CSE 560
Computer Systems Architecture

Multiprocessors

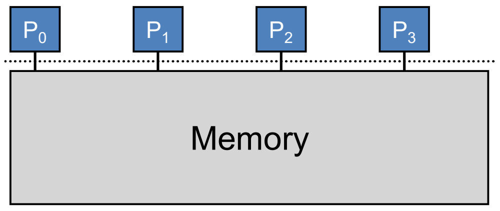
Flynn’s Taxonomy

- Proposed by Michael Flynn in 1966
- SISD – single instruction, single data
  - Traditional uniprocessor
- SIMD – single instruction, multiple data
  - Execute the same instruction on many data elements
  - Vector machines, graphics engines
- MIMD – multiple instruction, multiple data
  - Each processor executes its own instructions
  - Multicores are all built this way
  - SPMD – single program, multiple data (extension proposed by Frederica Darema)
    - MIMD machine, each node is executing the same code
- MISD – multiple instruction, single data
  - Systolic array

Shared-Memory Multiprocessors

Conceptual model

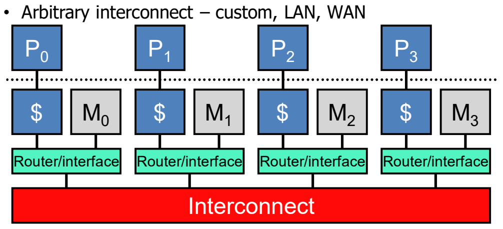
- The shared-memory abstraction
- Familiar and feels natural to programmers
- Life would be easy if systems actually looked like this…



Distributed-Memory Multiprocessors

…but systems actually look more like this

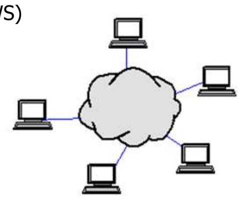
- Memory is physically distributed
  - Previously covered common address space and cache coherence
  - Scales to about 10s to 100 processors
- When we want to scale up to 1000s (or millions) of cores
  - Separate address spaces
- Arbitrary interconnect – custom, LAN, WAN



Connect Processors via Network

Cluster approach

- Off-the-shelf processors (each of which is a multicore)
- Connect using off-the-shelf networking technology
- Leverages existing components → inexpensive to design
- Cloud service providers do this a lot!
  - Amazon Web Services (AWS)
  - Microsoft Azure
- Scales up very easily
  - 1000s of nodes
- Long latency to move data
  - Traverse network for one cache line? Nope!



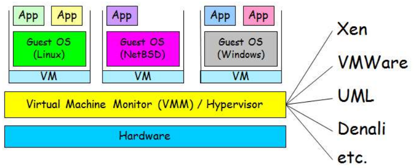
Programming Models

- The interconnect is a Local-Area Network (LAN)
  - TCP/IP message delivery
    - IP addresses
    - Network handles routing, etc.
  - Socket-based programming
- Higher-level abstractions
  - Distributed shared memory
    - Works but performs poorly – latency again
  - Map-Reduce
    - Hadoop, etc.
  - Streaming data
    - Apache Storm, etc.
  - Explicit message passing (more later)

Virtualization

Sharing the processor cores

- VM technology allows multiple virtual machines to run on a single physical machine
- Hypervisor schedules VMs onto physical cores



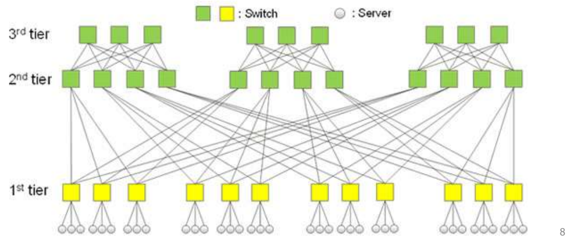
7

7

Cluster Interconnect

Ethernet Switches

- 1st tier are top-of-rack (ToR) switches
- Additional tiers connect racks, top tier talks to outside world
- Lots of redundant paths



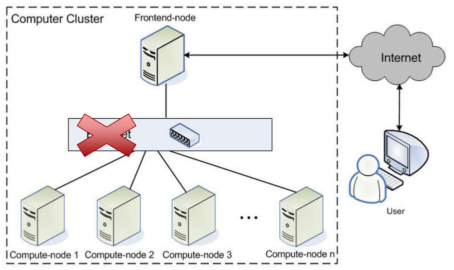
8

8

Can we fix latency issue?

Cluster approach

- TCP/IP network technology is dominant
  - But is it needed? Or just readily available?



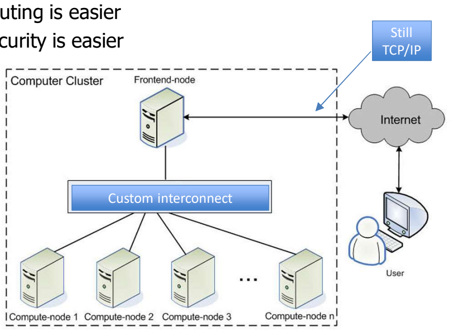
9

9

Custom Interconnect

Known topology, trusted environment

- Routing is easier
- Security is easier

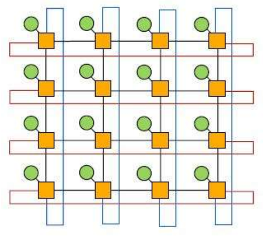


10

10

Interconnect Topologies

- Mesh
- Torus (wraparound mesh)
- Low-overhead message delivery
- Routing is straightforward
  - Move along row to destination column
  - Move along column to destination
- Forwarding can be fast
  - Old-school: store-and-forward
  - Modern: cut-through


2D Torus

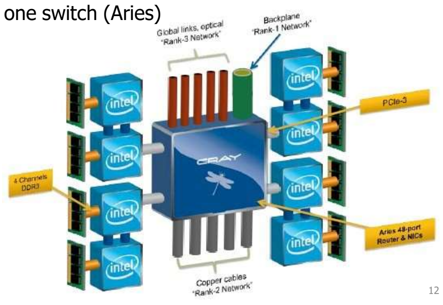
11

11

Cray Dragonfly

Custom Design for Supercomputers

- Big applications with lots of parallelism
- All tiers in one switch (Aries)

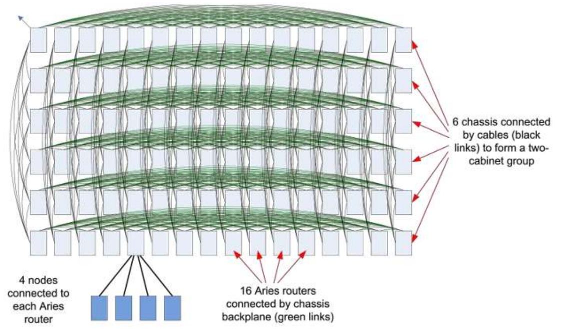

Global links, optical "Rank-3 Network"
Backplane "Rank-1 Network"
PCIe-3
4 Channels DDR3
Aries 48-port Router & NICs
Copper cables "Rank-2 Network"
intel
CRAY

12

12

Cray Dragonfly Network

Mesh with additional links

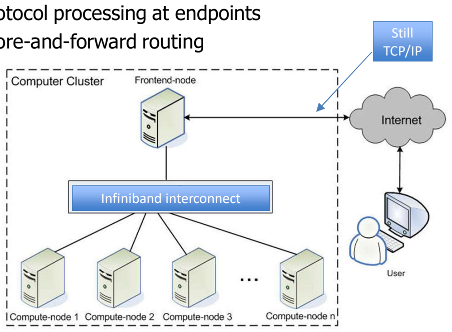

6 chassis connected by cables (black links) to form a two-cabinet group
4 nodes connected to each Aries router
16 Aries routers connected by chassis backplane (green links)

13

Back to Standardized Interconnect

Issue with Ethernet is latency

- Protocol processing at endpoints
- Store-and-forward routing



14

Infiniband Network

- Standardized technology
  - Multiple vendors
    - Equipment works together
    - Competition
  - Not trying to be the "Internet"

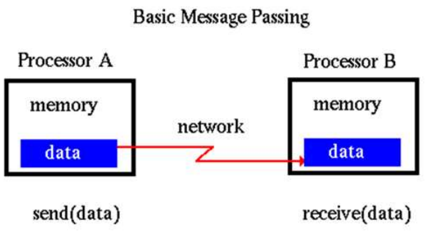
- Focus on low latency interconnect needs
  - Minimize protocol processing
    - E.g., easier routing, simpler security model
  - Fast forwarding
    - Cut-through packet delivery
  - Remote Direct Memory Access (RDMA)
    - Supports single-ended messaging

15

Programming Paradigm

Message Passing

- MPI (Message Passing Interface) is de facto standard
- Used by almost all supercomputing applications

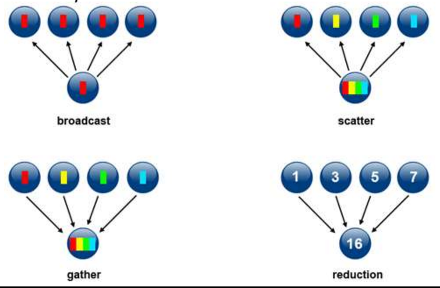


16

More MPI

MPI capabilities beyond just send() and rcve()

- One-sided communication: get() and put()
- Collective operations



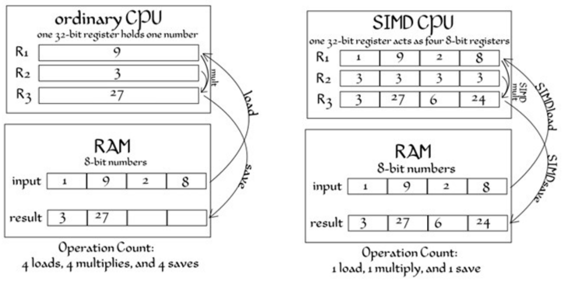
17

Flynn's Taxonomy

- Proposed by Michael Flynn in 1966
- SISD – single instruction, single data
  - Traditional uniprocessor
- SIMD – single instruction, multiple data
  - Execute the same instruction on many data elements
  - Vector machines, graphics engines
- MIMD – multiple instruction, multiple data
  - Each processor executes its own instructions
  - Multicores are all built this way
  - SPMD – single program, multiple data (extension proposed by Frederica Darema)
    - MIMD machine, each node is executing the same code
- MISD – multiple instruction, single data
  - Systolic array

18

SIMD Instructions

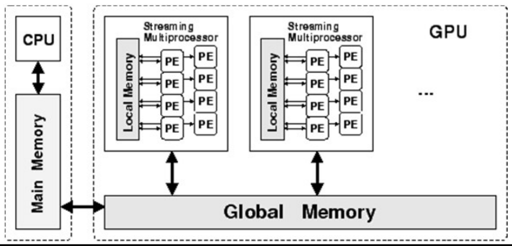


By Decora at English Wikipedia, CC BY-SA 3.0,
https://commons.wikimedia.org/w/index.php?curid=30547549

Graphics Engines

Heterogeneous Multiprocessor

- Many processing elements (PE), many threads per PE
- Collections of threads execute in lock-step (SIMD-like)
  - Hide latency to memory by switching threads



Flynn’s Taxonomy

- Proposed by Michael Flynn in 1966
- SISD – single instruction, single data
  - Traditional uniprocessor
- SIMD – single instruction, multiple data
  - Execute the same instruction on many data elements
  - Vector machines, graphics engines
- MIMD – multiple instruction, multiple data
  - Each processor executes its own instructions
  - Multicores are all built this way
  - SPMD – single program, multiple data (extension proposed by Frederica Darema)
    - MIMD machine, each node is executing the same code
- MISD – multiple instruction, single data
  - Systolic array

Systolic Arrays

H.T. Kung, “Why Systolic Architectures?,” *Computer*, 1982

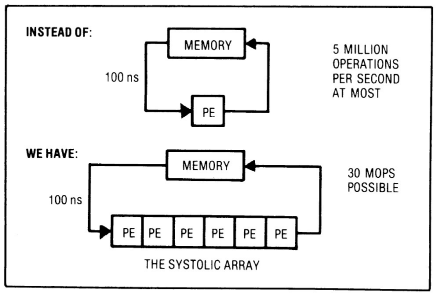
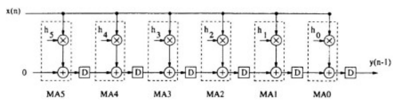


Figure 1. Basic principle of a systolic system.

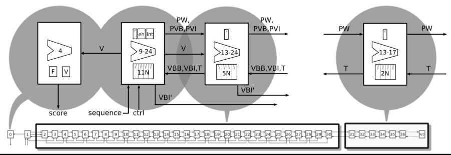
Systolic Arrays

Purpose-built design for specific problem

- Custom PE, replicated many times
- E.g., array of MA (multiply-accumulate) units for FIR filter



- RNA folding [Jacob et al. 2010]



Tensor Processing Unit

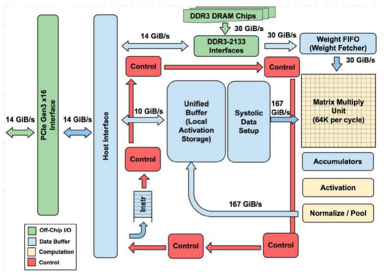


Figure 1. TPU Block Diagram. The main computation part is the yellow Matrix Multiply unit in the upper right hand corner. Its inputs are the blue Weight FIFO and the blue Unified Buffer (UB) and its output is the blue Accumulators (Acc). The yellow Activation Unit performs the nonlinear functions on the Acc, which go to the UB.

Tensor Processing Unit

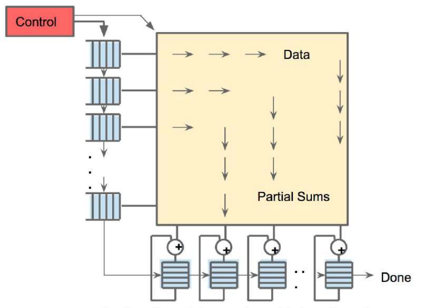


Figure 4. Systolic data flow of the Matrix Multiply Unit. Software has the illusion that each 256B input is read at once, and they instantly update one location of each of 256 accumulator RAMs.

25

25

Flynn's Taxonomy

- Proposed by Michael Flynn in 1966
- SISD – single instruction, single data
  - Traditional uniprocessor
- SIMD – single instruction, multiple data
  - Execute the same instruction on many data elements
  - Vector machines, graphics engines
- MIMD – multiple instruction, multiple data
  - Each processor executes its own instructions
  - Multicores are all built this way
  - SPMD – single program, multiple data (extension proposed by Frederica Darema)
    - MIMD machine, each node is executing the same code
- MISD – multiple instruction, single data
  - Systolic array

26

26