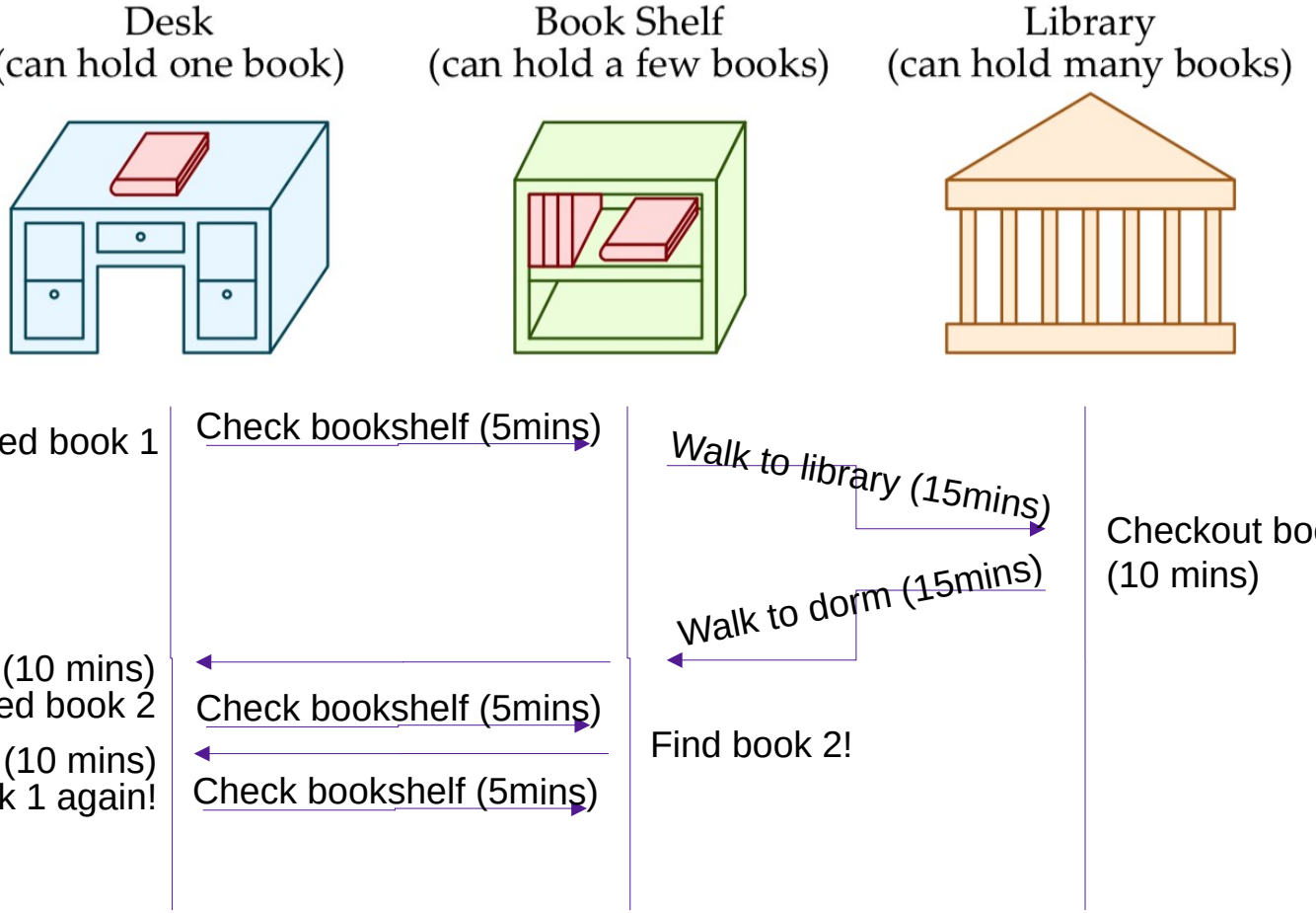
Life with caching



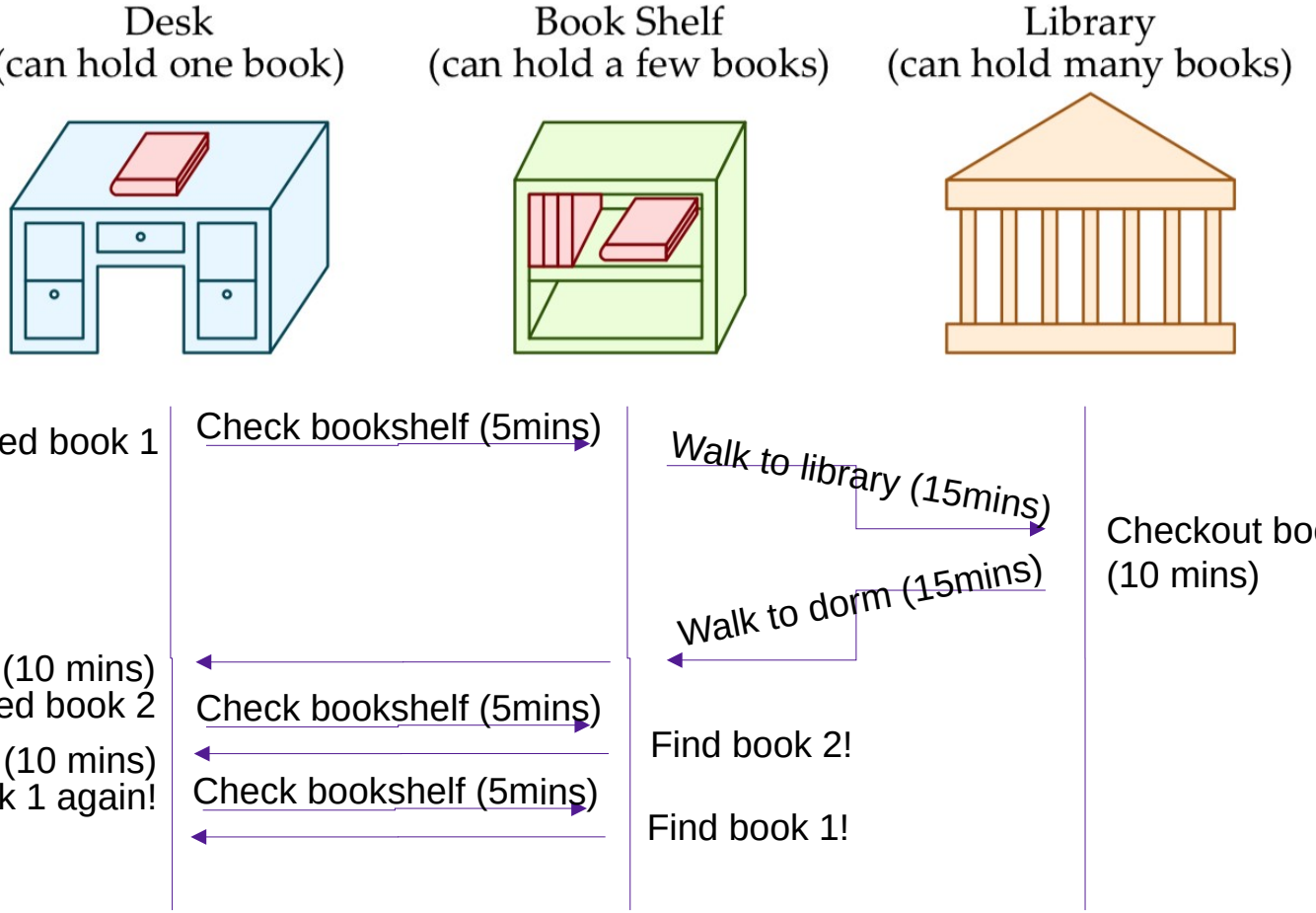
Life with caching



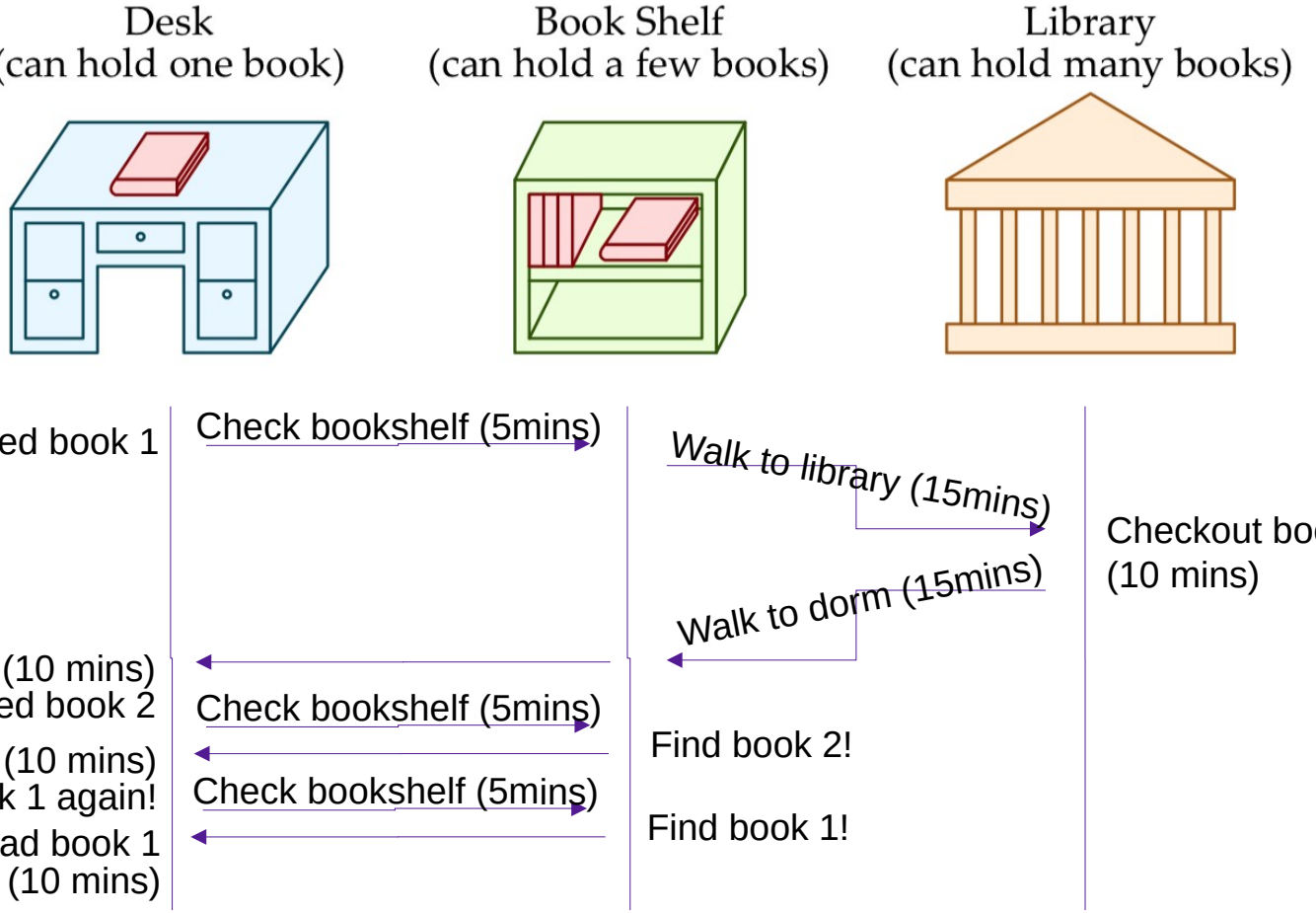
Life with caching



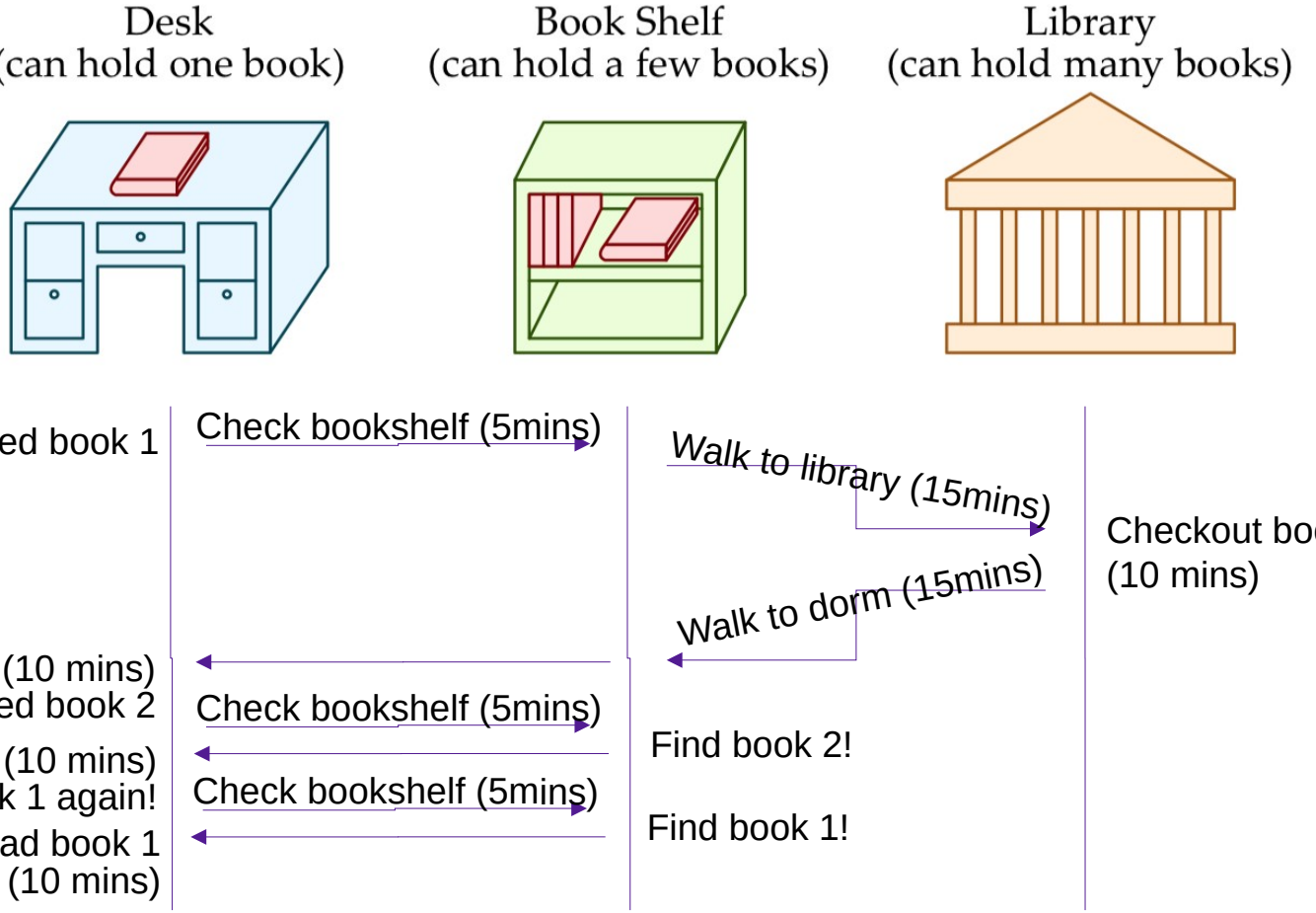
Life with caching



Life with caching

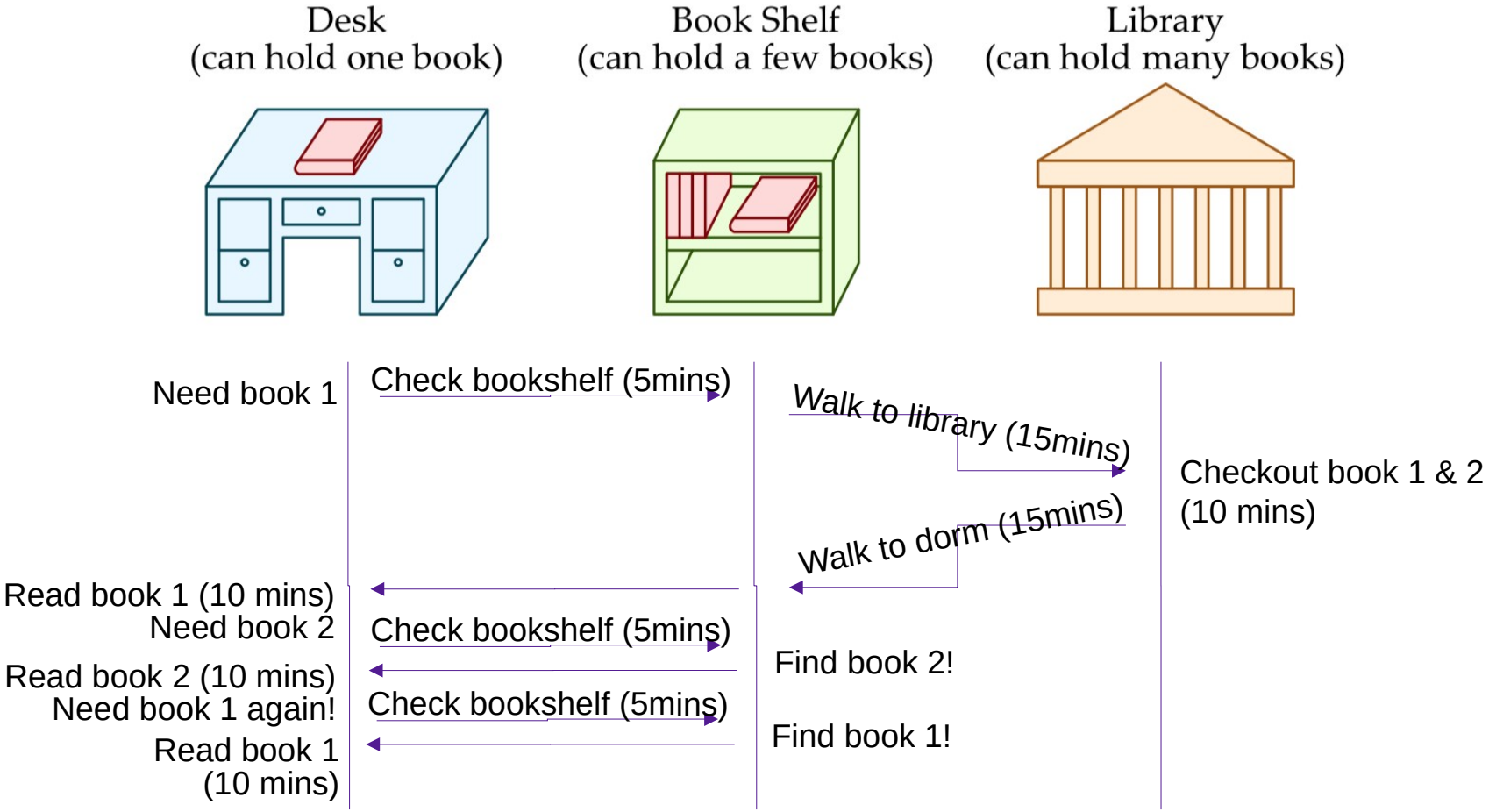


Life with caching



- Average latency to access a book: <20mins

Life with caching



- Average latency to access a book: <20mins
- Average throughput (incl. reading time): ~2 books/hr

Caching—The Vocabulary

- **Size:** the total number of bytes that can be stored in the cache

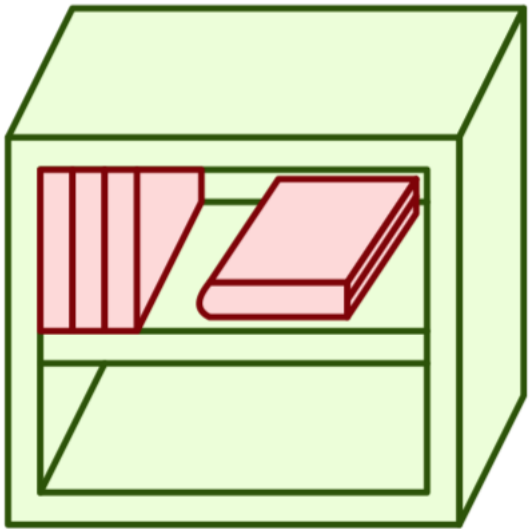
Caching—The Vocabulary

- **Size:** the total number of bytes that can be stored in the cache

- **Cache Hit:** the desired value is in the cache and returned quickly
- **Cache Miss:** the desired value is not in the cache and must be fetched from a more distant cache (or ultimately from main memory)

Exercise 1: Caching Strategies

How should we decide which books to keep in the bookshelf?



Example Access Patterns

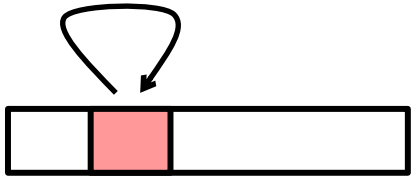
```
int sum = 0;
for (int i = 0; i < n; i++){
    sum += a[i];
}
return sum;
```

- Data references
  - Reference array elements in succession.
  - Reference variable **`sum`** each iteration.
- Instruction references
  - Reference instructions in sequence.
  - Cycle through loop repeatedly.

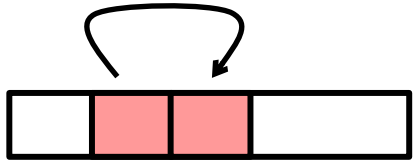
Principle of Locality

Programs tend to use data and instructions with addresses near or equal to those they have used recently

- Temporal locality:
  - Recently referenced items are likely to be referenced again in the near future

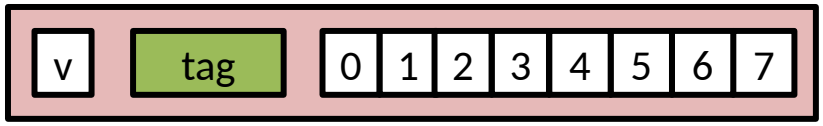


- Spatial locality:
  - Items with nearby addresses tend to be referenced close together in time

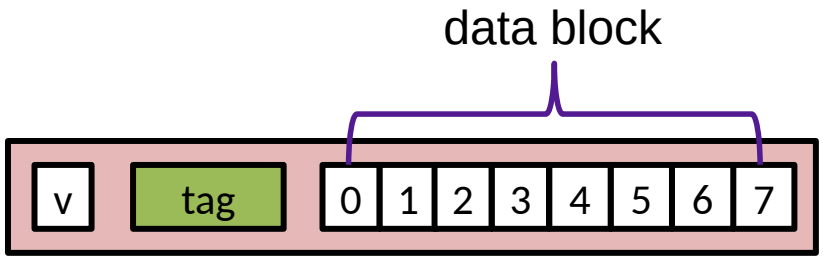


CACHE ORGANIZATION

Cache Lines

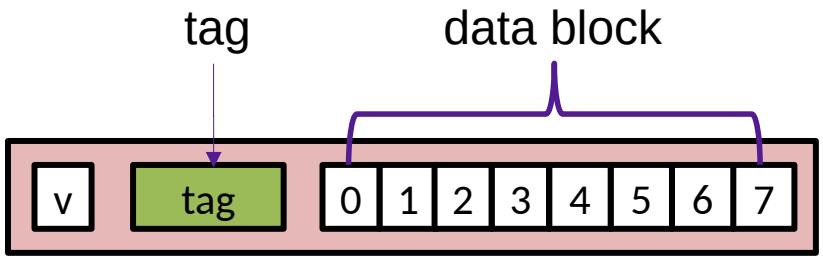


Cache Lines



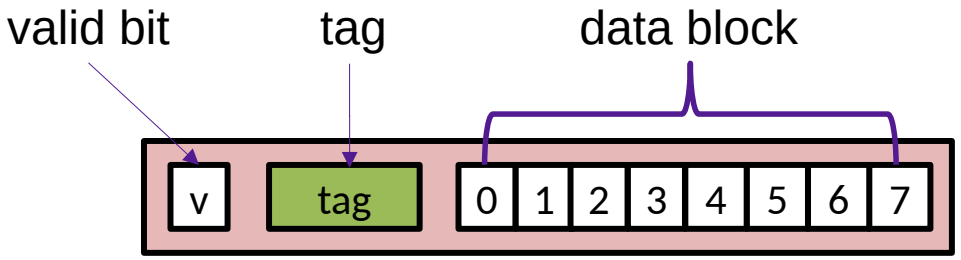
- **data block:** cached data (i.e., copy of bytes from memory)

Cache Lines



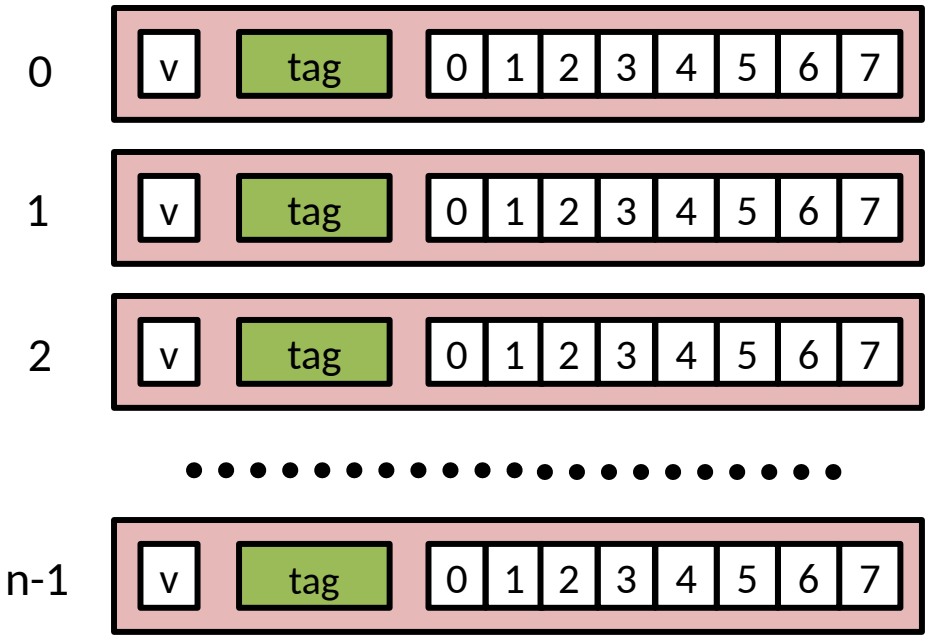
- **data block:** cached data (i.e., copy of bytes from memory)
- **tag:** uniquely identifies which data is stored in the cache line

Cache Lines



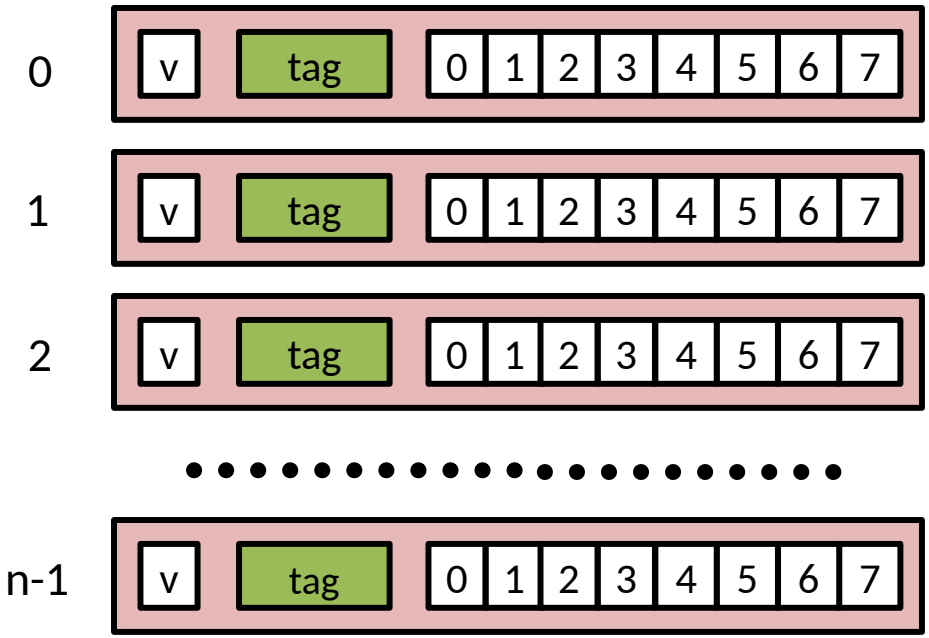
- **data block:** cached data (i.e., copy of bytes from memory)
- **tag:** uniquely identifies which data is stored in the cache line
- **valid bit:** indicates whether or not the line contains meaningful information

Direct-mapped Cache

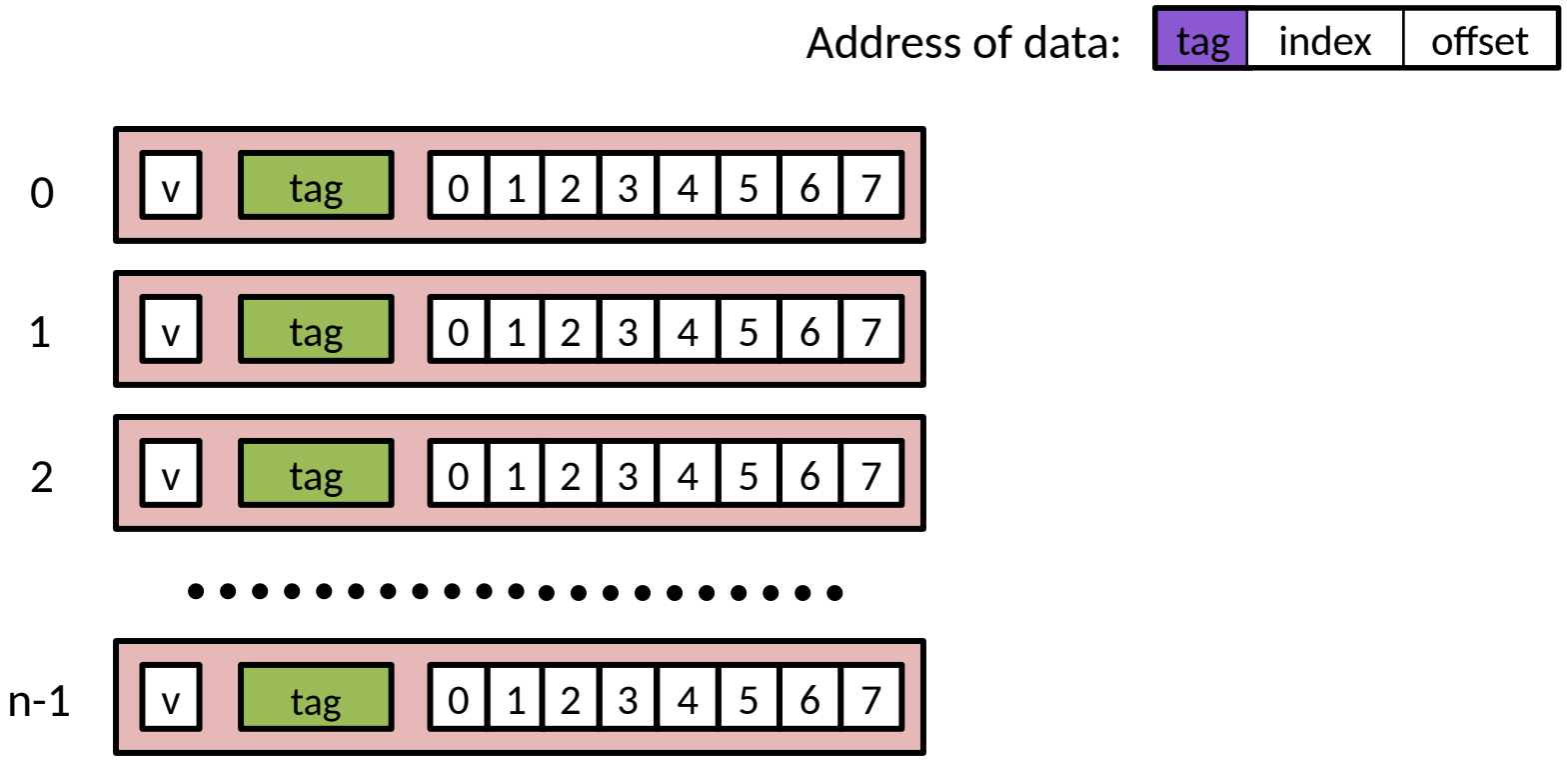


Direct-mapped Cache

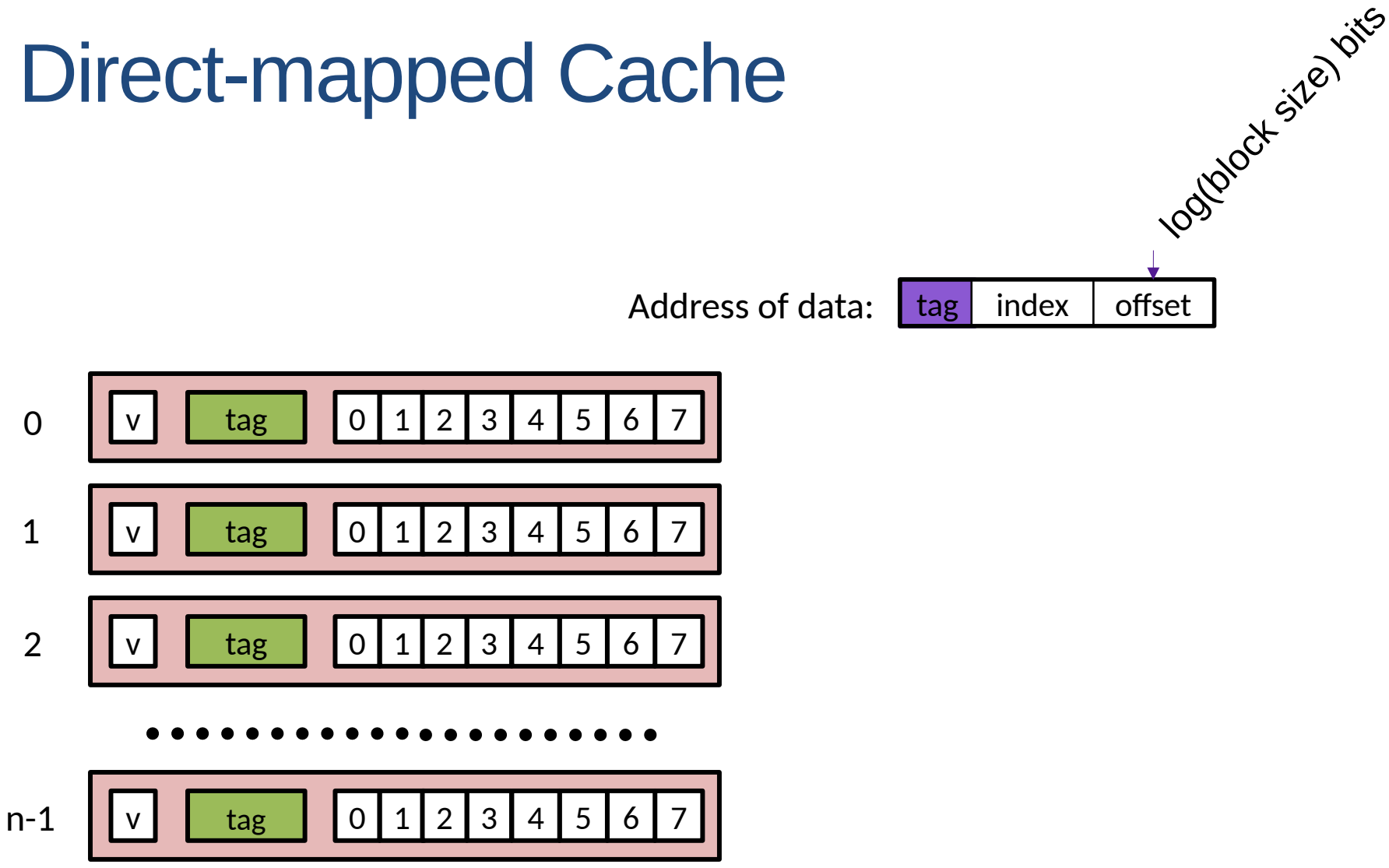
Address of data:



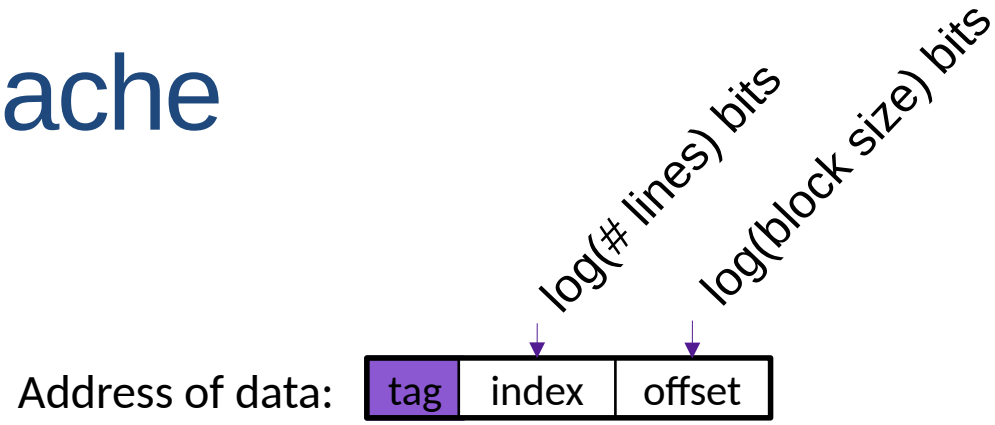
Direct-mapped Cache

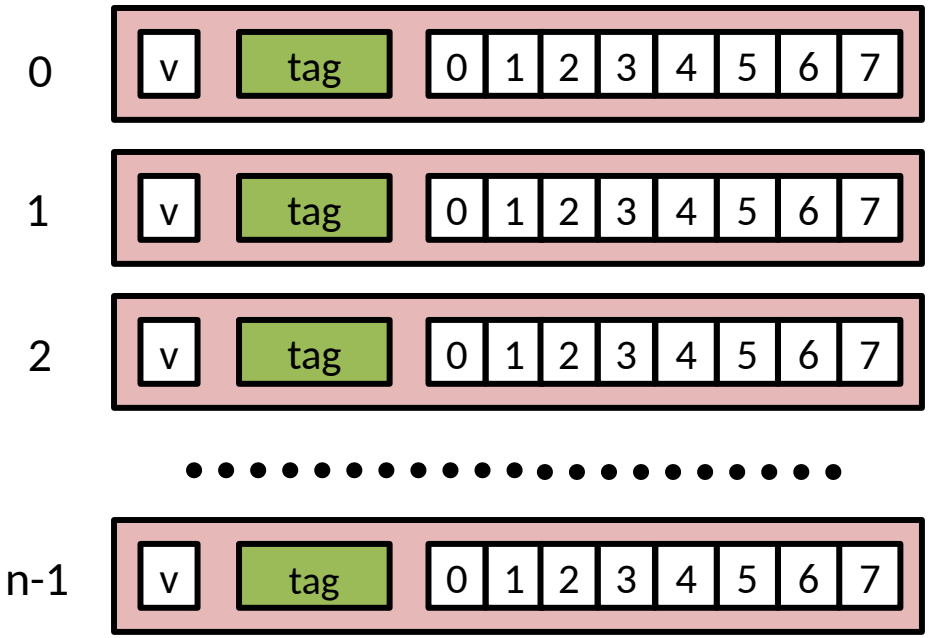


Direct-mapped Cache

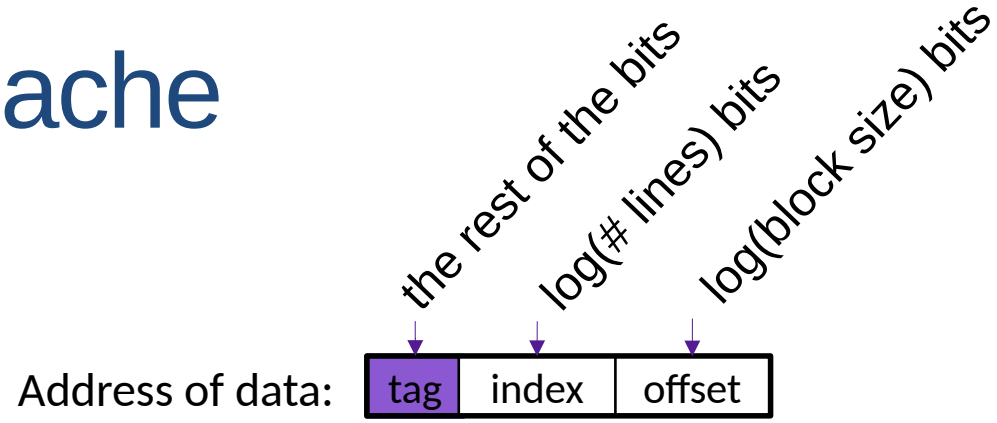


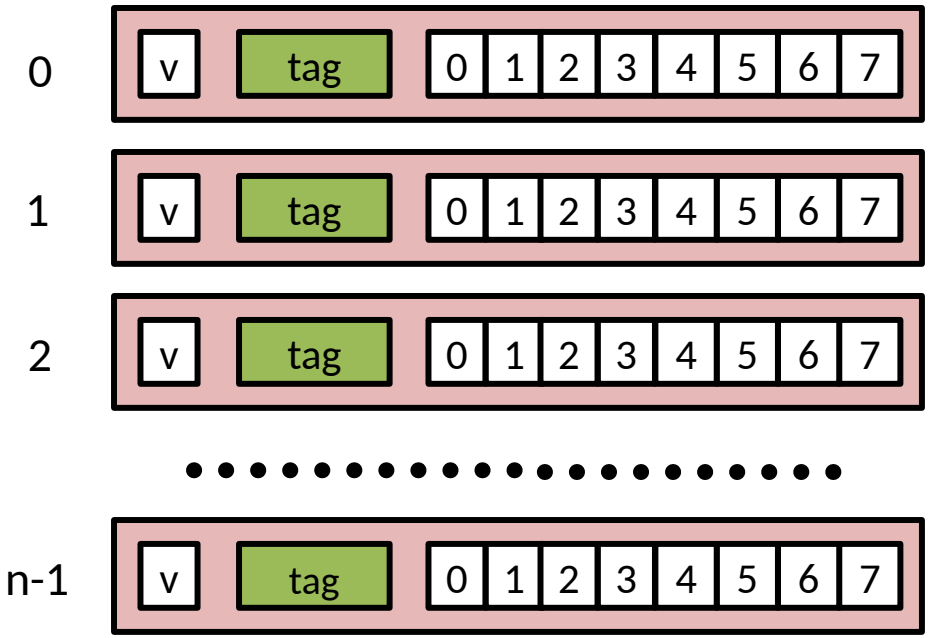
Direct-mapped Cache



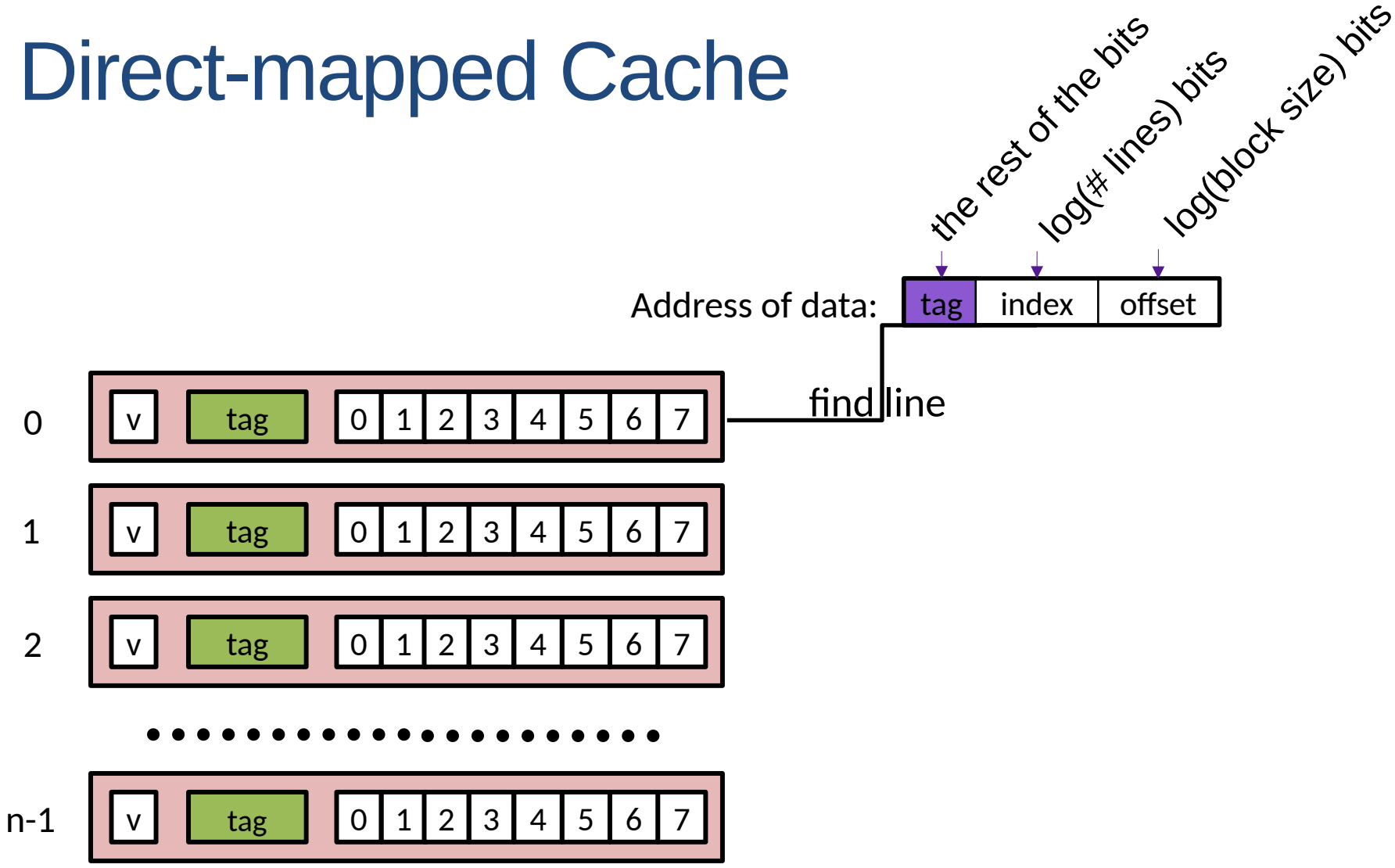


Direct-mapped Cache

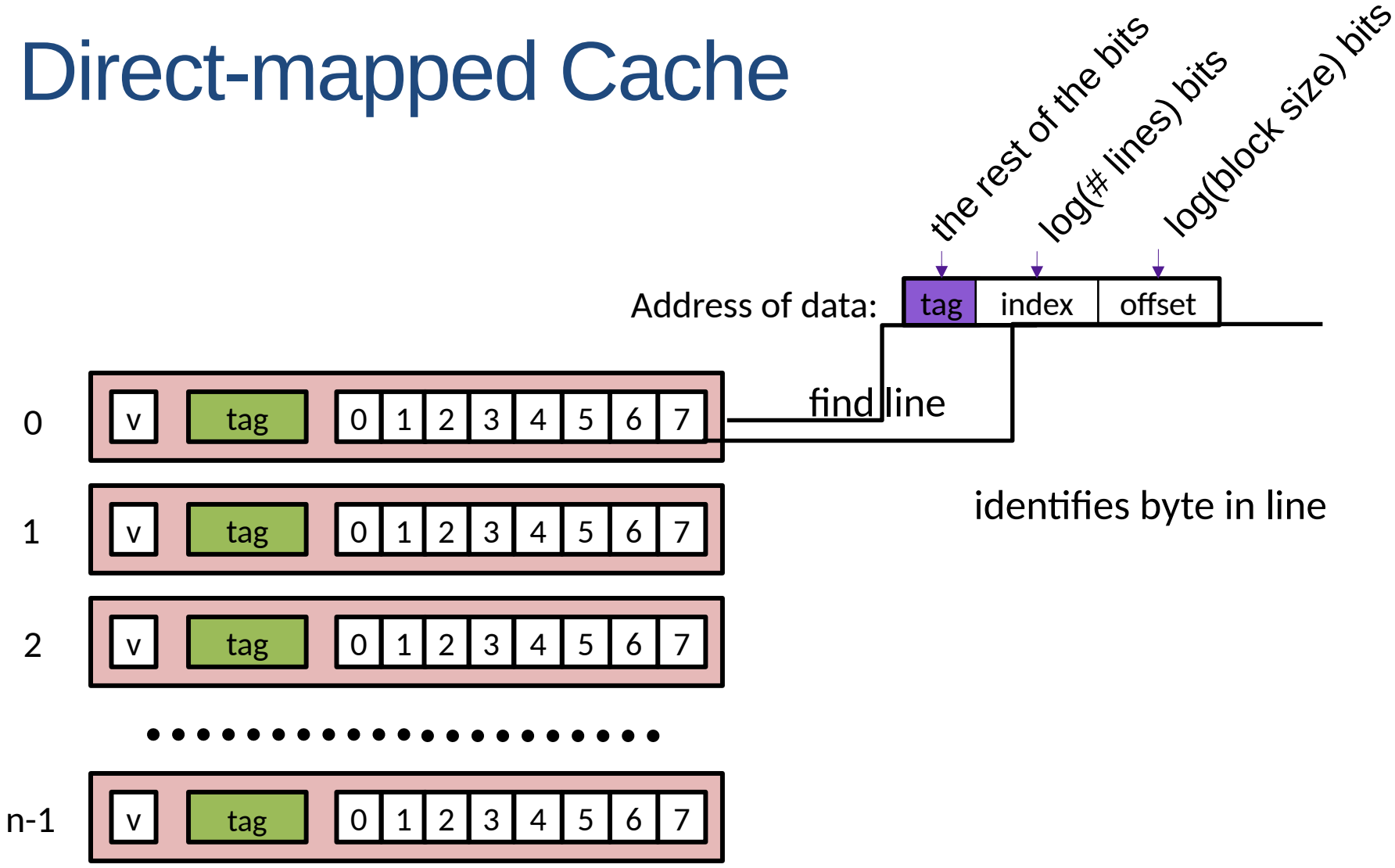




Direct-mapped Cache



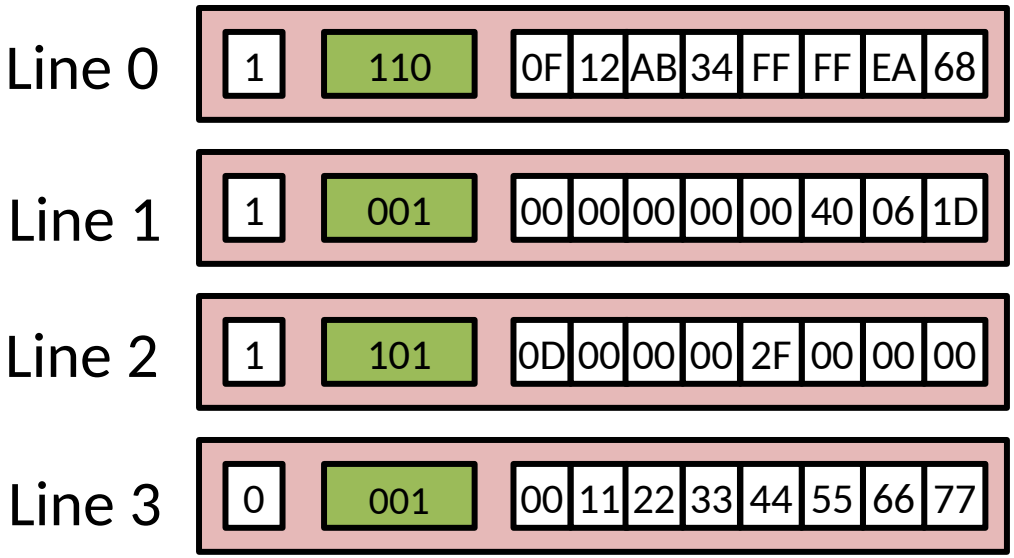
Direct-mapped Cache



Example: Direct-mapped Cache

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

Address of data: 0xB4

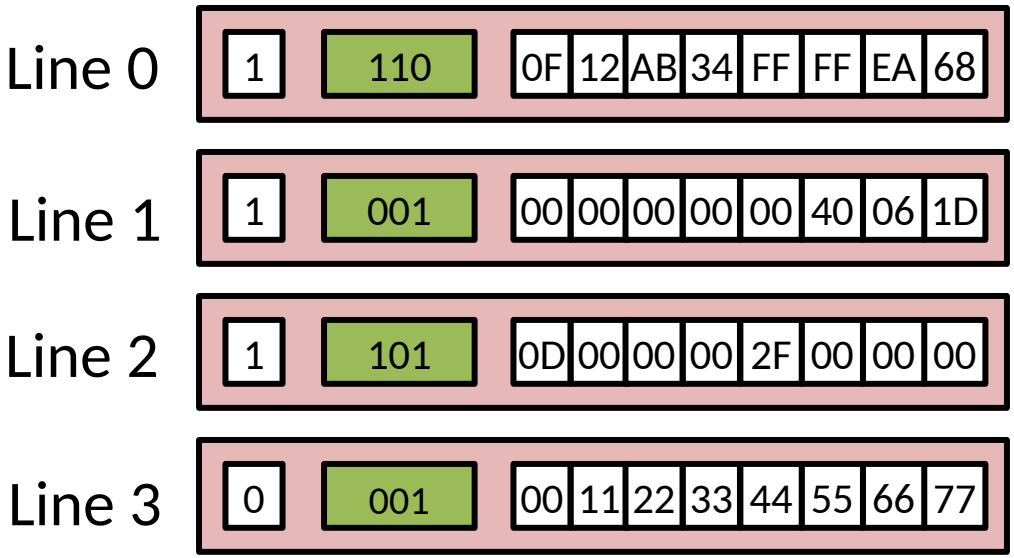


Example: Direct-mapped Cache

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

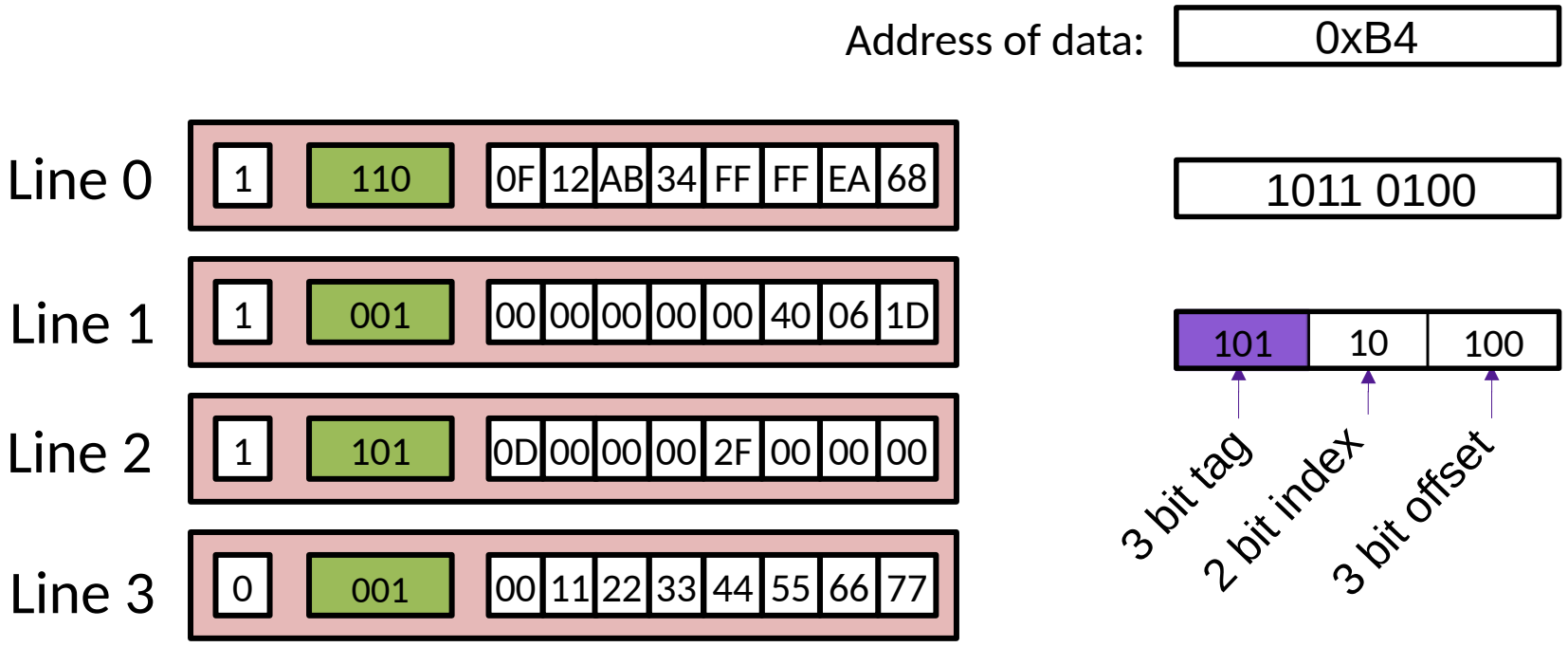
Address of data: 0xB4

1011 0100



Example: Direct-mapped Cache

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine



Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations?

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

3. A direct-mapped cache with 16 cache lines and 8-byte data

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations?

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

   offset = 01 = 0x1

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

   index = 0110 = 0x6 offset = 01 = 0x1

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

   tag = 1010 01 = 0x29 index = 0110 = 0x6 offset = 01 = 0x1

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

   tag = 1010 01 = 0x29 index = 0110 = 0x6 offset = 01 = 0x1

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

   offset = 001 = 0x1

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

   tag = 1010 01 = 0x29 index = 0110 = 0x6 offset = 01 = 0x1

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

   index = 1011 = 0xB offset = 001 = 0x1

Exercise 2: Interpreting Addresses

Consider the 12-bit address 0xA59. What would be the tag, index, and offset for this address with each of the following cache configurations? **0xA59 = 1010 0101 1001**

1. A direct-mapped cache with 8 cache lines and 8-byte data blocks

   tag = 1010 01 = 0x29 index = 011 = 0x3 offset = 001 = 0x1

2. A direct-mapped cache with 16 cache lines and 4-byte data blocks

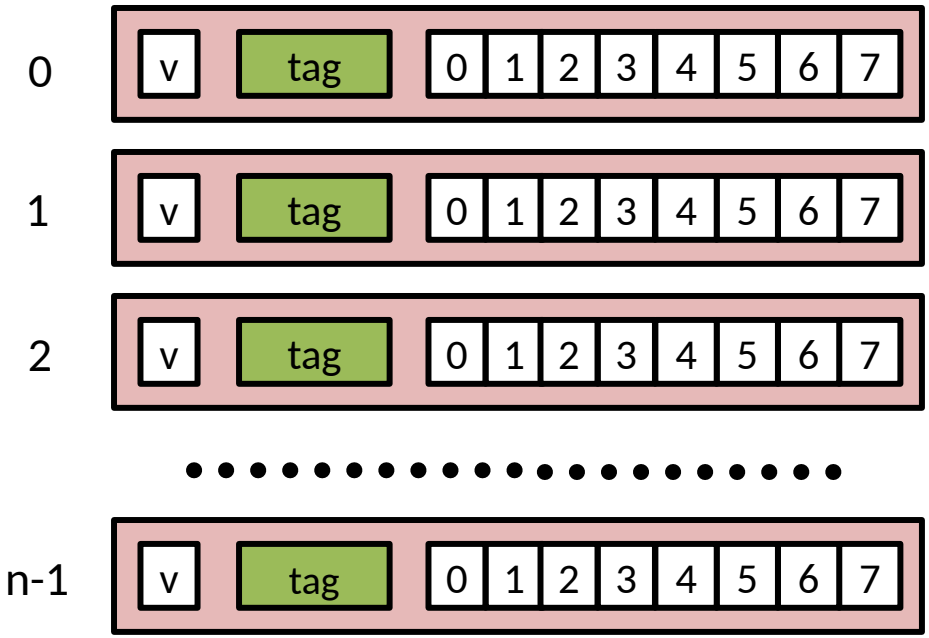
   tag = 1010 01 = 0x29 index = 0110 = 0x6 offset = 01 = 0x1

3. A direct-mapped cache with 16 cache lines and 8-byte data blocks

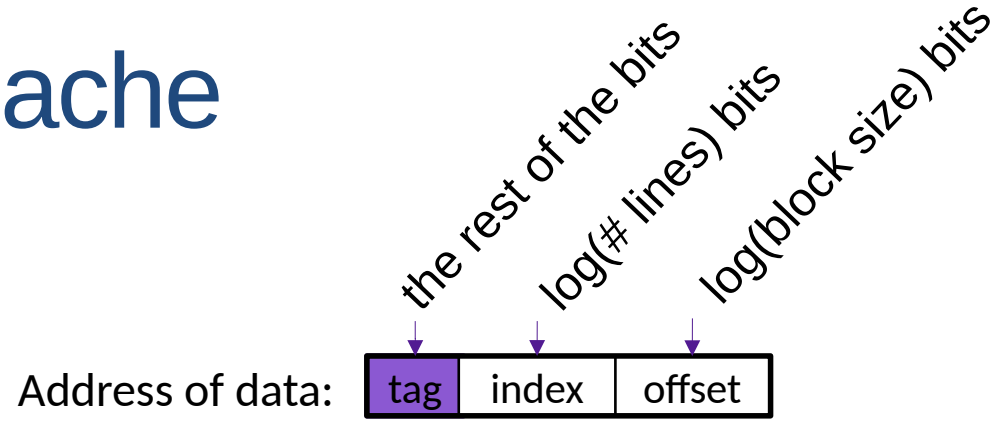
   tag = 1010 0 = 0x14 index = 1011 = 0xB offset = 001 = 0x1

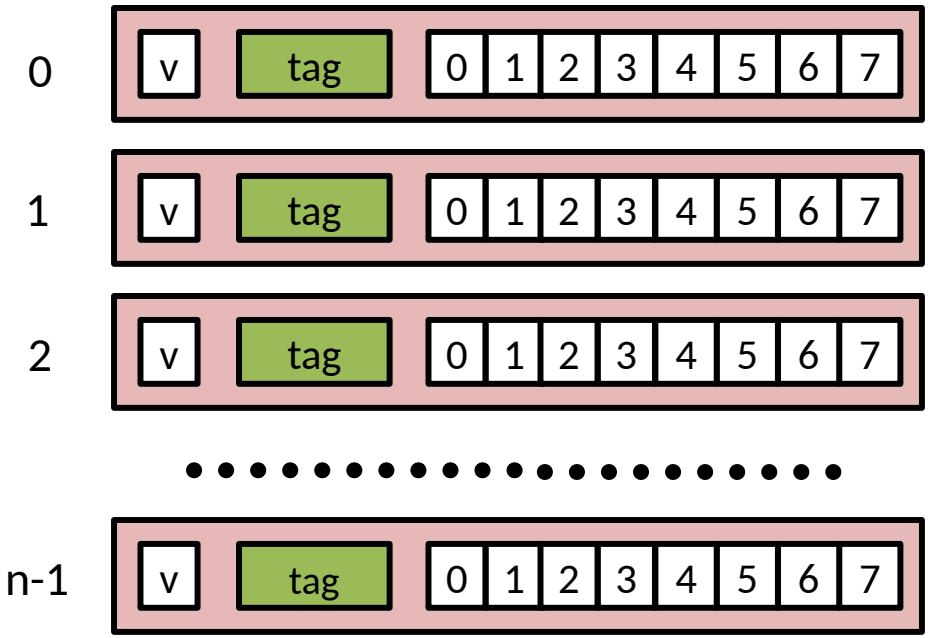
Direct-mapped Cache

Address of data:

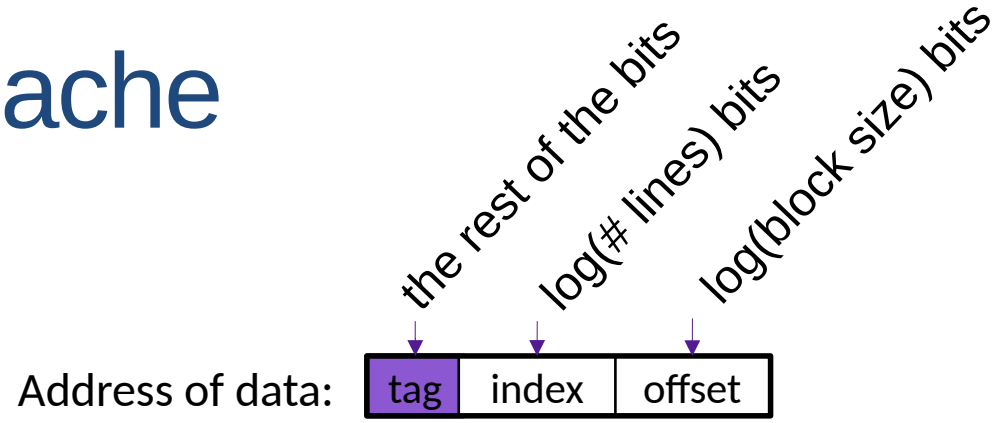


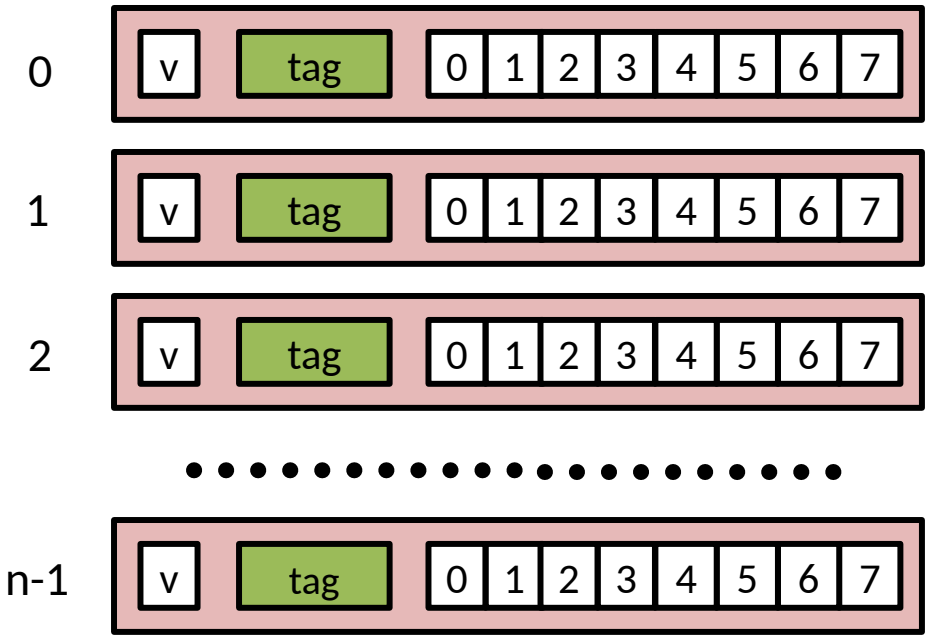
Direct-mapped Cache





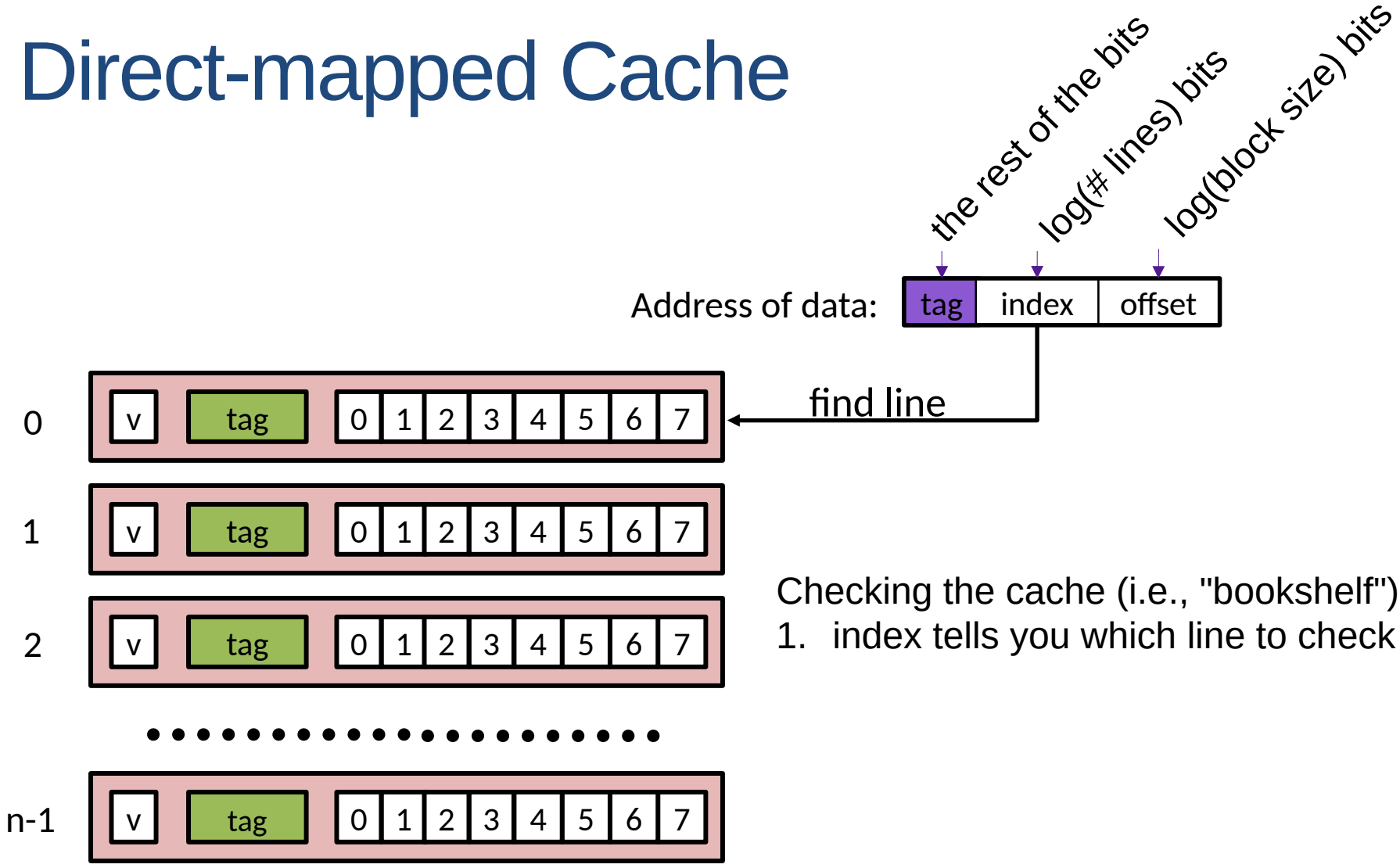
Direct-mapped Cache





Checking the cache (i.e., "bookshelf"):

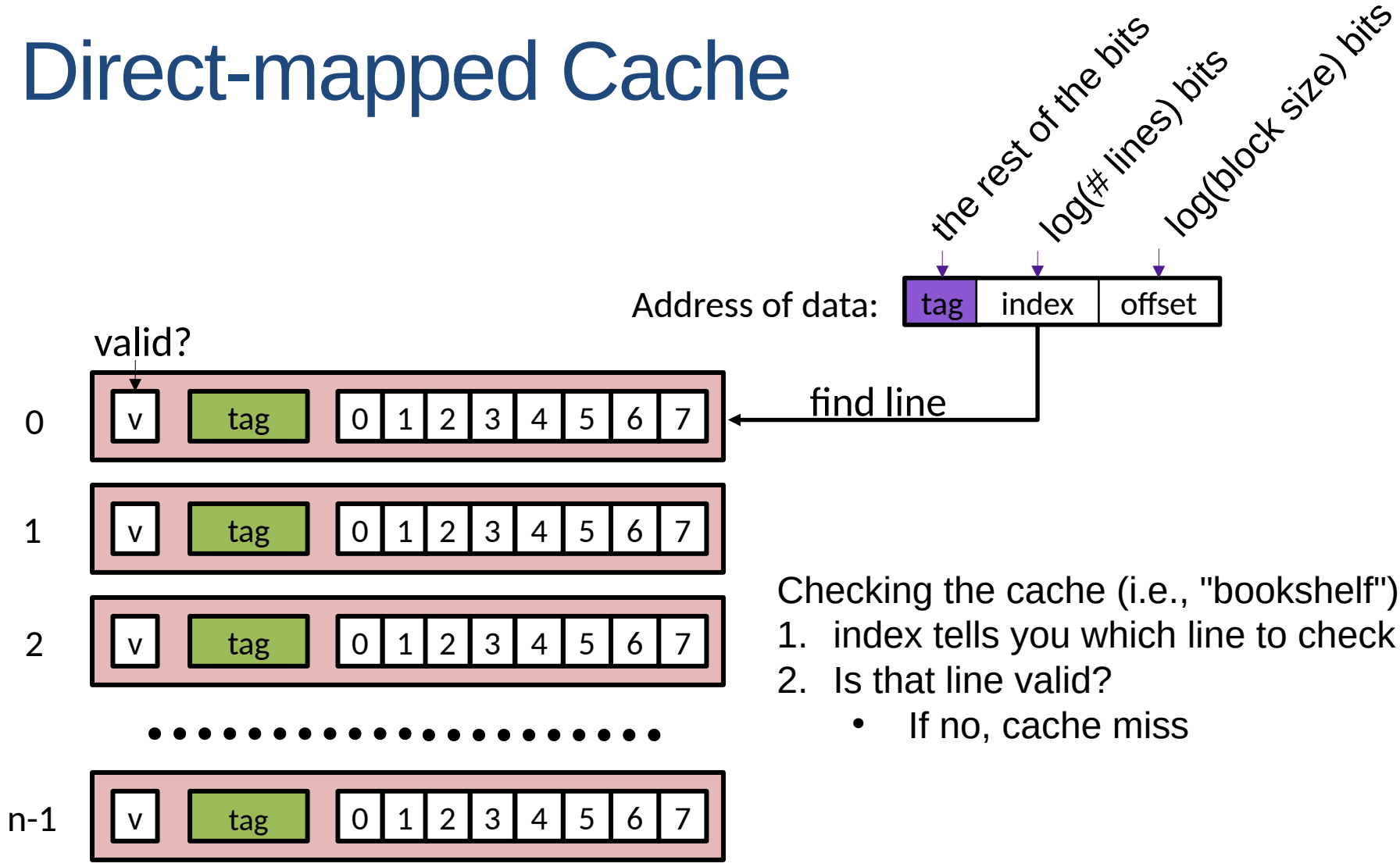
Direct-mapped Cache



Checking the cache (i.e., "bookshelf"):

1. index tells you which line to check

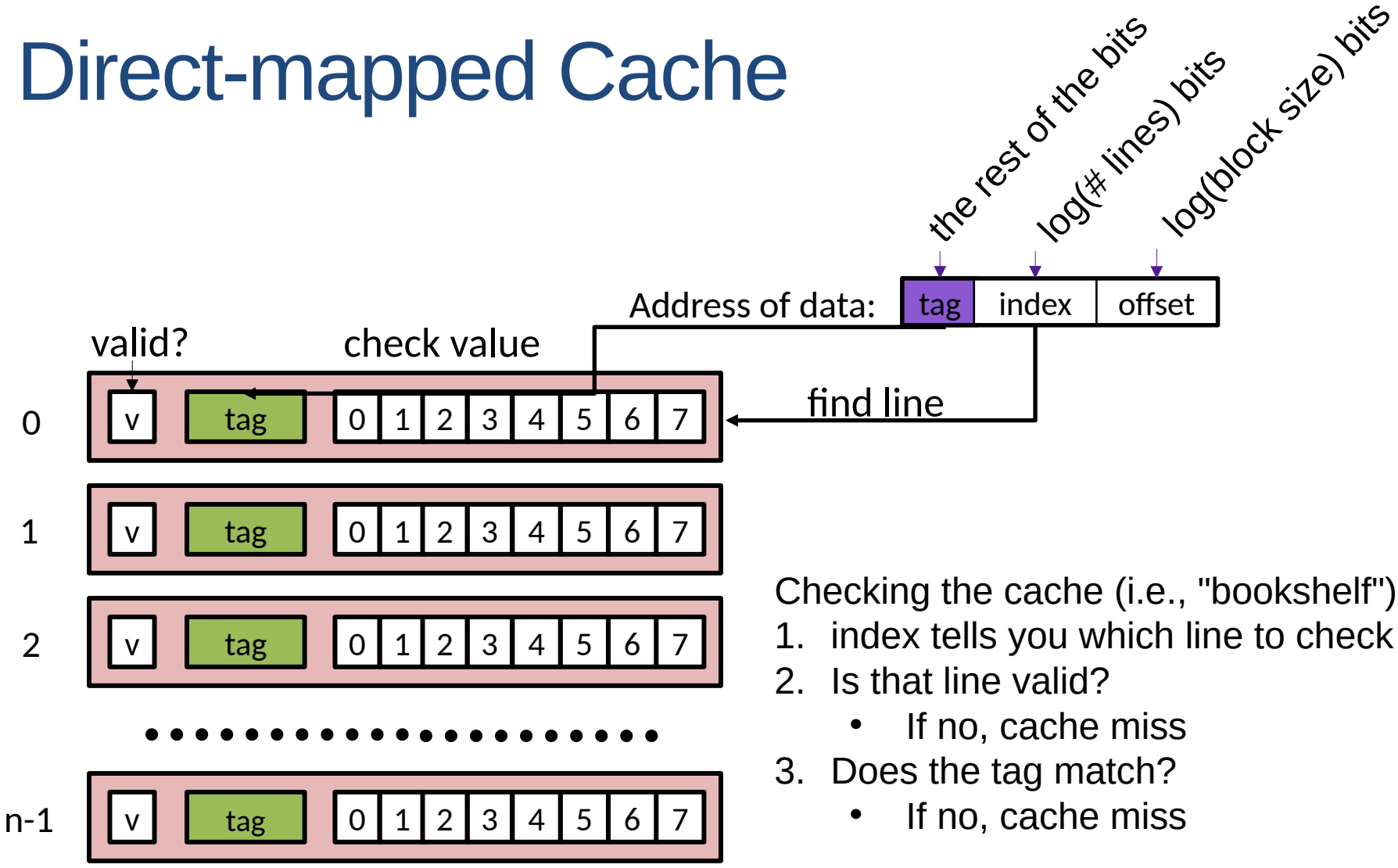
Direct-mapped Cache



Checking the cache (i.e., "bookshelf"):

1. index tells you which line to check
2. Is that line valid?
   - If no, cache miss

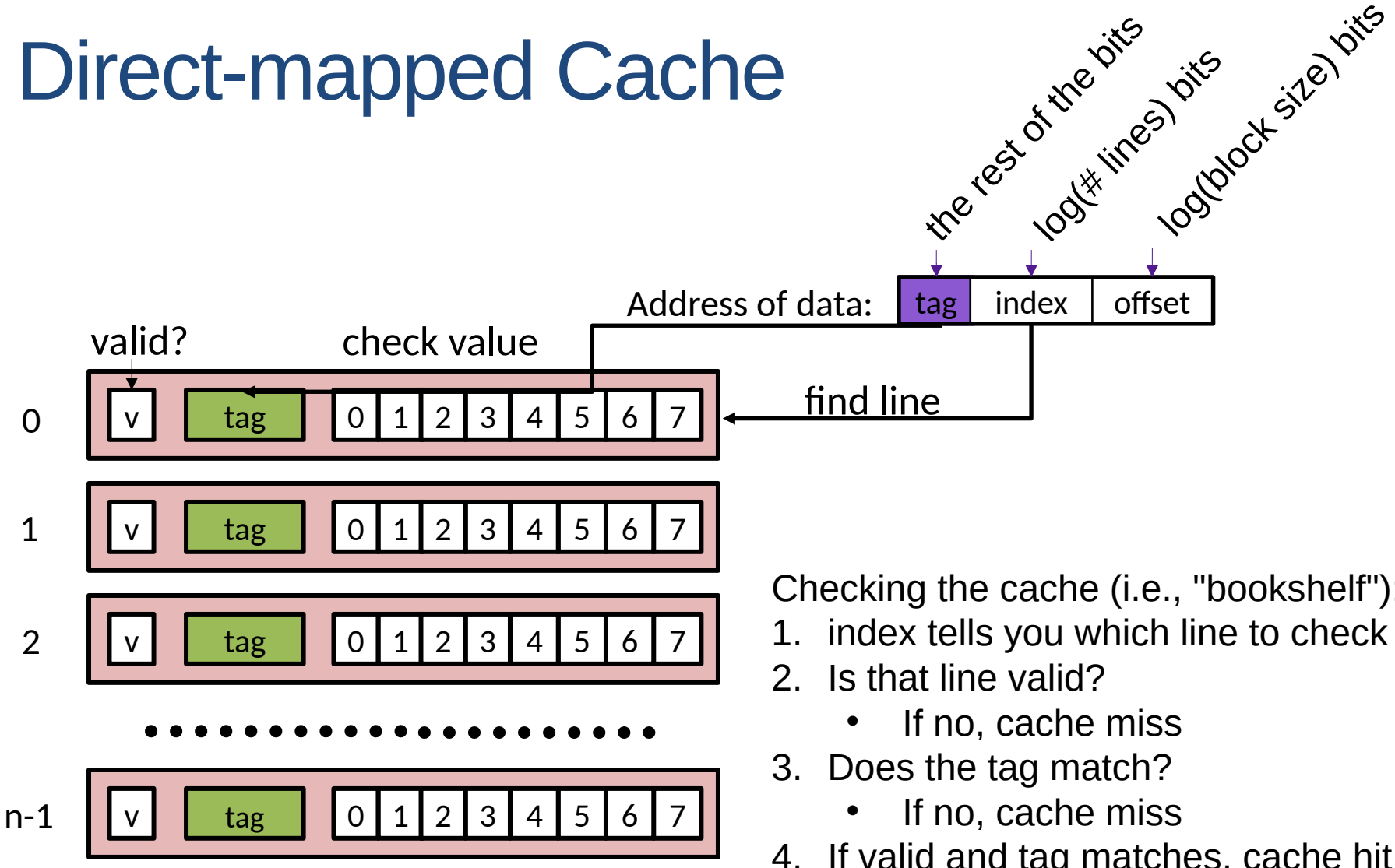
Direct-mapped Cache



Checking the cache (i.e., "bookshelf"):

1. index tells you which line to check
2. Is that line valid?
   - If no, cache miss
3. Does the tag match?
   - If no, cache miss

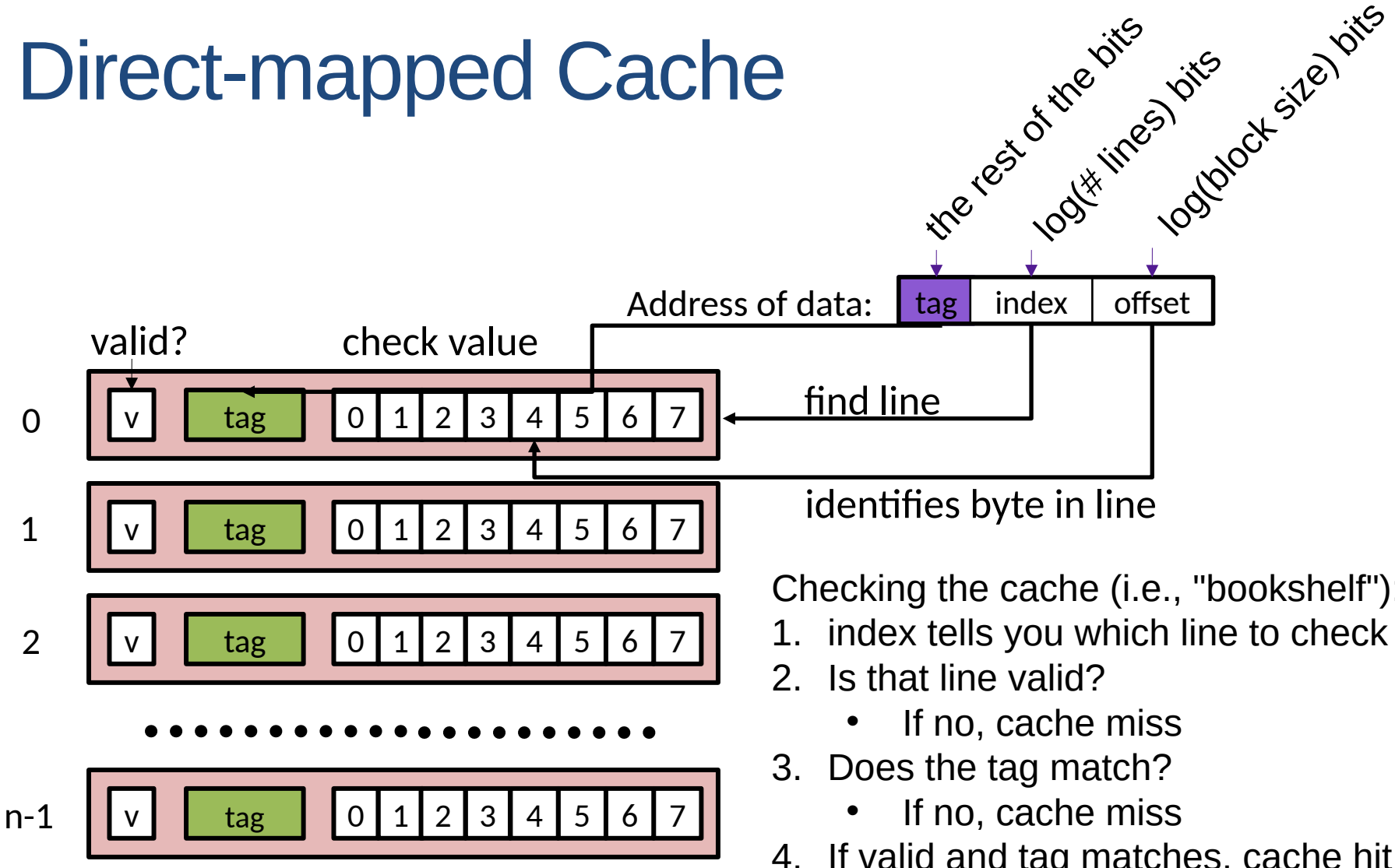
Direct-mapped Cache



Checking the cache (i.e., "bookshelf"):

1. index tells you which line to check
2. Is that line valid?
   - If no, cache miss
3. Does the tag match?
   - If no, cache miss
4. If valid and tag matches, cache hit!

Direct-mapped Cache

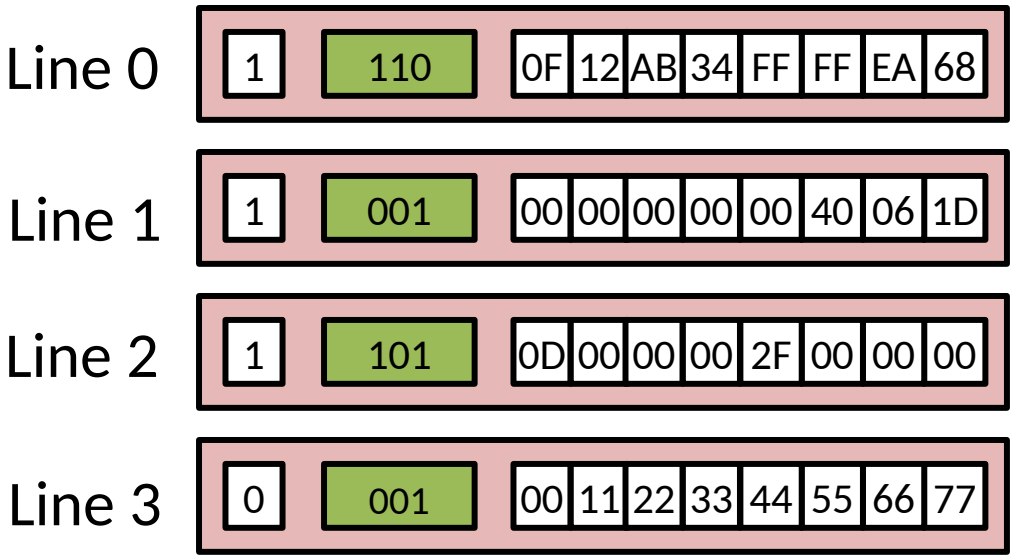


Checking the cache (i.e., "bookshelf"):

1. index tells you which line to check
2. Is that line valid?
   - If no, cache miss
3. Does the tag match?
   - If no, cache miss
4. If valid and tag matches, cache hit!
   - Read from datablock at offset

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

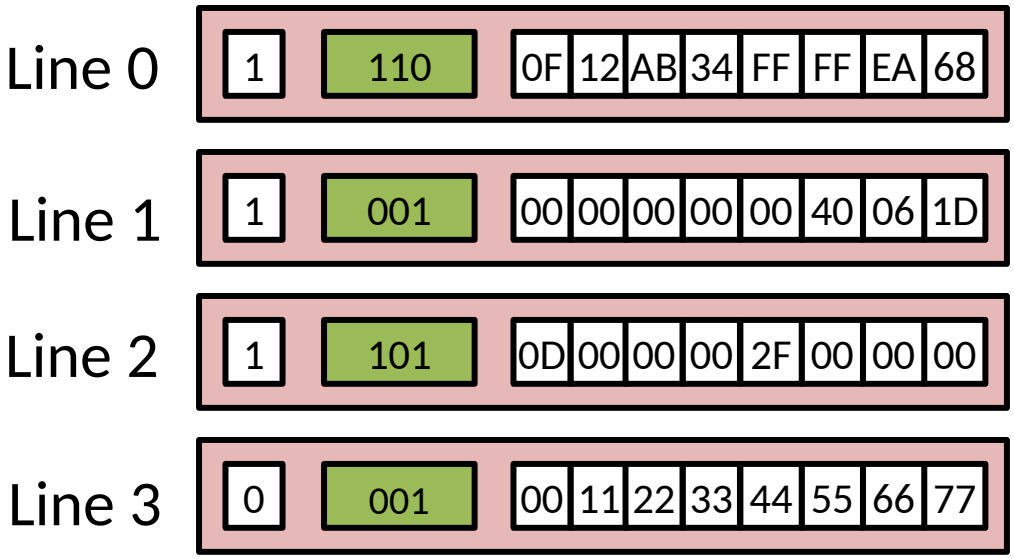


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

- 0x2D
- 0x2E
- 0x74
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

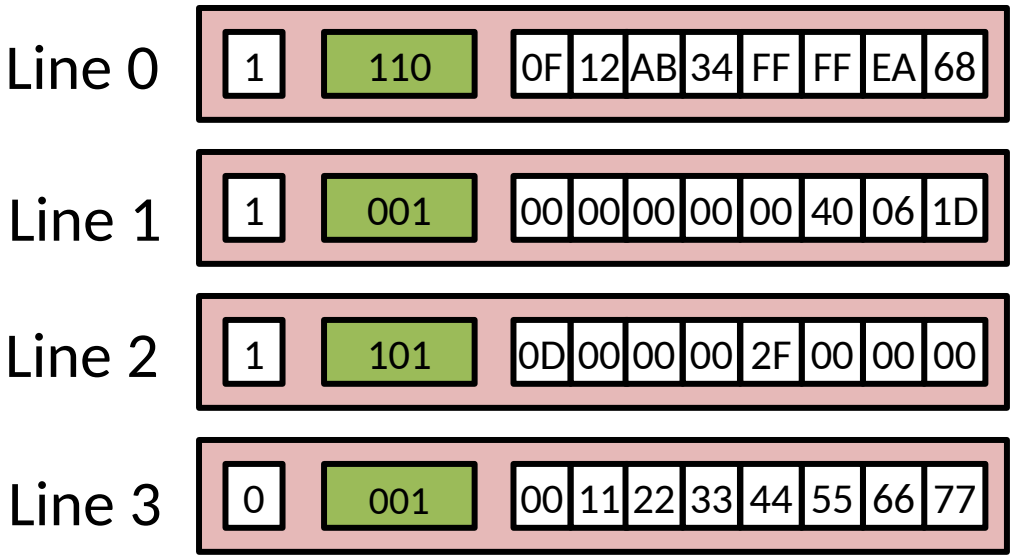


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

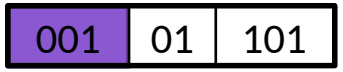
- 0x2D
- 0x2E
- 0x74
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine



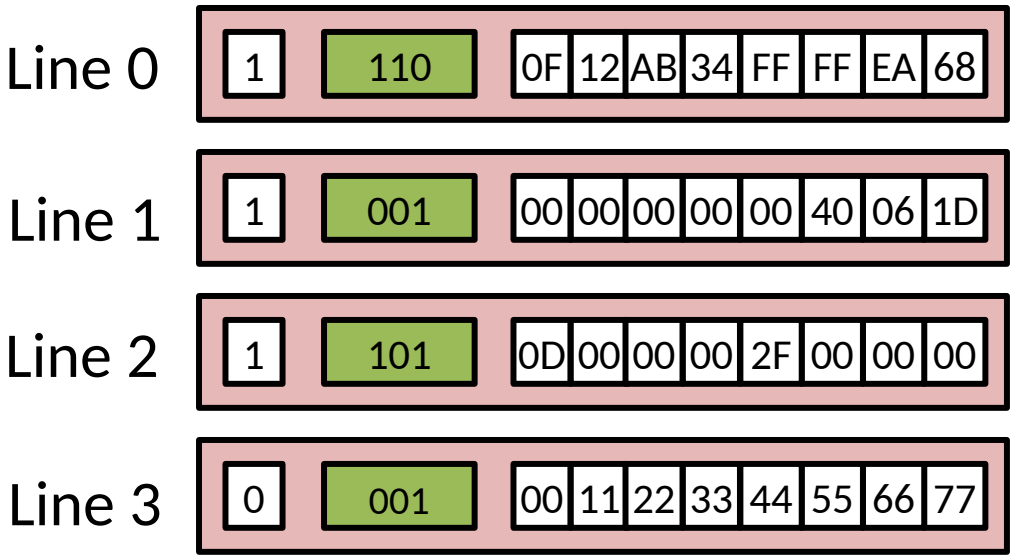
For each address, is it a hit or a miss? For hits, what data is at that address in memory?

- 0x2D 

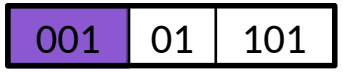
- 0x2E
- 0x74
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

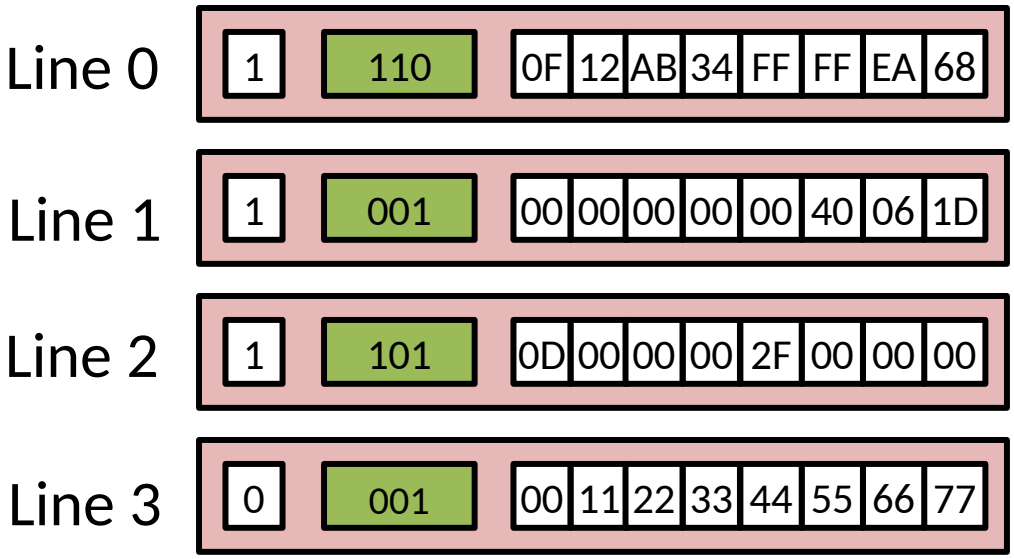


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

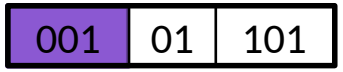
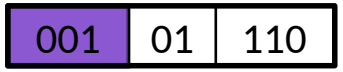
- 0x2D  **Hit 0x40**
- 0x2E
- 0x74
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

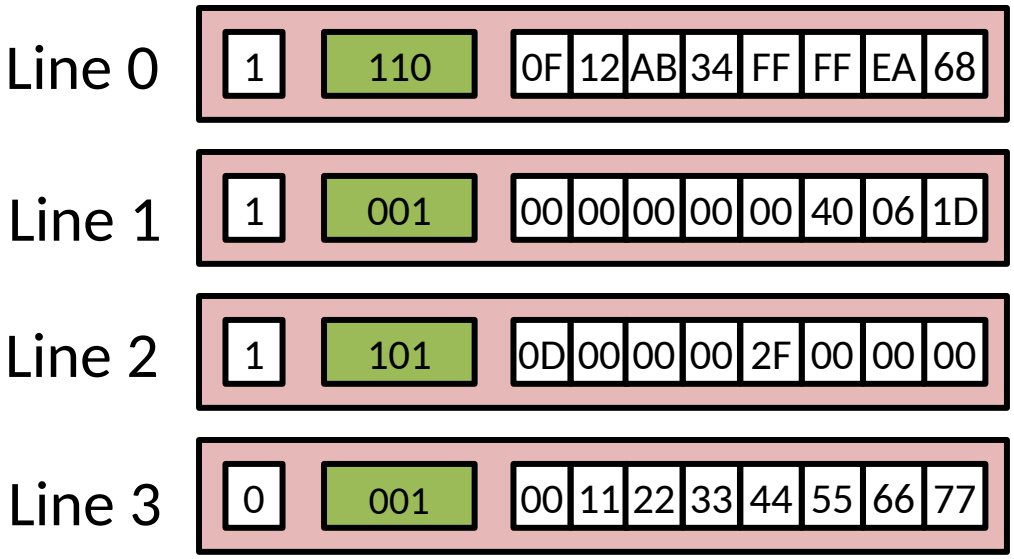


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

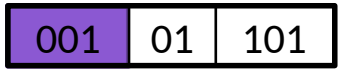
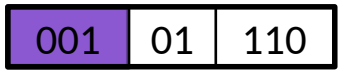
- 0x2D  **Hit 0x40**
- 0x2E 
- 0x74
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

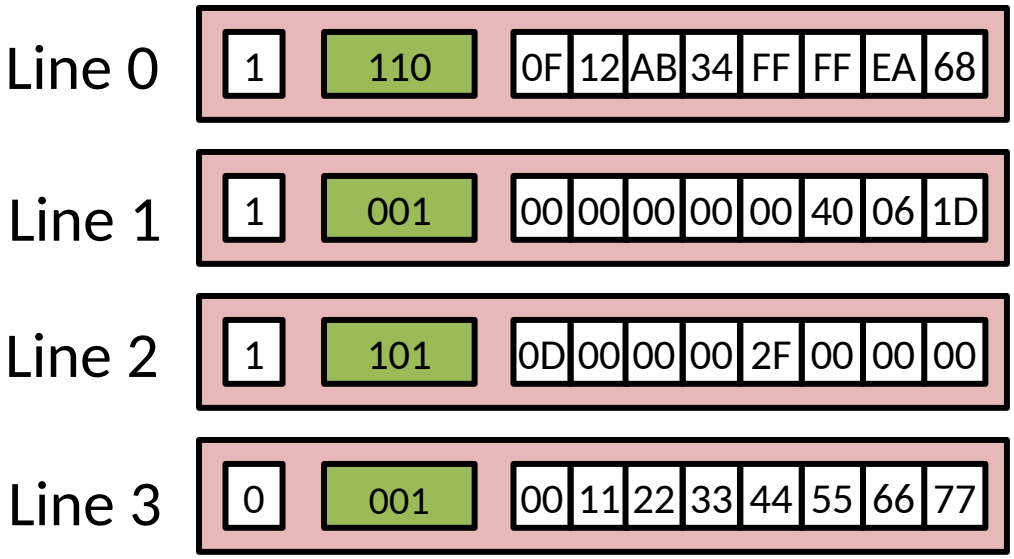


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

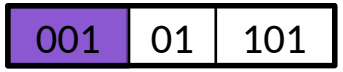
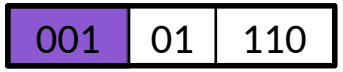
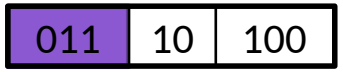
- 0x2D  **Hit 0x40**
- 0x2E  **Hit 0x06**
- 0x74
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

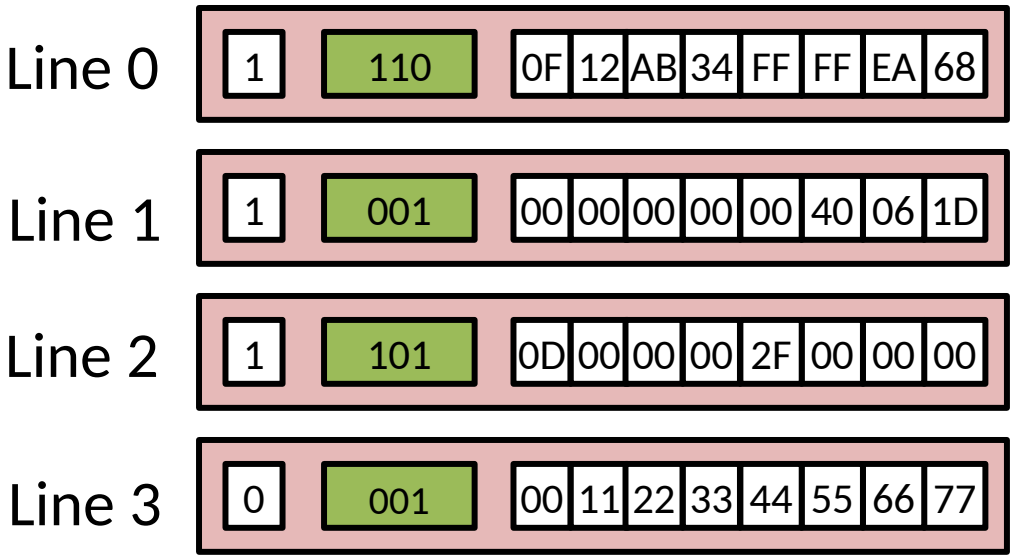


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

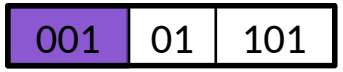
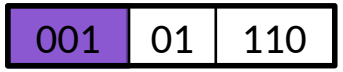
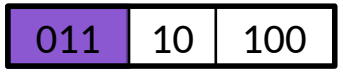
- 0x2D  **Hit 0x40**
- 0x2E  **Hit 0x06**
- 0x74 
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

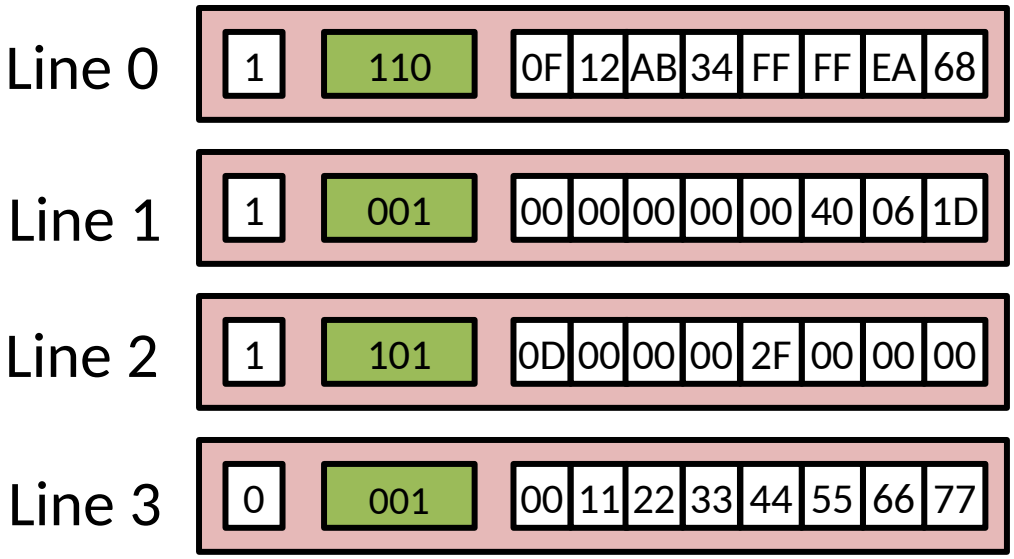


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

- 0x2D  **Hit 0x40**
- 0x2E  **Hit 0x06**
- 0x74  **Miss**
- 0x3A

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine

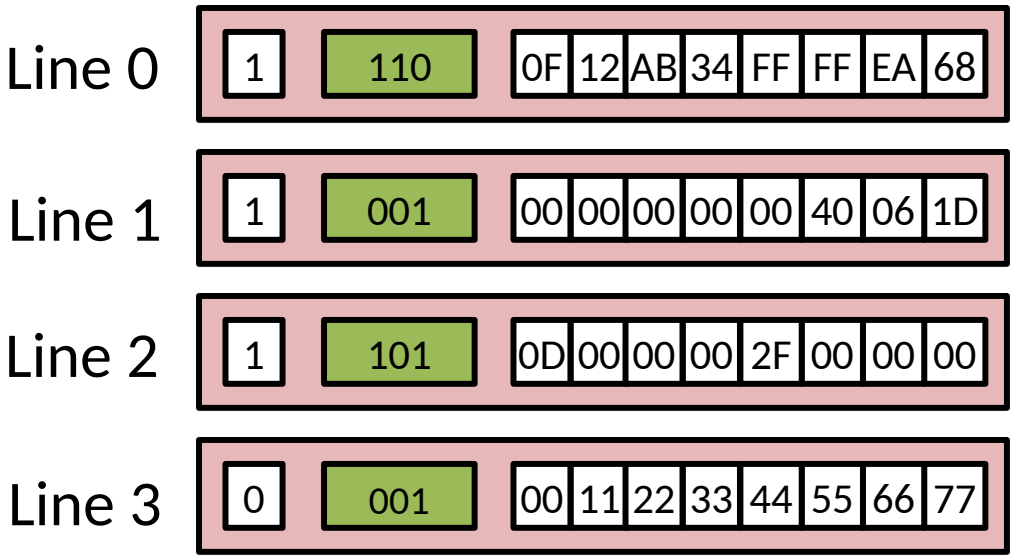


For each address, is it a hit or a miss? For hits, what data is at that address in memory?

- 0x2D | 001 | 01 | 101 | **Hit 0x40**
- 0x2E | 001 | 01 | 110 | **Hit 0x06**
- 0x74 | 011 | 10 | 100 | **Miss**
- 0x3A | 001 | 11 | 010 |

Exercise 3: Cache Hits and Misses

Assume: cache block size 8 bytes, total cache size 32 bytes
Assume: assume 8-bit machine



For each address, is it a hit or a miss? For hits, what data is at that address in memory?

- 0x2D | 001 | 01 | 101 | **Hit 0x40**
- 0x2E | 001 | 01 | 110 | **Hit 0x06**
- 0x74 | 011 | 10 | 100 | **Miss**
- 0x3A | 001 | 11 | 010 | **Miss**

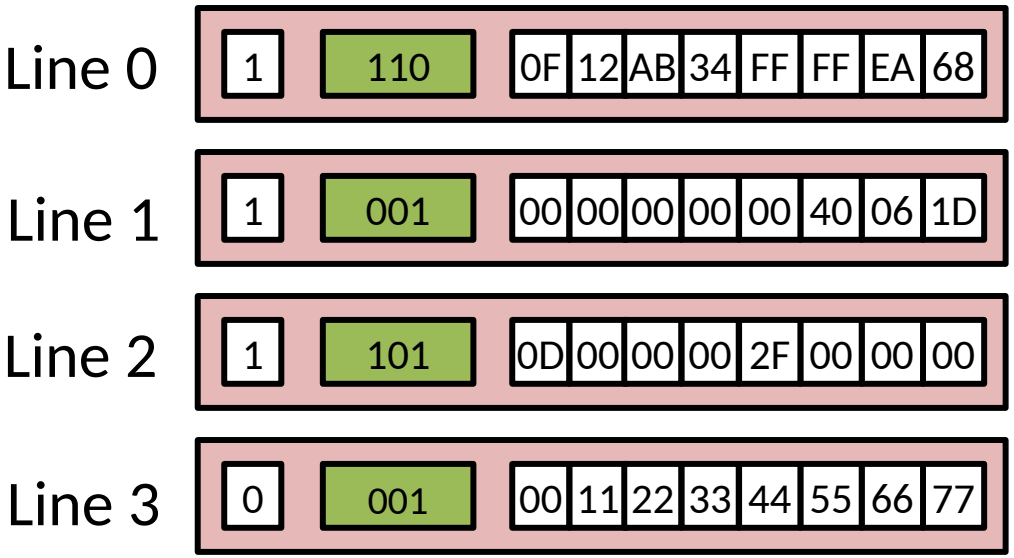
Handling a Cache Miss

When a cache miss occurs update cache line at that index:

Address of data: 0x74

Handling a Cache Miss

When a cache miss occurs update cache line at that index:

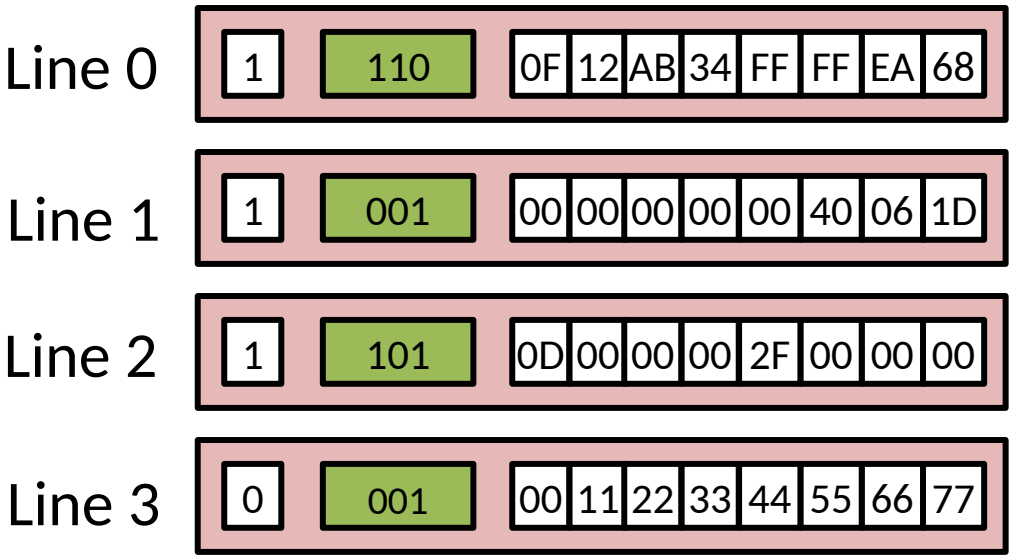


Handling a Cache Miss

Address of data: 0x74

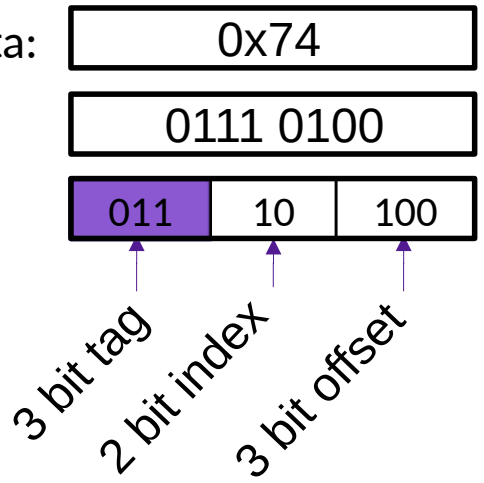
0111 0100

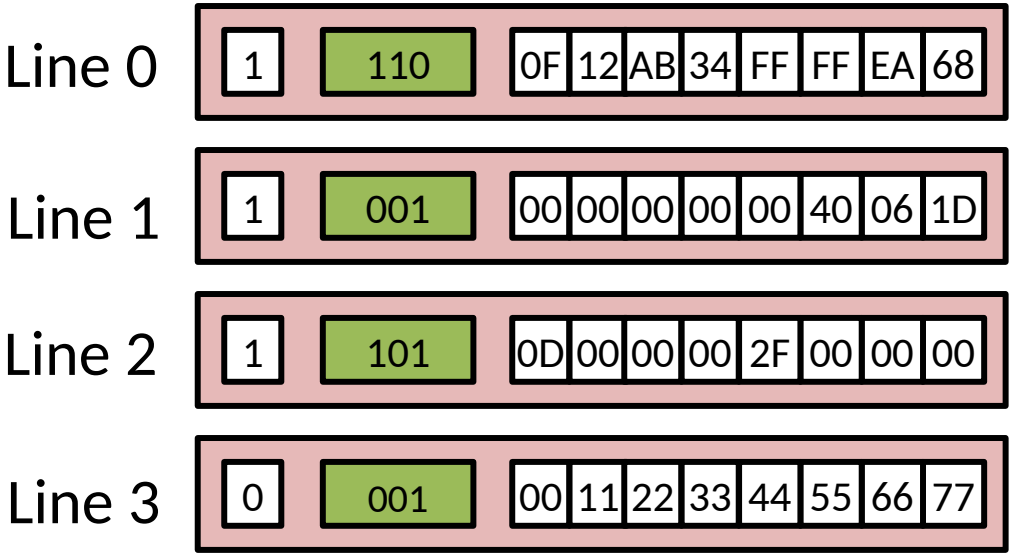
When a cache miss occurs update cache line at that index:



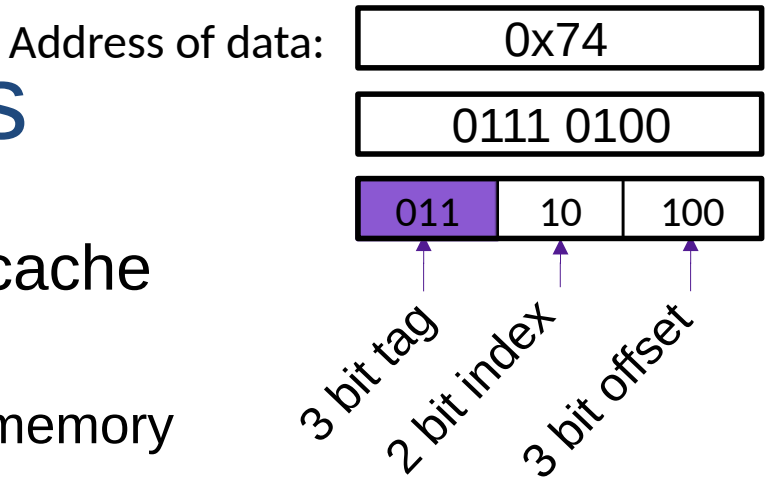
Handling a Cache Miss

When a cache miss occurs update cache line at that index:



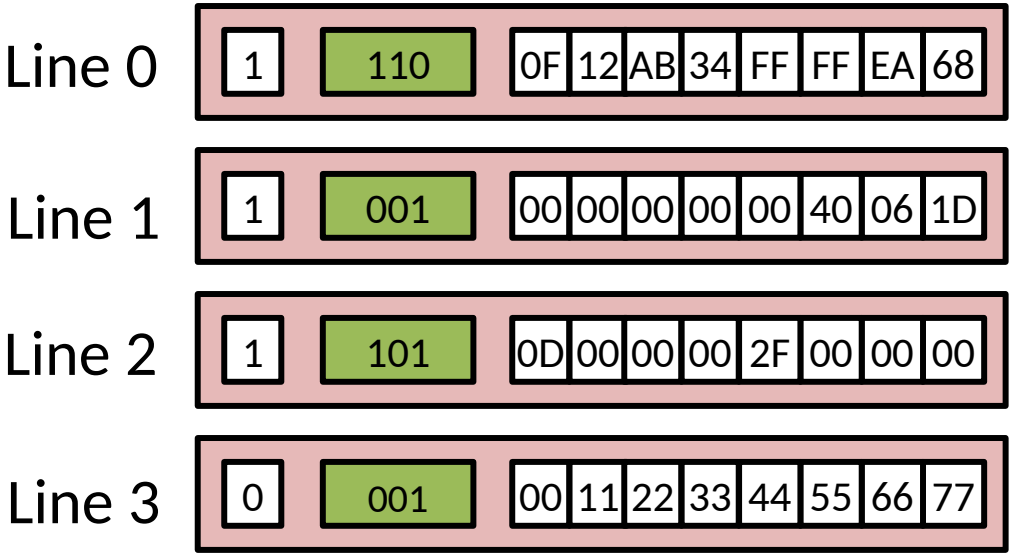


Handling a Cache Miss



When a cache miss occurs update cache line at that index:

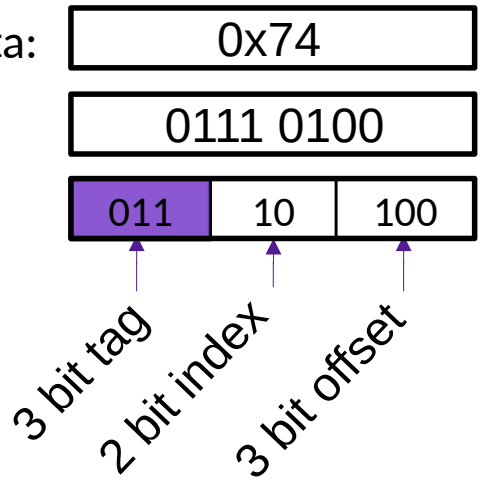
1. Replace data block with bytes from memory
   - Copies all bytes with same tag + index

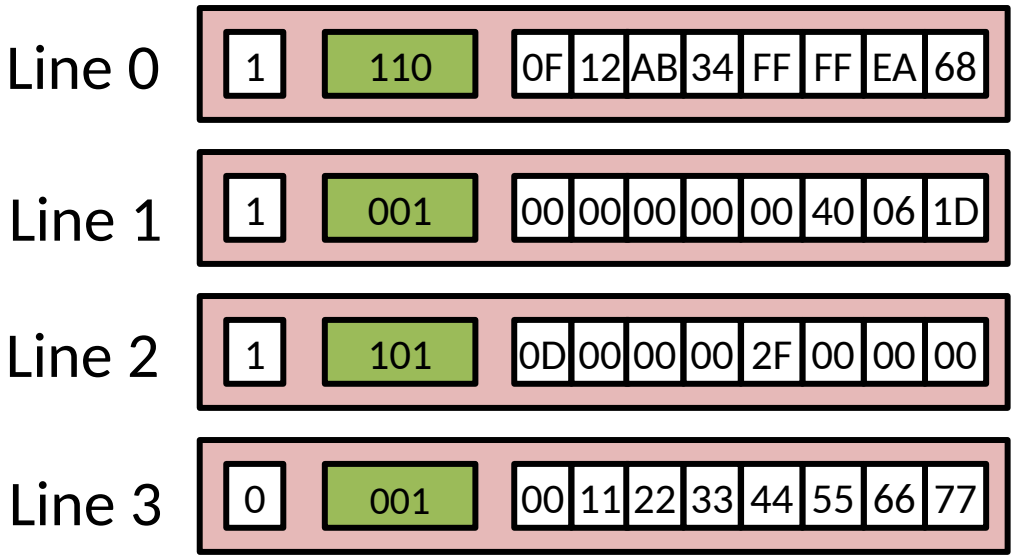


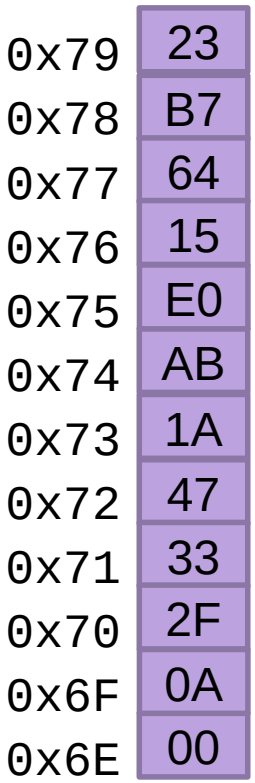
Handling a Cache Miss

When a cache miss occurs update cache line at that index:

1. Replace data block with bytes from memory
   - Copies all bytes with same tag + index



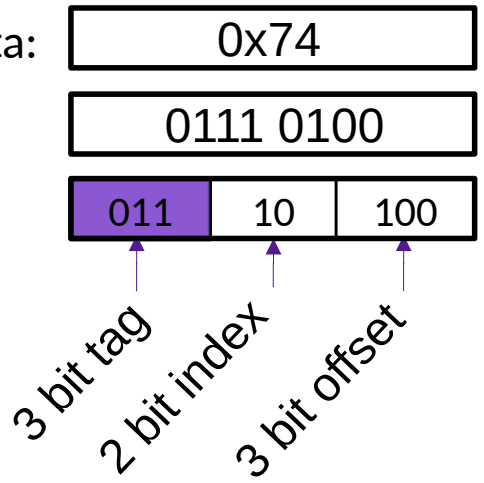


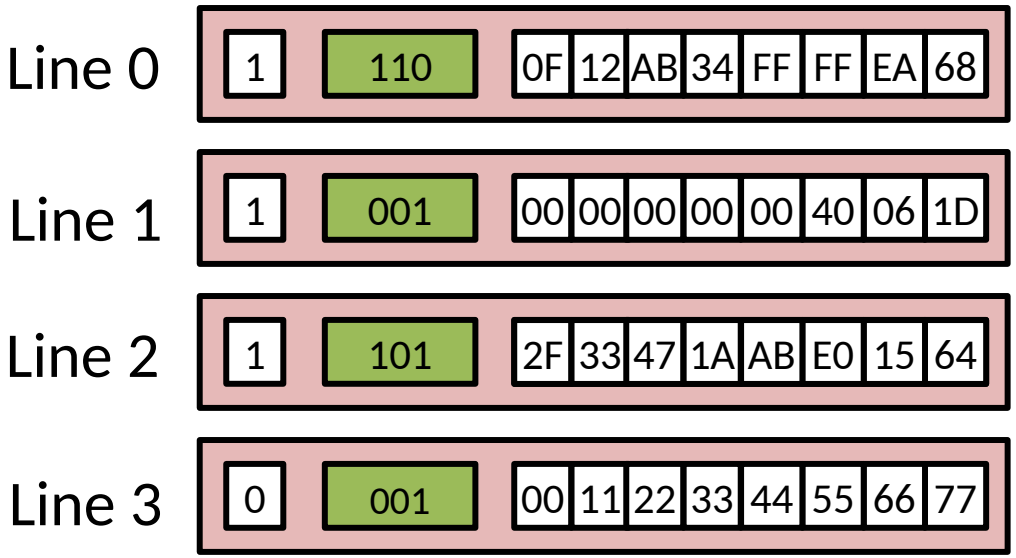


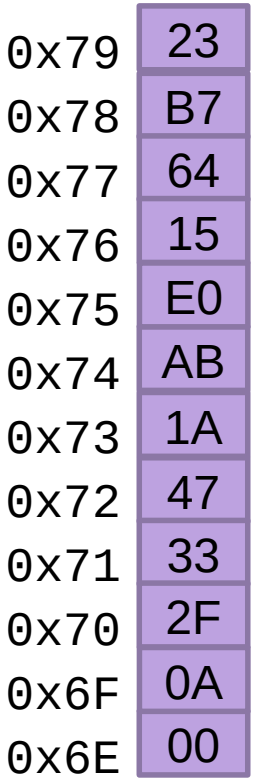
Handling a Cache Miss

When a cache miss occurs update cache line at that index:

1. Replace data block with bytes from memory
   - Copies all bytes with same tag + index



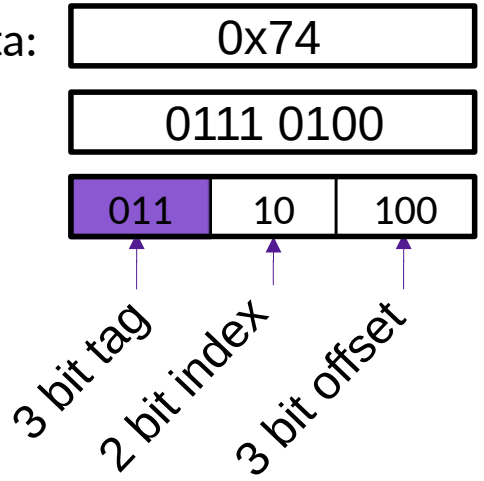


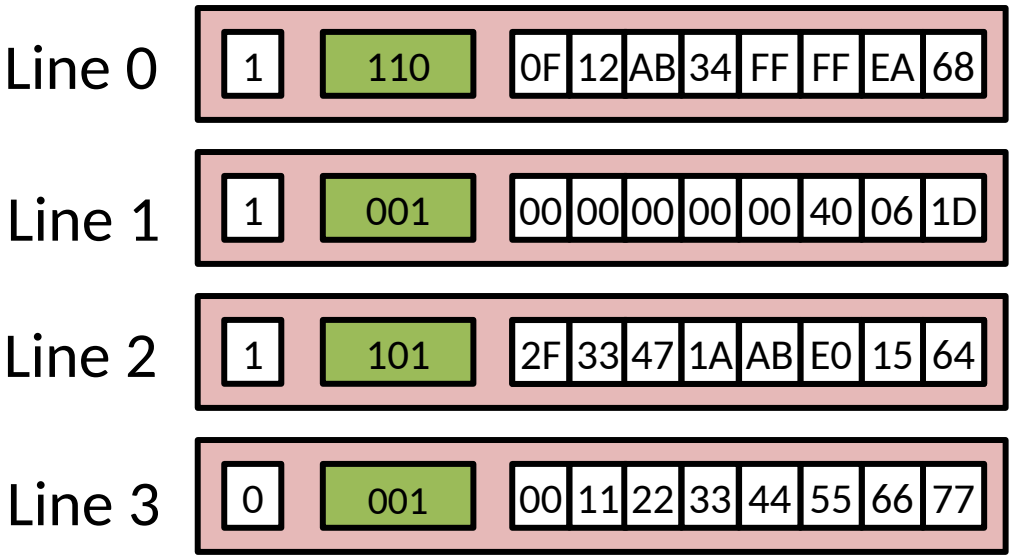


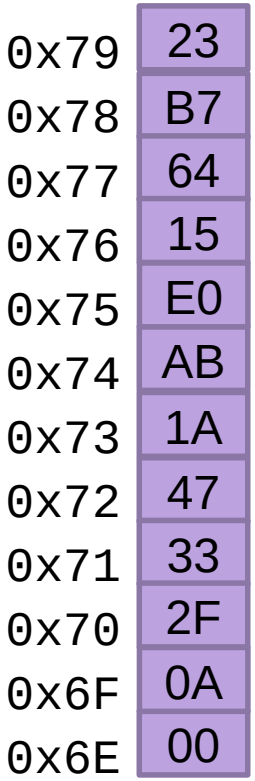
Handling a Cache Miss

When a cache miss occurs update cache line at that index:

1. Replace data block with bytes from memory
   - Copies all bytes with same tag + index
2. Update tag



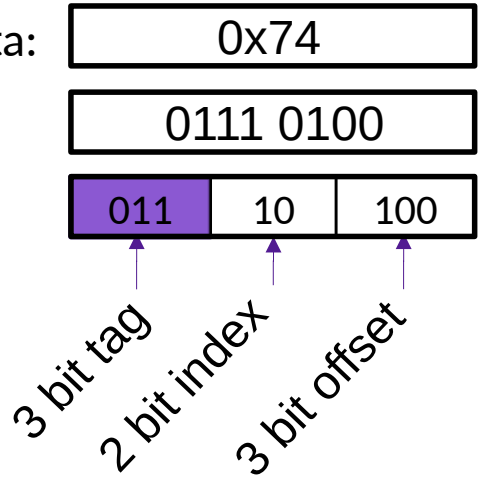


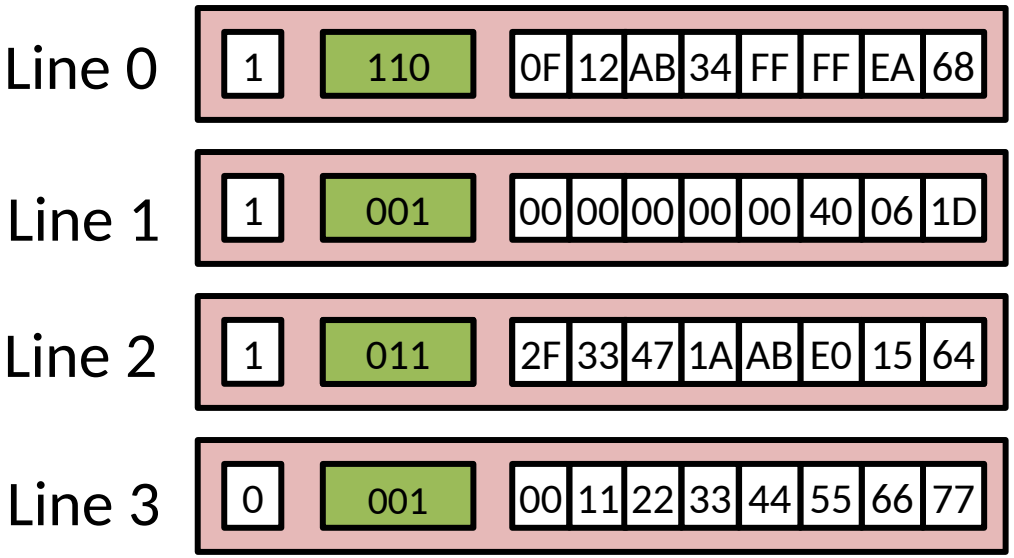


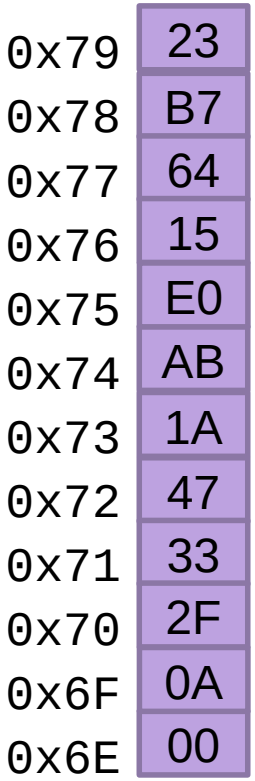
Handling a Cache Miss

When a cache miss occurs update cache line at that index:

1. Replace data block with bytes from memory
   - Copies all bytes with same tag + index
2. Update tag



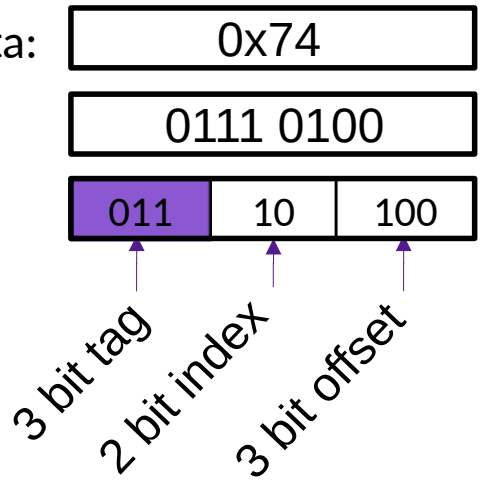




Handling a Cache Miss

When a cache miss occurs update cache line at that index:

1. Replace data block with bytes from memory
   - Copies all bytes with same tag + index
2. Update tag
3. Set valid bit to 1 (if not already)



Line 0	1	110	0F	12	AB	34	FF	FF	EA	68
Line 1	1	001	00	00	00	00	00	40	06	1D
Line 2	1	011	2F	33	47	1A	AB	E0	15	64
Line 3	0	001	00	11	22	33	44	55	66	77

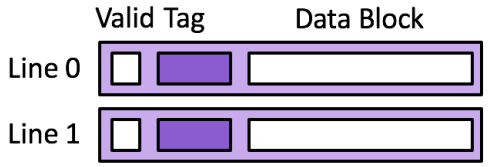
0x79	23
0x78	B7
0x77	64
0x76	15
0x75	E0
0x74	AB
0x73	1A
0x72	47
0x71	33
0x70	2F
0x6F	0A
0x6E	00

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60					
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Assume 8 byte data blocks

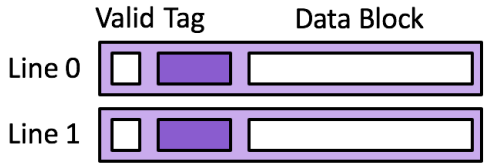
Line 0				Line 1			
0	0000	47	48	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60					
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Assume 8 byte data blocks

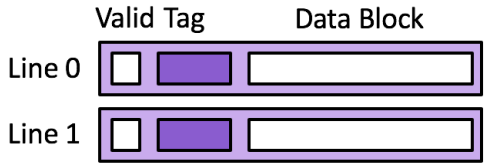
Line 0				Line 1			
0	0000	47	48	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60					
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

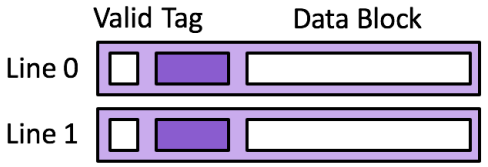
Line 0				Line 1			
0	0000	47	48	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000		
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

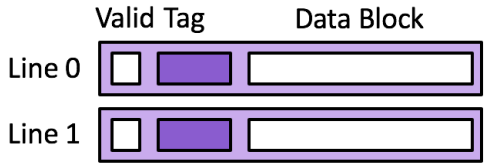
Line 0				Line 1			
0	0000	47	48	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

Assume 8 byte data blocks

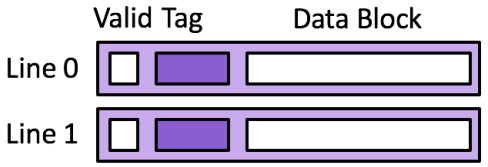
Line 0				Line 1			
0	0000	47	48	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

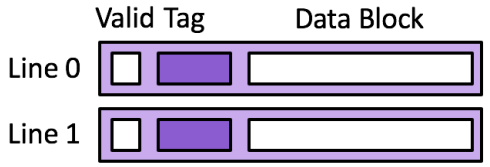
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14				

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64					
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

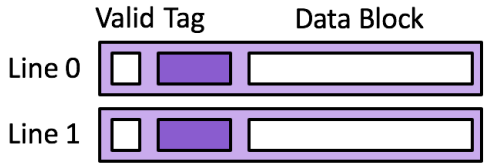
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14				

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100		
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

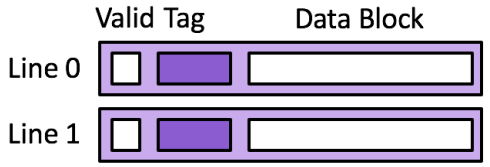
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14				

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

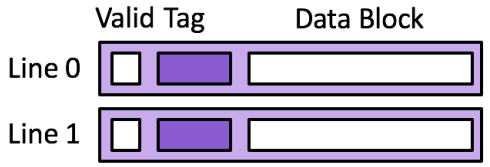
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14				

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70					
rd 0x64					
rd 0x64					
rd 0x60					

Time

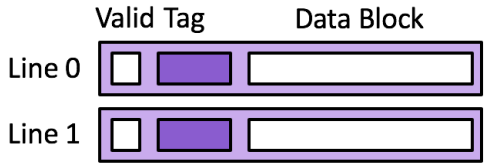
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000		
rd 0x64					
rd 0x64					
rd 0x60					

Time

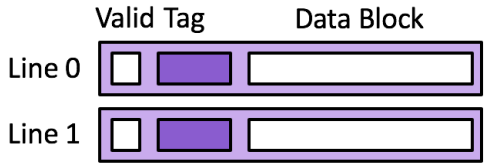
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	
rd 0x64					
rd 0x64					
rd 0x60					

Time

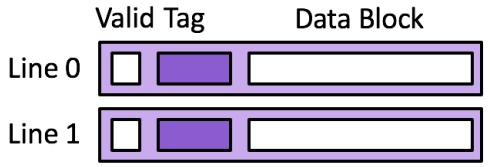
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	
rd 0x64					
rd 0x64					
rd 0x60					

Time

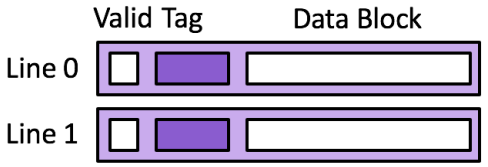
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64					
rd 0x64					
rd 0x60					

Time

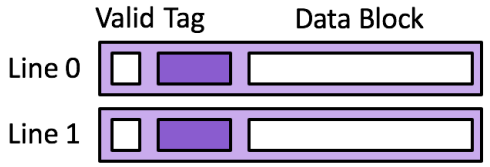
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100		
rd 0x64					
rd 0x60					

Time

Assume 8 byte data blocks

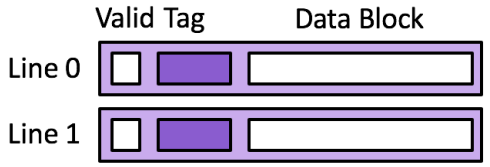
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	
rd 0x64					
rd 0x60					

Time

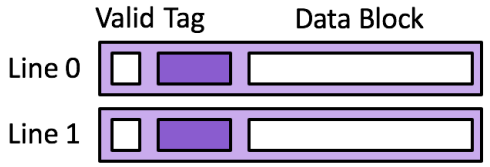
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	
rd 0x64					
rd 0x60					

Time

Assume 8 byte data blocks

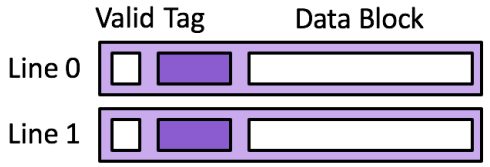
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64					
rd 0x60					

Time

Assume 8 byte data blocks

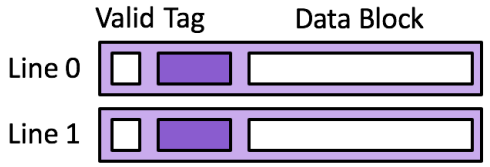
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100		
rd 0x60					

Time

Assume 8 byte data blocks

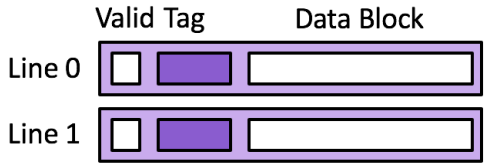
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100	Hit	
rd 0x60					

Time

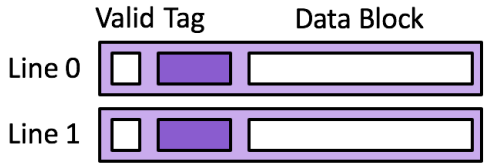
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100	Hit	14
rd 0x60					

Time

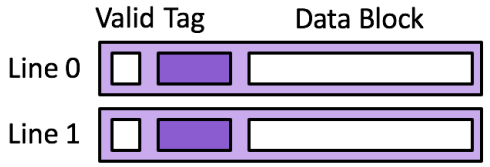
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14				
1	0111	17	18				
1	0110	13	14				

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100	Hit	14
rd 0x60	0110	0	000		

Time

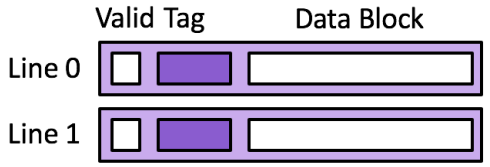
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100	Hit	14
rd 0x60	0110	0	000	Hit	

Time

Assume 8 byte data blocks

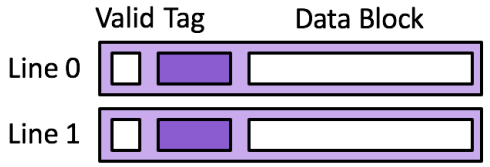
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100	Hit	14
rd 0x60	0110	0	000	Hit	13

Time

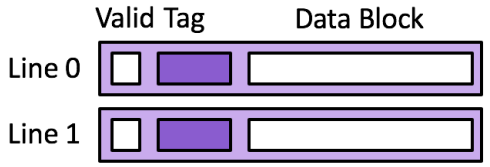
Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

Exercise 4: Direct-mapped Cache

Memory

0x74	18
0x70	17
0x6c	16
0x68	15
0x64	14
0x60	13

Cache



Assume 8 byte data blocks

Access	tag	idx	off	h/m	val
rd 0x60	0110	0	000	Miss	13
rd 0x64	0110	0	100	Hit	14
rd 0x70	0111	0	000	Miss	17
rd 0x64	0110	0	100	Miss	14
rd 0x64	0110	0	100	Hit	14
rd 0x60	0110	0	000	Hit	13

Time

Line 0				Line 1			
0	0000	47	48	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0111	17	18	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50
1	0110	13	14	0	0000	49	50

How well does this take advantage of spatial locality?

How well does this take advantage of temporal locality?