



NSF CDI Panel

Arun Rodrigues

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy's National Nuclear Security Administration
under contract DE-AC04-94AL85000.



Ideal Programming Language

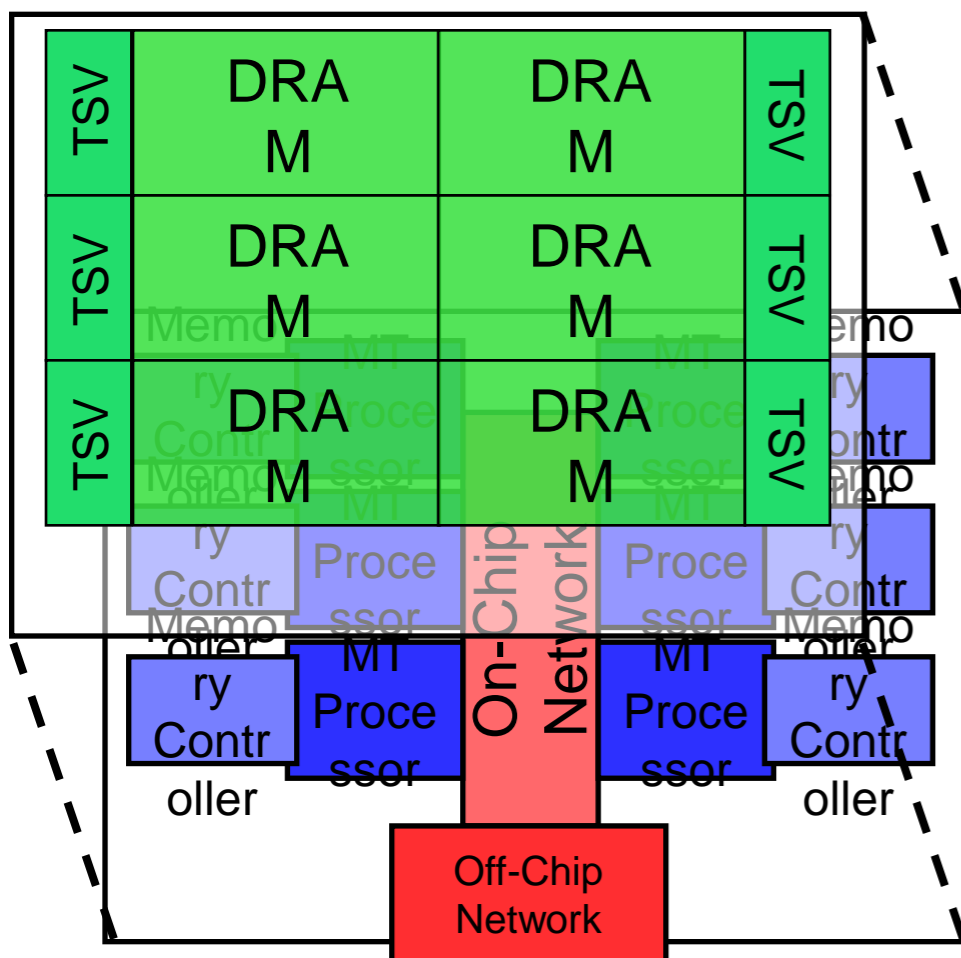
- **No Silver Bullet Language**
- **Two types of language**
 - Low-level
 - Domain-Specific (finite-element, graph, neural net, etc...)
- **Low-level Language (C which knows about big computers)**
 - Parallelism
 - Synchronization
 - Locality: Where data lives, and how to get to it
 - Data distribution
 - Thread migration support (Move the computation to the data)
- **Domain-Specific (Languages which wish they were MATLAB)**
 - May be a library, or thin veneer over low-level
 - Uses the notation best suited to given application



Ideal Machine



- **Advanced packaging (3D Stack, Quilt)**
 - Interchangeable tiers for customization
- **Memory: Close & fast**
 - (10-20 ns random access)
 - Single address space
 - internals exposed to processor
 - “DRAM Aware Processing”
- **Processor: Simple & Multithreaded**
 - Hardware thread support
 - Support for thread synchronization and migration
- **Optical Network**
 - Hardware to accelerate message processing (improve injection rate, offload)
 - Scalable!



How can we write Scalable, Portable, Optimal, Correct Code?

- **How do we keep up with Moore's Law?**
 - Recognize Moore's Law has changed – Cores, not clocks
 - Need to exploit new levels of parallelism
- **Do we need to rethink compilers/interpreters/OSes?**
 - Need Feedback!
 - From Runtime to compiler/programmer: Where are the bottlenecks
 - From programmer to compiler: Generic hints
- **Do we need to rethink OSes?**
 - Smaller, gets out of the way
 - Modular: Minimal set of services

How do we guarantee reproducible correct computational experiments?

- **We Don't.**

- Real experiments are never completely reproducible or correct
- Bigger machines **will** fail frequently

- **Need to...**

- ...bound & identify error
- ...create fault-tolerant algorithms
- ...learn from mainframes (virtualization, data “evacuation”)

	Red Storm	Petascale	Trans-Peta
“Nodes”	13,000	65,000	325,000
Node MTBF	250,000	500,000	750,000
MTBF (hours)	19.7	8.2	2.8