SoC for HPC Workshop Overview

John Shalf and James Ang

LBL/SNL Computer Architecture Laboratory

Denver, Colorado
August 26, 2014
The HPC-SoC workshop will focus on semi-custom, application-targeted designs, and server processing for HPC and data-centers.

The goal is to develop a strategy for a robust open ecosystem for SoC designs that serve the needs of energy efficient HPC applications for multiple government agencies.
HPC Market Overview

High End Systems (>$1M)
- Most/all Top 500 systems
- Custom SW & ISV apps
- Technology risk takers & early adopters

IDC:
- 2005: $2.1B
- 2010: $2.5B

Volume Market
- Mainly capacity; <~150 nodes
- Mostly clusters; >50% & growing
- Higher % of ISV apps
- Fast growth from commercial HPC; Oil & Gas, Financial services, Pharma, Aerospace, etc.

IDC:
- 2005: $7.1B
- 2010: $11.7B

Total market >$10.0B in 2006
Forecast >$15.5B in 2011

HPC is built with of pyramid investment model

IDC Segment
System Size  | 2005  | 2010  | CAGR
-------------|-------|-------|-----
$250K-$1M   | $1.9B | $3.4B | 11.8%
$50K-$250K  | $2.9B | $4.9B | 10.7%
0-$50K      | $2.2B | $3.4B | 9.6%

Mark Seager LLNL
HPC Market Overview

IDC:
2005: $2.1B
2010: $2.5B

Volume Market
Mainly capacity; <~150 nodes
Mostly clusters; >50% & growing
Higher % of ISV apps
Fast growth from commercial HPC; Oil & Gas, Financial services, Pharma, Aerospace, etc.

Total market >$10.0B in 2006
Forecast >$15.5B in 2011

HPC is built with of pyramid investment model

IDC Segment System Size
<table>
<thead>
<tr>
<th>Size</th>
<th>2005</th>
<th>2010</th>
<th>CAGR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$250K-$1M</td>
<td>$1.9B</td>
<td>$3.4B</td>
<td>11.8%</td>
</tr>
<tr>
<td>$50K-$250K</td>
<td>$2.9B</td>
<td>$4.9B</td>
<td>10.7%</td>
</tr>
<tr>
<td>0-$50K</td>
<td>$2.2B</td>
<td>$3.4B</td>
<td>9.6%</td>
</tr>
</tbody>
</table>

Mark Seager LLNL

Totally Bogus Prediction
IDC 2010 puts HPC market at
$10B
• **1990s** - R&D computing hardware dominated by desktop/COTS
  - Had to learn how to use COTS technology for HPC
  - Thomas Sterling’s “Beowulf Cluster”

• **2010** - R&D investments moving rapidly to consumer electronics/ embedded processing
  - Must learn how to leverage embedded/consumer processor technology for future HPC systems

**Market in Japan (B$)**

- **PC**
- **TV**
- **DVD**
- **DSC**
- **TV+DVD+DSC**

**IDC Segment**

- **System Size**
  - **Volume Market**
    - Mainly capacity; <~150 nodes
    - Mostly clusters; >50% & growing
    - Higher % of ISV apps
    - Fast growth from commercial HPC; Oil & Gas, Financial services, Pharma, Aerospace, etc.

<table>
<thead>
<tr>
<th>2005</th>
<th>2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1B</td>
<td>2.5B</td>
</tr>
<tr>
<td>$7.1B</td>
<td>$11.7B</td>
</tr>
</tbody>
</table>

Total market >$10.0B in 2006
Forecast >$15.5B in 2011

**CAGR**

<table>
<thead>
<tr>
<th>0-$50K</th>
<th>10.7%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$4.9B</td>
<td>$2.9B</td>
</tr>
</tbody>
</table>

| $50K-$250K | 11.8% |
| $3.4B      | $1.9B |

| $250K-$1M  | 9.6%  |
| $3.4B      | $2.2B |
IDC 2010 Market Study
Embedded market is too large to ignore

Worldwide Intelligent Systems Unit Shipments Comparison - Embedded Systems vs. Mainstream Systems 2011 Share and Growth


Notes:
Size of bubble equals 2011 share of system shipments. Growth of cell phone system shipments is driven by smartphones and multi core processor designs.
Consumer Electronics is the NEW Driver (and surprisingly aligned with HPC needs)

• High Performance embedded is aligned with HPC
  – HPC used to be performance without regard to power
  – Now HPC is power limited (max delivered performance/watt)
  – Embedded has always been driven by max performance/watt (max battery life) and minimizing cost ($1 cell phones)
  – Now HPC and embedded requirements are aligned

• The R&D investments in the embedded ecosystem is too large to ignore (dwarfs the current server market)

• Your “smart phone” is driving technology development
  – Desktops are no longer in the drivers seat
  – This is not a bad thing because high-performance embedded has longer track record of application-driven design
  – Hardware/Software co-design comes from embedded design
  – And its based on Specialization & use of SoC Design
Design Verification Costs

• Design complexity scales linearly (if you are optimistic)
• Verification complexity grows exponentially
• Motivates use of pre-verified commodity IP blocks
  – Verification costs shared by broader market

Source: SIA Roadmap, 2001
Redefining “commodity”

• Must use “commodity” technology to build cost-effective design
• The primary cost of a chip is development of the intellectual property
  – Design and verification dominate costs
  – Design rules make design/verification even harder!
  – Embedded computing has a vibrant market for IP/circuit-design (pre-verified, place & route)
  – Redefine your notion of “commodity”!

The ‘chip’ is not the commodity...

*The stuff you put on the chip is the commodity*
Embedded Design Automation (co-design)
(Using FPGA emulation to do rapid prototyping)

Processor configuration
1. Select from menu
2. Automatic instruction discovery (XPRES Compiler)
3. Explicit instruction description (TIE)

Processor Generator (Tensilica)
Tailored SW Tools:
Compiler, debugger,
simulators, Linux,
other OS Ports
(Automatically generated together with the Core)

Application-optimized processor implementation (RTL/Verilog)
Base CPU OCD
Apps Datapaths Cache Timer
Extended Registers FPU

Build with any process in any fab
Or “tape out” To FPGA

RAMP FPGA-accelerated Emulation of ASIC

Modeling/Simulation is central to CoDesign (and it ain’t new)
Seymour Cray 1977: “Don’t put anything into a supercomputer that isn’t necessary.”

Mark Horowitz 2007: “Years of research in low-power embedded computing have shown only one design technique to reduce power: reduce waste.”
Building an SoC from IP Logic Blocks

It's legos with a some extra integration and verification cost

Processor Core (ARM, Tensilica, MIPS deriv)
With extra “options” like DP FPU, ECC
IP license cost $150k-$500k

NoC Fabric: (Arteris, Denali, other OMAP-4)
IP License cost: $200k-$350k

DDR3 1600 memory controller
(Denali / Cadence, SiCreations)
+ Phy and Programmable PLL
IP License: $250-$350k

PCIe Gen3 Root complex
IP License: $250k

Integrated FLASH Controller
IP License: $150k

10GigE or IB DDR 4x Channel
IP License: $150k-$250k

With Marty Deneroff
Applying Embedded to HPC (climate)

Must maintain 1000x faster than real time for practical climate simulation

- ~2 million horizontal subdomains
- 100 Terabytes of Memory
  - 5MB memory per subdomain
- ~20 million total subdomains
  - 20 PF sustained (200PF peak)
  - Nearest-neighbor communication

- **New discretization for climate model**
  - CSU Icosahedral Code

200km Typical resolution of IPCC AR4 models

25km Upper limit of climate models with cloud param

1km Cloud system resolving models transformational !!!

fvCAM

Icosahedral
Climate System Design Concept

Strawman Design Study (w/ Chris Rowen 2007)

10PF sustained
~120 m²
<3MWatts
< $75M

32% boards
per rack
100 racks @
~25KW

32 chip + memory
clusters per board (2.7
TFLOPS @ 700W)

VLIW CPU:
• 128b load-store + 2 DP MUL/ADD + integer op/ DMA per cycle:
• Synthesizable at 650MHz in commodity 65nm
• 1mm² core, 1.8-2.8mm² with inst cache, data cache
data RAM, DMA interface, 0.25mW/MHz
• Double precision SIMD FP : 4 ops/cycle (2.7GFLOPs)
• Vectorizing compiler, cycle-accurate simulator,
debugger GUI (Existing part of Tensilica Tool Set)
• 8 channel DMA for streaming from on/off chip DRAM
• Nearest neighbor 2D communications grid

32 processors per 65nm chip
83 GFLOPS @ 7W

32K
8 chan DMA

64-128K D
2x128b

External DRAM interface

External DRAM interface

8 DRAM per processor chip:
~50 GB/s

Opt. 8MB embedded DRAM

External DRAM interface

Opt. BMB embedded DRAM

Comm Link
Control

Master Processor

Proc Array

RAM
RAM

RAM
RAM

power + comms

32 boards
per rack

32 chip + memory
clusters per board (2.7
TFLOPS @ 700W)

External DRAM interface
Hardware Demo (Green Flash)

- Demonstrated during Supercomputing 2008
- Proof of concept
  - CSU atmospheric model ported to Tensilica Architecture
  - Single Tensilica processor running atmospheric model at 50MHz
- Emulation performance advantage
  - Processor running at 50MHz vs. Functional model at 100 kHz
  - 500x Speedup
- Actual code running - not representative benchmark
Application Driver: Seismic Imaging

- Seismic imaging used extensively by oil and gas industry
  - Dominant method is RTM (Reverse Time Migration)

- RTM models acoustic wave propagation through rock strata using explicit PDE solve for elastic equation in 3D
  - High order (8th or more) stencils
  - High computational intensity

- Typical survey requires months of computing on petascale-sized resources
We cannot touch an end-to-end engineered design? but can get damned close.

big win for efficiency from what is NOT included

Further improvements primarily constrained by the memory technology
At this point we are confident that SoC with off-the-shelf embedded RTL can compete with leading edge server chip designs.
Berkeley Supercomputer Predicts Your Doom

The University of California at Berkeley is rolling out a new breed of supercomputer, specially designed to predict the challenges presented by climate change, ultimately leading humanity to our doom and the computers to their rightful place as the masters of our earthly domain.

The idea driving the claim that supercomputers can be revolutionized is the radical notion that they can help us save the planet by predicting the future.
Prototypes are critical to accelerate software development
- System software stack + applications
Redefining “commodity”

- Must use “commodity” technology to build cost-effective design
- The primary cost of a chip is development of the intellectual property
  - Mask and fab typically 10% of NRE in embedded
  - Design and verification dominate costs
  - SoC’s for high perf. consumer electronics is vibrant market for IP/circuit-design (pre-verified, place & route)
  - Redefine your notion of “commodity”!

The ‘chip’ is not the commodity...

*The stuff you put on the chip is the commodity*
Technology Continuity for
A Sustainable Hardware Ecosystem

Very High Bandwidth
Energy Efficient
Photonic Interconnect

Ultra Energy
Efficient
Embedded
System Platform

Floating Point
Application
Resilience
Low Latency
Memory and
Interconnect

Energy Efficient
Cloud Computing
(Future Data
Centers)

High Performance
(Exascale)
Computing System

With Keren Bergman (Columbia LRL)
SoC for HPC Workshop Scope

1) **State of the Art:** What can be done to leverage commodity embedded IP components, tools, and design methodologies to create HPC-targeted designs.

2) **Technology Inventory and Requirements Analysis:** Survey the currently available IP building blocks and identify where gaps exist in current IP circuit technologies and design tools that will be crucial to HPC and datacenter-targeted SoC ASICs.

3) **Software Infrastructure:** What will be required of our software environment to take full advantage of a rapidly evolving SoC designs. What would need to change in our software engineering practices keep up with a more flexible and rapidly evolving hardware design target?

4) **Simulation/Modeling:** SoC poses challenges to existing monolithic CPU-centric simulation environments that were originally designed for cell-phone scale systems. What new technologies will be required to bring the kind of design agility to the HPC-SoC design space that is currently relied upon for competitive consumer electronic designs.

5) **OpenSoC:** What open technologies, tools, and open-source gate-ware are available to engage the academic and research community involved in exploring the design space for high performance SoCs.
• Plot a roadmap for creating an embedded component ecosystem for HPC
  – That leverages the enormous investments taking place in the embedded/consumer electronics market
  – Is effective for HPC

• Identify opportunities and weaknesses in the SoC strategy
  – What is the performance & market potential of this approach.
  – What is missing from the commodity IP component market
  – Where will market forces NOT deliver the kinds of components required for effective Server/HPC/WSC SoC designs

• Where should government agencies (DOE, DOD, DARPA, NASA, NSF) concentrate their R&D expenditures to open up an alternative path for technology innovation

Write a report documenting our findings
LBNL/Sandia Computer Architecture Laboratory
http://www.cal-design.org/
Computer Architecture Laboratory
Design Space Exploration
Interoperable Components

John Shalf
Simulator/Emulator Interoperability

Objectives and Approach using Chisel
Interoperability between Emulators and Simulators

• Complementary Skill Sets
  – **Software Simulators:**
    • *Fast to reconfigure HW parameters (instantaneous)*
    • *Slow clock rates for large devices (~kilohertz) for cycle accurate (Simulate small kernels)*
  – **Hardware Emulators:**
    • *Fast clock rates for large devices (50MHz) (Simulate larger applications)*
    • *Slow to reconfigure (takes hours to re-synthesize)*
  – **Design Synthesis:**
    • *For Novel hardware, need to synthesize circuit to calibrate power and timing models (need hardware synthesis path)*

• **Solution: CHISEL**
  – DSL for describing parameterized hardware simulator components
  – Single specification will generate C++ (software simulator), FPGA, and synthesizable RTL.
  – CAL is funding UCB subcontract that will extend Chisel to automatically generate SystemC bindings for software components
  – Other focus is to build up a NoC design for data movement experimentation
Steps in Design Space Exploration

Need to be Modeling Same Thing across each of these steps!
(or else this doesn’t make sense)

- **Conceptual Model**
- **Software Simulation**
- **Hardware Emulation**
- **Circuit/Design Synthesis**

Is there any value in this idea at all?
Where are the interesting parts of the parameter space?
Explore the hardware parameter space
Kernels on detailed model of hardware

Explore the software optimization space
Run larger apps and more detailed model

Calibrate Timing and Energy Model
Timing and energy for novel circuits unknown without synthesis step
Tensilica Processor Generator Example

- Core Architecture:
  - VLIW/SIMD Width
  - Pipeline Depth
- ISA:
  - Add/remove instructions
  - Write your own instructions
  - Add functional unit (FPU)
- Cache:
  - Sizes
  - Set associativity
- Local Stores *(stuff we added)*
  - Size
  - Map to Global Address Space
  - RDMA
- Direct inter-core Message Queues
- Collective DMA

Parameterized Hardware *(for Design Space Exploration)*
Tensilica Processor Generator Example

- **Tensilica Processor Generator**
- **Parameterized Hardware (for Design Space Exploration)**
- **C++ Simulator Component (SystemC)**
- **FPGA Emulator Component**
- **Synthesizable RTL**

**Conceptual Model** → **Software Simulation** → **Hardware Emulation** → **Circuit/Design Synthesis**
Tensilica Processor Generator Example

Can this concept be Generalized for all Components?
CHISEL: Generalizing the Solution!!!

Parameterized Simulator Components (for Design Space Exploration)

- CHISEL Component Generator
- SST/C++ Simulator Component (SystemC)
- CoDEx/FPGA Emulator Component
- Synthesizable RTL

Conceptual Model
Software Simulation
Hardware Emulation
Circuit/Design Synthesis
End

Discussion?
Consumer Electronics has Replaced PCs as the Dominant Market Force in CPU Design!!

Netbooks based on Intel Atom embedded processor is the fastest growing portion of “laptop” market.
## Evaluated Architectures

<table>
<thead>
<tr>
<th>Core Architecture</th>
<th>Intel Nehalem</th>
<th>NVIDIA GF100</th>
<th>Tensilica LX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>superscalar</td>
<td>dual-warp</td>
<td>VLIW</td>
</tr>
<tr>
<td>Type</td>
<td>out-of-order</td>
<td>in-order</td>
<td>in-order</td>
</tr>
<tr>
<td>SIMD</td>
<td></td>
<td>SIMT</td>
<td>custom</td>
</tr>
<tr>
<td>Clock (GHz)</td>
<td>2.40</td>
<td>1.15</td>
<td>1.00</td>
</tr>
<tr>
<td>SP GFlop/s</td>
<td>19.2</td>
<td>73.6</td>
<td>2.00</td>
</tr>
<tr>
<td>L1 Data $/s</td>
<td>32 KB</td>
<td>16 KB</td>
<td>8 KB</td>
</tr>
<tr>
<td>L2 Data $/LS</td>
<td>256 KB</td>
<td>48 KB</td>
<td>256 KB</td>
</tr>
<tr>
<td>SMP Architecture</td>
<td>Xeon E5530 (Gainestown)</td>
<td>Tesla C2050 (Fermi)</td>
<td>Green Wave</td>
</tr>
<tr>
<td>Threads/core</td>
<td>2</td>
<td>48 (max)</td>
<td>1</td>
</tr>
<tr>
<td>Cores/socket</td>
<td>4</td>
<td>14†</td>
<td>128</td>
</tr>
<tr>
<td>Sockets/SMP</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Shared Last $/socket</td>
<td>8 MB</td>
<td>768 KB</td>
<td>—</td>
</tr>
<tr>
<td>memory parallelism</td>
<td>HW prefetch</td>
<td>Multithreading</td>
<td>DMA</td>
</tr>
<tr>
<td>On-chip RAM</td>
<td>18.3 MB</td>
<td>3.4 MB</td>
<td>32 MB</td>
</tr>
<tr>
<td>DRAM Pin GB/s</td>
<td>51.2</td>
<td>144 (no ecc)</td>
<td>51.2</td>
</tr>
<tr>
<td>SP GFlop/s</td>
<td>153.6</td>
<td>1030.4</td>
<td>256</td>
</tr>
<tr>
<td>Power under RTM load</td>
<td>298 W</td>
<td>390 W (System)</td>
<td>66W‡</td>
</tr>
<tr>
<td></td>
<td></td>
<td>214 W (GPU-only)</td>
<td></td>
</tr>
<tr>
<td>Die Area</td>
<td>263mm²</td>
<td>576mm²</td>
<td>294mm²</td>
</tr>
<tr>
<td>Process</td>
<td>45nm</td>
<td>40nm</td>
<td>45nm</td>
</tr>
</tbody>
</table>
Reducing overheads for communication

- Lightweight energy efficient cores
- Better control of data movement
  - Direct message queues between cores
  - Local Store into the global address space
- Local-store for more efficient use of memory bandwidth
  - Can put Local store side-by-side with conventional cache
  - Design library enables incremental porting to local store
- Hardware support for lightweight synchronization
  - Enables direct inter processor communication for low-overhead synchronization
  - Maintain consistency between memory-mapped local stores
Green Wave ASIC Design
(power and area breakdown)

Power Breakdown
(70W total for SoC+ memory)

Area Breakdown
(240 mm² for SoC)

- Developed RTL design for SoC in 45 nm technology using off-the-shelf embedded technology + simulated with RAMP FPGA platform
Green Wave ASIC Design
*(power and area breakdown)*

---

**Power Breakdown**
*(70W total for SoC + memory)*

- NoC: 9%
- Core: 23%
- DRAM + Controllers: 49%
- Store: 7%

**Area Breakdown**
*(240 mm² for SoC)*

- Can reduce this power fraction substantially using Micron Hybrid Memory Cube technology.

**HMC-Gen2** = 15W device with 360+GB/s performance.

---

- Developed RTL design for SoC in 45 nm technology using off-the-shelf embedded technology + simulated with RAMP FPGA platform
CNS/SMC: Working set size for register file

Chemistry FP State Variables by Rank

- x86 has 16 FP named registers!

Allocate to registers

Leave in L1 cache
Register Spilling Behavior

Motivates inclusion of more explicitly managed memory near core

Chemistry State Variable Accesses Spilled to L1 Cache

- Having more register can filter cache traffic for state variables and prevent cache spills

128 or 256 registers greatly reduce the register spills
Cache Blocking

Ability (or inability) to use standard optimizations

- Cache blocking exposes trade-off between cache size and memory bandwidth:
  - **PRO:** Smaller working set — allows working set to fit into cache, enabling reuse
  - **CON:** Redundant memory traffic — pulls overlapping ghost zones in from memory

- This is a very standard optimization
  - Programming environments make it difficult to automate
  - Requires tedious architecture-dependent tuning

Are automatically managed caches *really* working for us??

- No blocking
- 2x blocking
- 4x blocking
Cache and Block Size Is Crucial for Memory Performance
(but current programming systems make it hard to infer block size)

Current use of Fortran or C as base languages is unable to connect data layout to iteration space. Currently forces manual optimization of blocking factor (or autotuning). This *should* be computable analytically (*strict data parallel semantics would enable that*).

**Bytes per Flop vs. Block Size for 128^3 Baseline CNS Code**

Optimal block size depends on available cache.
Loop Fusion
(nonstandard use of a standard optimization)

• Merge the bodies of two loops so that they are in the same loop nest
• Saves the memory traffic cost for:
  – streaming common input arrays into cache multiple times
  – streaming intermediate arrays in and out of memory (can eliminate the array completely)

Scenario 1:

L1: for i = 1 to N
   B[i] = f(A[i])
L2: for i = 1 to N
   C[i] = g(A[i], B[i])

L1: Stream A in, B out
L2: Stream A and B in, C out

Memory Traffic: 5N (with cache bypass)

Scenario 2:

L1: for i = 1 to N
   B = f(A[i])
   C[i] = g(A[i], B)

L1: Stream A in, C out

Memory Traffic: 2N
Dependency Graph for CNS and SMC
(effect on data locality and reuse distance)

Baseline
2.9 GB/sweep
1.78 Bytes/Flop

Simple Fusion
1.6 GB/sweep (−46%)
0.96 Bytes/Flop

Aggressive Fusion
0.48 GB/sweep (−84%)
0.29 Bytes/Flop
Benefits of Loop Fusion for CNS

*(are lost due to current semantic deficiencies of our programming model)*

Huge opportunity to reduce memory bandwidth requirements!!

*Current execution environments do not enable us to reason about this kind of fusion*

![Graph showing Byte to Flop Ratios vs Cache Size for Loop Fusion Scenarios ("best" block size)]

Select “best” strategy for each cache size

- Baseline
- Simple
- Aggressive
- Baseline (no cache)
- Fused (no cache)
- "Best" Strategy
Power Consequences of Big L1 Scratchpads

Byte to Flop Ratios vs Cache Size for Loop Fusion Scenarios ("best" block size)

Power consumed by large caches

Power w/DRAM

Power Reduction Opportunity

Baseline & Fusion
Power Breakdown for SRAM (its mostly data movement)

- Sub-array drvr. 31%
- Bitlines 52%
- Wordline 3%
- Decoder 6%
- Bitline mux 5%
- Sense amp mux 2%
- Sense amp 1%

Energy Breakdown for 256k SRAM (mostly data movement)
Bandwidth Tapering for HPC apps (interconnect)

% Bandwidth Utilization

Fat-tree level

BB3D (P=512)
Cactus (P=1024)
GTC (P=8192)
LBCFD (P=1024)
Mbench (P=256)
PARATEC (P=256)
PMEMD (P=256)
SuperLU (P=256)