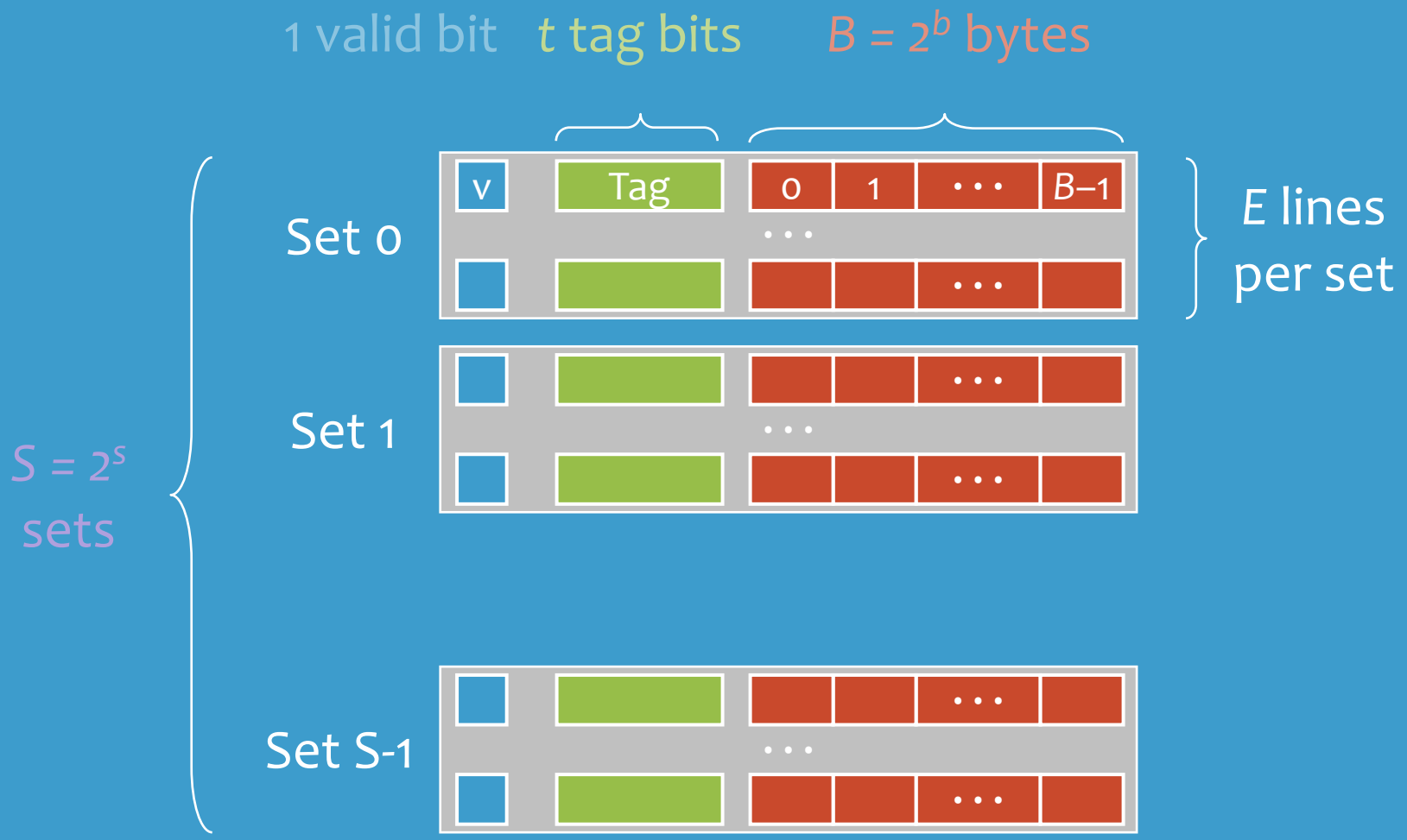
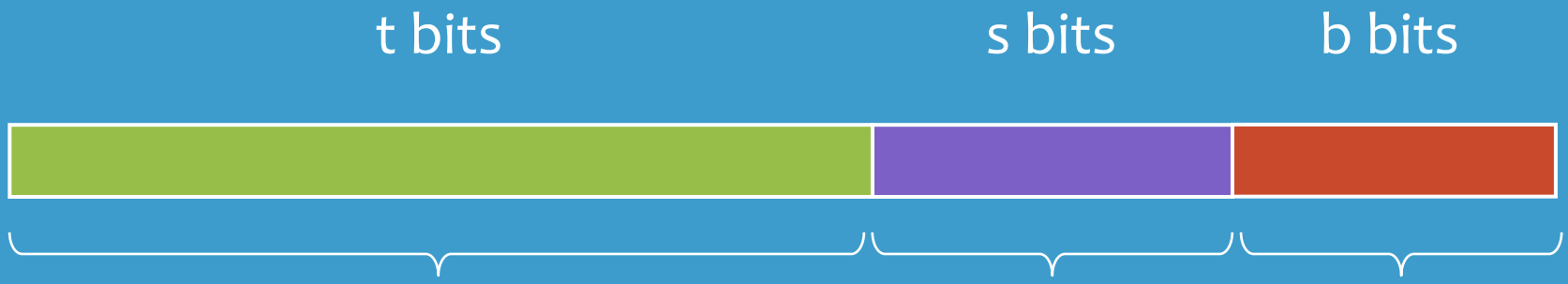
Cache organization



Direct-Mapping



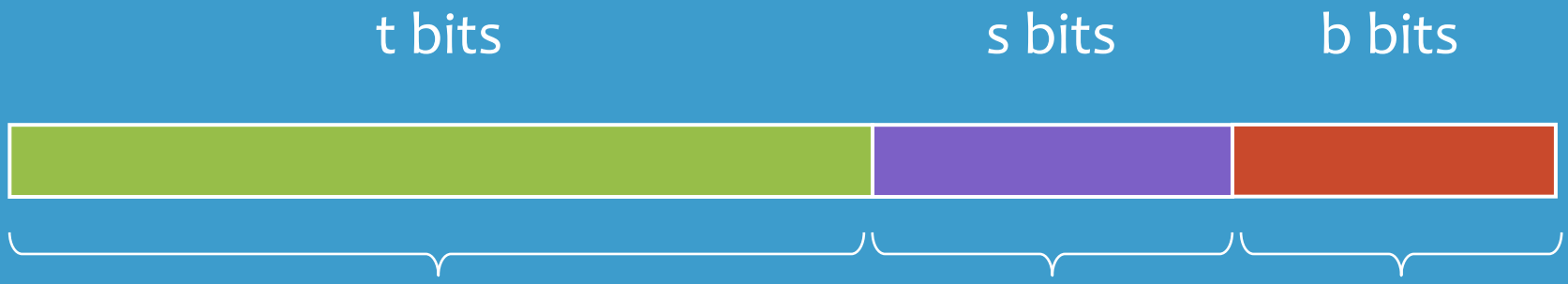
128B-cache, 12bit-addr, direct-mapped, 32B block

b= 5 bits; S = 128/32 = 4 sets; s = 2 bits; t=12-5-2=5 bits

ADDR	TAG (5 bits)	SET (2 bits)	OFFS (5 bits)	1ST	2ND
0X070					
0X080					
0X068					
0X190					
0X084					
0X178					
0X08C					
0XF00					
0X064					

V	Tag

Set-Associative



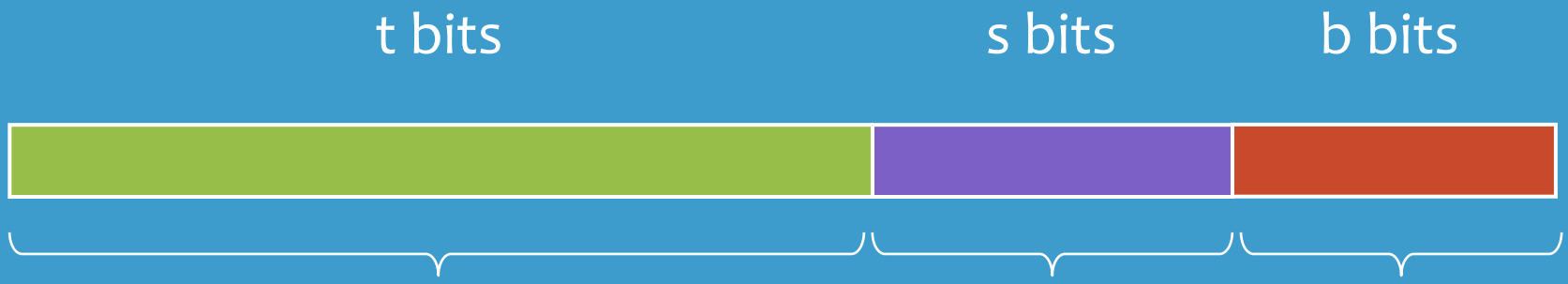
128B-cache, 12b-addr, 2-way set-associative, 32B block

b= 5 bits; S = 128/(32x2) = 2 sets; s =1 bit; t=12-5-1=6bits

ADDR	6b-T	1b-S	5b-off	1ST	2ND
0X070					
0X080					
0X068					
0X190					
0X084					
0X178					
0X08C					
0XF00					
0X064					

		Way 1		Way 2	
Set	LRU	V	Tag	V	Tag
0					
1					

Set-Associative



128B-cache, 12b-addr, 2-way set-associative, 16B block

b=4 bits; S=128/(16x2)=4 sets; s =2 bits; t=12-4-2=6bits

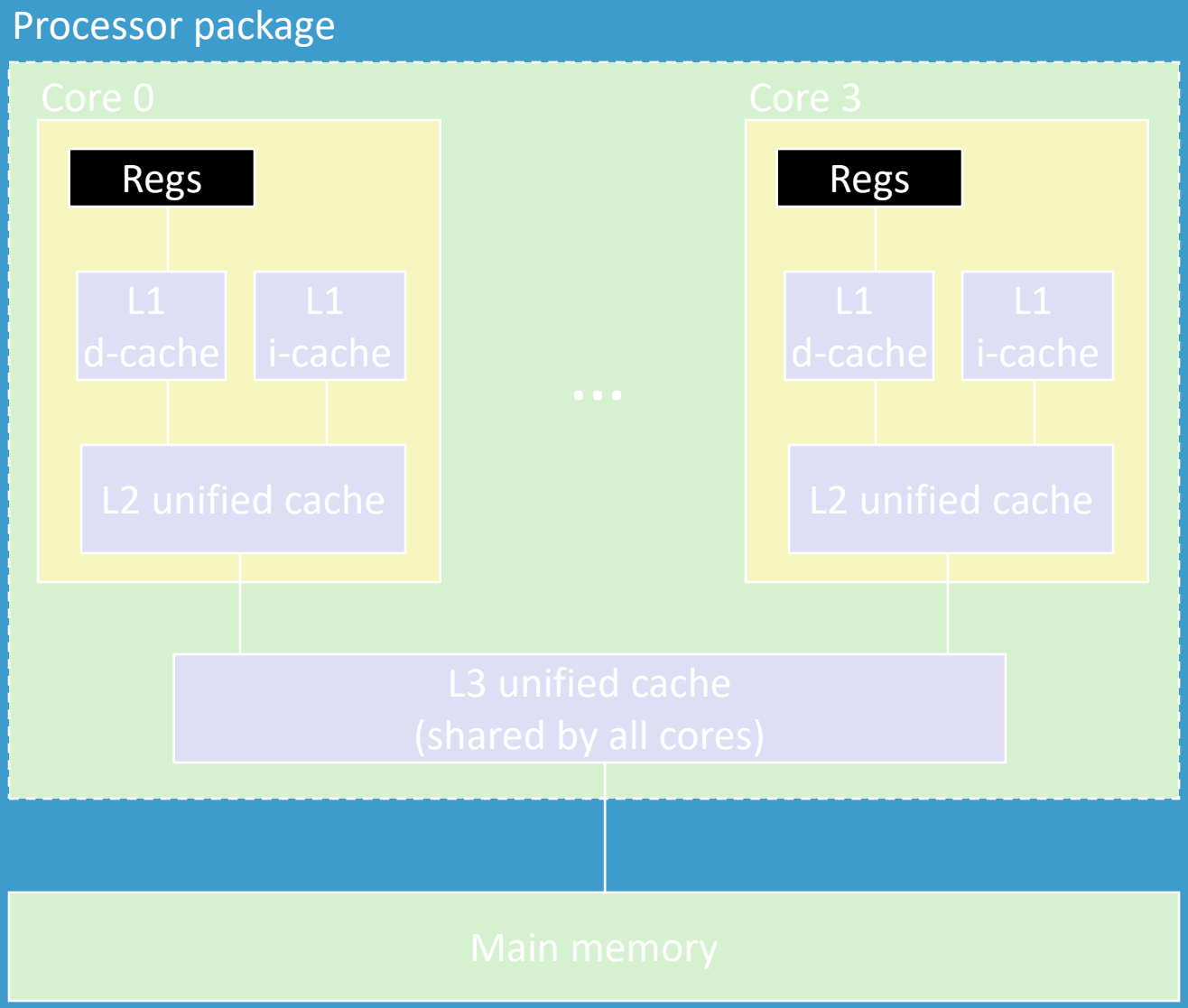
ADDR	6b-T	2b-S	4b-off	1ST	2ND
0X070					
0X080					
0X068					
0X190					
0X084					
0X178					
0X08C					
0XF00					
0X064					

		Way 1		Way 2	
Set	LRU	V	Tag	V	Tag
00					
01					
10					
11					

What about writes?

- Multiple copies of data exist:
  - L1, L2, L3, Main Memory, Disk
- What to do on a write-hit?
  - Write-through (write immediately to memory)
  - Write-back (defer write to memory until replacement of line)
    - Need a dirty bit (line different from memory or not)
- What to do on a write-miss?
  - Write-allocate (load into cache, update line in cache)
    - Good if more writes to the location follow
  - No-write-allocate (writes straight to memory, does not load into cache)
- Typical
  - Write-through + No-write-allocate
  - **Write-back + Write-allocate**

Intel Core i7 Cache Hierarchy



L1 i-cache and d-cache:
32 KB, 8-way,
Access: 4 cycles

L2 unified cache:
256 KB, 8-way,
Access: 10 cycles

L3 unified cache:
8 MB, 16-way,
Access: 40-75 cycles

Block size: 64 bytes for all caches.

Cache Performance Metrics

- Miss Rate
  - Fraction of memory references not found in cache (misses / accesses)
    = 1 – hit rate
  - Typical numbers (in percentages):
    - 3-10% for L1
    - can be quite small (e.g., < 1%) for L2, depending on size, etc.
- Hit Time
  - Time to deliver a line in the cache to the processor
    - includes time to determine whether the line is in the cache
  - Typical numbers:
    - 4 clock cycle for L1
    - 10 clock cycles for L2
- Miss Penalty
  - Additional time required because of a miss
    - typically 50-200 cycles for main memory (Trend: increasing!)

Let's think about those numbers

- Huge difference between a hit and a miss
  - Could be 100x, if just L1 and main memory

- Would you believe 99% hits is twice as good as 97%?
  - Consider:
    cache hit time of 1 cycle
    miss penalty of 100 cycles

  - Average access time:
    97% hits: 1 cycle + 0.03 * 100 cycles = **4 cycles**
    99% hits: 1 cycle + 0.01 * 100 cycles = **2 cycles**

- This is why "miss rate" is used instead of "hit rate"

Writing Cache Friendly Code

- Make the common case go fast
  - Focus on the inner loops of the core functions

- Minimize the misses in the inner loops
  - Repeated references to variables are good (temporal locality)
  - Stride-1 reference patterns are good (spatial locality)

Key idea: Our qualitative notion of locality is quantified through our understanding of cache memories

Gene Myron Amdahl

formulating Amdahl's law

“

$$S_{latency}(s) = \frac{1}{(1-F) + \frac{F}{S}}$$

”

Computer Performance

"X is N% faster than Y."

$$\frac{\text{Execution Time of Y}}{\text{Execution Time of X}} = 1 + \frac{N}{100}$$

Amdahl's law for overall speedup

$$\text{Overall Speedup} = \frac{1}{(1-F) + \frac{F}{S}}$$

F = The fraction enhanced

S = The speedup of the enhanced fraction

Using Amdahl's law

Overall speedup if we make 90% of a program run 10 times faster.

F = 0.9 S = 10

$$\text{Overall Speedup} = \frac{1}{(1-0.9)+\frac{0.9}{10}} = \frac{1}{0.1+0.09} = 5.26$$

Overall speedup if we make 80% of a program run 20% faster.

F = 0.8 S = 1.2

$$\text{Overall Speedup} = \frac{1}{(1-0.8)+\frac{0.8}{1.2}} = \frac{1}{0.2+0.66} = 1.153$$

Summary

- Cache Concepts
  - Cache Hit
  - Cache Miss
- Cache Organization
  - Direct-mapped
  - E-way Set Associative
  - Fully Associative