

Shared Memory Paradigm

Methods for high performance computing

Nicolas Roy

Février 2018

SPHYM126

University of Namur

$$\mathcal{F} = \mathcal{C}_{\text{Cores}} * \mathcal{A}_{\text{AVX}} * \mathcal{V}_{\text{vector length}} * \mathcal{H}_{\text{Hyperthreading}} * \mathcal{F}_{\text{Freq}}$$

Around 0.5 teraflops for a high-end laptop using stock overclock.

Different parallel paradigms

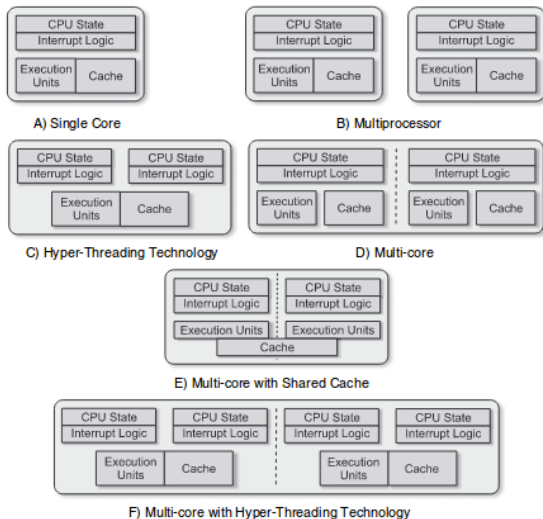
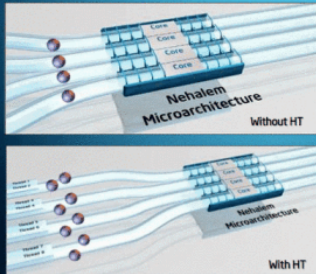


Figure 1.4 Simple Comparison of Single-core, Multi-processor, and Multi-Core Architectures

Hyperthreading - Corporate version

Shut-up, it's magic ! *Intel*

Intel® Hyper-Threading Technology For Better Multitasking³



- Helps you do more and wait less
- Like having a freeway with multiple car lanes instead of just one
- With multiple lanes cars more cars can move more quickly

2 thread engines per core, enabling 8-way processing in 4-core systems. It means that a quad-core processor could run up to eight threads simultaneously

Hyperthreading - Honest version

Amazing (we cannot see the difference with real cores) for :

- Bad (for the meme) or heterogenous code
- Async, Async IO (Network operations, File access)
- Compiling your code
- Playing games (linked to first item)
- Battery, cost-per-unit
- Posting on messenger while your data gets Zucked

Ok (like sad 20% improvement) for :

- Pure number crunching

Don't think there are downsides.

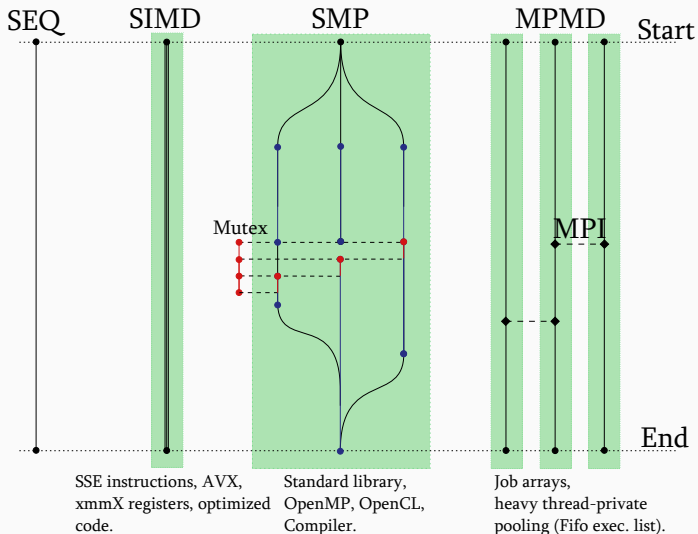
Why should I care about SMP parallel programming?

Embarrassingly parallel execution is king, unless

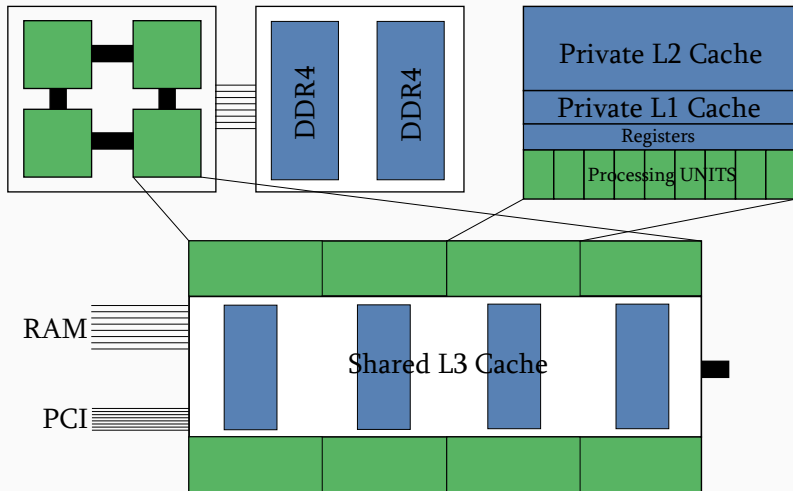
- Your task can be parallelized but is still heavily interlinked
- The algorithm is heavy and parallel (main sequence is long)
- You load a bunch of (common) data
- The main sequence is so small that process overhead is hard on you

TL ;DR : You need *Shared* memory.

Different parallel paradigms

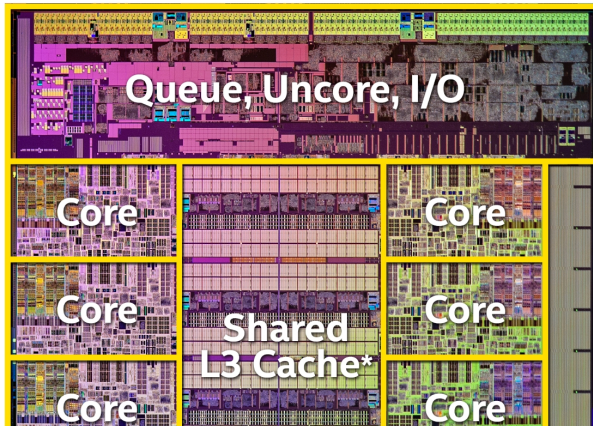


CPU architecture



Shut-up, it's magic ! *Intel*

New 8-Core Intel® Core™ i7 Processor Extreme Edition



Code : Manipulations de base (C++)

Merci de votre attention !