Single-chip Cloud Computer

A 48 Core Research Microprocessor

Sebastian Steibl
Director
Germany Microprocessor Lab

30th June 2010
Evolving User Experiences

Single-Core Era

Multi-Core Era

Many-Core Era

Textual
Multi-purpose
Productive

Visual
Mobile
Entertaining

Immersive
Social
Perceptive
Performance Scaling Challenges

Energy Efficiency
Design Complexity
Programming Strategy
Emerging Applications
Motivations for SCC

- Many-core processor research
  - High-performance power-efficient fabric
  - Fine-grain power management
  - Message-based programming support

- Parallel Programming research
  - Better support for scale-out model servers
    - Operating system, communication architecture
    - Scale-out programming model for client
    - Programming languages, runtimes

An experimental processor, not a product!
High-Level Overview

- 2\textsuperscript{nd} Generation Intel Labs experimental processor
  - 567mm\(^2\) in 45nm high-K metal-gate CMOS
  - 1.3B transistors
- “Cluster-on-die” architecture
  - 48 Pentium\textsuperscript{®} Processor cores (x87FP only)
  - No HW cache coherency support
- Message-passing HW support
- SW-controlled Fine-grain power management
On-die Interconnect

• Architecture
  – 6x4 2D Mesh NoC
  – 16B wide data links + 2B sideband
  – 8 Virtual Channels in 2 classes
  – Fixed (X-Y) routing

• Performance
  – Target freq: 2GHz @ 1.1V
  – Link Bandwidth 64GB/s
  – 4 cycle latency

• Power Management
  – Independent Frequency & Voltage control
  – Sleep mode, clock gating, low power RF

![Graph showing supply voltage vs frequency with different supply voltages and corresponding frequencies: 0.5V 60MHz, 0.73V 300MHz, 0.94V 1.4GHz, 1.32V 1.3GHz, 1.34V 2.6GHz with labels for temperatures: 50°C, 0.94V, 1.4GHz and 1.32V, 1.3GHz, 0.94V, 0.9GHz.](image)
Memory Architecture

- **Memory**
  - Up to 64GB DDR3 via 4 MCs
  - 16KB SRAM in each tile as Message Passing Buffer (MPB)

- **Caching**
  - 32KB L1 per core (16KB I,D)
  - 256KB L2 per core (12MB total)
  - No HW cache-coherent shared memory

- **Ability to model compute / memory / NoC limited architectures for scalability experiments**
Power Management

- Configurable MC, mesh, SIF voltage & frequency
- Software-controlled DVFS* of cores
  - Fine-grain voltage control at 4 tile cluster level (6.25mV)
  - Frequency control at tile level (divide by 1…16)
  - Closed loop - thermal sensors per tile, current through BMC

Power management latencies
V : High latency, \(O(\text{Million})\) cycles
F : Low latency, \(O(\text{few})\) cycles

Wide operating range
125W @ 1.14V, 1GHz
25W @ 0.7V, 125MHz

*Dynamic voltage and frequency scaling
SCC Platform

- Up to 64GB main memory
- µC for hardware control & monitoring
- SCC “Chipset” in FPGA to PC for system control & debug

Management Console PC
SCC SW architecture

Each core can run its own OS
- Standard Linux* kernel with SCC specific extensions available
- Can run different / heterogeneous OS at the same time
- Experimental OSs possible / desired

Message passing SW support
- On-die TCP/IP network drivers
- High-level message passing API for applications – “RCCE”
  - One-sided communication (Get, Put, Send, Recv)
  - MPB allocation, synchronization
- MPICH2 demo implementation
SCC Memory Model

• Virtual to physical address translation (page table)
  – Cacheable/uncacheable memory
  – Write-back/write-through behavior
  – Message Passing Data Type (MPDT)

• Message passing data
  – Reads>writes bypass L2 cache
  – Write-combine buffer to aggregate cache lines
  – Tagged in L1 as MPDT
  – New instruction to selectively invalidate MPDT lines in L1

• Atomic test & set register bits in each tile
  – Enable global synchronization
Physical Address Mapping

- Translation from core-physical to system-physical address
  - Look-Up Table (LUT) with 16MB granularity
  - Unique configuration for each core (runtime reprogrammable)

- LUT content:
  - NoC location
    - Coordinate
  - MC, MPB, register bits
  - 34b system-physical address
Message Passing on SCC

- Regions of memory mapped to multiple cores
  - Message Passing Buffer (MPB) for small, fast messages
  - Larger buffers in off-die memory
  - Transfer of 16MB chunks through LUT reprogramming

- Memory consistency **not** enforced by hardware
  - Explicit cache invalidation of MPDT lines
  - Shared access synchronization via atomic register bits

- Core-core signaling
  - Polling
  - Assertion of interrupts signal through register bits
**SW managed shared memory**

- Leverage shared memory (SHM) support in SCC
- Golden copy is saved at SHM, needn’t communicate with other nodes
- Do memcpy between cacheable private memory & uncacheable SHM
SCC Case Studies

- Financial Analytics w/ shared virtual memory
- Microsoft® Visual Studio® Advanced Power Management
- JavaScript Physics Modeling
- HPC Parallel Workloads
- MPICH2 & Intel® Trace Analyzer
Designing SCC

Aggressive design paradigms
- Use one HDL for behavioral, RTL and validation
- Follow top-down structural refinement of tiled architecture
- Employ system emulation as early as possible
- Code RTL “friendly” for silicon & FPGA at the same time

Validation Flow
- RTL simulation
  - Tile, Cluster and FC level
  - Broad coverage on tile level
- System emulation
  - Running complete subsystem
  - Coverage for long running tests
  - Used as software development vehicle
SCC FPGA Emulation

- MCEMU manycore emulator
  - Proprietary emulation hardware
  - Presented at WARP’08 and IDF’09
  - Five Xilinx* Virtex*4/5 FPGAs per card
  - Serial interconnect for NoC connectivity

- 12 core SCC running on MCEMU at 6MHz
  - 6 tiles/ 12 cores with memory controller

- Stable software development vehicle prior to silicon
  - Bring up of a OSless C flow
  - Cycle accurate optimization of RCCE
  - Bring up of Linux OS, drivers & apps
  - Development of post-silicon test suite
SCC Design Flow Benefits

- Stable validation environment early in design flow
  - Followed tiled approach in implementation, simulation & emulation
  - Straight composition of design blocks from three int. sites
- Synergy among RTL simulation & emulation
  - Sharing same code base
- High code coverage and confidence before tape-out
  - Running full applications on top of OS
  - Emulating complete subsystem

Emulation friendly design flow, design regularity are key
SCC Co-Travelers Program

- Currently building SCC software research community
  - Research partners for 2010 have been selected according to objective criteria and research proposals

- SCC community website available today
  - Communities.intel.com/community/marc
  - Shared ideas to help overcome research problems
  - Peer assistance on “How To” issues, design research plans

- Future Intel Sponsored Conferences and Workshops
  - Present and discuss research findings and results
  - Generate new ideas for future many-core software research
  - Memory, Power Mgmt, Languages, etc
Sample S/W Research Projects

• System balance for datacenter workloads
  – I/O to compute
  – Fabric to compute

• On-die communication
  – Exploiting on-die core-core data passing
  – I/O flows and multi-core sharing

• Operating system architecture
  – Distributed kernels for cluster on die
  – Power managed & variation aware scheduling

• Programming language support
  – Partitioned global address space and messaging languages
  – Software management of cache-coherence

...
"We're very excited about Intel's SCC. In the Barrelfish project we are designing OS architectures for future multi-core and many-core systems. The chip's memory system and message passing support are a great fit for us, and it's an ideal vehicle for us to test and validate our ideas."
– Prof. Timothy Roscoe, ETH Zurich

"The upcoming Single-chip Cloud Computer is of great interest to application developers and tools researchers. The availability of the hardware will greatly accelerate our development of applications and tools for massively parallel computing platforms."
– Prof. Wen-Mei Hwu, University of Illinois, UPCRC@Illinois co-director
The Barrelfish Operating System

http://www.barrelfish.org [A. Baumann et al., 22nd ACM SOSP 2009]

- Being developed under 3-clause BSD-style license by
  - ETH Zürich Systems Group
  - Microsoft Research Cambridge

- OS modeled as a distributed system
- Replicated (not shared) state
- Hardware-neutral structure
- Explicit communication based on message-passing

- Ported to SCC within 2 man-months
- RCCE enabled with time-sharing of SCC’s message passing buffers with same scalability as Linux RCCE
Summary

• SCC provides a unique experimental platform for many-core software research
  – Cluster-on-die architecture with many IA cores
  – Flexible configurations w/ rich NoC and memory subsystem
  – Designed for exploring scalability on future architectures
  – Power efficiency through fine-grain DVFS

• Please monitor & participate in the SCC community:
  • Communities.intel.com/community/marc
Thank You!

Partner Disclaimer: It is acknowledged that the use of the word "Partner" is a commonly used term in the technology industry to designate a marketing relationship between otherwise unaffiliated companies and is used in accordance with this common usage herein. The use of the word “Partner” herein shall not be deemed to nor is it intended to create a partnership, agency, joint venture or other similar arrangement between Intel and such partners and the employees, agents and representatives of one party shall not be deemed to be employees, agents or representatives of the other. Intel and the partners shall be deemed to be independent contractors and shall have no authority to bind each other.

Copyright © 2010, Intel Corporation. All Rights Reserved.

Intel and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

* Other names and brands may be claimed as the property of others