Power and Area Efficient Network-on-Chip Architectures

SRC TASK 1204.001

Israel Cidon
Ran Ginosar
Avinoam Kolodny

Technion—Israel Institute of Technology
Research Objective

Power and Area Efficient Network on Chip (NoC):

- Network layer architecture
  - switching techniques
  - routing
  - congestion control
  - topology
- Fast on-chip communication links
- Circuit design for NoC components

2005 SRC Review
The Team

- **Faculty:**
  Israel Cidon, Ran Ginosar, Avinoam Kolodny

- **Graduate Students:**
  Evgeny Bolotin, Zvika Guz, Zigi Walter, Arkadiy Morgenshtein, Reuven Dobkin

- **Industrial liaisons:**
  Odi Dahan, Freescale
  Shlomo Greenberg, Freescale
  Michael Zimin, Freescale

- **SRC monitor:**
  David C. Yeh

- **Funding (in part):**
  SRC/Freescale, Intel, Ceva-DSP
Industrial Cooperation

• Presentations and discussions with
  ▪ Freescale
  ▪ Intel
  ▪ Zoran
  ▪ Ceva-DSP
  ▪ Mellanox
  ▪ Connexant
  ▪ EZ-Chip
Publications

Presentation Outline

• Research motivation
  ▪ Problems
  ▪ Advantages of NoC
  ▪ Scalability analysis
  ▪ Challenges

• Results of 2004

• Future work
Problems in Evolutionary Approach to SOC

2005 SRC Review
NoC Paradigm Advantages

- Efficient sharing of wires
- Lower cost / lower risk / faster design
- Scalability
- NoC employs statistical multiplexing via packets
- NoC is an infrastructure (e.g. power, clock)
- NoC is customized for each chip
NoC: Why Now?

- Global interconnect delay, noise, power
- Full-chip productivity crisis
- Chip Multi-Processors
NoC scalability vs. alternatives

For Same Performance, compare the cost of:

NoC:

Non-Segmented Bus:

Point-to-Point:

Segmented Bus:
Asymptotic cost scalability

Power and Area required to provide same bandwidth versus number of system modules $n$

<table>
<thead>
<tr>
<th>Arch</th>
<th>Total Area</th>
<th>Power Dissipation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS-Bus</td>
<td>$O(n^3 \sqrt{n})$</td>
<td>$O(n\sqrt{n})$</td>
</tr>
<tr>
<td>S-Bus</td>
<td>$O(n^2 \sqrt{n})$</td>
<td>$O(n\sqrt{n})$</td>
</tr>
<tr>
<td>NoC</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td>PTP</td>
<td>$O(n^2 \sqrt{n})$</td>
<td>$O(n\sqrt{n})$</td>
</tr>
</tbody>
</table>

* E. Bolotin, I. Cidon, R. Ginosar and A. Kolodny, “Cost Considerations in Network-on-Chip”, INTEGRATION – the VLSI journal, 2004

2005 SRC Review
NoC Challenges

• Low cost:
  - Area (routers, interfaces and links)
  - Power (dynamic, leakage)
• Flexible standard interface
• Multiple levels of service (QoS)
  - Throughput
  - End-to-end delay
• Low design effort
Presentation Outline

• Research motivation

• Results of 2004
  ▪ QNoC architecture
  ▪ Design flow for a QNoC-based system
  ▪ Circuit level aspects: links and routers

• Future work
QNoC: Quality-of-service NoC architecture

Define Service Levels like:

- **Signaling** – interrupts, signals.
- **Real-Time** - audio, video.
- **Read/Write (RD/WR)** – bus semantics
- **Block-Transfer** – DMA semantics

✓ Different QoS (delay characteristics) for each Service Level

QNoC topology and routing-path

- Mesh topology
  - Variable capacity links
- Fixed shortest path routing (X-Y)
  - Simple Router (no tables, simple logic)
  - No deadlock scenario
  - No retransmission
  - No reordering of messages
  - Power-efficient
Wormhole routing

- For reduced buffering
- Reduced Latency
- Simple router hardware
- Virtual channels enable variable link speeds

Wormhole Packet:
Router structure

- Flits stored in input ports
- Output port schedules transmission of pending flits according to:
  - Priority (Service Level)
  - Buffer space in next router
  - Round-Robin on input ports of same SL
  - Preempt lower priority packets
QNoC router with multiple Virtual Channels

Multiple VCs link:

The QNoC Router:
Simulation Model

- OPNET Models for QNoC:
  - Node (Source/Sink)
  - Router
  - Port
  - Link
- Any topology and traffic load
- Statistical traffic generation at source nodes
- Flit level simulations
Simulation example

QNoC Example:

Results Example:
NoC Customization

Place Modules

Trim routers / ports / links

Adjust link capacities
QNoC-based System Design Flow

- Behavioral simulation with “ideal network”
  - Define traffic requirements
- Placement
  - NoC Cost function
- Adjust link capacities
  - Satisfy QoS
  - Use analytical delay estimation
- Verify timing by statistical simulation on full network model
Capacity Allocation Problem

- Classical wormhole networks: uniform link capacity
  - Simple but delay unbalanced!
- Slacks should be minimized
- Optimization problem:

  Given:
  - System topology and routing
  - Each flow's bandwidth ($f^i$) and delay bound ($T^i_{REQ}$)

  Minimize total link capacity ($\sum_{e \in E} C_e$)
  Such that:

\[
\forall \text{link } e : \quad \sum_{i|e \in \text{path}(i)} f^i < C_e
\]

\[
\forall \text{flow } i : \quad T^i \leq T^i_{REQ}
\]
Network Delay Model

- **Analysis of** mean packet delay in wormhole network
  - Multiple Virtual-Channels
  - Different link capacities
  - Different communication demands
- Iteratively use the analysis to allocate capacities subject to delay requirements
Capacity Allocation Example

- A SoC-like system with specific traffic demands and delay requirements
- “Classic” design: 41.8Gbit/sec
- Using the algorithm: 28.7Gbit/sec
- Total capacity reduced by 30%
QNoC VHDL Hardware Generation Tool

Standard Topology Example:

Router Layout:

Custom Topology Example:

Synthesis results for various routers (Mosis 0.35 um):

<table>
<thead>
<tr>
<th>Number of ports</th>
<th>Area [µm²]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1,323,955</td>
</tr>
<tr>
<td>3</td>
<td>1,720,618</td>
</tr>
<tr>
<td>4</td>
<td>2,117,281</td>
</tr>
<tr>
<td>5</td>
<td>2,513,964</td>
</tr>
</tbody>
</table>
Long-haul serial links?
Comparing Serial & Parallel Links

The benefit of Serial link in 70nm is more pronounced than in 130nm because of increased leakage current of repeaters in the parallel link.
Area of parallel vs. serial link

Serial link consumes lower area for long wires, where the area of serializer is not dominant, and throughput is achieved without excessive device sizing.
Asynchronous Router

Solves synchronization, clock domain crossings, timing, long connects

2005 SRC Review
High speed asynchronous serial links

Clock Domain 1
SYNCHRONIZER
ENCODER
SERIALIZER

Asynchronous Domain

Clock Domain 2
SYNCHRONIZER
DECODER
DeSERIALIZER
Phase and State dual-rail encoding

- "Self shielding": Only one differential pair switches

00 ↔ 01
10 ↔ 11

Phase bit phase bit: '0'

State bit: '1'

2005 SRC Review
Presentation Outline

• Research motivation

• Results of 2004

• Future work
  ▪ Hot-Spots
  ▪ Latency-sensitive connections
• HS is not a local problem. Traffic not destined to the HS suffers too!

The Green packet experiences long delay even though it doesn’t share any link with HS traffic

2005 SRC Review
Source fairness problem

- QNoC routers are designed to be simple
  - Fast, low-area and power efficient
  - But cannot assure fair sharing of system’s modules
- Instead, a simple end-to-end flow-control mechanism can be applied
NoC for Latency-Sensitive Communication

The Problem:

*Low Latency is crucial for urgent data*

In Mesh:
Latency can reach many hop cycles!
NoC for Latency Sensitive Communication

Hierarchical solution approach:

Irregular Mesh:

Hierarchical Mesh:

- Where to add such express links
- Performance improvements vs. cost
Summary

• Develop the QNoC design paradigm:
  ▪ Architecture
  ▪ Links
  ▪ Circuits
  ▪ Design flows & tools

• Start to investigate NoC-based multiple-core processors, as a proof-of-concept.
Backup – Mean Delay Equations

• Simple M/M/1 model: \( t_j^i = \frac{1}{l \cdot C_j - \Lambda_j^i} \)
  (mean flit interleaving delay of flow \( i \) on link \( j \))

• Accounting for inter-link dependencies:
  \[
  \tilde{t}_j^i = t_j^i + \sum_{k \mid k \in \pi_j^i} \frac{BW_k^i}{C_k} \cdot \frac{t_k^i}{\text{dist}^i(j, k)}
  \]
  (flit interleaving delay is affected by the delay in subsequent hops
  weighted by their utilization and distance)

• The total transfer time is dominated by the hop with the
  lowest service rate:
  \[
  T^i \approx \frac{l \cdot m^i}{\max(\tilde{t}_j^i \mid j \in \pi^i)}
  \]
4-by-4 System