



# Politechnika Wrocławska

## Computer Architecture and Organization

### Lecture 6

Dr. Radosław Michalski

Department of Computational Intelligence, Faculty of Computer Science  
and Management, Wrocław University of Science and Technology

Version 1.0, spring 2017



# Source and licensing

The most current version of this lecture is here:  
<https://github.com/rmhere/lecture-comp-arch-org>

This material is licensed by Creative Commons Attribution  
NonCommercial ShareAlike license 4.0 ([CC BY-NC-SA 4.0](#)).



# Overview of this lecture

**Embedding assembly in C**

**Floating point representation**

**Caching**



# Embedding assembly in C

## Demo

Jain.PK - Using Inline Assembly in C/C++



# Floating point representation

## Integers

How to work with integer numbers in computer systems?

- ▶ exemplary integer: 1283093714 (31 bits)
- ▶ integers - precise representation
- ▶ maximum length - as defined by architecture
- ▶  $2^n$ , where  $n$  represents the number of bits
- ▶ signed/unsigned
- ▶ overflow



# Floating point representation

## Real numbers - introduction

How to work with real numbers in computer systems?

- ▶ exemplary real number: 3.82379102
- ▶ no possibility of holding some real numbers precisely
- ▶ registers have fixed length (32 bits in case of MIPS)
- ▶ precision or approximation
- ▶ how to use these 32 bits effectively?
- ▶ fixed point vs. floating point



# Floating point representation

Real numbers - fixed point

*integerpart . fraction*

3.82379102

00000011 (8 bits) . 100111010010000000101011110 (28 bits)



# Floating point representation

Real numbers - floating point

*significand* \* *base*<sup>*exponent*</sup>

$$3.82379102 = 382379102 * 10^{-8}$$





# Floating point representation

Real numbers - IEEE 754 standard

IEEE 754 / binary32

- ▶ sign bit (1 bit)
- ▶ exponent (8 bits)
- ▶ significand/mantissa (24 bits, 1 bit implicit)
- ▶ base: 2



# Floating point representation

Real numbers - binary32

$$\text{significand} * 2^{\text{exponent}}$$

3.82379102

0 (sign)

10000000 (exponent - 1)

11101001011100011111110 (mantissa - 1.9114999771118164)



# Floating point representation

Real numbers - IEEE 754 standard

IEEE 754 / binary64

- ▶ sign bit (1 bit)
- ▶ exponent (11 bits)
- ▶ significand/mantissa (53 bits, 1 bit implicit)
- ▶ base: 2



# Floating point representation

## Real numbers - MIPS

- ▶ MIPS has 32 single precision (32-bit) floating point registers.
- ▶ \$f0 – \$f31
- ▶ \$f0 is not special
- ▶ special instructions that work on single precision
- ▶ these cannot use general purpose registers, only floating point



# Floating point representation

## Double precision in MIPS

- ▶ using the same sets of registers pairwise, e.g., \$f0 and \$f1
- ▶ addressing the first register from a pair, e.g., \$f0, \$f2
- ▶ instructions for integer, single and double precision arithmetic
  - ▶ add - integers
  - ▶ add.s - single precision
  - ▶ add.d - double precision



# Floating point representation

## Sources & recommended materials

- ▶ S. Hollasch, [IEEE Standard 754 Floating Point Numbers](#) (website)
- ▶ Wikipedia, [IEEE floating point](#) (website)
- ▶ H. Schmidt, [IEEE-754 Floating Point Converter](#) (website)
- ▶ J. King, [IEEE Floating Point Standard \(The Implicit 1\)](#) (video)



# Caching

## Introduction

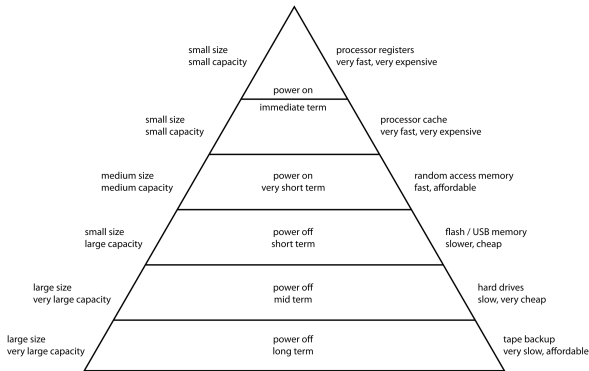
- ▶ from computer's perspective (RISC):
  - ▶ registers
  - ▶ memory
- ▶ memory is continuous
- ▶ the CPU does not know with what type of memory it interacts
- ▶ this is why we can introduce different strategies regarding memory



# Memory hierarchy

## Schema

### Computer Memory Hierarchy



*Computer memory hierarchy*, public domain





# Caching

## Memory access times

Processor registers:

- ▶ 32 \* 32 bits (registers) + 32 \* 32 bits (floating point registers)
- ▶ the fastest, matched in speed to the CPU
- ▶ 0.25 ns

Cache:

- ▶ megabytes
- ▶ 1ns

RAM:

- ▶ gigabytes
- ▶ 20ns

External memory

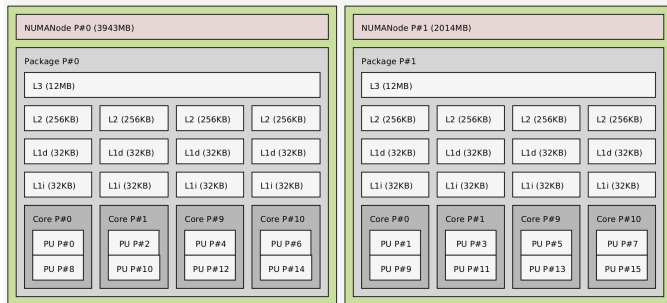


# Memory hierarchy

## Lstopo output

Lenovo™ ThinkStation® D20, Intel® Xeon™ E5640, 6 GB RAM

Machine (5958MB total)



Screenshot from the application lstopo (package [Portable Hardware Locality](#))



# Caching

## Accessing cache

- ▶ cache hit
- ▶ cache miss
- ▶ spatiality
- ▶ temporality



# Caching

## Sources & recommended materials

- ▶ Imagination Technologies Limited, **MIPS Software Training - caches**, Hertfordshire, UK (training materials)
- ▶ J. Kwiatkowski, *"Computer Architecture and Organization"*, Wrocław University of Science and Technology (course materials)