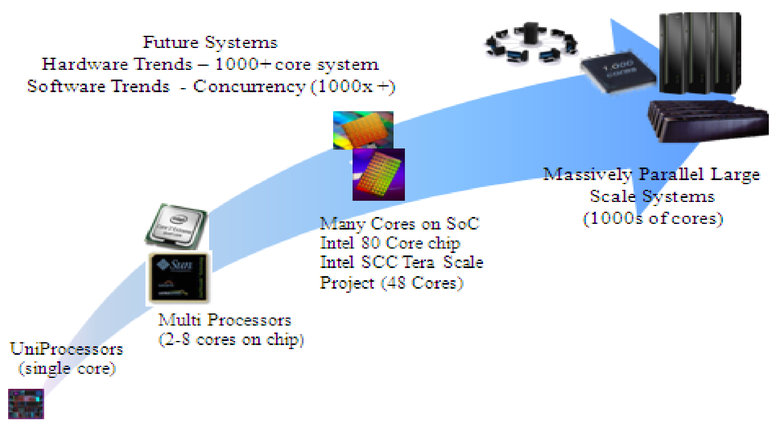
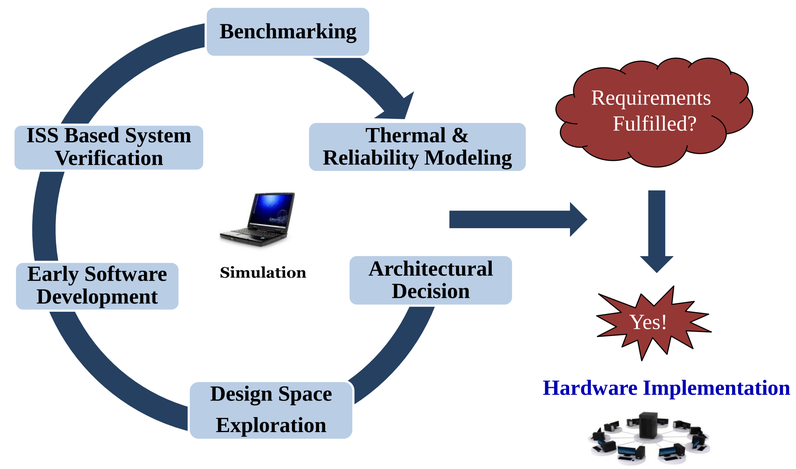
GPGPU-Accelerated Instruction Accurate and Fast Simulation of Thousand-core Platforms

Shivani Raghav , Christian Pinto, Martino Ruggiero, Andrea Marongiu, David Atienza and Luca Benini
Ecole Polytechnique Federale de Lausanne (EPFL) - ESL, CH and Università di Bologna – DEIS, IT

Motivation





Requirements

- ✓Scalable (up to 1000s of cores)
- ✓High Performance (1000s of MIPS)
- ✓Modular (Simple Customization)
- ✓Cost Effective (Academic)
- ✓Debugging (Fast time to market)
- ✓Credible (Reproducible)
- ✓Re-targetable (Imitate many ISAs)

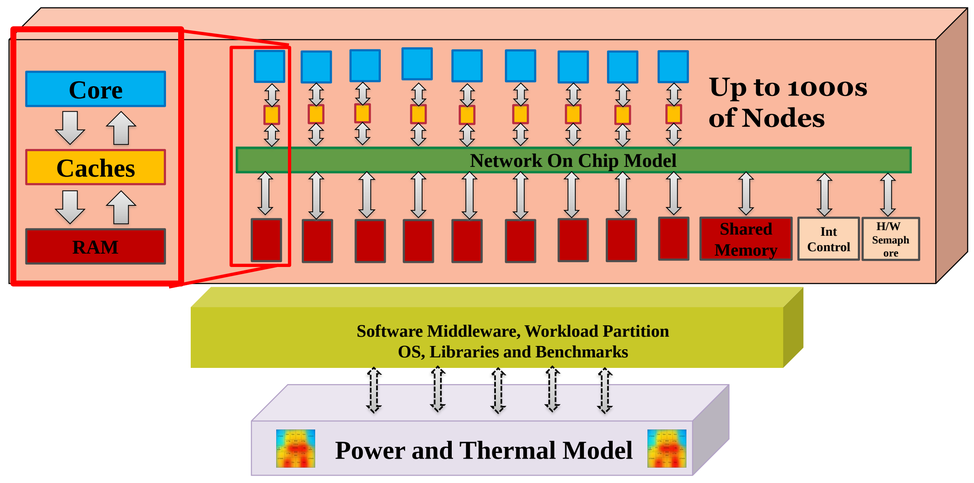
Goal : To develop a novel, parallel and scalable high performance simulator for large many-core platforms

Related Work

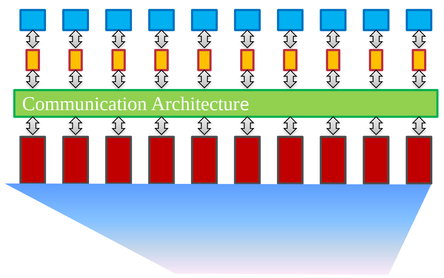
Methodology	Tool	Drawbacks
Cycle Accurate	GEMS[1], PTLSim[2], etc	Very Slow up to just tens of cores!!
Functionally Accurate	OVPSim[3]	Slow and Low scalability
Statistical Sampling	Cotson[4]	Low Accuracy, Slow
Distributed Host	Graphite[5]	Expensive ($$$) Low Manageability
FPGA Acceleration	RAMP Project[6]	Slow Development Cycle

Most current system simulators are slow or have poor scalability
With many cores (1000+), simulation time increases exponentially

Vision of 1000 cores system simulation



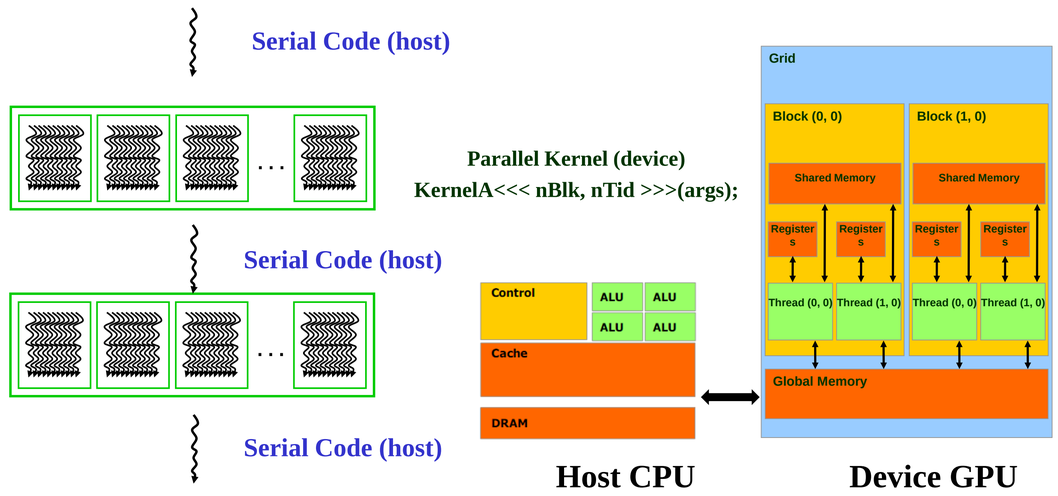
Proposed Solution : GPGPU as host Platforms



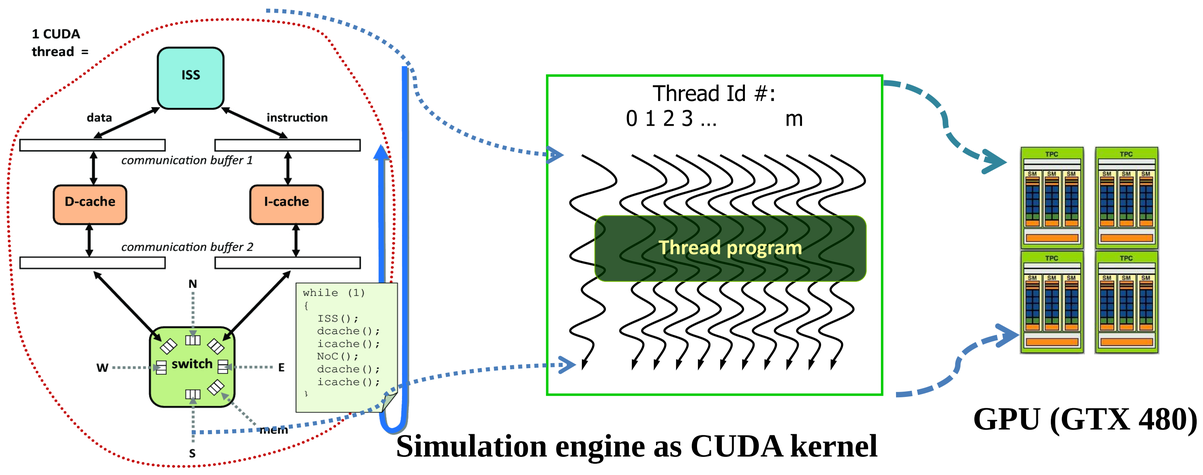
- ❖ Massively parallel, Many core architecture
- ❖ Multithreaded (thousands CUDA of threads)
- ❖ Cost Effective
- ❖ Easily available
- ❖ CUDA Programming Model

- Virtual many-core architecture Simulation
- HW many-core architecture of GPU host

GPU Architecture and CUDA Programming



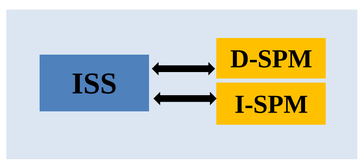
Implementation



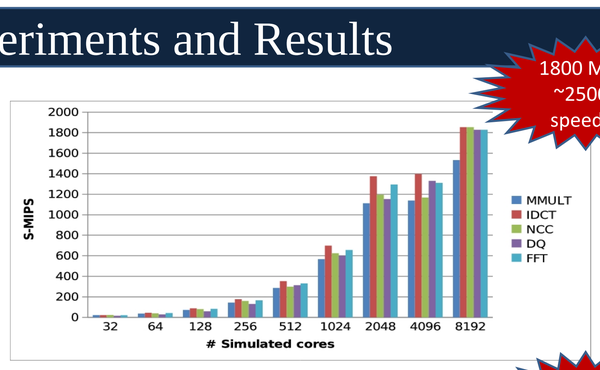
Code Optimizations

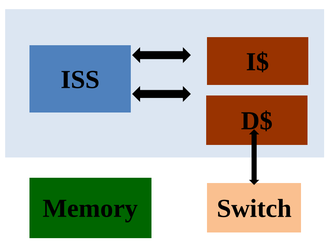
- ❑ **GPUs – SIMT (Single Instruction Multiple Thread)**
  - ❑ Mapping MIMD application to SIMT architecture.
  - ❑ Partitioning the Computation to keep the GPU processors busy.
  - ❑ Maximizing independent parallelism. Many threads, many thread blocks
- ❑ **Slow Device Memory**
  - ❑ High latency of Device Memory (400-600 Clock cycles)
  - ❑ Right Access Patterns are crucial. Optimizing Memory Access for Coalescing
  - ❑ Exploiting on chip Shared Memory (~100 times faster than global memory)
  - ❑ Minimizing shared memory Bank Conflicts
- ❑ **Control Flow Divergence hurts Performance**
  - ❑ Thread scheduled in group of 32, Warp
  - ❑ Conditional Braches in warp cause serialization Avoiding small branch-granularity

Experiments and Results

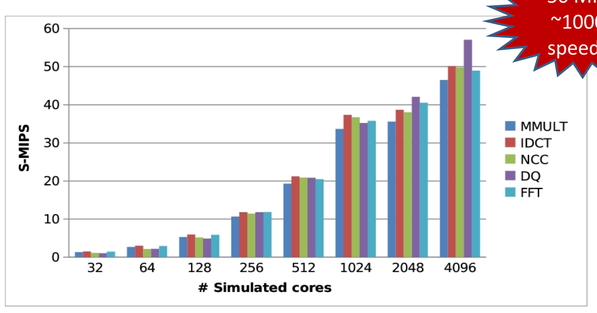


Architecture 1





Architecture 2



Conclusion and References

- **A novel approach for simulation of many core platform using high computational power and parallelism of GPGPUs.**
- **Results show very fast simulation, good accuracy and high scalability for up to 1800 MIPS from simulation of up to 8K cores.**

[1] M. M. K. Martin et al. “Multifacet’s general execution-driven multiprocessor simulator (gems) toolset,” *SIGARCH Computer Architecture News*, 2005
[2] Yourst, M.T.; , "PTLsim: A Cycle Accurate Full System x86-64 Micro architectural Simulator," Performance anlysis of Systems & Software, 2007. ISPASS 2007. IEEE International Symposium on , pp.23-34, April 2007
[3] The Open Virtual Platforms (OVP) portal, http://www.ovpworld.org/.
[4] E. Argollo et al., “Cotson: infrastructure for full system simulation,” *Operating Systems Review*, vol. 43, no. 1, pp. 52–61, 2009.
[5] N. Beckmann et al. “Graphite: A Distributed Parallel Simulator for Multicores,” MIT, Tech. Rep., November 2009.[Online]. Available: http://dspace.mit.edu/handle/1721.1/49809
[6] Z. Tan et al. “Ramp gold: An fpga-based architecture simulator for multiprocessors.”