Fault Tolerance in Networks on Chip

Martin Radetzki, Xueqian Zhao, Chaochao Feng, Axel Jantsch

April 25, 2012
Outline

Introduction
Physical Failure Mechanisms
Data Link Layer
Network Layer
Transport Layer
Summary
Communication Layers and Modules

SW

- higher layers
- NoC application
- NoC operating system
- processing elements
  (cores)

NoC Hardware

- transport layer
- data link layer
- network layer
- packet transmission
  between cores
- flit transmission
  and flow control
- packet routing

- physical layer
- phy
- physical transmission
  of phits
- interconnect
  links

Links between switches and NIs

Routing

Switching

End to end

network interfaces

Nodes

with routing
functionality
Faults, Errors, and Failures

**Physical failure mechanism:** The physics behind faults

**Fault:** Deviation in logic behavior

**Fault Model:** Type of fault, probability, duration, etc.

**Error:** Deviation in information flow

**Masked Errors** are errors that do not become visible

**Module failure**
Redundancy

Information Redundancy

Temporal Redundancy

Spatial Redundancy
Triple Modular Redundancy TMR

Module A

Module B

Module C

Voter
Fault Classes

**Transient faults** Random appearance and disappearance

**Intermittent faults** appear only under certain conditions

- Occur repeatedly at the same location
- Tend to occur in bursts
- Replacement of the faulty component removes the fault

**Permanent faults** occur always but may be masked

- Information redundancy, temporal redundancy, spatial redundancy
- Information redundancy, temporal redundancy, spatial redundancy
- Information redundancy, temporal redundancy, spatial redundancy
- Information redundancy, spatial redundancy
# When and How to Deal with Faults

## Diagnosis

<table>
<thead>
<tr>
<th>Offline</th>
<th>External test vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online disruptive</td>
<td>Built-in self test - BIST</td>
</tr>
<tr>
<td>Online during operation</td>
<td>Continuous monitoring</td>
</tr>
</tbody>
</table>

## Counter measure

<table>
<thead>
<tr>
<th>Offline</th>
<th>Stop - reconfiguration - restart</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online disruptive</td>
<td>Reconfiguration during operation</td>
</tr>
<tr>
<td>Online during operation</td>
<td>Continuous error handling</td>
</tr>
</tbody>
</table>
Physical Failure Mechanisms

- Radiation: Neutrons and Alpha particles
- Electromagnetic interference
- Electrostatic discharge
- Aging

Influencing Factors

- Process variability
- Temperature variability
Radiation

- Neutrons due to cosmic rays
- Alpha particles due to impurities in substrate and packaging materials
Alpha Particle Hits DRAM Cell

5 MeV alpha particle
Alpha Particle Hits DRAM Cell

From (May and Woods, 1978).
Soft Error Rate

Personal communication from Shakar Borkar, Jose Maiz, Intel, April 2012.
Contributions to Chip Level SER

From (Mitra et al., 2005).
Longer and thinner wires increase the cross coupling, leading to conservative design rules.
Aging

- Electromigration
- Hot Carrier Injection
- Negative Biased Temperature Inversion
- Oxide Breakdown
Process Variability

Variability of

- Material impurities
- Doping concentrations
- Geometry

Leads to

- Variation of critical charges
- Different pace of damage built-up
- Variations of fault probabilities across the chip
Temperature Variation

- All aging processes are temperature dependent.
- Different chip parts exhibit higher activity levels $\Rightarrow$ higher temperature.
- Fault occurrence probabilities develop at different rates.

Lifetime of an inverter chain decreases by 2.2x for every 10°C increase in operating temperature due to NBTI (Zhang and Jiang, 2008).
### Overview of Fault Causes

<table>
<thead>
<tr>
<th>Physical cause</th>
<th>Effected by process variation</th>
<th>Fault class</th>
<th>Bursti-ness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radiation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Neutrons</td>
<td>No</td>
<td>Transient logic</td>
<td>Yes</td>
</tr>
<tr>
<td>$\alpha$-particles</td>
<td>No</td>
<td>Transient logic</td>
<td>Yes</td>
</tr>
<tr>
<td>Electromagnetic interference</td>
<td>No</td>
<td>Intermittent delay</td>
<td>No</td>
</tr>
<tr>
<td>Cross coupling of parallel wires</td>
<td>No</td>
<td>Intermittent delay</td>
<td>No</td>
</tr>
<tr>
<td>Self coupling, Skin effect leading to higher resistance</td>
<td>No</td>
<td>Intermittent delay</td>
<td>No</td>
</tr>
<tr>
<td>Aging</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electromigration</td>
<td>Yes</td>
<td>intermittent $\rightarrow$ permanent delay and logic fault</td>
<td>No</td>
</tr>
<tr>
<td>Bias Temperature Instability (BTI)</td>
<td>Yes</td>
<td>intermittent $\rightarrow$ permanent delay fault</td>
<td>No</td>
</tr>
<tr>
<td>Hot carrier injection</td>
<td>Yes</td>
<td>intermittent $\rightarrow$ permanent delay faults</td>
<td>No</td>
</tr>
<tr>
<td>Oxide breakdown/Time Dependent Dielectric Breakdown (TDDB)</td>
<td>Yes</td>
<td>intermittent $\rightarrow$ permanent logic fault</td>
<td>No</td>
</tr>
<tr>
<td>Power density variation and heat flux</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leakage power variation due to temperature differences</td>
<td>Yes</td>
<td>Intermittent $\rightarrow$ permanent delay and logic faults</td>
<td>Yes</td>
</tr>
<tr>
<td>Performance variation due to temperature differences; variations in wear-out effects due to temperature differences</td>
<td>Yes</td>
<td>Intermittent delay failures</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>intermittent and permanent, delays, opens and shorts</td>
<td>Maybe</td>
</tr>
</tbody>
</table>
Outline

Introduction
Physical Failure Mechanisms
Data Link Layer
Network Layer
Transport Layer
Summary
Data Link Layer

- Fault models
- Error detection
- Temporal redundancy techniques
- Information redundancy techniques
- Spatial redundancy techniques
- Summary
Data Link Fault Models - Switch Logic

- **Permanent faults:** Stuck-at faults, Shorts
- **Transient faults:** Single Event Upset, Voltage glitches with exponentially distributed duration

From (Gadlage et al., 2010).
Transient disturbances are modeled as voltage noise and error probabilities are deduced.

\[ \epsilon = \int_{\frac{V_{\text{swing}}}{2\sigma_N}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{(-y^2/2)} dy. \]

\( \epsilon \) ... link error probability
\( V_{\text{swing}} \) ... Voltage swing
\( \sigma_N^2 \) ... Variance of the voltage noise

(Lehtonen et al., 2007; Fu and Ampadu, 2009; Hegde and Shanbhag, 2000)
Maximum Aggressor Fault Model (MAF)

<table>
<thead>
<tr>
<th>Links</th>
<th>Fault Effect</th>
<th>Positive Glitch</th>
<th>Negative Glitch</th>
<th>Falling Delay</th>
<th>Rising Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link 1 (aggressor)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link i-1 (aggressor)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link i (victim)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link i+1 (aggressor)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link n (aggressor)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

expected behavior

faulty behavior
Test Data Generator  
TDG  
router  
TRA  

Test Response Analyzer  
TDG  
router  
TRA  

(Cota et al., 2008; Grecu et al., 2006)
Information Redundancy

Convolutional Codes based on signal history

Block Codes based on fixed sized word

$k$ bit of information

$n$ bit code word

$r = n - k$ bits of redundancy

$(n, k)$-block code

Linear block codes:

$y_1 = \text{encode}(x_1)$

$y_2 = \text{encode}(x_2)$

$y_1 + y_2 = \text{encode}(x_1 + x_2)$
## (7,4)-Hamming Code

<table>
<thead>
<tr>
<th>data</th>
<th>codeword</th>
<th>data</th>
<th>codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000000</td>
<td>0001</td>
<td>0001111</td>
</tr>
<tr>
<td>1000</td>
<td>1000110</td>
<td>1001</td>
<td>1001001</td>
</tr>
<tr>
<td>0100</td>
<td>0100101</td>
<td>0101</td>
<td>0101010</td>
</tr>
<tr>
<td>1100</td>
<td>1100011</td>
<td>1101</td>
<td>1101100</td>
</tr>
<tr>
<td>0010</td>
<td>0010011</td>
<td>0011</td>
<td>0011100</td>
</tr>
<tr>
<td>1010</td>
<td>1010101</td>
<td>1011</td>
<td>1011010</td>
</tr>
<tr>
<td>0110</td>
<td>0110110</td>
<td>0111</td>
<td>0111001</td>
</tr>
<tr>
<td>1110</td>
<td>1110000</td>
<td>1111</td>
<td>1111111</td>
</tr>
</tbody>
</table>

### Parity Calculation

- \( \text{bit2} = \text{bit6} + \text{bit5} + \text{bit3} \)
- \( \text{bit1} = \text{bit6} + \text{bit4} + \text{bit3} \)
- \( \text{bit0} = \text{bit5} + \text{bit4} + \text{bit3} \)
Hamming Codes

\[ v = u \times G \]
\[ s = v' \times H^T \]
\[ = (v + e) \times H^T \]
\[ = v \times H^T + e \times H^T \]
\[ = e \times H^T \]

\[ G \times H^T = 0 \]

\[ u \ldots \text{information word} \]
\[ v \ldots \text{code word} \]
\[ s \ldots \text{syndrome} \]
\[ e \ldots \text{error vector} \]
\[ H \ldots \text{parity check matrix} \ (n - k) \times n \]
\[ G \ldots \text{generator matrix} \ k \times n \]
(7,4) Hamming Code Example

<table>
<thead>
<tr>
<th>data</th>
<th>codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000000</td>
</tr>
<tr>
<td>1000</td>
<td>1000110</td>
</tr>
<tr>
<td>0100</td>
<td>0100101</td>
</tr>
<tr>
<td>1100</td>
<td>1100011</td>
</tr>
<tr>
<td>0010</td>
<td>0010011</td>
</tr>
<tr>
<td>1010</td>
<td>1010101</td>
</tr>
<tr>
<td>0110</td>
<td>0110110</td>
</tr>
<tr>
<td>1110</td>
<td>1110000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6 3 2 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>信息</td>
</tr>
</tbody>
</table>

bit2 = bit6 + bit5 + bit3  
bit1 = bit6 + bit4 + bit3  
bit0 = bit5 + bit4 + bit3

\[
G = \begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1
\end{pmatrix}
\]

\[
H = \begin{pmatrix}
1 & 1 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 & 1
\end{pmatrix}
\]

\[
v = u \times G
= \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1
\end{pmatrix}
= \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}
\]

\[
s = v' \times H^T
= \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
= \begin{pmatrix} 1 + 1 + 1 & 1 + 1 + 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}
\]
Hamming Codes with $d_{\text{min}} = 3$ and $d_{\text{min}} = 4$

**Hamming code with $d_{\text{min}} = 3$:**
For every positive integer $r \geq 3$ there exists a Hamming code with

$$n = 2^r - 1 \quad k = 2^r - 1 - r$$

with a minimum distance $d_{\text{min}} = 3$ and can be used as SEC or as DED code. When used as DED code many multiple bit errors are detected also.

**Hamming code with $d_{\text{min}} = 4$:**
By adding one more parity bit, a Hamming code can be constructed to work as SEC and DED at the same time.
Block Partitioning and Interleaving

- Smaller and faster encoders and decoders
- Need more protection wires
- Interleaving mitigates burst faults
<table>
<thead>
<tr>
<th>data bits</th>
<th>Hamming-SEC</th>
<th>SEC-DED (Hamming/Hsiao)</th>
<th>logic depth of SEC-DED decoder</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hamming-DED</td>
<td>Hsiao</td>
<td>Hamming</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>14</td>
<td>5</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>18</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
</tbody>
</table>
# Protection Trade-off and Service Classes

## Header Protection

<table>
<thead>
<tr>
<th>Code</th>
<th>Wires</th>
<th>Max interl. path</th>
<th>Crit. path</th>
<th>$P_{\text{err,UC}}$</th>
<th>$P_{\text{err,UD}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>no protection</td>
<td>16</td>
<td>-</td>
<td>0</td>
<td>$5.33 \cdot 10^{-8}$</td>
<td>$5.33 \cdot 10^{-8}$</td>
</tr>
<tr>
<td>$1 \times (22, 16)$  SEC-DED</td>
<td>22</td>
<td>5</td>
<td>9</td>
<td>$1.47 \cdot 10^{-11}$</td>
<td>$2.37 \cdot 10^{-18}$</td>
</tr>
<tr>
<td>$2 \times (13, 8)$  SEC-DED</td>
<td>26</td>
<td>9</td>
<td>8</td>
<td>$1.21 \cdot 10^{-15}$</td>
<td>$4.63 \cdot 10^{-23}$</td>
</tr>
<tr>
<td>$4 \times (8, 4)$  SEC-DED</td>
<td>32</td>
<td>16</td>
<td>6</td>
<td>$4.33 \cdot 10^{-16}$</td>
<td>$1.32 \cdot 10^{-23}$</td>
</tr>
</tbody>
</table>

## Payload Protection

<table>
<thead>
<tr>
<th>Mode</th>
<th>Code</th>
<th>Bandwidth</th>
<th>$P_{\text{err,UC}}$</th>
<th>$P_{\text{err,UD}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximum bandwidth</td>
<td>none</td>
<td>102</td>
<td>$3.36 \cdot 10^{-7}$</td>
<td>$3.36 \cdot 10^{-7}$</td>
</tr>
<tr>
<td>guaranteed integrity</td>
<td>$4 \times (25, 20)$ DED</td>
<td>64</td>
<td>$9.82 \cdot 10^{-8}$</td>
<td>$1.59 \cdot 10^{-17}$</td>
</tr>
<tr>
<td>minimum latency</td>
<td>$4 \times (25, 20)$ SEC</td>
<td>64</td>
<td>$8.64 \cdot 10^{-11}$</td>
<td>$8.64 \cdot 10^{-11}$</td>
</tr>
<tr>
<td>high reliability</td>
<td>$3 \times (34, 27)$ SEC-DED</td>
<td>65</td>
<td>$5.88 \cdot 10^{-10}$</td>
<td>$3.20 \cdot 10^{-12}$</td>
</tr>
</tbody>
</table>

(Zimmer and Jantsch, 2003, 2004)
Temporal Redundancy

(Ravindran, 2009)
Data Link - Razor FlipFlops

(Sender) → Pipelined buffer → Pipelined buffer → Receiver

\[
\text{Main Flip Flop} \\
\text{Shadow Latch}
\]

(RAZOR FF)

(0, 1, CLK, ERROR, Comparator)

(CLK_DELAYED)

(Ernst et al., 2003; Tamhankar et al., 2005)
Hop2Hop Retransmission

(Murali et al., 2005)
64 data bit

- protected by $4 \times (21, 16)$ Hamming code
- In error mode the data is transmitted in two cycles over the correct parts of the link

(Lehtonen et al., 2007)
- 64 data bit
- protected by $4 \times (21, 16)$ Hamming code.
- 4 spare wires are available, one for each 32 bit block.
- In error mode the link is reconfigured and the spare wires used.

(Lehtonen et al., 2007)
4 Phases:

1. Fault occurrence
2. Error detection
3. Fault vector generation
4. Mask vector generation, retransmission and flit-reassembly

(Vitkovskiy et al., 2010)
Data Link vs. End-to-End Protection

- Link layer payload protection: $4 \times (25, 20)$ SEC-DED Hsiao block code; 125 bit in total;
- End-to-end payload protection: $1 \times (94, 80)$ SEC-DED Hsiao block code; 119 bit in total;
- Power comparison considers wires, encoders, decoders, and is normalized to the same performance;

(Vitkovski et al., 2008)
Data Link Summary

- **Fault models:**
  - Stuck-at
  - Maximal Aggressor Fault Model (MAF)
  - Voltage noise

- **Fault detection:**
  - Off-line based on BIST
  - on-line based on coding
  - temporal and spatial redundancy (e.g. TMR)

- **Correction:**
  - Codes
  - Retransmission
  - Reconfiguration of spares


References - cont’d


3 Network Layer

Martin Radetzki
martin.radetzki@informatik.uni-stuttgart.de
Dept. of Embedded Systems Engineering
www.iti.uni-stuttgart.de

Supported by the Deutsche Forschungsgemeinschaft (DFG) under grant Ra1889/4-1
NoC Layers

cf. ISO/IEC 7498-1:1994 (OSI seven layer model)
Path Redundancy

Number of *minimal* paths from \((x_s, y_s)\) to \((x_r, y_r)\) in 2-*mesh*:

\[
\begin{pmatrix}
\Delta x + \Delta y \\
\Delta x \\
\end{pmatrix} = \frac{(\Delta x + \Delta y)!}{\Delta x! \cdot \Delta y!}
\]

where

\[
\Delta x = |x_s - x_r| \\
\Delta y = |y_s - y_r|
\]

Path redundancy can be exploited in different ways:
- “on demand“
  - spatial redundancy
- “by packet replication“
  - information redundancy

No temporal redundancy
3.0 Overview

3.1 Fault models and fault diagnosis

3.2 Information redundancy techniques

3.3 Strictly local adaptation

3.4 Fault regions

3.5 Fault lookahead

3.6 Global topology exploration

3.7 Offline routing adaptation

Spatial redundancy techniques further subdivided according to the extent of fault information available to a single switch.
3.1 Fault models and fault diagnosis

**Link fault**
a link is unavailable
- unidirectionally
- bidirectionally

**Switch fault**
a switch is unavailable
= faults of all incident links

- Link fault
- Switch fault
3.1 Fault models and fault diagnosis

Throughput may break down even at low failure rates

- employ fine-grained fault models
- utilize remaining functionality of partially-defective switches

Example: 8x8 mesh, uniform random traffic, fault tolerance through deflection
NoC Specific Fault Models

**Port stuck-at faults** model the failure of individual switch ports (instead of the switch as a whole).


**Crossbar faults** model the failure of individual connections from input ports to output ports. Switch status is captured with a **fault matrix**.

Fault diagnosis

Error detection identifies erroneous information.

Fault diagnosis determines the cause of errors, including fault localization and possibly fault classification (transient? intermittent? permanent?)

Fault diagnosis can be classified as online concurrent, online preemptive (e.g. BISD) and offline (e.g. by analyzing production test responses).

Structural offline diagnosis techniques have been applied to NoC switches to diagnose port stuck-at faults.


3.1 Fault models and fault diagnosis
Diagnosis with Boundary Test Access

A technique to diagnose link faults and crossbar faults with directed test packets. Each test packet either passes (arrives at destination without changes) or fails. A diagnostic tree locates the actual fault from the pass/fail information (centralized).

Concurrent online diagnosis is performed with regular data packets. To identify packet corruption, EDC code checking is required (in each switch).

Parity check to detect and locate link and switch faults.

Packet CRC checks and error counters allow to detect and classify functional **crossbar faults**. Additional preemptive pattern-based testing before permanent disabling of a connection.

Information Redundancy Techniques

**Multiple copies** of each packet are sent out to increase the likelihood of at least one packet arriving at its destination intact. Copies must take **disjoint paths**, otherwise **single point of failure**.

Significant overhead, 2x or worse (1 copy is the least…). Ensuring disjoint paths without deadlock can be difficult. Packets that do not arrive have to be discarded eventually (TTL, EDC).

Other terms: probabilistic routing, stochastic communication.

Flooding, in its extreme form, lets switches replicate each incoming packet to each of the $k$ output ports. In $d$ routing steps, $k^d$ copies are produced. Large loss in effective packet throughput or cost of overdesigned NoC traded off against fault tolerance. Flooding with test packets may also be used as a basis for fault diagnosis.

Gossip flooding replicates packets with a given probability, $p < 1$, only (instead of always).

Directed flooding distinguishes between productive outputs (which bring the packet closer to its destination), and non-productive ones. Productive outputs are favored with a packet replication probability, $p^+$, that is higher than the probability for non-productive outputs, $p^-$. 


The extreme case, $p^+ = 1$ and $p^- = 0$, keeps a wavefront of packets traveling in a rectangular region between the sender and the receiver. Copies in $O(d)$ if switches eliminate received identical copies.

However, no fault tolerance if sender (S) and destination (D) share the same x or y coordinate.
Random walk replicates packets at the sender only. Thus, the number of packet copies is constant, independent from the distance to the destination.


The minimal overhead of 2x can be achieved. However, path diversity (while ensuring deadlock-freeness) is essential.

This can be achieved with two virtual channels that employ disjoint routing schemes.

Strictly Local Adaptation

**Strictly local** techniques add resources (=> spatial redundancy) to enable switches to tolerate faults of their own (or of their incident links). A switch requires knowledge of its own fault status (and of incident links) only. Thus, no fault information has to be distributed (transmitted). However, switches cannot route proactively to avoid remote faults.

**Vicis:** internal resource sharing (FIFO buffers), port swapping, escape bus. If a fault cannot be contained in a switch, a higher-level rerouting takes over.

**Strictly Local Adaptation (2)**

**BulletProof**: application of generic fault tolerance techniques (error detection and reconfiguration, spares) on switch and component level (arbiter, routing).


**RoCo**: A faulty switch can request arbitration and routing computation services from its neighbors (“modular router” architecture). A bypass path is available to avoid faulty switches and buffers since these are not shared externally.


**Self-correcting arbitration**: Blocks for detection of erroneous routing and arbitration results; trigger recomputation and correction actions (involving temporal / info redundancy).

Strictly Local Adaptation (3)

**DBP**: Switches have internal **default backup paths**. These are connected to form a ring topology if the regular topology has faulty resources.

![Diagram of DBP mechanism](image)


3.3 Strictly local adaptation
Strictly Local Adaptation (4)

**Redundant multiplexers & sharing of FIFO buffers:**

**Spatial and temporal redundancy of MUXes:**

**Crosspoint redundancy:**

**Redundant physical channels:** Each switch has dual physical channels. When failure of the one channel is detected, the other takes over.
Local Routing Adaptation

Routing that avoids use of defective links:

Routing to avoid faults captured by the crossbar fault model & diagnosis:
Based on Nostrum (KTH). If the normal routing for a packet would send it through a defective part of the switching path, the packet is deflected instead. Exploration of deflection alternatives so that low priority packets don’t run into faults.
[KOHLER et al, 2009 (NoCS), 2010 (TCAD).]

<table>
<thead>
<tr>
<th>Packet</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>max (4)</td>
<td>high (3)</td>
<td>low (2)</td>
<td>min (1)</td>
</tr>
<tr>
<td>Input port</td>
<td>N</td>
<td>E</td>
<td>S</td>
<td>W</td>
</tr>
<tr>
<td>Preferred</td>
<td>S, W</td>
<td>N, W</td>
<td>N</td>
<td>E</td>
</tr>
<tr>
<td>Alternative</td>
<td>E</td>
<td>S</td>
<td>W,E</td>
<td>N,S</td>
</tr>
<tr>
<td>Reflection</td>
<td>N</td>
<td>E</td>
<td>S</td>
<td>W</td>
</tr>
</tbody>
</table>

Table showing routing preferences for different packets and priority levels.
Routing Around Fault Regions

A **fault region** is a connected subset of NoC switch nodes that includes at least one faulty node. Routing around fault regions requires knowledge of the *region boundaries* only. It is sufficient if the region boundary is known to its *directly adjacent* switches. Often there are requirements on region shape, e.g. rectangular or convex.

**Challenges** include:
- To determine fault regions, especially if this is to be done online.
- To make sure that the “routing around” interacts with the normal routing of the NoC in a deadlock-free manner.
- To keep the fault region as small as possible; ideally, a fault region should contain faulty switches only.

**Note**: the term “fault region” should be distinguished from “region based routing”.

3.4 Fault regions
Smallest Possible Fault Region: Single Switch

Classic result: in an $n$-mesh, $n-1$ failed switches can be tolerated w/o losing intact ones. Constructive proof: a variant of negative-first routing.


Local turn model
- No 180° turns allowed
- Straight-through routing permitted
- Of eight 90° turns, at least two forbidden
  ⇒ No cyclic dependencies possible
  ⇒ Deadlock-free routing

(negative first routing)
No adaptivity at the W and N boundary. Solution: allow deviation from the default routing policy (other turns). => Deadlock?

*Intuitively:* fault disables other turns. *Formally:* need suitable proof.

**Global resource numbering scheme** \( f(x,y,p,io) \); allocation limited to strictly increasing order (below: not strictly)
Cycle-free contours

Idea: The immediate neighbours of a failed switch form a shell that isolates the fault (contour). Packets that would be routed through the failed switch first reach the contour which takes care of routing around the fault. Resource dependency cycles must be avoided (cycle free contour).

Limitations: only a single cycle-free contour has been evaluated.

**More dimensions:** each additional mesh dimension adds one degree of freedom that can be used to route around one additional fault.  
[Glass and Ni’s method: \( n-1 \) faults tolerable in \( n \)-mesh.]

**Virtual channels (VC):** provide additional degree of freedom for deadlock-free routing. When choice of increasingly-numbered resources is exhausted in one VC, lift the packet into the next VC (VCs numbered, too).

**Larger fault regions:** multiple failed switches are handled as a single faulty block that may include intact switches, too.
Oversized IP blocks that break the regular structure of a mesh have been circumvented with odd-even based routing. However, not all placements are permitted.


The odd-even turn model provides balanced adaptivity. All turns allowed (but not everywhere).
Use of Virtual Channels

**Dimension order routing** (e-cube, XY) is used as a basis, with one VC per dimension.

The model of **link faults** is employed.

**f-cube-2 routing**: In the VC for X (Y), packets can move up or down (left or right) to surround obstacles without deadlock.

Within each dimension, packets travelling the opposite direction use **disjoint subnetworks**.

If two f-rings (intact switches and link that enclose a rectangular fault region) overlap, E→W, W→E subnets and N→S, S→N subnets are no longer disjoint.

=> f-cube-4 routing with 4 virtual channels, one per subnet
Reducing Virtual Channels

Controlled by a flag in each router, use either clockwise surround routing for $E \rightarrow W$ messages and counterclockwise for $W \rightarrow E$ or vice versa.

- $E \rightarrow W$ and $W \rightarrow E$ become disjoint without VC differentiation
- Support for some fault shapes beyond rectangles
- When not affected by faults, packets can use all VCs (XY turns added)

Reducing Virtual Channels (2)

Combine each X subnet with one Y subnet (=> 2 VCs) and add one VC for fault-free XY routing. Add a flag to each packet to represent the subnet it is traveling in.

- Restriction on (counter)clockwise orientation can be lifted


Virtual channel 0 & 1 (X)

Virtual channel 2 & 3 (Y)

VC 1

E→W, N→S

VC 2

W→E, S→N

Virtual channel 0

XY routing until occurrence of fault

3.4 Fault regions
No Virtual Channels

Row first (RF) routing to W destinations. Otherwise, column first (CF) routing, \( x \neq x_d \) or column only (CO) routing, \( x = x_d \).

RF \( \Rightarrow \) CF \( \Rightarrow \) RO

f-ring/chain surround routing turns can form cycles, but are never aligned to actually cause a deadlock.


Smaller fault regions by allowing more overlap between fault rings. Nodes on overlapped rings must not inject packets, but can still route and receive.

Convex fault regions are formed with the shape of overlapping rectangles. The **odd-even turn model** is used as the basis for routing. This enables the use of all turns, deadlock-free **without VCs**. Fault regions are **joined** if they do not have at least two columns of intact switches between them. The intact “connector” nodes are deactivated.

Number of deactivated nodes can be reduced by defining oddness and evenness of nodes based on the **sum** of their (x,y) coordinates instead of x alone. More even distribution of odd and even and their allowed turns (checkerboard).


Previously: fault region formation is based on interaction of direct neighbors. Now: transmit more remote fault information up to a distance of \( N \) hops.

Design choices: deflection routing, store-and-forward (SAF) switching, \( N = 2 \), off-band transmission of fault information (dedicated wires)

Fault Shapes for Lookahead of 2

(a) Permitted shapes incl. some concave shapes
(b) not permitted two subsequent concave points

Global Topology Exploration

Previously: interaction between switches up to $N$ hops apart.  
**Now:** global exchange of fault information; each switch has the complete picture.

**Online techniques**

**Distance vector routing**

Each switch, $i$, stores distance (hop count, delivery time) information for sending a packet via neighbor $x$ to destination switch $j$: 
\[ d(i, j, x) = d(x, j) + \text{hop\_delay}. \]

The neighbor sends its own distance, 
\[ d(x, j) = \min_y d(x, j, y). \]

Cycles $\Rightarrow$ count-to-infinity problem  
$\Rightarrow$ Link state routing

**Q routing**

Switches “learn” distance information only gradually, controlled by learning factor $\alpha$:
\[ Q^i_{jx} := (1 - \alpha) Q^i_{jx} + \alpha \left( t_{ix} + \min_y Q^x_{jy} \right) \]

The role of Q is similar to the role of d in distance vector routing.
Q Routing

### To Via
<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>E</th>
<th>S</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>9</td>
<td>17</td>
<td>15</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### To Via
<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>E</th>
<th>S</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>13</td>
<td>13</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>14</td>
<td>14</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>...</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>

- **Ports to adjacent switches**
- **Q values from adj. switches**

*3.6 Global topology exploration*
Distance vector routing with in-band transmission (special flits). “Force directed wormhole routing” (FDWR) algorithm.


Q routing in dynamically changing networks on FPGA.


Q routing for fault tolerance in networks-on-chip; off-band transmission.


Hardwire knowledge of fault-free base topology in switches; transmit differential information \( \Delta Q = Q_{\text{faulty}} - Q_{\text{fault-free}} \) only; allow topologies other than mesh.

Hierarchical Routing Tables

Each switch keeps complete Q info only for other switches in its own quadrant. Other quadrants are represented by 3 further table entries. [Feng 2010]

With static partitioning, a border switch knows nothing about some immediate neighbors.

[Radetzki 2011] introduces partitions that are individual (relative) to each switch.

This is enabled by the relative nature of the ΔQ values.
Preemptive Exploration

**Immunet**

fault occurrence => spanning tree construction => fallback ring network depth-first traversal


Instead of ring topology, up*/down* routing could be used along the spanning tree.
Ariadne

*Exploration phase*
A switch that detects an error has ID 0, sends out exploration packets with ID 1. Other switches accept the first (smallest) incoming ID, send out packets with ID++. Deadlock-free routing along IDs in up/down order. The switches form a DAG rather than a spanning tree.

*Route-finding phase*
Each switch sends packets to all destinations. Each switch records the direction $d$ of the first incoming packet from $s$ as routing table entry.

Can deal with all connected topologies.

Routing Tables for Offline Techniques

**Standard routing table**
1 entry {N,E,S,W} per destination

**Turns table**
Entries only for destinations which (may) require a turn.


**Region based routing**
Entry represents a set of destinations reachable in the same way. Simplified encoding of rectangular regions.


3.7 Offline techniques
Routing Tables for Offline Techniques (2)

**XY deviation table** [BOLOTIN et al. 2007]
Combinational implementation of the basic routing policy; deviations stored in table.

**Logic-Based Distributed Routing (LBDR)**
Each switch has two routing bits per output port to store permitted turns in neighbor switches. Connection bit represents availability of a link.
Switch also uses information on packet destination’s relative distance. Derouting bits to support fault tolerance; several variants (eLBDR, uLBDR).

Example: XY Routing Represented with LBDR

3.7 Offline techniques
Offline Routing Computation

**Segment based routing**
The topology is (re)constructed iteratively with single cycle segments. In each segment, one bidirectional turn is disabled.


---

3.7 Offline techniques
Rerouting rules

Removal of forbidden turns in presence of faults; heuristic choice.

![Diagram of rerouting rules](image)

Research emphasis on fault-tolerant, deadlock-free wormhole routing.

Often, regular fault-free topology assumed; predominantly mesh.

Distributed routing provides cost-efficient solutions.

Global approaches are more flexible but require routing tables.

Network / routing reconfiguration is effective against permanent faults only.

Little research on broadcast / multicast routing (e.g. for cache coherency).

Often, coarse-grained fault models and diagnosis not described.
4 Transport Layer

with contributions by
Chaochao FENG

KTH Stockholm and
Nat’l Univ. of Defense Technology, China
NoC Layers

SW

higher layers

NoC application
NoC operating system

Processing Resources
(Cores)

transport layer

network layer

data link layer

network layer

data link layer

network layer

packet transmission between cores

network interfaces

packet routing

flit transmission and flow control

switches with routing functionality

physical layer

phy

phy

physical transmission of phits

interconnect links

cf. ISO/IEC 7498-1:1994 (OSI seven layer model)
4.0 Overview

4.1 Fault models and fault diagnosis

4.2 Information redundancy

4.3 Time redundancy

4.4 Hybrid techniques

4.5 Spatial redundancy
Packet corruption
A packet arrives at the correct destination node, but the payload differs from the data originally sent.
Detection: Error-detecting codes (EDC)

Packet misrouting
A packet arrives at a node different from the one it was originally destined to. Reason: corruption of address information in the packet header.
Detection: Error-detecting codes, mismatch in message ID

Packet loss
A packet does not arrive at all; instead, it is dropped, stuck or excessively delayed.
Detection: Gap in the sequence of packet IDs belonging to a message

System-level fault models for NoC and relation to low-level faults:
**Fault Diagnosis (Localization)**

**Probabilistic method**

NIs send information on positions of bit errors to a host CPU. CPU increments error counter for all links traversed by the packet. Maximum indicates most likely fault.


---

(a) Packets routed in mesh

(b) Error pattern on packet G

(c) List of suspected bit positions

<table>
<thead>
<tr>
<th>Packet name</th>
<th>Suspected bit positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet G</td>
<td>12</td>
</tr>
<tr>
<td>Packet B</td>
<td>3, 12</td>
</tr>
<tr>
<td>Packet R</td>
<td>2</td>
</tr>
</tbody>
</table>

(d) Scoreboard of suspected wires
4.2 Information redundancy

- Information redundancy
- Additional wires or flits
- or reduced payload

**Error Detecting / Correcting Codes**

- Convolutional Code
  - Polynomial(s)
    - NSC
      - Viterbi Decoding
    - RSC
      - Turbo Decoding

- Block Code
  - Parity
  - Hamming Variants
    - linear Code Matrix
    - cyclic Generator Polynomial

- ECC / EDC
  - Reed Solomon
    - Algebraic Decoding
  - BCH
    - Algebraic Decoding

- Reasonable in NoCs
Error Detection

**Parity bit**
Detects all numbers of bit errors.

**Multi-dimensional parity scheme**
Enables detection of double errors.

**CRC code**
Detects all odd numbers of bit errors.
Detects all error bursts up to the length of the code polynomial.
Further capabilities depend on the choice of the coding polynomial.

\[
\text{CRC-16: } x^{16} + x^{15} + x^2 + 1
\]
Sender adds redundant information (ECC) to enable error correction in the receiver with forward communication only.

**Comparison of transport layer vs. link layer error protection**

End-to-end FEC is found more power-efficient than link-level FEC. Trade-off with latency has to be considered, depending on error rate.


**Comparison of Hamming, BCH and RS codes in NoC context**

Comparison of SECDED code with ECC (parity, CRC) and retransmission

Forward Error Correction (FEC)

Automated Repeat reQuest (ARQ)

Receiver requests retransmission of corrupted packets via a backward path.

**Sender**
- EDC encoder
- Packet buffering for future retransmission
- Receiving ACK => free buffer
- Receiving NACK => retransmit
- Timeout waiting on ACK/NACK (packet lost) => retransmit

**Receiver**
- EDC decoder
- Packet OK => send ACK
- Packet corrupt => send NACK
- Packet reordering buffer vs. Go-Back-N policy
- Timeout waiting on missing packets (packet lost) => send NACK
**ARQ Protocol With Go-Back-N Flow Control**

**Receiver**: All packets with ID > corrupt-packet-ID are dropped.

**Sender**: The NACK’ed packet and all packets with a higher ID are resent.

S: Sender  
R: Receiver  
A: ACK  
N: NACK  
D: Drop  
X: Transient error

Hybrid ARQ/FEC (HARQ) schemes employ FEC protection against a limited number of transient or permanent errors. Only if correction capacity is exceeded, ARQ retransmission is employed.

- Low latency in the more frequent case (few errors).
- Excessive coding overhead for rare cases avoided at cost of higher latency.
- Requires capability to detect errors beyond the correction limit.

Comparative study of ARQ, FEC, HARQ
- Evaluation of performance, fault tolerance and energy efficiency trade-offs.


Adaptive HARQ scheme
- Depending on network state, selects between detection (ARQ), correction (FEC) and mixed mode (HARQ).

Source routing can be employed to exploit path diversity for routing around permanent faults. Processing capability of the attached PE can be used to perform more sophisticated diagnosis and adaptive routing algorithms.


Multiple network interfaces for a single PE help ensure that no single NI or switch failure can disconnect a PE.


Internal redundancy and reconfiguration contain faults inside an NI.

Retransmission without path diversity helps against transient faults only.

Lightweight FEC (SECDED) can handle all fault classes well if error bursts are a rare event.

Hybrid scheme: if errors exceed FEC capacity, retransmit.

Path diversity can be achieved with source routing or on network layer.

Room for more research, especially on interaction between techniques. What is the best combination of FEC, retransmission and path diversity?
5 Conclusion
Conclusion

Large body of fault tolerance and interconnection network research.

Specifics of NoC: on-chip implementation, no replacement of faulty parts.

No fault-tolerant NoC system solution yet.

Need clever combination of techniques on multiple layers.

Need solutions that integrate reconfiguration with diagnosis.

Need accurate, quantitative, and fine-grained fault model.

Methodology: NoC optimization in FT context – objective functions and constraints.

Standard evaluation (benchmarks, fault patterns) for comparability.

Going beyond meshes? For sure, going beyond 8x8!

**FT NoC design to be holistic, goal-directed, linked to physical failure causes.**