Introduction to HPC
content and definitions

Jan Thorbecke, Section of Applied Geophysics

Motivation and Goal

- Get familiar with hardware building blocks, how they operate, and how to make use of them in writing efficient programs.
- Understand the function of an Operating System (Linux).
- Participants should be able to write, compile and run numerical code efficiently on standard hardware.
- Start to think how to solve problems in Parallel (try to avoid thinking in sequential loops).
- During the course keep in mind a numerical problem which you would like to implement/improve.

Learning Outcomes

- Understand the architecture of modern CPU's and how this architecture influences the way programs should be written.
- Optimize all aspects in the processes of programming: from compilation, starting and running program by OS, executing (parallel) instructions by CPU, to writing output to disk.
- Write numerical software, that exploits the memory hierarchy of a CPU, to obtain a code with close to optimal performance.
- Analyze an existing program for OpenMP and MPI parallelization possibilities.
- Evaluate the possibilities of accelerators to speed up computational work.

Organization

Dates: 8, 17, 22, 24 and 31 May 2018

Time: 13:30 17:00

Points: 3

Examination: attendance, exercises and discussions during the course. For MSc students there is a multiple-choice exam with discussion afterwards.

Schedule and rooms

Civil Engineering & Geosciences, Building 23:
Stevinweg 1

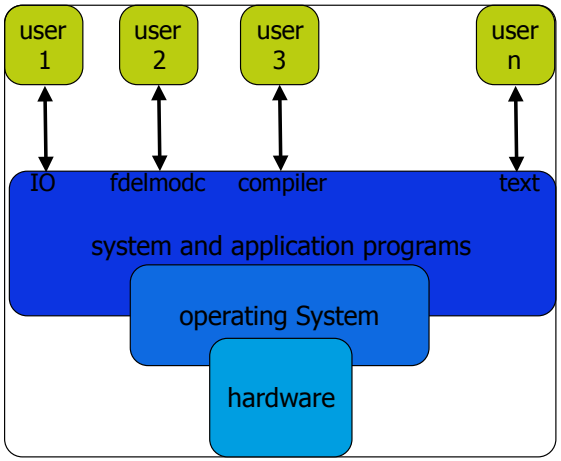
Electrical Engineering, Mathematics and Computer Science, Building 28
Van Mourik Broekmanweg 6

- 8 May room 2.110 (23)
- 17 May room E (23)
- 22 May room 1.98 (23)
- 24 May room G (28) <=
- 31 May room 1.96 (23)

What is HPC

- **High-performance computing** (**HPC**) uses supercomputers and computer clusters to solve advanced computation problems. Today, computer systems in the teraflops-region are counted as HPC-computers. (Wikipedia)
- A list of the most powerful high performance computers can be found on the TOP500 list. The TOP500 list ranks the world's 500 fastest high performance computers as measured by the HPL benchmark.
- For this course: Scientific programs written for optimal use of computational hardware.

Four Components Computer System



Computer System Structure

- Computer system can be divided into four components
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – to solve the computing problems of users
    - Word processors, compilers, web browsers, database systems, games
  - Users
    - People, machines, other computers

Contents

- Introduction (1)
- Hardware (1,2)
- OS (2)
- Programming (2/3)
- Optimization (3)
- Multi-processor hardware(3/4)
- Writing Parallel programs (4/5)
- Parallel programming + wrap-up and discussion (5)

Part 1

- Components of a computer system
  - addressing
  - memory hardware
- Memory Hierarchy
  - types of cache mappings
  - cache levels
  - virtual memory, TLB
- Processors
  - out of order
  - pipelining
  - branches
  - multi-core cache-coherency
  - vector instructions: SSE / AVX
- Modern processors
  - Haswell, Power, ARM, GPU
- Future trends in hardware

Part 2

- Operating System
- IO management
- Processes
- Virtual memory management
- File systems
  - storage, RAID
- Linux, Unix, OSX, XP

Part 3

- User environment
  - login, shells
- Compiling and linking
  - compile stages
  - Makefiles
- Number representations
  - IEEE 754 standard
- Programming languages
  - C, Fortran
  - profiling and debugging
  - numerical libraries

Part 4

- Computer Performance
  - Amdahl's law
  - standard benchmarks Spec, Linpack
- Optimization
  - cache blocking
  - trics on loops
  - many examples

Part 5

- Programming for parallel systems
- Multi-processor hardware
  - Classification
  - HPC hardware
- Programming for parallel systems
  - OpenMP
  - MPI
  - examples
- Finish the last parts of the course
- Followed by questions, discussion and evaluation of exercises

The Exercises

- Exercises are set up to get familiar with concepts and ideas. If you already have enough experience with some parts in the exercise you can skip those parts.
- Exercises are simple and straightforward, no programming skills are needed, and consists of compilation and running of small programs. Every exercise comes with a step by step procedure to follow.
- The exercises are hopefully fun to do and are not obligatory to pass the course, but highly recommended.

The slides

- There are quite a few of them...
- Not all of them will be discussed
- Sometimes techniques are explained in two different ways, one will be used in the lectures, the other one can be used to understand the concepts better.
- At the moment there are no lecture notes and the detailed slides try to compensate that.

Definitions and Terms

- bit = binary number with states 0 or 1
- byte = 8 bits
- word = 4 or 8 bytes (the number of bits used in a register)
- Kbit = 1000 bits, used for communication speeds
- KiB = 1024 (2^{10}) Bytes, used for memory and storage
- MiB = 1048576 (2^{20}) Bytes
- flops = floating point operations per second
- CPU = Central Processing Unit
- FPU = Floating Point Unit
- ALU = Arithmetic Logic Unit
- AGU = Address Generation Unit

- **http://physics.nist.gov/cuu/Units/binary.html**

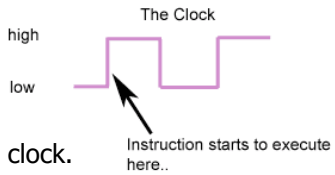
Definitions

Latency: Latency is a time delay between the moment something is initiated, and the moment one of its effects begins or becomes detectable. The word derives from the fact that during the period of latency the effects of an action are latent, meaning "potential" or "not yet observed".

Bandwidth: a data transmission rate; the maximum amount of information (bits/second) that can be transmitted along a channel.

flops: number of floating point operations (addition multiplication) per second.

Clock



Most modern CPUs are driven by a clock.

The CPU consists internally of logic and memory (flip flops).

When the clock signal arrives, the flip flops take their new value and the logic then requires a period of time to decode the new values.

Then the next clock pulse arrives and the flip flops again take their new values, and so on.

Feedback

Positive and negative feedback about the course is highly appreciated.

You can always send me an e-mail or come to my desk in room 3.03 (CITG Building on Tuesdays and Thursdays only).

Book References

Computer Architecture , A Quantitative Approach
Fifth Edition, John L. Hennessy, David A. Patterson 2011 Morgan Kaufmann

The C Programming Language (2nd edition)
Kernighan & Ritchie, 1988 Prentice Hall

Modern Fortran Explained
(Numerical Mathematics and Scientific Computation)
Michael Metcalf, John Reid , Malcolm Cohen, 2011 Oxford University Press

Fortran 90 Programming
T.M.R. Ellis, Ivor R. Phillips, Thomas M. Lahey, 1994 Addison Wesley

Parallel Programming With MPI
Peter Pacheco, 1996 Morgan Kaufmann

References on internet

http://arstechnica.com/index.ars

http://www.tomshardware.co.uk/

http://insidehpc.com/

http://www.gpgpu.org/

http://www.theregister.co.uk/

wikipedia.org