CS 3410 Lab 9

Fall 2025

Cornell Bowers C·IS
Computer Science

Agenda

1. Intro to Caches
2. Worksheet
3. Assignment Tips

Intro to Caches

Cache Parameters

- Tag: High-order bits used to compare addresses of the same cache mapping
- Index: Bits that determine where in the cache an address can go
- Offset: Low-order bits to select a byte within a cache block

Address breakdown:

Direct Mapped Cache

Each address maps to exactly 1 cache block

ex) 4 byte direct mapped cache

- 4 blocks, 1 byte each
- Need 4 = 2^2 indices, so 2 index bits

CACHE

index	Tag	Data
11	00	D
10	00	C
01	00	B
00	00	A

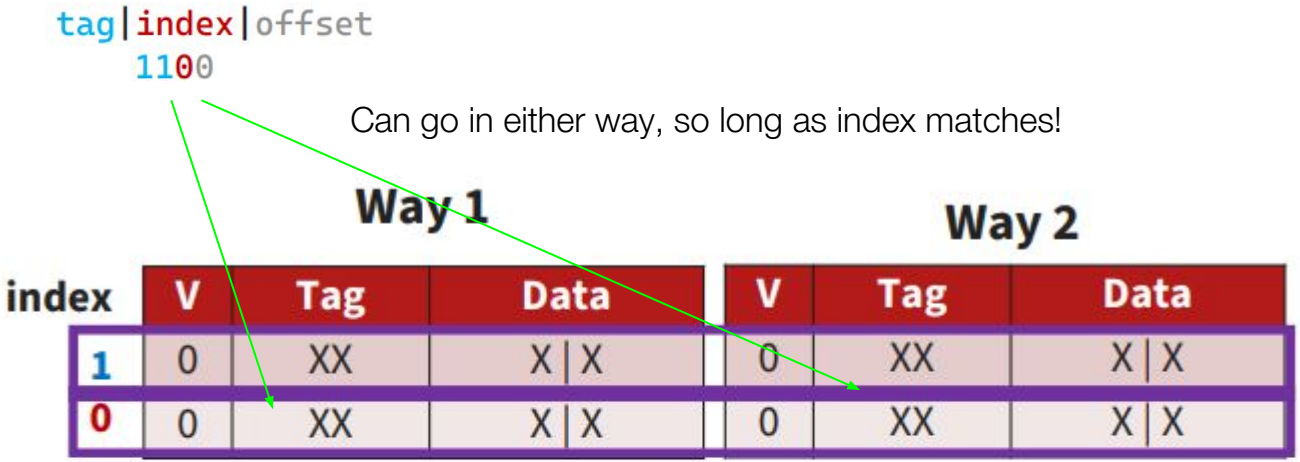
tag|index
1101

MEMORY

addr	Data
1111	P
1110	O
1101	N
1100	M
1011	L
1010	K
1001	J
1000	I
0111	H
0110	G
0101	F
0100	E
0011	D
0010	C
0001	B
0000	A

Set Associative Cache

- Divide cache into **sets**; can store data in any **way** within a set
- Index is used to map addresses to a specific **set**, so not fully associative



Fully Associative Cache

Each address can map to any cache block

ex) 8 byte fully associative cache

- 4 blocks, 2-bytes each (use offset to access each byte)
- 4-bit addresses, no need for index bit

CACHE

V	Tag	Data
0	010	E | F
0	000	A | B
0	001	C | D
0	011	G | H

tag|offset
1101

No more index!

- Any address, any entry
- 4-bit addresses
- 2-byte blocks

MEMORY

addr	Data
1111	P
1110	O
1101	N
1100	M
1011	L
1010	K
1001	J
1000	I
0111	H
0110	G
0101	F
0100	E
0011	D
0010	C
0001	B
0000	A

Replacement Policies

- LRU (Least Recently Used): evict block used longest ago
  - Requires extensive bookkeeping
- NMRU (Not Most Recently Used): evict any block that was not the most recently accessed
  - Less bookkeeping
- … and more
- Replacement policies are only required for **associative** caches

Cache Hits/Misses

- Cache Hit: data **is** in cache

- Cache Miss: data is **not** in cache
  - Have to retrieve data from memory -> extra time!

- For the 4 byte direct mapped cache on the right, would the following loads be a hit or a miss?
  - **Load 1100**
  - **Load 1101**?

CACHE

index	V	Tag	Data
11	0	XX	X
10	0	XX	X
01	0	XX	X
00	1	11	M

```
tag|index
 1101
```

Worksheet

Worksheet

- address = tag + index + offset
- AMAT = access time + miss rate x miss penalty
    - miss rate = # cache misses / # cache accesses
    - miss penalty = extra time it takes to retrieve data from lower memory

Assignment Tips

Assignment Overview

- **Objective**: simulate a single core cache
- Task 1-2: initialize cache and implement functions to extract offset, index, and tag
- Task 3: implement a direct-mapped cache
- Task 4: add support for set associativity
- Task 5: implement a write-back cache
- Task 6: compute write-thru statistics
- Task 7: cleanup cache
- Remember to use `my_malloc` and `my_free`!