TDTS08:
Advanced Computer Architecture

Lesson

LINKÖPING UNIVERSITY

Outline

- Lab organization and goals
- SimpleScalar architecture and tools
- Lab 5: article review
- Exercises

Organization

- Assistant: Rouhollah Mahfouzi
- Web page
  - http://www.ida.liu.se/~TDTS08
  - Check the lab page!

Organization

- Sign up in Webreg latest Sep. 8 (**Today!**).
- Deadline for the assignments:

Lab 1, Lab 2	Sep. 24
Lab 3, Lab 4	Oct. 15
Lab 5	Oct. 27

- Rules: Read them!

Examination

Written report for each lab:

- Hand in the report, in PDF or DOC format
- Submit the report via Teams

Labs

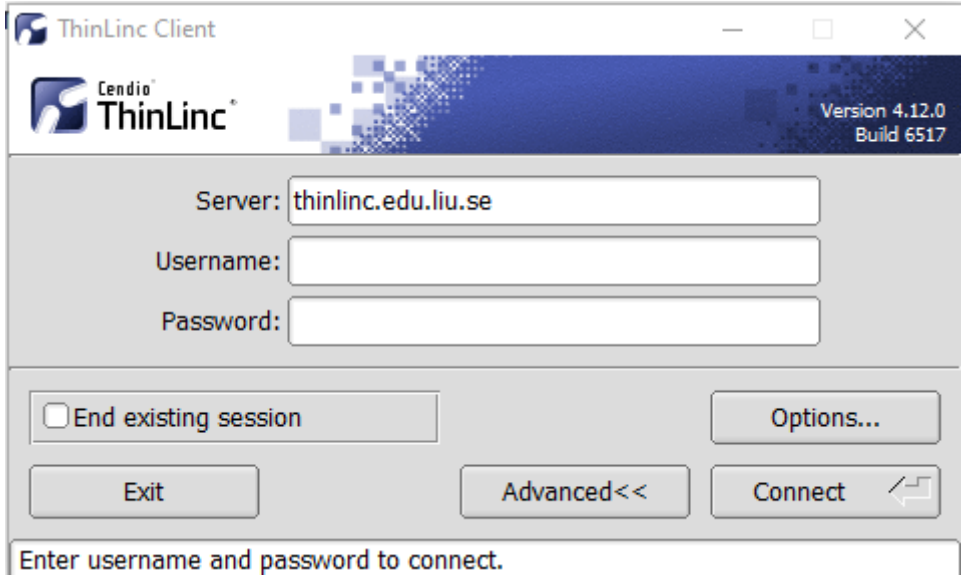
- Five labs:
  1. Cache Memories (2 lab sessions)
  2. Instruction Pipelining (2 lab sessions)
  3. Superscalar Processors (2 lab sessions)
  4. VLIW processors (2 lab sessions)
  5. Article review on multiprocessor systems (no lab session)

Remote

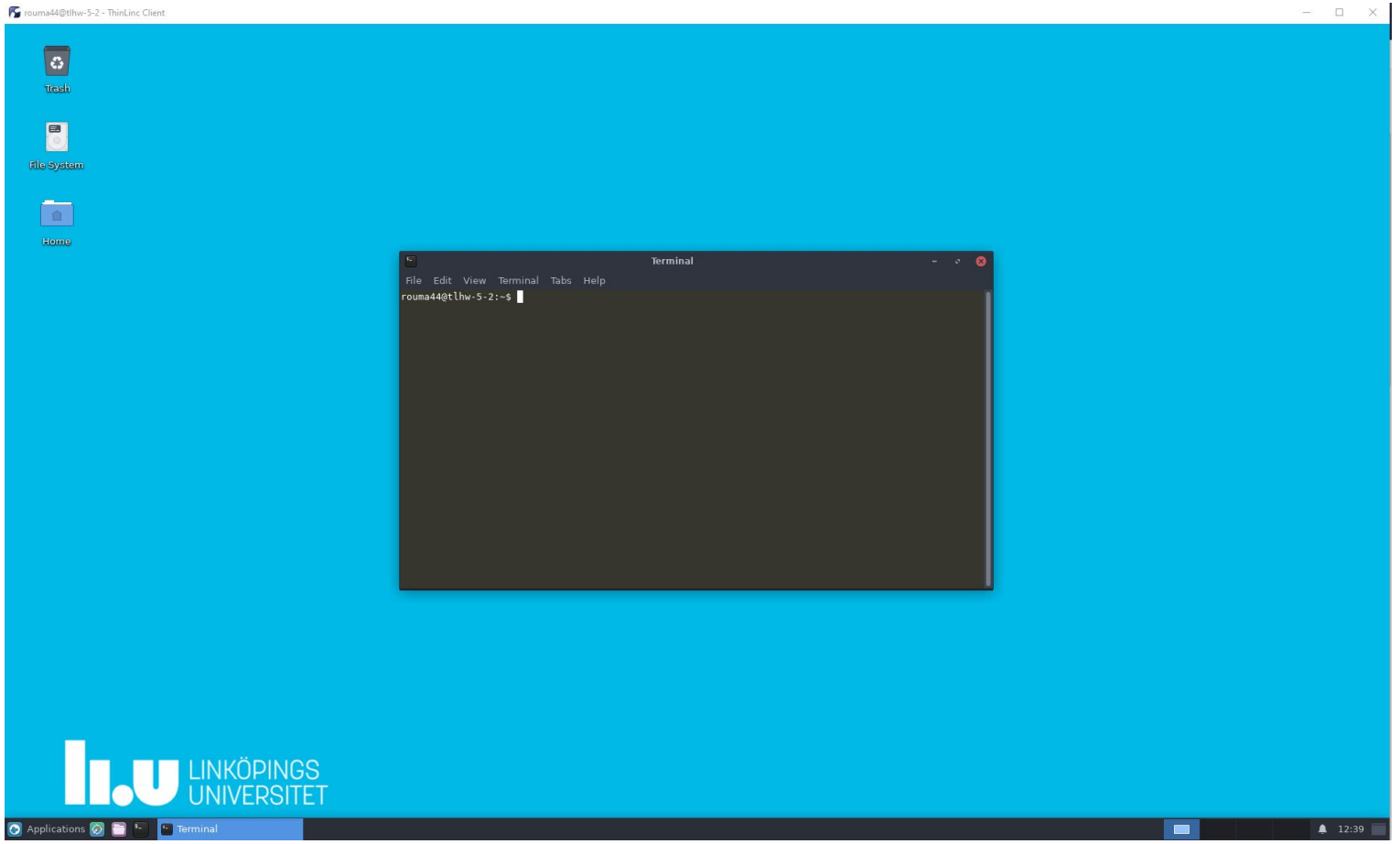
- Thinlinc client: thinlinc.edu.liu.se

- SSH client: ssh.edu.liu.se

ThinLinc


ThinLinc Client
Cendio ThinLinc
Version 4.12.0
Build 6517
Server: thinlinc.edu.liu.se
Username:
Password:
End existing session
Options...
Exit
Advanced<<
Connect
Enter username and password to connect.

ThinLinc



Press F8 for options

How to ask questions during lab sessions?

- General channel [put yourself in queue by posting a message]
- Private Team for each subgroup
  - Ask questions (video chat)
  - Upload your lab reports

Environment

- Linux
- Simulations are started from a command line (i.e., terminal)
  - To open a new terminal you can press ctrl+alt+t
- Get yourself familiarized with the terminal
  - Ask Google first
  - Ask your assistant
- Make sure you learn the basic commands (i.e., *cd, ls, cp*, …)

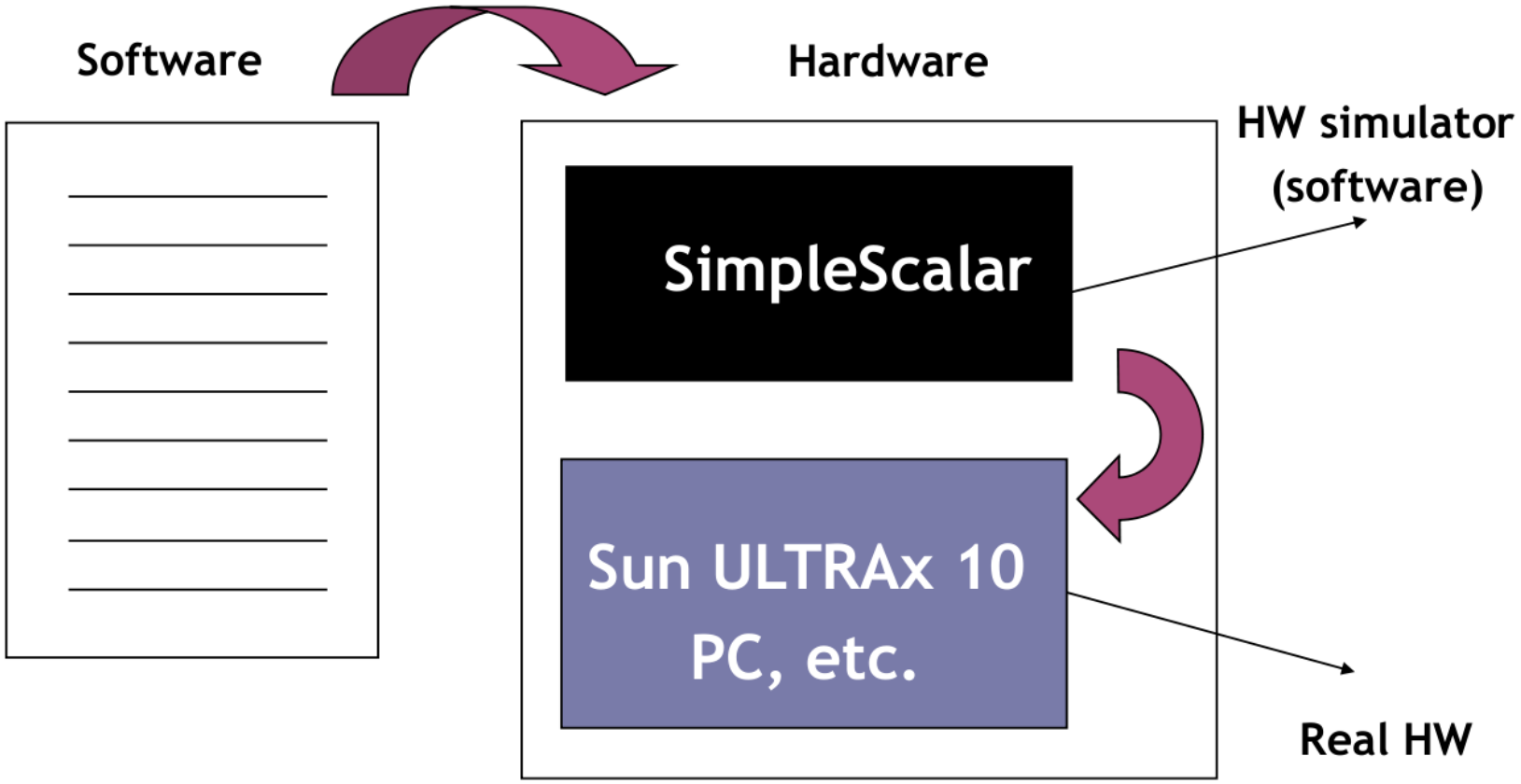
Tool Setup

- Don’t forget the instructions in lab0
- Instructions should be clear and easy to follow, but if you face difficulties
  - Don’t get frustrated :)
  - Read again carefully (without skipping over the lines)
  - Consult your assistant

Outline

- Lab organization and goals
- SimpleScalar architecture and tools
- Lab 5: article review
- Exercises

Architecture Simulation



SimpleScalar: Literature

- "*The SimpleScalar Tool Set, Version 2.0*", by Doug Burger and Todd M. Austin
  - Very important preparation for the labs
  - This is your main reference for the tool!
- "User's and Hacker's guide", slides by Austin

SimpleScalar Architecture

- Virtual architecture derived from MIPS
  - Control (j, jr,..., beq, bne,...)
  - Load/Store (lb, lbu, ...)
  - Integer Arithmetic (add, addu, ...)
  - Floating Point Arithmetic (add.s, add.d, ...)
  - Miscellaneous (nop, syscall, break)

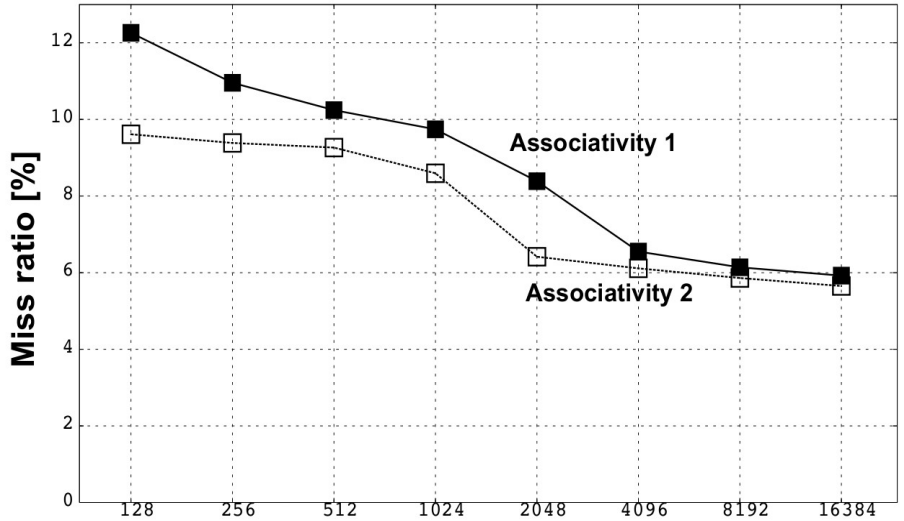
SimpleScalar Architecture (cont'd)

- Several simulators
  - Sim-fast: Fast, only functional simulation (no timing)
  - Sim-safe: Sim-fast + memory checks
  - Sim-cache: Sim-safe + cache simulation and various timing properties (simulation time, measured time, ...)
  - Sim-cheetah: Simulation of multiple cache configurations
  - Sim-outorder: Superscalar simulator

Won't use these two!

An Example

- Lab1, assignment 3
  - Dump the default configuration of sim-cheetah
  - Modify the configuration and simulate
  - Plot the results (e.g. OpenOffice, Gnuplot, Matlab, Excel)



Outline

- Lab organization and goals
- SimpleScalar architecture and tools
- Lab 5: article review
- Exercises

Lab 5: Article Review

- Select an article on a multi-core, multiprocessor, multi-computer system, or a graphics processor
  - List of papers is available on the course page
  - You may select other articles if your lab assistant agrees
- Review the selected article
- Write a review report on the article
- Self-learning based; No lab session allocated
- Read and understand the paper
  - If the course literature does not help you, investigate the referenced papers

Lab 5: Article Review (cont'd)

- Analyze the paper
- Classify the architecture (e.g. MIMD, SIMD, NUMA)
- Possible questions to ask
  - Why has the actual method/approach been selected?
  - What are the advantages and disadvantages?
  - What is the application area?
  - What has been demonstrated?
  - …

Lab 5: Article Review (cont’d)

- Write a report
  - ~1000 words
  - Submit, in PDF format, to your lab assistant's urkund account

Outline

- Lab organization and goals
- SimpleScalar architecture and tools
- Lab 5: article review
- Exercises

Exercises

Problem 1. Review questions

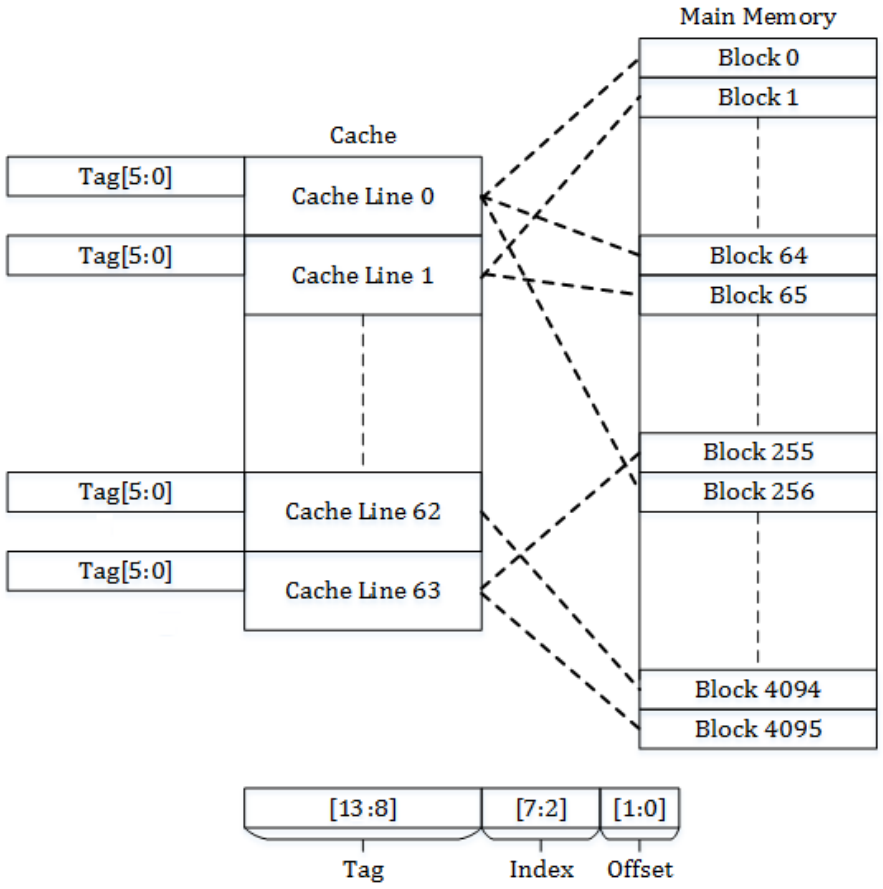
Problem 2 and 3. Mandatory for lab 1

Problem 4 and 5. Additional exercises (if you feel up to the challenge)

Problem 1 (review questions)

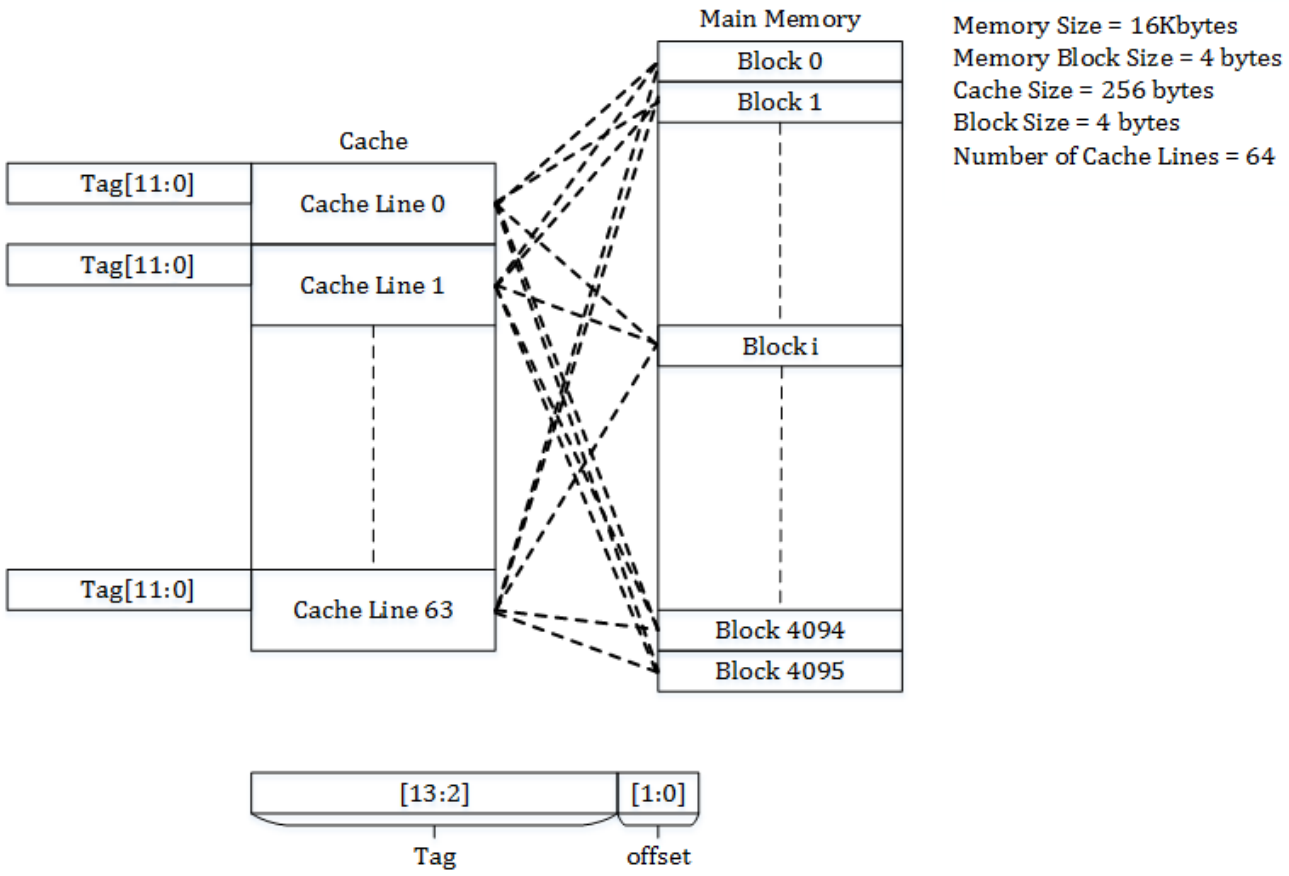
1) What are the differences among direct mapping, associative mapping, and set-associative mapping?
2) What is the distinction between spatial locality and temporal locality?

Cache placement policies (Direct-Mapped)



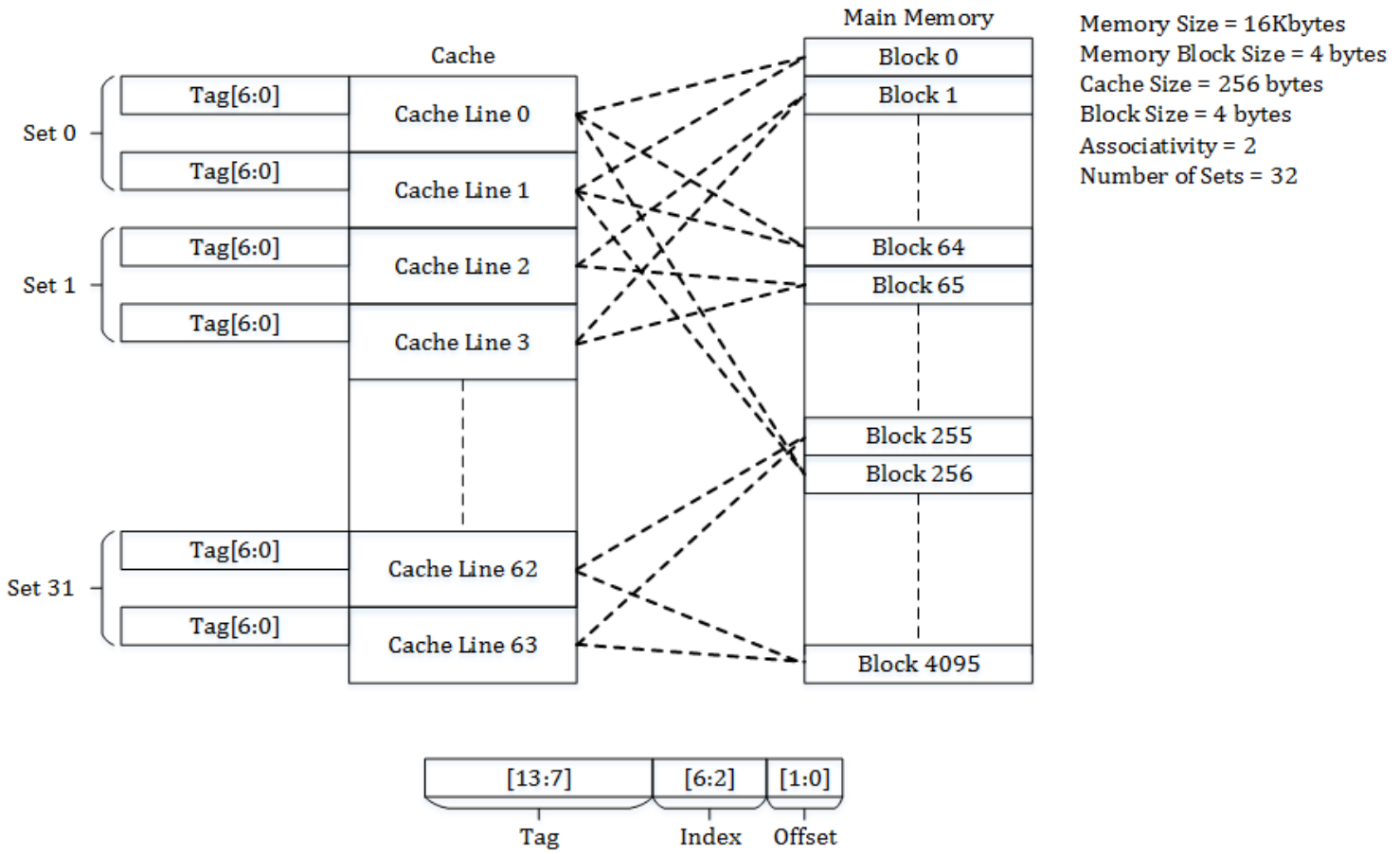
Memory Size = 16Kbytes
Memory Block Size = 4 bytes
Cache Size = 256 bytes
Block Size = 4 bytes
Associativity = 1
Number of Sets = 64

Cache placement policies (Fully-Associative)



Memory Size = 16Kbytes
Memory Block Size = 4 bytes
Cache Size = 256 bytes
Block Size = 4 bytes
Number of Cache Lines = 64

Cache placement policies (Set-Associative)



Memory Size = 16Kbytes
Memory Block Size = 4 bytes
Cache Size = 256 bytes
Block Size = 4 bytes
Associativity = 2
Number of Sets = 32

Question 2

Principle of Locality

Program instructions access a small proportion of their address space at any time

Question 2

Principle of Locality

Program instructions access a small proportion of their address space at any time

- **Temporal locality**
  - Items accessed recently are likely to be accessed again soon

Question 2

Principle of Locality

Program instructions access a small proportion of their address space at any time

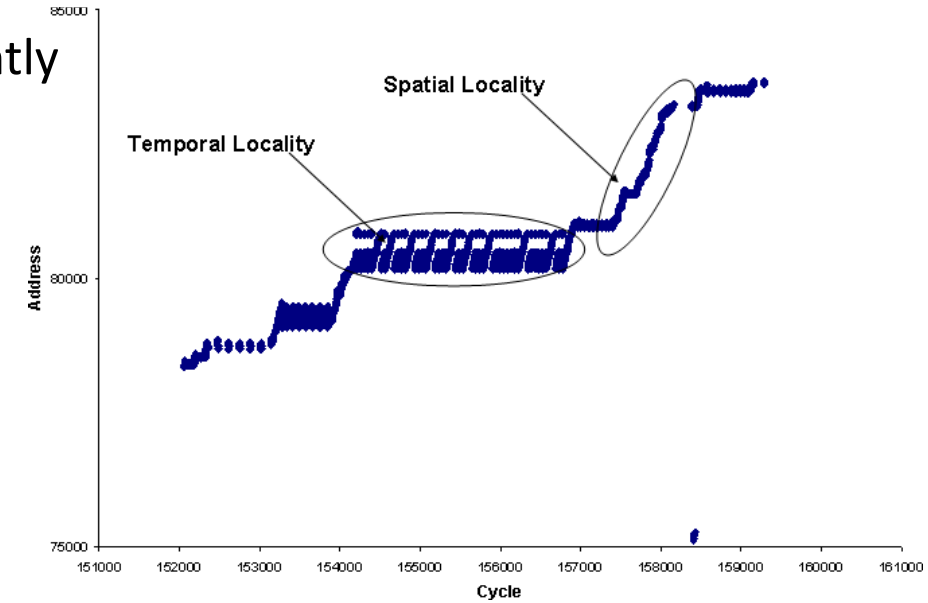
- **Temporal locality**
  - Items accessed recently are likely to be accessed again soon
- **Spatial locality**
  - Items near those accessed recently
    are likely to be accessed soon

Question 2

Principle of Locality

Program instructions access a small proportion of their address space at any time

- **Temporal locality**
  - Items accessed recently are likely to be accessed again soon
- **Spatial locality**
  - Items near those accessed recently are likely to be accessed soon



Problem 2 (mandatory for lab 1)

Consider a machine with a byte addressable main memory of 2^8 bytes and block size of 4 bytes. Assume that a direct mapped cache consisting of 8 lines is used with this machine.

1) How is an 8-bit memory address divided into tag, line number, and byte number?
2) Into what line would bytes with each of the following addresses be stored?

   0001 1011
   0011 0100
   1101 0000
   1010 1010

3) Suppose the byte with address 1010 0001 is stored in the cache. What are the addresses of the other bytes stored along with it?
4) How many total bytes of memory can be stored in the cache?
5) Why is the tag also stored in the cache?

Problem 2 (mandatory for lab 1)

Consider a machine with a byte addressable main memory of 2^8 bytes and block size of 4 bytes. Assume that a direct mapped cache consisting of 8 lines is used with this machine.

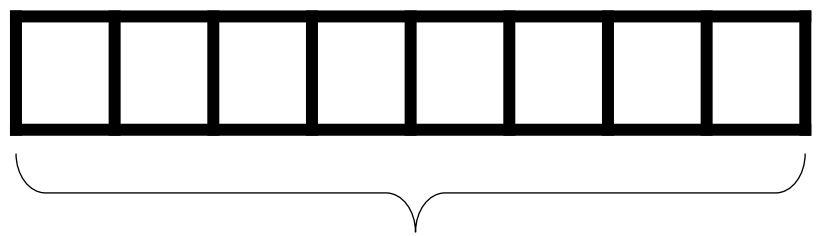
1) How is an 8-bit memory address divided into tag, line number, and byte number?
2) Into what line would bytes with each of the following addresses be stored?

0001 1011

0011 0100

1101 0000

1010 1010



Problem 2 (mandatory for lab 1)

Consider a machine with a byte addressable main memory of 2^8 bytes and block size of 4 bytes. Assume that a direct mapped cache consisting of 8 lines is used with this machine.

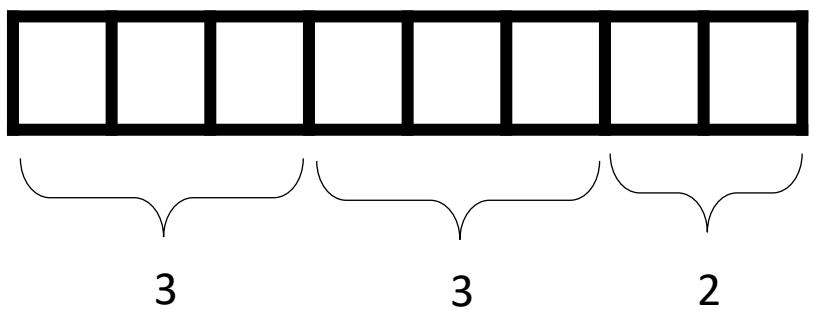
1) How is an 8-bit memory address divided into tag, line number, and byte number?
2) Into what line would bytes with each of the following addresses be stored?

0001 1011

0011 0100

1101 0000

1010 1010

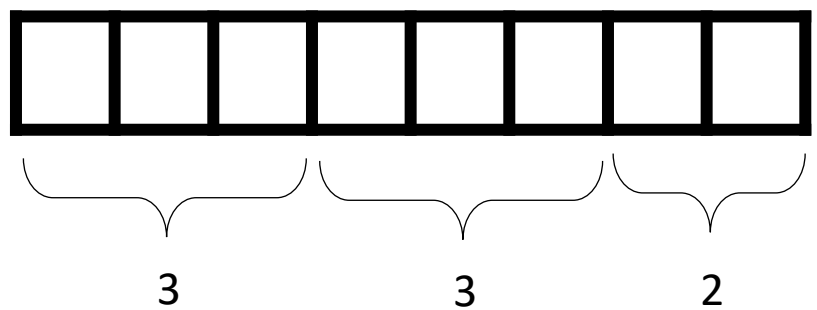


Problem 2 (mandatory for lab 1)

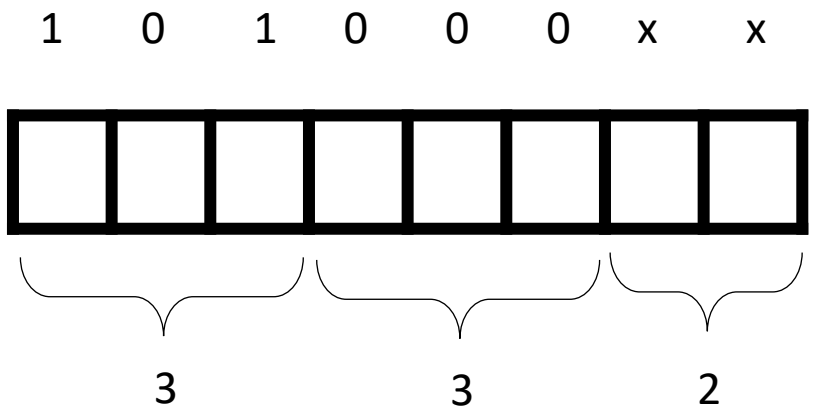
Consider a machine with a byte addressable main memory of 2^8 bytes and block size of 4 bytes. Assume that a direct mapped cache consisting of 8 lines is used with this machine.

1) How is an 8-bit memory address divided into tag, line number, and byte number?
2) Into what line would bytes with each of the following addresses be stored?

0001 1011
0011 0100
1101 0000
1010 1010

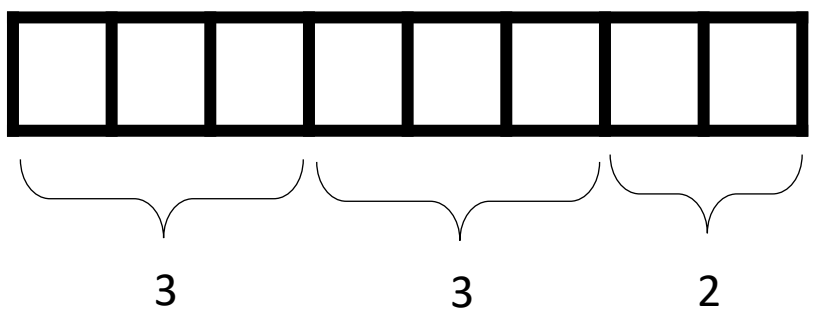


3) Suppose the byte with address 1010 0001 is stored in the cache. What are the addresses of the other bytes stored along with it?
4) How many total bytes of memory can be stored in the cache?
5) Why is the tag also stored in the cache?

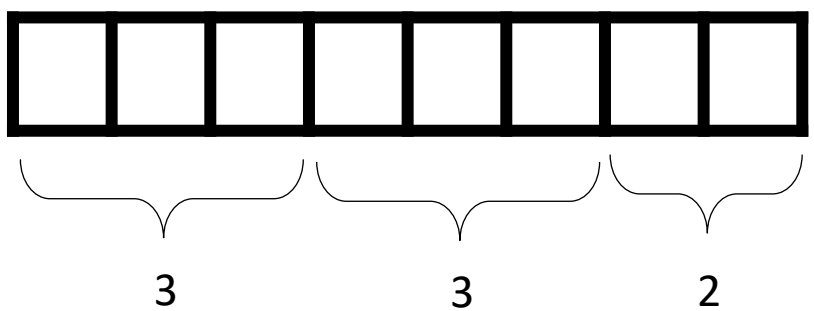


3) Suppose the byte with address 1010 0001 is stored in the cache. What are the addresses of the other bytes stored along with it?
4) How many total bytes of memory can be stored in the cache?
5) Why is the tag also stored in the cache?

8 lines x 4 bytes = 32 bytes

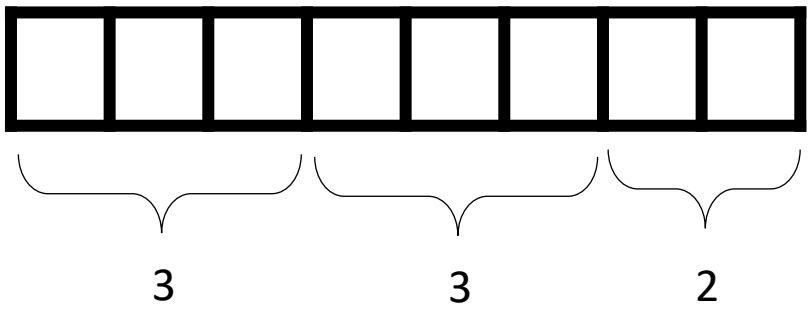


3) Suppose the byte with address 1010 0001 is stored in the cache. What are the addresses of the other bytes stored along with it?
4) How many total bytes of memory can be stored in the cache?
5) Why is the tag also stored in the cache?



3) Suppose the byte with address 1010 0001 is stored in the cache. What are the addresses of the other bytes stored along with it?
4) How many total bytes of memory can be stored in the cache?
5) Why is the tag also stored in the cache?

Because we have a large main memory but a limited and finite set of cache lines. More than one address go in a particular cache line. We need tag to identify which block is in the cache line.



Problem 3 (mandatory for lab 1)

Consider the following code:

```
cout << "Hello World";
cin >> a;
for(i = 0; i < 50; i++)
    cout<<i;
```

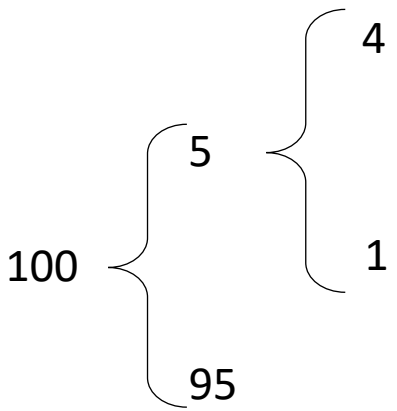
1) Give one example of the spatial locality in the code.
2) Give one example of the temporal locality in the code.

Problem 4 (additional)

Average memory-access time. A computer has a cache, main memory, and a disk used for virtual memory. If a referenced word is in the cache, 15 ns are required to access it. If it is in main memory but not in the cache, 70 ns are needed to load it into the cache, and then the reference is started again. If the word is not in main memory, 10 ms are required to fetch the word from disk, followed by 50 ns to copy it to the cache, and the reference is started again. The cache hit ratio is 0.95 and the main memory hit ratio is 0.8. what is the average time in nanoseconds required to access a referenced word on this system?

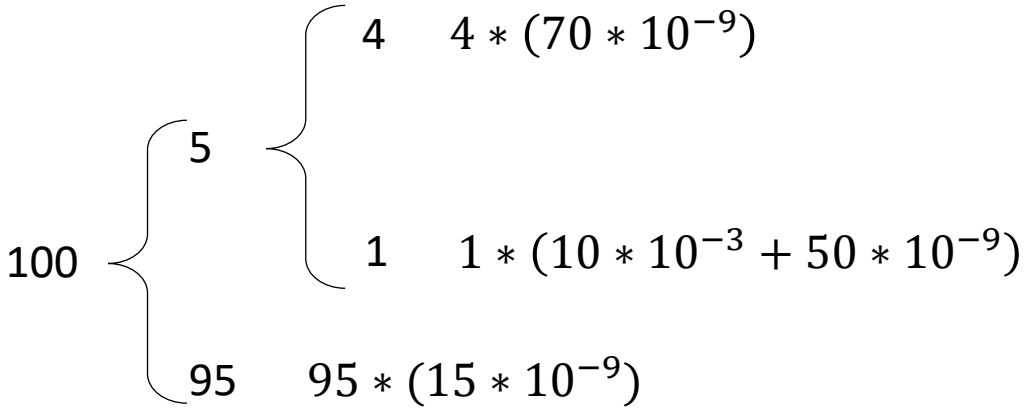
Problem 4 (additional)

Average memory-access time. A computer has a cache, main memory, and a disk used for virtual memory. If a referenced word is in the cache, 15 ns are required to access it. If it is in main memory but not in the cache, 70 ns are needed to load it into the cache, and then the reference is started again. If the word is not in main memory, 10 ms are required to fetch the word from disk, followed by 50 ns to copy it to the cache, and the reference is started again. The cache hit ratio is 0.95 and the main memory hit ratio is 0.8. what is the average time in nanoseconds required to access a referenced word on this system?



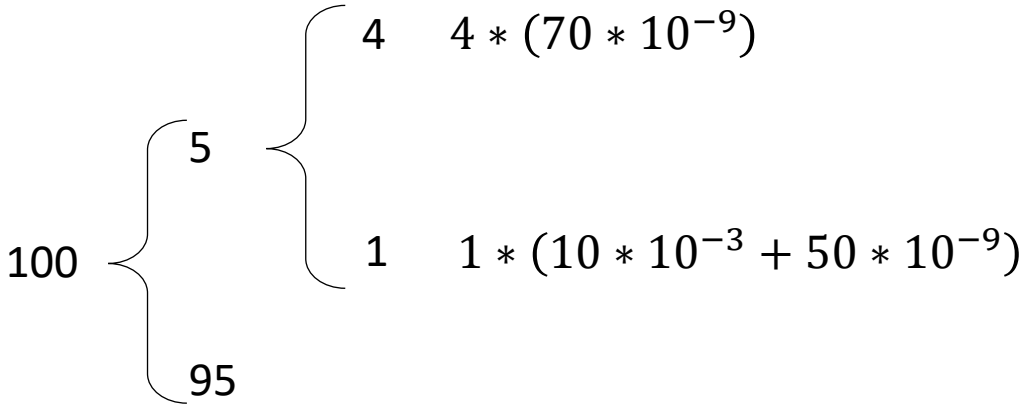
Problem 4 (additional)

Average memory-access time. A computer has a cache, main memory, and a disk used for virtual memory. If a referenced word is in the cache, 15 ns are required to access it. If it is in main memory but not in the cache, 70 ns are needed to load it into the cache, and then the reference is started again. If the word is not in main memory, 10 ms are required to fetch the word from disk, followed by 50 ns to copy it to the cache, and the reference is started again. The cache hit ratio is 0.95 and the main memory hit ratio is 0.8. what is the average time in nanoseconds required to access a referenced word on this system?



Problem 4 (additional)

Average memory-access time. A computer has a cache, main memory, and a disk used for virtual memory. If a referenced word is in the cache, 15 ns are required to access it. If it is in main memory but not in the cache, 70 ns are needed to load it into the cache, and then the reference is started again. If the word is not in main memory, 10 ms are required to fetch the word from disk, followed by 50 ns to copy it to the cache, and the reference is started again. The cache hit ratio is 0.95 and the main memory hit ratio is 0.8. what is the average time in nanoseconds required to access a referenced word on this system?



$$\frac{1 * (10 * 10^{-3} + 50 * 10^{-9}) + 4 * (70 * 10^{-9}) + 95 * (15 * 10^{-9})}{100}$$

Problem 5 (additional)

Performance enhancement using cache. A computer system contains a main memory of 32K 16-bit words. It also has a 4K-word cache divided into four-line sets with 64 words per line. Assume that the cache is initially empty. The processor fetches words from locations 0, 1, 2, …, 4351 in that order. If then repeats this fetch sequence nine more times. The cache is 10 times faster than main memory. Estimate the improvement resulting from the use of the cache. Assume an LRU policy for block replacement

Problem 5 (additional)

Performance enhancement using cache. A computer system contains a main memory of 32K 16-bit words. It also has a 4K-word cache divided into four-line sets with 64 words per line. Assume that the cache is initially empty. The processor fetches words from locations 0, 1, 2, …, 4351 in that order. If then repeats this fetch sequence nine more times. The cache is 10 times faster than main memory. Estimate the improvement resulting from the use of the cache. Assume an LRU policy for block replacement

4 x 1024 = 4 x 64 x #sets ➔ #sets = 16

set	tag	data	tag	data	tag	data	tag	data
0		64 word						
1								
2								
3								
	…	…	…	…	…	…	…	…
15								

Cache structure

set	tag	data	tag	data	tag	data	tag	data
0		[0-63]		[1024-1087]		[2048-2111]		[3072-4159]
1		[64-127]						
2								
3								
	...	...	...	...	...	...	...	...
15		[960-1023]		[1984-2047]		[3008-3071]		[4032-4095]

Cache structure

set	tag	data	tag	data	tag	data	tag	data
0		[4096-4159]		[1024-1087]		[2048-2111]		[3072-4159]
1		[4160-4223]						
2		[4223-4287]						
3		[4287-4351]						
	…	…	…	…	…	…	…	…
15		[960-1023]		[1984-2047]		[3008-3071]		[4032-4095]

Cache structure

1st round: 4 x 16 + 4 misses
2nd-9th round: 4 x 4 + 4 misses
Total misses: 4 x 16 + 4 + (4 x 4 + 4) x 9 = 248

With cache: 248 x 10s + (4351 x 10 – 248) x 1s = 45742
Without cache: 4351 x 10 x 10s = 435100

Speed-up:
435100/45742 ~= 9.5