Parallel Hardware Architecture (Self-Study)

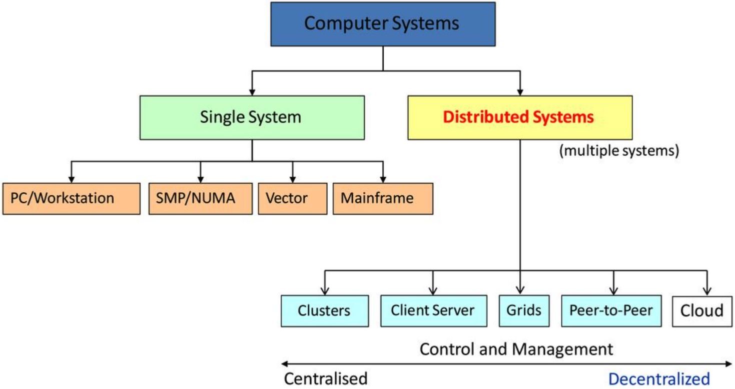
Shuhao Zhang

Nanyang Technological University

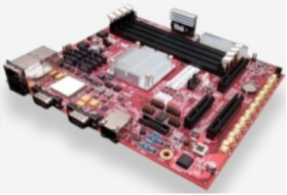
shuhao.zhang@ntu.edu.sg

June 4, 2024

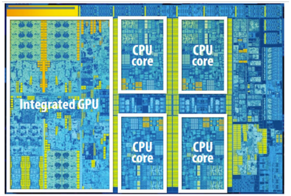
Types of Computer Systems



Parallel Computing and Parallel Machines

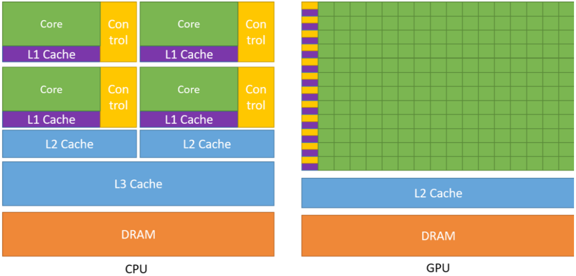


ARM big.LITTLE 64-bit (Cortex-A53 + Cortex-A73) 4-8 cores, >2GHz, 128GB RAM

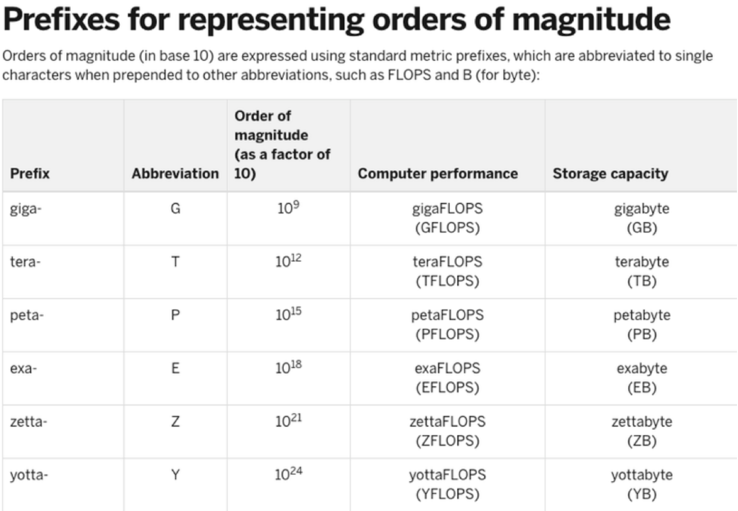


Intel Skylake architecture: Quad-core CPU + multi-core GPU integrated on one chip

GPUs


Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
L2 Cache
L2 Cache
L3 Cache
DRAM
CPU
L2 Cache
DRAM
GPU

2020 Fastest Supercomputer



Prefixes for representing orders of magnitude

Orders of magnitude (in base 10) are expressed using standard metric prefixes, which are abbreviated to single characters when prepended to other abbreviations, such as FLOPS and B (for byte):

Prefix	Abbreviation	Order of magnitude (as a factor of 10)	Computer performance	Storage capacity
giga-	G	10^9	gigaFLOPS (GFLOPS)	gigabyte (GB)
tera-	T	10^{12}	teraFLOPS (TFLOPS)	terabyte (TB)
peta-	P	10^{15}	petaFLOPS (PFLOPS)	petabyte (PB)
exa-	E	10^{18}	exaFLOPS (EFLOPS)	exabyte (EB)
zetta-	Z	10^{21}	zettaFLOPS (ZFLOPS)	zettabyte (ZB)
yotta-	Y	10^{24}	yottaFLOPS (YFLOPS)	yottabyte (YB)

2020 Fastest Supercomputer

- Supercomputer Fugaku
- 415 petaflops
- ~7300000 cores ARM A64FX
- 28 Mwatts
- No. 9 in GREEN500

Usage of Fugaku

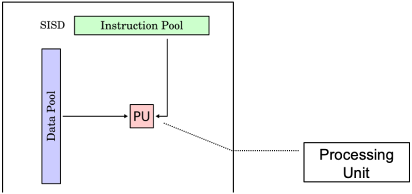
- Social and scientific priority issues
  - Innovative drug discovery
  - Personalized and preventive medicine
- Disaster prevention and global climate problems
  - Meteorological and global environmental predictions
- Innovative clean energy systems
- Evolution of the universe

Flynn's Parallel Architecture Taxonomy

- One commonly used taxonomy of parallel architecture:
  - Based on the parallelism of *instructions* and *data* streams in the most constrained component of the processor
  - Proposed by M.Flynn in 1972 (!)
- Instruction Stream:
  - A single execution flow
  - a single Program Counter (PC)
- Data Stream:
  - Data being manipulated by the instruction stream

Single Instruction Single Data (SISD)

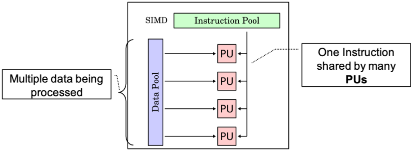
- A single instruction stream is executed
- Each instruction work on single data
- Most of the uniprocessors fall into this category



⚠This corresponds to the von Neumann architecture.

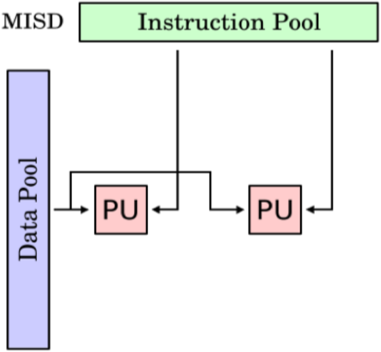
Single Instruction Multiple Data (SIMD)

- A single stream of instructions
- Each instruction works on multiple data
- Popular model for supercomputer during 1980s:
  - Exploit data parallelism, commonly known as vector processor
- Modern processor has some forms of SIMD:
  - E.g. the SSE, AVX instructions in intel x86 processors


SIMD
Instruction Pool
Data Pool
PU
PU
PU
PU
Multiple data being processed
One Instruction shared by many **PUs**

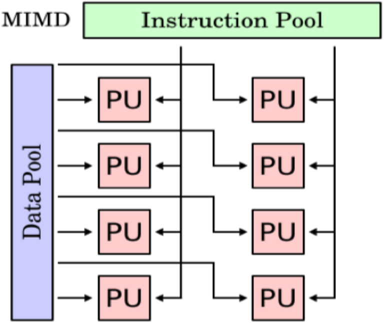
Multiple Instruction Single Data (MISD)

- Multiple instruction streams
- All instruction work on the same data at any time
- No actual implementation // systolic array



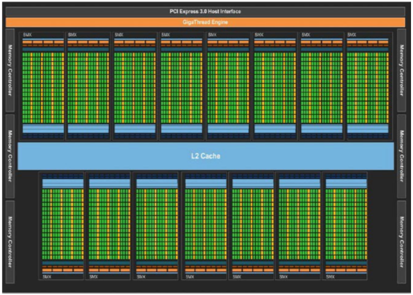
Multiple Instruction Multiple Data (MIMD)

- Each PU fetch its own instruction
- Each PU operates on its data
- Currently the most popular model for multiprocessor



Variant – SIMD + MIMD

- Stream processor (nVidia GPUs)
    - A set of threads executing the same code (effectively SIMD)
    - Multiple set of threads executing in parallel (effectively MIMD at this level)

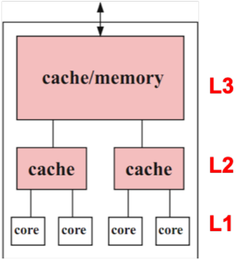


Architecture of Multicore Processors

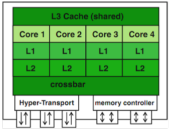
- Hierarchical design
- Pipelined design
- Network-based design

Hierarchical Design

- Multiple cores share multiple caches
- Cache size increases from the leaves to the root
- Each core can have a separate L1 cache and shares the L2, L3 cache with other cores
- All cores share the common external memory
- Common usages:
    - Standard desktop
    - Server processors
    - Graphics processing units

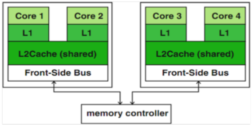


Hierarchical Design – Examples

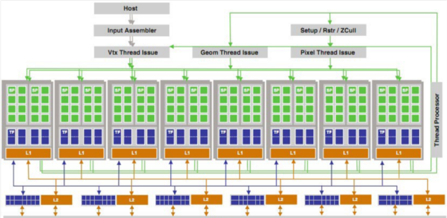


Quad-Core AMD Opteron

Each core is sophisticated, out-of-order processor to maximize ILP



Intel Quad-Core Xeon



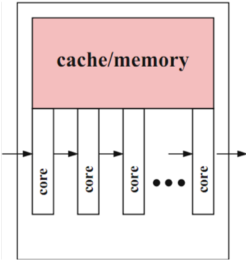
128 stream processors (SP), 16 texture / process clusters each with 8 SPs

Each core processors vectors of data

Nvidia GeForce 8800

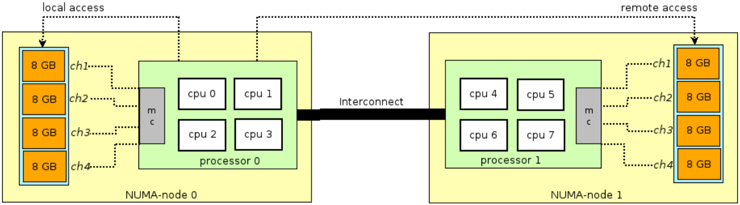
Pipelined Design

- Data elements are processed by multiple execution cores in a *pipelined way*
- Useful if same computation steps have to be applied to a long sequence of data elements
  - E.g. processors used in routers and graphics processors



Network-based Design

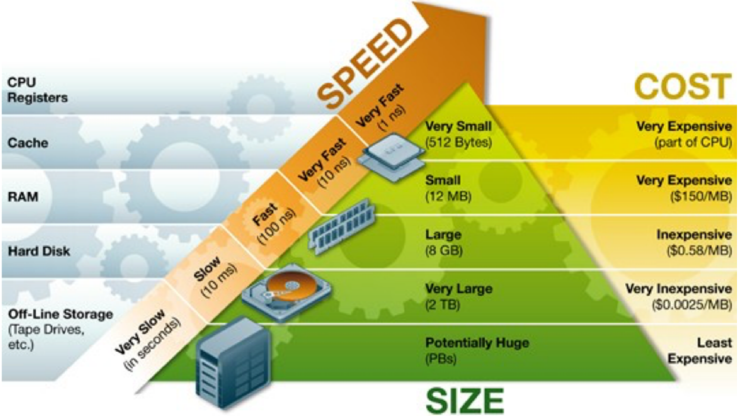
- Cores and their local caches and memories are connected via an interconnection network



Remark

The interconnection can be NUMA link, TCP/IP, RDMA, NVLink, etc.

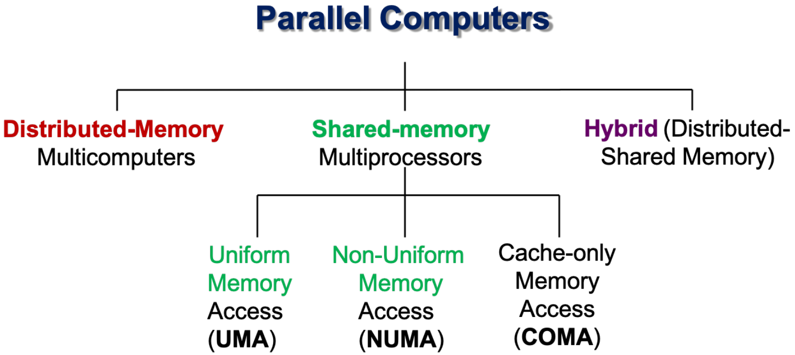
The Memory Hierarchy



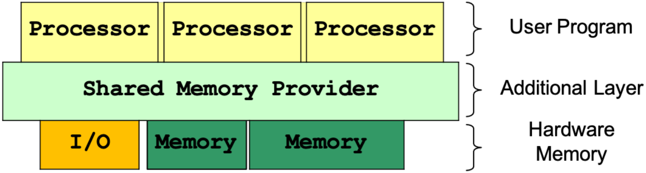
Memory Latency and Bandwidth

- Memory latency: the amount of time for a memory request (e.g., load, store) from a processor to be serviced by the memory system
  - Example: 100 cycles, 100 nsec
- Memory bandwidth: the rate at which the memory system can provide data to a processor
  - Example: 20 GB/s
- Processor "stalls" when it cannot run the next instruction in an instruction stream because of a dependency on a previous instruction

Memory Organization of Parallel Computers

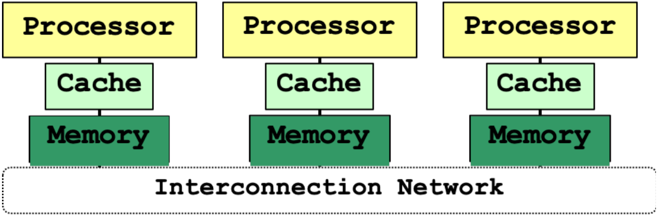

Parallel Computers
Distributed-Memory Multicomputers
Shared-memory Multiprocessors
Hybrid (Distributed-Shared Memory)
Uniform Memory Access (UMA)
Non-Uniform Memory Access (NUMA)
Cache-only Memory Access (COMA)

Shared Memory System



- Parallel programs / threads access memory through the shared memory provider:
  - which maintain the illusion of shared memory
- Program is unaware of the actual hardware memory architecture
- Data exchanges between nodes:
  - shared variables

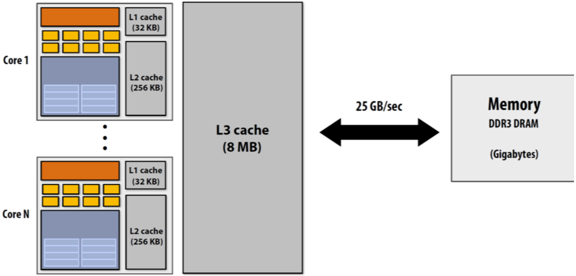
Distributed-Memory Systems



- Each node is an independent unit:
  - With processor, memory and, sometimes, peripheral elements
- Physically distributed memory module:
  - Memory in a node is private
- Data exchanges between nodes:
  - Message-passing (e.g., MPI, Java socket/RMI).

The Cache

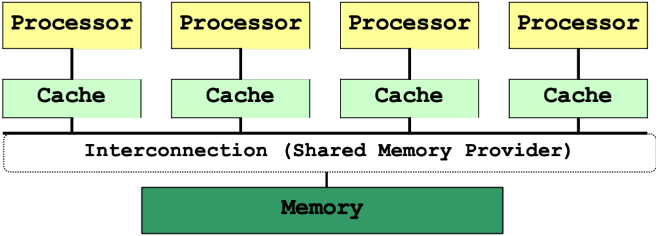
- Processors run efficiently when data is resident in caches
  - Caches reduce memory access latency
  - Caches provide high bandwidth data transfer to CPU



Further Classification – Shared Memory

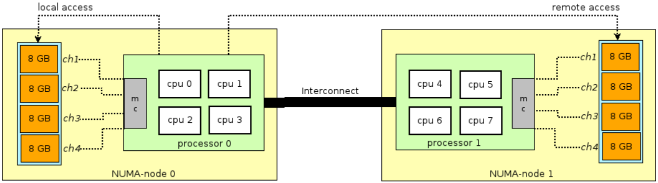
- Two factors can further differentiate shared memory systems:
    - Processor to Memory “distance” (UMA / NUMA):
        - Whether the distance is uniform
- Presence of a local cache with cache coherence protocol (CC/NCC):
    - Same shared variable may exist in multiple caches
    - Hardware ensures correctness via cache coherence protocol

Uniform Memory Access (Time) (UMA)



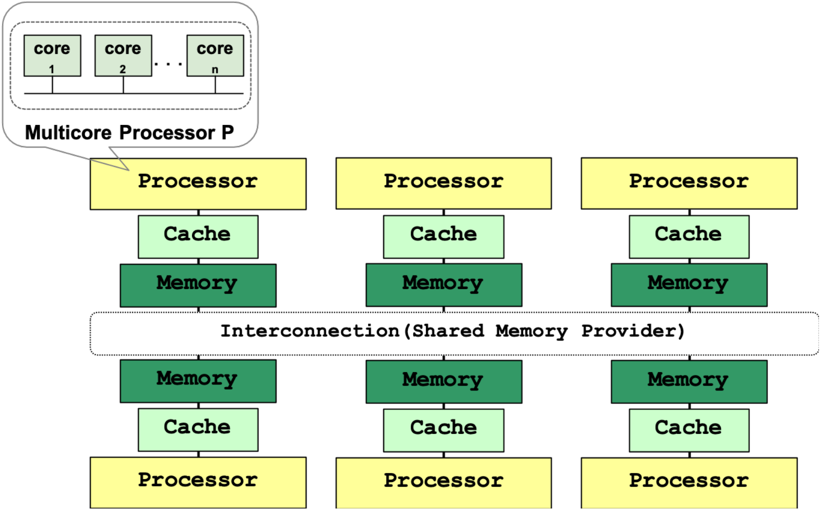
- Latency of accessing the main memory is the same for every processor:
  - Uniform access time, hence the name
- Suitable for small number of processors – due to contention
  - Related: Symmetric Multiprocessor (SMP)

Non-Uniform Memory Access (NUMA)

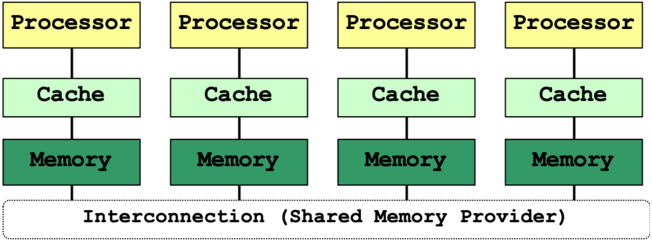


- Physically distributed memory of all processing elements are combined to form a global shared-memory address space:
  - also called *distributed shared-memory*
- Accessing local memory is faster than remote memory for a processor
  - Non-uniform access time

Example: Multicore NUMA

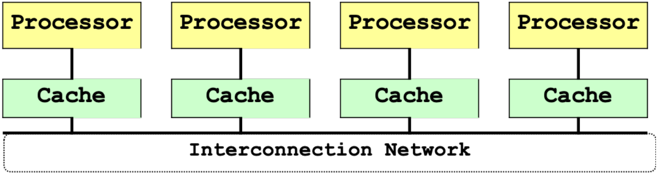


ccNUMA



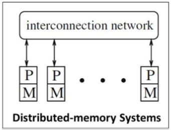
- Cache Coherent Non Uniform Memory Access
  - Each node has cache memory to reduce contention

COMA

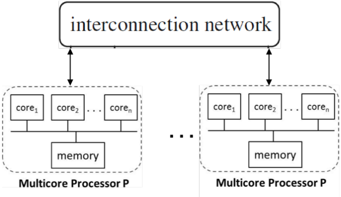


- Cache Only Memory Architecture
  - Each memory block works as cache memory
  - Data migrates dynamically and continuously according to the cache coherence scheme

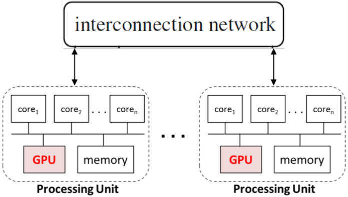
Hybrid (Distributed-Shared Memory)



Distributed-memory Systems



Hybrid with Shared-memory Multicore Processors



Hybrid with Shared-memory Multicore Processor and Graphics Processing Unit