Orange Coast College
Business Division
Computer Science Department

CS 116- Computer Architecture

Chapter 9
Multiprocessing & Multiprogramming
(Some material from Tannenbaum and Null & Lobur)
Multiprocessors

- **Idea:**
  - Create powerful computers by connecting many smaller ones

- **Good news:**
  - Works for timesharing (better than supercomputer)

- **Bad news:**
  - Its really hard to write good concurrent programs many commercial failures
Connection Options

- Single-bus
Connection Options

- Network-connected

Diagram:

- Processor
- Cache
- Memory

Connect to Network
Questions

- How do parallel processors share data?
  - Single address space (SMP vs. NUMA)
  - Message passing

- How do parallel processors coordinate?
  - Synchronization (locks, semaphores)
  - Built into send / receive primitives
  - Operating system protocols

- How are they implemented?
  - Connected by a single bus
  - Connected by a network
Parallel Computer Architecture

- For many applications any computing power available today is still insufficient
- **Examples**
  - Science
  - Engineering
  - Industry
- **Problem:**
  - Circuit speed can’t be increased indefinitely (Amdahl’s Law).
- **Reasons:**
  - Speed of light & getting electrons to move faster
  - Heat dissipation
  - Electromagnetic interferences
  - Transistor size & quantum mechanics
    - Speed of a single CPU is limited
    - We need more CPU’s (Parallelism)
Parallel Computer Architecture

- Parallelism can be achieved at
  - Hardware level
  - Software level

- For hardware the complexity is in the CPU
  - Design differs in
    - How many elements are present
    - What are they like
    - How are they connected

- For software the complexity is in the compiler
Parallel Computer Architecture

- Hardware level:
  - Instruction level:
    - Pipelining & superscalar designs
  - CPU level:
    - Superscalar:
      - Replicate entire CPUs or portion of CPUs & make them work together
    - Super-pipelining
      - If pipeline has stages that needs less than 1/2 a clock cycle to execute, internal clock can be added which, when running at double speed of the external clock can complete two tasks per external clock cycle
Parallel Computer Architecture

Software level:

- VLIW (Very Long Instruction Word)
  - Rely entirely on the compiler
  - Pack independent instructions into one long instruction
    - Usually 4-8 instructions
  - The compiler arbitrates all dependencies
  - Any modifications that could affect the scheduling of instructions requires a re-compilation of the code
  - Leads to significant amount of code generated

Examples:

- IA-64 Intel’s Itanium uses VIEW processor
Design Issues for Parallel Computers

- Processing elements (CPU)
- Memory Modules
- Interconnection of elements
- Nature of application

Each is affected by:
  - Nature
  - Size
  - Number
Design Issue: CPU

- **Nature:**
  - ALU
  - Partial CPU
  - Complete CPU

- **Size:**
  - Fraction of a chip:
    - One computer having several CPU’s
    - Complete computer (Own memory & I/O devices)

- **Number limitations:**
  - If chip => ~ millions of elements
  - If Computer => ~ hundreds?
Design Issue: Memory Elements

- **Nature:**
  - One memory
  - Several memories
  - Integrated with the CPU or Located on different circuit board
  - Cache or RAM
  - Number of levels in the hierarchy

- **Size:**
  - Kbytes
  - Mbytes
  - Gbytes
  - Terabytes
  - Whatever comes next ...

- **Number**
  - Depend on size & nature
Design Issue: Interconnection

- Different parts of the same job must communicate to exchange information

- Static
  - All components are wired up in a fixed way
    - Star
    - Ring
    - Grid
    - . . .

- Dynamic
  - Switching network that can dynamically route messages between components
Design Issue: Nature of Application

- **Multiple independent jobs**
  - Accommodate more users
  - No communication needed between jobs
  - Examples:
    - UNIX timesharing system handling thousands of remote users
    - ATMs (Teller machines)
    - Airline reservation system
    - Web servers
    - Independent simulation runs using different parameter sets

- **Single (dependent) job**
  - Used to speed up a single job
  - All processors used to run that job
  - Use pipelining or parallel processors
  - Example:
    - Chess program: Board analyzed to find all possible moves
Vector Processors

- Referred to as “Super computers”
- Specialized, heavily pipelined
- Performs operations on entire vectors & matrices at once
- Arithmetic operations can be overlapped
- Use “Vector Registers” that hold several vector elements at one time and work in FIFO fashion
- Output from the vector registers is sent to the pipeline, and output from the pipeline is sent back to the vector registers
- Two register categories:
  - Register-register vector processor:
    - Source & destinations are registers
  - Memory-memory vector processor:
    - Operands from memory routed directly to ALU, & result directed back to memory
- Applications:
  - Weather forecast
  - Medical diagnostics
  - Image processing
Parallelism's Grain size

- Usually refers to algorithms & SW
- Has direct analogy in HW
- Categories:
  - Course-grained
  - Fine-grained
Coarse-grained Parallelism

- Running large pieces of SW or HW units in parallel with large sizes

Characteristics
- Low speed
- Loosely coupled
  - Little or no communication between the pieces

Examples:
- Complete user program
- Independent CPU
Fine-grained Parallelism

- Pieces of SW or HW communicate heavily

Characteristics

- Small sizes
- High-speed
- Tightly coupled
  - Components interact over high bandwidth communication network
Parallel Computer Models

Parallel Computer Architecture
- Von Newman Architecture SISD (single instruction single data stream)
- SIMD (single instruction multiple data stream)
- MISD (multiple instruction single data stream) (Impractical)
- MIMD (Multiple instruction multiple data stream)

- Vector processor
- Array processor
- multiprocessor
- multi-computer

- UMA (Uniform memory access)
- COMA (cache only memory access)
- NUMA (Non-uniform memory access)
- MPP (massively parallel processors)
- COW (Cluster of workstations)

- Bus
- Switched
Flynn’s Taxonomy

- A classification of computer architectures based on the number of streams of instructions and data
- See Google search
  - SISD: Single instruction stream, single data stream
  - SIMD: Single instruction stream, multiple data streams
  - MISD: Multiple instruction streams, single data stream
  - MIMD: Multiple instruction streams, multiple data streams
UMA vs. NUMA

- Shared Memory MIMD machines can be divided into
  - UMA (Uniform Memory Access)
    - All memory access takes the same amount of time
    - One shared memory connected to a group of processors through a bus or switch network
    - All processors have equal access to the memory, according to an established protocol
    - Limited scalability
    - Used by Sun’s Ultra Enterprise, IBM’s iSeries & pSeries, & DEC’s AlphaServer
  - NUMA (Non-uniform Memory Access)
    - Each processor has its own piece of memory, but sees the memory as a contiguous addressable entity
    - Memory access time is inconsistent across address space
      - Nearby memory takes less time to read than a memory that is further away
    - Used by Sequent’s NUMA-Q & Origin2000 by Silicon Graphics
Communication Models

- Multiprocessors (Shared memory/Single address-space)
- Multicomputers (Distributed memory)
- Hybrid (Partially shared, partially distributed)
Communication Models

- Multiprocessors (Shared memory)
  - All CPUs share a common memory
  - Single virtual address space
  - Single Physical address space
  - Single copy of OS
  - Single set of tables (memory allocation table)
  - Any CPU can read/ write using Load/Store instructions
  - CPUs communicate through memory contents

Advantages:
- Easy to program
- Applicable in a wide range of problems

Disadvantages:
- Hard to build but
Shared Memory Models

- **UMA Example:**
  - Image partitioned into sections
  - Each section analyzed by a different CPU
  - CPUs have access to the entire image
  - Some objects can occupy multiple sections
  - Coordination between CPU’s is needed
  - Load/store used to access memory
Distributed Memory Models

- Each CPU has its own private memory that can be accessed using load/store
- One physical address space per CPU
- Each machine has its own page table
- OS simulates shared memory by providing a single system-wide paged shared virtual address space
- OS satisfies page fault requests
- Communication mechanism needed (Send/Receive) to pass messages between CPU’s
- Interconnection network passes messages between CPUs
Distributed Memory Models

- **Advantages:**
  - Easy to build

- **Disadvantages:**
  - Difficult to program. Languages must provide statements for handling unshared memories
  - Can’t communicate through Read/Write
  - Correctly dividing up data and placing them in optimal location is a major issue
Distributed Memory Models

- NUMA Example:
Hybrid Memory Models

- Partially shared, partially distributed
- Relatively easy to build & relatively easy to program
- Combine the strengths of the previous two approaches
Alternative Parallel Processing Approaches

- **Dataflow computing**
  - Control of the program (execution flow) is directly tied to the data
  - An instruction is executed when the data necessary for execution becomes available
  - The actual order of instruction has no bearing on the order in which they are eventually executed
  - Instructions reference other instructions, not the memory
Alternative Parallel Processing Approaches

- **Neural networks**
  - Useful in dynamic situations where we can’t formulate an exact algorithmic solution, & processing is based on accumulated previous behavior
  - Based on parallel architecture of human brains
  - Can deal with imprecise & probabilistic information and can learn from experience
  - Composed of large number of processing elements (PE) that individually handle one piece of a much larger problem
  - Network’s learning algorithm: The set of rules that governs changes to be made
  - See: [http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html](http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html)
Alternative Parallel Processing Approaches

- Systolic arrays
  - Analog to how blood rhythmically flows through a biological heart
  - A network of processing elements that rhythmically compute data by circulating it through the system
  - Are a variation of SIMD computers that incorporates large array of simple processors that use vector pipelines for data flow
  - Employ a high degree of parallelism through pipelining and can sustain very high throughput
  - Used in very specific types of problems
More Multiprocessing Links:

- **FOLDOC:**
  - [http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?parallel+processing](http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?parallel+processing)

- **Michael S. Eldred, & William E. Hart:**

- **David A. Bader:**
  - [http://dsonline.computer.org/parallel/](http://dsonline.computer.org/parallel/)
  - Berkley:
  - University of Missouri:
    - [http://web.umr.edu/~ercal/387/387.html](http://web.umr.edu/~ercal/387/387.html)
From Chapter 9 on CD
Supercomputers

Plot of top 500 supercomputer sites over a decade:

Single Instruction multiple data (SIMD)

Cluster (network of workstations)

Cluster (network of SMPs)

Massively parallel processors (MPPs)

Shared-memory multiprocessors (SMPs)

Uniprocessors
Using Multiple Processors is an Old Idea

- Some SIMD designs:

<table>
<thead>
<tr>
<th>Institution</th>
<th>Name</th>
<th>Maximum no. of proc.</th>
<th>Bits/proc.</th>
<th>Proc. clock rate (MHz)</th>
<th>Number of FPUs</th>
<th>Maximum memory size/system (MB)</th>
<th>Communications BW/system (MB/sec)</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>U. Illinois</td>
<td>Illiac IV</td>
<td>64</td>
<td>64</td>
<td>13</td>
<td>64</td>
<td>1</td>
<td>2,560</td>
<td>1972</td>
</tr>
<tr>
<td>ICL</td>
<td>DAP</td>
<td>4,096</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>2,560</td>
<td>1980</td>
</tr>
<tr>
<td>Goodyear</td>
<td>MPP</td>
<td>16,384</td>
<td>1</td>
<td>10</td>
<td>0</td>
<td>2</td>
<td>20,480</td>
<td>1982</td>
</tr>
<tr>
<td>Thinking Machines</td>
<td>CM-2</td>
<td>65,536</td>
<td>1</td>
<td>7</td>
<td>2048 (optional)</td>
<td>512</td>
<td>16,384</td>
<td>1987</td>
</tr>
<tr>
<td>Maspar</td>
<td>MP-1216</td>
<td>16,384</td>
<td>4</td>
<td>25</td>
<td>0</td>
<td>256 or 1024</td>
<td>23,000</td>
<td>1989</td>
</tr>
</tbody>
</table>

FIGURE 9.11.1 Characteristics of five SIMD computers. Number of FPUs means number of floating-point units.

- Costs for the Illiac IV escalated from $8 million in 1966 to $32 million in 1972 despite completion of only ¼ of the machine. It took three more years before it was operational!

  “For better or worse, computer architects are not easily discouraged”

- Lots of interesting designs and ideas, lots of failures, few successes
Topologies

a. 2-D grid or mesh of 16 nodes

b. n-cube tree of 8 nodes ($8 = 2^3$ so $n = 3$)
Clusters

- Constructed from whole computers
- Independent, scalable networks

Strengths:
- Many applications amenable to loosely coupled machines
- Exploit local area networks
- Cost effective / Easy to expand

Weaknesses:
- Administration costs not necessarily lower
- Connected using I/O bus
- Highly available due to separation of memories
- In theory, we should be able to do better
Google

- Serve an average of 1000 queries per second
- Google uses 6,000 processors and 12,000 disks
- Two sites in silicon valley, two in Virginia
- Each site connected to internet using OC48 (2488 Mbit/sec)
- Reliability:
  - On an average day, 20 machines need rebooted (software error)
  - 2% of the machines replaced each year
- In some sense, simple ideas well executed. Better (and cheaper) than other approaches involving increased complexity
Concluding Remarks

- Evolution vs. Revolution
  - “More often the expense of innovation comes from being too disruptive to computer users”
  - “Acceptance of hardware ideas requires acceptance by software people; therefore hardware people should learn about software. And if software people want good machines, they must learn more about hardware to be able to communicate with and thereby influence hardware engineers.”
Concluding Remarks

**Figure 9.10.1** The evolution-revolution spectrum of computer architecture. The first four columns are distinguished from the last column in that applications and operating systems may be ported from other computers rather than written from scratch. For example, RISC is listed in the middle of the spectrum because user compatibility is only at the level of high-level languages (HLLs), while microprogramming allows binary compatibility, and parallel processing multiprocessors require changes to algorithms and extending HLLs. You see several flavors of multiprocessors on this figure. “Timeshared multiprocessor” means multiprocessors justified by running many independent programs at once. “CC-UMA” and “CC-NUMA” mean cache-coherent UMA and NUMA multiprocessors running parallel sub-systems such as databases or file servers. Moreover, the same applications are intended for “message passing.” “Parallel processing multiprocessor” means a multiprocessor of some flavor sold to accelerate individual programs developed by users. (See section 9.11 to learn about SIMD.)