Concepts for Robust NoC Communication

Martin Radetzki
Department of Embedded Systems Engineering
Institute of Computer Architecture and Computer Engineering
Universität Stuttgart

www.iti.uni-stuttgart.de/ese.phtml
martin.radetzki@informatik.uni-stuttgart.de

ITI
Institut für Technische Informatik
Universität Stuttgart

www.simtech.uni-stuttgart.de

Future Challenge: Reliability

"Within a decade we will see ***100 billion transistor chips***. That is the good news. The bad news is that ***20 billion*** of those transistors ***will fail in manufacture*** and a further ***10 billion will fail in the first year*** of operation."

[Steve Furber, "Living With Failure", Proc. European Test Symposium, pp. 4-8, 2006]

Reliability Issues

Single Event Effects

e.g. ion, neutron impact

causing transient faults

Variability

manufacturing-induced variations, e.g. in wire thickness, transistor parameters

causing permanent and inter-mittent faults that must be tolerated for acceptable yield

Stress & Ageing

e.g.Electromigration, HCI, NBTI

causing permanent faults that appear during system operation

Environmental Impacts

make everything worse...

e.g. accellerated ageing under high temperatures, high levels of radiation on space missions

Failure – Fault - Error

Terminology

Failure mechanism

- Physical cause

Fault (model)

- Formal description

Error

- Effect on information

Examples

- Single-event effect
- Fabrication defect
- Chip ageing

- Transient fault
- Permanent fault (e.g. stuck-at)
- Delay fault

- Soft error
- Recurring data corruption
- NoC: Packet loss

Steps for Fault-Tolerance

Offline | **Online**

Testing

Error detection

Diagnosis (model)

- Fault location
- Fault classification

Correction

- Methods depend on the class of fault

Examples on various layers

- DL: Syndrome ≠ 0
- NL: Packet corruption
- TL: Packet loss

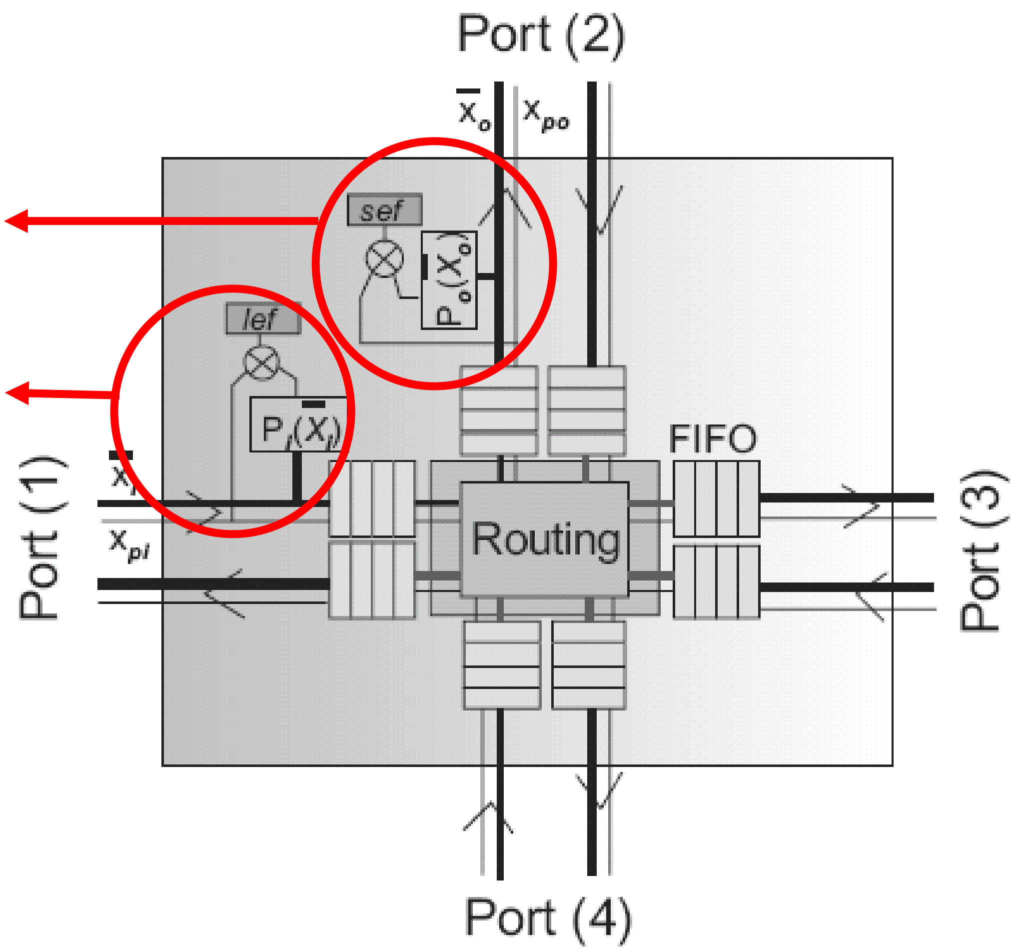
- DL: Analyze syndrome
- NL: Observe the regular traffic
- TL: Inject test packets

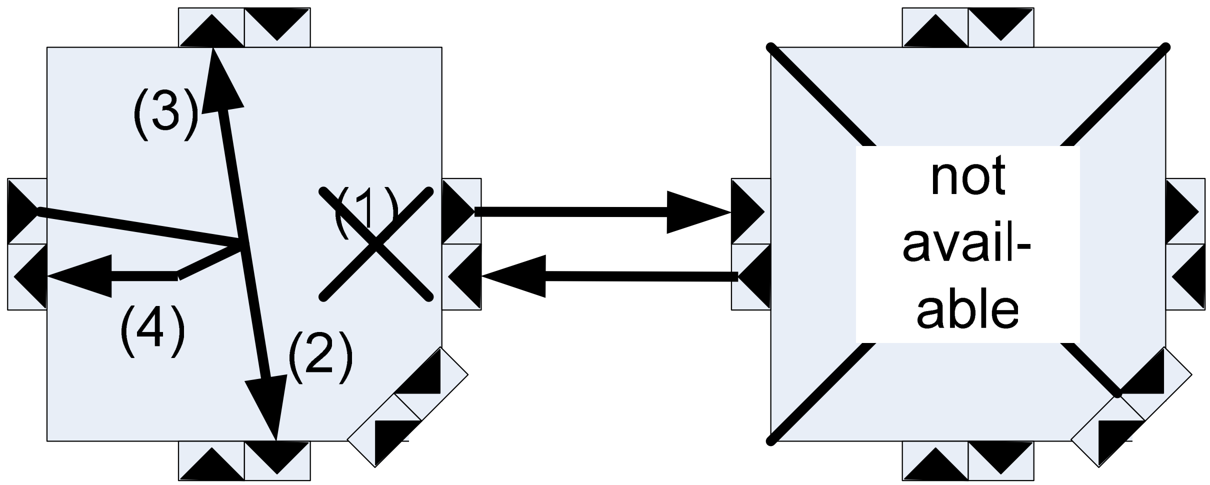
- DL: Toggle erroneous bits
- NL: Adapt routing to bypass fault
- TL: Re-send packet (ARQ)

- Fault Models and Fault Diagnosis
- Fault-Adaptive Routing
- Experimental (Simulation) Results

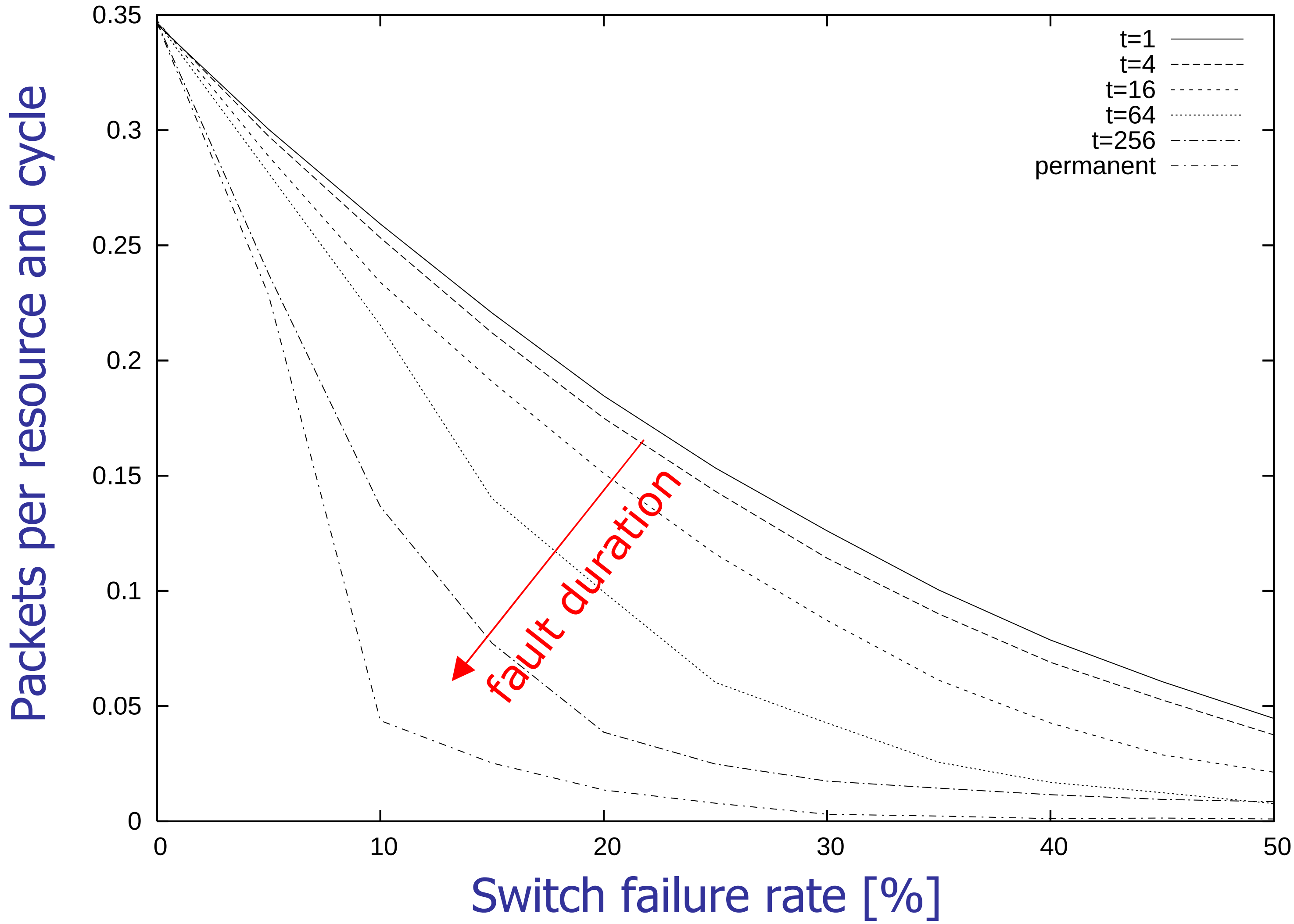
e.g. [Grecu, Pande 2006]

- Parity checks at switch inputs and outputs
- Output error => diagnosis of switch fault
- Input error => diagnosis of switch fault
- Drop and retransmit packet in case of error
- Does not help against permanent faults





- Switch faults: the complete switch is assumed to be unavailable
- Link faults: refers to a link between two switches (and/or a directional output of a switch)

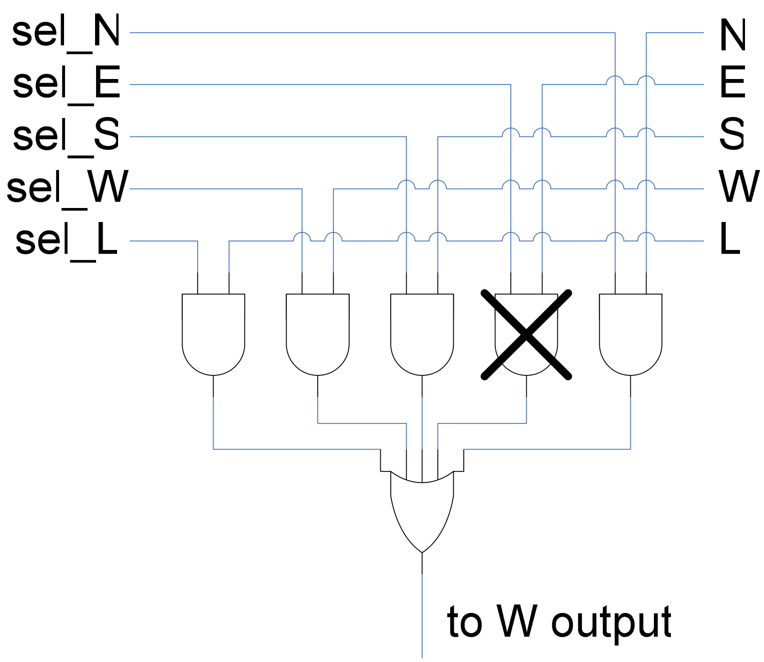


8x8 mesh

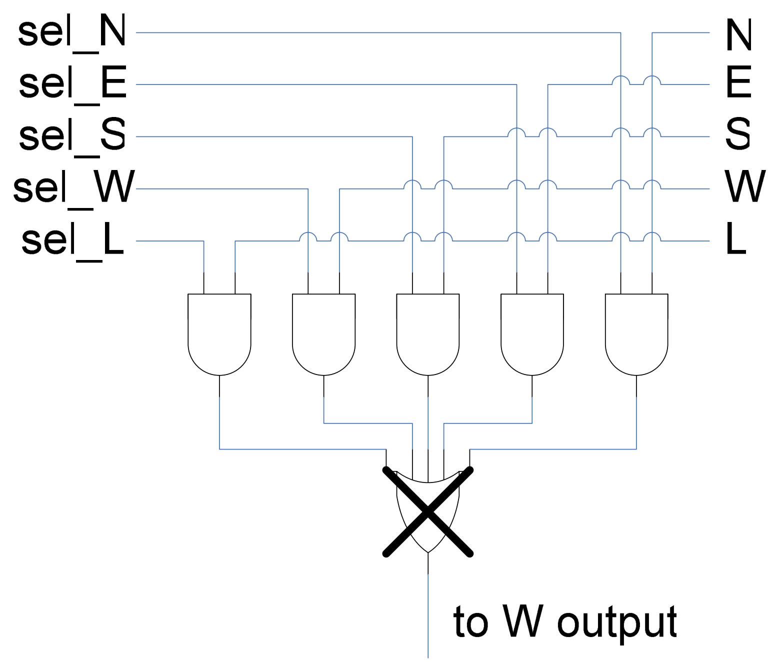
- Switch faults: Throughput breaks down even at low failure rates

Structural Fault Model

- One multiplexer, part of a crossbar switch:



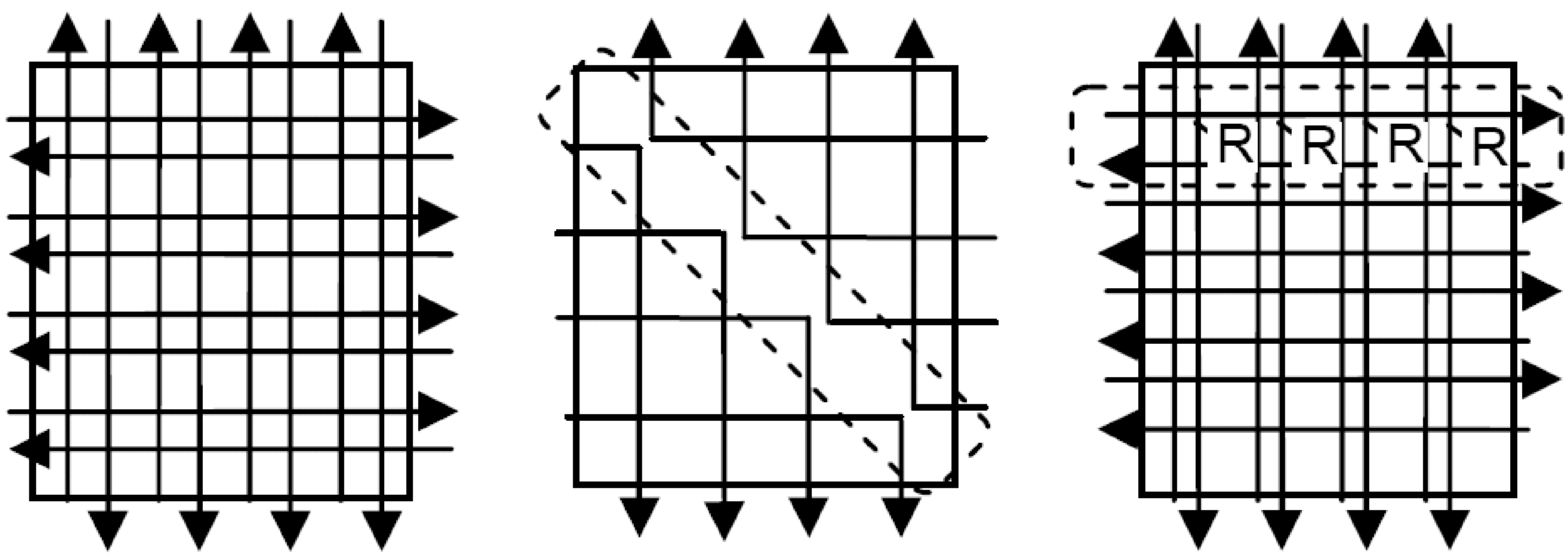
structural defect causes a single connection to fail (E → W)



structural defect invalidates a complete link (→ W)

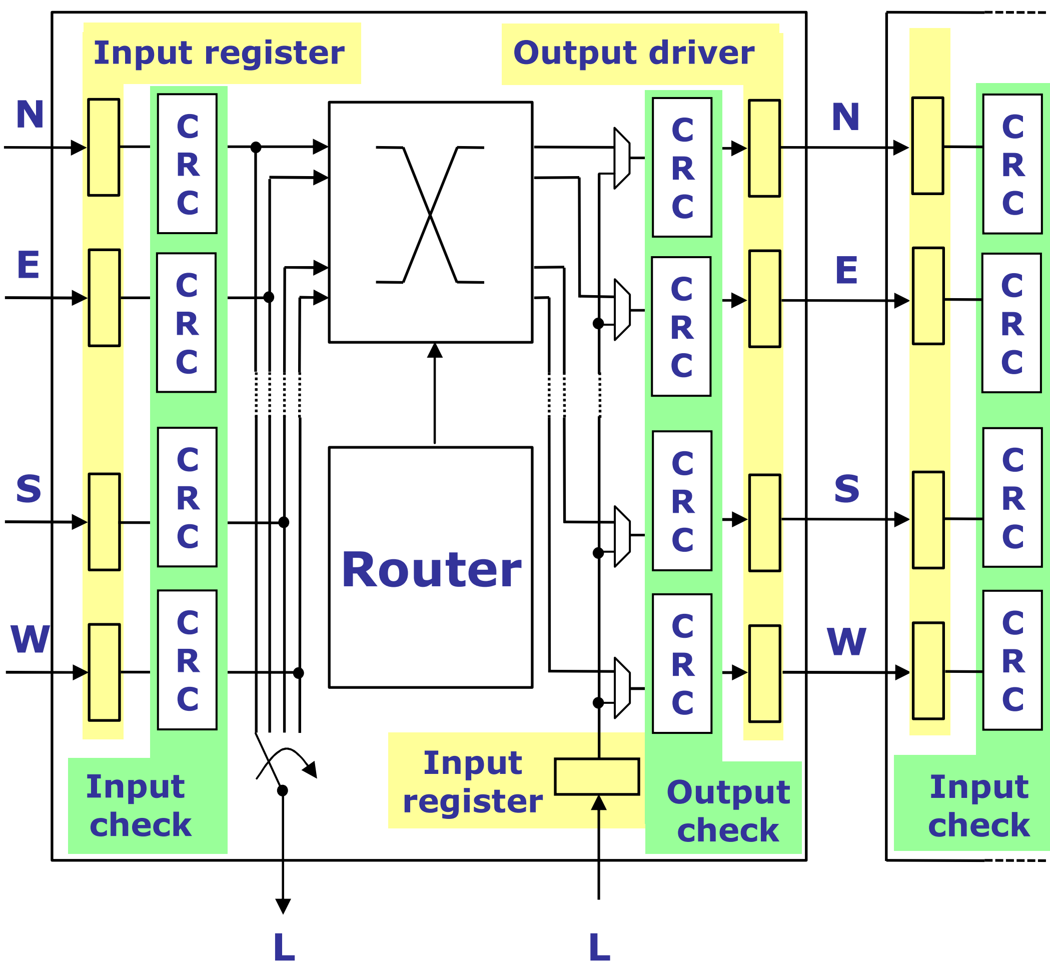
One approach [Raik, Ubar 2007]:

- Sending directed test patterns
- Assumption: Links are fault-free
- Diagnostic access at the network boundaries
- Central control, not distributed (scalability?)
- Regular NoC operation needs to be interrupted (offline)



Proposed Diagnostic Infrastructure

Nostrum switch

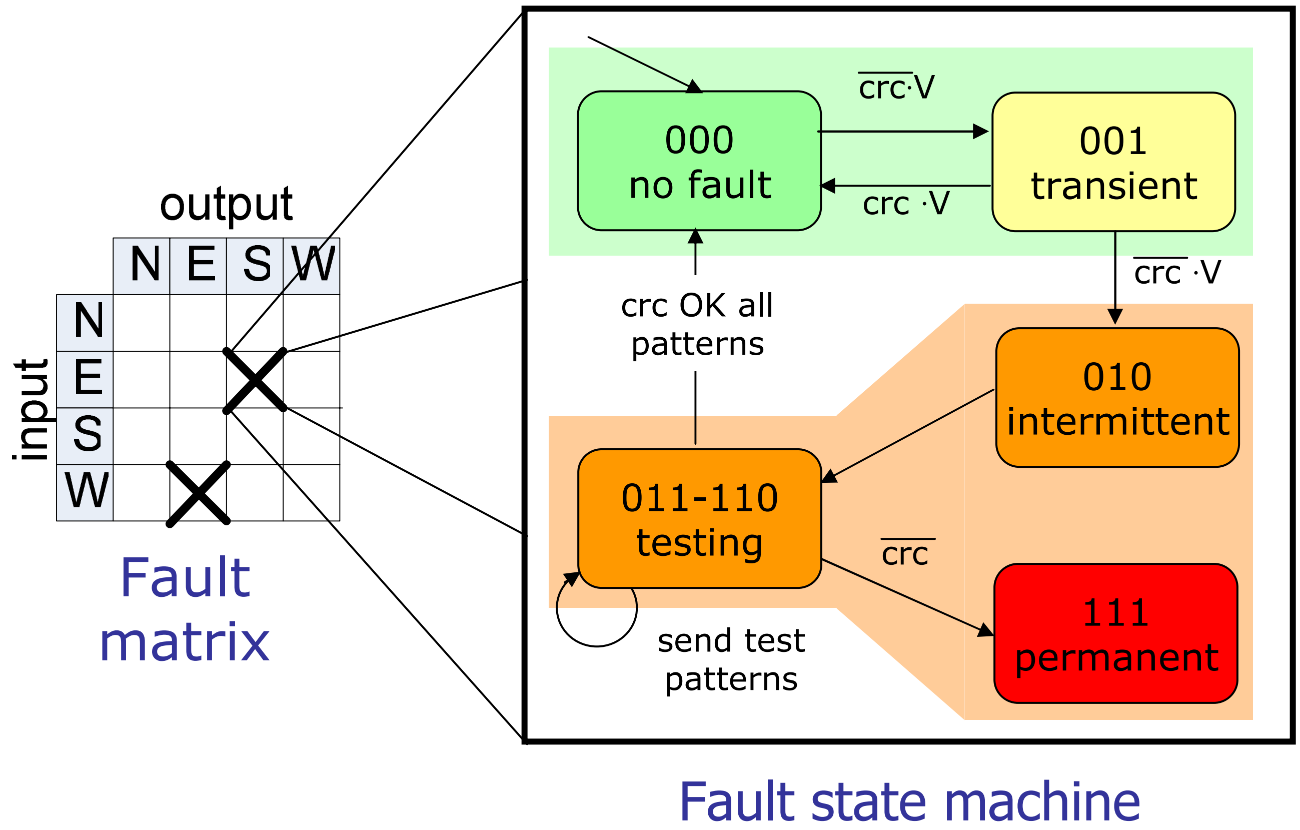


- Packet format extended by a CRC checksum:

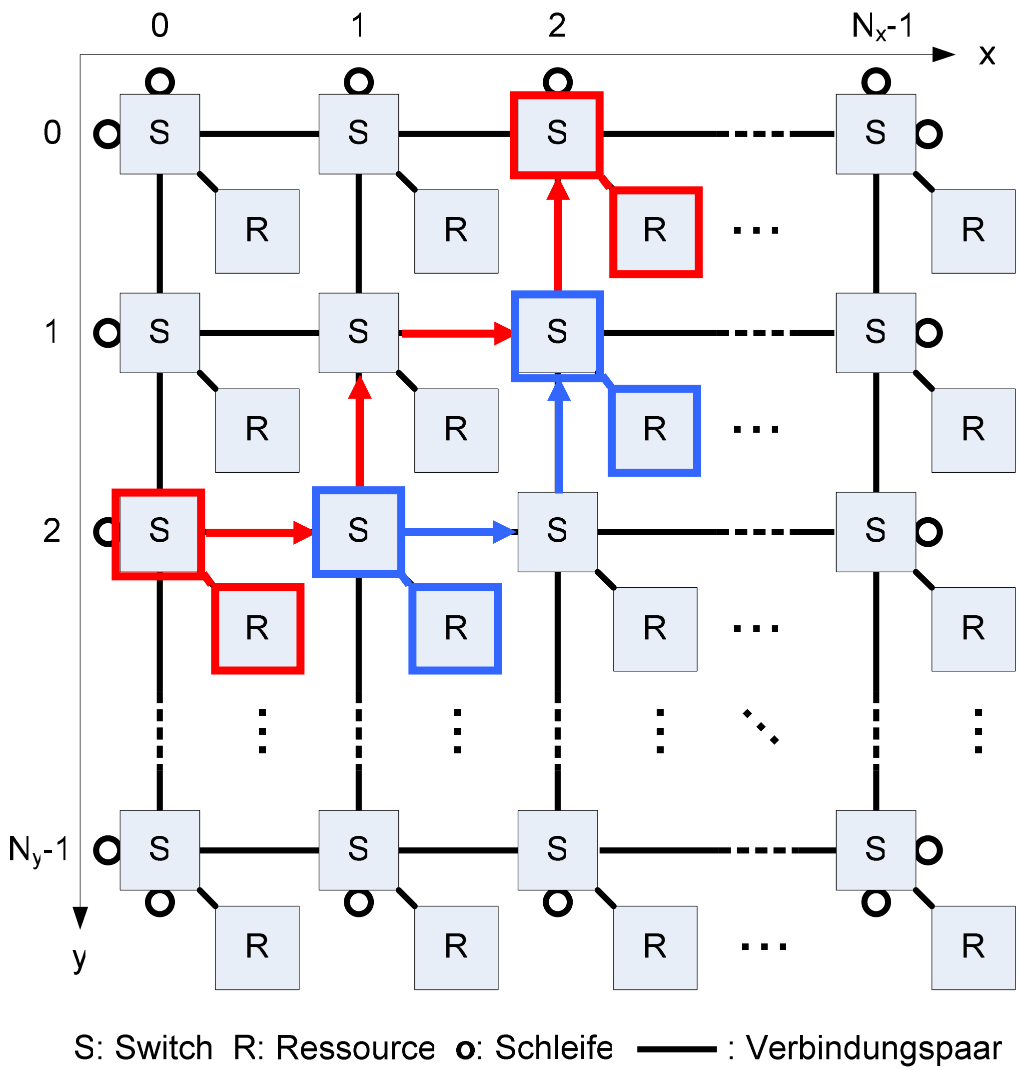
V	HC	Δx	Δy	Payload	CRC
1	11	10	10	88	8 bits

- CRC check on all inputs and outputs
  - Input: Detecting *Inter*-Switch-Faults
  - Output: Dectecting *Intra*-Switch-Faults

- Model of individual crossbar connection faults
- Implemented on chip to store online diagnosis results



- Fault Models and Fault Diagnosis
- Fault-Adaptive Routing
- Experimental (Simulation) Results

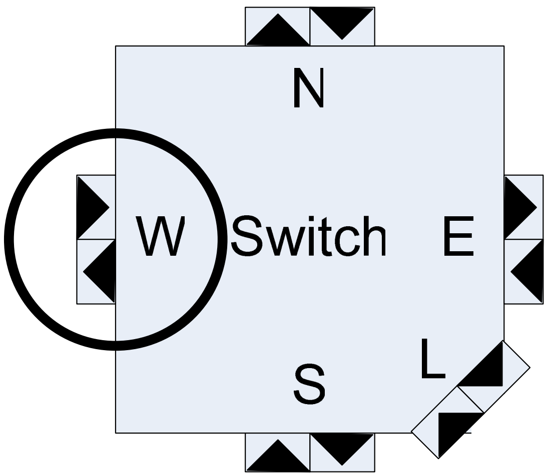


- Packets are routed with a preferred direction, e.g. following XY route
- If that is not possible, the packet is deflected into another direction

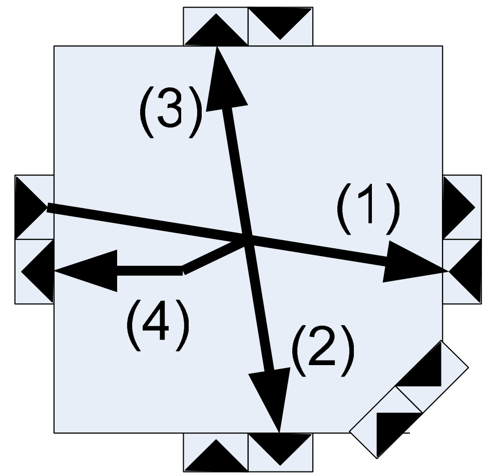
- No blocking
- No buffers
- Free of deadlocks
- Non-minimal paths possible
- Livelock avoidance through priority for „old" packets

Implementing Deflection in Switches

1.) Choose input based on priority



2.) Select output according to deflection policy



N, E, S, W: directions to neighbour switches
L: connection with the local resource

(1) Routing in *preferred* direction (here: x towards destination)

(2) Routing in other direction *towards* destination

(3) Routing in other direction *away* from destination

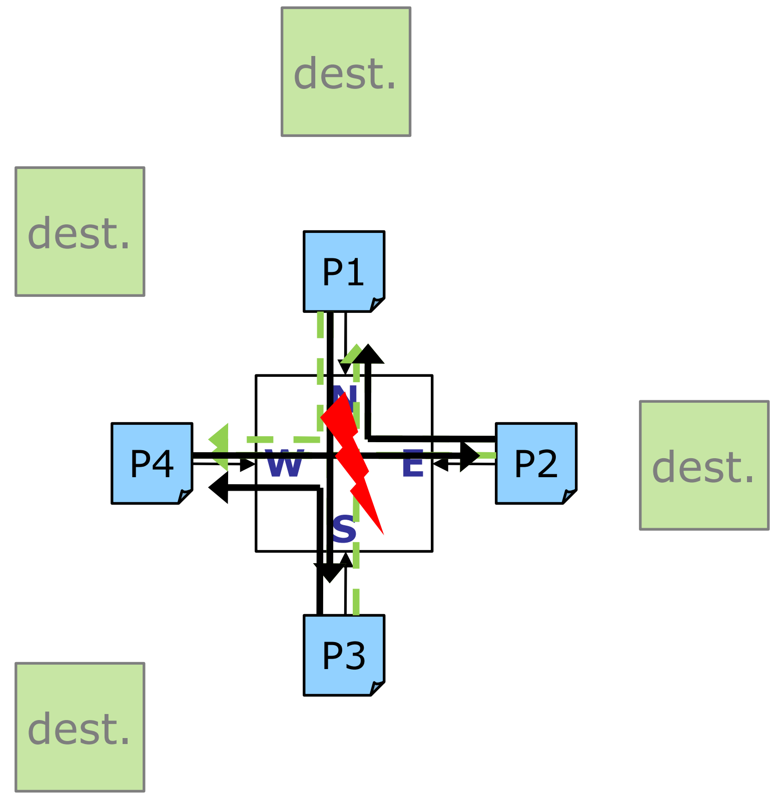
(4) Reflection to the packet source of the previous hop

Routing Adaptation

- In order to find feasible routes in case of faults, all incoming packets have to be considered as a whole:

packet	P1	P2	P3	P4
priority	max (4)	high (3)	low (2)	min (1)
from	N	E	S	W
preferred route	S, W	N, W	N	E
veer away	E	S	W,E	N,S
Reflection	N	E	S	W

	N	E	S	W
N			P1	
E	P2			
S				P3
W		~~P4~~		



Cost Driven Routing (1)

- Assign a *cost* to a routing decision for one packet:
  - Reflects the progress of the packet
  - Cost is weighted by priority

packet	P1	P2	P3	P4
priority	max (4)	high (3)	low (2)	min (1)
from	N	E	S	W
preferred route	S, W	N, W	N	E
veer away	E	S	W,E	N,S
reflection	N	E	S	W

	N	E	S	W
N	8	4	0	0
E	0	6	X	0
S	0	2	4	2
W	1	X	1	2

$$c_{ij} = \begin{cases} \infty & \text{if} \quad f_{ij} > 001_2 \vee \overline{avl_j}, \\ 2 & \text{elsif} \quad i = j, \\ 0 & \text{elsif} \quad i \to j \text{ reduces } |P_i.addr|, \\ 1 & \text{else} \end{cases}$$

⇐ connection unavailable

⇐ reflection

⇐ approach destination

⇐ veer away

Cost Driven Routing (2)

- In order to consider all packets at once, build the sum of weighted costs

$$c(\pi) = \sum_i r_{i\pi(i)}$$

	N	E	S	W
N	8	4	0	0
E	0	6	∞	0
S	0	2	4	2
W	1	∞	1	2

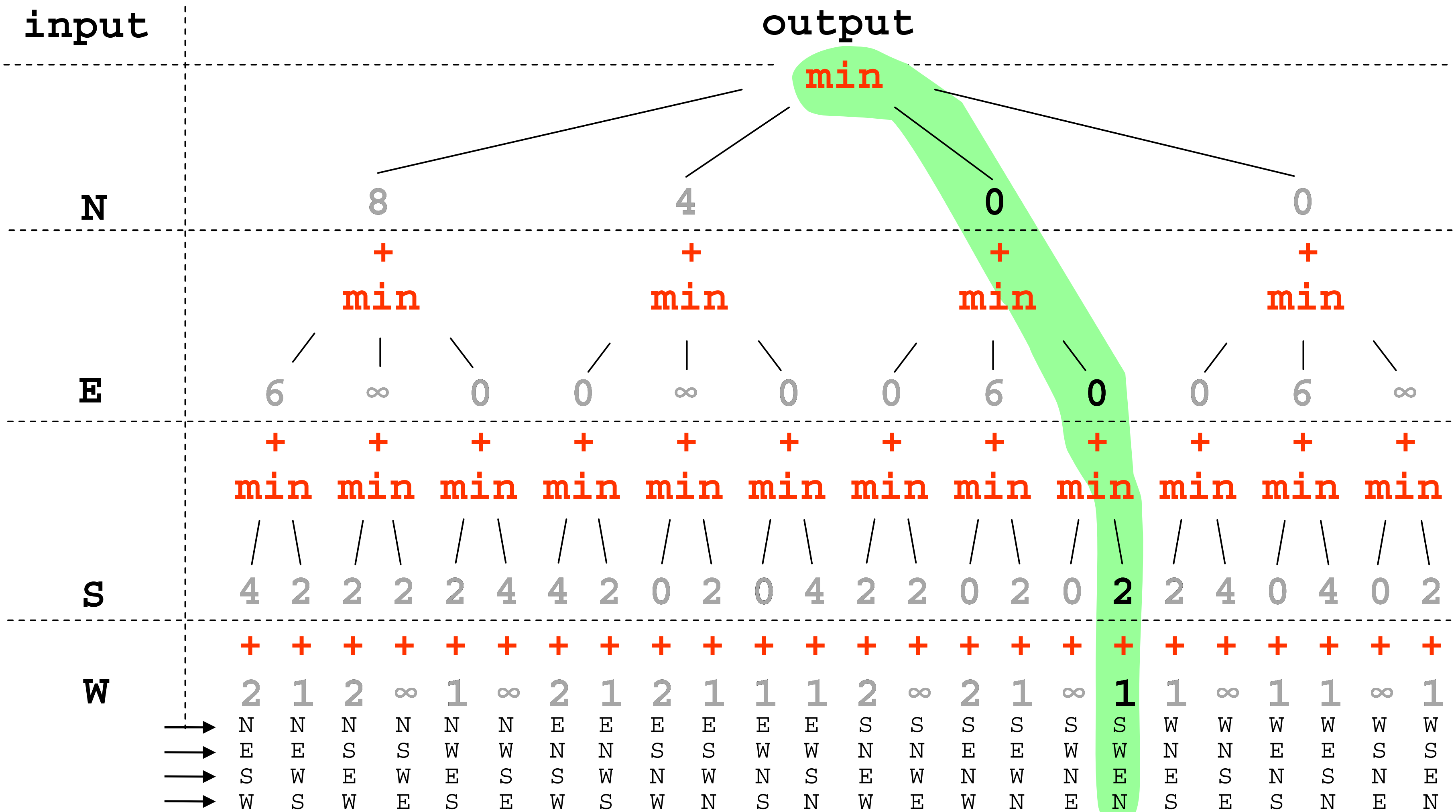
$$c(\pi_{\text{bad}}) = 9$$

$$c(\pi_{\text{good}}) = 4$$

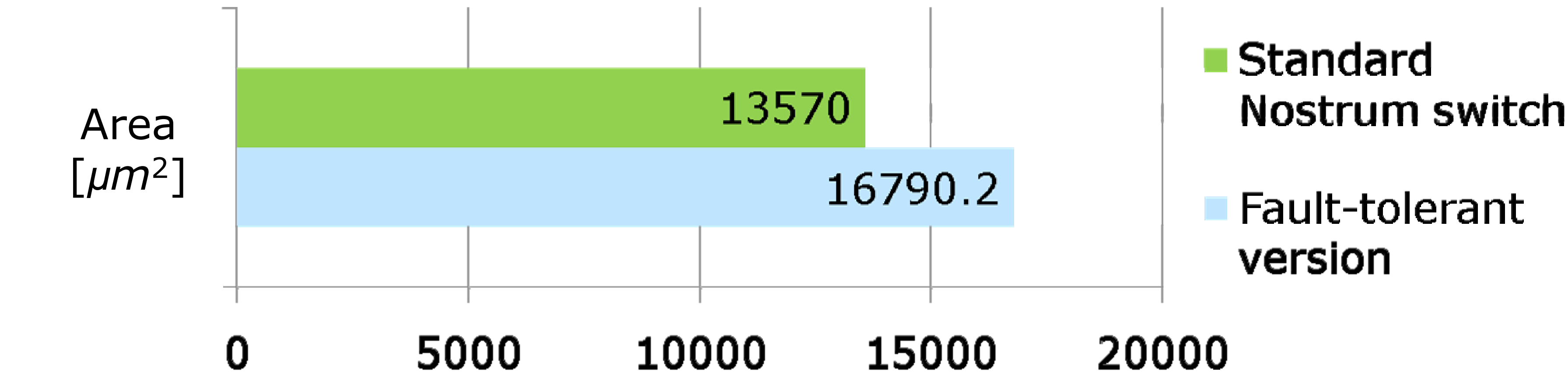
- Objective: Find routing permutation π_{opt} with minimal cost sum in a combinational way
  - Construct permutation tree
  - Bottom-up traversal of the permutation tree
  - Use minimum operations to discard “bad” routes early

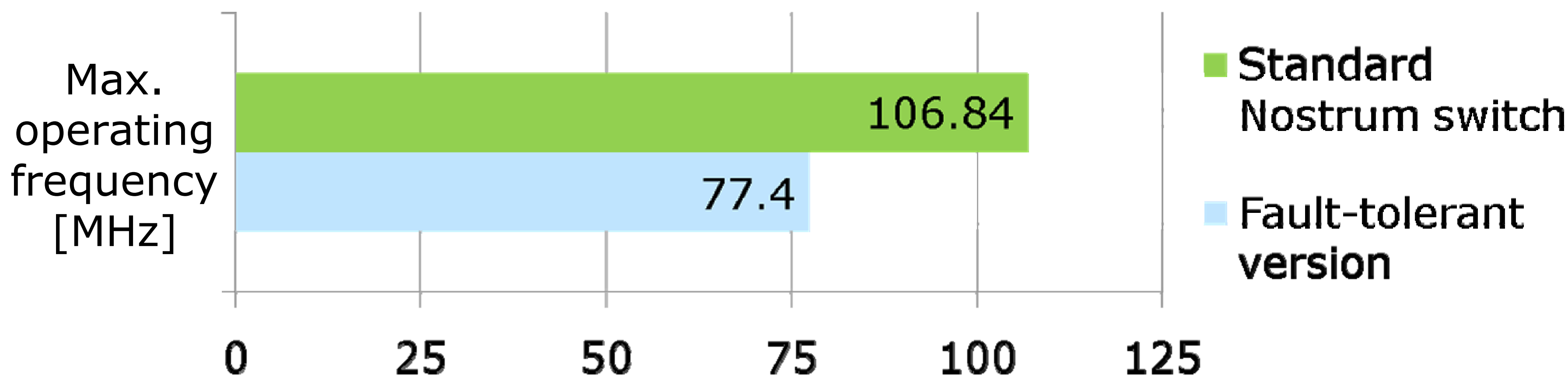
Routing Implementation

Example:



Switch Area and Timing





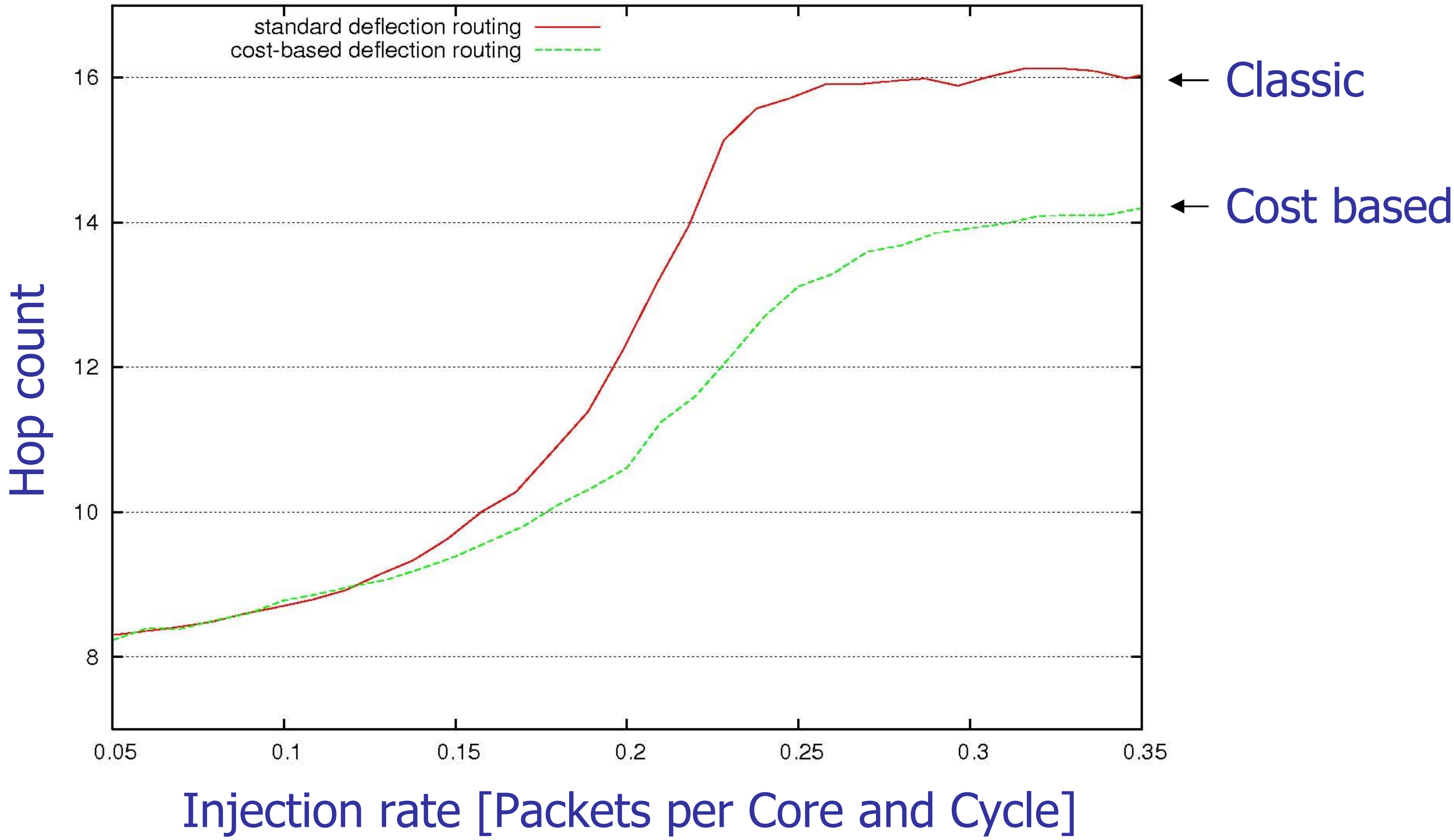
Outline

- Fault Models and Fault Diagnosis
- Fault-Adaptive Routing
- Experimental (Simulation) Results

Simulation Setup

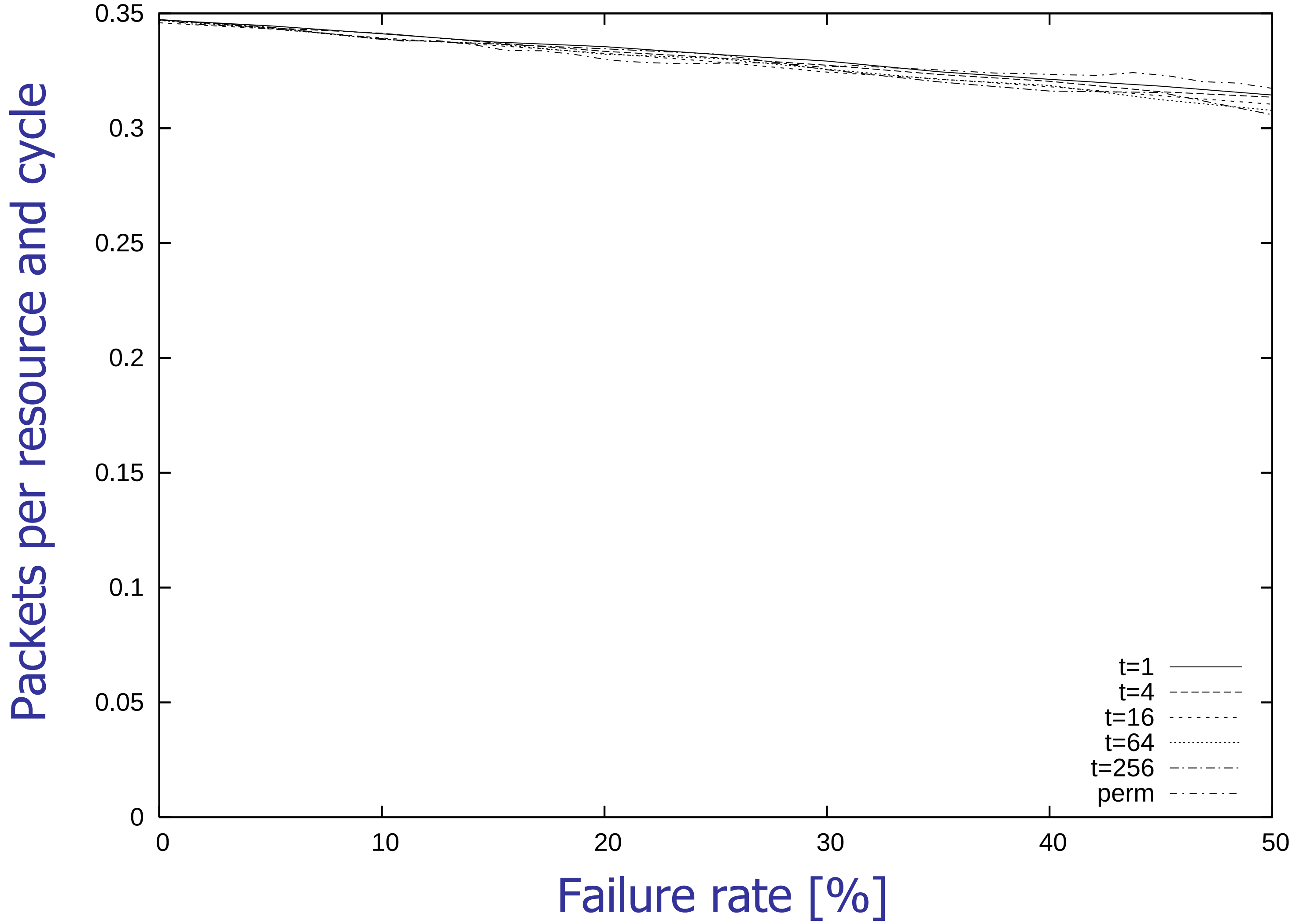
- 8x8 mesh with uniformly distributed and transposed data traffic
- Incremental injection of faults according to a fixed, randomly selected fault pattern
- Measurement of latency and throughput
- Comparison of three models:
  - Diagnosis and adaptation as presented
  - Automated Repeat Request (ARQ) at receiving end
  - Automated Repeat Request (ARQ) on switch level

Fault-Free Case



- Average network latency reduced by ~12%

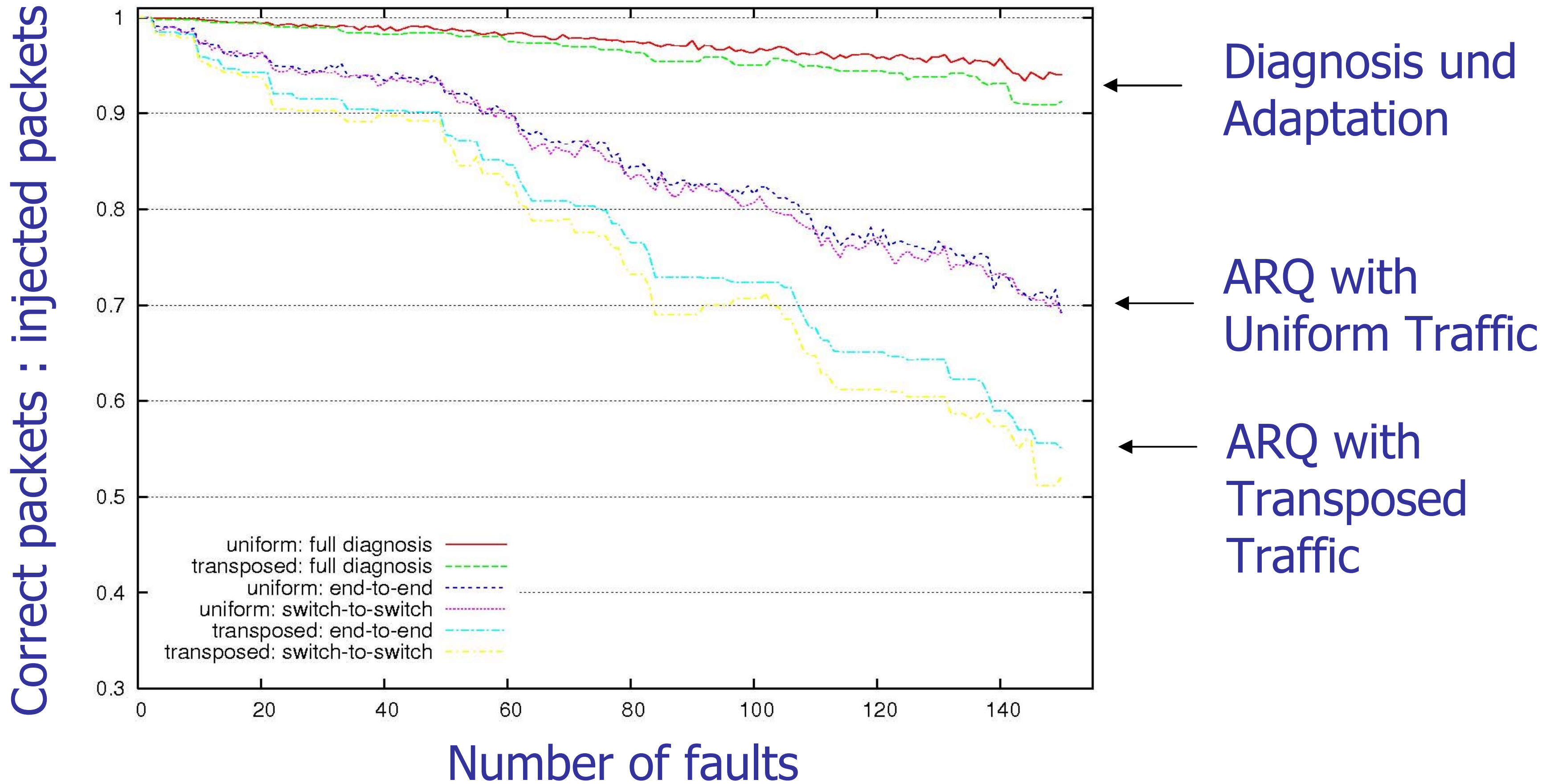
Packet Throughput with Single Crossbar Faults



8x8 mesh

- Minor throughput reduction even at large failure rates
- Fault duration has no significant impact

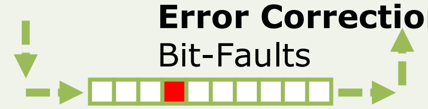
Alternative Methods

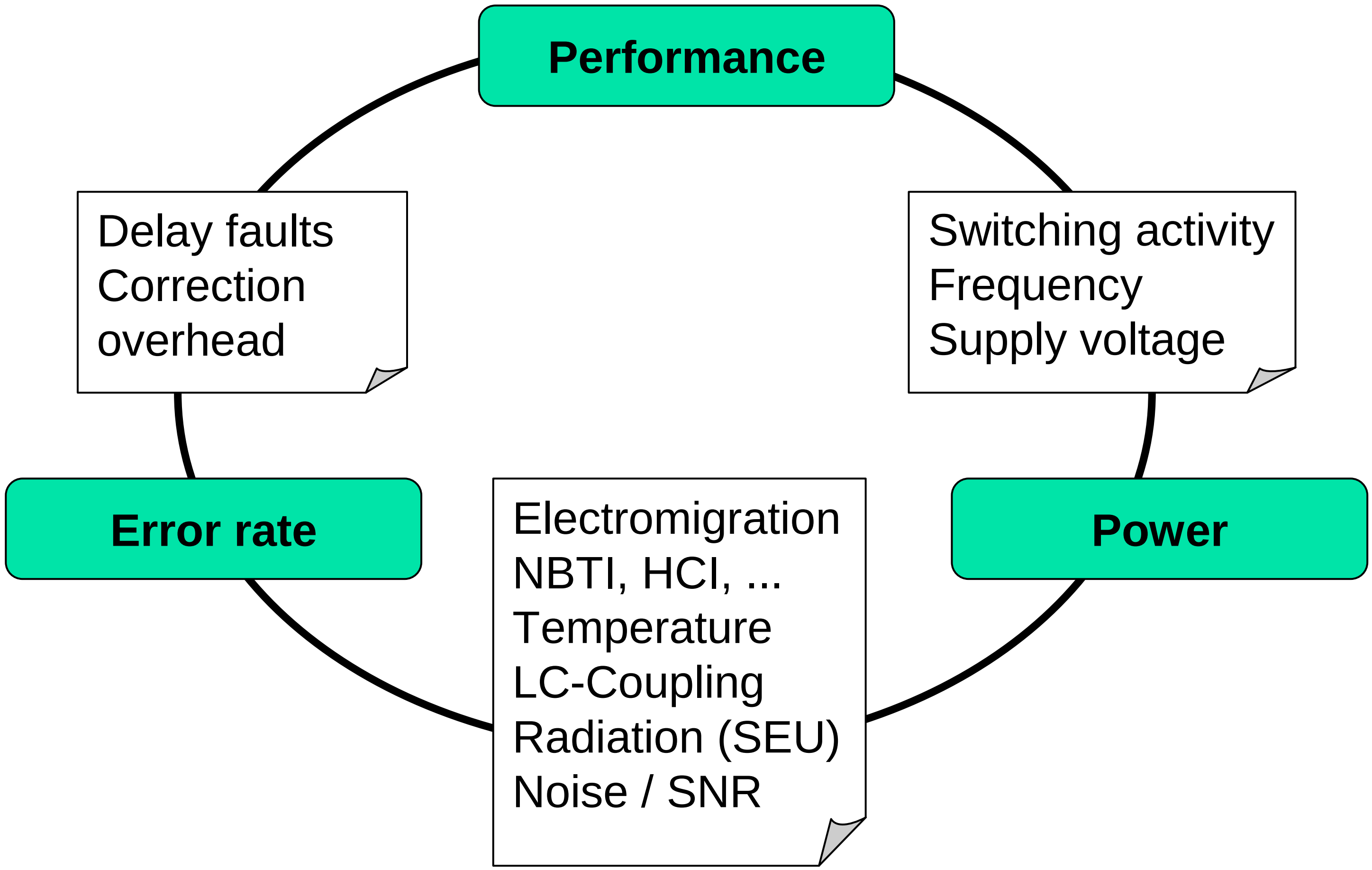


- Routing around faults is better than accepting faults and performing retransmission

- Routing method using information on NoC permanent fault status
- Distributed implementation (scalable)
- Can be implemented with acceptable cost
- Current research: combined with retransmission to handle transient faults
- Future work: research optimal combination of techniques on different layers

Hierarchical Approach to Fault Tolerance

	Fault Models	Monitoring	Diagnosis	Reconfiguration
Transport				
	↑ Packets	↑ Congestion Packet Loss Packet-Fault	↓ Packet Loss Error	↑ Re-Routing
Network **Distributed Online-Algorithms, Regular Structures**	**Error Detection**			**Error Correction**
	↑ Word-stream	↑ Bandwidth Fault Rate	↓ Fault	↑ Bandwidth-Request
Data Link **Encoding (Logic)**	**Error Detection** Bit-Faults			**Error Correction** Bit-Faults
	↑ Bitstream	↑ Fault Rate Electr. Param. Temperature	↓ Defect	↑ Hardening-Suggestions
Physical **Encoding (electrical), Drivers, Monitoring**	**Error Detection** Delay Faults Voltage-, Current-, Temperature-Monitoring			**Error Correction** Adjust Voltage, Current Use multiple lines Spare-Lines


Performance
Delay faults
Correction
overhead
Switching activity
Frequency
Supply voltage
Error rate
Electromigration
NBTI, HCI, ...
Temperature
LC-Coupling
Radiation (SEU)
Noise / SNR
Power