EXERCISES on CACHES - V2024

Exercise 1

Let us consider a memory hierarchy (main memory + cache) given by:
- Memory size 1 Giga words of 16 bit (word addressed);
- Cache size 1 Mega words of 16 bit (word addressed);
- Cache block size 256 words of 16 bit.

1. *Calculate the number of cache blocks;*

2. *Calculate the structure of the addresses for the following cache structures:*
   - direct mapped cache;
   - fully associative cache;
   - 2-way set-associative cache;
   - 4-way set-associative cache;
   - 8-way set–associative cache.

3. *Calculate the number of sets for the previous set-associative caches.*

Solution (1):

Number of cache blocks = Cache size / Block size = 1M Word / 256 Word = $2^{20} / 2^{8} = 2^{12} = 4096$

Solution (2):

Main memory address: 30 bit
Cache memory address: 20 bit

Direct Mapped Cache
Tag: 10 bit | Index: 12 bit | Word Offset (in the block): 8 bit

Fully Associative Cache
Tag: 22 bit | Word Offset (in the block): 8 bit

2-way Set-Associative Cache
Tag: 11 bit | Index: 11 bit | Word Offset (in the block): 8 bit

4-way Set-Associative Cache
Tag: 12 bit | Index: 10 bit | Word Offset (in the block): 8 bit

8-way Set-Associative Cache
Tag: 13 bit | Index: 9 bit | Word Offset (in the block): 8 bit

Solution (3):

Number of sets in n-way SA cache = Number of blocks / Number of ways = 4096 / n

Number of sets in 2-way SA cache = 4096 /2 = 2048
Number of sets in 4-way SA cache = 4096 /4 = 1024
Number of sets in 8-way SA cache = 4096 /8 = 512

EXERCISES on CACHES - V2024

Exercise 2

Let us consider a memory hierarchy (main memory + cache) given by:
- Memory size 2 Giga Byte (Byte addressed);
- Cache size 512 K Byte (Byte addressed);
- Cache block size 512 Byte.

1. *Calculate the number of cache blocks*
2. *Calculate the structure of the addresses for the following cache structures:*

   - direct mapped cache;
   - fully associative cache;
   - 4-way set-associative cache;

Solution (1)

Number of cache blocks = Cache size / Block size = 512 KB / 512 B = 1 K = 1024

Solution (2)

Main memory: 31 bit address
Cache memory: 19 bit address

Direct Mapped Cache

Tag: 12 bit | Index: 10 bit | Byte Offset (in the block): 9 bit

Fully Associative Cache

Tag: 22 bit | Byte Offset (in the block): 9 bit

4-way Set-Associative Cache

Tag: 14 bit | Index: 8 bit | Byte Offset (in the block): 9 bit

EXERCISES on CACHES - V2024

Exercise 3 – Cache simulation

Let us consider a memory hierarchy composed of the main memory and a **direct mapped cache** with:
- Memory size 4 KByte (Byte addressed);
- Cache size 1 Kbyte (Byte addressed);
- Cache block size 256 Byte.

1. *Calculate the following parameters:*

Number of blocks in memory: **16 blocks**
Number of blocks in cache: **4 blocks**
Structure of the addresses:
Memory address: **12 bit**
Cache address: **10 bit**
Byte Offset (in the block): **8 bit**

Structure of 12-bit memory address: **| 4bit MEM BLOCK INDEX | 8bit BYTE OFFSET |**
| 2bit TAG | 2bit CACHE INDEX | 8bit BYTE OFFSET |

Example: **| 1 1 | 1 0 | 1 1 1 0 0 0 1 0 |**
| 2bit TAG | 2bit CACHE INDEX | 8bit BYTE OFFSET |

2. *Given the following sequence of memory accesses, please complete the following table by **simulating** the direct mapped cache behavior in terms of:*
   - **1-bit HIT/MISS**
   - **1-bit VALID**
   - **2-bit TAG**
   - **DATA** to report the number **in base$_{10}$** of the memory block loaded/accessed in cache.
   - **ACTION** to report the memory block accessed in the cache block (in case of HIT) or loaded in cache (in case of MISS)

	Requested Memory Address	H / M	Block_0_cache index 00			Block_1_cache index 01			Block_2_cache index 10			Block_3_cache index 11			ACTION
			V	TAG	DATA bl.$_{10}$	V	TAG	DATA bl.$_{10}$	V	TAG	DATA bl.$_{10}$	V	TAG	DATA bl.$_{10}$	
			1	**00**	**0**	**0**	**11**	**2**	**1**	**01**	**6**	**0**	**10**	**7**	**Initial situation**
1	11 10 1110 0010	M	1	00	0	0	11	2	**1**	**11**	**14**	0	10	7	Load bl.14 in block_2_cache
2	00 01 1110 0101	M	1	00	0	**1**	**00**	**1**	1	11	14	0	10	7	Load bl.1 in block_1_cache
3	00 01 0011 1001	H	1	00	0	**1**	**00**	**1**	1	11	14	0	10	7	Access bl.1 in block_1_cache
4	11 00 0000 0010	M	**1**	**11**	**12**	1	00	1	1	11	14	0	10	7	Load bl.12 in block_0_cache
5	00 11 1111 1110	M	1	11	12	1	00	1	1	11	14	**1**	**00**	**3**	Load bl.3 in block_3_cache
6	10 01 0100 0111	M	1	11	12	**1**	**10**	**9**	1	11	14	1	00	3	Load bl.9 in block_1_cache

EXERCISES on CACHES - V2024

Exercise 4 – Cache simulation

Let us consider a memory hierarchy composed of the main memory and a **2-way set associative cache** with:
- Memory size 4 KByte (Byte addressed)
- Cache size 512 byte (Byte addressed)
- Cache block size 128 Byte

1. *Calculate the following parameters:*

Number of blocks in memory: **32 blocks**
Number of blocks in cache: **4 blocks** (marked as block_a, block_b, block_c and block_d)
Number of sets= 2 sets (marked as Set_0 and Set_1)

Structure of the addresses:
Memory address: **12 bit**
Cache address: **9 bit**
Byte Offset (in the block): **7 bit**

Structure of 12-bit memory address: **| 5bit MEM BLOCK INDEX | 7-bit BYTE OFFSET |**
| 4-bit TAG | 1-bit cache INDEX | 7-bit BYTE OFFSET |

Example: **| 1 1 1 0 | 1 | 1 1 0 0 0 1 0 |**
| 4-bit TAG | 1-bit cache INDEX | 7bit BYTE OFFSET |

Given the following sequence of memory accesses, please complete the following table describing the behavior of the:

2. *Given the following sequence of memory accesses, please complete the following table by **simulating** the* 2-way set associative cache assuming the **LRU (Least Recently Used)** substitution policy*:*
   - **1-bit HIT/MISS**
   - **1-bit VALID**
   - **4-bit TAG**
   - **DATA** to report the number **in base$_{10}$** of the memory block loaded/accessed in cache.
   - **ACTION** to report the memory block accessed in the cache block (in case of HIT) or loaded in cache (in case of MISS)

	Requested Memory Address	H / M	Set_0 block_a			Set_0 block_b			Set_1 block_c			Set_1 block_d			ACTION
			V	TAG	DATA bl.$_{10}$	V	TAG	DATA bl.$_{10}$	V	TAG	DATA bl.$_{10}$	V	TAG	DATA bl.$_{10}$	
			1	**0001**	**2**	**0**	**0001**	**2**	**1**	**0001**	**3**	**0**	**0010**	**7**	**Initial situation**
1	0010 1 1010010	M	1	0001	2	0	0001	2	1	0001	3	**1**	**0010**	**5**	Load bl. 5 in block_d_cache
2	0001 1 1010100	H	1	0001	2	0	0001	2	**1**	**0001**	**3**	1	0010	5	Access bl.3 in block_c_cache
3	0001 0 0111111	H	**1**	**0001**	**2**	0	0001	2	1	0001	3	1	0010	5	Access bl.2 in block_a_cache
4	1100 0 0001010	M	1	0001	2	**1**	**1100**	**24**	1	0001	3	1	0010	5	Load bl.24 in block_b_cache
5	0111 1 1010010	M	1	0001	2	1	1100	24	1	0001	3	**1**	**0111**	**15**	Load bl.15 in block_d_cache
6	1011 0 1000110	M	**1**	**1011**	**22**	1	1100	24	1	0001	3	1	0111	15	Load bl.22 in block_a_cache

EXERCISES on CACHES - V2024

Question 1 (format Multiple Choice – Single answer) - 10 Jan. 2022 – PART1

Please indicate the structure of the 32-bit memory address of a 4-way set-associative cache with 1 MB cache size and a block size of 256 B.

(SINGLE ANSWER)
1 point

Answer 1: | 24-bit tag | 8-bit offset |

Answer 2: |14-bit tag | 10-bit index | 8-bit offset | **(TRUE)**

Answer 3: |13-bit tag | 11-bit index | 8-bit offset |

Answer 4: |12-bit tag | 12-bit index | 8-bit offset |

Solution:
Answer 2: |14-bit tag | 10-bit index | 8-bit offset | **(TRUE)**

Question 5 (format true/false) - 10 Jan. 2022 – PART1

*In a **write-back cache,** the most updated value of a dirty cache block is always in the cache.*

Answer: **T / F**
1 point

SOLUTION: ***True***

Feedback:
In a write-back cache, a modified (dirty) cache block is not coherent with the main memory but the most updated value is always in the cache. The modified block will be written in the main memory only when it will be replaced due to a miss. See slides on L08: Memory Hierarchy.

EXERCISES on CACHES - V2024

Question 12 (format SINGLE ANSWER) - 10 Jan. 2022 – PART1

*For a **direct-mapped cache,** which strategy can be used for the block replacement?*

(SINGLE ANSWER)
1 point
Answer 1: *LRU (Least Recently Used)*.
Answer 2: FIFO (First In First Out).
Answer 3: None of them **(TRUE)**

Solution
Answer 3: None of them **(TRUE)**
Feedback:
For a direct mapped cache, there is no need of any block replacement strategy, because there is only one block to be replaced by direct mapping.

EXERCISES on CACHES - V2024

Question 3 (format Multiple Choice – Single answer) – 25 Aug. 2021 - PART2

Let's consider a fully associative write-back cache with many cache entries that at cold start is empty and receives the following sequence of memory accesses:

Read Mem[AAAA]
Write Mem[AAAA]
Read Mem[BBBB]
Write Mem[BBBB]
Read Mem[AAAA]
Read Mem[BBBB]

What are the number of cache hits and misses when using a ***"write allocate"*** versus a ***"no-write allocate"*** policy?
(SINGLE ANSWER)
2 points

Answer 1: Write allocate has 2 hits & 4 misses | No-write allocate has 4 hits & 2 misses
Answer 2: Write allocate has 4 hits & 2 misses | No-write allocate has 4 hits & 2 misses
Answer 3: Write allocate has 4 hits & 2 misses | No-write allocate has 2 hits & 4 misses
Answer 4: Write allocate has 5 hits & 1 miss | No-write allocate has 5 hits & 1 miss
Answer 5: Write allocate has 3 hits & 3 misses | No-write allocate has 2 hits & 4 misses

Solution:
Answer 2: Write allocate has 4 hits & 2 misses | No-write allocate has 4 hits & 2 misses **(TRUE)**
Feedback
For the write allocate policy, the first read accesses to Mem[AAAA] and Mem[BBBB] are misses, and the 2 blocks corresponding to [AAAA] and [BBBB] are allocated in cache. The next four accesses are all hits, since the blocks corresponding to [AAAA] and [BBBB] are found in cache. Globally, the write allocate has 2 misses & 4 hits.

For the no-write allocate policy there is the same behavior as in the write allocate policy because the first accesses to Mem[AAAA] and Mem[BBBB] are done on read and the corresponding blocks are allocated in cache. Therefore, also the no-write allocate policy has 2 misses & 4 hits.

EXERCISES on CACHES - V2024

Question 4 (complete table format) from PART2 – Exam 21 July, 2022
2 points

Let's consider 32-block main memory with a **direct-mapped** 8-block cache based on a ***write allocate*** with ***write-through*** protocol.

The addresses are expressed as decimal numbers:
Memory Address: $[0, 1, 2, \ldots\ldots\ldots 31]_{10}$
Cache Address: $[0, 1, 2, \ldots 7]_{10}$
and Cache Tags are expressed as binary numbers.

At cold start the cache is empty, then there is the following sequence of memory accesses.
Please complete the following table:

	Type of memory access	Memory Address	HIT/MISS Type	Cache Tag	Cache Address	Write in memory
1	Read	$[24]_{10}$	Cold-start Miss	$[11]_2$	$[0]_{10}$	No
2	Write	$[24]_{10}$	Hit	$[11]_2$	$[0]_{10}$	Yes, Wr. in M$[24]_{10}$
3	Read	$[24]_{10}$	Hit	$[11]_2$	$[0]_{10}$	No
4	Read	$[10]_{10}$				
5	Read	$[16]_{10}$				
6	Write	$[10]_{10}$				
7	Write	$[24]_{10}$				
8	Read	$[12]_{10}$				
9	Read	$[21]_{10}$				
10	Write	$[18]_{10}$				

Solution:

	Type of memory access	Memory Address	HIT/MISS Type	Cache Tag	Cache Address	Write in memory
1	Read	$[24]_{10}$	Cold-start Miss	$[11]_2$	$[0]_{10}$	No
2	Write	$[24]_{10}$	Hit	$[11]_2$	$[0]_{10}$	Yes, Wr. in M$[24]_{10}$
3	Read	$[24]_{10}$	Hit	$[11]_2$	$[0]_{10}$	No
4	Read	$[10]_{10}$	Cold-start Miss	$[01]_2$	$[2]_{10}$	No
5	Read	$[16]_{10}$	Conflict Miss	$[10]_2$	$[0]_{10}$	No
6	Write	$[10]_{10}$	Hit	$[01]_2$	$[2]_{10}$	Yes, Wr. in M$[10]_{10}$
7	Write	$[24]_{10}$	Conflict Miss	$[11]_2$	$[0]_{10}$	Yes, Wr. in M$[24]_{10}$
8	Read	$[12]_{10}$	Cold-start Miss	$[01]_2$	$[4]_{10}$	No
9	Read	$[21]_{10}$	Cold-start Miss	$[10]_2$	$[5]_{10}$	No
10	Write	$[18]_{10}$	Conflict Miss	$[10]_2$	$[2]_{10}$	Yes, W in M$[18]_{10}$

EXERCISES on CACHES - V2024

EXERCISE 3 – CACHE MEMORIES (3 points) – exam 22/1/2024

Let's consider a **fully associative** cache with 4 blocks [a, b, c, d] that at cold start is empty and receives the following sequence of memory accesses:

```
Read  Mem[AAAA]
Write Mem[AAAA]
Read  Mem[BBBB]
Write Mem[BBBB]
Read  Mem[AAAA]
Write Mem[CCCC]
Read  Mem[CCCC]
Read  Mem[BBBB]
```

Part A) *Assuming* ***Write Allocate*** *and* ***Write Back*** *cache policies, complete the following table:*

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Block	Dirty bit	Write-back in memory
1	read	[AAAA] $_{ex}$	Cold start MISS	[AAAA] $_{ex}$	a	0	no
2	write	[AAAA] $_{ex}$	HIT	[AAAA] $_{ex}$	a	1	no
3	read	[BBBB] $_{ex}$					
4	write	[BBBB] $_{ex}$					
5	read	[AAAA] $_{ex}$					
6	write	[CCCC] $_{ex}$					
7	read	[CCCC] $_{ex}$					
8	write	[BBBB] $_{ex}$					

How many cache misses? _____

How many cache hits? _____

Calculate the Miss Rate: _____

EXERCISES on CACHES - V2024

Part b) *Assuming* ***No-write Allocate*** *and* ***Write Through*** *cache policies, complete the following table:*

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Block	Write in memory
1	read	[AAAA] $_{ex}$	Cold start MISS	[AAAA] $_{ex}$	a	no
2	write	[AAAA] $_{ex}$	HIT	[AAAA] $_{ex}$	a	yes
3	read	[BBBB] $_{ex}$				
4	write	[BBBB] $_{ex}$				
5	read	[AAAA] $_{ex}$				
6	write	[CCCC] $_{ex}$				
7	read	[CCCC] $_{ex}$				
8	write	[BBBB] $_{ex}$				

How many cache misses? _____

How many cache hits? _____

Calculate the Miss Rate: _____

EXERCISES on CACHES - V2024

Solution

Part A) *Assuming the **Write Allocate** and **Write Back** cache policies, complete the following table:*

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Block	Dirty bit	Write-back in memory
1	read	[AAAA] ex	Cold start MISS	[AAAA] ex	a	0	no
2	write	[AAAA] ex	HIT	[AAAA] ex	a	1	no
3	read	[BBBB] ex	*Cold start MISS*	[BBBB] ex	*b*	*0*	*no*
4	write	[BBBB] ex	*HIT*	[BBBB] ex	*b*	*1*	*no*
5	read	[AAAA] ex	*HIT*	[AAAA] ex	*a*	*1*	*no*
6	write	[CCCC] ex	*Cold start MISS*	[CCCC] ex	*c*	*1*	*no*
7	read	[CCCC] ex	*HIT*	[CCCC] ex	*c*	*1*	*no*
8	write	[BBBB] ex	*HIT*	[BBBB] ex	*b*	*1*	*no*

How many cache misses?

3

How many cache hits?

5

Calculate the Miss Rate:

Miss Rate = Number of misses / Number of memory access = 3 / 8= 0.375

EXERCISES on CACHES - V2024

Solution

Part b) *Assuming the* ***No-write Allocate*** *and* ***Write Through*** *cache policies, complete the following table:*

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Block	Write in memory
1	read	[AAAA] ex	Cold start MISS	[AAAA] ex	a	no
2	write	[AAAA] ex	HIT	[AAAA] ex	a	yes
3	read	[BBBB] ex	*Cold start MISS*	[BBBB] ex	*b*	*no*
4	write	[BBBB] ex	*HIT*	[BBBB] ex	*b*	*yes*
5	read	[AAAA] ex	*HIT*	[AAAA] ex	*a*	*no*
6	write	[CCCC] ex	**Miss with no-write allocation in cache**	--	--	*Yes (*)*
7	read	[CCCC] ex	*Cold start MISS*	[CCCC] ex	*c*	*no*
8	write	[BBBB] ex	*HIT*	[BBBB] ex	*b*	*yes*

(*) Write done directly in memory with no-write allocation in cache

How many cache misses?

4

How many cache hits?

4

Calculate the Miss Rate:

Miss Rate = Number of misses / Number of memory access = 4 / 8 = 0.5

EXERCISES on CACHES - V2024

EXERCISE 4 – CACHE MEMORIES (3 points) – exam 22/1/2024

Let's consider 32-block main memory with a **direct-mapped** 8-block cache based on a ***write allocate*** with ***write-back*** protocol.

The addresses are expressed as decimal numbers:

Memory Block Addresses: [0, 1, 2, …….. 31] $_{10}$

Cache Block Addresses: [0, 1, 2, …7] $_{10}$

At cold start the cache is empty, then there is the following sequence of memory accesses.

1. *Please complete the following table:*

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Index	Dirty bit	Write-back in memory
1	write	[12] $_{10}$	Cold start MISS	[01] $_2$	[100] $_2$	1	no
2	read	[12] $_{10}$	HIT	[01] $_2$	[100] $_2$	1	no
3	read	[10] $_{10}$	*Cold start MISS*	*[01] $_2$*	*[010] $_2$*	*0*	*no*
4	write	[10] $_{10}$	*HIT*	*[01] $_2$*	*[010] $_2$*	*1*	*no*
5	write	[26] $_{10}$	*Conflict MISS*	*[11] $_2$*	*[010] $_2$*	*1*	*WB in M[10] $_{10}$*
6	read	[20] $_{10}$	*Conflict MISS*	*[10] $_2$*	*[100] $_2$*	*0*	*WB in M[12] $_{10}$*
7	read	[30] $_{10}$	*Cold start MISS*	*[11] $_2$*	*[110] $_2$*	*0*	*no*
8	write	[16] $_{10}$	*Cold start MISS*	*[10] $_2$*	*[000] $_2$*	*1*	*no*
9	write	[6] $_{10}$	*Conflict MISS*	*[00] $_2$*	*[110] $_2$*	*1*	*no (block 30 not modified)*
10	read	[18] $_{10}$	*Conflict MISS*	*[10] $_2$*	*[010] $_2$*	*0*	*WB in M[26] $_{10}$*

2. *Calculate the Miss Rate:*

Miss Rate = Number of misses / Number of memory access = 8/10 = 0.8

EXERCISES on CACHES - V2024

EXERCISE 3 – CACHE MEMORIES (3 points) – exam 15/2/2024

Let's consider 32-block main memory with a **fully-associative** 4-block cache and **LRU** cache replacement policy. The addresses are expressed as:

Memory Block Addresses: $[0, 1, 2, \ldots\ldots\ldots\ldots 31]_{10}$

Cache Block Addresses: [a, b, c, d]

At cold start the cache is empty, then there is the following sequence of memory accesses.

- *Assuming the **Write Allocate** and **Write Back** cache policies, complete the following table:*

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Block Address	Dirty bit	Write-back in memory
1	read	$[12]_{10}$	Cold start MISS	$[01100]_2$	a	0	no
2	write	$[12]_{10}$	HIT	$[01100]_2$	a	1	no
3	write	$[10]_{10}$					
4	read	$[10]_{10}$					
5	read	$[26]_{10}$					
6	write	$[7]_{10}$					
7	read	$[12]_{10}$					
8	read	$[4]_{10}$					
9	write	$[26]_{10}$					
10	write	$[18]_{10}$					
11	read	$[5]_{10}$					
12	write	$[24]_{10}$					

Calculate the Miss Rate:
Miss Rate = _____

EXERCISES on CACHES - V2024

Solution:

	Type of memory access	Memory Address	HIT / MISS Type	Cache Tag	Cache Block Address	Dirty bit	Write-back in memory
1	read	$[12]_{10}$	Cold start MISS	$[01100]_2$	a	0	no
2	write	$[12]_{10}$	HIT	$[01100]_2$	a	1	no
3	write	$[10]_{10}$	*Cold start MISS*	$[01010]_2$	*b*	*1*	*no*
4	read	$[10]_{10}$	*HIT*	$[01010]_2$	*b*	*1*	*no*
5	read	$[26]_{10}$	*Cold start MISS*	$[11010]_2$	*c*	*0*	*no*
6	write	$[7]_{10}$	*Cold start MISS*	$[00111]_2$	*d*	*1*	*no*
7	read	$[12]_{10}$	*HIT*	$[01100]_2$	*a*	*1*	*no*
8	read	$[4]_{10}$	*Conflict MISS*	$[00100]_2$	*b*	*0*	*Yes: WB in* $[10]_{10}$
9	write	$[26]_{10}$	*HIT*	$[11010]_2$	*c*	*1*	*no*
10	write	$[18]_{10}$	*Conflict MISS*	$[10010]_2$	*d*	*1*	*Yes: WB in* $[7]_{10}$
11	read	$[5]_{10}$	*Conflict MISS*	$[00101]_2$	*a*	*0*	*Yes: WB in* $[12]_{10}$
12	write	$[24]_{10}$	*Conflict MISS*	$[11000]_2$	*b*	*1*	*No*

Calculate the Miss Rate:

Miss Rate = *Number of misses / Number of memory access = 8 / 12= 0,67*