Alternative Approaches

To

Concurrency
* Concurrency Basics
  -- Granularity
  -- SIMD/MIMD
  -- Supercomputers vs. Multi
  -- Data Flow vs. Control Flow

* Data Flow Basics
  -- Fire when ready
  -- Irregular parallelism
  -- Instances
  -- Example programs

* Single instruction stream
  -- SIMD (Vectors, Arrays)
  -- VLIW (now EPIC)
  -- DAE
  -- HPS

* MP Basics
  -- Metrics: Speedup, Redundancy, Efficiency
  -- Amdahl’s Law
  -- Cache Coherency (Consistency)
  -- Interconnection Networks
    (cost, latency, contention)

* NOT Single instruction stream
  -- cm* (NUMA)
  -- HEP (today, SMT)
  -- Hypercube
  -- Target-triggered (the MOV instruction)
  -- CMP
  -- Tiling the plane
    x early: nonVon, BVM, CM-1
    x today: TRIPS, Cell, Niagara, RAW, Wavescalar
Granularity of Concurrency

* Intra-Instruction (Pipelining)

* Parallel Instructions (SIMD, VLIW)

* Tightly-coupled MP

* Loosely-coupled MP
SIMD/MIMD

SISD  The Typical Pentium-Pro, for example
MISD
SIMD  Array Processor, Vector Processor
MIMD  Multiprocessor

and, Note:

Pipelined SISD

SIMD
Pipelining

Pipelined:

Superscalar:

Superpipelined:
One Supercomputer

vs.

“The Multi”

(...Except Even Supercomputers have adopted the multi approach)

\[ 1 \times 2^n \quad 2^k \times 2^{n-k} \quad 2^n \times 1 \]

Why do we care?

-- Economic Answer
-- Strategic Answer
-- Scientific Answer

Scalability
Amdahl's Law

* Speed-up as a function of the parallelizability ($\alpha$) of the application

**Speedup vs. Parallelizability for a given number of processors ($p$)**

* Speed-up of an application as we add more and more processors ($p$)

**Speedup vs. Number of Processors ($p$) for a given $\alpha$**
**MP vs. Multicomputer Network**

* Shared memory vs. Message passing

* Easier for software, or easier for hardware

* No free lunch
  - Cache Consistency
  - Memory Contention
A Unit of Computation:
The Data Flow Node

OR,

* R ARG1 R ARG2 Dest. Of Result

The Operation
(In Larger Granularity Systems, "The Compound Function")

Fires When Ready
The Firing Rule:

When all Inputs Have Tokens

(Note: Safe vs. Queues)

*Conditional

*Relational

*Barrier Synch
An Example Data Flow Program:

**Factorial** (Done, Iteratively)
Characteristics of Data Flow

* Data Driven Execution of Instruction-level Graphical Code
  -- Nodes are Operators
  -- Arccs are I/O

* Only REAL Dependencies Constrain Processing
  -- Anti-Dependencies Don't (write-after-read)
  -- Output Dependencies Don't (write-after write)
  -- NO Sequential l-stream (No PC)

* Operations Execute Asynchronously

* Instructions Do Not Reference Memory
  (at least, memory as we understand it)

* Execution Triggered By Presence of Data
  -- Safe vs. Queues
**SIMD**

Vector Processors, Array Processors

<table>
<thead>
<tr>
<th>LD</th>
<th>*</th>
<th>@</th>
<th>ST</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>LD&lt;sub&gt;1&lt;/sub&gt;</th>
<th>LD&lt;sub&gt;2&lt;/sub&gt;</th>
<th>LD&lt;sub&gt;3&lt;/sub&gt;</th>
<th>LD&lt;sub&gt;4&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD&lt;sub&gt;1&lt;/sub&gt;</td>
<td>LD&lt;sub&gt;2&lt;/sub&gt;</td>
<td>LD&lt;sub&gt;3&lt;/sub&gt;</td>
<td>LD&lt;sub&gt;4&lt;/sub&gt;</td>
</tr>
<tr>
<td>* 1</td>
<td>* 2</td>
<td>* 3</td>
<td>* 4</td>
</tr>
<tr>
<td>@1</td>
<td>@2</td>
<td>@3</td>
<td>@4</td>
</tr>
<tr>
<td>ST&lt;sub&gt;1&lt;/sub&gt;</td>
<td>ST&lt;sub&gt;2&lt;/sub&gt;</td>
<td>ST&lt;sub&gt;3&lt;/sub&gt;</td>
<td>ST&lt;sub&gt;4&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**time**
VLIW

* Static Scheduling
  - Everything in lock step
  - Trace Scheduling

* Generic Model

```
  Memory

  PC → α β γ β

  PE   PE   PE   PE
```

38
Early Form of Decoupled - Access/Execute

* Andrew Plezskun, Univ. of Illinois
  SMA

* James E. Smith, Univ of Wisconsin
  DAE
HPS As Evolution

Diagram showing a hierarchical structure with labeled nodes and arrows indicating connections.
Note: A well-meaning student told me to get rid of this slide. cm* is old. People will think you are an old man, and not take you seriously.
Cosmic Cube

(Example: $k = 4$)