How to Write Fast Code
18-645, spring 2008
4th Lecture, Jan. 28th

Instructor: Markus Püschel
TAs: Srinivas Chellappa (Vas) and Frédéric de Mesmay (Fred)
Today

- Architecture
- Microarchitecture (numerical software point of view)
- First thoughts on fast code
Definitions

- **Architecture**: (also instruction set architecture: ISA) The parts of a processor design that one needs to understand to write assembly code.

- **Examples**: instruction set specification, registers.

- **Counterexamples**: cache sizes and core frequency.

- Example ISAs (Intel): x86, ia, ipf
Intel Architectures (Focus Floating Point)

Architectures
- X86-16
- X86-32
- MMX
- SSE
- SSE2
- SSE3
- X86-64 / em64t
- SSE4

Processors
- 8086
- 286
- 386
- 486
- Pentium
- Pentium MMX
- Pentium III
- Pentium 4
- Pentium 4E
- Pentium 4F
- Core 2 Duo

ia: often redefined as latest Intel architecture
ISA SIMD (Single Instruction Multiple Data)
Vector Extensions

■ What is it?
- Extension of the ISA. Data types and instructions for the parallel computation on short (length 2-8) vectors of integers or floats.
  - Names: MMX, SSE, SSE2, …

■ Why do they exist?
- Useful: Many application (e.g., multimedia) have the necessary fine-grain parallelism. Then, large potential speedup (by a factor close to vector length).
- Doable: Chip designers have enough transistors to play with.

■ We will have an extra lecture on vector instructions
- What are the problems?
- How to use them efficiently.
Definitions

- **Microarchitecture**: Implementation of the architecture.

- Includes caches, cache structure, ....

- **Examples**
  - Intel processors ([Wikipedia](https://en.wikipedia.org/wiki/Intel_processor))
  - Intel *microarchitectures*
Microarchitecture: The View of the Computer Architect

we take the software developers view … (blackboard)

Source: “The Microarchitecture of the Pentium 4 Processor, ”
Intel Technology Journal Q1 2001
Core 2 Duo

Detailed information about Core 2 Duo
Execution Units

Latency/throughput (double)
FP Add: 3, 1
FP Mult: 5, 1

Theoretical peak performance (3 GHz, 1 core, no SIMD, double precision): 6 Gflop/s
SIMD, 1 core, double precision: 12 Gflop/s
SIMD, 1 core single precision: 24 Gflop/s
2 or 4 cores: multiply by 2 or 4
Requires: computation has 50% adds and 50% mults
Evolution of FP Peak Performance (Intel)

Floating point peak performance [Mflop/s]
CPU frequency [MHz]

work required

free speedup

1993 1995 1997 1999 2001 2003 2005 2007
Year

data: www.sandpile.org
Cache

- Blackboard
- Example parameters taken from L1 D cache, Core 2 Duo
  - Size 32 KB
  - 8-way
  - 64B = 8 doubles cacheline
Remarks

- **Microarchitecture optimizations**
  - Partially frees programmer from optimization
  - Targets most common code patterns and most important benchmarks
  - Are often not or not well documented

- **Performance/runtime of code is hard to understand/predict**
  - Very complex microarchitecture, not everything is documented
  - Actual execution not exactly known
  - Compiler optimizations
Microarchitectural Parameters Most Important for Programmers

- Memory hierarchy:
  - How many caches
  - Cache sizes and structure
  - Number of registers

- Processor
  - Frequency
  - Execution units
  - Latency and throughput of fadd, fmult, etc.
  - Floating point peak performance

- How to get it?
  - Digging through manuals, vendor websites; e.g., for Core 2
  - Measuring. E.g., cpuid (Windows only), X-Ray
Optimization of Numerical Software: First Thoughts

- It’s all about keeping the floating point units busy

- Need to optimize for memory hierarchy
  - For several levels
  - Goal: increase reuse of data
  - Often requires algorithm modifications or proper algorithm choice
  - Divide-and-conquer algorithms are in principal good
    (recursive is better than iterative)

- Need for fine-grain instruction parallelism

- Use a good compiler and make sure you understand flags