CSC 112: Computer Operating Systems
Lecture 7

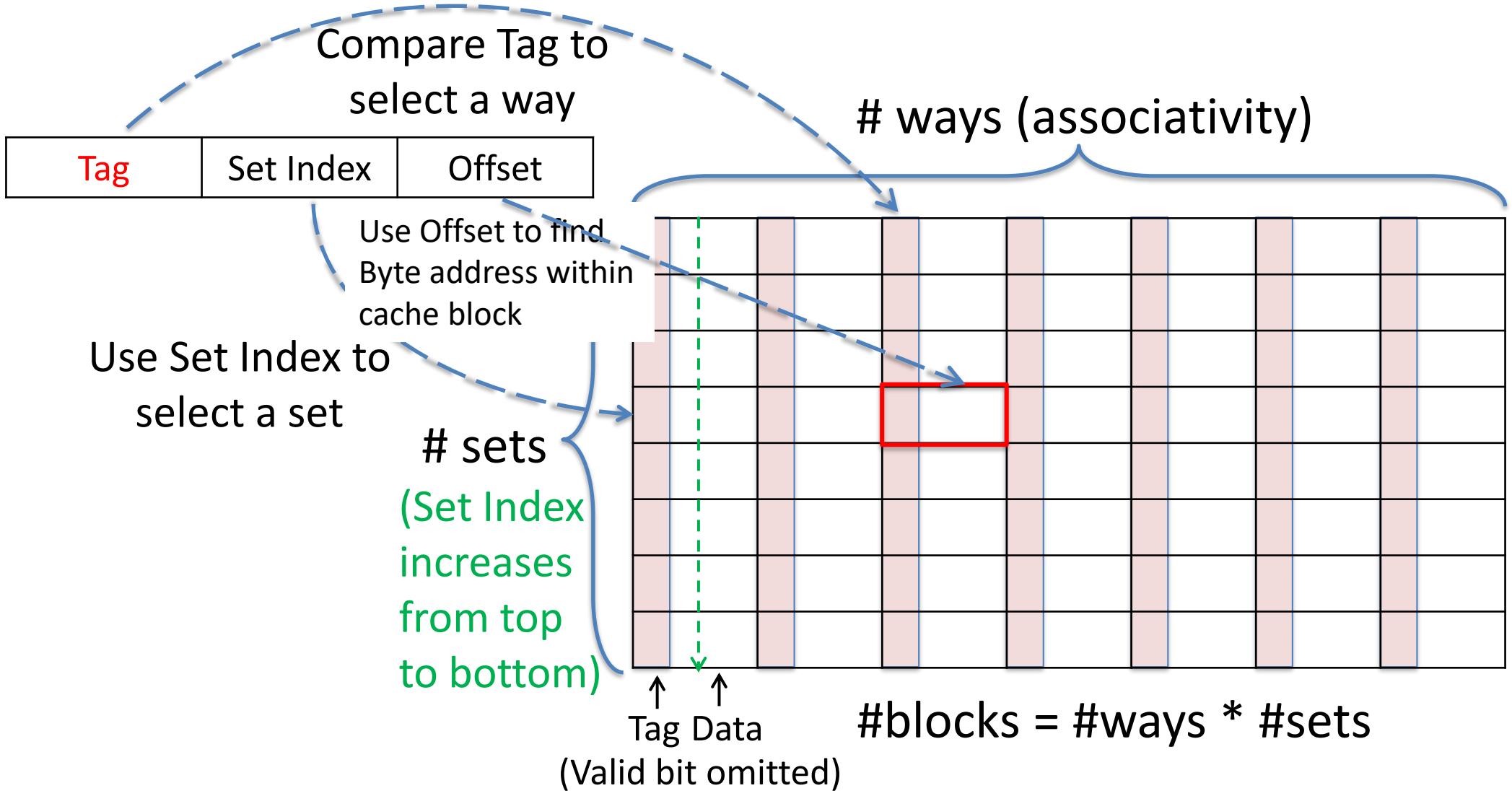
Memory System I: Cache
Exercises ANS

Department of Computer Science,

Hofstra University

Review

Cache Organization



Review

Key Equations

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Tag size does not affect cache capacity; depends on memory address length

SI size determines # sets = $2^{\text{SI size}}$

Offset size determines Bytes/block = $2^{\text{Offset size}}$

Review

Decimal, Binary and Hex

Decimal	Binary	Hex
0	0000	0x0
1	0001	0x1
2	0010	0x2
3	0011	0x3
4	0100	0x4
5	0101	0x5
6	0110	0x6
7	0111	0x7
8	1000	0x8
9	1001	0x9
10	1010	0xA
11	1011	0xB
12	1100	0xC
13	1101	0xD
14	1110	0xE
15	1111	0xF

Prefix 0x denotes hex

4

Quiz

- Memory hierarchies take advantage of spatial locality by keeping the most recent data items closer to the processor.
  - False. This is called temporal locality.
- For a given cache size, a larger block size may cause lower hit rate than a smaller one.
  - True. The relationship between block size and hit rate is non-monotonic. A large block size leads to fewer cache blocks, so it may cause lower hit rate since useless junk may be brought into cache along with useful data. But it may lead to higher hit rate if the program has good locality.
- If you know your computer's cache size, you can often make your code run faster.
  - True. By tuning your code to be cache-aware.

Quiz

- Q: How many 32-bit integers can be stored in a DM cache with 15 tag bits, 15 index bits, and 2 offset bits?
- A: Each cache block is 2^2=4 Bytes and can store one 32-bit integer. The cache has a total number of 2^15=32K blocks, hence it can store 32K integers. (The tag bits are irrelevant here since it is related to memory size, not cache size)

Quiz I

- Q: Consider 32-bit address space; a direct-mapped cache with size 16KB;each cache block is 4 words. What is the TIO breakdown?
- A:
- Cache size = 16KB = 16 * 2^10 bytes
- Cache block size = 4 words = 4 * 4 bytes = 16 bytes = 2^4
- Number of cache blocks= 16 * 2^10 bytes / 16 bytes = 2^10
- Index bits = 10
- Offset bits = 4
- Tag bits = 32 – 10 – 4 = 18

7

Quiz II

- Q: Consider 32-bit address space; a two-way set-associative cache with size 16KB;each cache block is 4 words. What is the TIO breakdown?
- A:
- Cache size = 16 * 2^10 bytes
- cache block size = 16 bytes
- Set size = cache block size * set associativity = 16 bytes * 2 = 32 bytes
- Number of sets = 16 * 2^10 bytes / 32 bytes = 2^9
- Index bits = 9
- Offset bits = 4
- Tag bits = 32 – 9 – 4 = 19

Quiz

- Q: Consider 32-bit address space; a DM cache with size 32 KB; each cache block is 8 words. What is the TIO breakdown?
- ANS: T=17, I=10, O=5
  - 8-word per block => 32 bytes / block => O = 5
  - 32 KB / (32 bytes / block) = 2^10 blocks total => I = 10
  - T = 32 – 10 – 5 = 17
- Q: Consider 32-bit address space; a 4-way SA cache with size 32 KB; each cache block is 8 words. What is the TIO breakdown?
- ANS: T=19, I=8, O=5 (Tag 19b, Index 8b, Offset 5b)
  - 8-word per block => 32 bytes / block => O = 5
  - 32 KB / (32 bytes / block) = 2^10 blocks total
  - 2^10 blocks / (4 blocks / set) = 2^8 sets total => I = 8
  - T = 32 – 8 – 5 = 19
- Q: Consider 32-bit address space; an FA cache with size 32 KB; each cache block is 8 words. What is the TIO breakdown?
  - ANS: T=27, I=0, O=5
  - 8-word per block => 32 bytes / block => O = 5
  - FA cache => I = 0
  - T = 32 – 0 – 5 = 27

Quiz

- Q: Consider 32-bit address space; a DM cache with size 16 KB; each cache block is 4 words. What is the TIO breakdown?
- ANS: T=18, I=10, O=4
  - 4-word per block => 16 bytes / block => O = 4
  - 16 KB / (16 bytes / block) = 2^10 blocks total => I = 10
  - T = 32 – 10 – 4 = 18
- Q: Consider 32-bit address space; a 2-way SA cache with size 16KB; each cache block is 4 words. What is the TIO breakdown?
- ANS: T=19, I=9, O=4
  - 4-word per block => 16 bytes / block => O = 4
  - 16 KB / (16 bytes / block) = 2^10 blocks total
  - 2^10 blocks / (2 blocks / set) = 2^9 sets total => I = 9
  - T = 32 – 9 – 4 = 19
- Q: Consider 32-bit address space; an FA cache with size 16KB; each cache block is 4 words. What is the TIO breakdown?
  - ANS: T=27, I=0, O=5
  - 4-word per block => 16 bytes / block => O = 4
  - FA cache => I = 0
  - T = 32 – 0 – 4 = 28

IMPORTANT

Q: 12-bit DM Cache

- Consider 12-bit memory address; DM cache with block size 4B; total of 16 cache blocks, with contents shown below ("—“ means invalid data). All values are in hex. Within each block, B0 refers to Byte address 00, B1 refers to Byte address 01, and so on.
  - 1. What are the sizes of Tag, Set Index, Offset?
  - 2. Cache hit or miss for referencing the following memory addresses (individually, not sequentially) ? If cache hit, give the actual value returned: 0x7AC, 0x024, 0x99F

Direct-Mapped:

Set	Valid	Tag	B0	B1	B2	B3
0	1	15	63	B4	C1	A4
1	0	—	—	—	—	—
2	0	—	—	—	—	—
3	1	D	DE	AF	BA	DE
4	0	—	—	—	—	—
5	0	—	—	—	—	—
6	1	13	31	14	15	93
7	0	—	—	—	—	—

Set	Valid	Tag	B0	B1	B2	B3
8	0	—	—	—	—	—
9	1	0	01	12	23	34
A	1	1	98	89	CB	BC
B	0	1E	4B	33	10	54
C	0	—	—	—	—	—
D	1	11	C0	04	39	AA
E	0	—	—	—	—	—
F	1	F	FF	6F	30	0

IMPORTANT

A: 12-bit DM Cache

Direct-Mapped:

Set	Valid	Tag	B0	B1	B2	B3
0	1	15	63	B4	C1	A4
1	0	–	–	–	–	–
2	0	–	–	–	–	–
3	1	D	DE	AF	BA	DE
4	0	–	–	–	–	–
5	0	–	–	–	–	–
6	1	13	31	14	15	93
7	0	–	–	–	–	–

Set	Valid	Tag	B0	B1	B2	B3
8	0	–	–	–	–	–
9	1	0	01	12	23	34
A	1	1	98	89	CB	BC
B	0	1E	4B	33	10	54
C	0	–	–	–	–	–
D	1	11	C0	04	39	AA
E	0	–	–	–	–	–
F	1	F	FF	6F	30	0

- 1. What are the sizes of Tag, Set Index, Offset?
- # Bytes/block=4, hence Offset size=2
- # Sets=(# Blocks for DM cache)=16, hence SI size=4
- Tag size=12-4-2=6

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

IMPORTANT

A: 12-bit DM Cache

- 2. Cache hit or miss for referencing the following memory addresses? If cache hit, give the actual value returned: 0x7AC, 0x024, 0x99F
- 0x7AC = 0111 1010 1100 (bin). Set Index=1011(bin)=0xB. The set with index 0xB has a single block with Valid=0, hence it is a cache miss (no need to check for tag match. Even though the table shows some data in this block, all data is invalid with Valid=0).
- 0x024 = 0000 0010 0100 (bin). Set Index=1001(bin)=0x9. The set with index 0x9 has a single block with Valid=1, and the Tag 000000 (bin) = 0x0 matches, hence it is a cache hit. The Byte offset is 00, hence the actual data returned is 0x01 contained in B0.
- 0x99F = 1001 1001 1111 (bin). Set Index=0111(bin)=0x7. The set with index 0x7 has a single block with Valid=0, hence it is a cache miss (no need to check for tag match).

Direct-Mapped:

Set	Valid	Tag	B0	B1	B2	B3
0	1	15	63	B4	C1	A4
1	0	–	–	–	–	–
2	0	–	–	–	–	–
3	1	D	DE	AF	BA	DE
4	0	–	–	–	–	–
5	0	–	–	–	–	–
6	1	13	31	14	15	93
7	0	–	–	–	–	–

Set	Valid	Tag	B0	B1	B2	B3
8	0	–	–	–	–	–
9	1	0	01	12	23	34
A	1	1	98	89	CB	BC
B	0	1E	4B	33	10	54
C	0	–	–	–	–	–
D	1	11	C0	04	39	AA
E	0	–	–	–	–	–
F	1	F	FF	6F	30	0

IMPORTANT

Q: 12-bit 2-way SA Cache

- Consider 12-bit memory address; 2-way SA cache with block size 4B; total of 16 cache blocks with contents shown below ("—“ means invalid data). All values are in hex.
  - 1. What are the sizes of Tag, Set Index, Offset?
  - 2. If cache hit, give the actual value returned: 0x435, 0x388, 0x0D3

2-way Set Associative:

Set	Valid	Tag	B0	B1	B2	B3
0	0	—	—	—	—	—
1	0	—	—	—	—	—
2	1	3	4F	D4	A1	3B
3	0	—	—	—	—	—
4	0	6	CA	FE	F0	0D
5	1	21	DE	AD	BE	EF
6	0	—	—	—	—	—
7	1	11	00	12	51	55

Set	Valid	Tag	B0	B1	B2	B3
0	0	—	—	—	—	—
1	1	2F	01	20	40	03
2	1	0E	99	09	87	56
3	0	—	—	—	—	—
4	0	—	—	—	—	—
5	0	—	—	—	—	—
6	1	37	22	B6	DB	AA
7	0	—	—	—	—	—

IMPORTANT

A: 12-bit 2-way SA Cache

2-way Set Associative:

Set	Valid	Tag	B0	B1	B2	B3
0	0	–	–	–	–	–
1	0	–	–	–	–	–
2	1	3	4F	D4	A1	3B
3	0	–	–	–	–	–
4	0	6	CA	FE	F0	0D
5	1	21	DE	AD	BE	EF
6	0	–	–	–	–	–
7	1	11	00	12	51	55

Set	Valid	Tag	B0	B1	B2	B3
0	0	–	–	–	–	–
1	1	2F	01	20	40	03
2	1	0E	99	09	87	56
3	0	–	–	–	–	–
4	0	–	–	–	–	–
5	0	–	–	–	–	–
6	1	37	22	B6	DB	AA
7	0	–	–	–	–	–

- 1. What are the sizes of Tag, Set Index, Offset?
- # Bytes/block=4, hence Offset size=2
- # Sets=#blocks/#ways=16/2=8, hence SI size=3
- Tag size=12-3-2=7

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

IMPORTANT

A: 12-bit 2-way SA Cache

- 2. Cache hit or miss for referencing the following memory addresses? If cache hit, give the actual value returned: 0x435, 0x388, 0x0D3
- 0x435 = 0100 0011 0101 (bin). Set Index=101(bin)=0x5. The set with index 0x5 has 2 blocks, but only one block with Valid=1. The Tag 0100001 (bin) = 0x21 matches the valid block Tag, hence it is a cache hit. The Byte offset is 01, hence the actual data returned is 0xAD contained in B1.
- 0x388 = 0011 1000 1000 (bin). Set Index=010(bin)=0x2. The set with index 0x2 has 2 blocks, both with Valid=1, but the Tag 0011100 (bin) = 0x1C does not match any valid block Tag (0x03, 0x0E), hence it is a cache miss.
- 0x0D3 = 0000 1101 0011 (bin). Set Index=100(bin)=0x4. The set with index 0x4 has 2 blocks, both with Valid=0, hence it is a cache miss (no need to check for tag match).

2-way Set Associative:

Set	Valid	Tag	B0	B1	B2	B3
0	0	–	–	–	–	–
1	0	–	–	–	–	–
2	1	3	4F	D4	A1	3B
3	0	–	–	–	–	–
4	0	6	CA	FE	F0	0D
5	1	21	DE	AD	BE	EF
6	0	–	–	–	–	–
7	1	11	00	12	51	55

Set	Valid	Tag	B0	B1	B2	B3
0	0	–	–	–	–	–
1	1	2F	01	20	40	03
2	1	0E	99	09	87	56
3	0	–	–	–	–	–
4	0	–	–	–	–	–
5	0	–	–	–	–	–
6	1	37	22	B6	DB	AA
7	0	–	–	–	–	–

IMPORTANT

Q: 12-bit FA Cache

- Consider 12-bit memory address; FA cache with block size 4B; contents shown below ("—“ means invalid data). All values are in hex.
  - 1. What are the sizes of Tag, Set Index, Offset?
  - 2. If cache hit, give the actual value returned: 0x1DD, 0x719, 0x2AA

Fully Associative:

Set	Valid	Tag	B0	B1	B2	B3
0	1	1F4	00	01	02	03
0	0	—	—	—	—	—
0	1	100	F4	4D	EE	11
0	1	77	12	23	34	45
0	0	—	—	—	—	—
0	1	101	DA	14	EE	22
0	0	—	—	—	—	—
0	1	16	90	32	AC	24

Set	Valid	Tag	B0	B1	B2	B3
0	0	—	—	—	—	—
0	1	AB	02	30	44	67
0	1	34	FD	EC	BA	23
0	0	—	—	—	—	—
0	1	1C6	00	11	22	33
0	1	45	67	78	89	9A
0	1	1	70	00	44	A6
0	0	—	—	—	—	—

IMPORTANT

A: 12-bit FA Cache

Fully Associative:

Set	Valid	Tag	B0	B1	B2	B3
0	1	1F4	00	01	02	03
0	0	–	–	–	–	–
0	1	100	F4	4D	EE	11
0	1	77	12	23	34	45
0	0	–	–	–	–	–
0	1	101	DA	14	EE	22
0	0	–	–	–	–	–
0	1	16	90	32	AC	24

Set	Valid	Tag	B0	B1	B2	B3
0	0	–	–	–	–	–
0	1	AB	02	30	44	67
0	1	34	FD	EC	BA	23
0	0	–	–	–	–	–
0	1	1C6	00	11	22	33
0	1	45	67	78	89	9A
0	1	1	70	00	44	A6
0	0	–	–	–	–	–

- 1. What are the sizes of Tag, Set Index, Offset?
- # Bytes/block=4, hence Offset size=2
- # Sets=1 for FA cache, hence SI size=0
- Tag size=12-0-2=10

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

IMPORTANT

A: 12-bit FA Cache

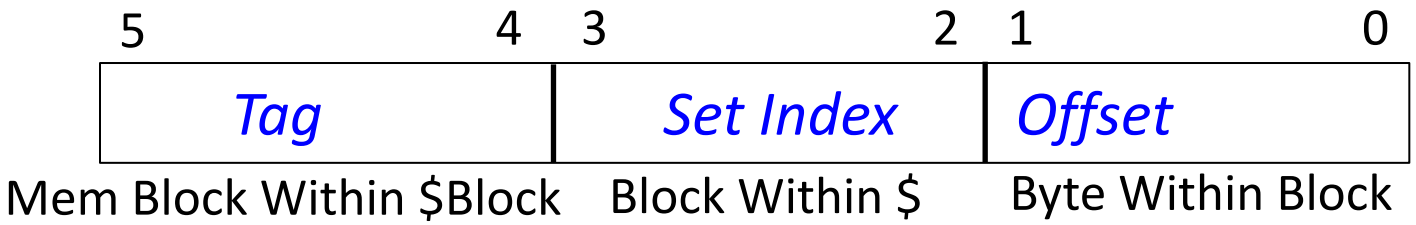
- 2. Cache hit or miss for referencing the following memory addresses? If cache hit, give the actual value returned: 0x1DD, 0x719, 0x2AA
- 0x1DD = 00001 1101 1101 (bin). The Tag 00001110111 (bin) = 0x77, which matches a block with Valid=1, hence it is a cache hit. The Byte offset is 01, hence the actual data returned is 0x23 contained in B1
- 0x719 = 0111 0001 1001 (bin). The Tag 0111000110 (bin) = 0x1C6, which matches a block with Valid=1, hence it is a cache hit. The Byte offset is 01, hence the actual data returned is 0x11 contained in B1
- 0x2AA = 0010 1010 1010 (bin). The Tag 0010101010 (bin) = 0xAA, which does not match any block with Valid=1, hence it is a cache miss.

Fully Associative:

Set	Valid	Tag	B0	B1	B2	B3
0	1	1F4	00	01	02	03
0	0	–	–	–	–	–
0	1	100	F4	4D	EE	11
0	1	77	12	23	34	45
0	0	–	–	–	–	–
0	1	101	DA	14	EE	22
0	0	–	–	–	–	–
0	1	16	90	32	AC	24

Set	Valid	Tag	B0	B1	B2	B3
0	0	–	–	–	–	–
0	1	AB	02	30	44	67
0	1	34	FD	EC	BA	23
0	0	–	–	–	–	–
0	1	1C6	00	11	22	33
0	1	45	67	78	89	9A
0	1	1	70	00	44	A6
0	0	–	–	–	–	–

Question: Tag



- Assume: DM cache; 6-bit memory address: 2-bit Tag, 2-bit index, 2-bit Offset. Compute cache capacity and memory size.
  - 2-bit Offset => Bytes/block = 4;
  - # sets = $2^{\text{SI Size}}$ = 4
  - # cache blocks = # ways * # sets = 1*4 = 4
  - cache capacity = # cache blocks * Bytes/block = 4*4 = 16B
- Memory size: 2^4 (2-bit tag +2-bit SI) = 16 blocks = 64 Bytes

Recall:

ways = # blocks/cache set = associativity
cache blocks = # ways * # sets
cache capacity = # cache blocks * Bytes/block

Question: T-SI-O Distribution

- Consider 32-bit memory address, DM cache with size 64KB, 16 Bytes/block. What are the bit-widths of Tag-SetIndex-Offset?

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Answer: T-SI-O Distribution

- Consider 32-bit memory address, DM cache with size 64KB, 16 Bytes/block. What are the bit-widths of Tag-Set Index-Offset?
- A: 16 Bytes/block → Offset size=4
- For DM cache, # Sets = # blocks = 64 KB/16 Bytes/block = 4K → SI size=12
- Tag size = 32-12-4=16

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Question: T-SI-O Distribution

- Consider 32-bit memory address, 8-way SA cache with size 64KB, 16 Bytes/block. What are the bit-widths of Tag-Set Index-Offset?

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Answer: T-SI-O Distribution

- Consider 32-bit memory address, 8-way SA cache with size 64KB, 16 Bytes/block. What are the bit-widths of Tag-Set Index-Offset?
- A: 16 Bytes/block → Offset size=4
- For 8-way SA cache, # Sets = # blocks/8 = (64 KB/16 Bytes/block)/8 = 0.5K → SI size=9
- Tag size = 32-9-4=19

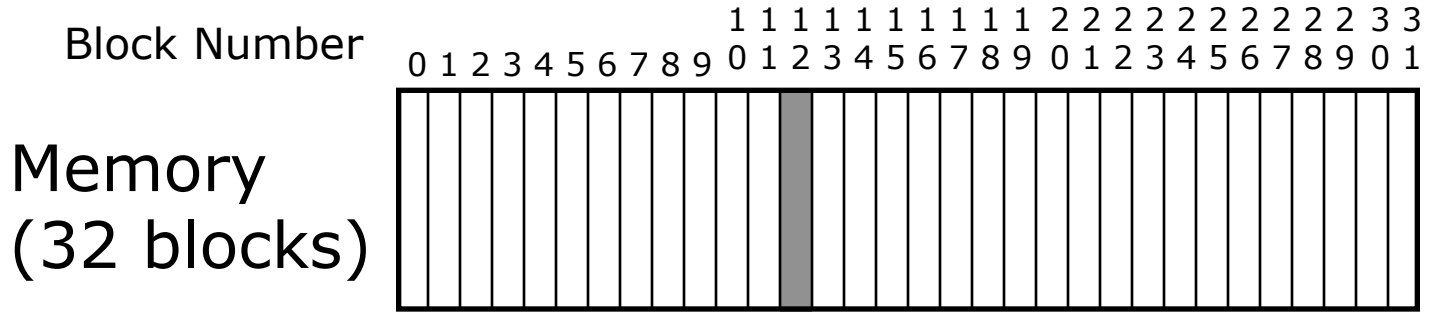
Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

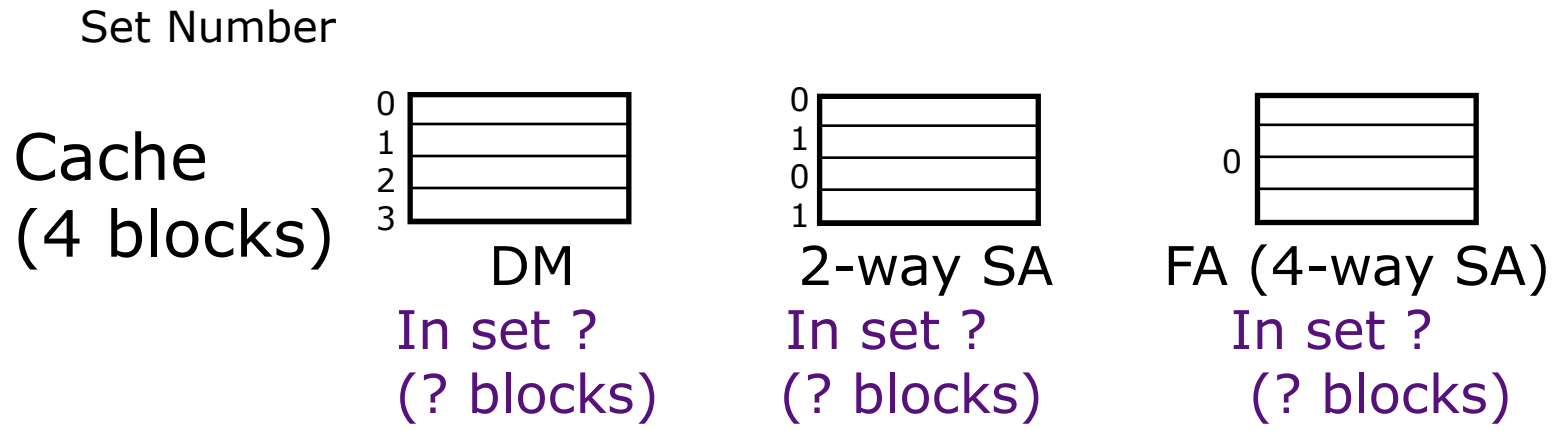
blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Q: Alternative Cache Organizations (4-block cache)



Where are possible locations in cache that block #12 in memory can be placed?



Recall:

ways = # blocks/cache set = associativity

cache blocks = # ways * # sets

cache capacity = # cache blocks * Bytes/block

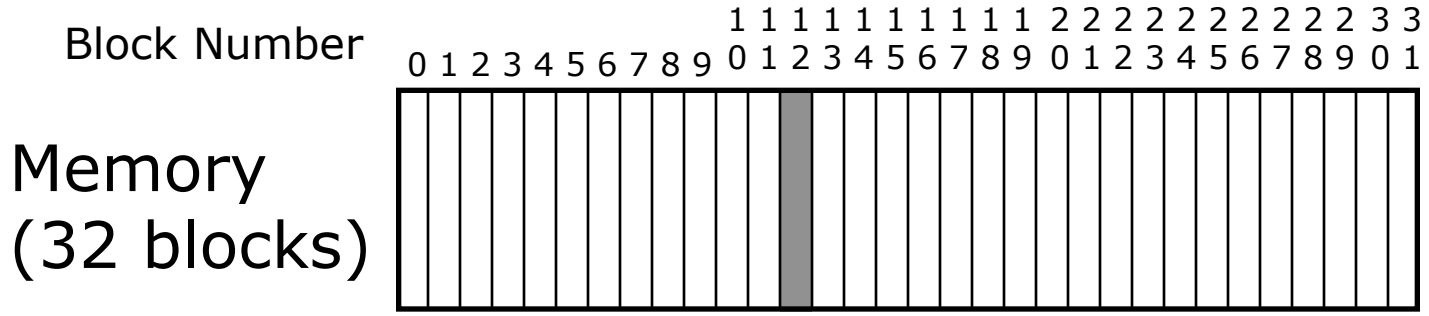
A: Alternative Cache Organizations (4-block cache)

- Memory block #12 (decimal) corresponds to memory address 01100XXXX in binary (we don't care about block size)
- For DM cache:
  - # cache blocks = 4 = # ways (1) * # sets
  - => # sets = 4 = 2^2 => Set Index has 2b => Set Index is 00 (last 2b in 01100)
  - Tag (3b); Set Index (2b)
- For 2-way SA cache:
  - # cache blocks = 4 = # ways (2) * # sets
  - => # sets = 2 = 2^1 => Set Index has 1b => Set Index is 0 (last 1b in 01100)
  - Tag (4b); Set Index (1b)
- For FA (4-way SA) cache:
  - # cache blocks = 4 = # ways (4) * # sets
  - => # sets = 1 = 2^0 => No Set Index
  - Tag (5b)

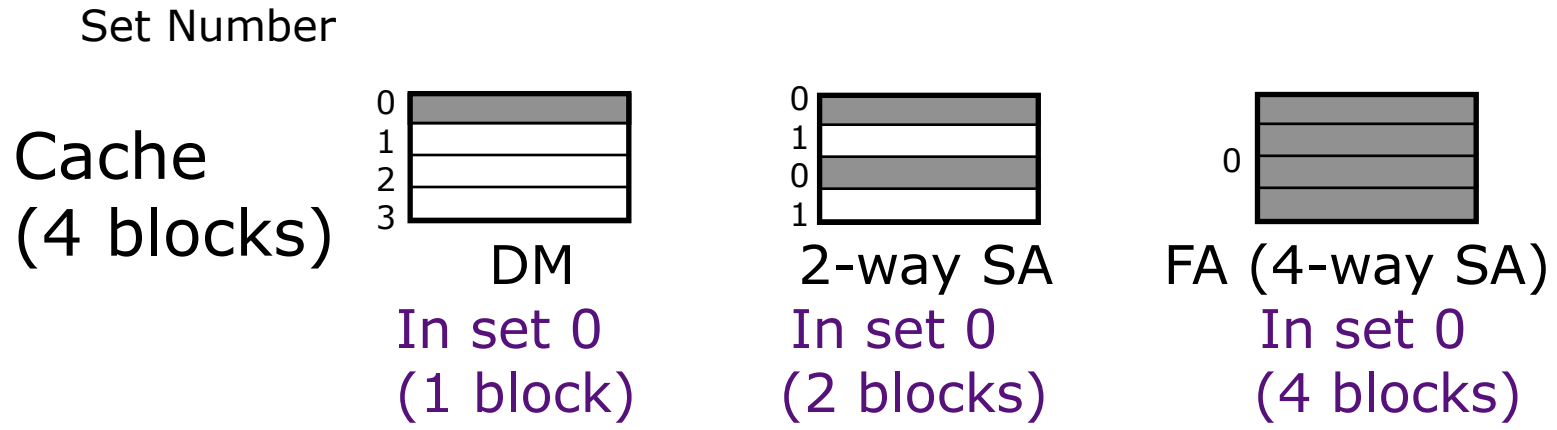
Recall:

ways = # blocks/cache set = associativity
cache blocks = # ways * # sets
cache capacity = # cache blocks * Bytes/block

A: Alternative Cache Organizations (4-block cache)



Where are possible locations in cache that block #12 in memory can be placed?



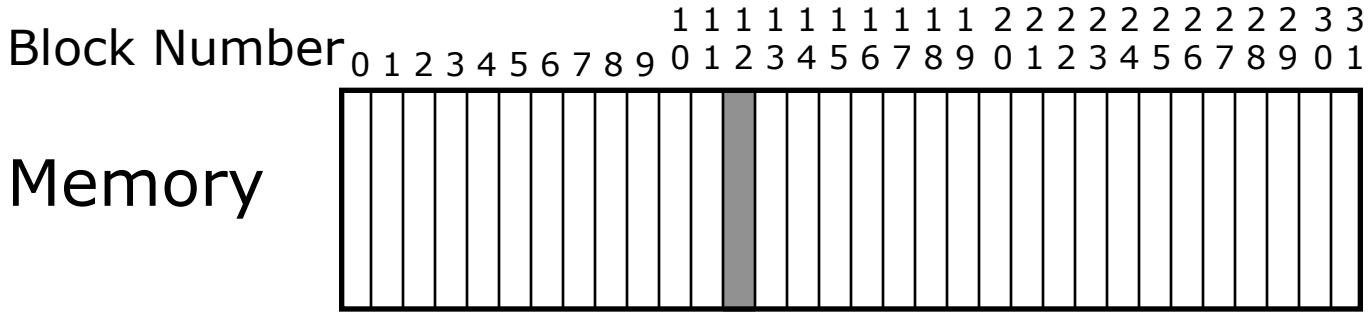
Recall:

ways = # blocks/cache set = associativity

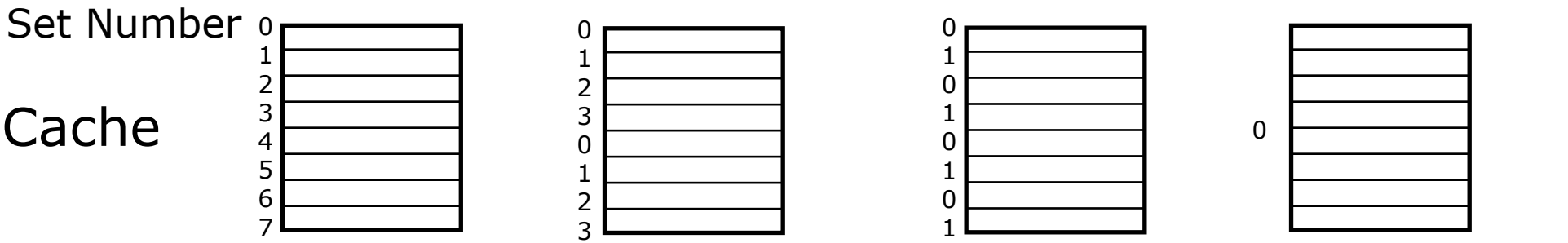
cache blocks = # ways * # sets

cache capacity = # cache blocks * Bytes/block

Q: Alternative Cache Organizations （8-block cache）



Where are possible locations in cache that block #12 in memory can be placed?



Recall:

ways = # blocks/cache set = associativity

cache blocks = # ways * # sets

cache capacity = # cache blocks * Bytes/block

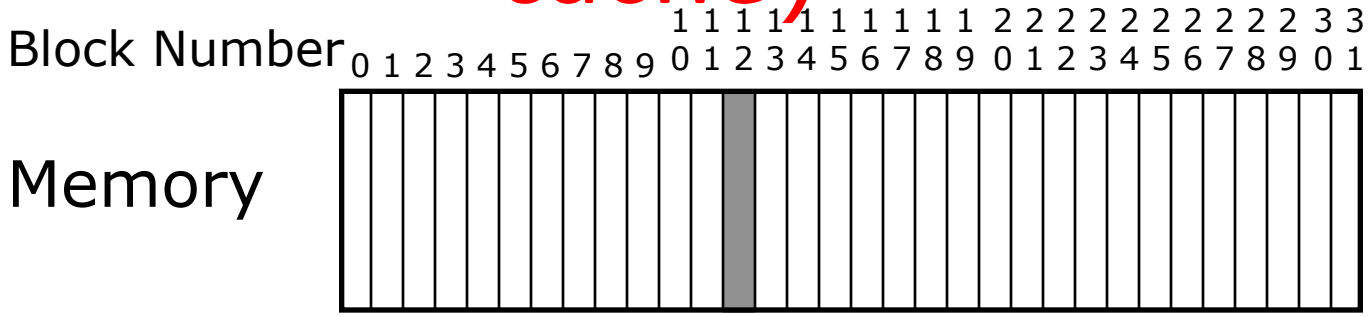
A: Alternative Cache Organizations (8-block cache)

- Memory block #12 (decimal) corresponds to memory address 01100XXXX in binary (we don't care about block size)
- For DM cache:
  - # cache blocks = 8 = # ways (1) * # sets
  - => # sets = 8 = 2^3 => Set Index has 3b => Set Index is 100 (4) (last 3b in 01100)
  - Tag (2b); Set Index (3b)
- For 2-way SA cache:
  - # cache blocks = 8 = # ways (2) * # sets
  - => # sets = 4 = 2^2 => Set Index has 2b => Set Index is 00 (last 2b in 01100)
  - Tag (3b); Set Index (2b)
- For 4-way SA cache:
  - # cache blocks = 8 = # ways (4) * # sets
  - => # sets = 2 = 2^1 => Set Index has 1b => Set Index is 0 (last 1b in 01100)
  - Tag (4b); Set Index (1b)
- For FA (8-way SA) cache:
  - # cache blocks = 8 = # ways (8) * # sets
  - => # sets = 1 = 2^0 => No Set Index
  - Tag (5b)

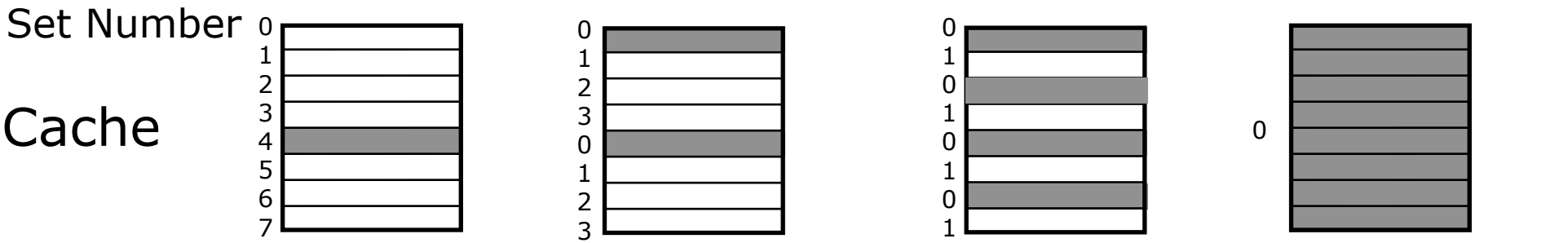
Recall:

ways = # blocks/cache set = associativity
cache blocks = # ways * # sets
cache capacity = # cache blocks * Bytes/block

A: Alternative Cache Organizations (8-block cache)



Where are possible locations in cache that block #12 in memory can be placed?



Recall:

ways = # blocks/cache set = associativity
cache blocks = # ways * # sets
cache capacity = # cache blocks * Bytes/block

A: Alternative Cache Organizations (4-block cache)

- # cache blocks = 32; Block #12 in decimal is 01100 in binary
- For DM cache: Tag (2b); Set Index (3b)
  - Set Index=100, hence it is set 4 (1 block)
- For 2-way SA cache: Tag (3b); Set Index (2b)
  - Set Index=00, hence it is set 0 (2 blocks)
- For 4-way SA cache: Tag (4b); Set Index (1b)
  - Set Index=0, hence it is set 0 (2 blocks)
- For FA (8-way SA) cache: Tag (5b)
  - Can be anywhere for SA cache

Recall:

ways = # blocks/cache set = associativity
cache blocks = # ways * # sets
cache capacity = # cache blocks * Bytes/block

Question: Cache Address Mapping

- What are the possible locations in the cache that memory address 0x1833 (0b0001 1000 0011 0011) be mapped? Assuming: 16-bit memory address, Bytes/block=16, # cache blocks=8
- For DM cache:
- For 2-way SA cache:
- For 4-way SA cache:
- For FA cache (8-way SA):

DM

Set	Tag	Data
0		
1		
2		
3		
4		
5		
6		
7		

2-way SA

Set	Tag	Data
0		
1		
2		
3		
0		
1		
2		
3		

4-way SA

Set	Tag	Data
0		
1		
0		
1		
0		
1		
0		
1		

FA (8-way SA)

Set	Tag	Data
0		
1		
0		
1		
0		
1		
0		
1		

Answer: Cache Address Mapping

- What are the possible locations in the cache that memory address 0x1833 (0b0001 1000 0011 XXXX) be mapped? Assuming: 16-bit memory address, Bytes/block=16, # cache blocks=8
- For DM cache: Tag (9b); Set Index (3b); Offset (4b)
  - Set Index=011, hence it is set 3 (1 block)
- For 2-way SA cache: Tag (10b); Set Index (2b); Offset (4b)
  - Set Index=11, hence it is set 3 (2 blocks)
- For 4-way SA cache: Tag (11b); Set Index (1b); Offset (4b)
  - Set Index=1, hence it is set 1 (4 blocks)
- For FA cache (8-way SA): Tag (12b); Offset (4b)

DM

Set	Tag	Data
0		
1		
2		
3		
4		
5		
6		
7		

2-way SA

Set	Tag	Data
0		
1		
2		
3		
0		
1		
2		
3		

4-way SA

Set	Tag	Data
0		
1		
0		
1		
0		
1		
0		
1		

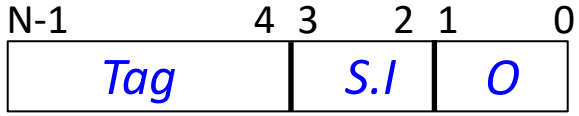
FA (8-way SA)

Set	Tag	Data
0		
0		
0		
0		
0		
0		
0		
0		

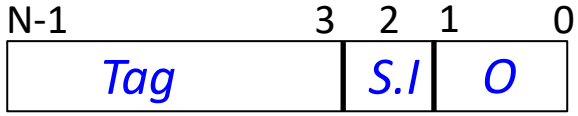
Question: Cache Capacity 1

- Work out the cache capacity :

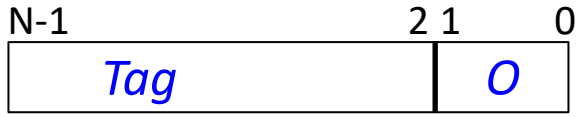
DM



2 Way SA



4-way SA (FA)



Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

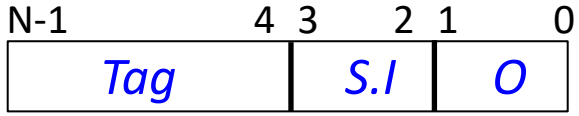
blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Answer: Cache Capacity 1

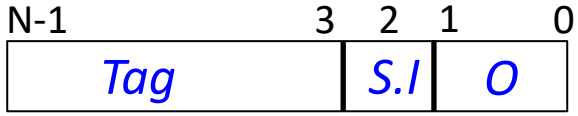
- Work out the cache capacity :

DM



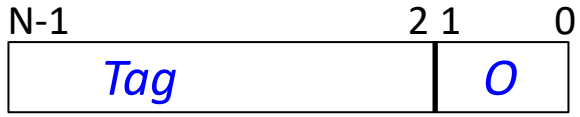
# cache blocks = 1 way * $2^{\wedge 2}$
sets = 4 blocks
cache capacity = 4 blocks *
4B/block = 16B

2 Way SA



# cache blocks = 2 ways * $2^{\wedge 1}$
sets = 4 blocks
cache capacity = 4 blocks *
4B/block = 16B

4-way SA



# cache blocks = 4 ways * $2^{\wedge 0}$
sets = 4 blocks
cache capacity = 4 blocks *
4B/block = 16B

(Just saying FA is not enough to determine cache capacity!)

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Question: Cache Capacity 2

- Q: What is the cache capacity of a DM cache with 15 Tag bits, 15 Set Index bits, and 2 Offset bits?

- Q: What is the cache capacity of a 2-way SA cache with 15 Tag bits, 15 Set Index bits, and 2 Offset bits?

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Answer: Cache Capacity 2

- Q: What is the cache capacity of a DM cache with 15 Tag bits, 15 Set Index bits, and 2 Offset bits?
- A: Bytes/block = 2^2; # sets = 2^{15}; # cache blocks = 1 way * 2^{15} = 2^{15}; cache capacity = 2^{15} blocks * 2^2 Bytes/block=2^{17} Bytes
- Q: What is the cache capacity of a 2-way SA cache with 15 Tag bits, 15 Set Index bits, and 2 Offset bits?
- A: Bytes/block = 2^2; # sets = 2^{15}; # cache blocks = 2 ways * 2^{15} = 2^{16}; cache capacity = 2^{16} blocks * 2^2 Bytes/block=2^{18} Bytes

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Answer: Cache Capacity 3

- For a cache of 64 blocks, each block 4 Bytes in size:

1. The capacity of the cache is: 256 bytes.
2. Given a 2-way SA organization, there are 32 sets, each of 2 blocks, and 2 places a block from memory could be placed.
3. Given a 4-way SA organization, there are 16 sets each of 4 blocks and 4 places a block from memory could be placed.
4. Given an 8-way SA organization, there are 8 sets each of 8 blocks and 8 places a block from memory could be placed.

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Question: Cache Capacity 4

- For an N-way SA cache, # cache blocks = B, # sets = S, which statements hold?
  (i) The cache has B number of tags
  (ii) The cache needs N comparators
  (iii) B = N x S
  (iv) Size of Set Index (in # bits) = $Log_2(S)$

Recall:

sets = $2^{SI\ size}$; # Bytes/block=$2^{Offset\ size}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Answer: Cache Capacity 4

- For an N-way SA cache, # cache blocks = B, # sets = S, which statements hold true?

  (i) The cache has B number of tags

  (ii) Size of Set Index (in # bits) = $Log_2(S)$
- A: All statements are true

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Q. Bits in Memory Address 1

- 32 bit address space, 32KB 4-way SA cache with 8-word blocks. What are the lengths of Tag - Set Index - Offset in the memory address?

ANS: T – 19, SI – 8, O – 5
Bytes/block=8 words=32B => Offset is 5b
cache capacity (32KB) = # cache blocks*32B/block
=> # cache blocks = 1K = 2^{10}
cache blocks (2^{10}) = # ways (4) * # sets
=> # sets = 2^8 => Set Index has 8b
Memory address length (32)
=> T = 32b – (8b+5b) = 19b

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Q. Bits in Memory Address 2

- 32 bit address space, 16KB DM cache with 4-word blocks. What are the lengths of Tag - Set Index - Offset?

ANS: T - 18, SI – 10, O – 4
Bytes/block=4 words=16B => Offset is 4b
cache capacity (16KB) = # cache blocks*16B/block
=> # cache blocks = 1K = 2^{10}
cache blocks (2^{10}) = # ways (1) * # sets
=> # sets = 2^{10} => Set Index has 10b
Memory address length (32)
=> T = 32b – (10b+4b) = 18b

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$
blocks = # ways (associativity) * # sets
cache capacity = # blocks * # Bytes/block

Q. Bits in Memory Address 3

- We have a cache of size 2 KB with block size of 128 Bytes. If our cache has 2 sets, what is its associativity? If memory address is 16 bits, how wide is the Tag field?

  Bytes/block=128B = 2^7B => Offset has 7b

  cache capacity (2KB=2^{11}B) = # cache blocks*2^7B/block

  => # cache blocks = 16 = 2^4

  # cache blocks (16) = # ways * # sets (2)

  $\Rightarrow$ # ways = 8 = 2^3

  Set Index has 1b

  Memory address length (16) = T + SI + O

  => T = 16b – (1b+7b) = 8b

Recall:

sets = $2^{\text{SI size}}$; # Bytes/block=$2^{\text{Offset size}}$

blocks = # ways (associativity) * # sets

cache capacity = # blocks * # Bytes/block

Question: Bits in Memory Address 4

- 32 bit address space, 32KB DM cache with 8-word blocks
- 32 bit address space, 16KB 2-way SA cache with 4-word blocks
- 32 bit address space, 32KB FA cache with 8-word blocks

Answer: Bits in Memory Address 4

- 32 bit address space, 32KB DM cache with 8-word blocks
- T - 17, SI - 10, O – 5 (# blocks = # sets = 2^{10} => SI has 10b, T = 32b–(10b+5b)=17b)
- 32 bit address space, 16KB 2-way SA cache with 4-word blocks
- T - 19, SI - 9, O – 4 (# blocks = 2^{10}; # sets = $2^{10}/2=2^9$ => SI has 9b, T = 32b–(9b+4b)=19b)
- 32 bit address space, 32KB FA cache with 8-word blocks
- T - 27, SI - 0, O – 5 (# blocks = 2^{10}; # sets = 1 => SI has 0b, T = 32b–(0b+5b) = 27b)

Question: Associativity 1

- For a cache with fixed total size, if we increase the number of ways by a factor of two, which statement is false:
  - **A** : The number of sets is halved
  - **B** : The tag width decreases
  - **C** : The block size stays the same
  - **D** : The set index decreases

Answer: Associativity 1

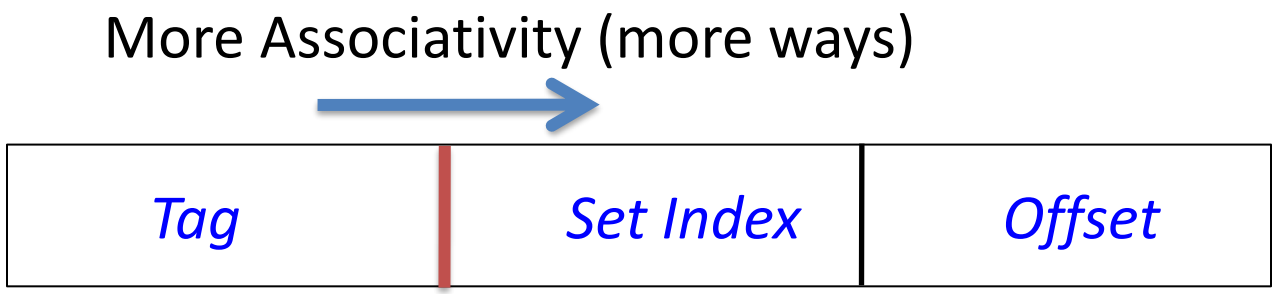
- For a cache with fixed total size, if we increase the number of ways by a factor of two, which statement is false:

  A : The number of sets is halved

  B : The tag width decreases

  C : The block size stays the same

  D : The set index width decreases

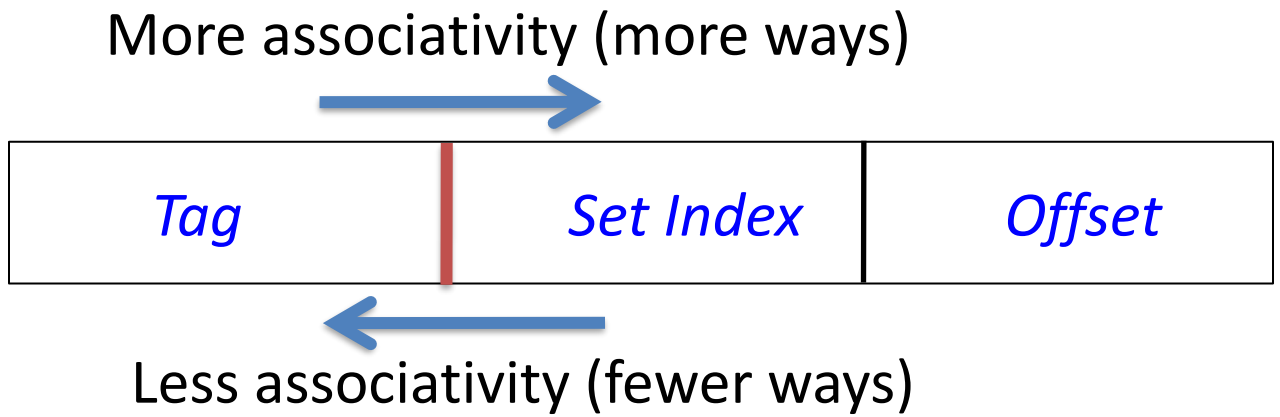


Question: Associativity 2

Push red bar right 1 bit
tag_size ?; index_size?; # sets*?*; # ways/associativity *?;*
 # HW comparators ?
Push red bar left 1 bit
tag_size ?; index_size ?; # sets ?; # ways/associativity *?;*
 # HW comparators ?

Answer: Associativity 2

Push red bar right 1 bit
tag_size +1; index_size -1; # sets halved; # ways/associativity doubled;
 # HW comparators doubled
Push red bar left 1 bit
tag_size -1; index_size +1; # sets doubled; # ways/associativity halved;
 # HW comparators halved

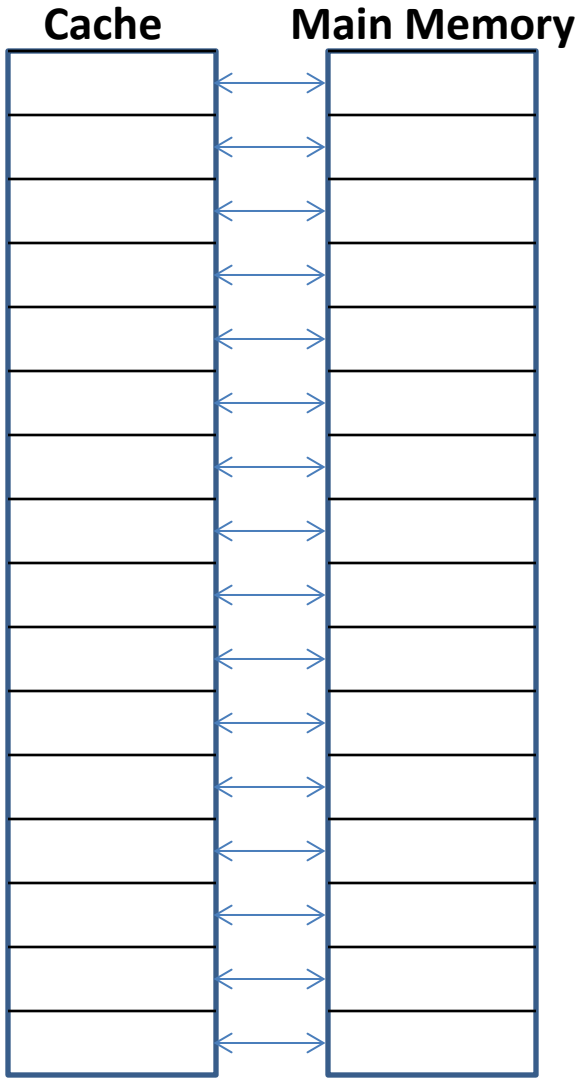


Question: Associativity vs. Performance

For a cache of fixed capacity and block size, increasing associativity causes _____ in hit time, and _____ in miss rate

Answer: Associativity vs. Performance

For a cache of fixed capacity and block size, increasing associativity causes _increase_ in hit time, and _decrease_ in miss rate

Question: Tag bits & Offset bits

- Q: Under what condition will we have # Offset bits = 0? Under what condition will we have # Tag bits = 0?
- A: # Offset bits = 0 when size of a cache block = 1 Byte
  - (not realistic, since it cannot even fit a 16b short or 32b int)
- # Tag bits = 0 when we have a DM cache with the same size as memory
  - Tag bits are needed to disambiguate among multiple possible memory blocks that may be mapped to one cache block; if there is a 1-to-1 correspondence between cache blocks and memory blocks, then Tag bits are not needed
  - (not realistic, since cache must be small in order to be fast)