Brisk and Limited-Impact NoC Routing Reconfiguration

Doowon Lee, Ritesh Parikh and Valeria Bertacco

Computer Science and Engineering
University of Michigan
Digital System Design Trends

Increasing transistor density

- Intel Pentium 4: 1 core
- Intel Core 2: 2 cores
- Intel SCC: 48 cores
- Tilera TILE: 72 cores
- Intel Core i7: 4 cores
- AMD FX: 8 cores
- Oracle Sparc T5: 16 cores
- MIT EM2: 110 cores

Increasing computation capability
Reliability Threats

Increasing transistor density
- Computation capability ↑
- Transistor reliability ↓

Permanent faults during lifetime

Fragile transistors
Implications on Network-on-Chip

Increasing transistor density

- Computation capability $\uparrow$
- Transistor reliability $\downarrow$

Network-on-chip...

- Increasing adoption
- Possible single point of failure

Solution: Network-on-chip reconfiguration
Network-on-Chip Routing 101

• Deterministic routing:
  - Fixed path between each source-destination pair
    – XY routing
    – Routing table

• Adaptive routing

Routing reconfiguration:
Changing routing function upon a fault occurrence
Detouring rules using local connectivity

Cannot tolerate more than a few faults!

Detouring rule:
*If +X is not available, take +Y direction*

Prior Global Routing Reconfiguration

Re-compute all the routing table entries in the network

Our Proposal: BLINC

Brisk and Limited-Impact NoC Routing Reconfiguration

• Quick reconfiguration
• Localized impact
• High fault tolerance

Uninterrupted availability via aggressive network testing
BLINC provides a quick response upon a fault occurrence, while a background re-routing computes new optimal routes.

Previous global solutions

Fault occurrence  
Long interruption

Routing w/ BLINC

Fault occurrence

Our contribution
BLINC provides a quick response upon a fault occurrence, while a background re-routing computes new optimal routes.
**BLINC Procedure**

**Offline computation**

**Network segmentation**: Partitioning a given network topology into smaller segments

**Routing metadata**: A compact representation for routing the segmented network

**Online computation**

**Partial reconfiguration**: Modifying the routing metadata to maintain them up-to-date
Segmentation Process

A deadlock-free routing rule by using turn restriction

Choose a root node

Span a segment (node + link)

Are all nodes visited?

Place turn restrictions

Root node

Two entrances

Hierarchical Tree

Repurpose segmentation for quick reconfiguration

Constructing a hierarchical tree
Use **path diversities** in turn restricted nodes

⇒ Emergency (alternative) route

**Example 1**

Link 1-3 failure (adjacent)

All connected!

**Example 2**

Link 0-1 failure (not adjacent)

{0,2} and {1} will be disconnected

{0,2} and {1} will be disconnected

Move turn restriction

(\textit{now adjacent}!)
Use **path diversities** in turn restricted nodes

⇒ Emergency (alternative) route

**Emergency route rules**

1. **Adjacent to turn restriction**
   ⇒ Take another entrance

2. **Not adjacent to turn restriction**
   ⇒ Change turn restricted node, then take another entrance
Routing Metadata: Port Type

Contain essential information for computing valid routes

1. **Port type**: Direction of output port
   - Parent (toward root)
   - Intra-segment (away from root, same segment)
   - Child (away from root, different segment)
2. **Children set**: Reachable nodes

(along intra-segment or child port)
Reconfiguration Principle

• What reconfiguration does
  1) Disabling invalid routes
  2) Enabling emergency routes

• How?
  – Routing metadata exchange with neighbor routers
  – Removing unnecessary turn restriction

![Diagram of reconfiguration principle]

Example segment

Before localized reconfiguration

Localized reconfiguration impact (failed and parent segments)
Reconfiguration Principle

- **What reconfiguration does**
  1) **Disabling** invalid routes
  2) **Enabling** emergency routes

- **How?**
  - Routing metadata exchange with neighbor routers
  - Removing unnecessary turn restriction

![Diagram showing reconfiguration impact](image_url)

**Localized reconfiguration impact** (failed and parent segments)
Reconfiguration Procedure

1) Swapping port types

2) Enabling emergency routes

3) Disabling invalid routes

In most cases, only a few nodes affected
Recap: Route Computation

Two route computation modules

1. Routing table (baseline)
   - Offline computation
     - Path selection
     - Routing table
     - Table lookup
   - Online computation
     - Normal

2. BLINC reconfiguration module
   - Offline computation
     - Topology analysis
     - Routing metadata
     - Emergency route compute
   - Reconfiguration controller
     - RECONFIGURE
     - Reconfiguration messages to neighbor routers
   - Online computation
     - Emergency
     - Packet forwarding direction
Evaluation: Number of Affected Routers

How many routers need to recompute their routing functions if one additional link failure occurs?

Each number is averaged from 100 fault situations (10 base topologies × 10 failure locations).

Evaluation: Reconfiguration Latency

Latency increases only slightly with network size

- 6x6 mesh: 21 cycles
- 8x8 mesh: 26 cycles
- 10x10 mesh: 30.1 cycles

5 cycles to process a children set message, 1 cycle for acknowledge (hop-by-hop).

more than 50× speedup

ARIADNE
BLINC
BLINC can provide uninterrupted availability under an aggressive network testing scenario.

- Links are deactivated one by one for testing (early detection).

Towards no network downtime

Two solutions compared
1. **BLINC**: packets are re-routed around the tested link
2. **Stall**: packets are stalled at the tested link

Timing diagram & parameters

- **User packets**
- **Testing packets**

- **Testing period (f)**
- **Normal execution period (1-f)**
- **Testing duration of each link (L)**
Average Packet Latency under Testing

Stall shows large latencies.

BLINC shows small latency increases.

Zoomed in Y-axis
Conclusions

• We proposed BLINC: fast, localized NoC routing reconfiguration.

• Three steps
  – Segmentation
  – Routing metadata
  – Partial reconfiguration

• 80% reduction in the number of affected routers.

• Uninterrupted availability by aggressive testing.
Thank you!  Questions?