

February 18, 2025

Announcements

- Contest results
- HW2
- Talks start Thu

Goals

- memory subsystem
- multi cores (sockets)

Review

- asymptotic optimality
- pebble game
- mixed news

Alignment

Alignment describes the process of matching the base address of:

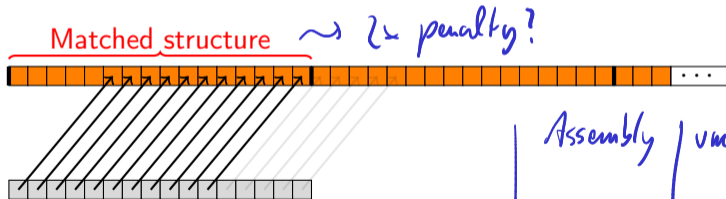
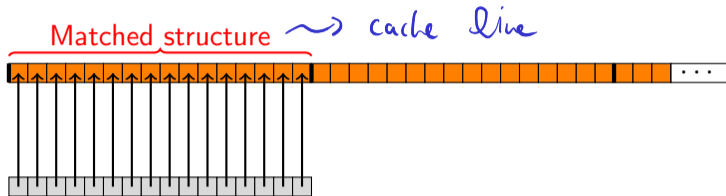
- ▶ Single word: double, float
- ▶ SIMD vector
- ▶ Larger structure

To machine granularities:

- Cache line
- Memory page
- Word size
- SIMD vector size
- DRAM access

Q: What is the performance impact of misalignment?

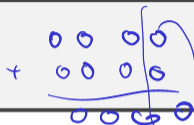
Performance Impact of Misalignment



Assembly | `vmovq %a | %ps`
C | `__attribute__((aligned(8)))`

SIMD: Basic Idea

What's the basic idea behind SIMD?



What architectural need does it satisfy?

Amortize control overheads vs. work done
architecturally + per instruction

Typically characterized by width of data path:

- ▶ SSE: 128 bit (4 floats, 2 doubles)
- ▶ AVX-2: 256 bit (8 floats, 4 doubles)
- ▶ AVX-512: 512 bit (16 floats, 8 doubles)

SIMD: Architectural Issues



Realization of inter-lane comm. in SIMD? Find instructions.

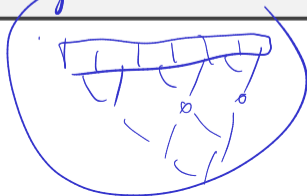
- Reduction / Broadcast
- Permutations
- misaligned mem. access (no!)

Name tricky/inefficient aspects in terms of expressing SIMD:

- Control flow \rightarrow Masking
- Re-convergence

$a[i]$
 $a[2*i]$
 $a[ind[i]]$

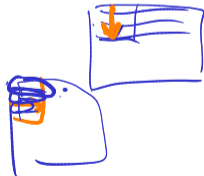
- Scatter
Gather



SIMD: Intel Instructions

x86 SIMD suffixes: What does the "ps" suffix mean? "sd"?

ps → packed single
(sd) → scalar double



SIMD: Transposes

Why are transposes important? Where do they occur?

Whenever SIMD runs into non-stride-1 access

Example implementation aspects:

- ▶ HPTT: [[Springer et al. '17](#)]
- ▶ github: [springer13/hptt](#) [8x8 transpose microkernel](#)
- ▶ Q: Why 8x8?

Outline

Introduction

Notes

Notes (unfilled, with empty boxes)

Notes (source code on Github)

About This Class

Why Bother with Parallel Computers?

Lowest Accessible Abstraction: Assembly

Architecture of an Execution Pipeline

Architecture of a Memory System

Shared-Memory Multiprocessors

Machine Abstractions

Performance: Expectation, Experiment, Observation

Performance Oriented Languages and Abstractions

Multiple Cores vs Bandwidth

Assume:

- ▶ memory latency of 100 ns
- ▶ peak DRAM bandwidth of 100 GB/s (per socket)

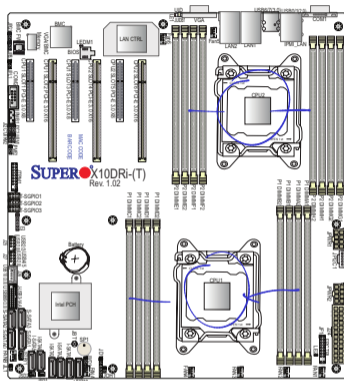
How many cache lines should be/are in flight at one time?

networking: bandwidth-delay product: 10 kB
156 cache lines

Historical ops: track 10 in-flight mem requests
per core

[McCalpin '18]

Topology and NUMA



[SuperMicro Inc. '15]

Demo:

- ▶ Show `lstopo` on porter, from [hwloc](#).
- ▶ [lstopo on MI300](#)

Placement and Pinning

Who decides on what core my code runs? How?

A large, empty rectangular box with a thin black border, intended for the user to provide an answer to the question above.

Who decides on what NUMA node memory is allocated?

A large, empty rectangular box with a thin black border, intended for the user to provide an answer to the question above.

[Demo: intro/NUMA and Bandwidths](#)

What is the main expense in NUMA?

A large, empty rectangular box with a thin black border, intended for the user to provide an answer to the question above.