HIGH PERFORMANCE COMPUTING

Dr Noor Mahamamd Sk

Center for High Performance Reconfigurable Computing

Hprcse

Logistics

- Course is @ A slot
- Class Timings
  - MONDAY – 0800 Hrs
  - TUESDAY – 0900 Hrs
  - THURSDAY – 1000 Hrs
- Attendance as per Institute Norms

Evaluation Policy

- Quiz 1 – 20 Marks
- Quiz 2 – 20 Marks
- End Semester – 60 Marks

Motivation

- A few examples of large scale problems for motivation. Currently that means Tera-scale or Peta-scale
- Proc. speed measured in Gigahertz (hertz = cycles per sec.)
- One operation may take several cycles.
- So a 1 GHz processor does > 10^8 operations per second.
- Exascale is a billion times more. (Also flops and bytes.)

Kilo	thousand (10^3)	2^{10}
Mega	million (10^6)	2^{20}
Giga	billion (10^9)	2^{30}
Tera	trillion (10^{12})	2^{40}
Peta	(10^{15})	2^{50}
Exa	(10^{18})	2^{60}

How long does it take to solve a linear system?

- Solving an n×n linear system Ax = b takes≈ $(1/3)n^3$ flops (using Gaussian elimination for a dense matrix)
- On a 100 MFlops system:

n	**flops**	**time**
10	3.3×10^2	0.0000033 seconds
100	3.3×10^5	0.0033 seconds
1000	3.3×10^8	3.33 seconds
10000	3.3×10^{11}	3333 seconds = 55.5 minutes
100000	3.3×10^{14}	3333333 seconds = 926 hours
1000000	3.3×10^{17}	926,000 hours = 105 years

- This assumes data transfer not a problem – which it is. Often it's the bottleneck.
- A 10^6x10^6 matrix has 10^{12} elements – this is 8 terabytes!

Parallel Computing

- **High performance available today only through parallelism**
- Provides alternatives when individual processor speeds reach limits imposed by fundamental physical laws
- Leverages inexpensive commodity parts
- Provides cost-effective means (sometimes only means) of meeting enormous demands of large-scale simulations
- **However**:
- Relatively immature computing environment, lack of available software
- Rapid pace of change (code soon obsolete)
- Complexity of parallel programming (algorithm and software development much harder; lack of standardization)
- Physical constraints (power, cooling) will limit growth

Some Terminology

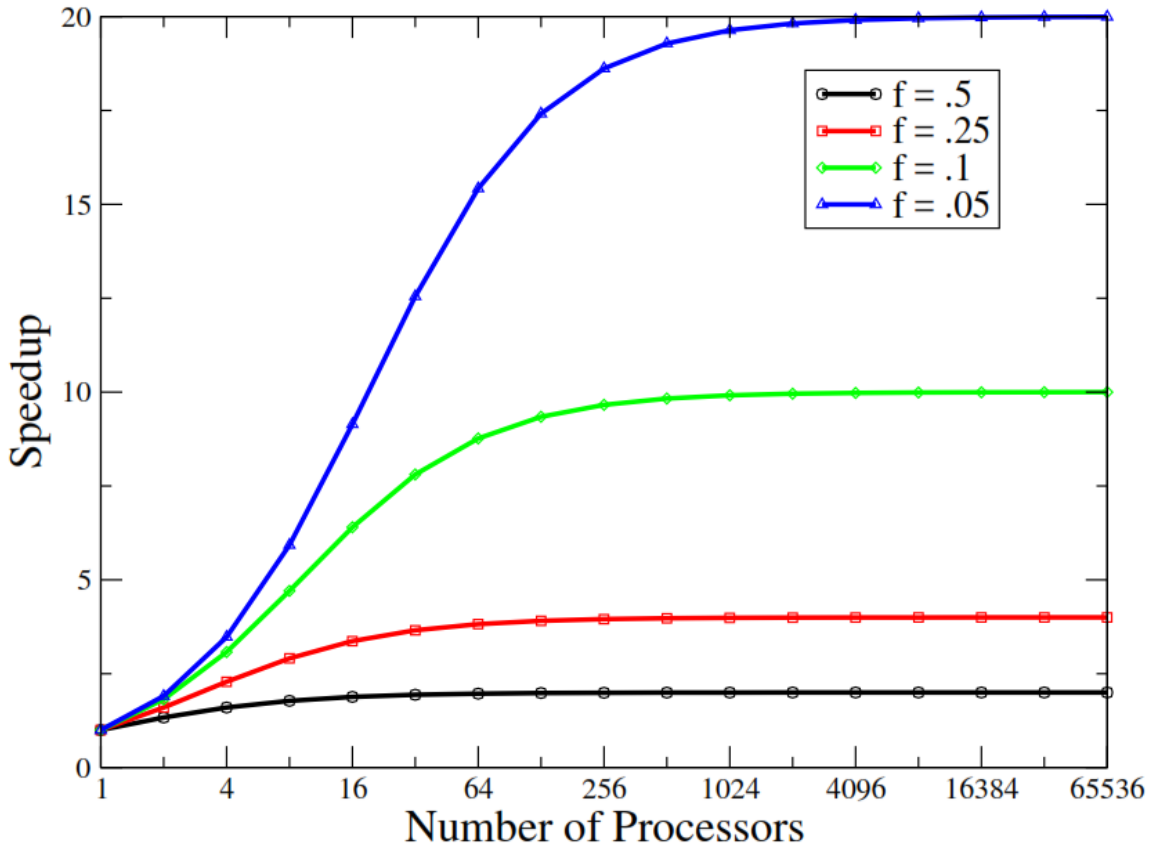
- Core a single computing unit with its own independent control
- Multicore a processor having several cores that can access the same memory concurrently
- A computation is decomposed into several parts called tasks that can be computed in parallel
- Finding enough parallelism is (one of the) crucial steps for high performance (Amdahl's law).

Amdahl's Law

- Used to predict maximum speedup using multiple processors
- Let f = fraction of work performed sequentially
- (1−f) = fraction of work that is parallelizable
- P = number of processors
- On 1 cpu: T1 = f + (1−f) = 1.
- On P processors: Tp = f + (1−f)/p
- Speedup T1/TP = {1/(f+(1−f)/P) } < 1/f
- Speedup limited by sequential part

Amdahl's Law

- Speedup as a function of sequential fraction

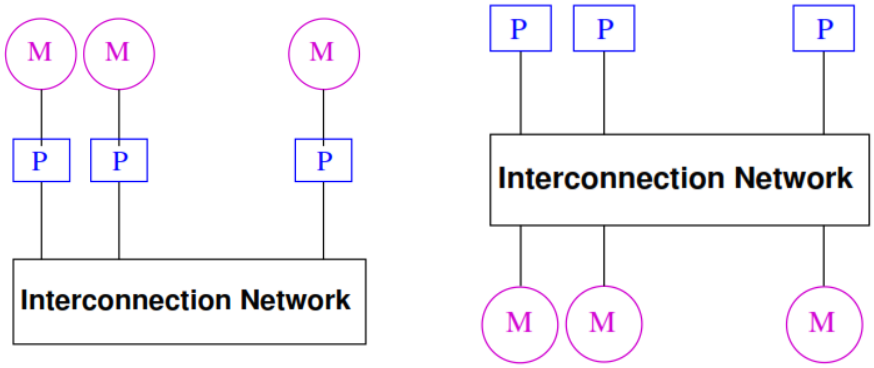


Terminology

- The size of the tasks is called the granularity; tasks may have different sizes
- The assignment of tasks in a given order is scheduling.
- Dependencies between tasks creates constraints for scheduling.
- Mapping: assignment of tasks to physical computation units
- If all tasks take the same amount of time work is load balanced. (otherwise some cpus are waiting for others finish - this is idle time)
- Parallel programs need synchronization to execute correctly, obey constraints, etc.
- Barrier operations are a form of coordination where all processes or threads wait until all others have also reached that point.
- Some of this done by the OS and some by the programmer. This is what makes parallel programming hard.

Terminology

- One way to classify machines distinguishes between shared memory global memory can be accessed by all processors or cores.
- Information exchanged between threads using shared variables written by one thread and read by another.
- Need to coordinate access to shared variables.
- Distributed memory private memory for each processor, only accessible this processor, so no synchronization for memory accesses needed.
- Information exchanged by sending data from one processor to another via an interconnection network using explicit communication operations.



Terminology

- We will also be concerned with Parallel execution time
- Time for the computation plus time for data exchange (communication), synchronization, other overhead (redundant computation), cost of starting threads/process)
- Quantitative measures of parallel execution time are
- Speedup = parallel execution time / sequential execution time on one processor
- Efficiency = speedup / number of processors

Principles of Parallel Computing

- Finding enough parallelism (Amdahl's law)
- Granularity need large enough amount of work to hide the overhead
- Locality large memories are slow fast memories are small.
- Load balance
- Coordination and synchronization
- Performance modeling
- All these things make parallel programming harder than sequential programming.

Flynn's Taxonomy

- Early classification of parallel computing architectures given by M. Flynn (1972) using number of instruction streams and data streams. Still used.
- **Single Instruction Single Data (SISD)** conventional sequential computer with one processor, single program and data storage.
- **Multiple Instruction Single Data (MISD)** used for fault tolerance (Space Shuttle) - from Wikipedia
- **Single Instruction Multiple Data (SIMD)** each processing element uses same instruction applied synchronously in parallel to different data elements (Connection Machine, GPUs).
- **Multiple Instruction Multiple Data (MIMD)** each processing element loads separate instruction and separate data elements; processors work asynchronously.
- Since 2006 top ten supercomputers of this type
- Update: **Single Program Multiple Data (SPMD)** autonomous processors executing same program but not in lockstep. Most common style of programming.

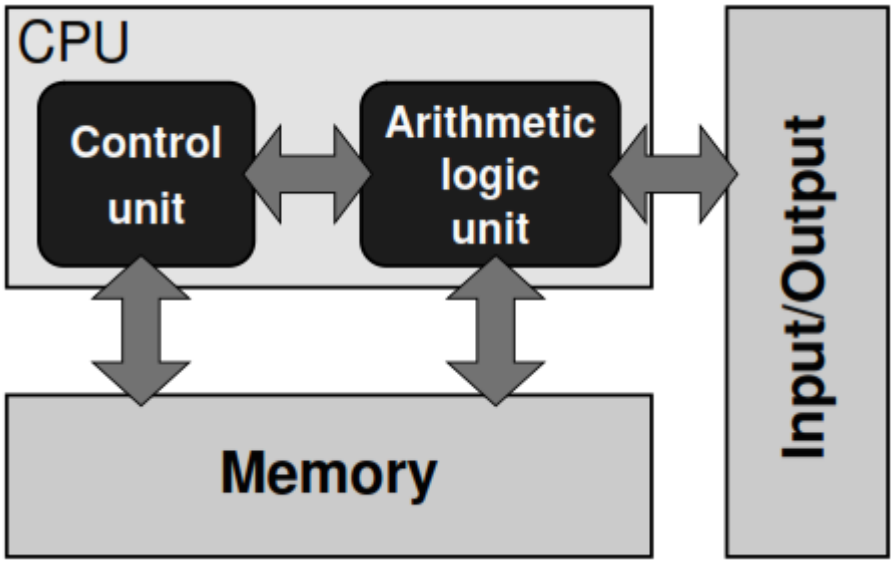
Algorithm Taxonomy - the 7 Dwarfs

- dwarf - coined by Phil Colella, an algorithmic kernel that captures a pattern of computation and communication.
- Specify at a high level of abstraction so that holds across a broad range of applications.
- Dense Linear Algebra
- Sparse Linear Algebra
- FFT (now Spectral Methods)
- Particle Methods (now N-Body Methods)
- Structured Grids (including locally structured, e.g. AMR)
- Unstructured Grids
- Monte Carlo (now MapReduce)
- **6 more subsequently added**:
- Combinatorial Logic, Graph Traversal, Dynamic Programming, Backtracking and Branch and Bound, Graphical Models, and Finite State Machines.

MODERN PROCESSORS

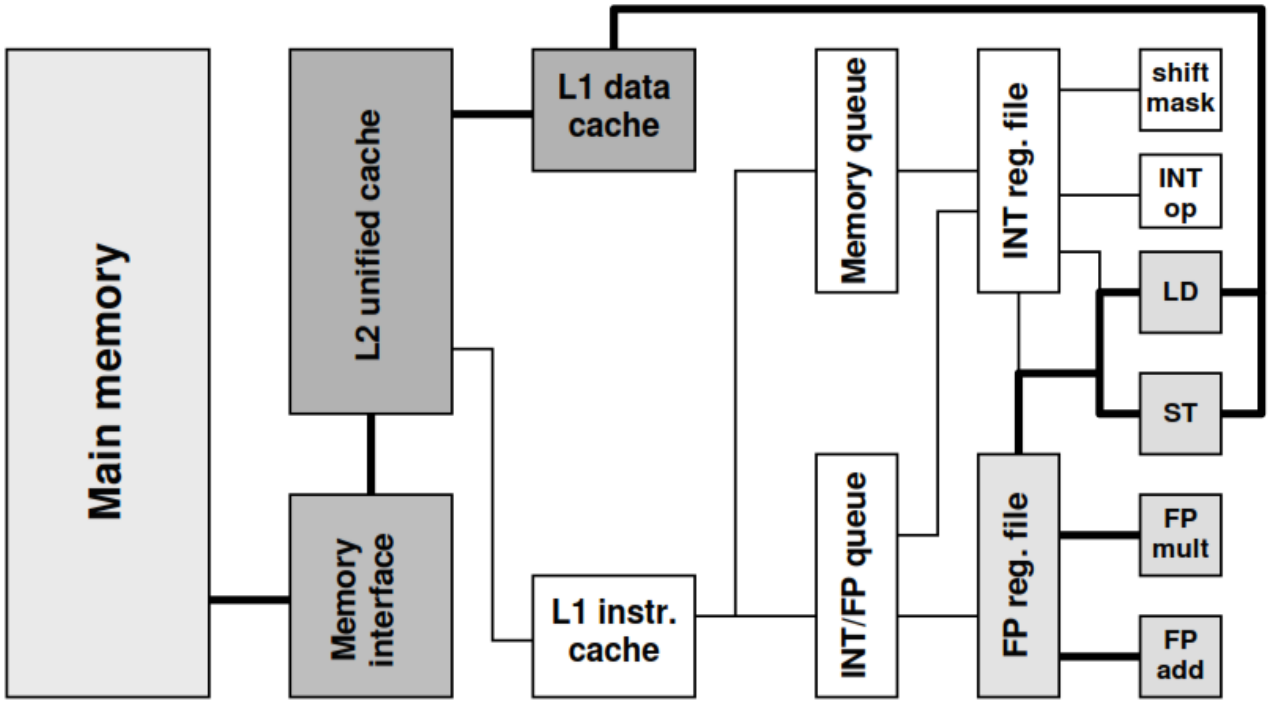
Stored-program computer architecture

- Stored-program computer architectural concept.
- The “program,” which feeds the control unit, is stored in memory together with any data the arithmetic unit requires.



General-purpose cache-based microprocessor architecture

- Simplified block diagram of a typical cache-based microprocessor (one core).
- Other cores on the same chip or package (socket) can share resources like caches or the memory interface.
- The functional blocks and data paths most relevant to performance issues in scientific computing are highlighted.



Performance metrics and benchmarks

- All the components of a CPU core can operate at some maximum speed called peak performance.
- The performance at which the FP units generate results for multiply and add operations is measured in floating-point operations per second (Flops/sec)
- Microprocessors are designed to deliver at most two or four double-precision floating-point results per clock cycle.
- With typical clock frequencies between 2 and 3GHz, this leads to a peak arithmetic performance between 4 and 12GFlops/sec per core.

FLOPS = Sockets x (Cores/Socket) x (Cycles/Second) x (FLOPS/Cycle)

Intel Core i7–970, capable of 4 double-precision or 8 single-precision floating-point operations per cycle.

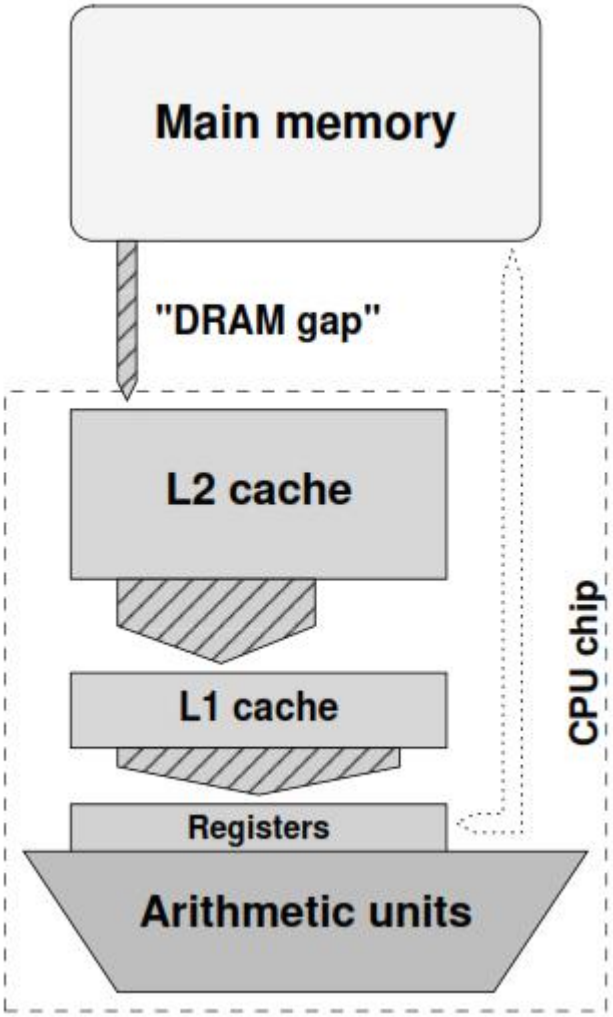
Example

- Intel Core i7–970 has 6 cores. If it is running at 3.46 GHz, the formula would be:
- 1 (socket) * 6 (cores) * 3,460,000,000 (cycles per second) * 8 (single-precision FLOPs per second) = 166,080,000,000 single-precision FLOPs per second = 166 GFLOPS
- or
- 83,040,000,000 double-precision FLOPs per second = 88 GFLOPS.

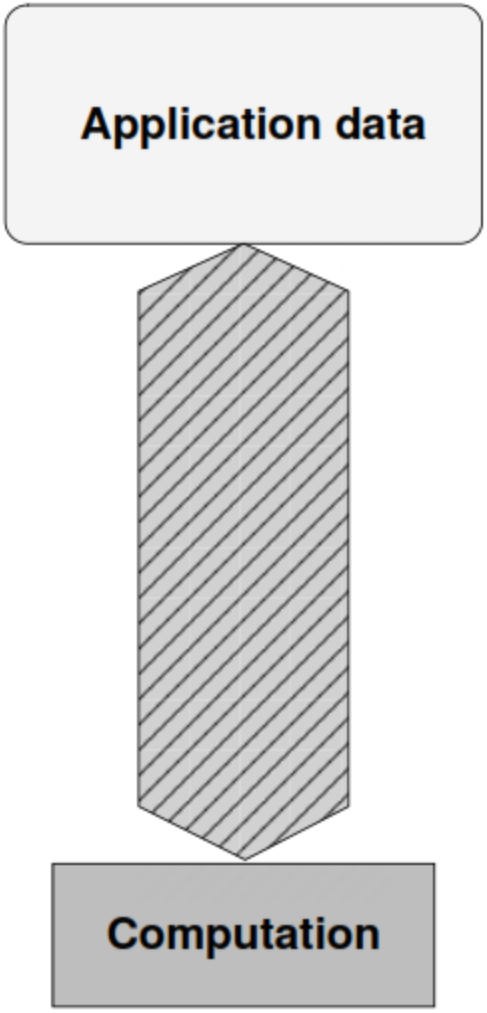
Performance Measurement

- The most sensible time measure in benchmarking is wall clock time, also called elapsed time.

Memory



- The “DRAM gap” denotes the large discrepancy between main memory and cache bandwidths.
- This model must be mapped to the data access requirements of an application.



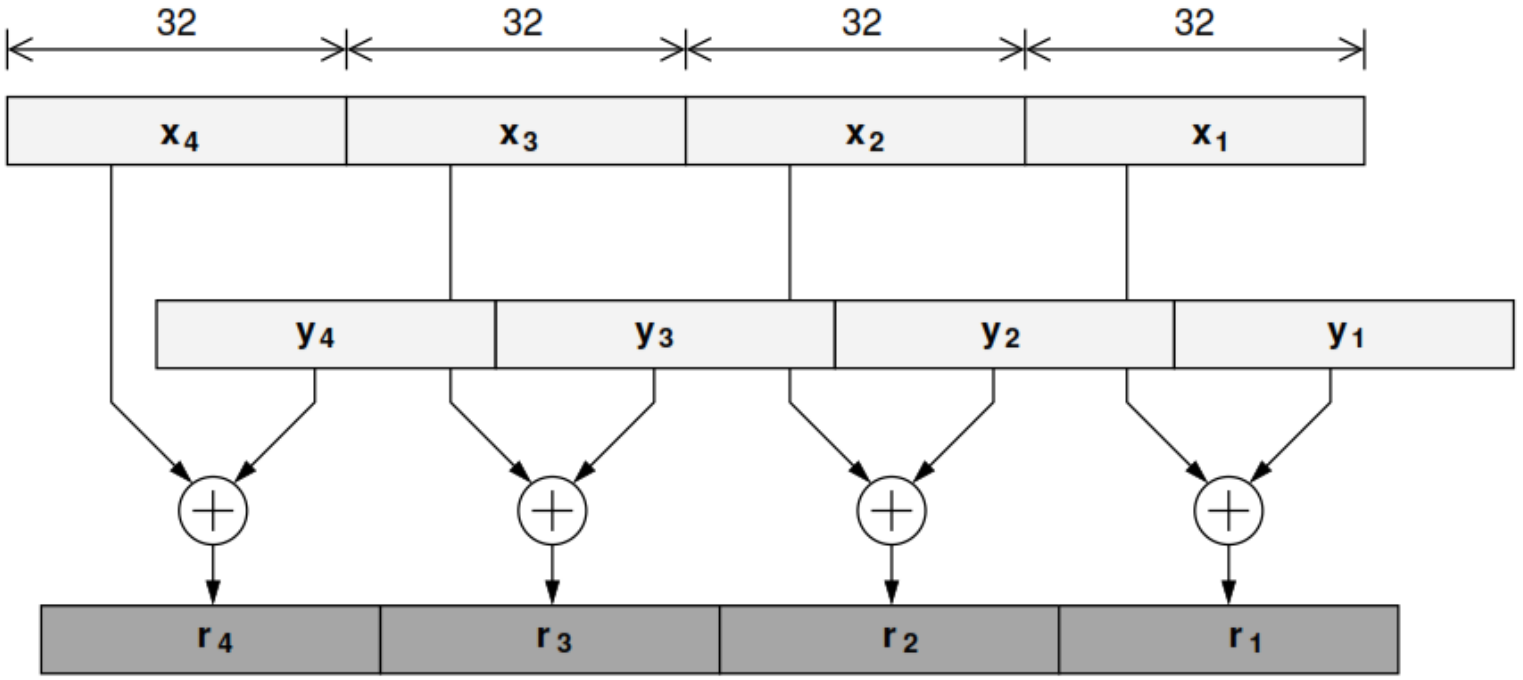
Superscalarity

- If a processor is designed to be capable of executing more than one instruction or, producing more than one “result” per cycle
- Multiple instructions can be fetched and decoded concurrently (3–6 nowadays).
- Address and other integer calculations are performed in multiple integer (add, mult, shift, mask) units (2–6).
- Multiple floating-point pipelines can run in parallel.
- Often there are one or two combined multiply-add pipes that perform a=b+c *d with a throughput of one each.
- Caches are fast enough to sustain more than one load or store operation per cycle
- The number of available execution units for loads and stores reflects that (2–4).

SIMD

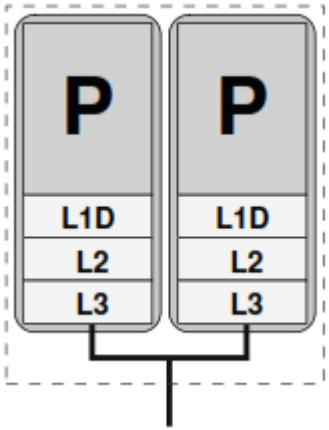
- The SIMD concept became widely known with the first vector supercomputers in the 1970s
- SIMD is the fundamental design principle for the massively parallel *Connection Machines in the 1980s and early 1990s*
- Many recent cache-based processors have instruction set extensions for both integer and floating-point SIMD operations
- They allow the concurrent execution of arithmetic operations on a “wide” register that can hold, e.g., two DP or four SP floating-point words

SIMD Operation

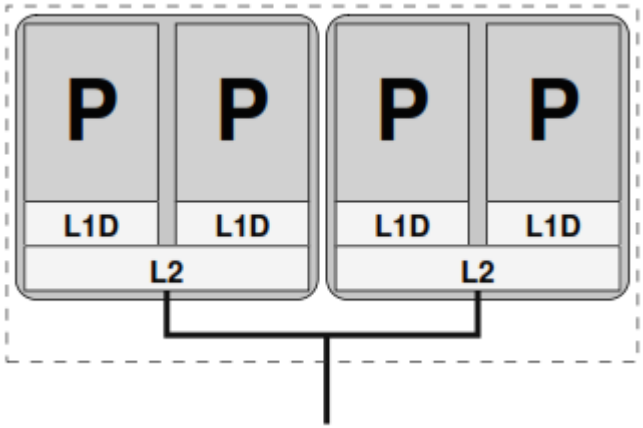


- Two 128-bit registers hold four single-precision floating-point values each.
- A single instruction can initiate four additions at once.

Multicore Processors

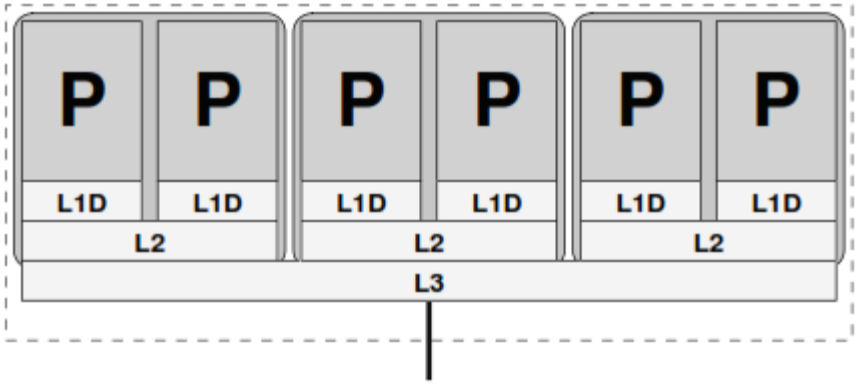


- Dual-core processor chip with separate L1, L2, and L3 caches.
- Each core constitutes its own cache group on all levels.

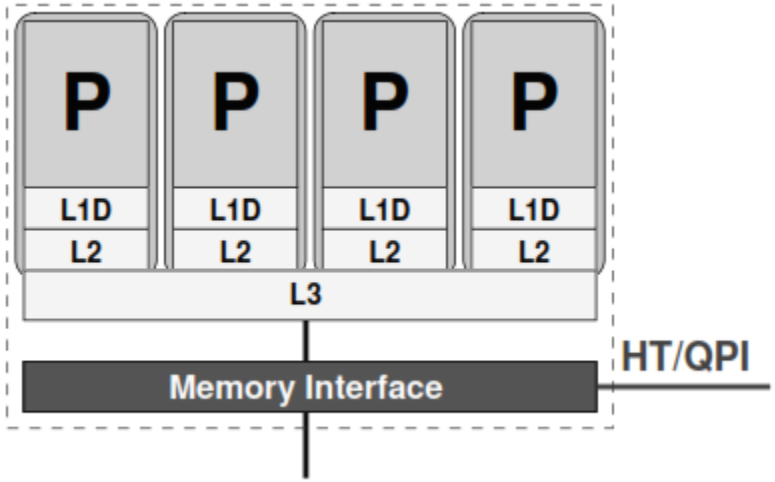


- Quad-core processor chip, consisting of two dual-cores.
- Each dual-core has shared L2 and separate L1 caches.
- There are two dual-core L2 groups.

Multicore Processors

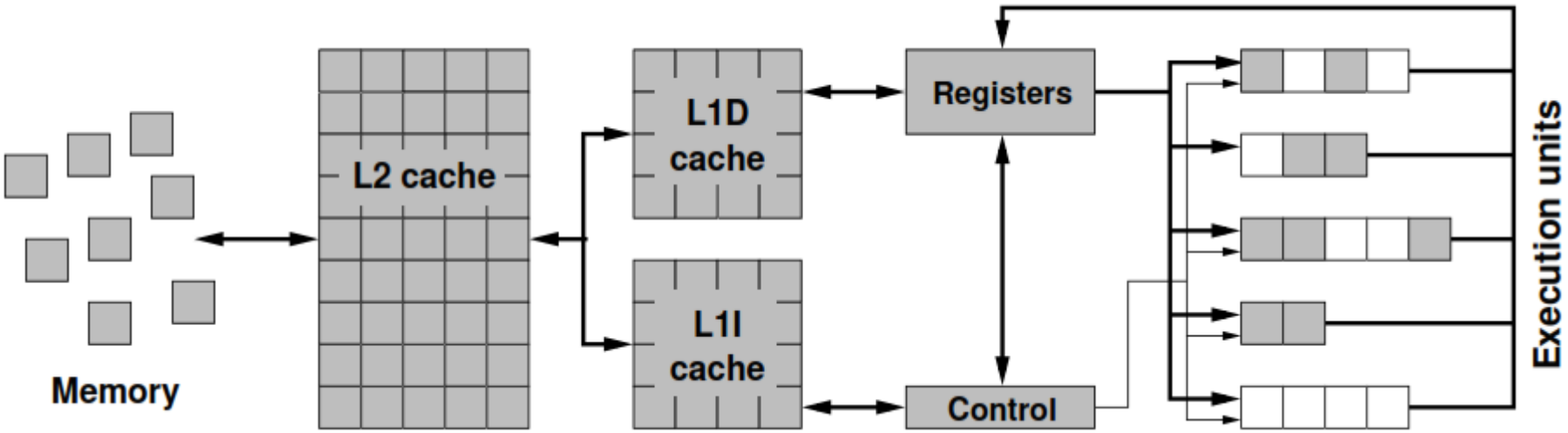


- Hexa-core processor chip with separate L1 caches, shared L2 caches for pairs of cores and a shared L3 cache for all cores.
- L2 groups are dual-cores, and the L3 group is the whole chip.



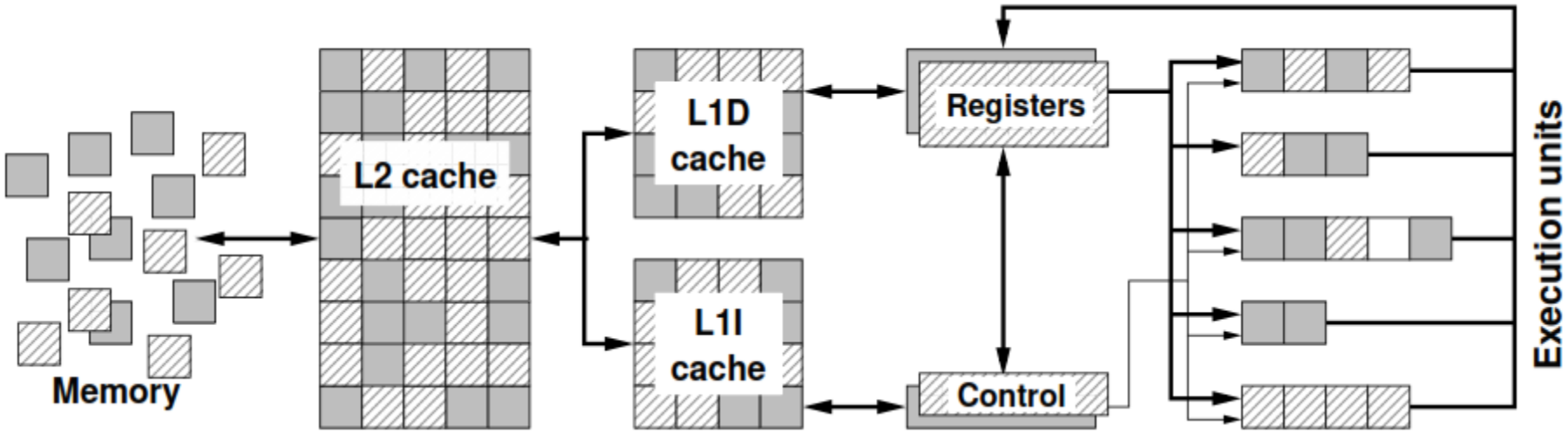
- Quad-core processor chip with separate L1 and L2 and a shared L3 cache.
- There are four single-core L2 groups, and the L3 group is the whole chip.
- A built-in memory interface allows to attach memory and other sockets directly without a chipset.

Multithreaded processors



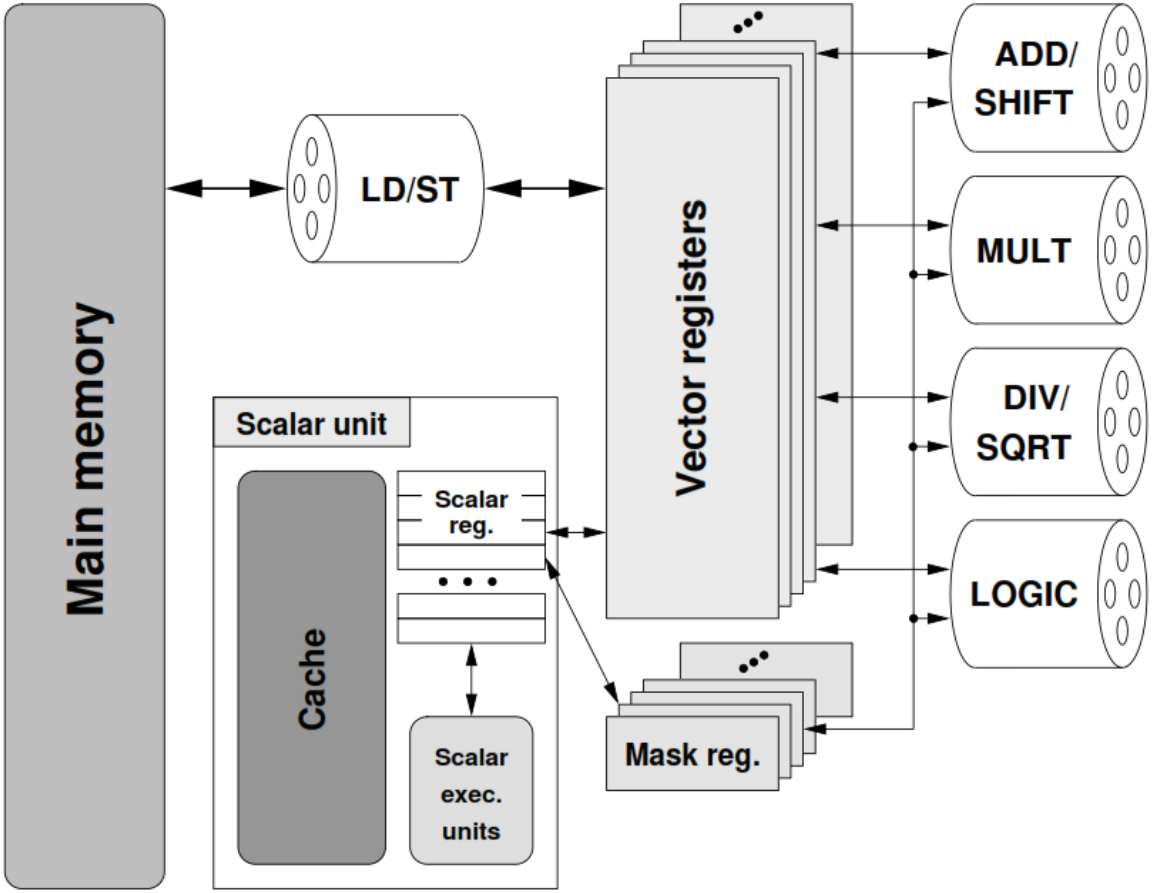
- Simplified diagram of control/data flow in a (multi-)pipelined microprocessor without SMT.
- White boxes in the execution units denote pipeline bubbles (stall cycles).

SMT



- Simplified diagram of control/data flow in a (multi-)pipelined microprocessor with fine-grained two-way SMT.
- Two instruction streams (threads) share resources like caches and pipelines but retain their respective architectural state (registers, control units).

Vector Processors



Reference

- **Georg Hager and Gerhard Wellein**, Introduction to High Performance Computing for Scientists and Engineers, CRC Press, 2011.