HPC System Architecture & Design Tradeoffs—

A Proposed Quantitative approach

Shekhar Borkar
Intel Corp.
Aug 13, 2015

Acknowledgment: David Scott, Josh Fryman, Dave Dunning

Modsim WS 2015

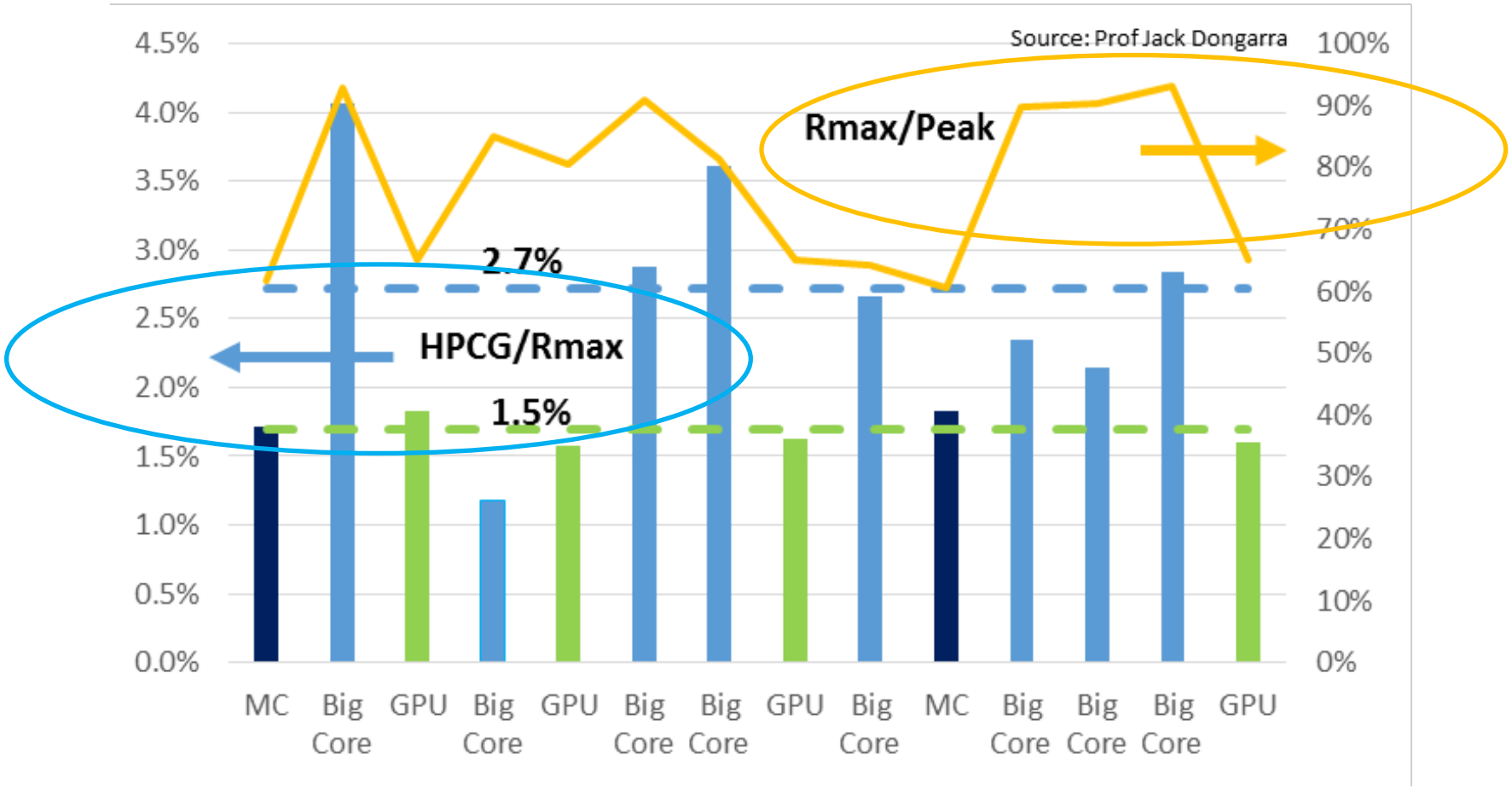
This research was, in part, funded by the U.S. Government, (DOE). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

Outline

- The problem…
- Challenges modeling future arch & design to avoid the problem
- A simple, but most certainly a controversial proposal
- Summary

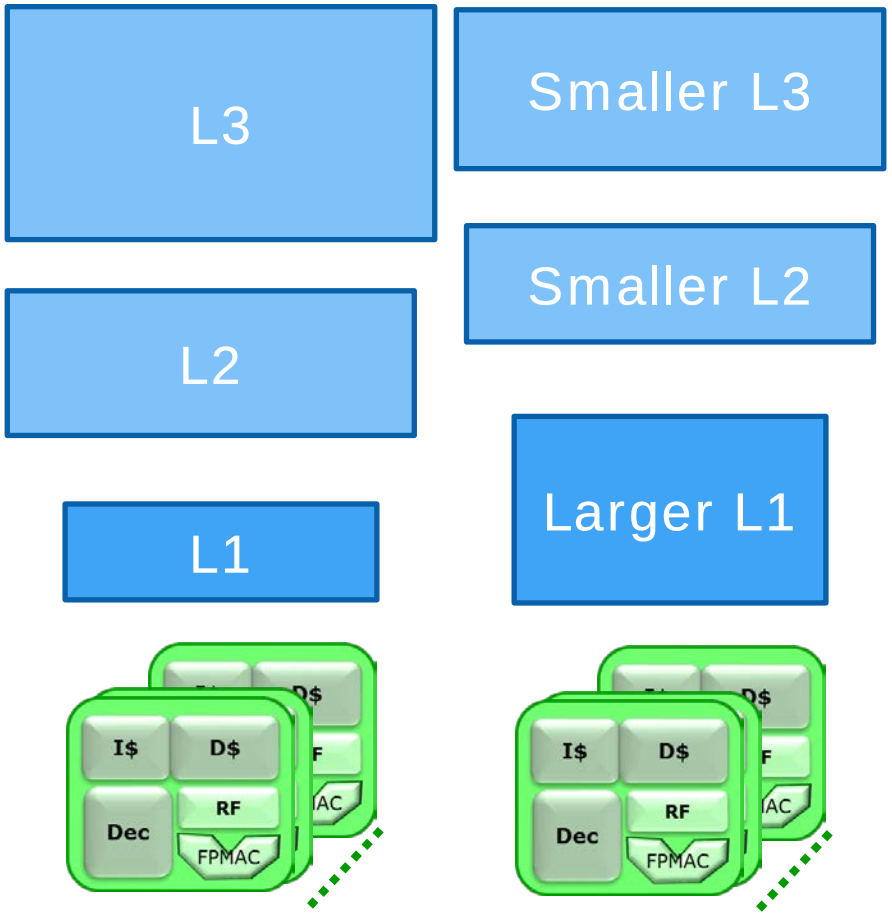
System Efficiency

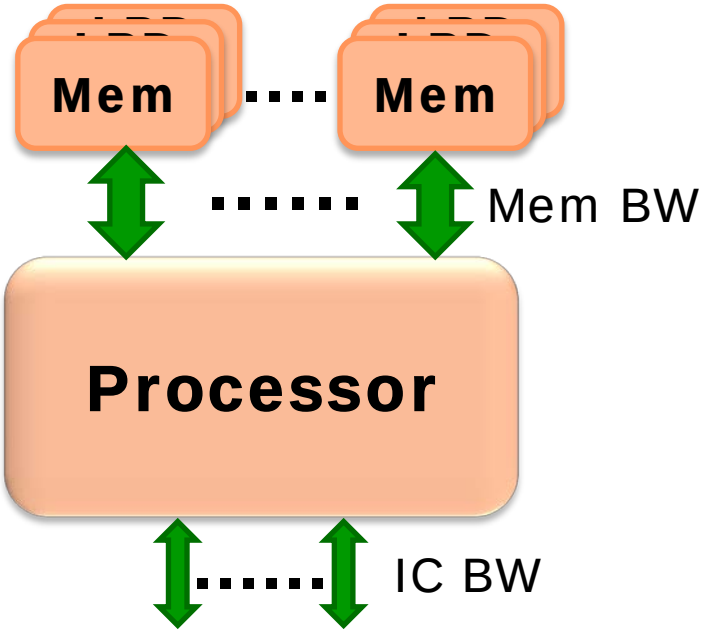
Using HPCG as an example representative workload



Architecture tuned for Linpack / DGEMM
Challenge: 20+% system efficiency

Node Arch & Design Tradeoff Examples





Larger L1 has higher latency
Reduced sharing in L2/L3
Which one is better?

Same number of pins
Trade-off Memory BW vs IC BW
Which one is better?

What about Proxy Apps?

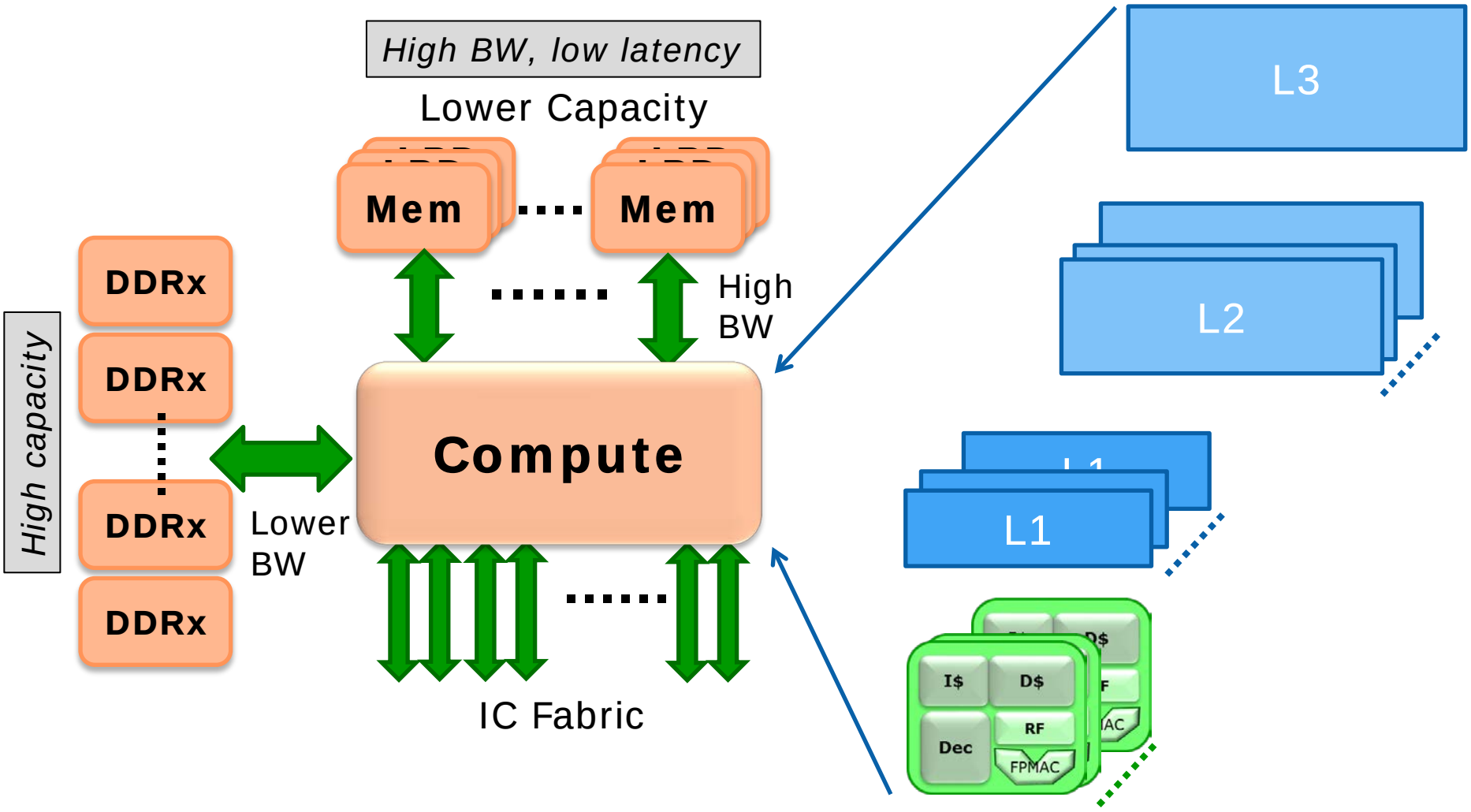
- Capture the essence of real apps
- Model behavior of the real apps
- May give wrong results—it's ok!
- Smaller, but not small enough!
- Simulator ~ 10 MIPS
- Proxy app run time ~ 4-12 days

Proxy apps are good for benchmarking
Not nimble enough to evaluate design tradeoffs

What is needed?

- **A framework simpler than proxy apps**
- **Goal:**
  - Optimize architecture for real applications
  - What-if analysis of the features
  - Understand incremental impact, not absolute
  - Understand and make right architectural & design tradeoffs
- **Non-goals:**
  - Estimate application performance
  - Benchmarking

Primary Components



Proposed Framework

$$\text{Application performance} = f\{\text{Compute (node)}, \text{Mem BW}, \text{Mem Latency}, \text{IC BW}, \text{IC Latency}\}$$

- Five SMALL synthetic kernels
- Each stresses one attribute (compute, MBW, …etc.)
- Independent of the other attributes (almost)

First order—assume linearly independent relationship

Performance = Ax(compute) + Bx(MBW) + Cx(ML) + Dx(ICBW) + Ex(ICL)

- Measure performance of each kernel on five machines
- Measure performance of a proxy app on the same five
- Determine A, B, C, D, E
- Simulate kernels on the proposed architecture
- Determine performance delta with features
- Tweak architecture, and iterate

Characteristics of the Kernels

Compute	DGEMM based Accessed data stored in the node processor (not DRAM) Minimal external memory and IC access
Mem BW	Stresses node memory hierarchy up to DRAM Move large arrays to/from DRAM Minimal compute and IC accesses
Mem Latency	Stresses node memory hierarchy up to DRAM Random accesses, in the processor memory and DRAM Minimal compute and IC accesses
IC BW	Stresses IC hierarchy, on the node and across the system Move large arrays, synthetic, not to/from DRAM Minimal compute and memory accesses
IC Latency	Stresses IC hierarchy, on the node and across the system Move small arrays, randomly, synthetic, not to/from DRAM Minimal compute and memory accesses

Summary

- Proxy apps are good for benchmarking
- Not for early architecture & design tradeoffs
- We propose a simple framework
- Invaluable to perform “what-if” analysis
- Who wants to help?