An introduction to OpenMP

Paolo Burgio
paolo.burgio@unimore.it

A history of OpenMP

- 1997
  - OpenMP for Fortran 1.0
- 1998
  - OpenMP for C/C++ 1.0
- 2000
  - OpenMP for Fortran 2.0

- 2002
  - OpenMP for C/C++ 2.5

Regular, loop-based parallelism

- 2008
  - OpenMP 3.0
- 2011
  - OpenMP 3.1

Irregular, parallelism ➔ tasking

- 2014
  - OpenMP 4.5

Heterogeneous parallelism, *à la* GP-GPU

What is OpenMP?

eng.wikipedia.org

OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior

Application Programming Interface

eng.wikipedia.org

OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior

- Eases programmers' life
- Can be specific for specific domain
  - Web, Databases…for parallel programming
- Examples
  - POSIX Threads
  - CUDA
  - OpenCL
  - …

A mix of …

eng.wikipedia.org

OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior

- Many ways to do the same things
  - Linux-like philosohpy
  - What's the difference?

Why OpenMP?

- What makes it better e.g., than PThreads?
  - Pragma-based interface
  - Transparent threading and memory management
  - We will this soon…
- What's missing?
  - Poor control on threads (no scheduling)
  - "Team of threads"
  - Automatic memory management
- What's left?
  - Programmer is in charge of synchronization
  - Programmer is in charge of memory consistency
  - As opposite to..caches

"it's elegant and appealing"

Cross-platform, cross-language

eng.wikipedia.org

OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior

Find the difference

- Between…
  - A standard
  - An implementation
- A standard usually gives guidelines on
  - Behavior
  - (Common) Interface
  - De facto vs de iure
- A implementation is
  - Platform(s)-specific
  - Lanugage(s)-specific
  - Has specific performance/Quality-of-Service QoS

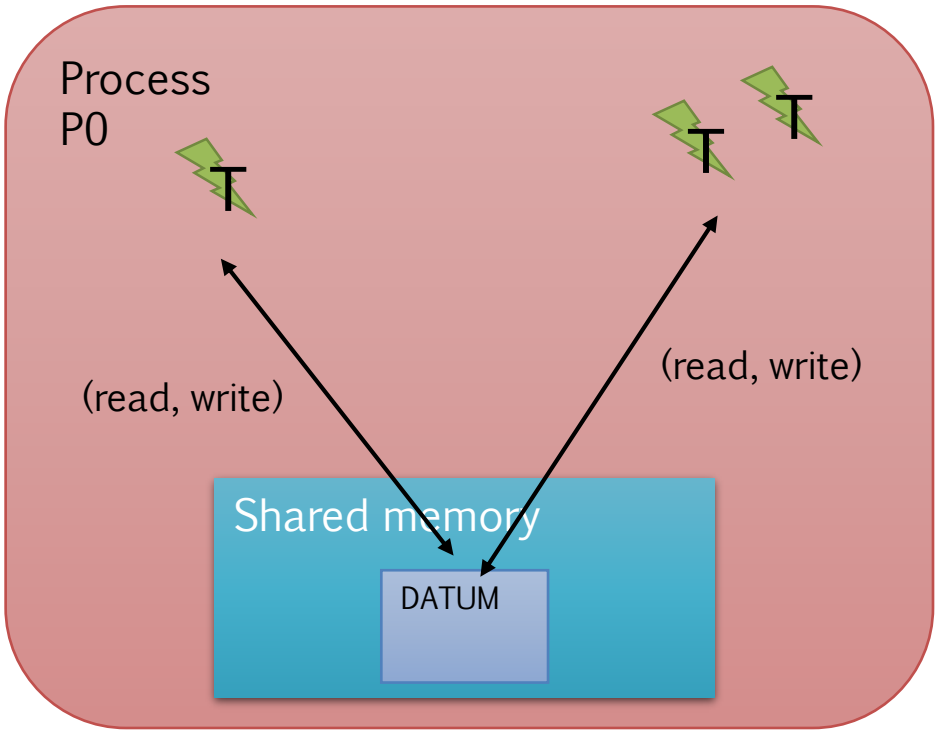
Shared memory

eng.wikipedia.org

OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior

Shared memory

- Coherence problem
  - Memory consistency issue
  - Data races
- Can share data ptrs
  - Ease-to-use
- Several paradigm ("flavours")
  - Symmetric Multiprocessing
  - Distributed Shared memory
  - Partitioned Global Access Space



Multi-processing

eng.wikipedia.org

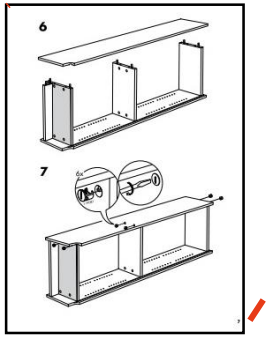
OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior

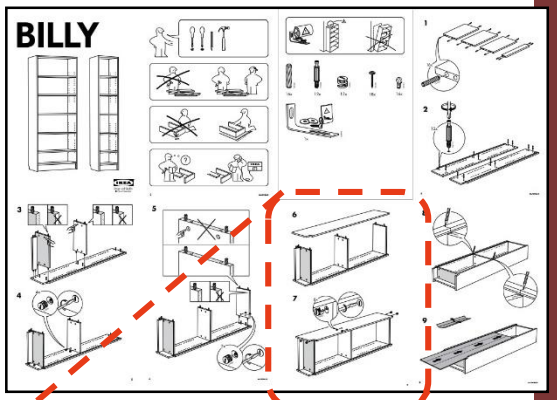
What is…

- ..a core?
- …a program?
- …a process?
- …a thread?
- ..a task?

What is…

- ..a core?
  - An electronic circuit to execute instruction (=> programs)
- …a program?
  - The implementation of an algorithm
- …a process?
  - A program that is executing
- …a thread?
  - A unit of execution (of a process)
- ..a task?
  - A unit of work (of a program)



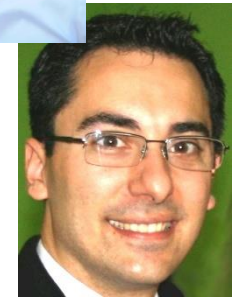


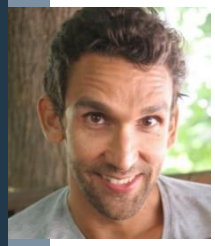
What is a task?

P-SOCRATES

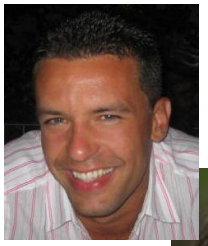
OpenMP task

Operating System task

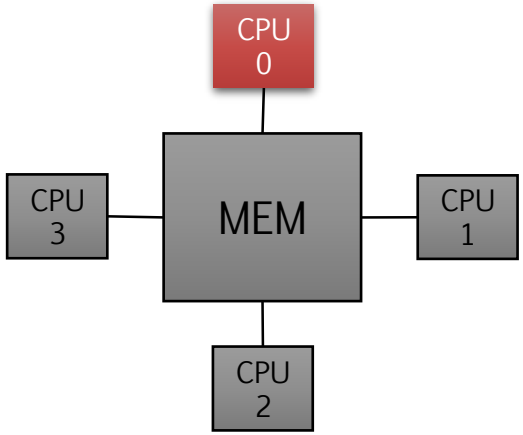




Real-time task

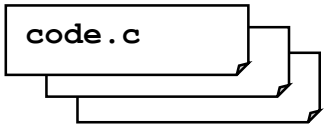


What is…



› ..a core?

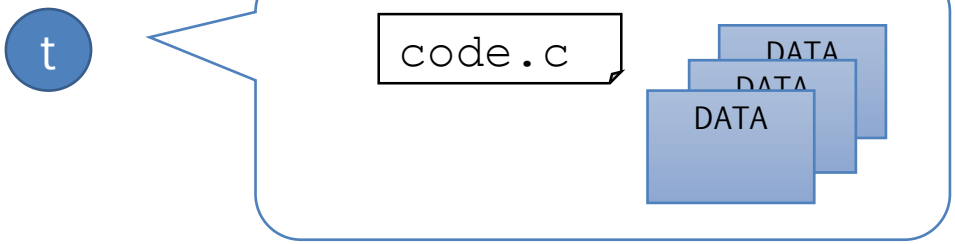
› …a program?



› …a process?

› …a thread?

› ..a task?



Outline

- Expressing parallelism
  - Understanding parallel threads
- Memory Data management
  - Data clauses
- Synchronization
  - Barriers, locks, critical sections
- Work partitioning
  - Loops, sections, single work, tasks…
- Execution devices
  - Target

References

- "Calcolo parallelo" website
  - http://hipert.unimore.it/people/paolob/pub/Calcolo_Parallelo/
- My contacts
  - paolo.burgio@unimore.it
  - http://hipert.mat.unimore.it/people/paolob/
- Useful links
  - http://www.openmp.org
- A "small blog"
  - http://www.google.com