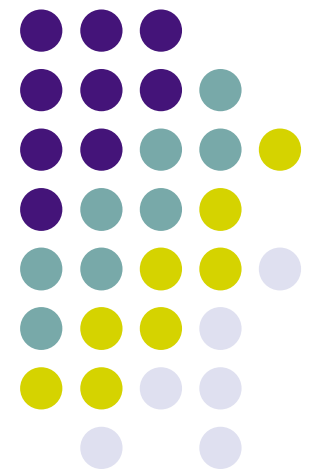
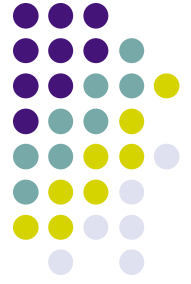


Research Directions for On-chip Network Microarchitectures

Luca Carloni, Steve Keckler,
Robert Mullins, Vijay Narayanan,
Steve Reinhardt, Michael Taylor

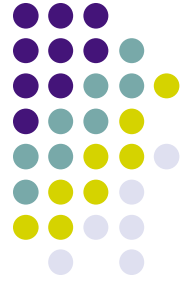


12/7/06



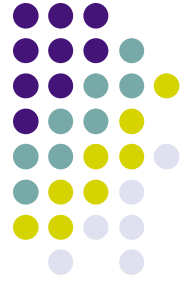
Overview

- Minimizing latency & power are key
 - Fundamental research needed in routers, interfaces, electrical design
 - Reliability and variability are emerging challenges
- Programming interface is key
 - Must expose low latency to software
 - Programmability drives network constraints & features
- Broader impact: making multicore systems viable, usable, and effective



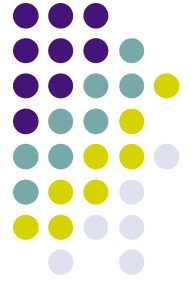
Outline

- Crosscutting issues
 - Latency
 - Power
- Other key issues
 - Programmability
 - Variability
 - Technology
 - System management
 - Design tools



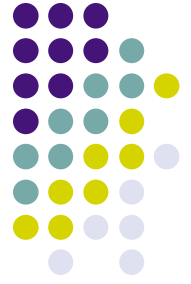
Driving Latency Down

- Motivation
 - Lower overheads simplify programming
 - Less need for programmers to avoid communication
 - Exploit low fundamental latency of integration
 - Don't throw away benefit by imposing interface/routing overheads
 - Enable closer cooperation between cores
- Enabling technologies
 - Network interfaces
 - Thin abstractions: expose hardware to software
 - Integration with processor core
 - Programming models to leverage abstractions (and vice versa)
 - Router innovations
 - Fewer pipe stages, higher frequency (within power envelope)
 - Maintaining low latency under load
 - Identifying/prioritizing latency critical communications
 - Exploiting static information (e.g., circuit switching)



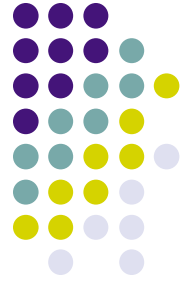
Power

- Different design points demand different solutions
- Absolute power
 - Embedded vs. high performance
 - Other intermediate points?
 - Power/thermal-constrained routers & routing
 - Stay within envelope
 - Exploit static information / common cases
- Ratio of compute/network power
 - Depends on compute/communicate ratio
 - Can we trade this off dynamically?
 - Across different apps
 - Due to phase behavior within app
 - E.g., DVFS in the network (as well as cores)



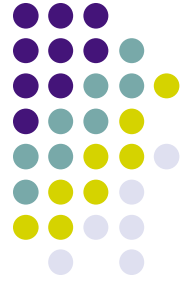
Programmer Support

- What does the programmer want?
 - Fast and robust networks
 - Easy to use (efficient network access, easy to program)
 - Ability to reason about performance, etc.
- Performance and Robustness
 - Low latency in hardware - fast routers, efficient NIs
 - Latency in software (programming model support)
 - Microarchitecture support for higher level mechanisms
 - Examples: data transfer (small/large), synchronization, invocation, etc.
 - Microarchitecture support for robustness
 - Priority/QOS
 - Microarchitecture support for end-to-end deadlock avoidance
 - Example: network driven exceptions for unusual cases
 - Pushing intelligence into the network
 - Cache coherence just one example
 - Common interface for different scales of network
 - On-chip, off-chip, board, rack, system
 - Can we unify to common protocols, user-interfaces?
 - Can microarchitecture make unification efficient?
- Understanding network behavior
 - Predictability / cost model for application programmer
 - Measurement & feedback to programmer
 - Is network power something that should be exposed for optimization in some way?



Variability

- Sources of variability
 - Workload, across and within applications
 - Burstiness, stream vs. unstructured, large vs. small messages
 - Message classes (data, synch, etc.)
 - Fabrication process
- Opportunities and challenges
 - What are the message types, what are the networks
 - How should individual networks be optimized based on different traffic characteristics
 - Variability provides opportunity to improve power efficiency
 - Dynamically ride the pareto curve (power/performance)
 - Shift power from network to execution (or vice versa)
 - Can this be hidden from programmer?
 - Fabrication process tolerant networks
 - Post fabrication tuning, exploit elastic network properties



Technology

Current: How do design flow choices impact NOC micro-architecture design?

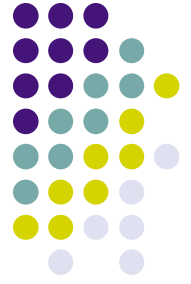
- custom vs asic
- floorplan impact on micro-architecture effectiveness

Short-Term: What will be the impact of technology scaling?

- router vs. link costs (delay/power)
- router vs. link features (diagnostics, error correction)

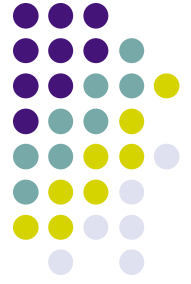
Long-Term: What will be the impact of emerging technologies?

- 3D integration, carbon nanotubes, optical communication
- new switching fabrics, arbitration, buffering



System Management

- NOC can facilitate distributed diagnostics and self-adaptation
 - not just for NOC, but for the overall system
 - process variations, reliability, dynamic variations, security, power management
- architectural support
 - sensing
 - online monitors and performance counters for network traffic
 - processing
 - aggregation, system-state recognition and future-state prediction
 - actuating
 - [power] knobs for dynamic voltage/frequency scaling (DVFS) of routers, cores, for dynamic shut-down of system subsets
 - [security] on-demand encryption and link blocking for security
- Challenge: How to do all this while keeping overheads low?



Design tools

- Stochastic vs. realistic workloads
- How valid is trace-driven evaluation?
- Rapid evaluation
 - FPGAs
 - Analytical techniques
- Repeatability of research experiments