Networks on Chips (NoC) – Keeping up with Rent’s Rule and Moore’s Law

Avi Kolodny
Technion – Israel Institute of Technology

International Workshop on System Level Interconnect Prediction (SLIP)

March 2007
Acknowledgements

- Research colleagues
  *Israel Cidon,*
  *Ran Ginosar,*
  *Idit Keidar,*
  *Uri Weiser*

- Students:
  *Evgeny Bolotin,*
  *Reuven Dobkin,*
  *Roman Gindin,*
  *Zvika Guz,*
  *Tomer Morad,*
  *Arkadiy Morgenshtein,*
  *Zigi Walter*

- Sponsors
Keeping up with Moore’s law:

Principles for dealing with complexity:

- Abstraction
- Hierarchy
- Regularity
- Design Methodology

Source: S. Borkar, Intel
NoC = More **Regularity** and Higher **Abstraction**

From: **Dedicated signal wires**

To: **Shared network**

Similar to:
- Road system
- Telephone system
NoC essentials

- Communication by packets of bits
- Routing of packets through several hops, via switches
- Parallelism
- Efficient sharing of wires

Switch (a.k.a. Router)
Origins of the NoC concept

- The idea was talked about in the 90’s, but actual research came in the new Millenium.
- Some well-known early publications:
  - Guerrier and Greiner (2000)
    - “A generic architecture for on-chip packet-switched interconnections”
  - Hemani et al. (2000)
    - “Network on chip: An architecture for billion transistor era”
  - Dally and Towles (2001)
    - “Route packets, not wires: on-chip interconnection networks”
  - Wingard (2001)
    - “MicroNetwork-based integration of SoCs”
  - Rijpkema, Goossens and Wielage (2001)
    - “A router architecture for networks on silicon”
  - Kumar et al. (2002)
    - “A Network on chip architecture and design methodology”
  - De Micheli and Benini (2002)
    - “Networks on chip: A new paradigm for systems on chip design”
From buses to networks

Original bus features:

- One transaction at a time
- Central Arbiter
- Limited bandwidth
- Synchronous
- Low cost
Advanced bus

Original bus features:
• One transaction at a time
• Central Arbiter
• Limited bandwidth
• Synchronous
• Low cost

New features:
• Versatile bus architectures
• Pipelining capability
• Burst transfer
• Split transactions
• Overlapped arbitration
• Transaction preemption and resumption
• Transaction reordering…
Evolution or Paradigm Shift?

- **Architectural paradigm shift**
  - Replace the wire spaghetti by a network

- **Usage paradigm shift**
  - Pack everything in packets

- **Organizational paradigm shift**
  - Confiscate communications from logic designers
  - Create a new discipline, a new infrastructure responsibility
  - Already done for power grid, clock grid, …

![Diagram showing network link, router, module, and bus connections.](image-url)
Past examples of paradigm shifts in VLSI

The Microprocessor
From: Hard-wired state machines
To: Programmable chips
  • Created a new computer industry

Logic Synthesis
From: Schematic entry
To: HDLs and Cell libraries
  • Logic designers became programmers
  • Enabled ASIC industry and Fab-less companies
  • “System-on-Chip”
Characteristics of a paradigm shift

- Solves a critical problem (or several problems)
- Step-up in abstraction
- Design is affected:
  - Design becomes more restricted
  - New tools
  - The changes enable higher complexity and capacity
  - Jump in design productivity
- Initially: skepticism. Finally: change of mindset!

Let's look at the problems addressed by NoC
Critical problems addressed by NoC

1) Global interconnect design problem: delay, power, noise, scalability, reliability

2) System integration productivity problem

3) Chip Multi Processors
   (key to power-efficient computing)
1(a): NoC and Global wire delay

Long wire delay is dominated by Resistance

Add repeaters

Repeaters become latches (with clock frequency scaling)

Latches evolve to NoC routers

Source: W. Dally
1(b): Wire Design for NoC

- **NoC links:**
  - Regular
  - Point-to-point (no fanout tree)
  - Can use transmission-line layout
  - Well-defined current return path

- Can be optimized for noise / speed / power
  - Low swing, current mode, ....
1(c): NoC Scalability

For Same Performance, compare the wire-area cost of:

NoC: $O(n)$

Simple Bus: $O(n^3 \sqrt{n})$

Point-to-Point: $O(n^2 \sqrt{n})$

Segmented Bus: $O(n^2 \sqrt{n})$

E. Bolotin et al., “Cost Considerations in Network on Chip”, *Integration*, special issue on Network on Chip, October 2004
1(d): NoC and communication reliability

- Fault tolerance and error correction

1(e): NoC and GALS

- System modules may use different clocks
  - May use different voltages
- NoC can take care of synchronization
- NoC design may be asynchronous
  - No waste of power when the links and routers are idle
2: NoC and engineering productivity

- NoC eliminates ad-hoc global wire engineering
- NoC separates computation from communication
  - NoC supports modularity and reuse of cores
- NoC is a platform for system integration, debugging and testing

Call for Participation

DATE 07 Friday Workshop on

Diagnostic Services in Network-on-Chips
— Test, Debug, and On-Line Monitoring —

Palais des Congrès Acropolis — Nice, France
Friday April 20, 2007
3: NoC and CMP

- Uniprocessors cannot provide power-efficient performance growth
  - Interconnect dominates dynamic power
  - Global wire delay doesn’t scale
  - Instruction-level parallelism is limited

- Power-efficiency requires many parallel local computations
  - Chip Multi Processors (CMP)
  - Thread-Level Parallelism (TLP)

- Network is a natural choice for CMP!
Why **Now** is the time for NoC?

- Difficulty of DSM wire design
- Productivity pressure
- CMPs
Characteristics of a paradigm shift

- Solves a **critical problem** (or several problems)
- Step-up in **abstraction**
- **Design** is affected:
  - Design becomes more **restricted**
  - New **tools**
  - The changes enable **higher complexity and capacity**
  - Jump in **design productivity**
- Initially: skepticism. Finally: change of mindset!

Now, let’s look at the **Abstraction provided by NoC**
Traffic model abstraction

- Traffic model may be captured from actual traces of functional simulation
- A statistical distribution is often assumed for messages
Data abstraction

Message

Packet

Flit
(Flow control digit)

Phit
(Physical unit)
Layers of Abstraction in Network Modeling

- **Software layers**
  - O/S, application

- **Network and transport layers**
  - Network topology e.g. crossbar, ring, mesh, torus, fat tree, …
  - Switching Circuit / packet switching: SAF, VCT, wormhole
  - Addressing Logical/physical, source/destination, flow, transaction
  - Routing Static/dynamic, distributed/source, deadlock avoidance
  - Quality of Service e.g. guaranteed-throughput, best-effort
  - Congestion control, end-to-end flow control

- **Data link layer**
  - Flow control (handshake)
  - Handling of contention
  - Correction of transmission errors

- **Physical layer**
  - Wires, drivers, receivers, repeaters, signaling, circuits, …

Let’s skip a tutorial here, and look at an example
Architectural choices depend on system needs

- A large design space for NoCs!

I. Cidon and K. Goossens, in "Networks on Chips", G. De Micheli and L. Benini, Morgan Kaufmann, 2006
Example: QNoC
Technion’s Quality-of-service NoC architecture

- Application-Specific system (ASIC) assumed
  - ~10 to 100 IP cores
  - Traffic requirements are known a-priori
- Overall approach
  - Packet switching
  - Best effort ("statistical guarantee")
  - Quality of Service (priorities)

Choice of generic network topology

- Simple mesh fits planar chip
- Short links
Topology customization

- Irregular mesh
  - Address = coordinates in the basic grid
Message routing path

- Fixed shortest-path routing (X-Y)
  - Simple Router
  - No deadlock scenario
  - No retransmission
  - No reordering of messages
  - Power-efficient
Wormhole Switching

- Small number of buffers
- Low latency
The “hot module” IP1 is not a local problem. Traffic destined elsewhere suffers too!

The Green packet experiences a long delay even though it does NOT share any link with IP1 traffic.
Statistical network delay

- Some packets get more delay than others, because of blocking
Average delay depends on load

![Graph showing the relationship between normalized time and utilization, with two lines indicating mean delay: one for analysis and one for simulation.](image-url)
Quality-of-Service in QNoC

- Multiple priority (service) levels
  - Define latency / throughput
  - Example:
    - Signaling
    - Real Time Stream
    - Read-Write
    - DMA Block Transfer
  - Preemptive

- Best effort performance
  - E.g. 0.01% arrive later than required

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
Virtual Channels
QNoC router with multiple Virtual Channels
Simulation Model

- OPNET Models for QNoC
- Any topology and traffic load
- Statistical or trace-based traffic generation at source nodes
Simulation Results

- Flit-accurate simulations

Delay of high-priority service levels is not affected by load.
## Perspective 1: NoC vs. Bus

<table>
<thead>
<tr>
<th><strong>NoC</strong></th>
<th><strong>Bus</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Aggregate bandwidth grows</td>
<td>- Bandwidth is limited, shared</td>
</tr>
<tr>
<td>- Link speed unaffected by N</td>
<td>- Speed goes down as N grows</td>
</tr>
<tr>
<td>- Concurrent spatial reuse</td>
<td>- No concurrency</td>
</tr>
<tr>
<td>- Pipelining is built-in</td>
<td>- Pipelining is tough</td>
</tr>
<tr>
<td>- Distributed arbitration</td>
<td>- Central arbitration</td>
</tr>
<tr>
<td>- Separate abstraction layers</td>
<td>- No layers of abstraction (communication and computation are coupled)</td>
</tr>
</tbody>
</table>

**However:**

- No performance guarantee
- Extra delay in routers
- Area and power overhead?
- Modules need network interface
- Unfamiliar methodology

**However:**

- Fairly simple and familiar
## Perspective 2: NoC vs. Off-chip Networks

<table>
<thead>
<tr>
<th>NoC</th>
<th>Off-Chip Networks</th>
</tr>
</thead>
</table>
| ✷ Sensitive to cost:  
  - area  
  - power  
| ✷ Cost is in the links |
| ✷ Wires are relatively cheap |
| ✷ Latency is critical |
| ✷ Traffic may be known a-priori |
| ✷ Design time specialization |
| ✷ Custom NoCs are possible |
| ✷ Latency is tolerable |
| ✷ Traffic/applications unknown |
| ✷ Changes at runtime |
| ✷ Adherence to networking standards |
NoC can provide system services

Example: Distributed CMP cache

Characteristics of a paradigm shift

- Solves a critical problem (or several problems)
- Step-up in abstraction

- **Design** is affected:
  - Design becomes more restricted
  - New tools
  - The changes enable higher complexity and capacity
  - Jump in design productivity

- Initially: skepticism. Finally: change of mindset!
VLSI CAD problems

- Application mapping
- Floorplanning / placement
- Routing
- Buffer sizing
- Timing closure
- Simulation
- Testing
VLSI CAD problems reframed for NoC

- Application mapping *(map tasks to cores)*
- Floorplanning / placement *(within the network)*
- Routing *(of messages)*
- Buffer sizing *(size of FIFO queues in the routers)*
- Timing closure *(Link bandwidth capacity allocation)*
- Simulation *(Network simulation, traffic/delay/power modeling)*
- Testing

- … combined with problems of designing the NoC itself *(topology synthesis, switching, virtual channels, arbitration, flow control,……)*

Let's see a NoC-based design flow example
QNoC-based SoC design flow

1. Inter-module Traffic model
2. Place Modules
3. Trim routers, ports, links
4. Determine routing
5. Adjust link capacities
Routing on Irregular Mesh

Goal: Minimize the total size of routing tables required in the switches

Routing Heuristics for Irregular Mesh

Routing Cost in 12x12 NoC
Random problem instances

- Distributed Routing (full tables)
- X-Y Routing with Deviation Tables
- Source Routing
- Source Routing for Deviation Points

Routing Cost [gates]

Hotspot Number

0 10,000 20,000 30,000 40,000 50,000 60,000 70,000 80,000 90,000

Log (Routing Cost)

Systems with real applications

- DR
- TT
- XYDT
- SR
- SRDP

MPEG4 VOPD
Timing closure in NoC

- Define inter-module traffic
- Place modules
- Increase link capacities

- QoS Satisfied?
  - Too low capacity results in poor QoS
  - Too high capacity wastes power/area
  - Uniform link capacities are a waste in application-specific systems!
Network Delay Modeling

- **Analysis of mean packet delay in wormhole network**
  - Multiple Virtual-Channels
  - Different link capacities
  - Different communication demands

Queuing delay:

\[
Q^i = \frac{1}{2 \cdot \left( \frac{1}{T_{\text{network}}^i} - \lambda^i \right)} - \frac{T_{\text{network}}^i}{2}
\]

Flit interleaving delay approximation:

\[
t_j^i = \frac{1}{C_j - l \cdot \sum_{f \mid j \in \pi^f \land f \neq i} \lambda^f \cdot m^f}
\]

Capacity Allocation Problem

- **Given:**
  - system topology and routing
  - Each flow’s bandwidth ($f_i$) and delay bound ($T_{REQ}^i$)
- **Minimize total link capacity**
  \[
  \sum_{e \in E} C_e
  \]

- **Such that:**
  \[
  \forall \text{link } e: \quad \sum_{i \mid e \in \text{path}(i)} f_i < C_e
  \]
  \[
  \forall \text{flow } i: \quad T_i \leq T_{REQ}^i
  \]
State of the art: NoC is already here!

- > 50 different NoC architecture proposals in the literature; 2 books; hundreds of papers since 2000
- Companies use (try) it
  - Freescale, Philips, ST, Infineon, IBM, Intel, …
- Companies sell it
  - Sonics (USA), Arteris (France), Silistix (UK), …

- 1st IEEE Conference: NOCS 2007
  - 102 papers submitted
NoC research community

- Academe and industry
- VLSI / CAD people
- Computer system architects
- Interconnect experts
- Asynchronous circuit experts
- Networking/Telecomm experts
Possible impact: Expect new forms of Rent’s Rule?

- View interconnection as *transmission of messages over virtual wires* (through the NoC)

- Model system interconnections among blocks in terms of required bandwidth and timing
  - Dependence on NoC topology
  - Dependence on the S/W application (in a CMP)
  - Usage for prediction of hop-lengths, router design, ....

NoC is a scalable platform for billion-transistor chips
Several driving forces behind it
Many open research questions
May change the way we structure and model VLSI systems