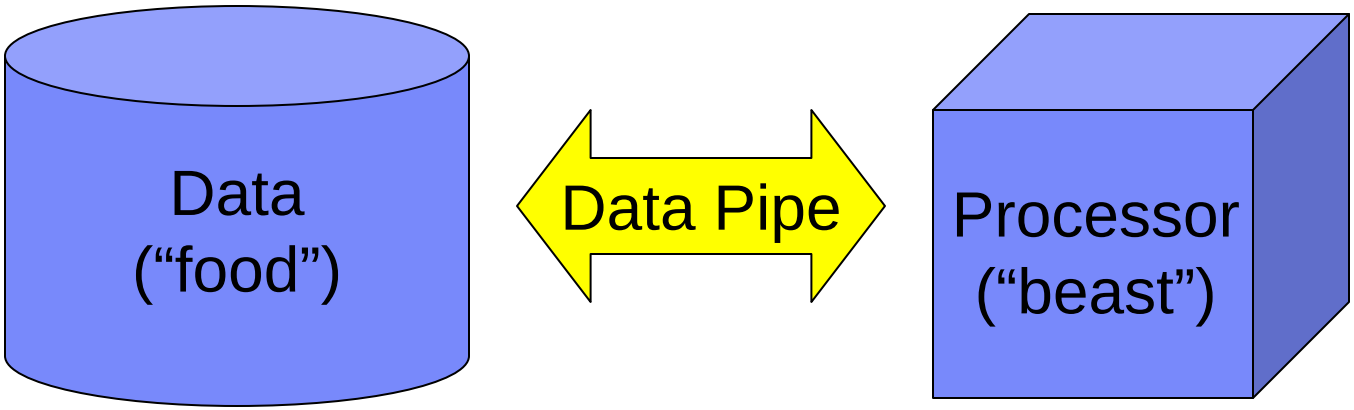
IBM Research

Proper Care & Feeding of Multicore Beasts

Michael Perrone
IBM Master Inventor
Mgr, Cell Solutions Dept.

© 2007 IBM Corporation

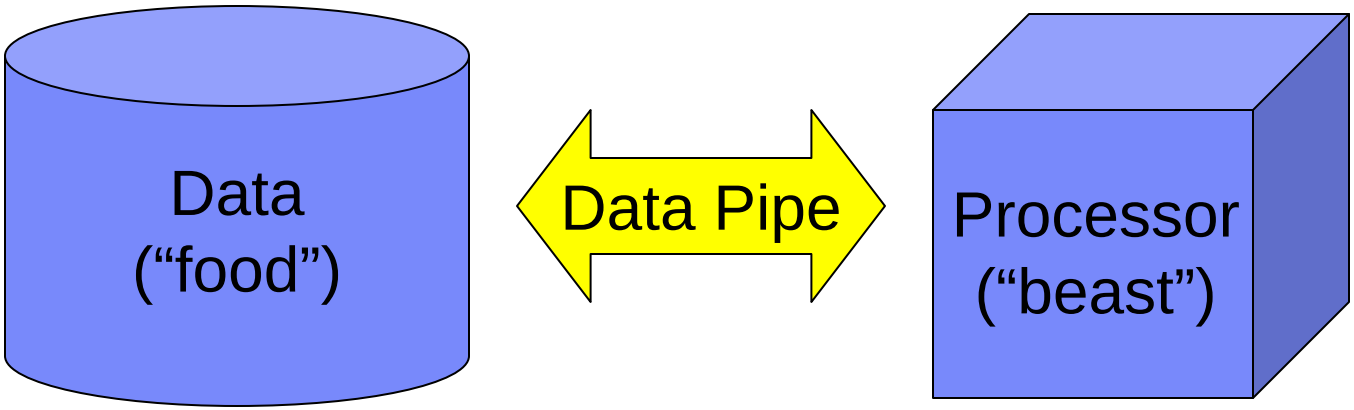
The Hungry Beast



- **Pipe too small = starved beast**
- **Pipe big enough = well-fed beast**
- **Pipe too big = wasted resources**

© 2007 IBM Corporation

The Hungry Beast



- **Pipe too small = starved beast**
- **Pipe big enough = well-fed beast**
- **Pipe too big = wasted resources**

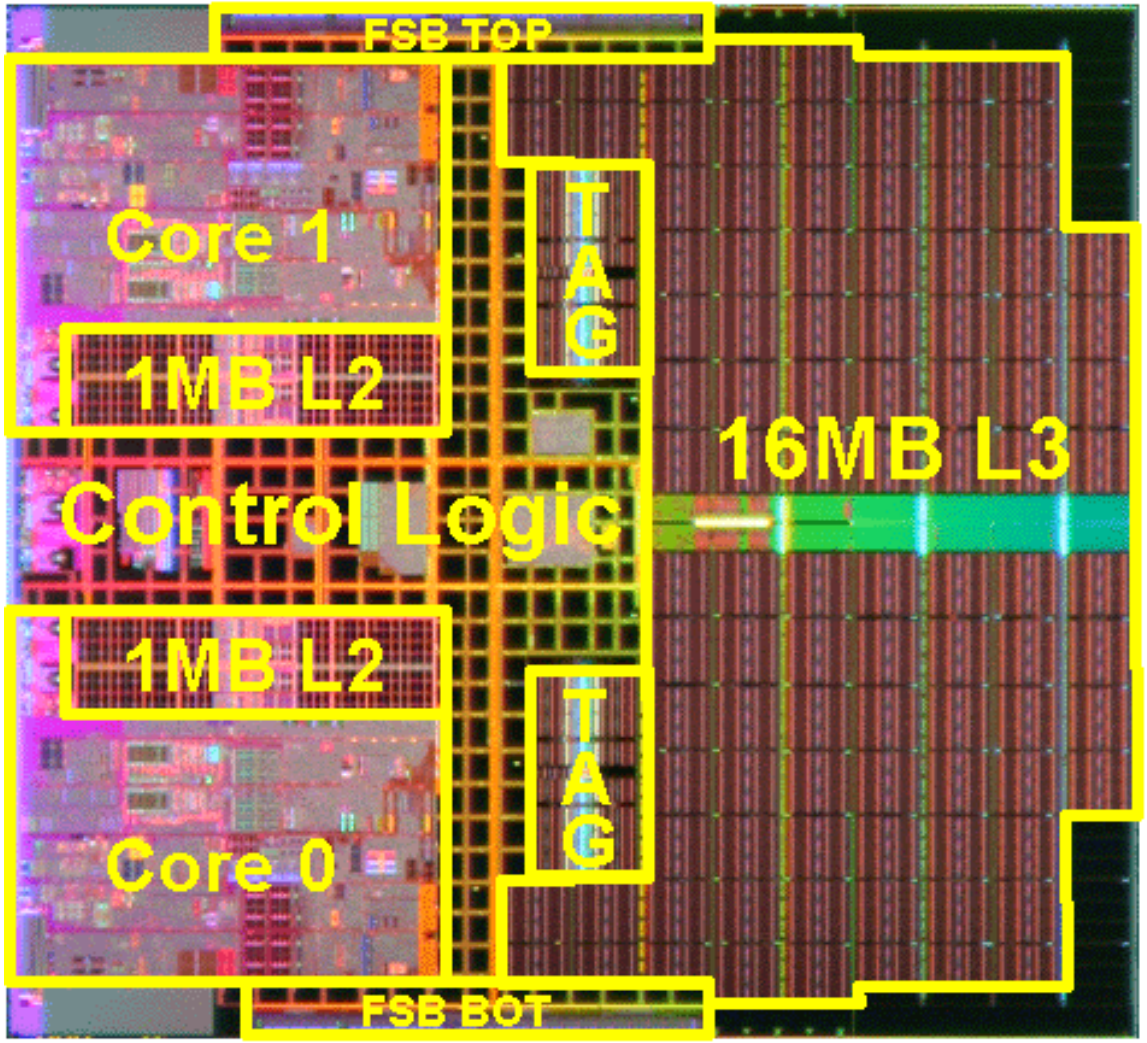
- **If flops grow faster than pipe capacity…**

  … the beast gets <u>hungrier</u>!

© 2007 IBM Corporation

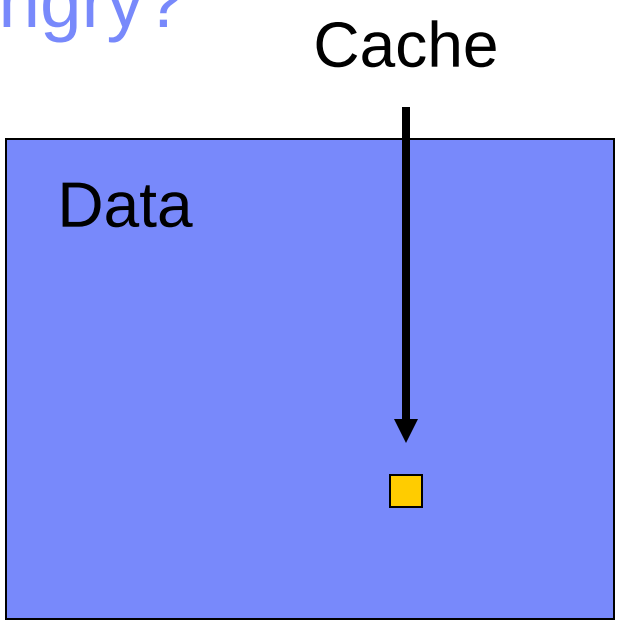
Move the food closer

- **Example: Intel Tulsa**
  - Xeon MP 7100 series
  - 65nm, 349mm2, 2 Cores
  - 3.4 GHz @ 150W
  - ~54.4 SP GFlops
  - http://www.intel.com/products/processor/xeon/index.htm
- **Large cache on chip**
  - ~50% of area
  - Keeps data close for efficient access
- **If the data is local, the beast is happy!**
  - True for many algorithms



© 2007 IBM Corporation

What happens if the beast is still hungry?

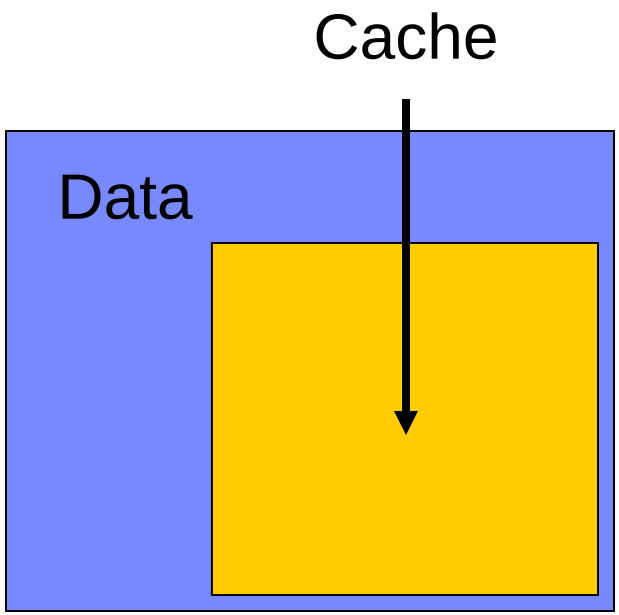


- **If the data set doesn't fit in cache**
  - Cache misses
  - Memory latency exposed
  - Performance degraded
- **Several important application classes don't fit**
  - Graph searching algorithms
  - Network security
  - Natural language processing
  - Bioinformatics
  - Many HPC workloads

© 2007 IBM Corporation

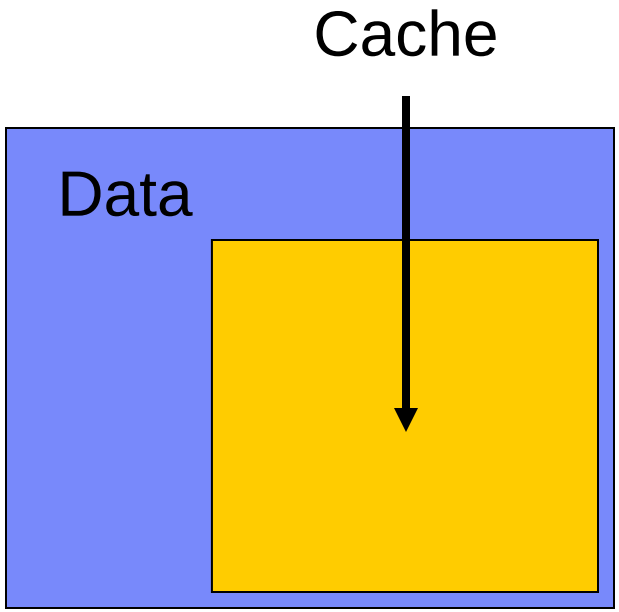
Make the food bowl larger

- **Cache size steadily increasing**
- **Implications**
  - Chip real estate reserved for cache
  - Less space on chip for computes
  - More power required for fewer FLOPS



© 2007 IBM Corporation

Make the food bowl larger



- **Cache size steadily increasing**
- **Implications**
  - Chip real estate reserved for cache
  - Less space on chip for computes
  - More power required for fewer FLOPS
- **But…**
  - Important application working sets are growing faster
  - Multicore even more demanding on cache than uni-core

© 2007 IBM Corporation

Cell Approach: Feed the beast more efficiently

- **Explicitly "orchestrate" the data flow between main memory and each SPE's local store**
  - Use the Memory Flow Controller (MFC) to post DMA commands to gather & scatter arbitrary segments of memory between main memory and local store
  - Enables detailed programmer control of data flow
    - Get & Put the data when & where you want it
    - Extremely powerful mechanism to efficiently support streaming applications
    - Enables latency hiding: Perform reads, writes & computes simultaneously
    - Critical when data access patterns are not easily predictable
  - Avoids restrictive HW cache management
    - No need to rely on (potentially inefficient) HW cache mechanisms
    - HW cache management unlikely to determine optimal data flow
  - Allows more efficient use of the existing bandwidth

© 2007 IBM Corporation

Cell Approach: Feed the beast more efficiently

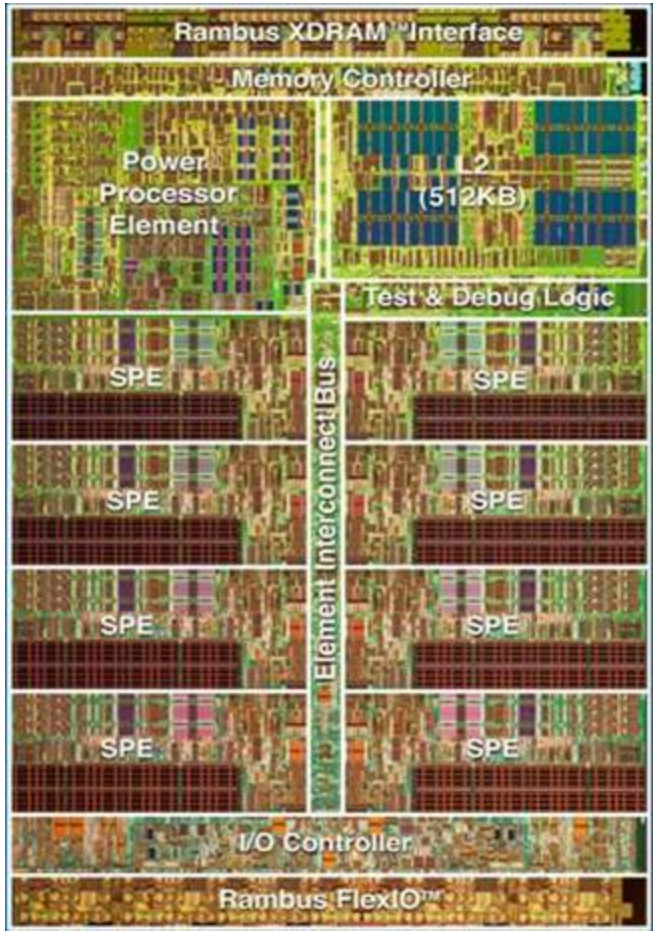
- **Explicitly "orchestrate" the data flow between main memory and each SPE's local store**
  - Use the Memory Flow Controller (MFC) to post DMA commands to gather & scatter arbitrary segments of memory between main memory and local store
  - Enables detailed programmer control of data flow
    - Get & Put the data when & where you want it
    - Extremely powerful mechanism to efficiently support streaming applications
    - Enables latency hiding: Perform reads, writes & computes simultaneously
    - Critical when data access patterns are not easily predictable
  - Avoids restrictive HW cache management
    - No need to rely on (potentially inefficient) HW cache mechanisms
    - HW cache management unlikely to determine optimal data flow
  - Allows more efficient use of the existing bandwidth
- **BOTTOM LINE:**

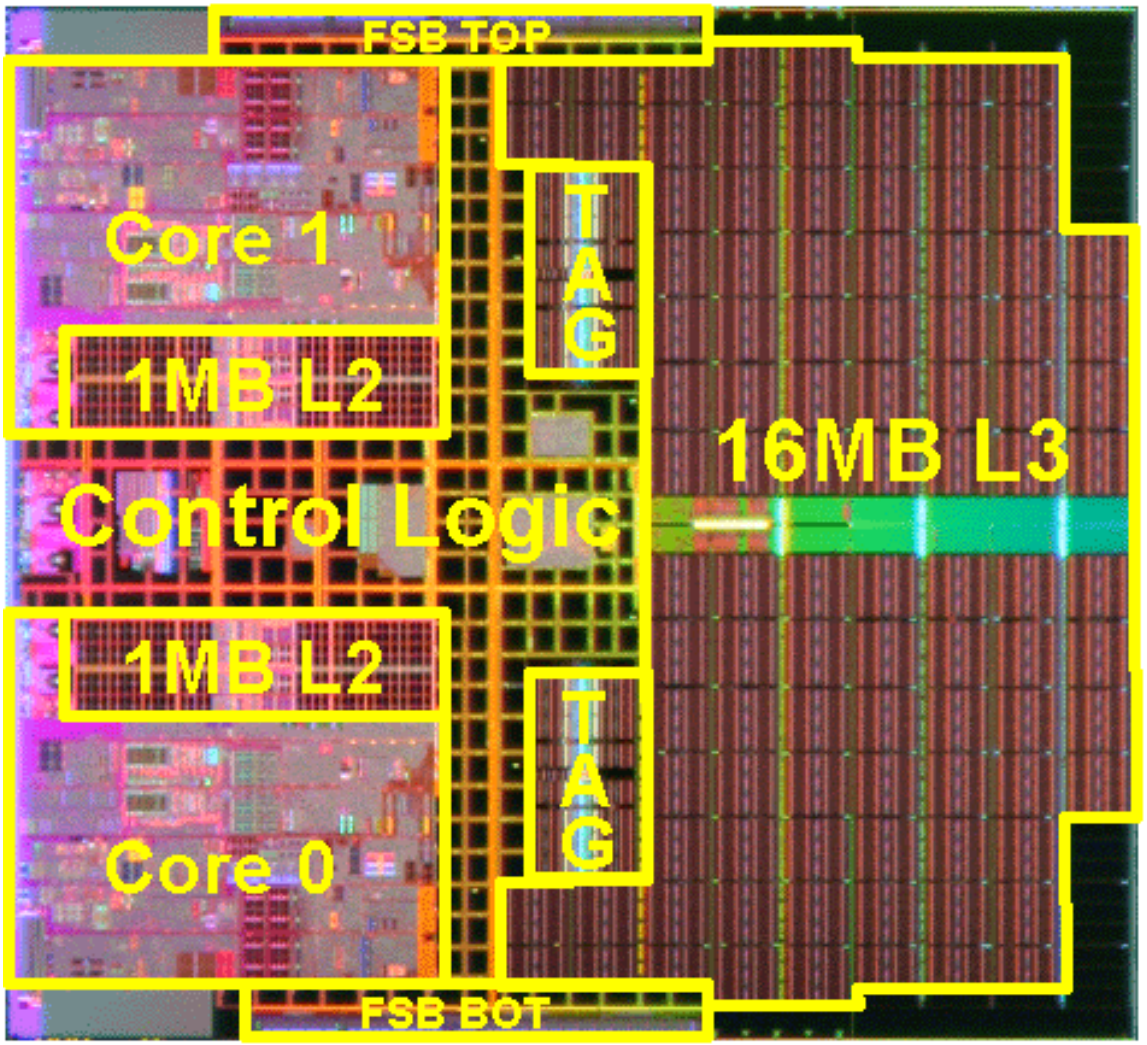
It's all about the data!

© 2007 IBM Corporation

Cell Comparison: ~4x the FLOPS @ ~½ the power

Both 65nm technology
(to scale)

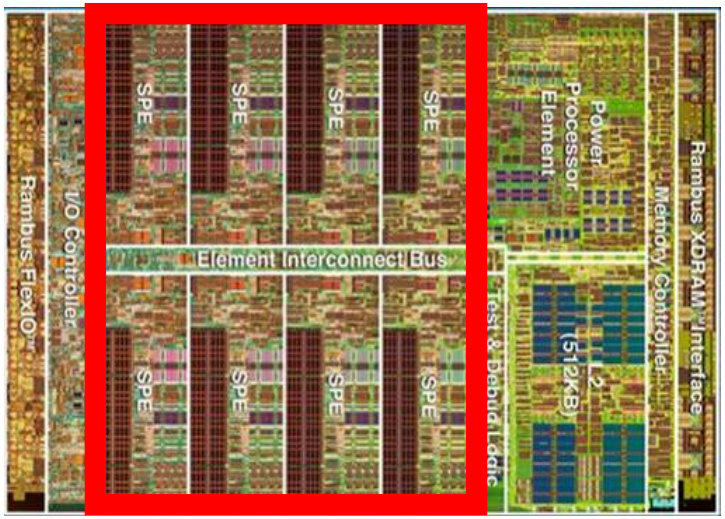




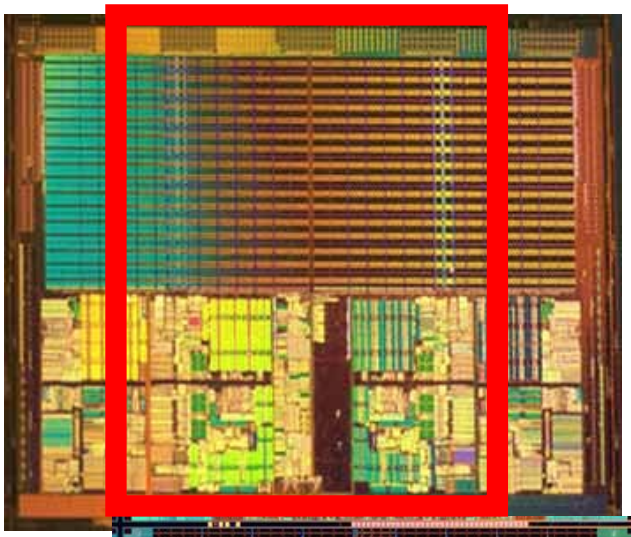
© 2007 IBM Corporation

Memory Managing Processor vs. Traditional General Purpose Processor

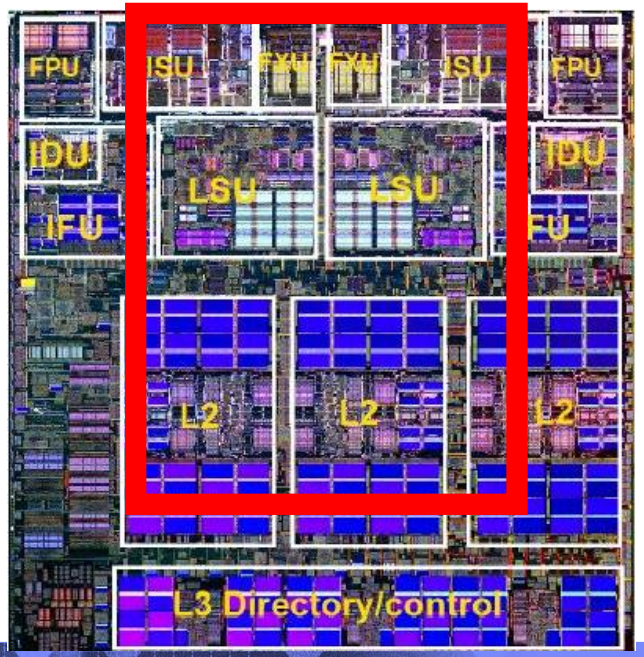
Cell BE



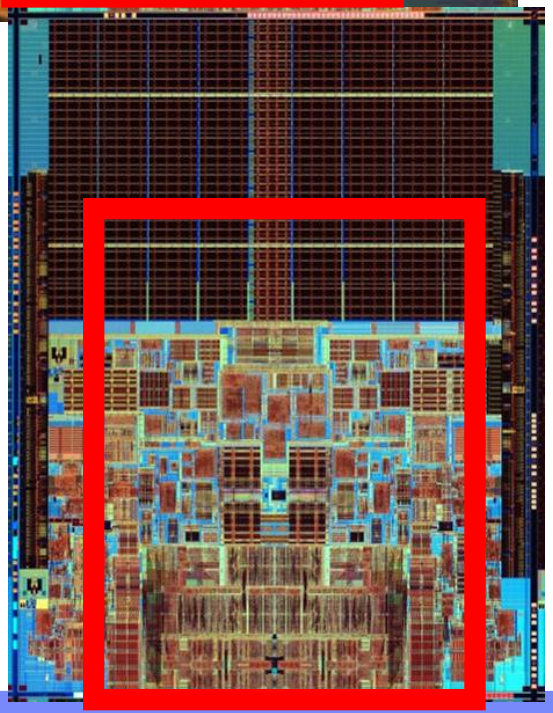
AMD



IBM

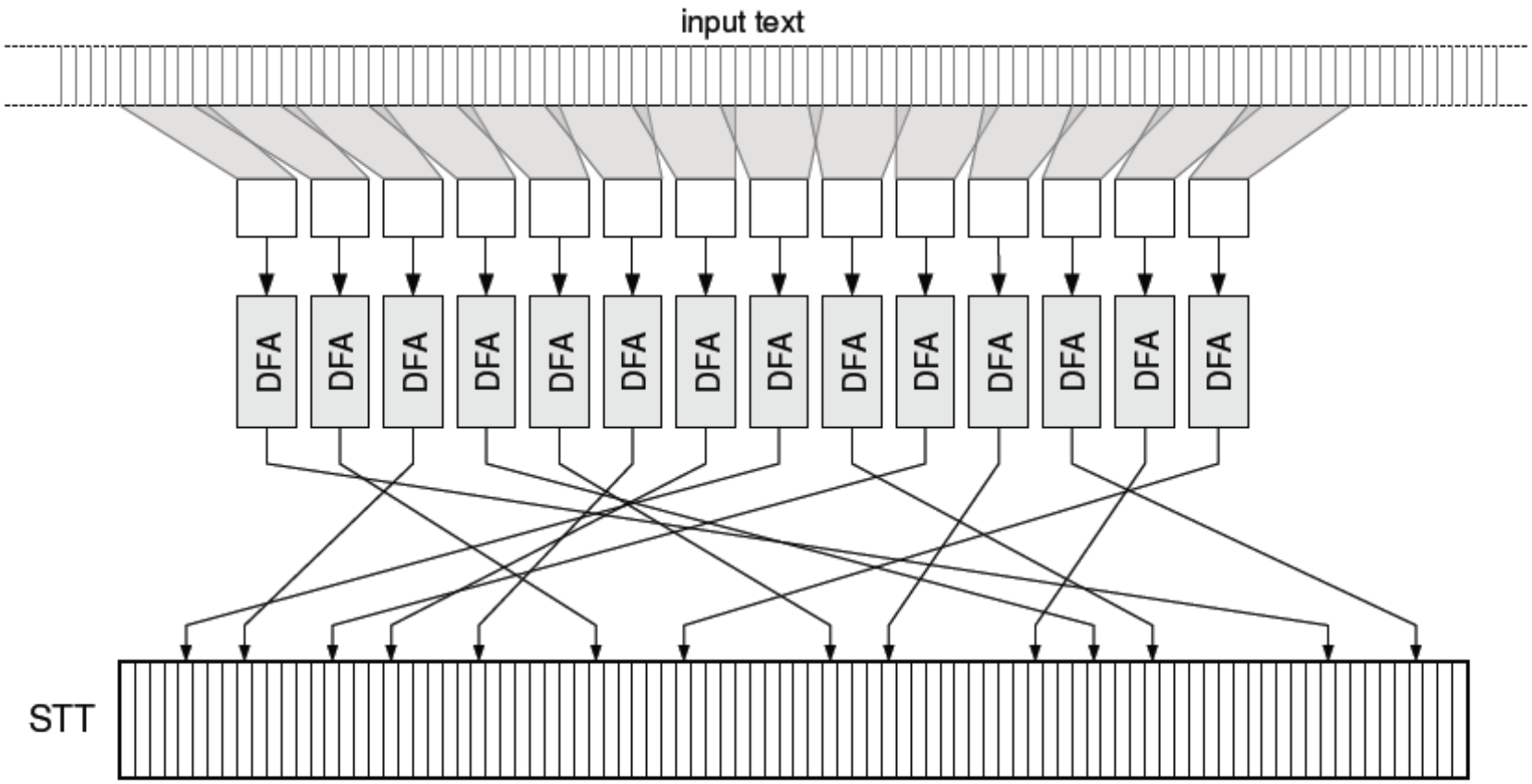


Intel



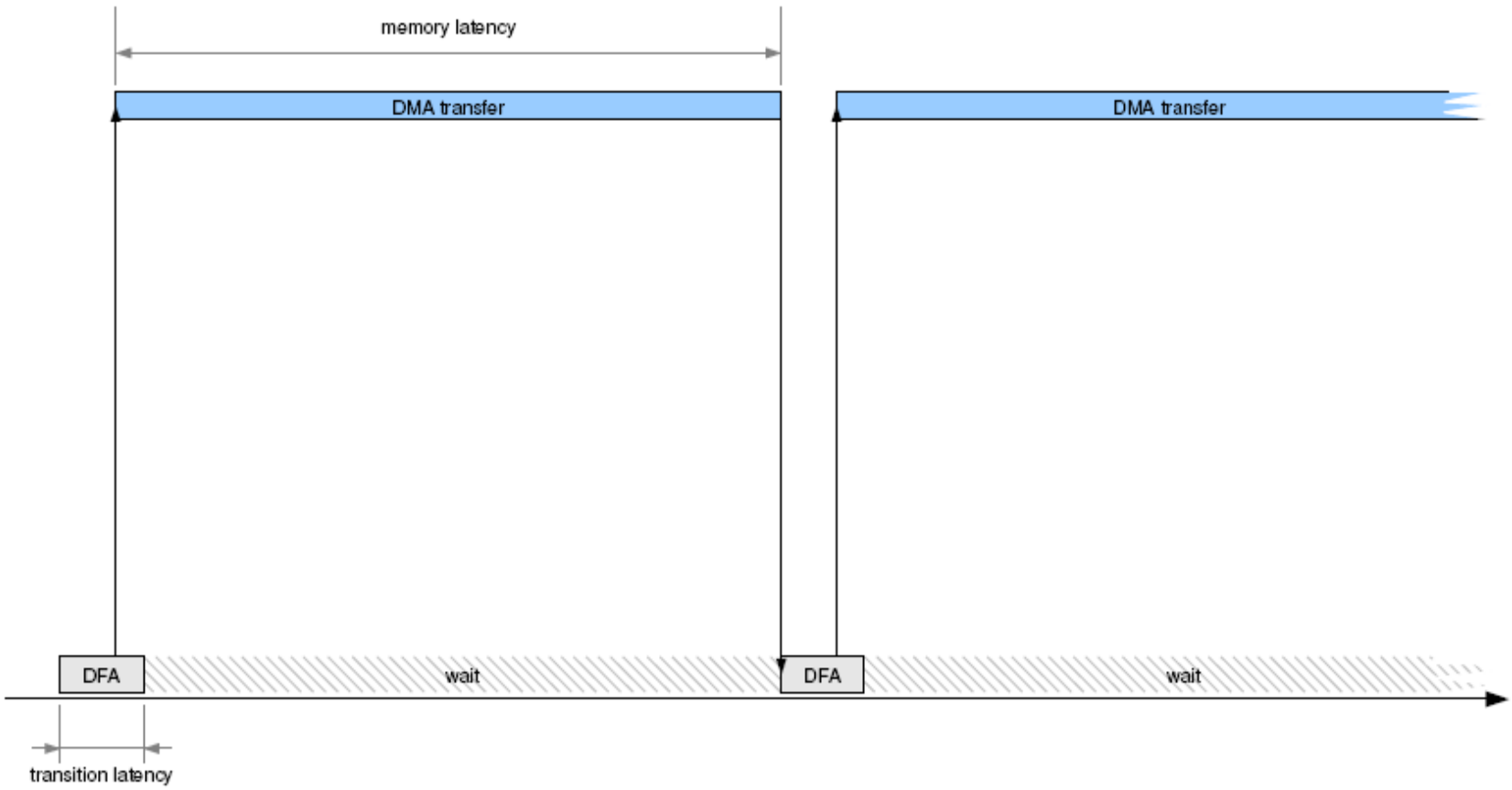
© 2007 IBM Corporation

EXAMPLE: Fine Grained Streaming Applications



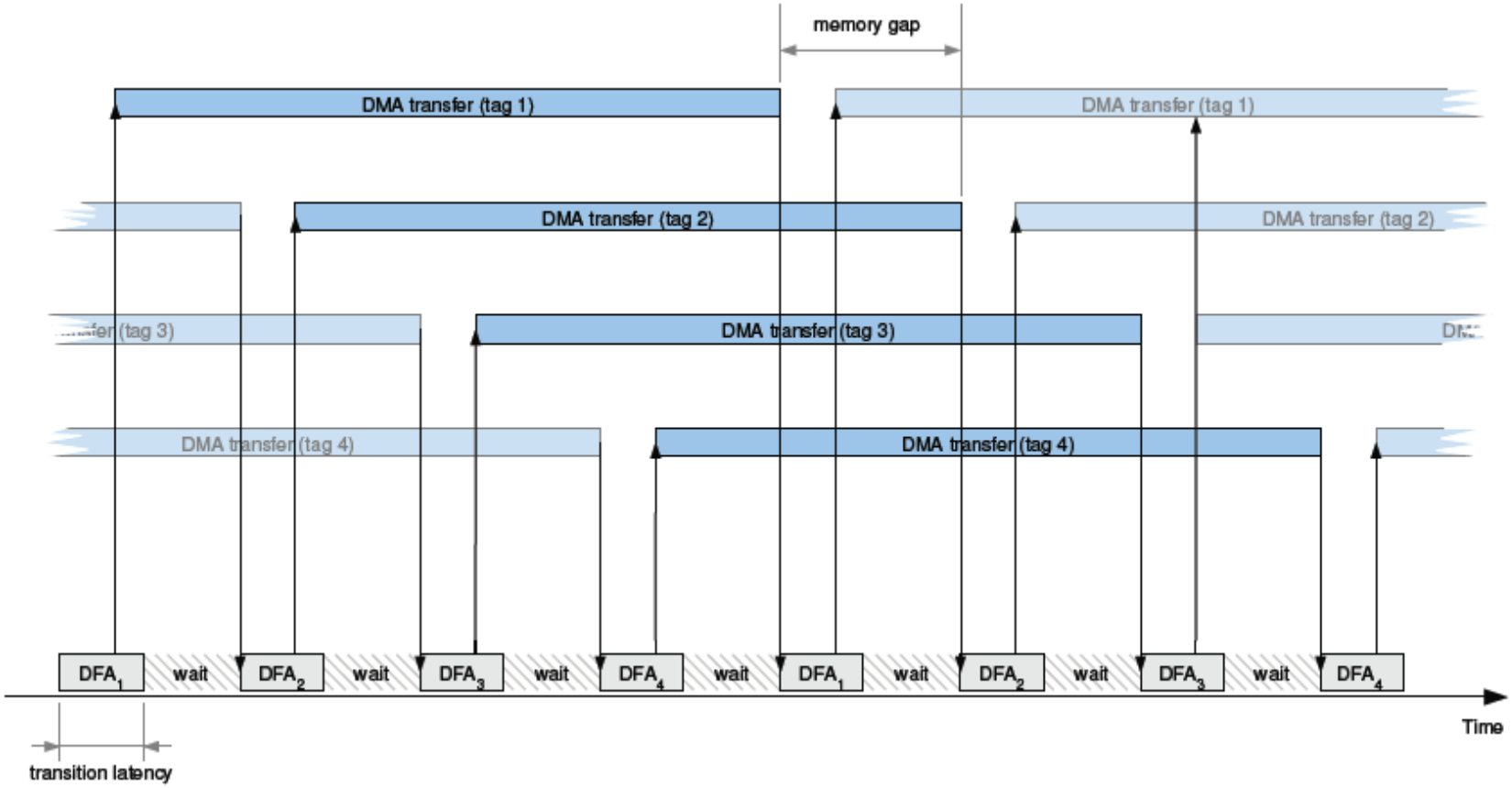
© 2007 IBM Corporation

Hiding Main Memory Latency



© 2007 IBM Corporation

Fighting the Memory Wall



© 2007 IBM Corporation

Back Up Slides

- **Cell Market focus areas**
- **Cell programming tool strategy**
- **Cell BW**
- **SIMDization**

© 2007 IBM Corporation

Multiple focus areas

Extreme Stream Computation and Bandwidth requirements

Real-time Analytics

Processing of Data
Information Synthesis
Analysis

Image/Video Creation/Mgt

Presentation of Data
Visualization
Imaging

Unstructured Data

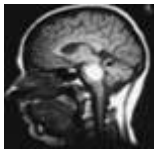
Multimodal Search
Data Transforms
Pattern Matching

Market & Solution Specific Assets

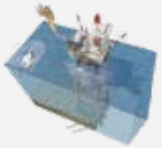
Digital Media

Home Media Consumer Electronics

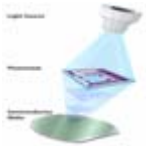
Financial Services Sector



Information Based Medicine



Chemicals & Petroleum



Electronic Design Automation

Digital Video Surveillance

Aerospace and Defense

© 2007 IBM Corporation

Cell Broadband Engine™(Cell/B.E.) Processor Tools Strategy

Decreasing level of attention by coder to architecture details

Bare metal programming | **Frameworks and Libraries** | **Complete HW Isolation**

- Communication & Synchronization Mechanisms(DMAs, Mailboxes, Signals)
- SPU Intrinsics, VMX
- H/W Decrementors
- Performance Counters

IBM

- ALF
- BML
- DaCS
- Overlays for Code partioning
- OpenMP
- Software Managed Cache

3rd Party Libaries

- **Mercury MCF**
- **CodeSourcery**

Segment Specific/Extended Libraries

- Mercury (SAL, PAS, ...)
- IBM FFT
- FFTW
- BLAS
- H.264
- JPEG
- MPEG
- Image Processing
- Vector Math Library
- MASS Library for SPU
- MASSV Library for PPU/SPU

Minimal knowledge of h/w needed !

High Level Application
1. Capture computations
2. Stream execution
3. Data and task parallelism

RapidMind, Gedae
[Development platforms]

Maps onto all computational resources

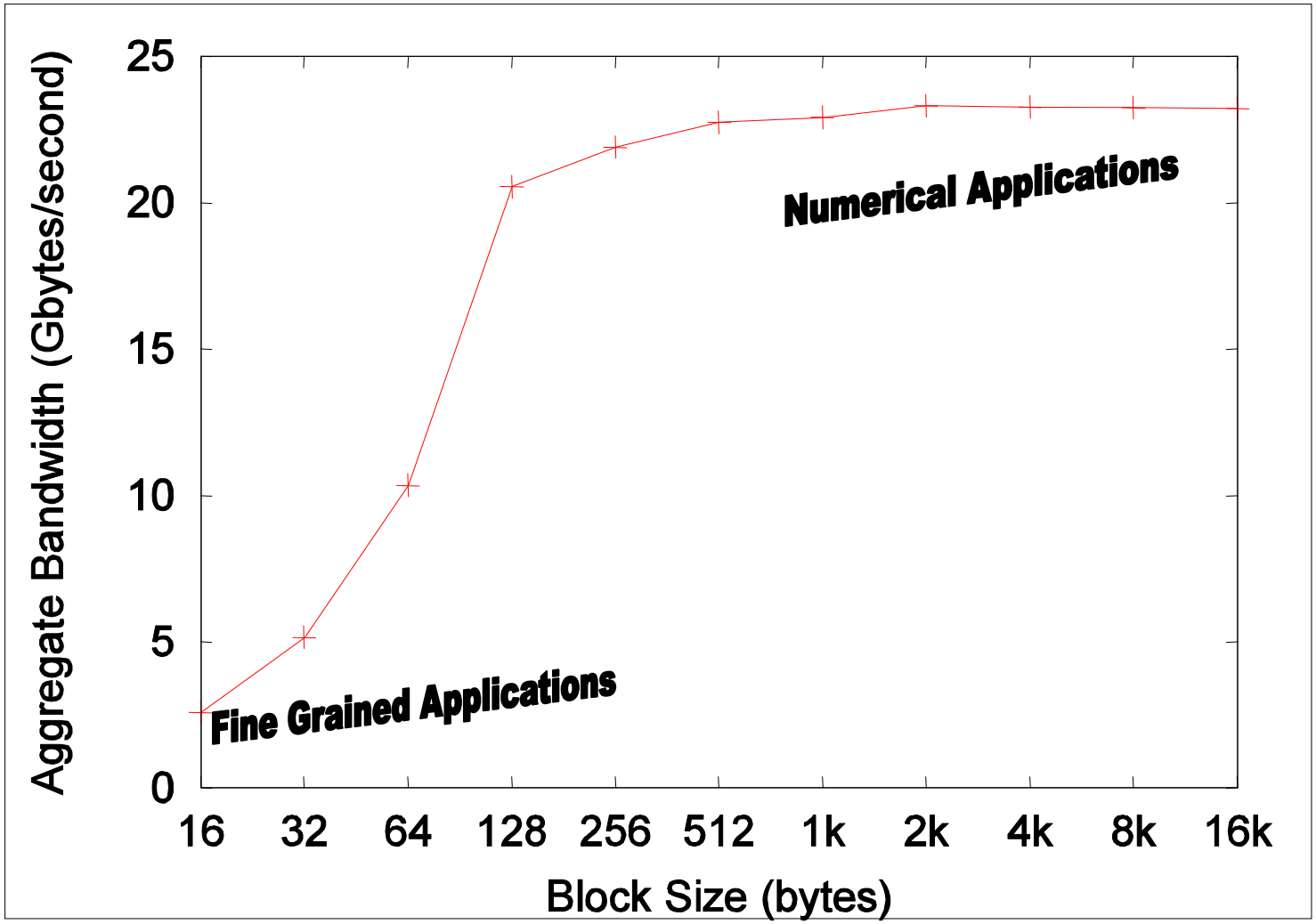
Multi-Core CPUs, Cell/B.E., GPU

3rd Party Tools	**Wind River, Mercury Tooling (TATL, Assembly, etc), Allinia, Mentor Graphics**
IBM Enterprise Tools	Cell/B.E. System IDE (Code Builder for heterogeneous PPE, SPE, Debugger, Dynamic Profiling Analysis, Static Timing Analysis) Performance Analysis (Visualization tools, Bandwidth, Latency, Lock analyzers), Performance debug tools
Base Tools	Critical RAS features, XLC/C++, GCC, Fortran 11.1, Cell/B.E. System Simulator, **CMPWare**, etc..
Environment	Red Hat Fedora Core (5, 6, 7…), Red Hat Enterprise Linux, Sony GameOS, **Windriver Linux and VXWorks, Mentor Graphics**

3rd Party/ISV Offering

© 2007 IBM Corporation

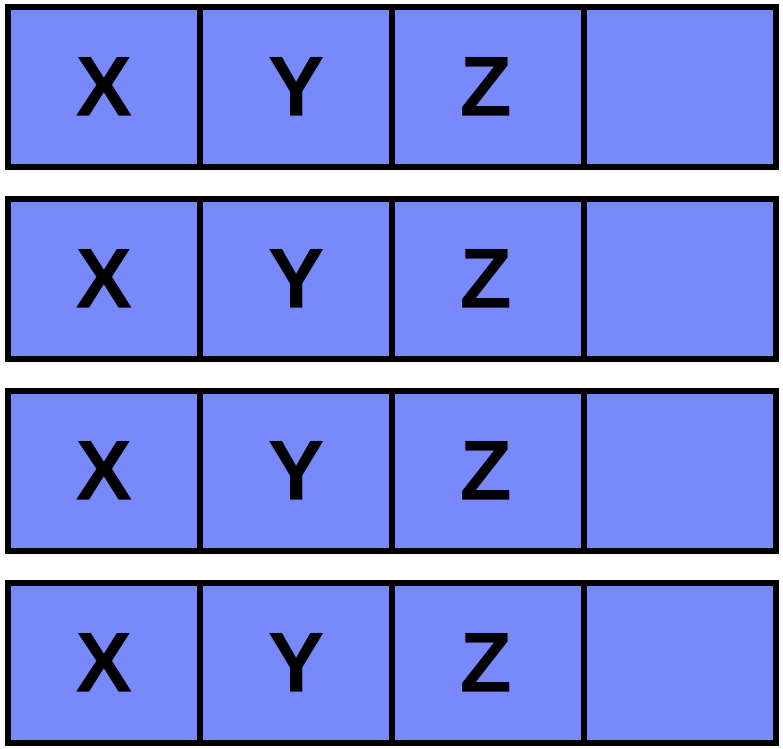
Aggregate Main Memory Bandwidth



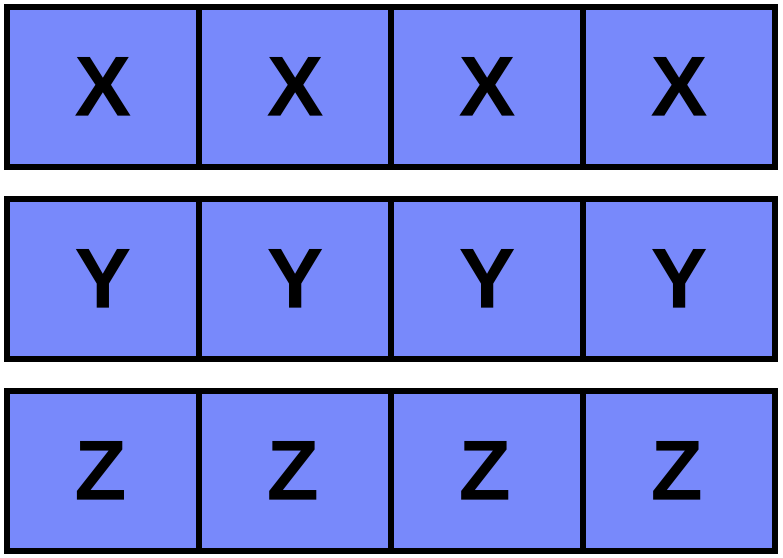
© 2007 IBM Corporation

AOS vs SOA Vectors

Array of Structures



Structure of Arrays



© 2007 IBM Corporation