Scalevisor: using hypervision to build a memory driver for NUMA machines

M2 Internship Report

Nicolas Derumigny

ENS Lyon - Télécom SudParis - UVSQ

28th May 2019

Introduction

- Both memory and computation are becoming more and more complex in HPC due to NUMA effect and specific x86 extensions

Introduction

- Both memory and computation are becoming more and more complex in HPC due to NUMA effect and specific x86 extensions

→ Need for a CPU/Memory driver

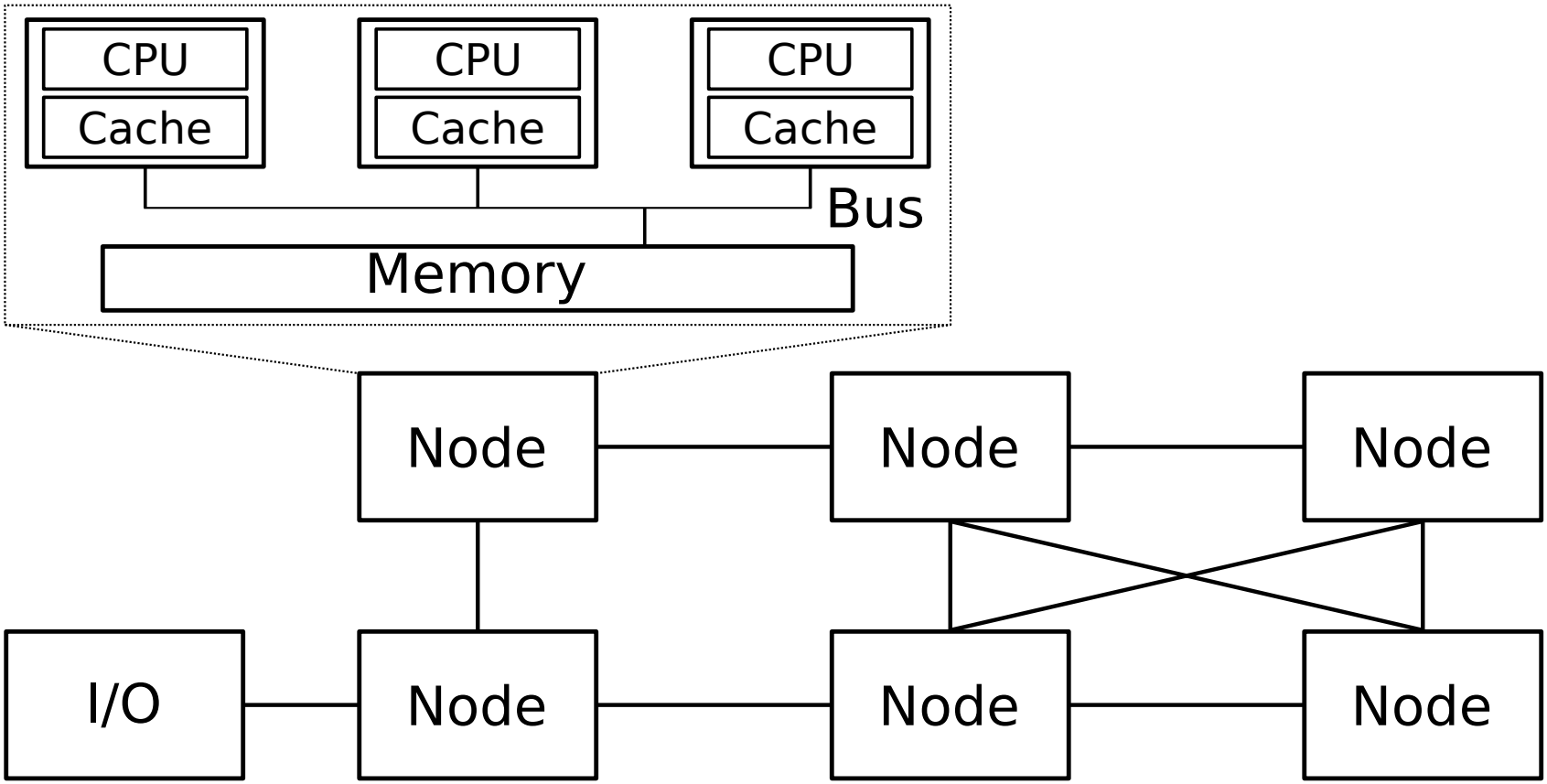
Introduction

- Both memory and computation are becoming more and more complex in HPC due to NUMA effect and specific x86 extensions

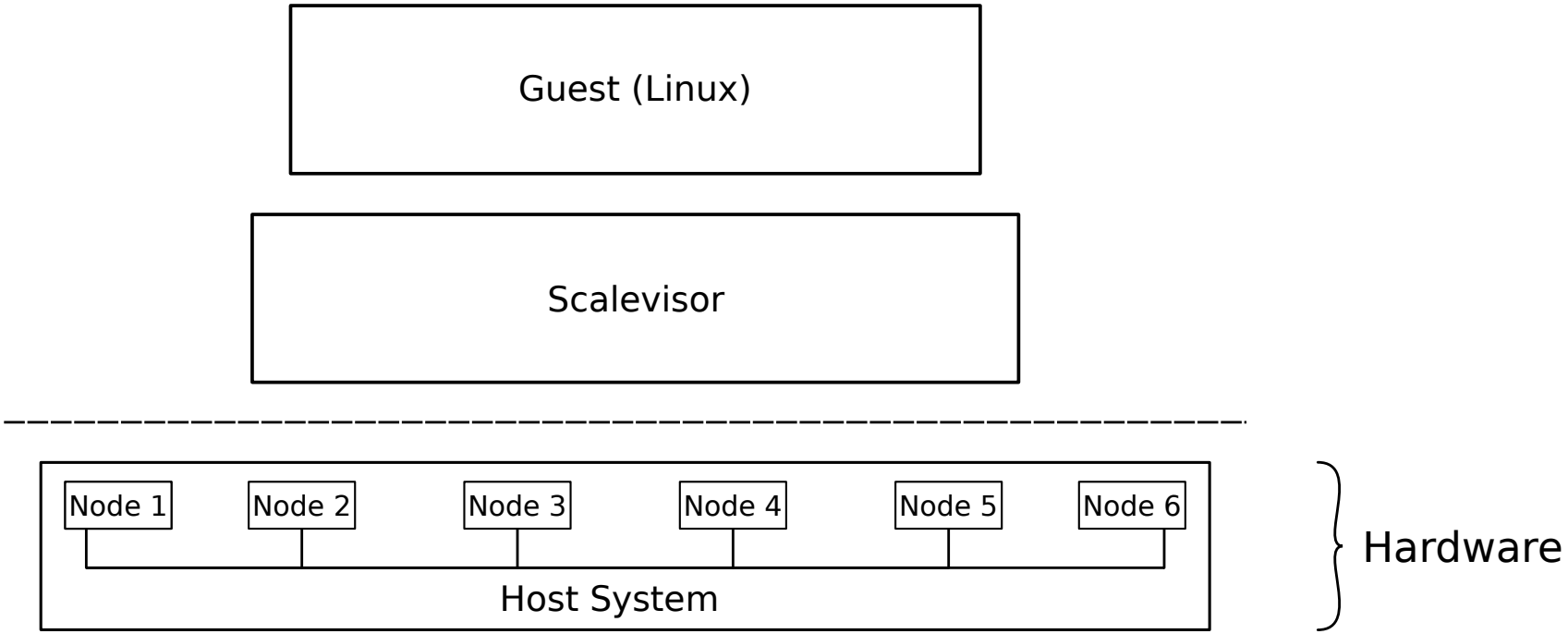
→ Need for a CPU/Memory driver

- The solution: *Scalevisor*, a NUMA-optimized hypervisor
  - Using virtualisation extensions...
  - ... to run a non-modified Linux kernel
  - under which threads and memory allocation are reorganised according to CPU-specific policies

Reminders about multicore systems



Hypervision



State of the implementation pre-internship

- Coded in C++
- No standard library (pre-booted execution) → some limitations + partially recoded

- Migration from a partly-working implementation from an AMD server to an Intel one
- Only early adaptations are complete
- Ill-documented → path to achieve a full-boot is not easy to divide
- → Development of stand-alone modules for future use in the project

Coherent boot of the APs

- One core named *BootStrap Processor* boots first
- Has to wake up the others Application Processors
  - $\rightarrow$ By sending a specific sequence of signals

- Was broken
- Overwriting of an ACPI table (SRAT) $\rightarrow$ misbehaviour
- Caused by an overflow of the page table - the structure dedicated to the correspondence between virtual and hardware addresses

CPUInfo Module

Issue:

- Several features available
- Extensions of the X86 ISA
- Same detection method
- Different enabling method
- Formerly disseminated throughout the code

Solution: the CPUInfo module

- Static C++ class entirely built at compile time
- Standardised detection
- Parsing of the `CPUID` assembly instruction
- Built-in API for the declaration of activator and deactivator

PEBS Module

Precise Event-Based Sampling

- Intel's implementation of Instruction Based Sampling
- Configuration via dedicated registers
- Supplementary layer built over Performance Counters

Solution: the PEBS Module

- Standardised high-level interface
- Automatic management of available counters
- C++ Iterator to browse the records

- Scalevisor will be optimised for a specific CPU...

- Scalevisor will be optimised for a specific CPU...
- ...so we need specific characteristics of this CPU...

- Scalevisor will be optimised for a specific CPU...
- ...so we need specific characteristics of this CPU...
- ...that are not always disclosed by the vendor.

- Scalevisor will be optimised for a specific CPU...
- ...so we need specific characteristics of this CPU...
- ...that are not always disclosed by the vendor.

Solution: Microbench, a NUMA memory latency measurement tool

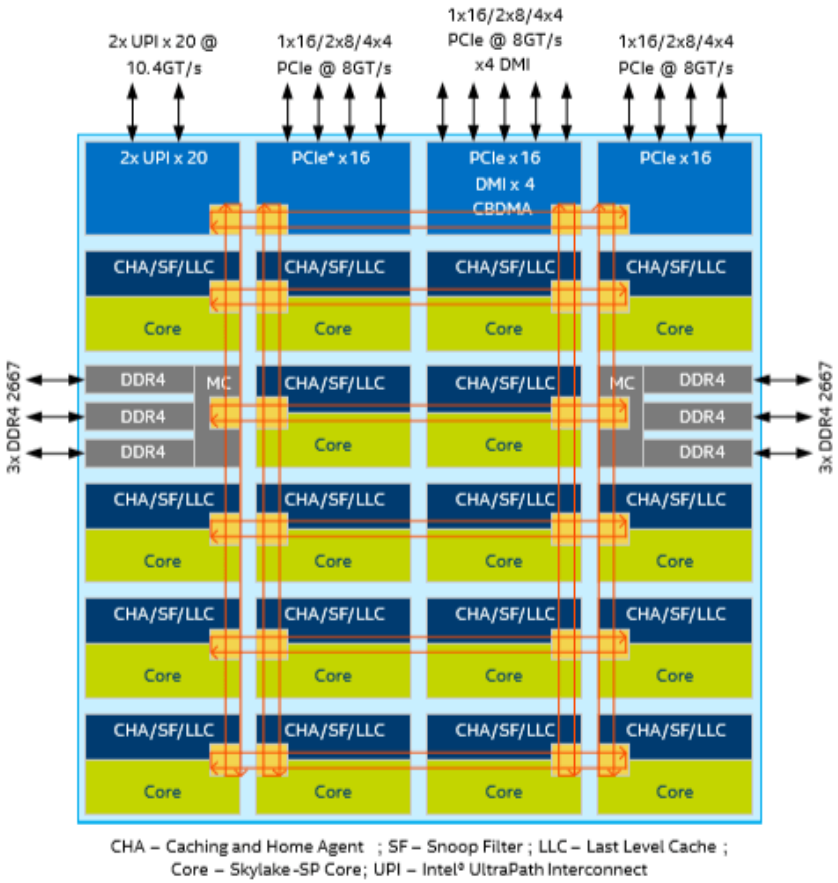
- Accurate *measures* in CPU cycles
- Target manycore systems
- Open Source
- Output in CSV files or pretty tabular
- R toolchain to generate `ggplot2` graphs

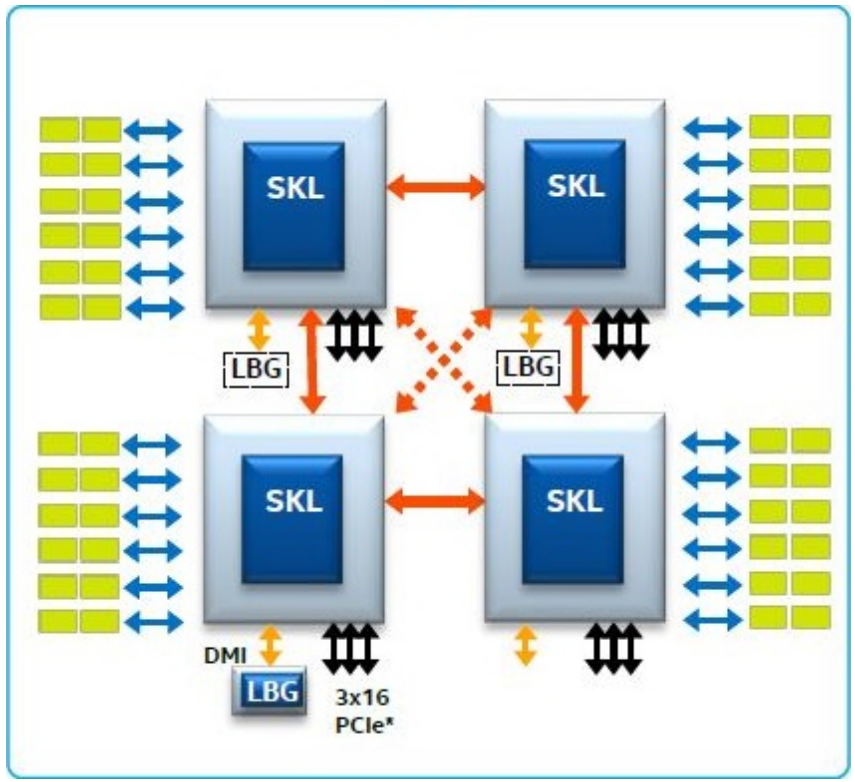
- The system causes noises $\rightarrow$ using FIFO scheduler

- The system causes noises $\rightarrow$ using FIFO scheduler
- The data may not fit in the cache $\rightarrow$ Use a smaller array than the desired cache size + randomized accesses

- The system causes noises → using FIFO scheduler
- The data may not fit in the cache → Use a smaller array than the desired cache size + randomized accesses
- The CPU changes its frequency and voltage → deactivate Turbo + take several measurements, output the minimum or the medianfa + alternate the first core to be tested on the NUMA node

Intel: Architecture





Intel: Latencies

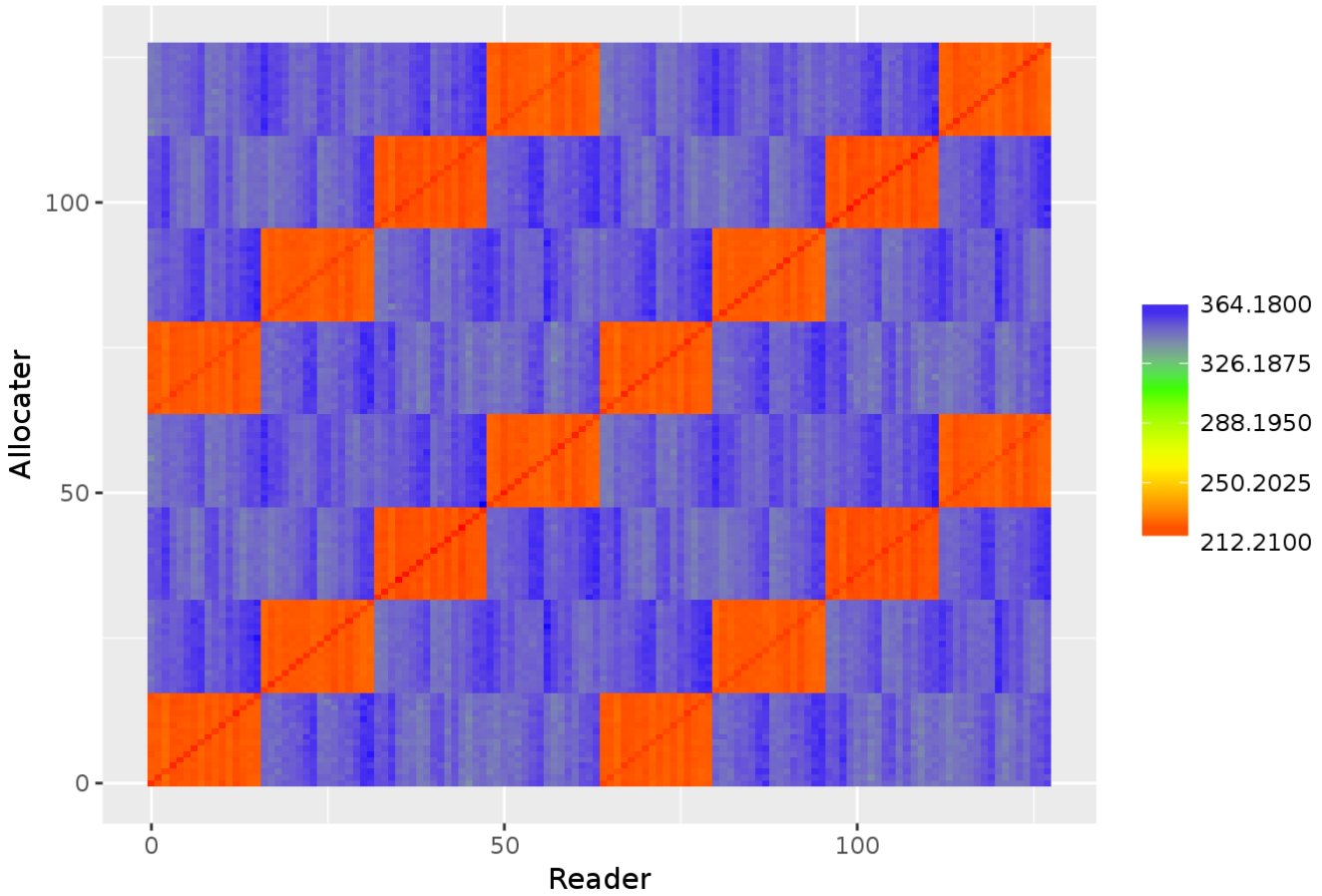
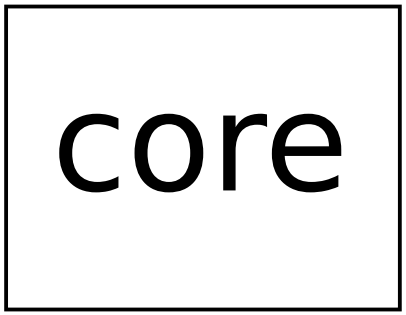


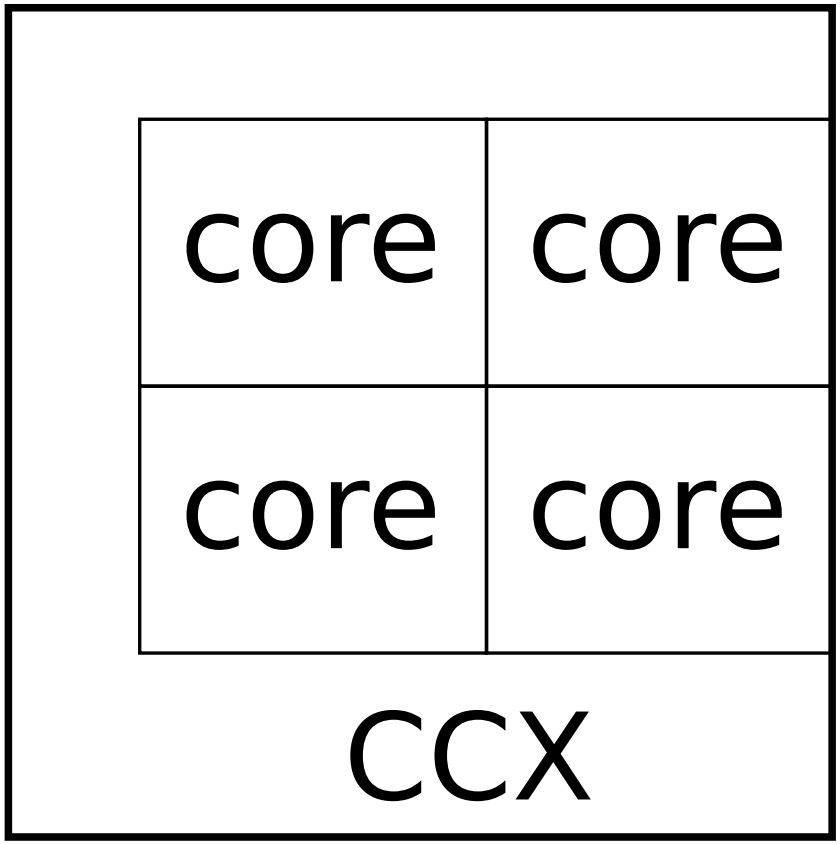
Figure: Latencies of 32-bit loads on a quad-socket Intel Xeon Gold 6130 (4x16 cores @ 2.1 GHz) depending on the allocator and the loader

AMD: Architecture



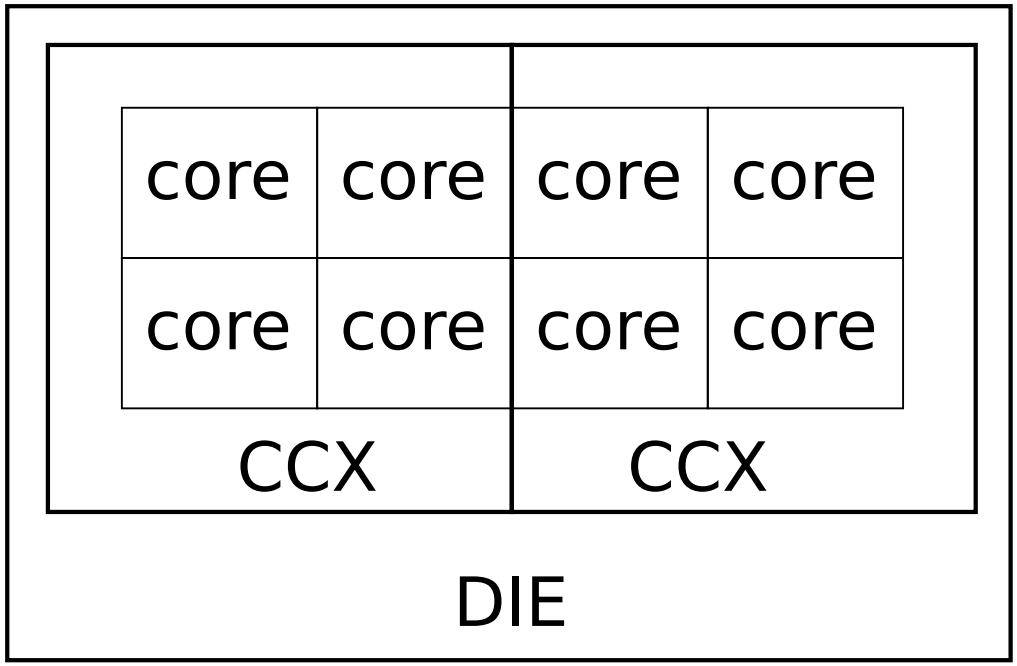
- X86 Out-of-Order Core
- Simultaneous Multi-Threading: sharing of compute units between two logical cores per physical core

AMD: Architecture

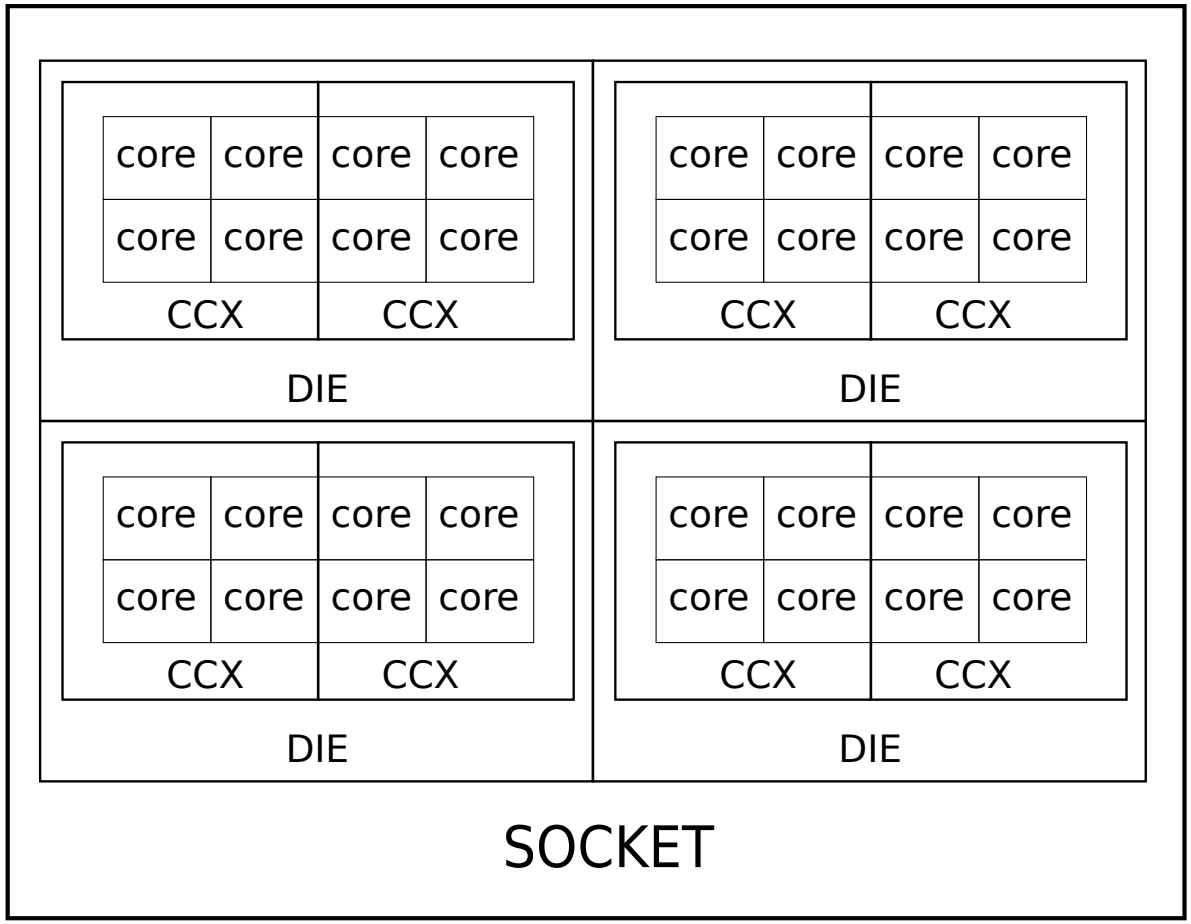


- Owns a memory controller

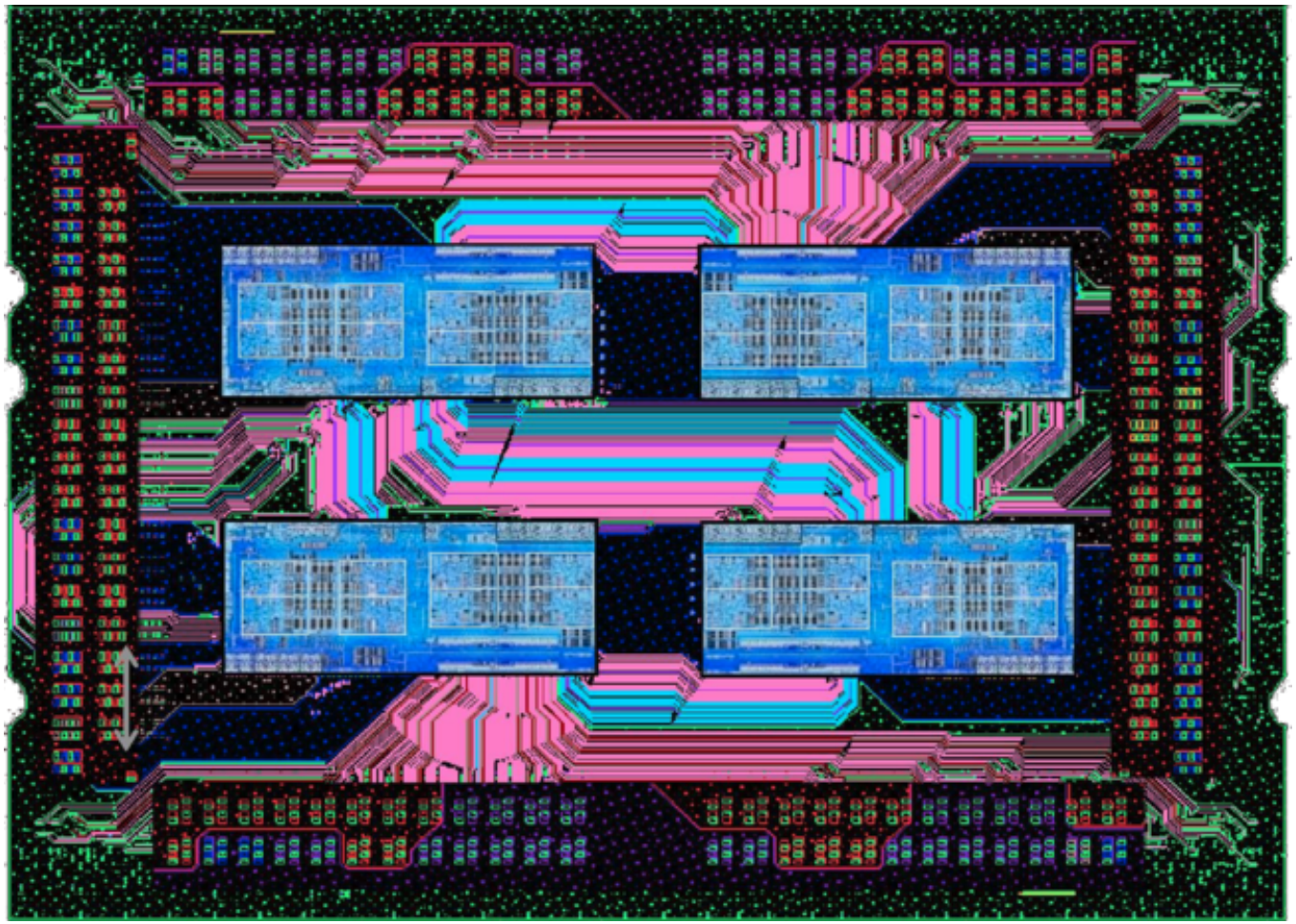
AMD: Architecture



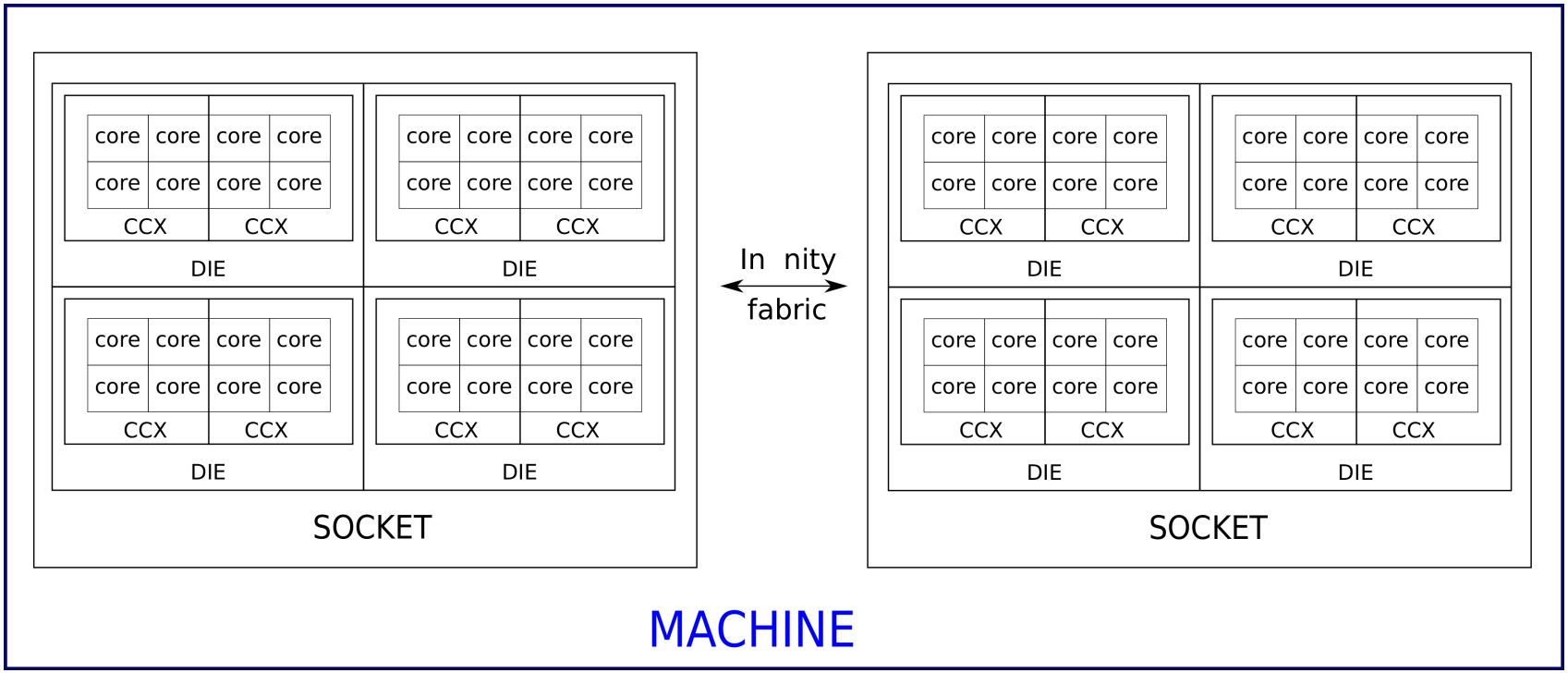
AMD: Architecture



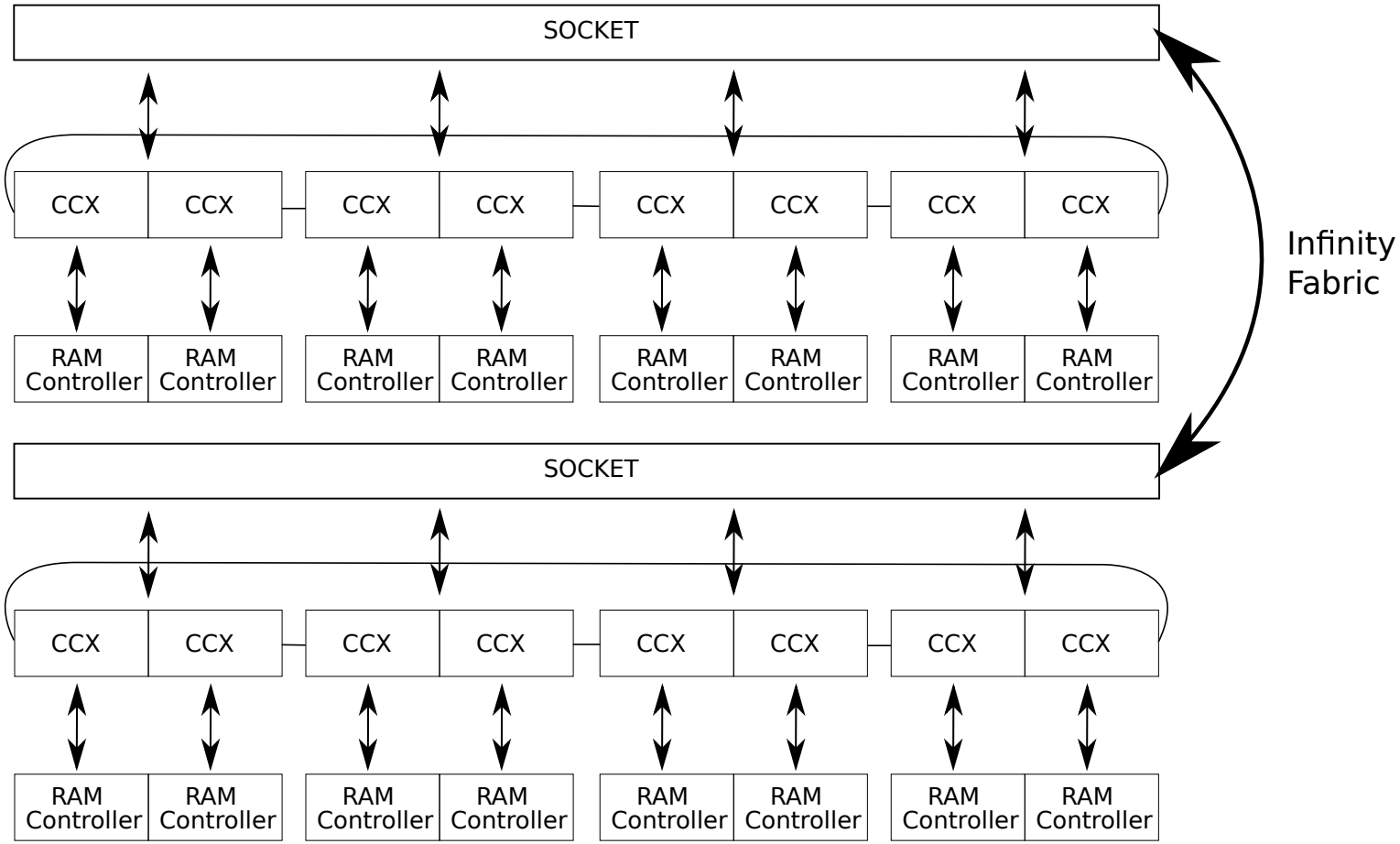
AMD: Architecture



AMD: Architecture



AMD: Topology



AMD: Latencies

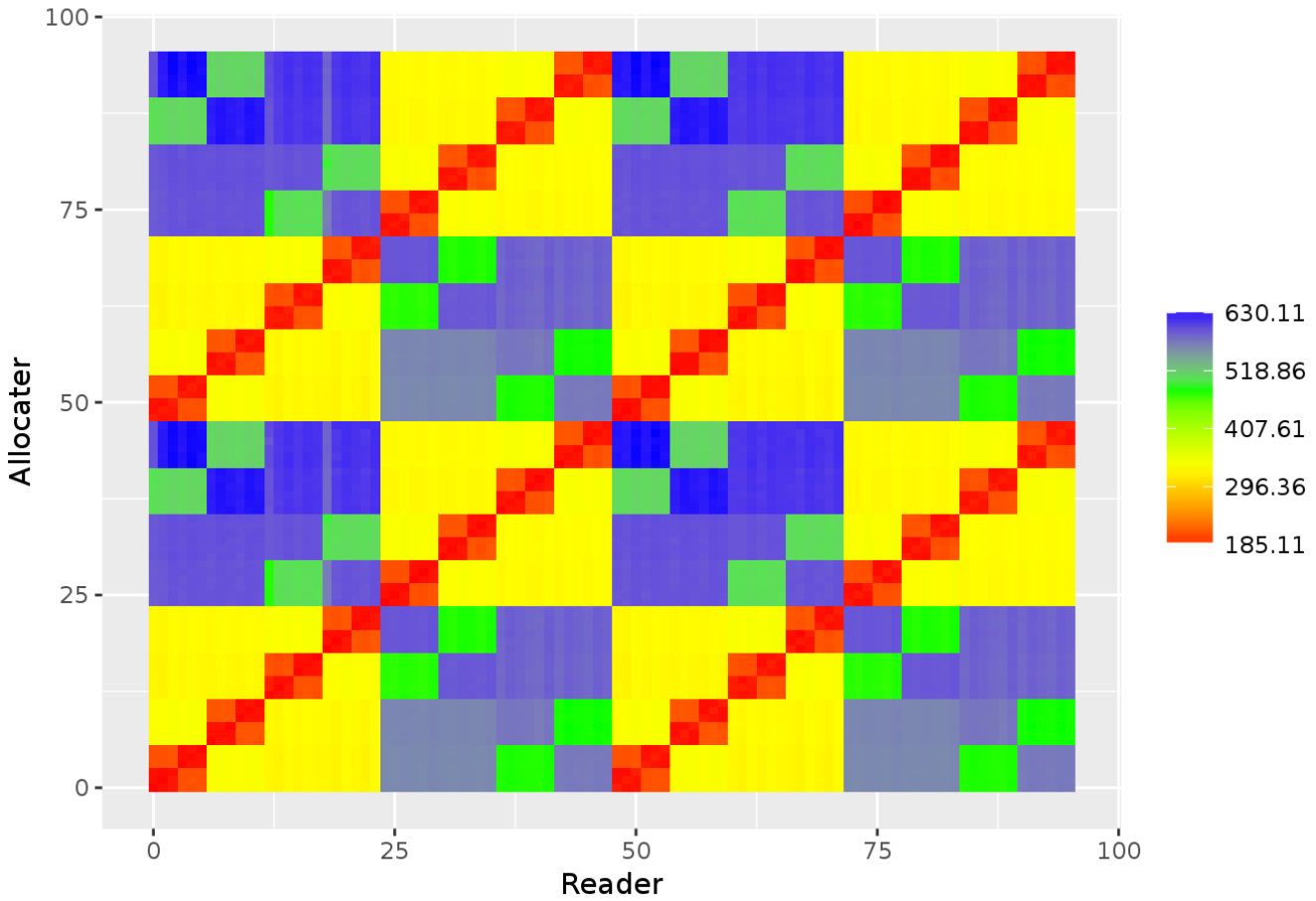
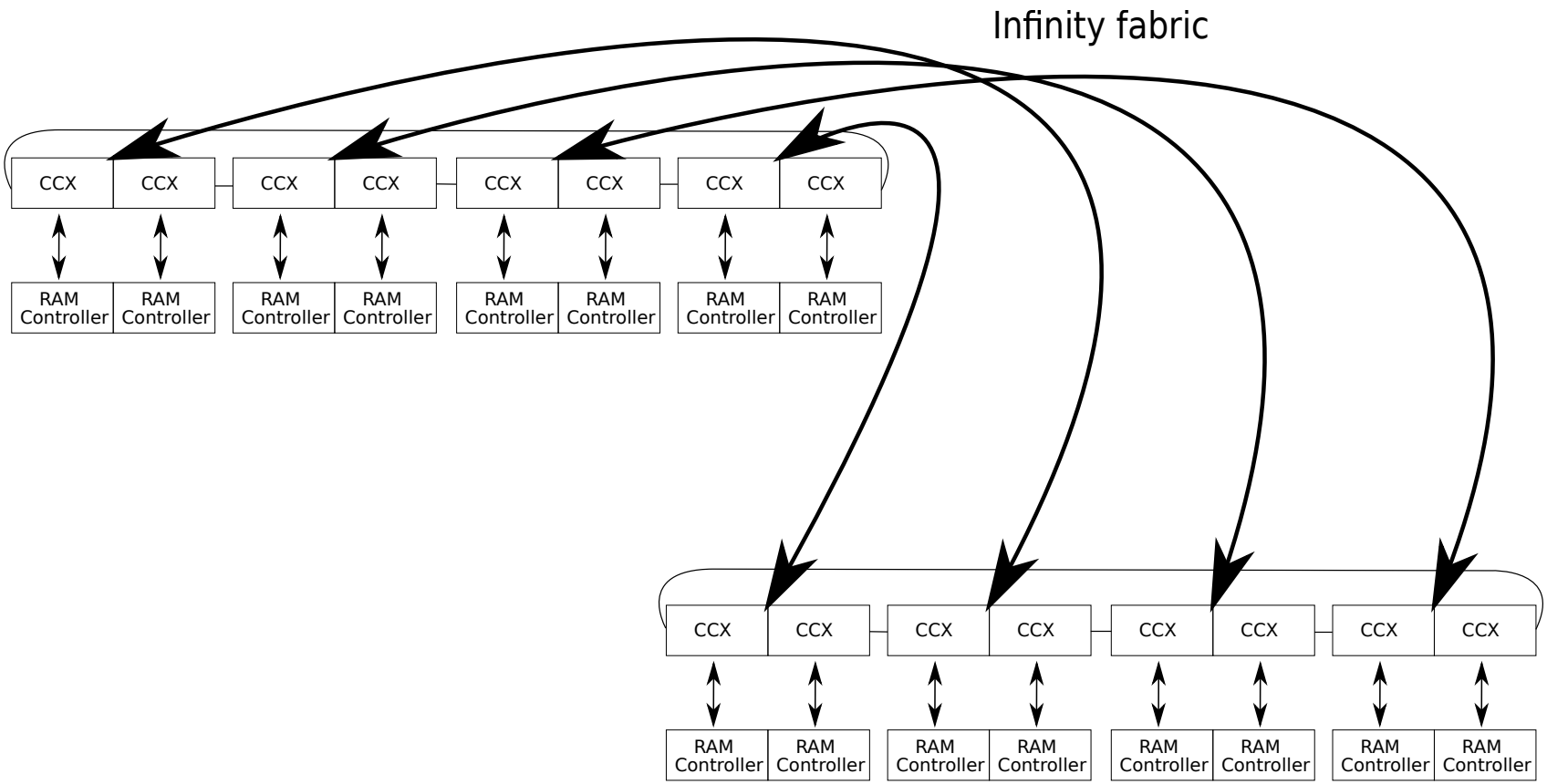


Figure:
Latencies of 32-bit loads on a dual-socket EPYC 7451 (2x24 cores @ 2.2 GHz) depending on the allocator and the loader

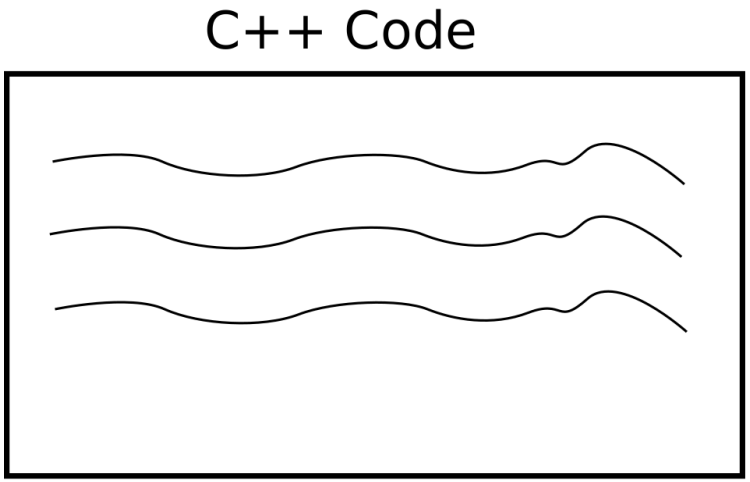
AMD: Real topology

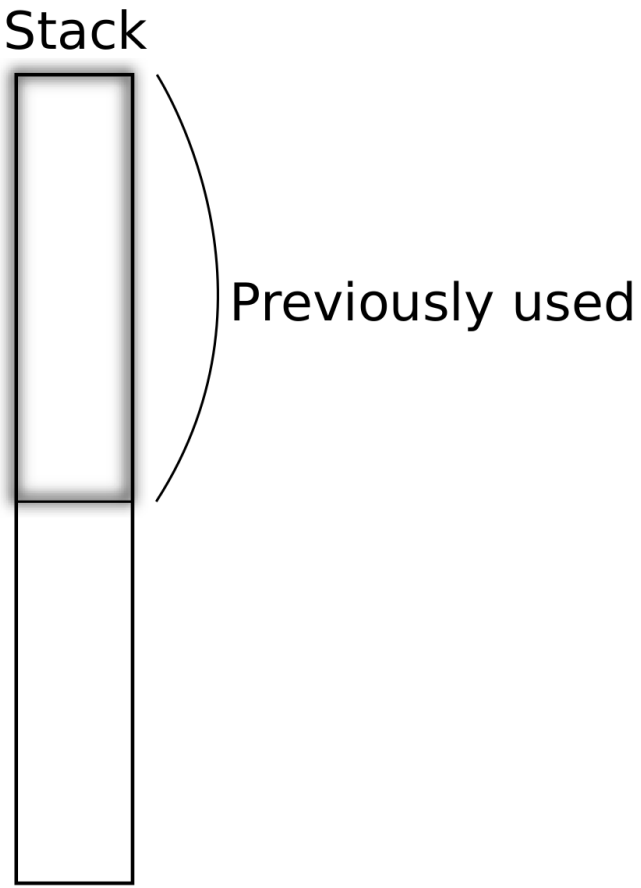


- An experience in a long-term project
- Research fully integrated in a team...
- ...But rather on stand-alone modules
- $\rightarrow$ Production of a reusable microbenchmark tool

Still some time?

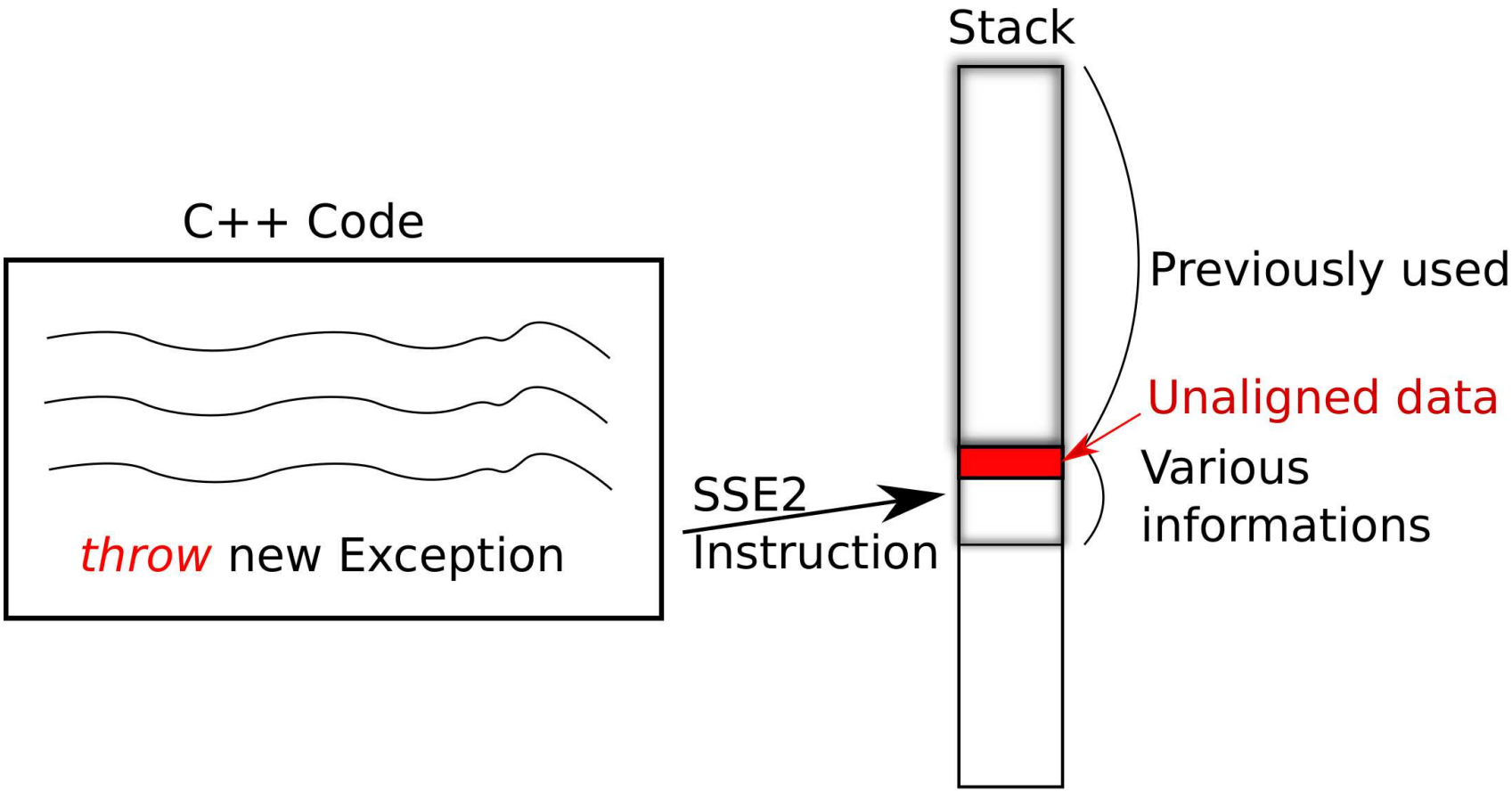
Exception subsystem in C++:





Still some time?

Exception subsystem in C++:



Still some time?

Exception subsystem in C++: