Cache Memory

1-Initial State

Request Byte Address

000	_ _	_	
001	_ _	_	
010	_ _	_	
011	_ _	_	
100	_ _	_	
101	_ _	_	
110	_ _	_	
111	_ _	_	

8 Blocks

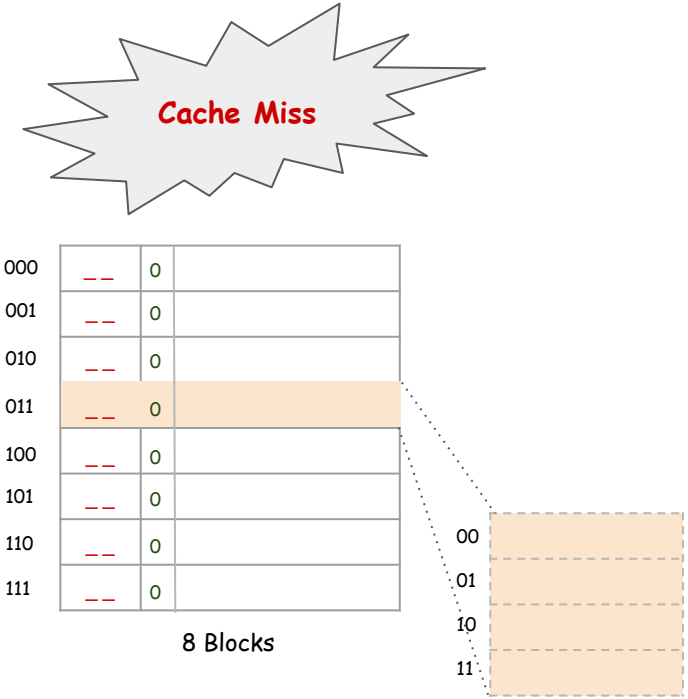
00000	
00011	
01000	
01011	
10000	
10011	
11000	
11011	

32 Blocks

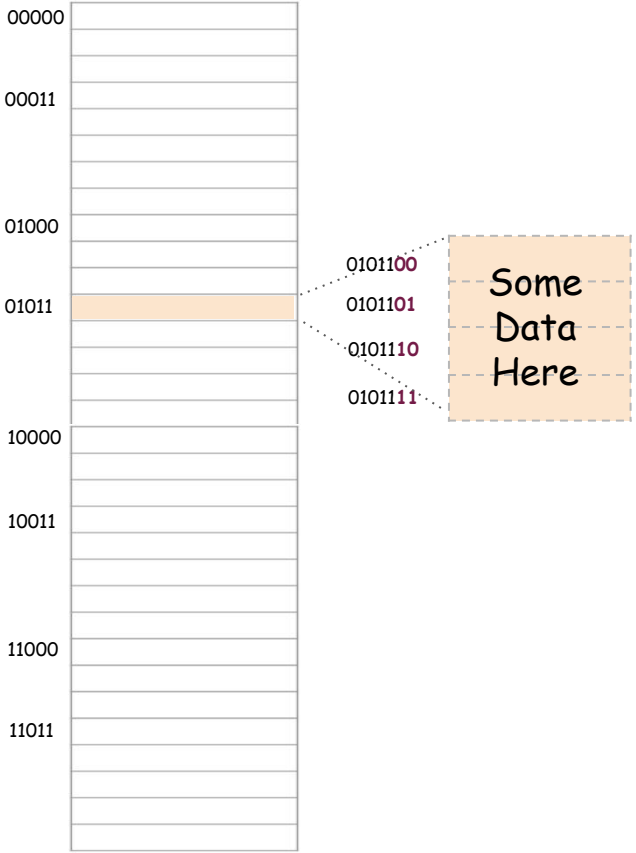
2.

Request Byte Address

0101110



8 Blocks



32 Blocks

3.

Request Byte Address

0101110

000	_ _	0	
001	_ _	0	
010	_ _	0	
011	0 1	1	Mem(01011)
100	_ _	0	
101	_ _	0	
110	_ _	0	
111	_ _	0	

8 Blocks

00
01
10
11

Some Data Here

00000
00011
01000
01011
10000
10011
11000
11011

32 Blocks

0101100
0101101
0101110
0101111

Some Data Here

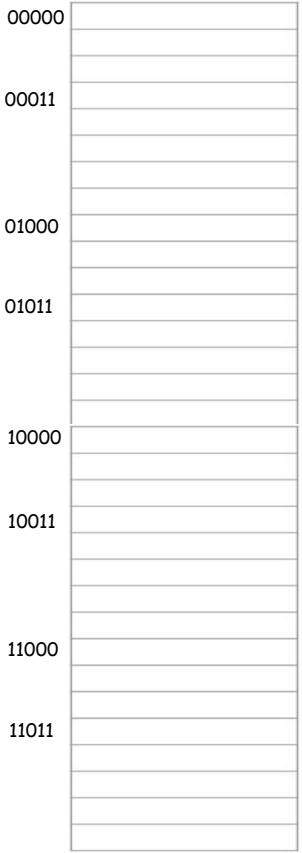
4.

Request Byte Address

1000001

000	_ _	0	
001	_ _	0	
010	_ _	0	
011	0 1	1	Mem(01011)
100	_ _	0	
101	_ _	0	
110	_ _	0	
111	_ _	0	

8 Blocks



32 Blocks

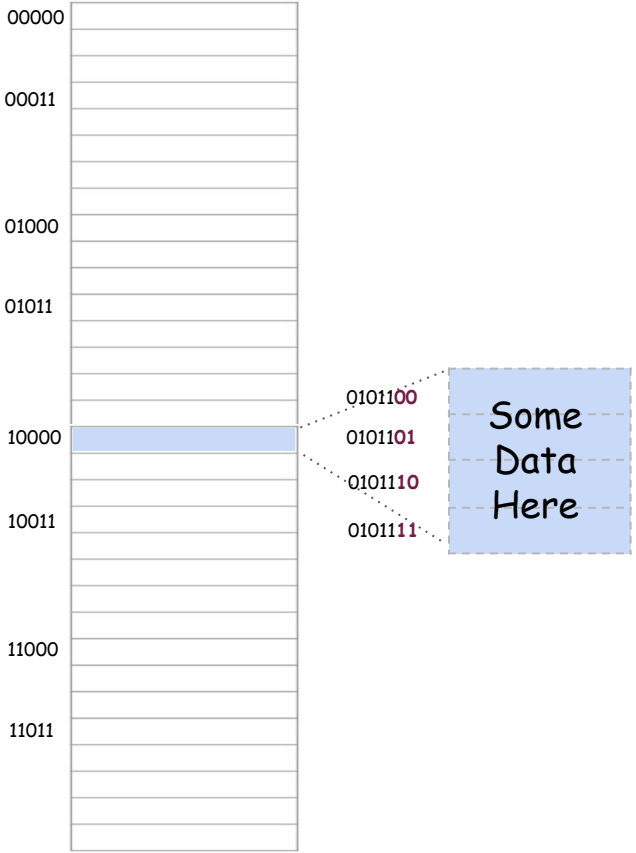
5.

Request Byte Address

1000001

000	1 0	1	Mem(10000)
001	_ _	0	
010	_ _	0	
011	0 1	1	Mem(01011)
100	_ _	0	
101	_ _	0	
110	_ _	0	
111	_ _	0	

8 Blocks

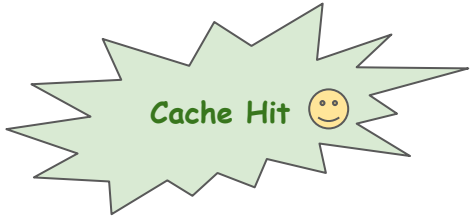


32 Blocks

6.

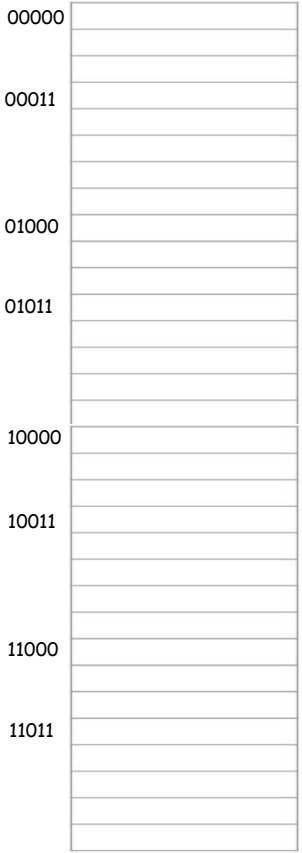
Request Byte Address

0101100



000	1 0	1	Mem(10000)
001	_ _	0	
010	_ _	0	
011	0 1	1	Mem(01011)
100	_ _	0	
101	_ _	0	
110	_ _	0	
111	_ _	0	

8 Blocks



32 Blocks

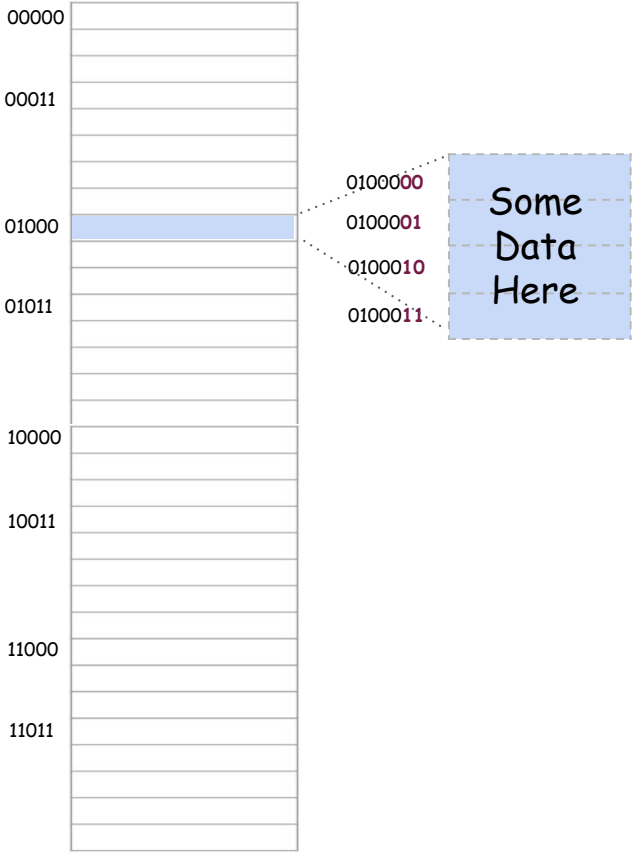
7.

Request Byte Address

0100011

000	1 0	1	Mem(10000)
001	_ _	0	
010	_ _	0	
011	0 1	1	Mem(01011)
100	_ _	0	
101	_ _	0	
110	_ _	0	
111	_ _	0	

8 Blocks



32 Blocks

7.

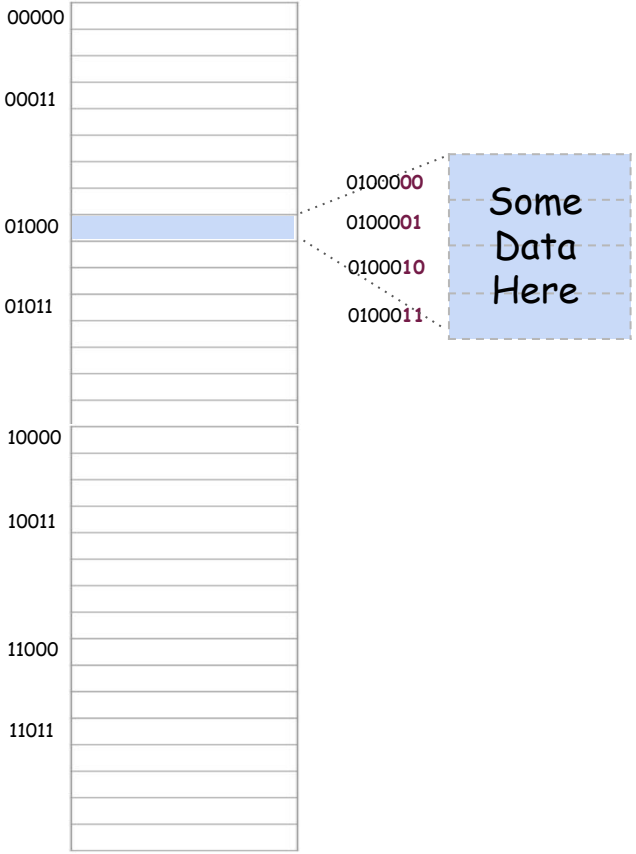
Request Byte Address

0100011

replace

000	0 1	1	Mem(01000)
001	_ _	0	
010	_ _	0	
011	0 1	1	Mem(01011)
100	_ _	0	
101	_ _	0	
110	_ _	0	
111	_ _	0	

8 Blocks



32 Blocks

Cache Performance

MissRate = #CacheMisses / #CacheAccesses

Average Memory Access Time

$$\boxed{\text{AMAT} = \text{HitTime} + \text{MissRate} \times \text{MissPenalty}}$$

Average Memory Access Time

AMAT = HitTime + MissRate x MissPenalty

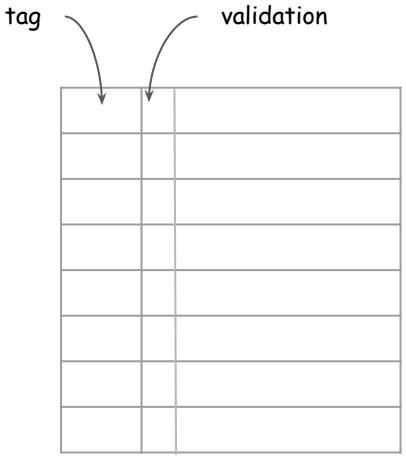
MissRate = #CacheMisses / #CacheAccesses

Cache Miss Categories (3Cs)

- Compulsory (First Access is always a miss)
- Capacity (program working set is larger than cache capacity)
- Conflict (several blocks are mapped to same block frame)

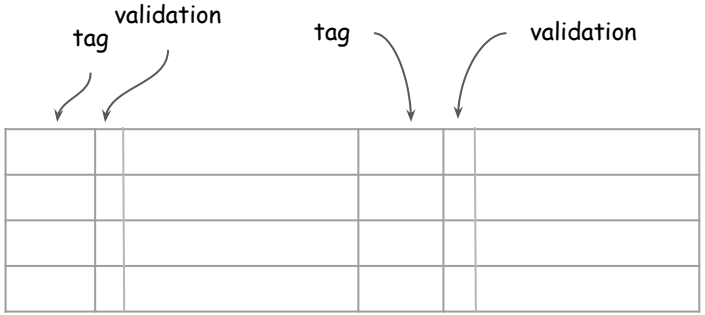
Associative Cache

1-way set associative
(direct mapping)



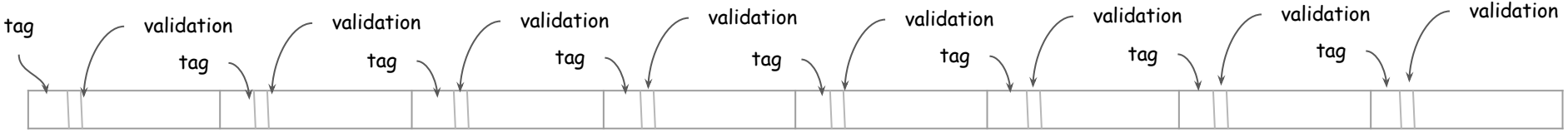
#Sets = #Blocks

2-ways set associative



#Sets ≠ #Blocks

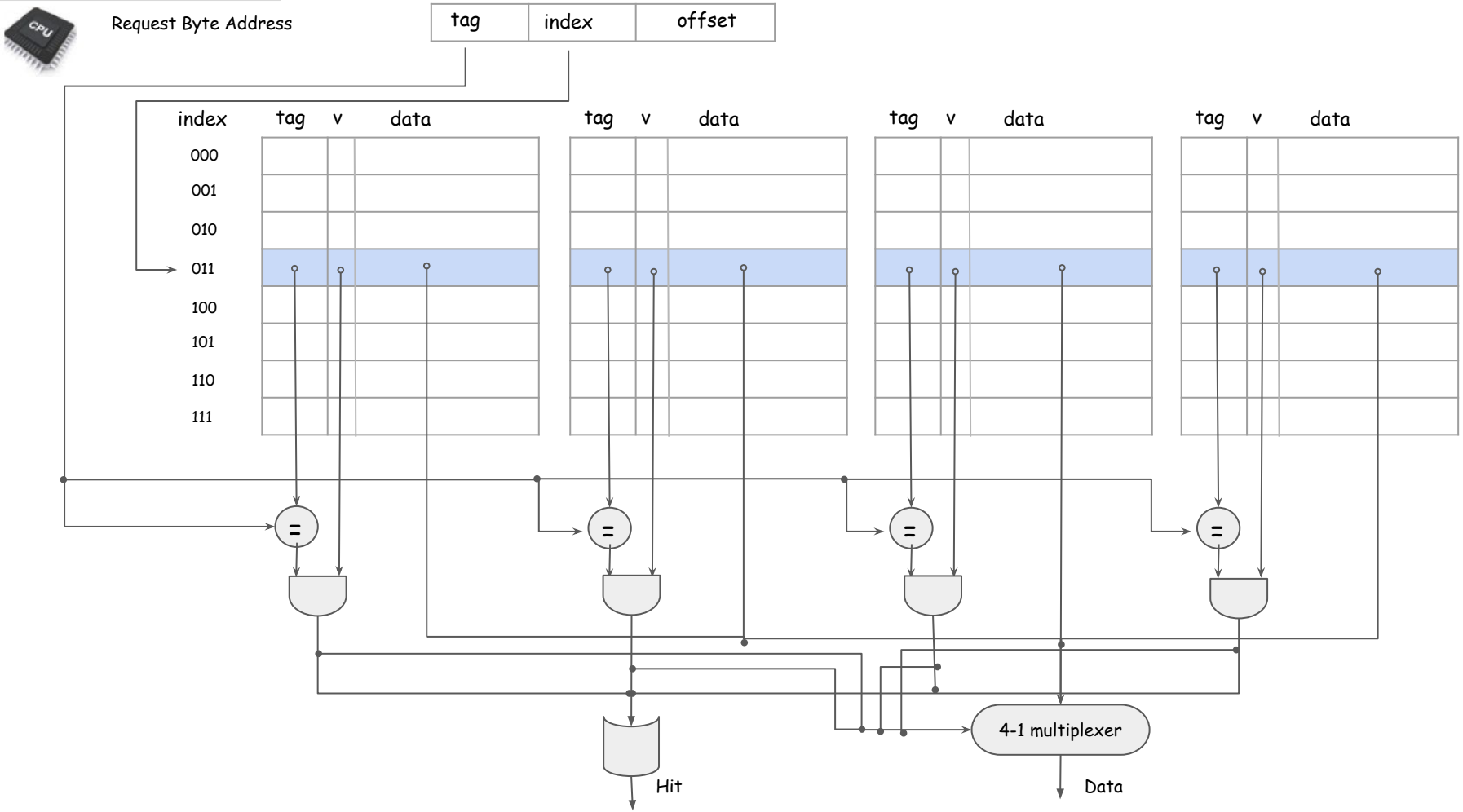
8-ways set associative
(fully associative)



#Sets = 1
(cacheIndex=0)

CacheSize = #Sets x Ways x #BlockSize

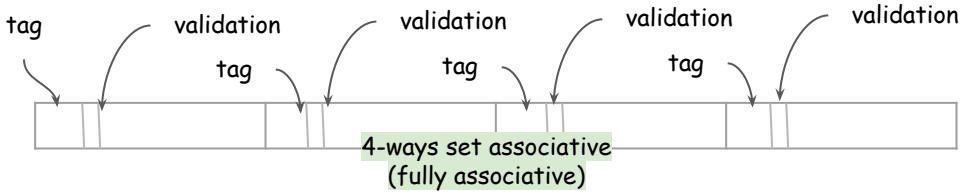
Associative Cache

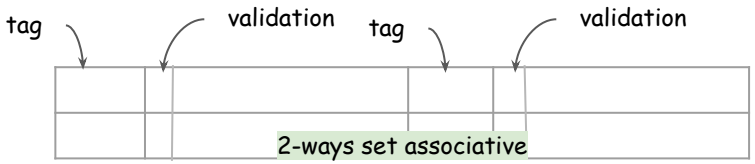


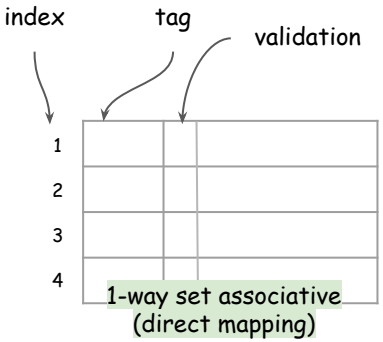
Associative Cache

Which cache organization reduces misses?

What type of misses can this organization reduce (compulsory, capacity, conflict) ?







Exercise:

Assume there are three small caches, each consisting of four one-word blocks. One cache is fully associative, a second is two-way set associative, and the third is direct mapped.

Find the number of misses for each cache organization, given the following sequence of block addresses: 0, 8, 0, 6, and 8. (16 bits address)

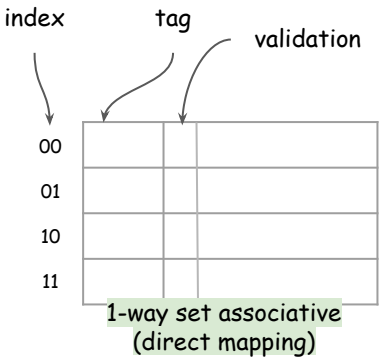
Associative Cache

Exercise Solution:

Step 1

Request Byte Address

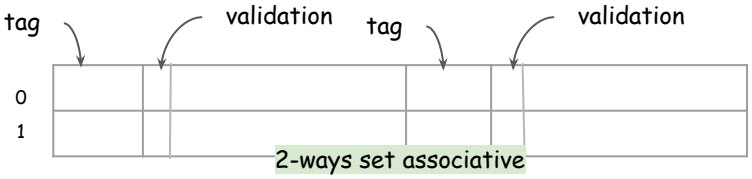
$(0)_{10} = (0000000000000000)_2$



Offset=1 bit
Index = 2 bits
Tag = 13 bits

Miss

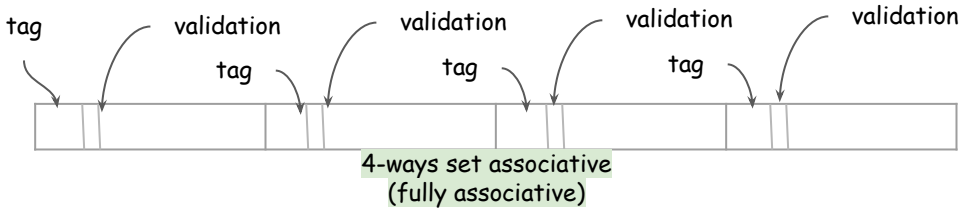
Total no. misses = 1



Offset=1 bit
Index = 1 bits
Tag = 14 bits

Miss

Total no. misses = 1



Offset=1 bit
Index = 0 bits
Tag = 15 bits

Miss

Total no. misses = 1

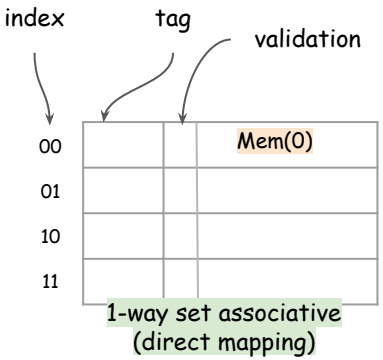
Associative Cache

Exercise Solution:

Step 2

Request Byte Address

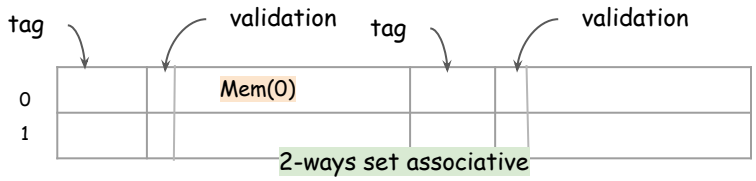
$(8)_{10} = (0000000000001000)_2$



Offset=1 bit
Index = 2 bits
Tag = 13 bits

Miss

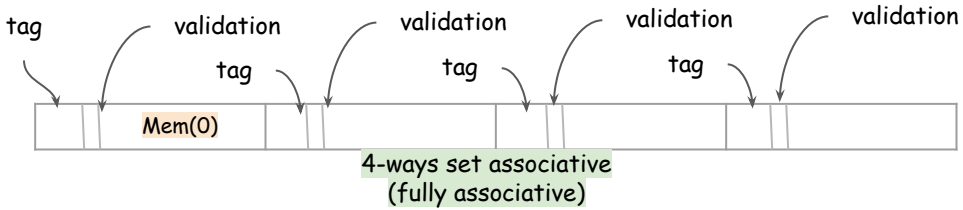
Total no. misses = 2



Offset=1 bit
Index = 1 bits
Tag = 14 bits

Miss

Total no. misses = 2



Offset=1 bit
Index = 0 bits
Tag = 15 bits

Miss

Total no. misses = 2

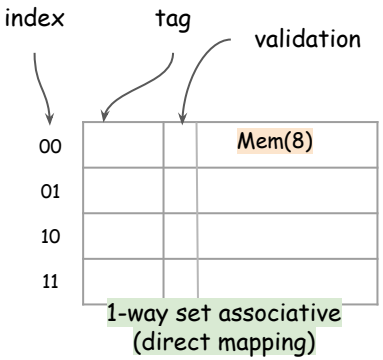
Associative Cache

Exercise Solution:

Step 3

Request Byte Address

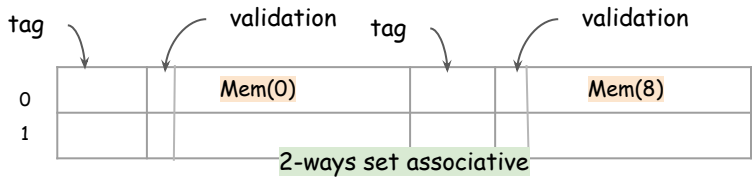
$(0)_{10} = (0000000000000000)_2$



Offset=1 bit
Index = 2 bits
Tag = 13 bits

Miss

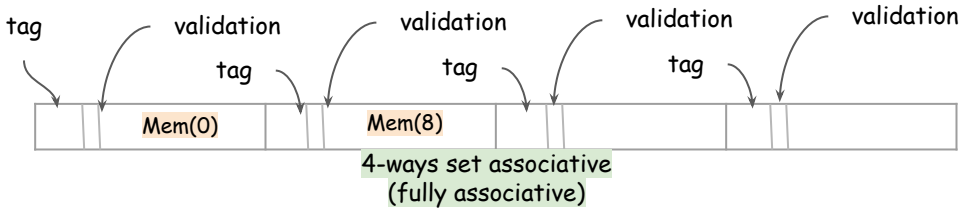
Total no. misses = 3



Offset=1 bit
Index = 1 bits
Tag = 14 bits

Hit

Total no. misses = 2



Offset=1 bit
Index = 0 bits
Tag = 15 bits

Hit

Total no. misses = 2

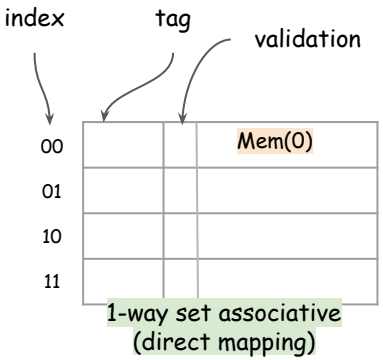
Associative Cache

Exercise Solution:

Step 4

Request Byte Address

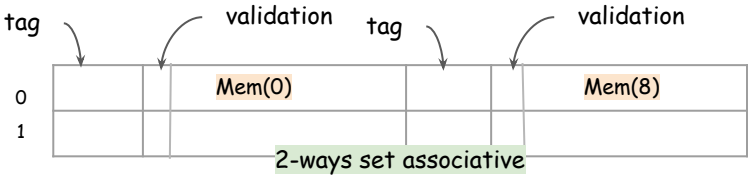
$(6)_{10} = (0000000000000110)_2$



Offset=1 bit
Index = 2 bits
Tag = 13 bits

Miss

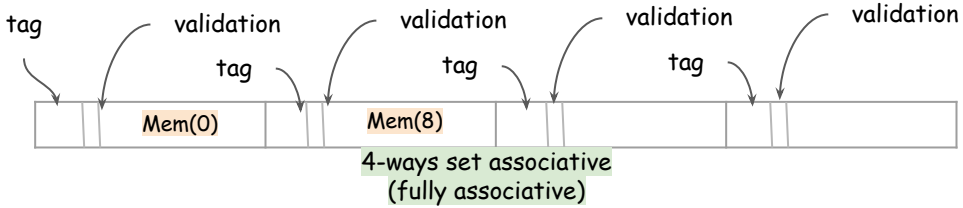
Total no. misses = 4



Offset=1 bit
Index = 1 bits
Tag = 14 bits

Miss

Total no. misses = 3



Offset=1 bit
Index = 0 bits
Tag = 15 bits

Miss

Total no. misses = 3

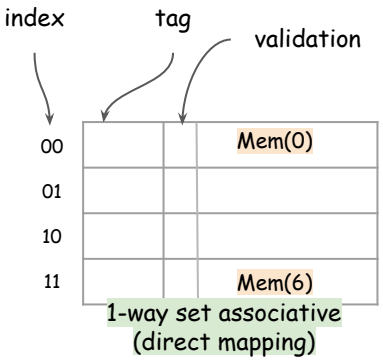
Associative Cache

Exercise Solution:

Step 5

Request Byte Address

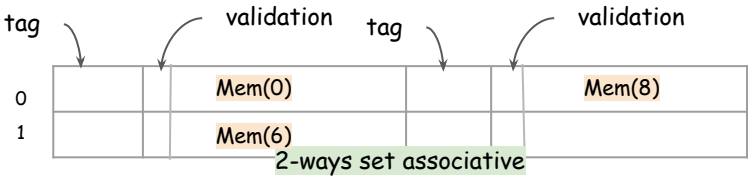
$(8)_{10} = (0000000000001000)_2$



Offset=1 bit
Index = 2 bits
Tag = 13 bits

Miss

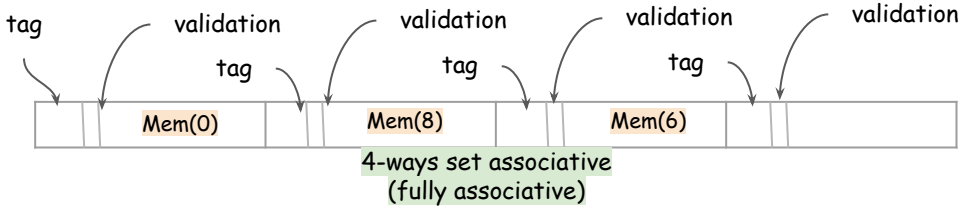
Total no. misses = 5



Offset=1 bit
Index = 1 bits
Tag = 14 bits

Hit

Total no. misses = 3



Offset=1 bit
Index = 0 bits
Tag = 15 bits

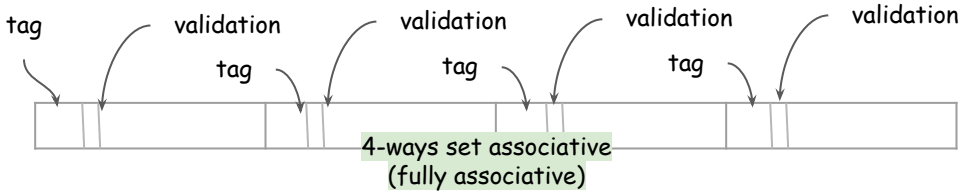
Hit

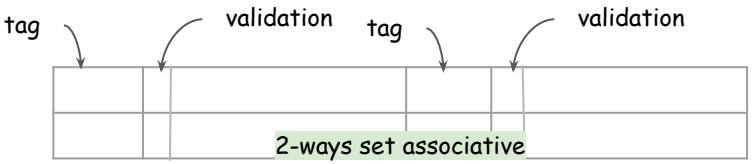
Total no. misses = 3

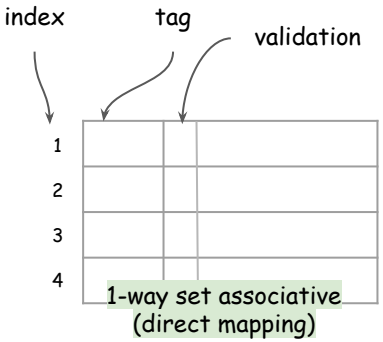
Associative Cache

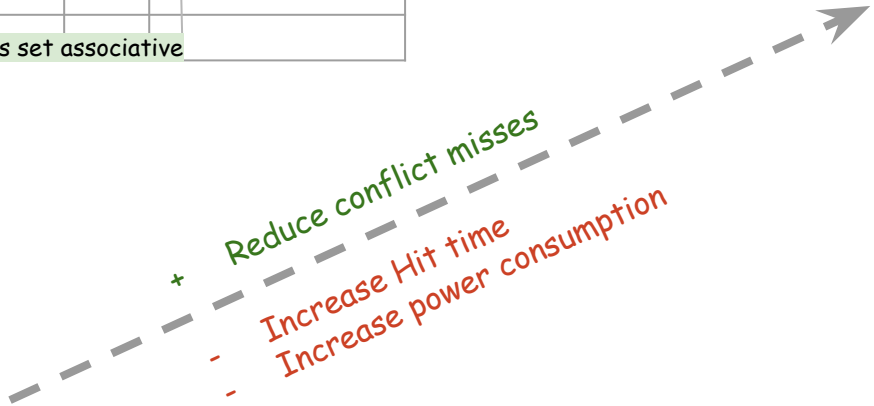
Which cache organization reduces misses?

What type of misses can this organization reduce (compulsory, capacity, conflict) ?

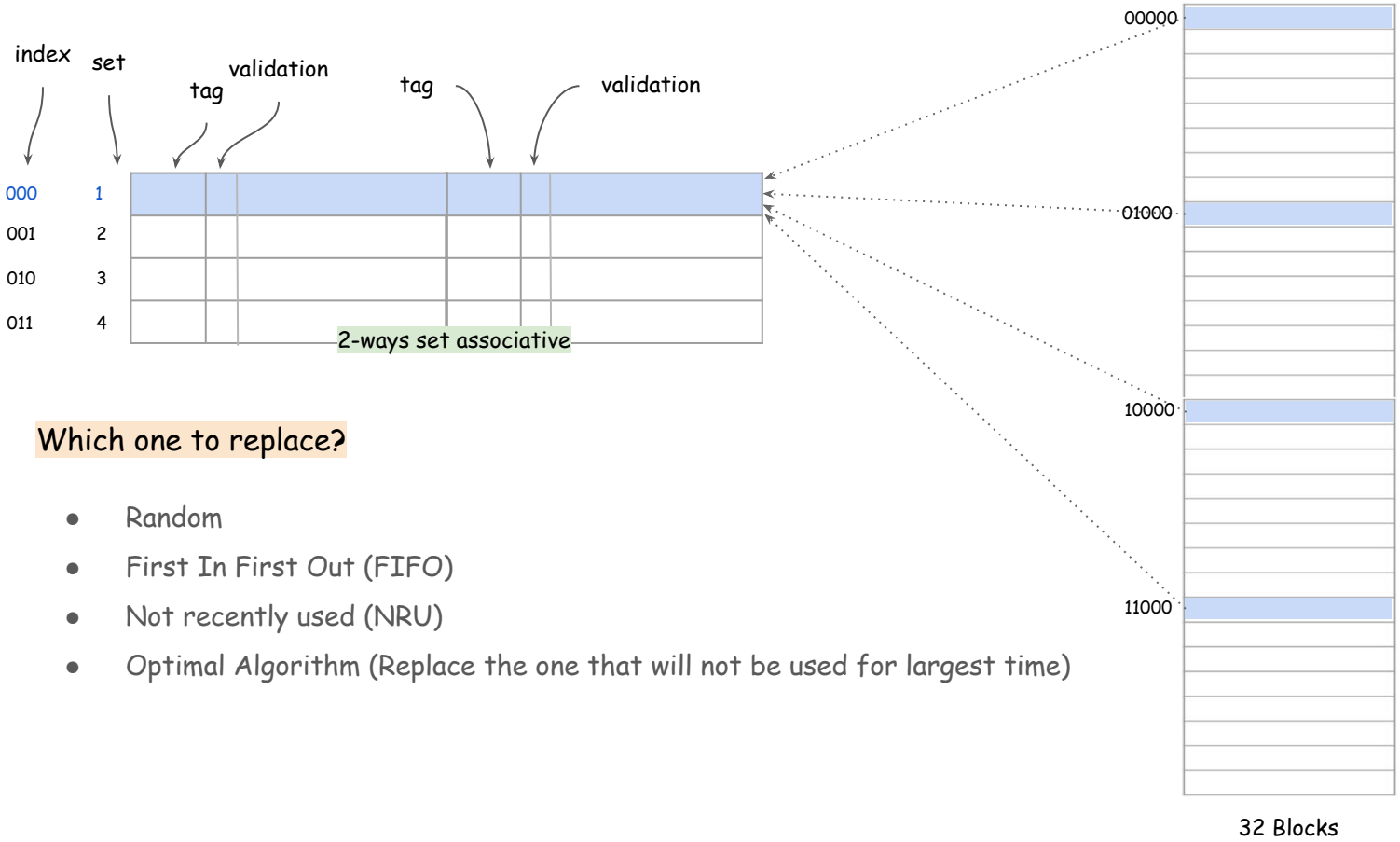








Associative Cache

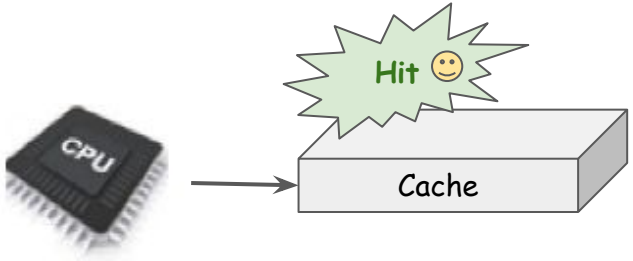


Which one to replace?

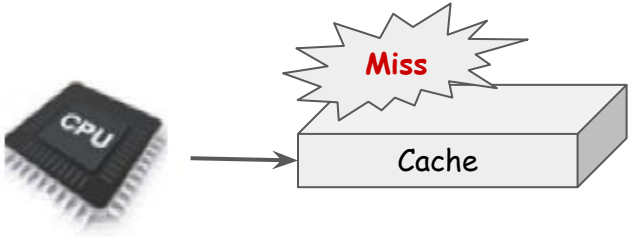
- Random
- First In First Out (FIFO)
- Not recently used (NRU)
- Optimal Algorithm (Replace the one that will not be used for largest time)

Writing Policies

Write Hit



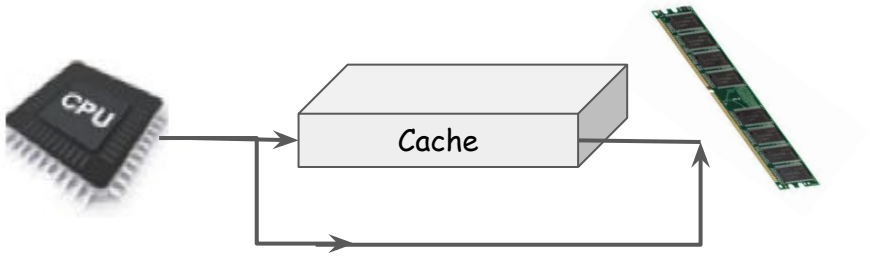
Write Miss



Writing Policies

Write Hit

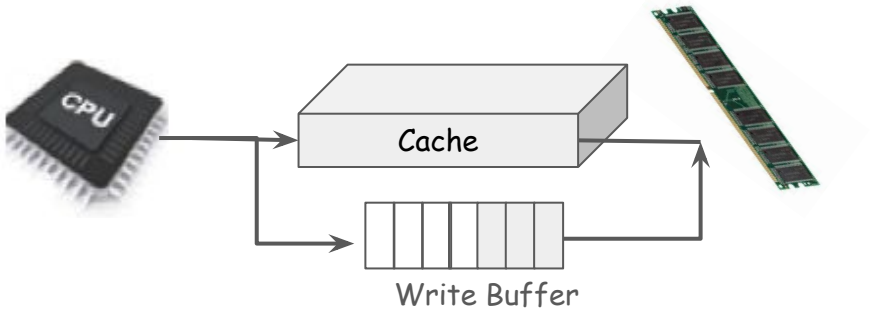
- Write Through Cache



Writing Policies

Write Hit

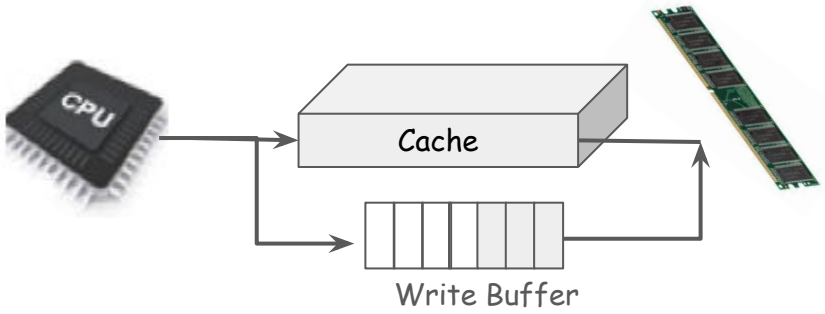
- Write Through Cache



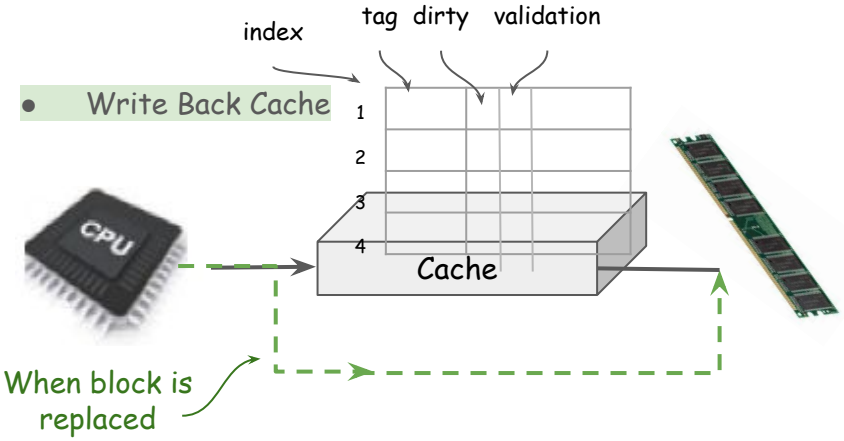
Writing Policies

Write Hit

- Write Through Cache



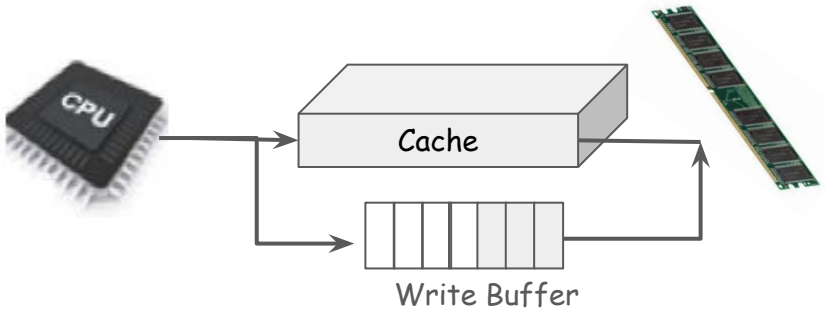
- Write Back Cache



Writing Policies

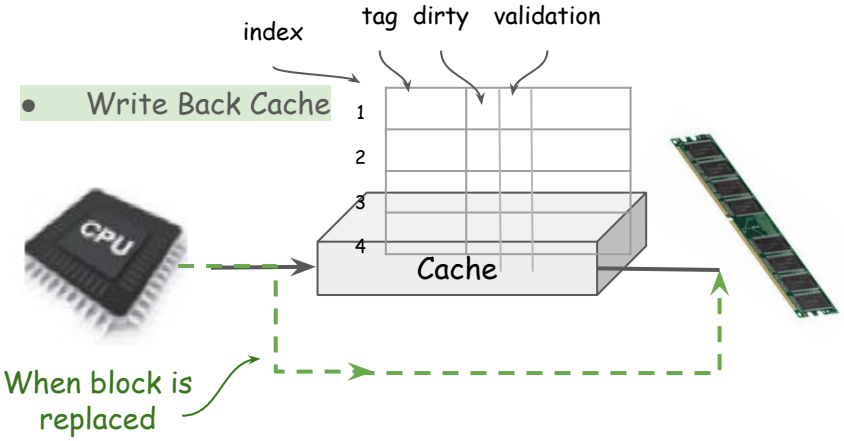
Write Hit

- Write Through Cache



+ Simple Cache logic
+ Cache and memory consistent
- Many writes to memory (more bus traffic)

- Write Back Cache



+ Fewer writes to memory
- Cache and memory inconsistent
- Requires dirty bit

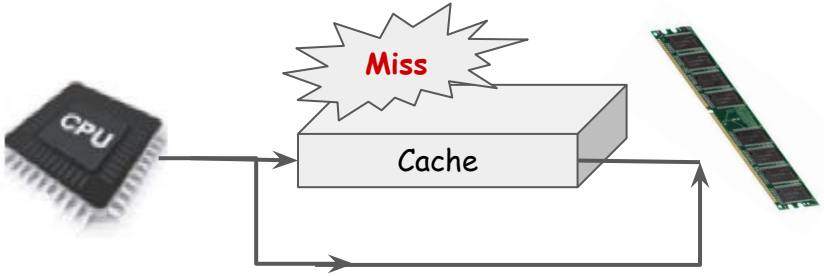
Writing Policies

Write Hit

- Write Through Cache

- Write Back Cache

Write Miss



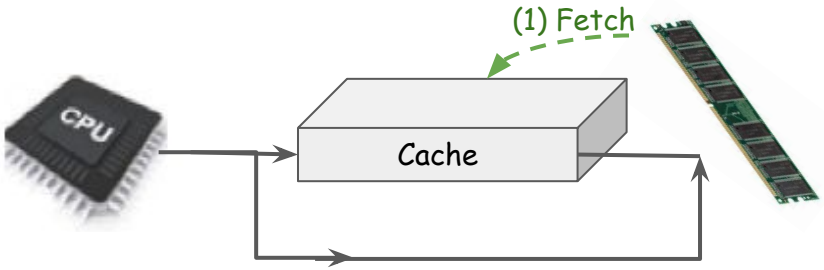
Writing Policies

Write Hit

- Write Through Cache

- Write Back Cache

Write Miss



Writing Policies

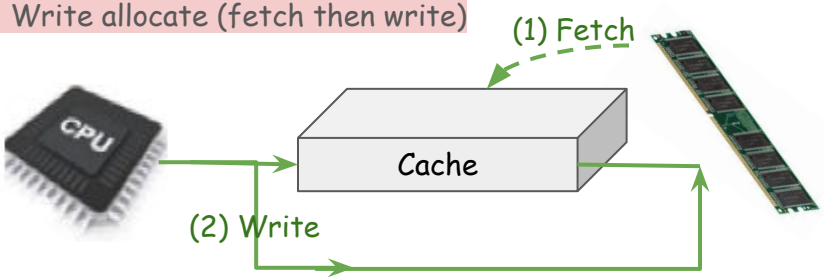
Write Hit

- Write Through Cache

- Write Back Cache

Write Miss

- Write allocate (fetch then write)



Writing Policies

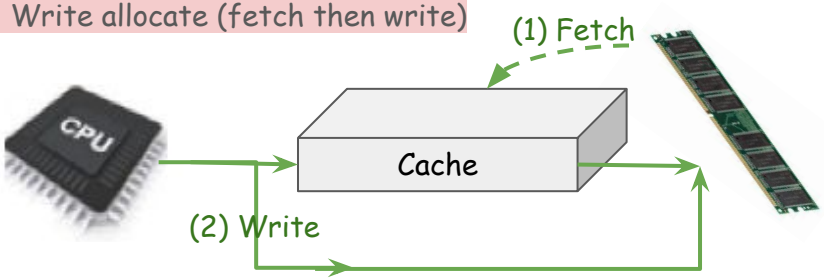
Write Hit

- Write Through Cache

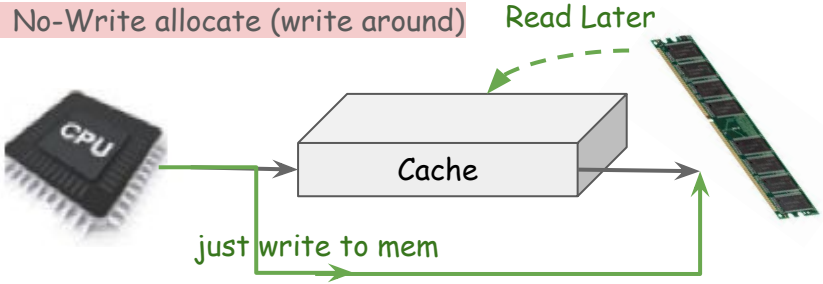
- Write Back Cache

Write Miss

- Write allocate (fetch then write)



- No-Write allocate (write around)



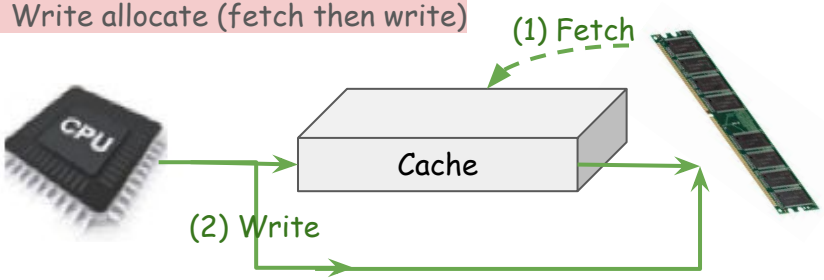
Writing Policies

+ Cache and memory consistent
- More writes

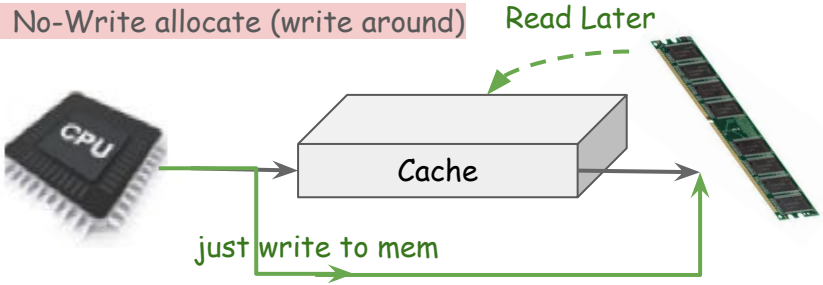
+ Fewer writes

Write Miss

- Write allocate (fetch then write)



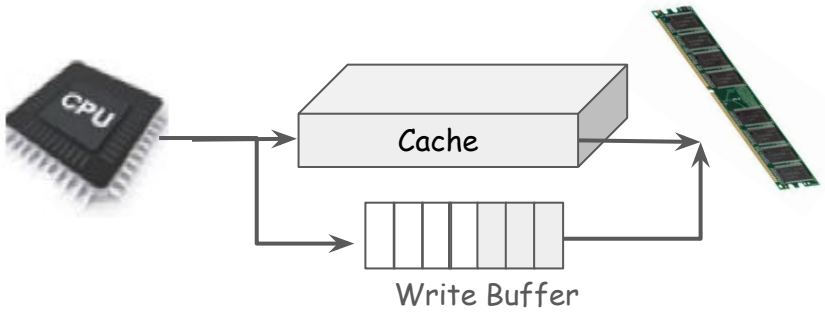
- No-Write allocate (write around)



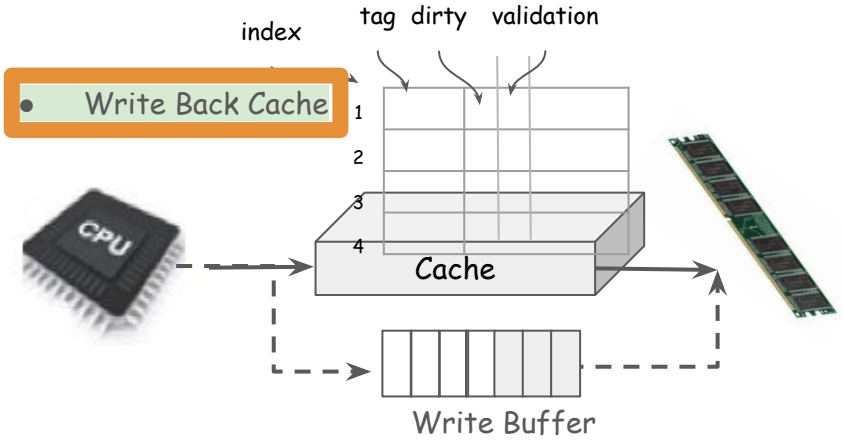
Writing Policies

Write Hit

- Write Through Cache

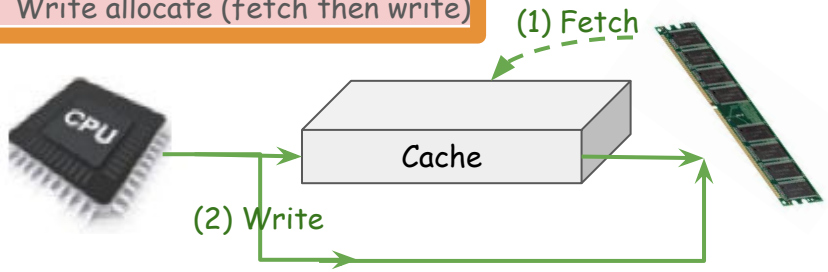


- Write Back Cache

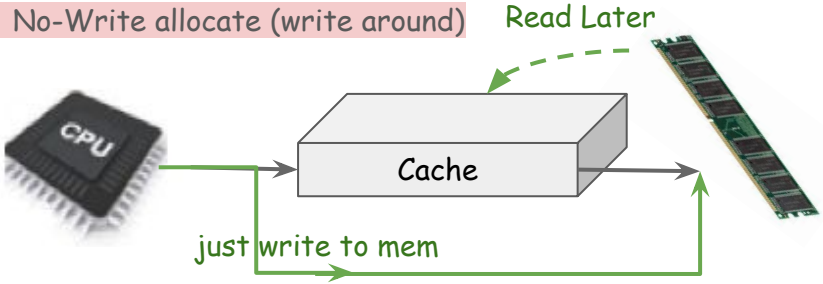


Write Miss

- Write allocate (fetch then write)



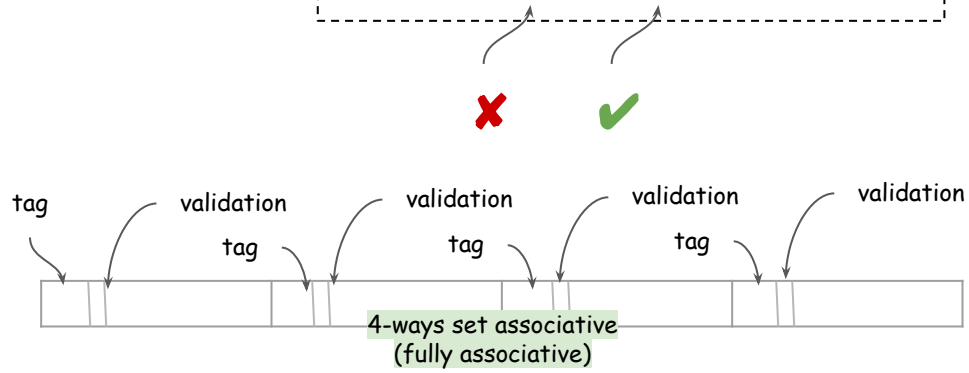
- No-Write allocate (write around)

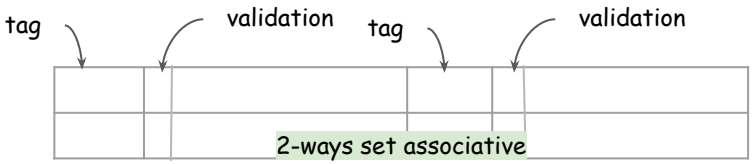


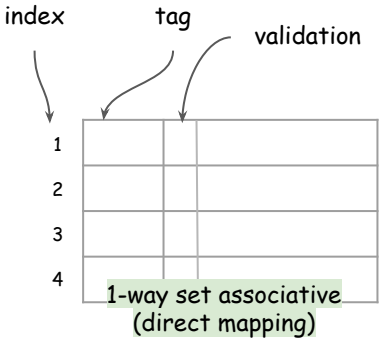
Performance

Average Memory Access Time

$$AMAT = HitTime + MissRate \times MissPenalty$$







Performance

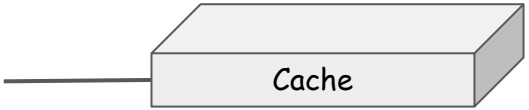
Average Memory Access Time

AMAT = HitTime + MissRate x MissPenalty

Small and Simple Cache

- Small:
  - Cache Size
- Simple:
  - Low Associativity

1-IF
2-ID
3-EX
4-Mem
5-WB



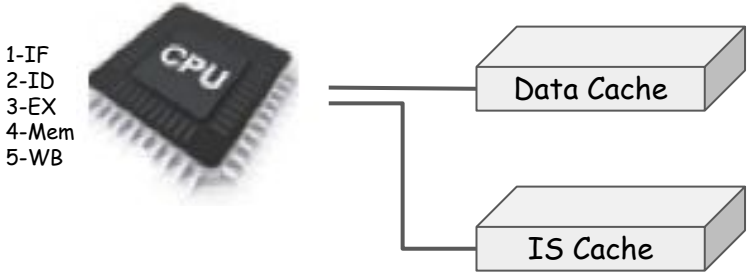
Performance

Average Memory Access Time

$$AMAT = HitTime + MissRate \times MissPenalty$$

Small and Simple Cache

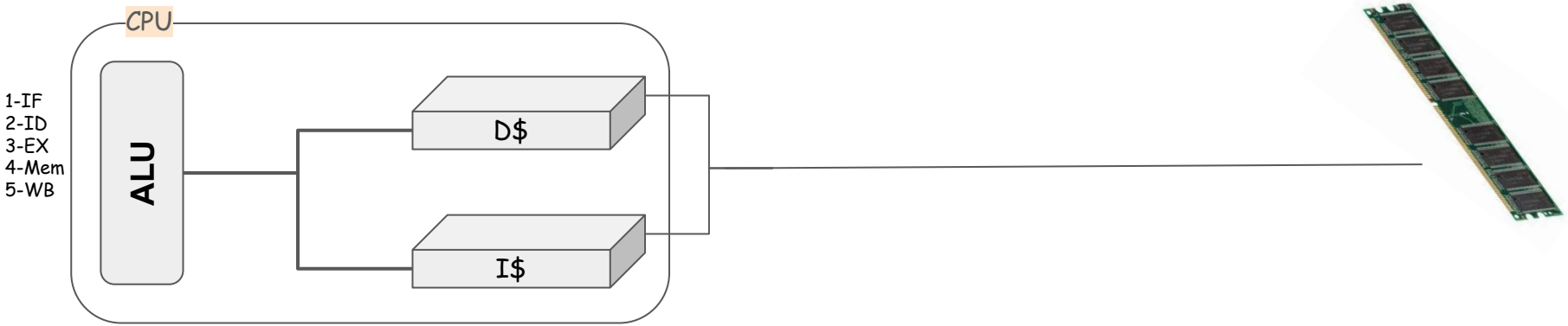
- Small:
  - Cache Size
  - Split Cache
- Simple:
  - Low Associativity
  - Split Cache



Performance

Average Memory Access Time

$$AMAT = HitTime + MissRate \times MissPenalty$$

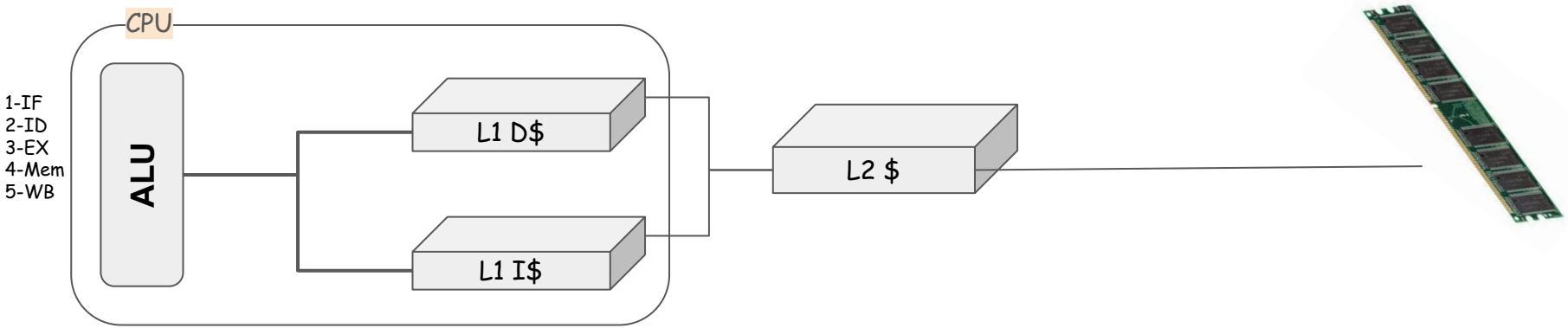


Performance

Average Memory Access Time

$$AMAT = HitTime + MissRate \times MissPenalty$$

Multi-Levels

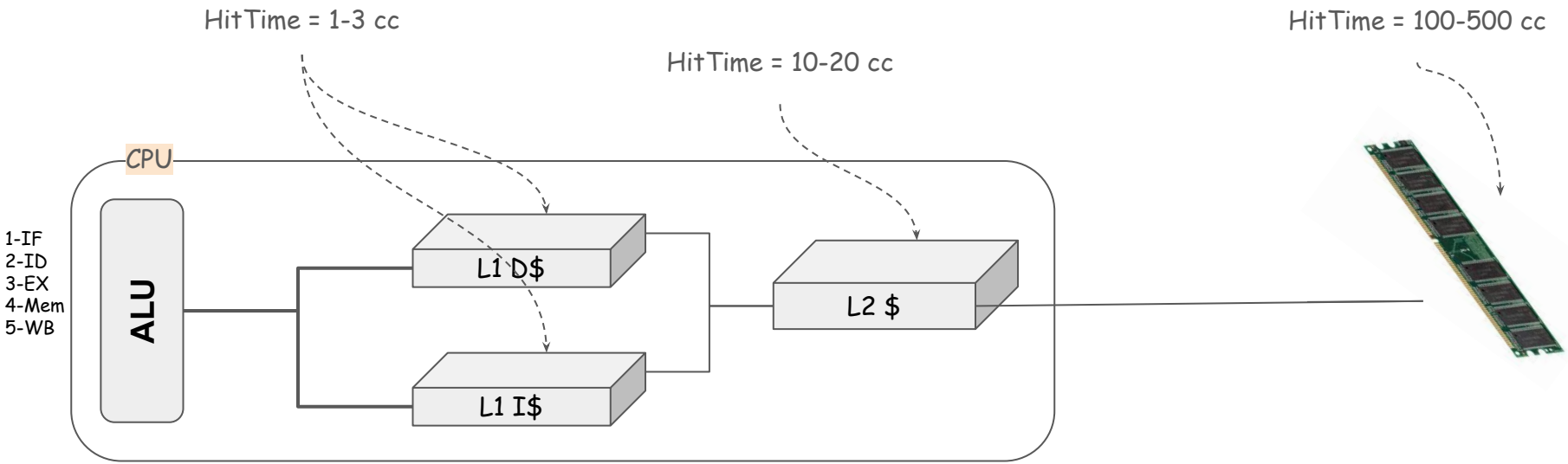


Performance

Average Memory Access Time

AMAT = HitTime + MissRate x MissPenalty

Multi-Levels

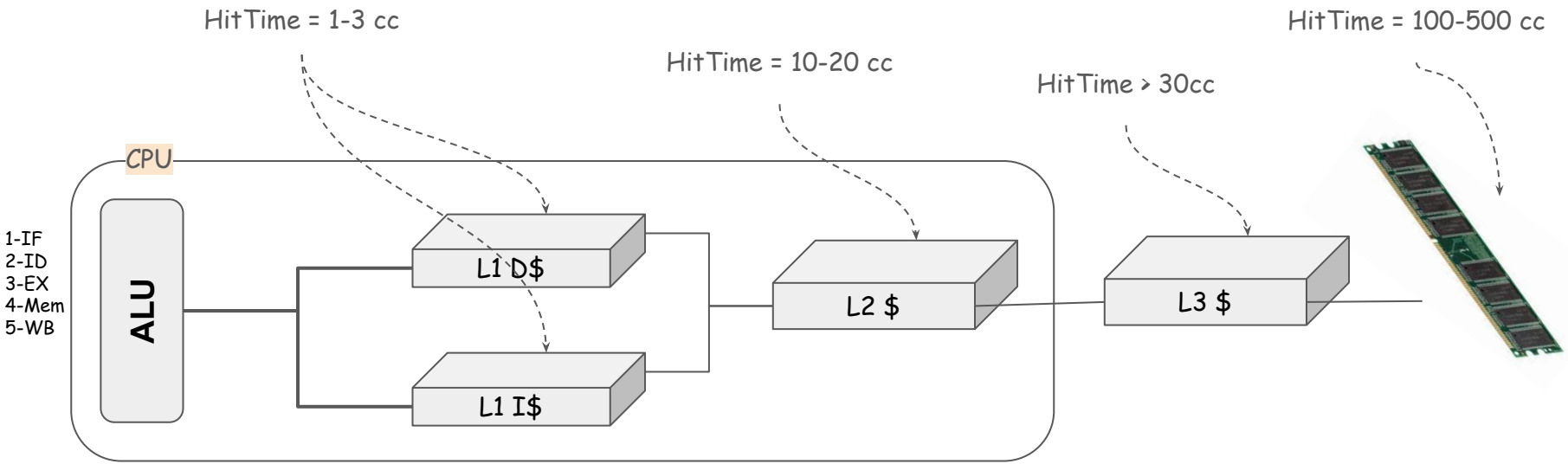


Performance

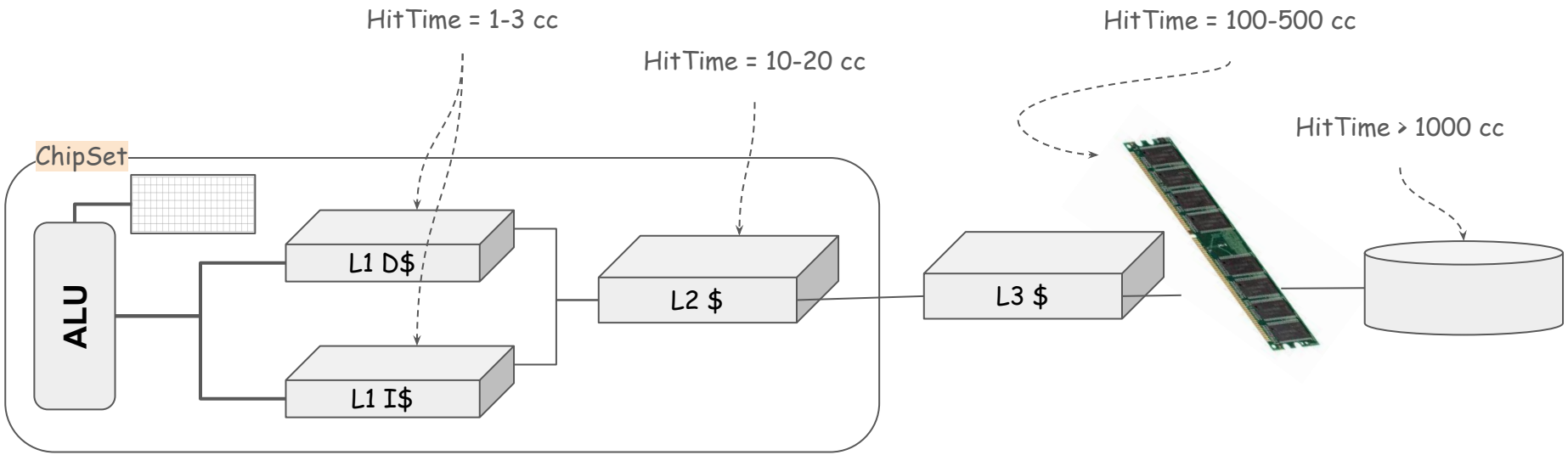
Average Memory Access Time

AMAT = HitTime + MissRate x MissPenalty

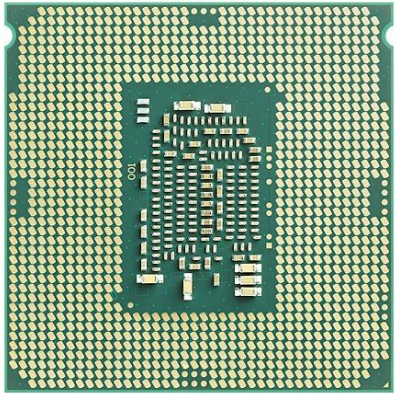
Multi-Levels

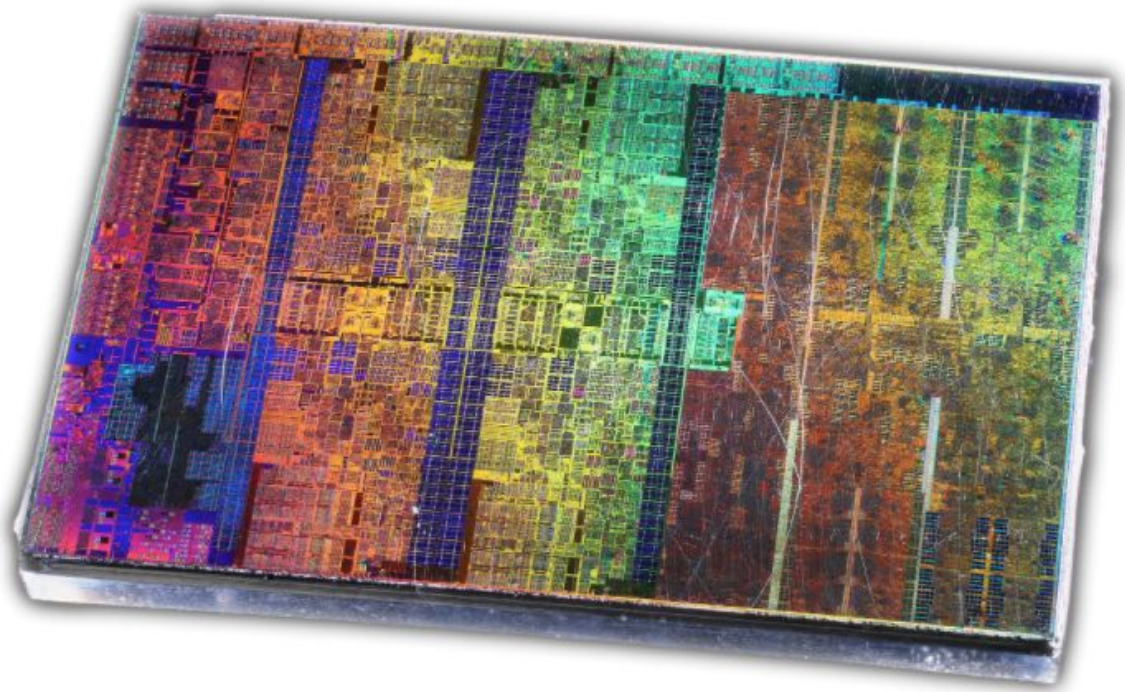


Performance



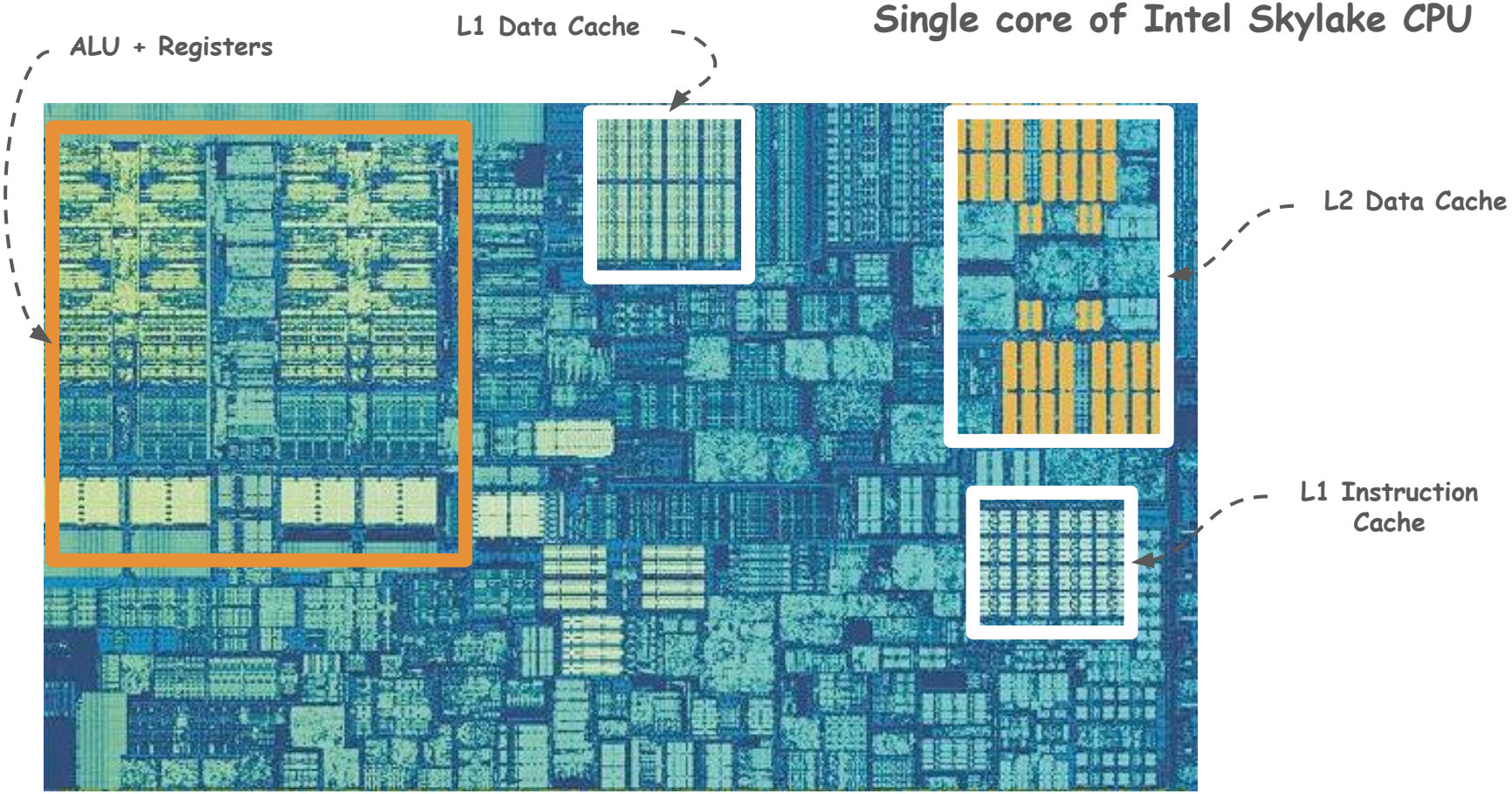
Intel Skylake CPU





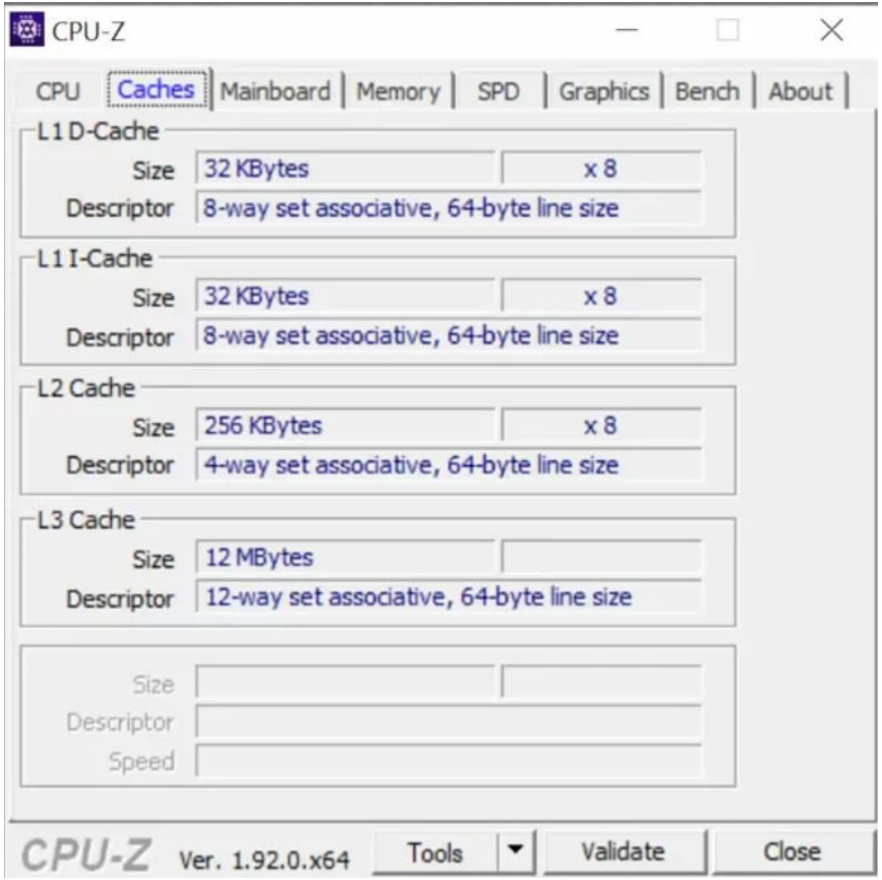
https://en.wikipedia.org/wiki/Skylake_(microarchitecture)

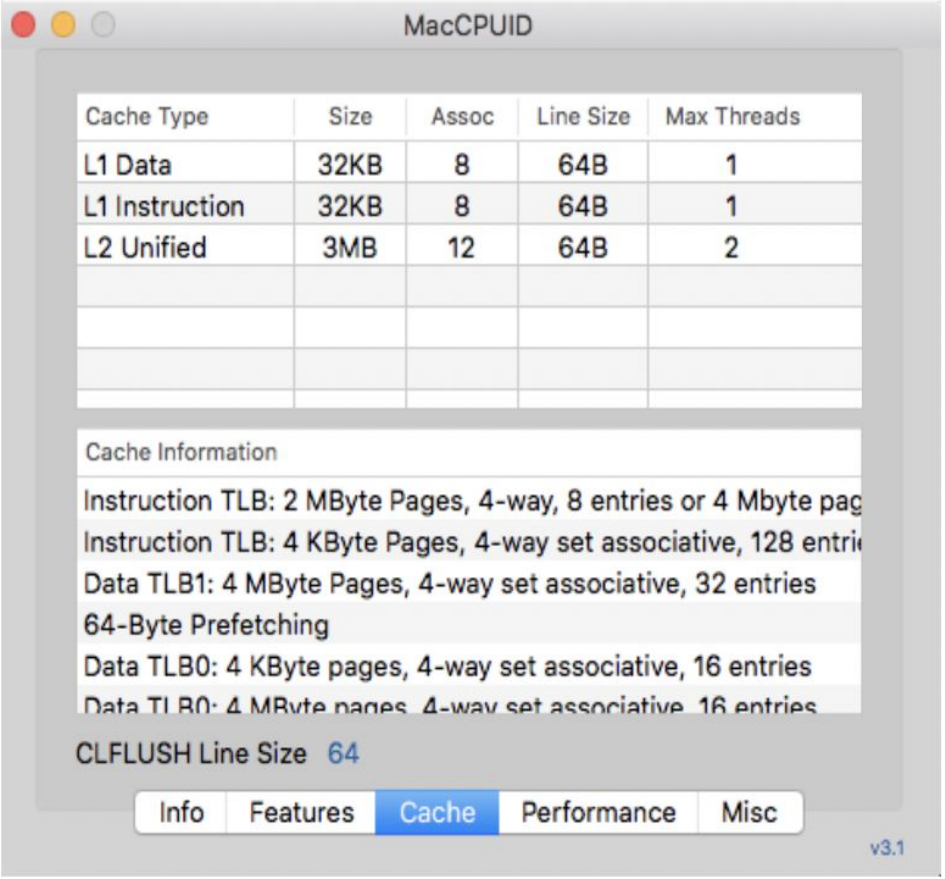
Single core of Intel Skylake CPU



Source: https://en.wikichip.org/wiki/File:skylake_core_die.png

Performance





Performance

MissRate = #CacheMisses / #CacheAccesses

Average Memory Access Time

$$\text{AMAT} = \text{HitTime} + \text{MissRate} \times \text{MissPenalty}$$

- Small
- Simple
- Low Associativity

- Large Block Size
- Large Cache
- High Associativity

- Small Block Size
- Multi-Level

Cache-Friendly Code

Which of this functions has good locality?

```
int sum_array_rows(int a[M][N]) {
    int i, j, sum = 0;

    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            sum += a[i][j];
    return sum;
}
```

```
int sum_array_cols(int a[M][N]){
    int i, j, sum = 0;

    for (j = 0; j < N; j++)
        for (i = 0; i < M; i++)
            sum += a[i][j];
    return sum;
}
```

0,0	0,1	0,2
1,0	1,1	1,2
2,0	2,1	2,2

0,0
0,1
0.2
1,0
1,1
1,2
2,0
2,1
2,2

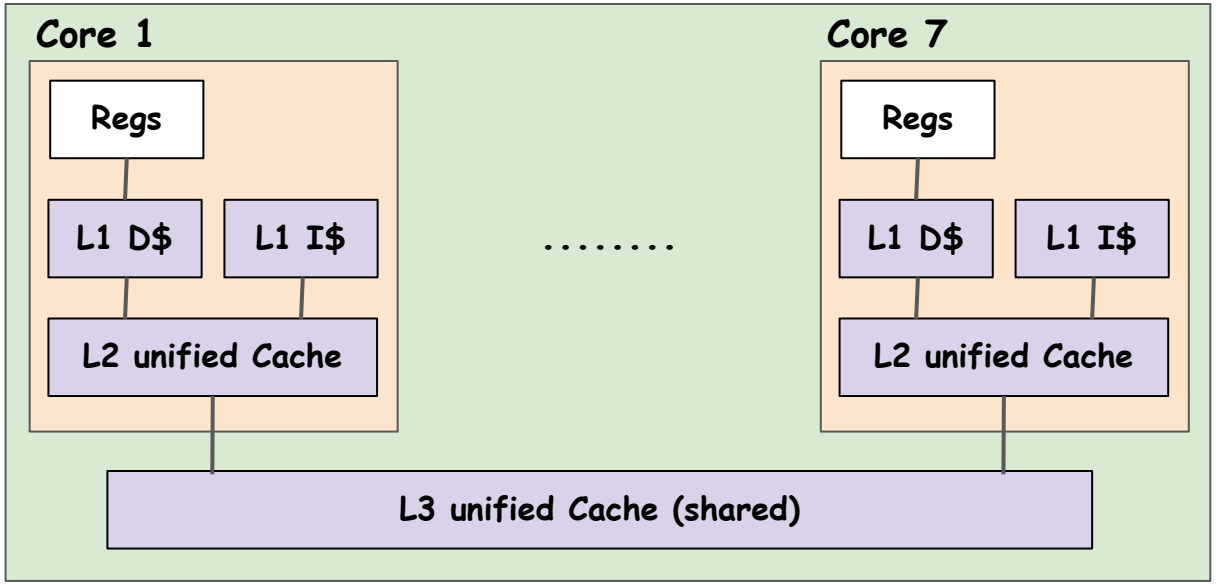
Programs with better locality will tend to have lower miss rates

Exercise:

(Core i7 CPU)

Address length (word size) = 48 bits, BlockSize = 64 Bytes)

L1 write through, L2/L3 write back



L1 (i-cache and d-cache)

Size= 32 KB

Associativity = 4-ways

Access = 4 Cycles

L2 Cache

Size= 256 KB

Associativity = 8-ways

Access = 11 Cycles

L3 Cache

Size= 8 MB

Associativity = 16-ways

Access = 30-40 Cycles

How many bits is the index?

How many bits is the tag?