



Computing Environment for Many-core

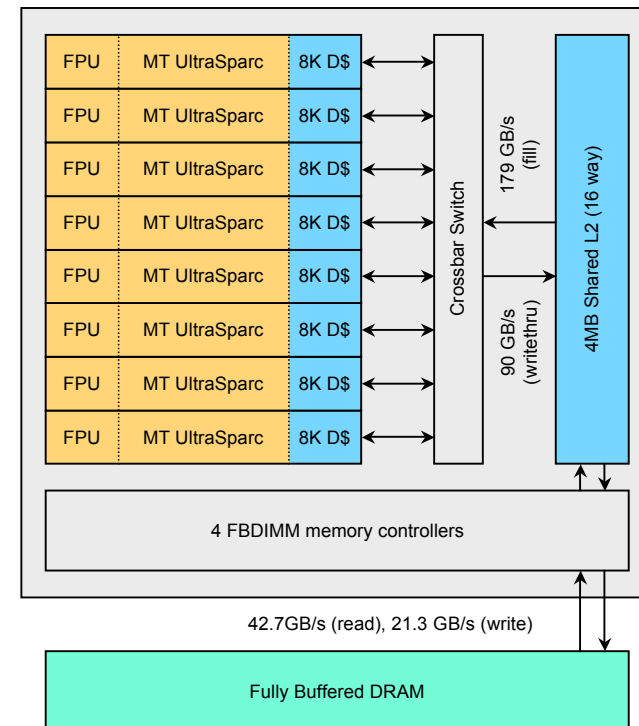
Kunle Olukotun
Pervasive Parallelism Laboratory
Stanford University

Multicore Days 2008 Stockholm, Sweden



Many-core Today

- The many-core chip
 - 8 cores
 - 64 threads
 - Specialized resources

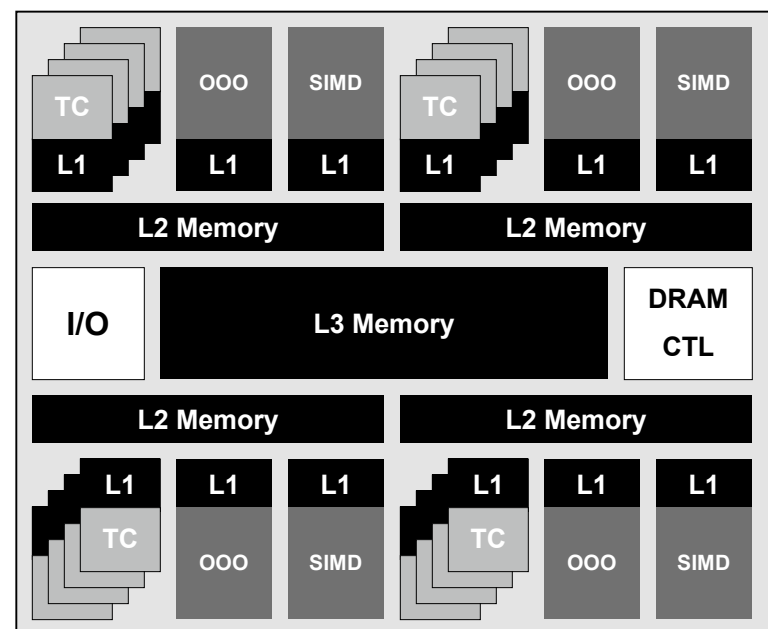


Sun Niagara2



Many-core Future

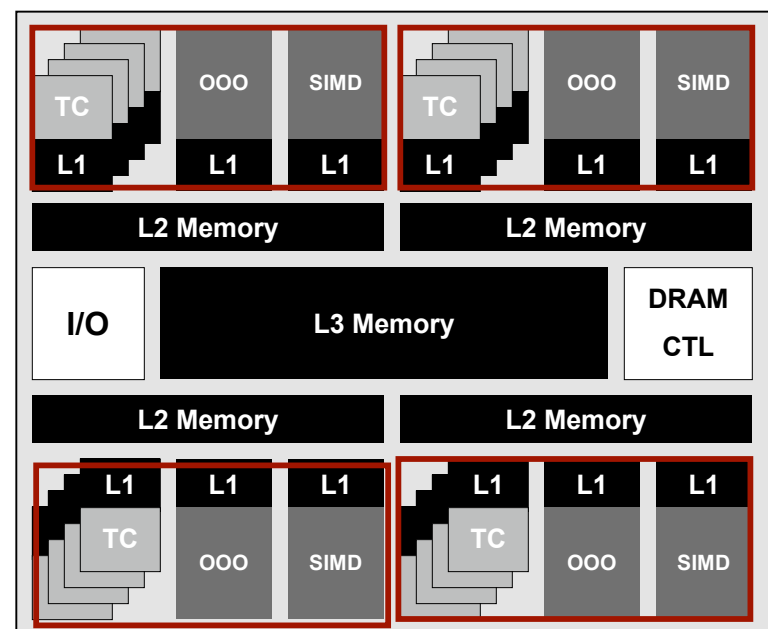
- The many-core chip
 - 100s of cores
 - OOO, threaded, & SIMD
 - Hierarchy of shared memories
 - Scalable, on-chip network (hierarchy of buses)





Many-core OS

- Resource sharing in an era of plentiful resources
 - Space sharing





Programming Many-core

- **Near term**
 - Java and libraries or C and OpenMP
 - Not satisfactory for average programmers

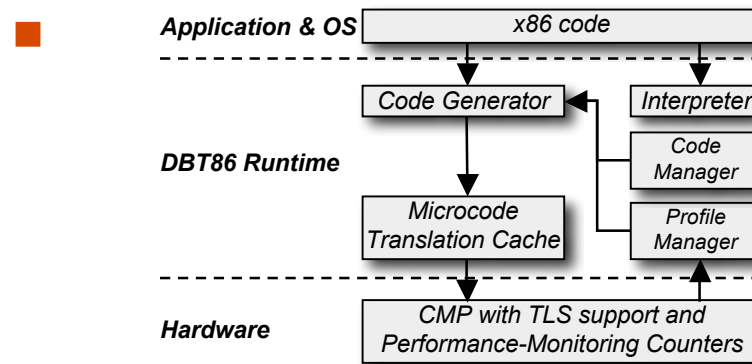
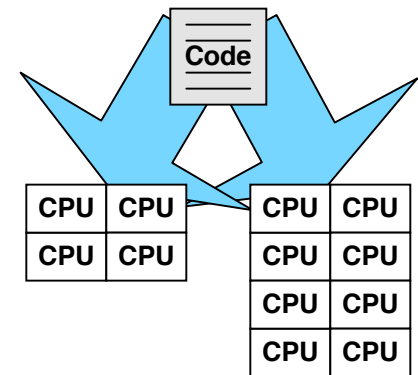
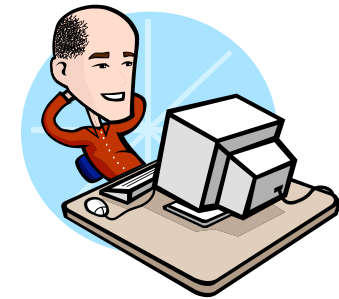
- **DSLs ⇒ higher productivity for developers**
 - High-level data types & ops tailored to domain
 - E.g., relations, triangles, matrices, ...
 - Express high-level intent without specific implementation artifacts
 - Programmer isolated from details of specific system

- **DSLs ⇒ scalable parallelism for the system**
 - Declarative description of parallelism & locality patterns
 - E.g., ops on relation elements, sub-array being processed, ...
 - Portable and scalable specification of parallelism
 - Automatically adjust data structures, mapping, and scheduling as systems scale up



Legacy Code

- **Java: Dynamic parallelizing compiler**
 - Free programmer from manual identification of fine-grained parallelism
 - Automatically change decompositions as CMP HW evolves to more CPUs and larger caches
 - Adapt thread decompositions to different datasets
 - Java JRPM speedups: 3-4 on floating-point, 2-3 on multimedia, 1.5-2.5 on integer
- **Binary Translation: DBT86**



1.6x on SPECint using 4 CPUS