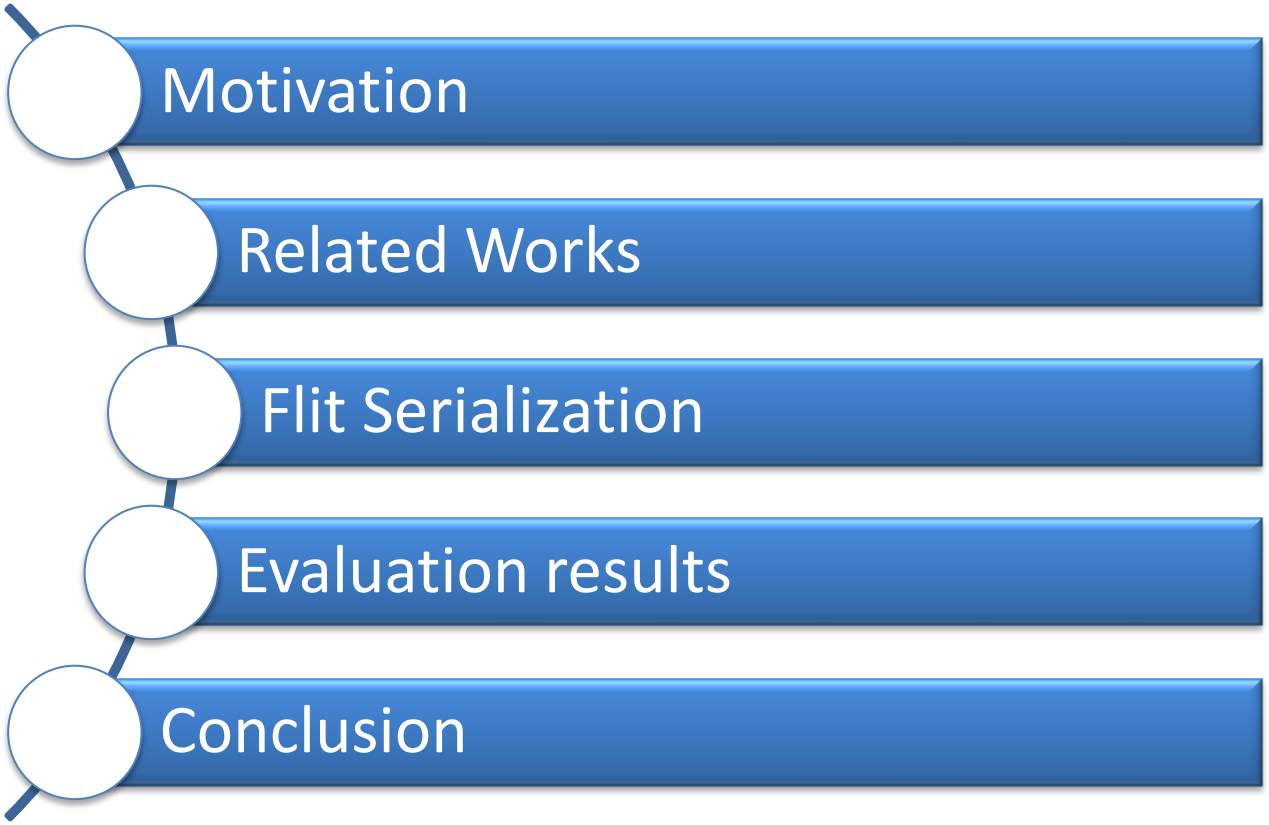
A Novel Flit Serialization Strategy to Utilize Partially Faulty Links in Networks-on-Chip

Changlin Chen*, Ye Lu†, Sorin D. Cotofana*

*Computer engineering, TU Delft
{c.chen-2, S.D.Cotofana}@tudelft.nl

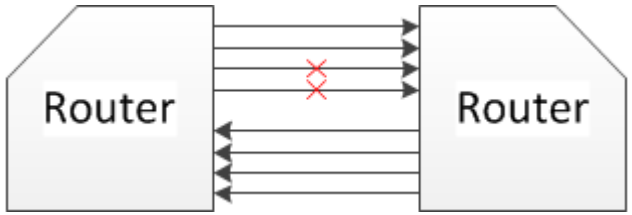
†ECIT, QUB
ylu10@qub.ac.uk

Outline


Motivation
Related Works
Flit Serialization
Evaluation results
Conclusion

Motivation

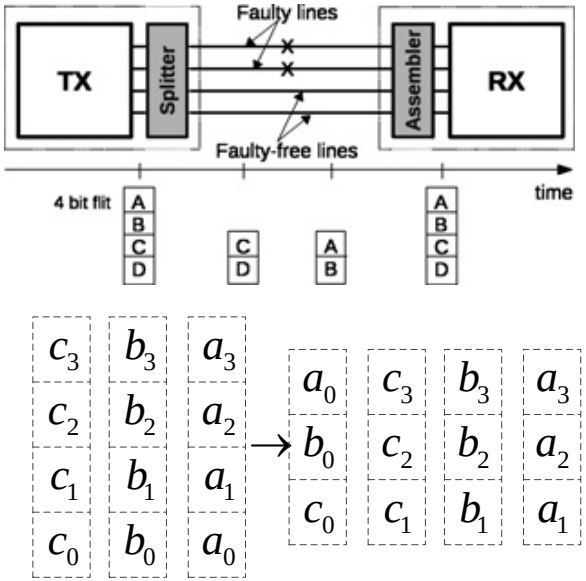
- NoC: Routers + Links + Network Interfaces
- Links are prone to
  - Manufacturing defects
  - Chip wear out effects
  - Process Parameter Variations



- Faulty links can be isolated by fault tolerant routing algorithm
- The remaining bandwidth of the partially faulty link is wasted
- Partially faulty links should be utilized

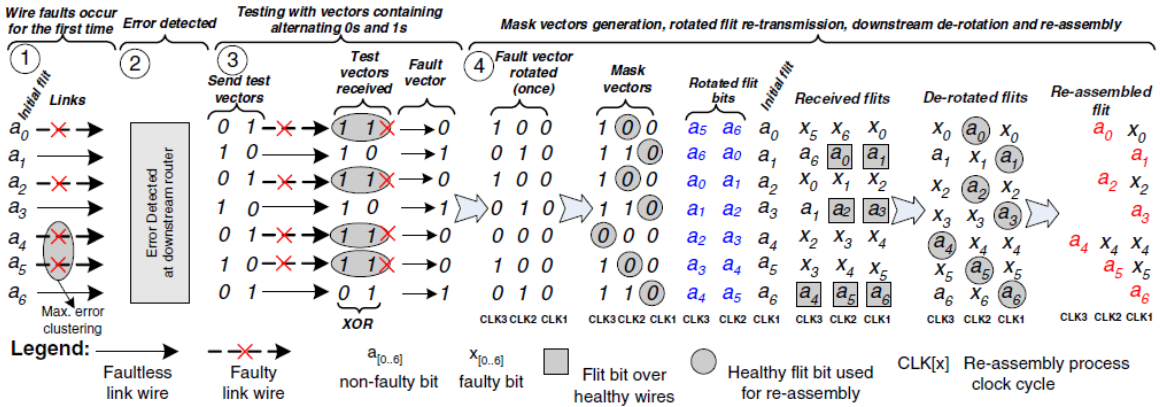
Related works

- Use pre-fabricated spare wires to replace faulty wires
  - Grecu et al.
  - Lehtonen et al.
- Simple flit quad splitting (SFQS)
  - *Palesi et al.
  - Lehtonen et al.
- Packet rebuilding/restoring
  - Yu et al.
- Partially faulty link recovery mechanism (PFLRM)
  - †Vitkovskiy et al.



*

$$\begin{bmatrix} c_3 & b_3 & a_3 \\ c_2 & b_2 & a_2 \\ c_1 & b_1 & a_1 \\ c_0 & b_0 & a_0 \end{bmatrix} \rightarrow \begin{bmatrix} a_0 & c_3 & b_3 & a_3 \\ b_0 & c_2 & b_2 & a_2 \\ c_0 & c_1 & b_1 & a_1 \end{bmatrix}$$

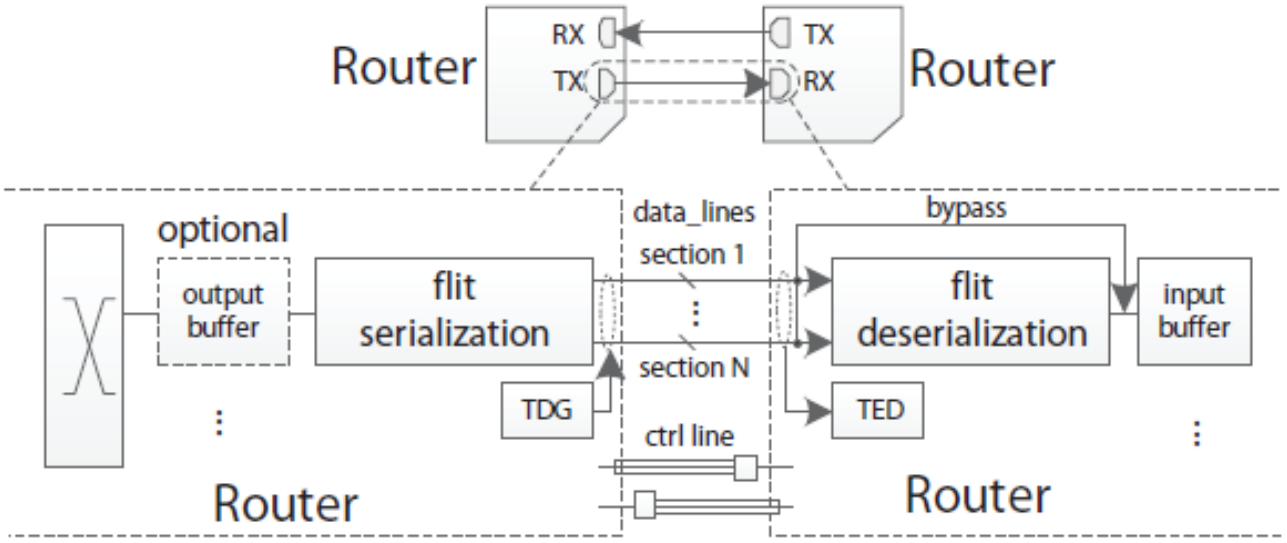


†

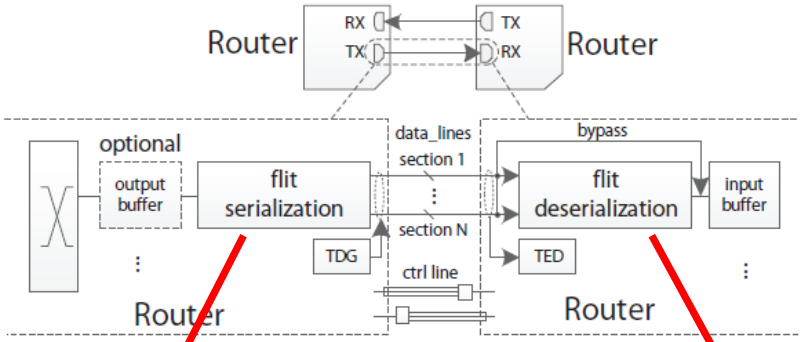
The remaining link bandwidth should be used more efficiently

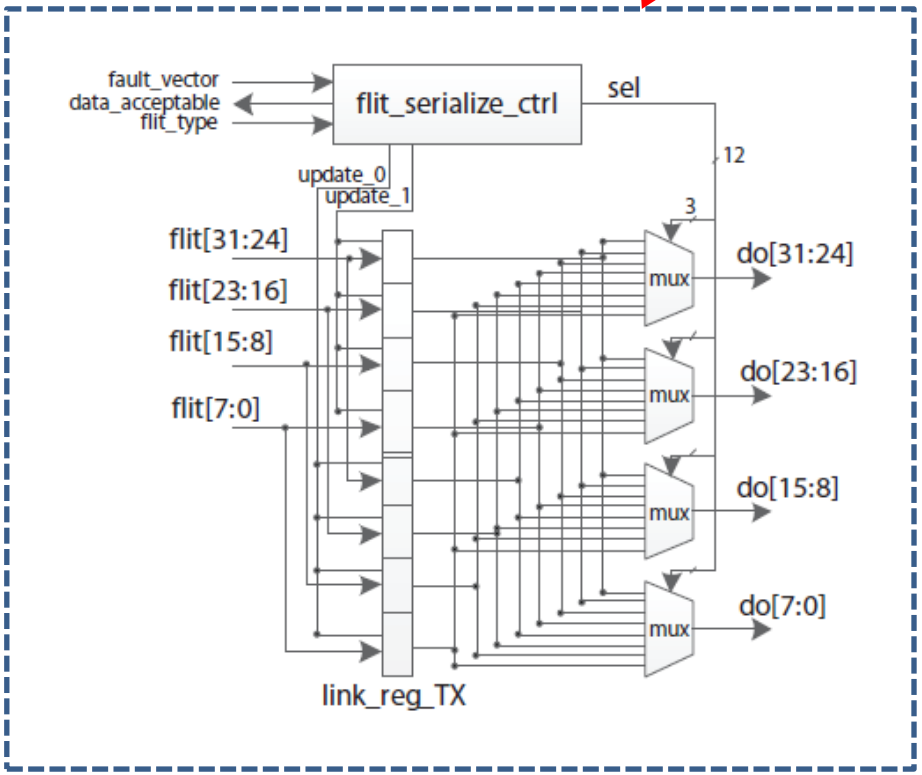
Flit Serialization

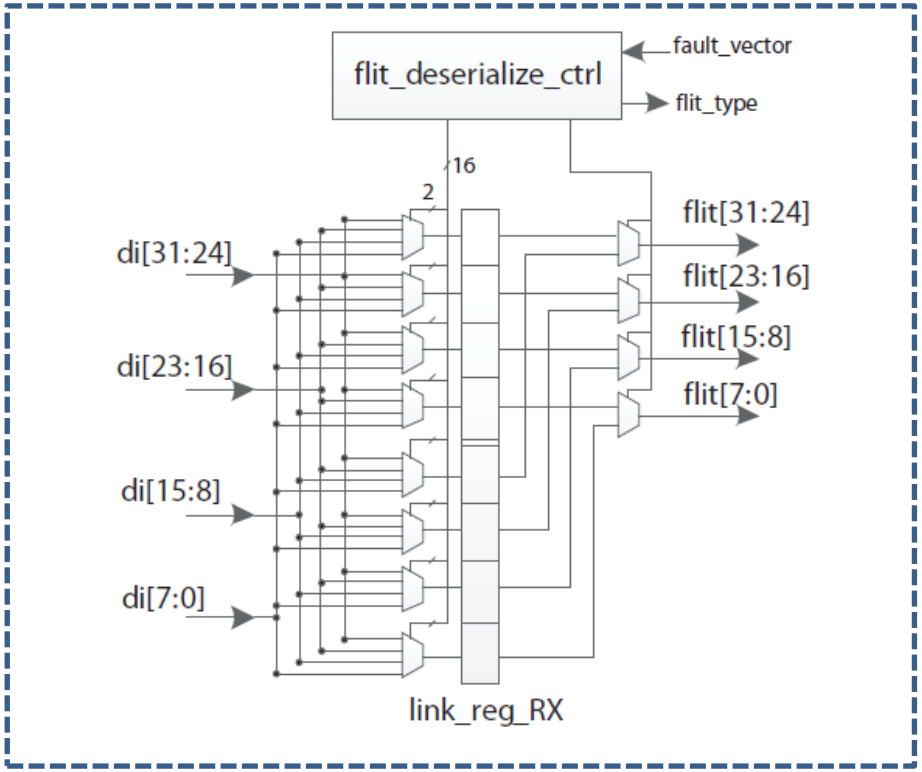
- Proposed link fault tolerant architecture



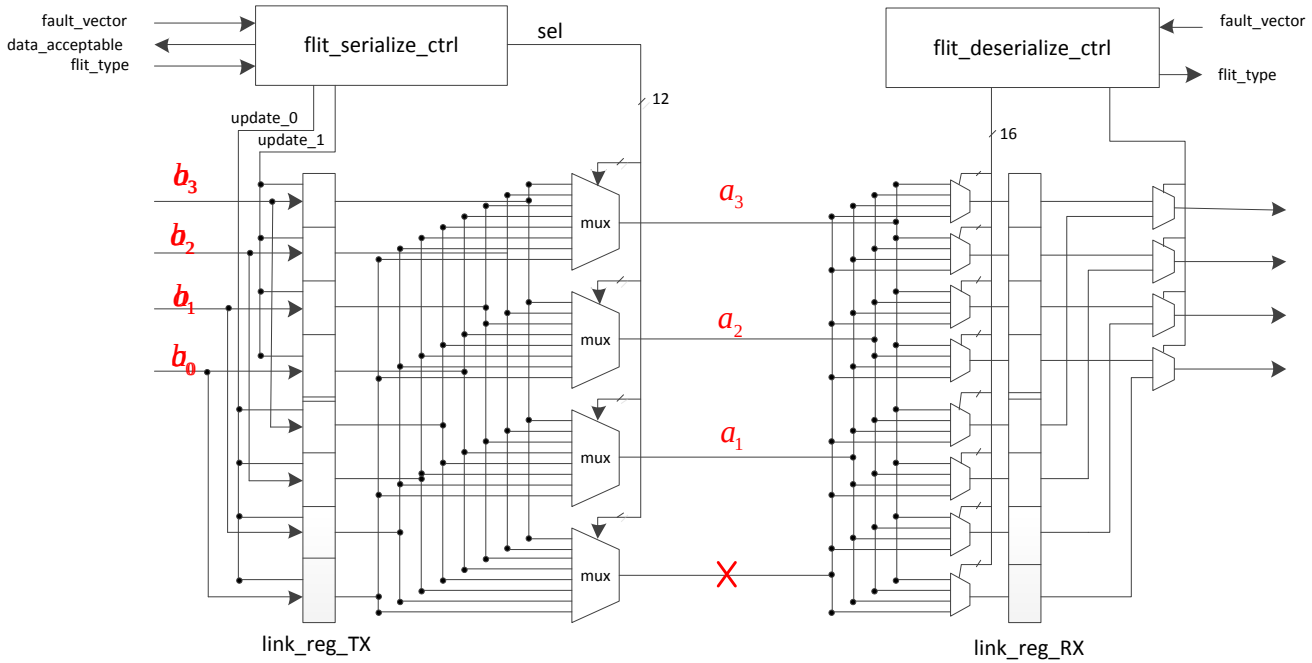
- Links are diagnosed periodically and the fault vector of a link is sent to the control logics at TX and RX side

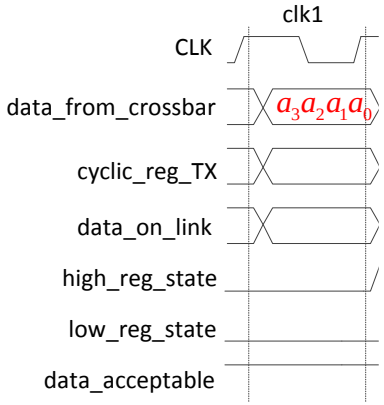

Router
RX
TX
TX
RX
Router
optional
output buffer
flit serialization
data_lines
section 1
section N
TDG
ctrl line
bypass
flit deserialization
input buffer
TED
Router
Router

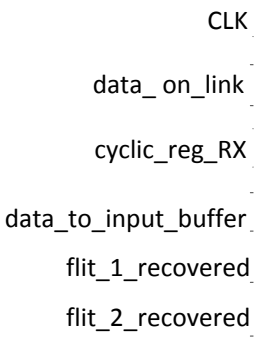

fault_vector
data_acceptable
flit_type
flit_serialize_ctrl
sel
12
update_0
update_1
3
flit[31:24]
flit[23:16]
flit[15:8]
flit[7:0]
mux
mux
mux
mux
do[31:24]
do[23:16]
do[15:8]
do[7:0]
link_reg_TX


flit_deserialize_ctrl
fault_vector
flit_type
16
2
di[31:24]
di[23:16]
di[15:8]
di[7:0]
flit[31:24]
flit[23:16]
flit[15:8]
flit[7:0]
link_reg_RX

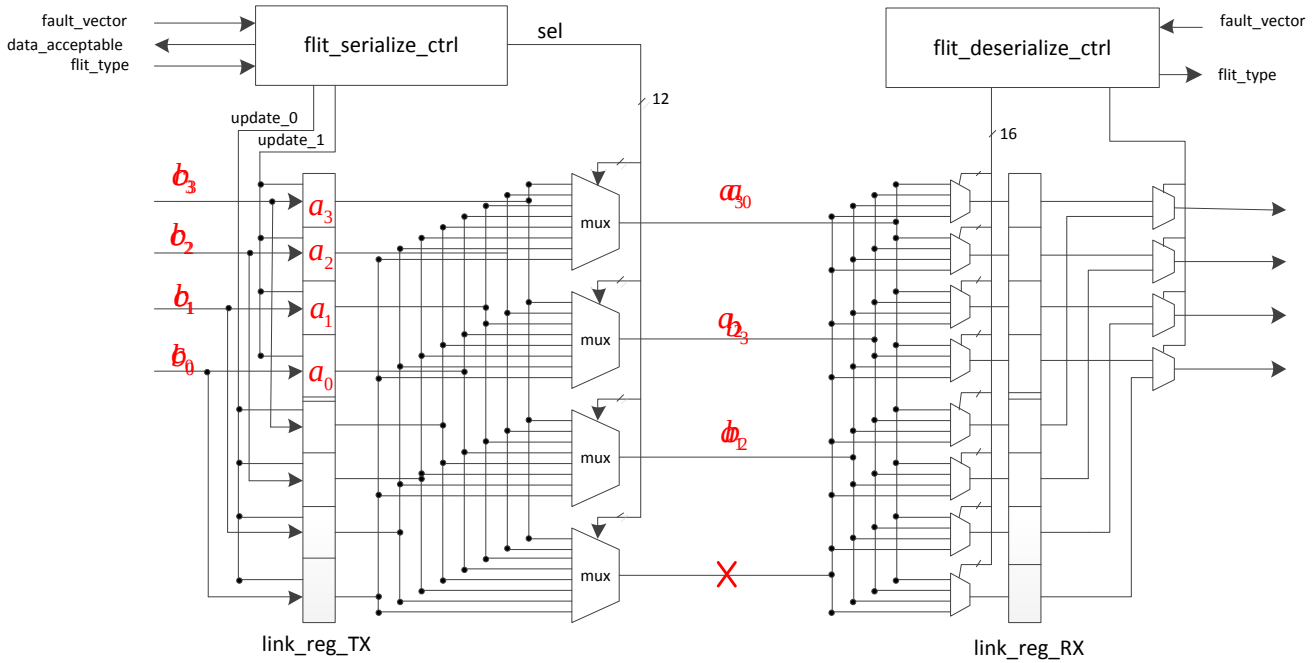
Flit Transmission Process:

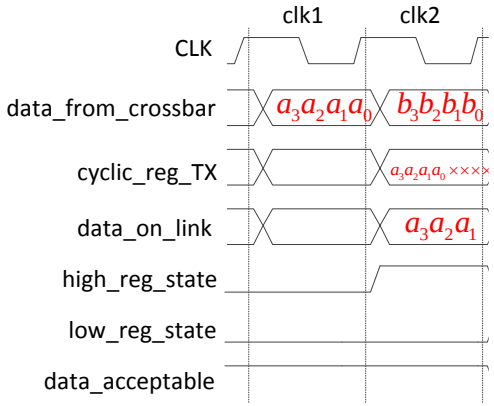


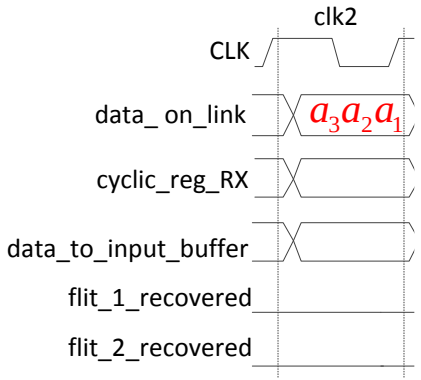




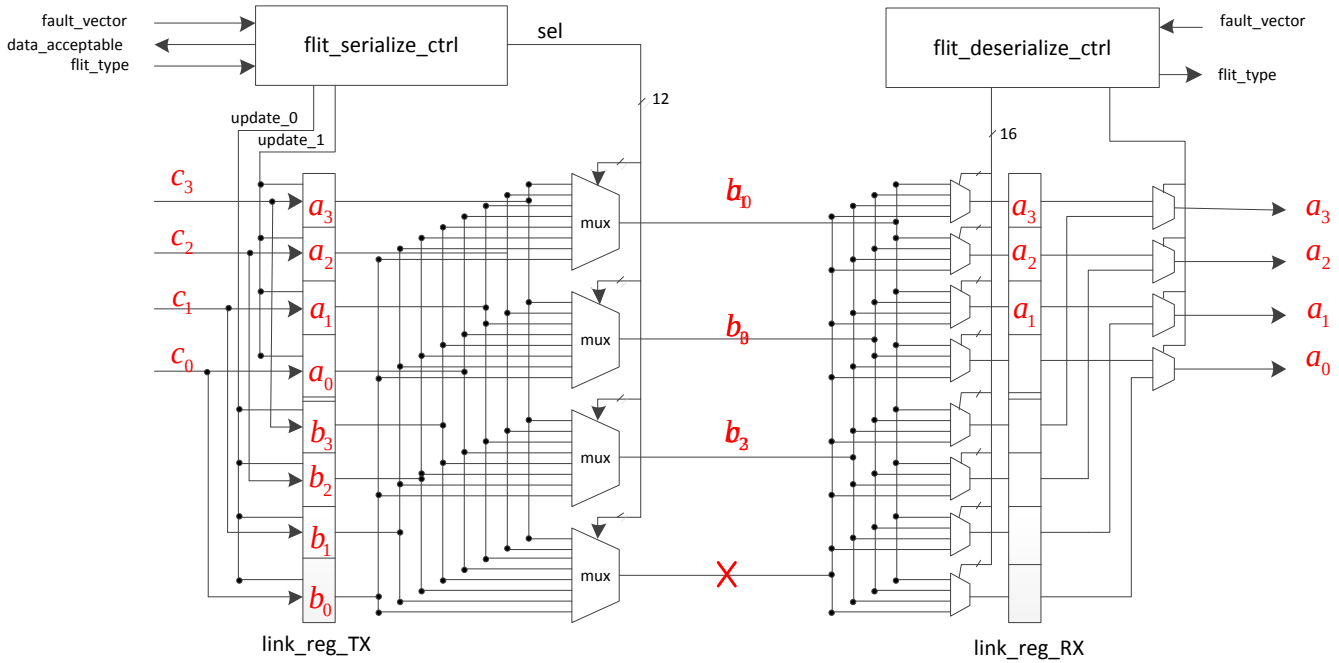
Flit Transmission Process:

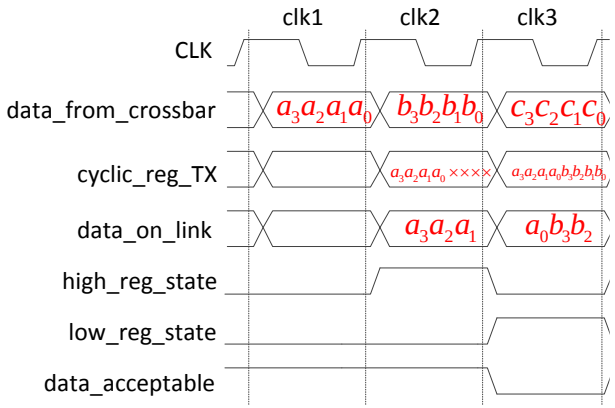


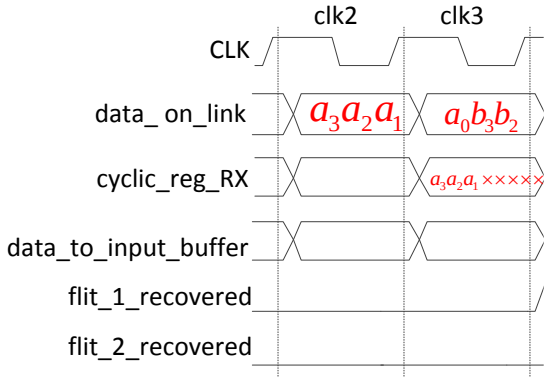




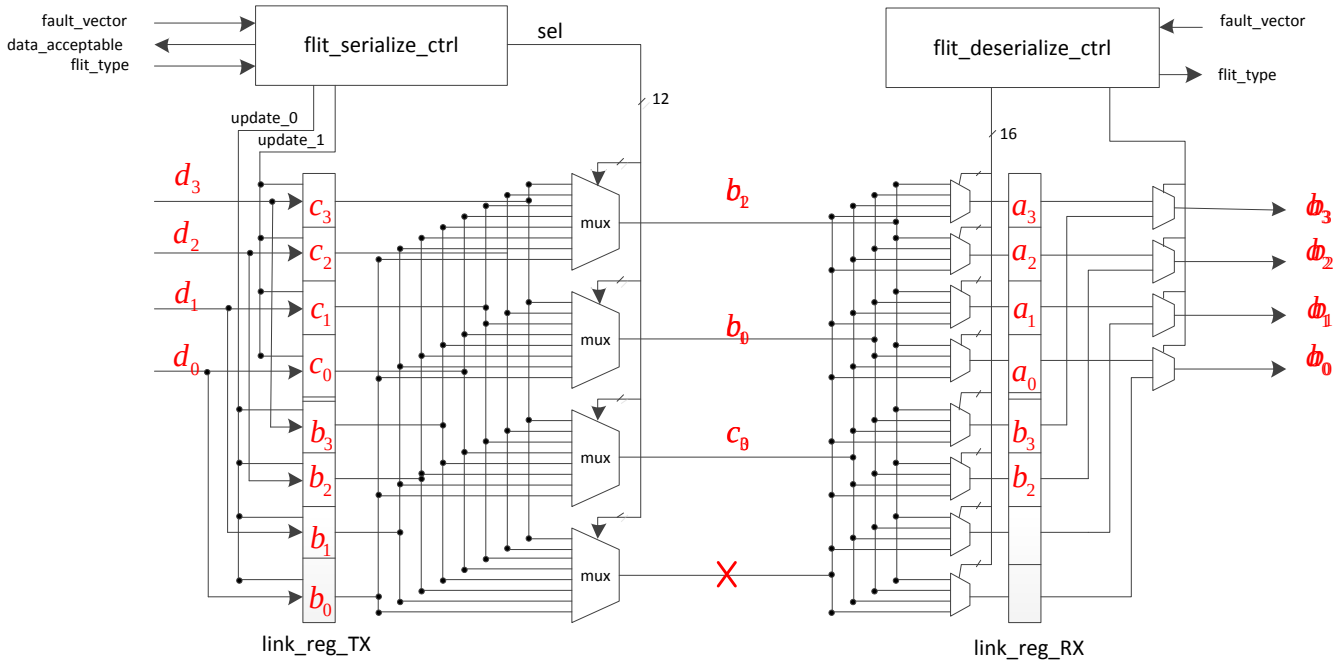
Flit Transmission Process:

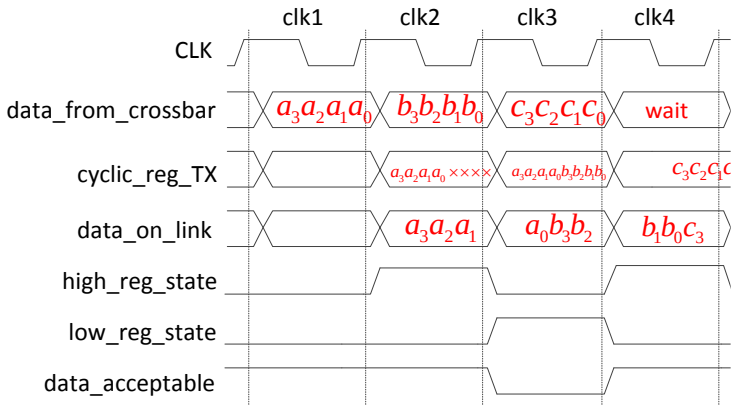


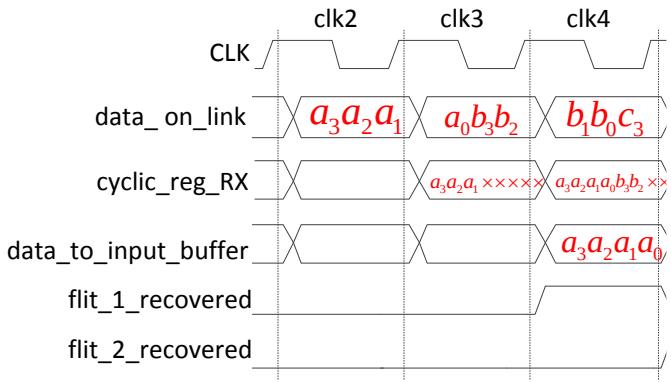




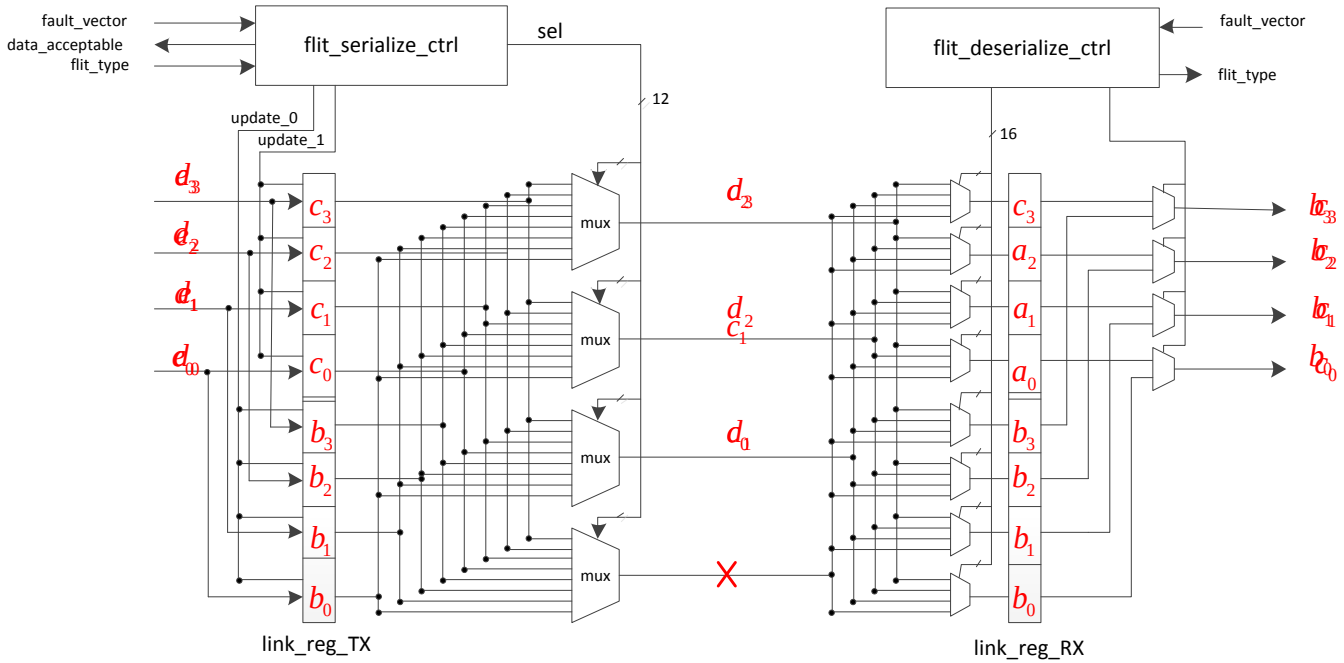
Flit Transmission Process:

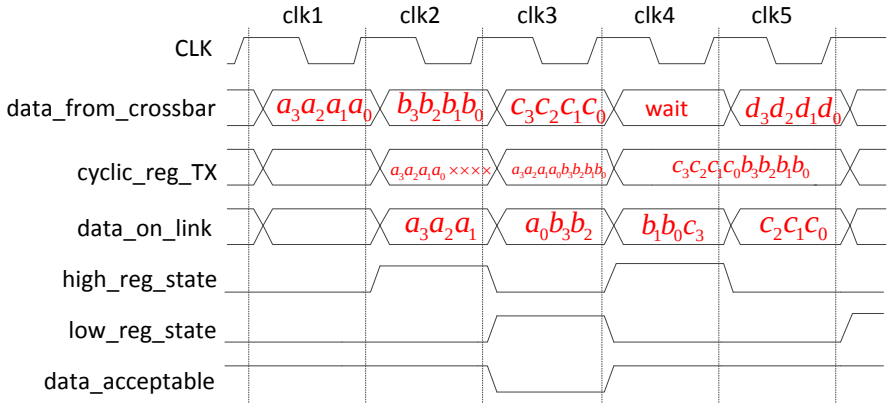


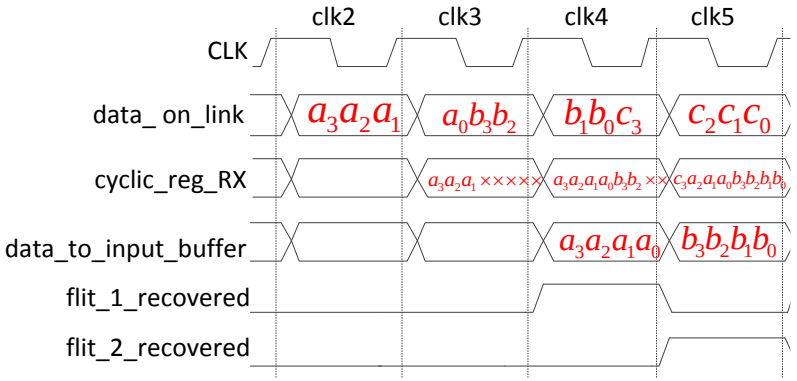




Flit Transmission Process:







Flit Serialization

Link Latency

$$Latency_{proposed} = \left\lceil \frac{section\_number \times flit\_number}{fault\_free\_section\_number} \right\rceil$$

$$Latency_{PFLRM} = (cluster\_size + 1) \times flit\_number$$

$$Latency_{SFQS} = \frac{section\_number \times flit\_number}{available\_section\_number}$$

Wire fault probability

Even distribution:

$$P_{N_e=k} = \binom{n}{k} p_e^{\,k} (1-p_e)^{n-k} \approx \binom{n}{k} p_e^{\,k}$$

Cluster Faults:

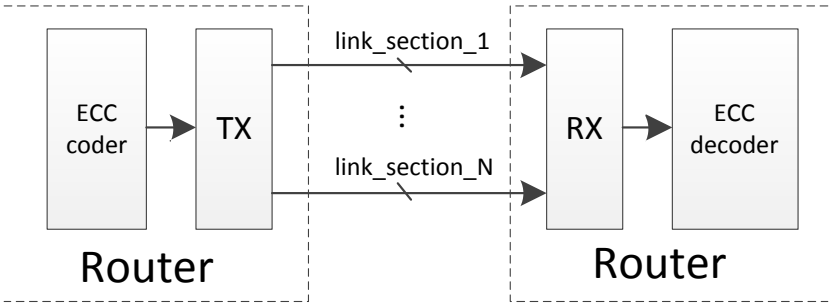
Faults may stay in one link section

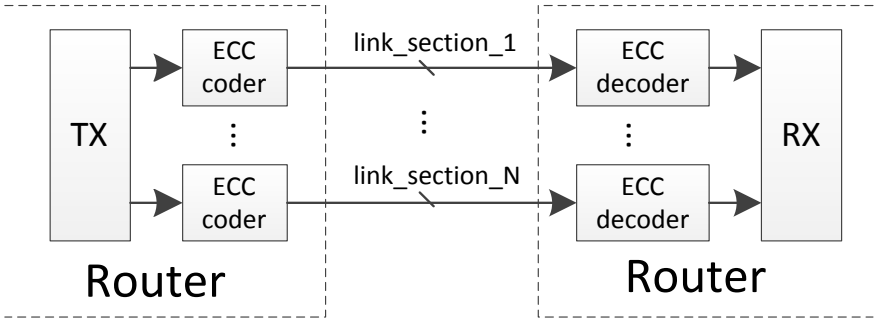
Table 1 Link latency overheads when flits are transmitted continuously

Fault number	Proposed 4 sections	Proposed 8 sections	PFLRM Best situation	PFLRM Worst situation	SFQS
0	0%	0%	0%	0%	0%
1	33.3%	14.3%	100%	100%	100%
2	100%	33.3%	100%	200%	100%
3	300%	60.0%	100%	300%	300%
4	--	100%	100%	400%	--
5	--	167%	100%	500%	--
6	--	300%	100%	600%	--
7	--	700%	100%	700%	--

ECC Integration

- Two ways to collaborate with ECC





- The realization of other parts can be conventional

Evaluation Results

- Platform
  - NoC Topology: 8 X 8 2D mesh
  - Router:
    - 3 pipeline stages
      - Look ahead routing & VC/Switch allocation
      - Switch traversal
      - Link Traversal
    - 5 physical channels with 5 virtual channels in each
    - Each VC is 4-flit deep and 32-bit wide
  - Realized at RTL level by using Verilog HDL
  - Synopsys Design Compiler, TSMC 65nm, 500MHz

Evaluation Results

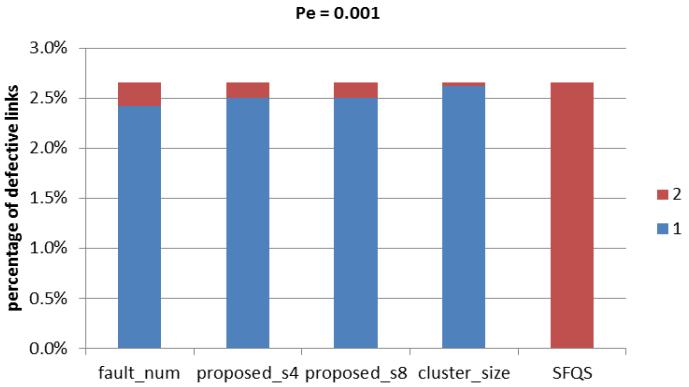
- Area and Power overhead

Table 2 Power and area overhead of different link fault tolerant methods

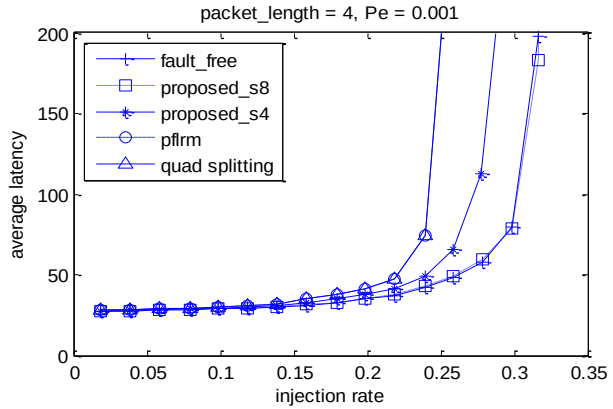
Link fault tolerant method		Dynamic Power (mW)	Leakage Power (mW)	Area(μm^2)
Basic router		18.20 / 0%	0.5269 / 0%	69560 / 0%
Proposed	4 sections	20.81 / 14.3%	0.7994 / 51.7%	89567 / 28.8%
	8 sections	21.18 / 16.4%	0.9294 / 76.4%	96538 / 38.8%
Spare wires	4	26.71 / 46.8%	0.9959 / 89%	102383 / 47.2%
	8	29.03 / 59.5%	1.0442 / 98.2%	116789 / 67.9%
PFLRM		19.30 / 6.0%	0.5789 / 9.9%	83326 / 19.8%
SFQS		20.27 / 11.4%	0.6807 / 29.1%	81288 / 16.9%

Evaluation Results

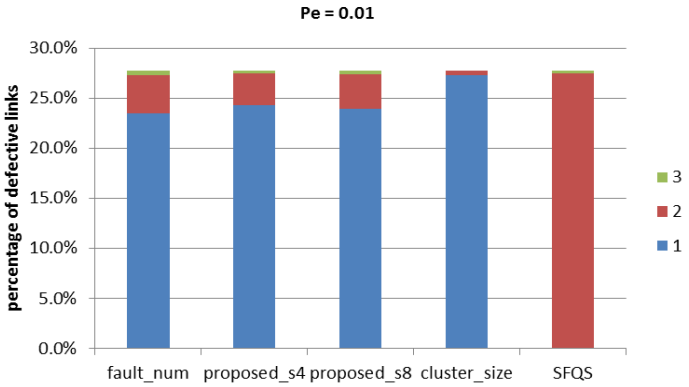
- System performance (uniform traffic)



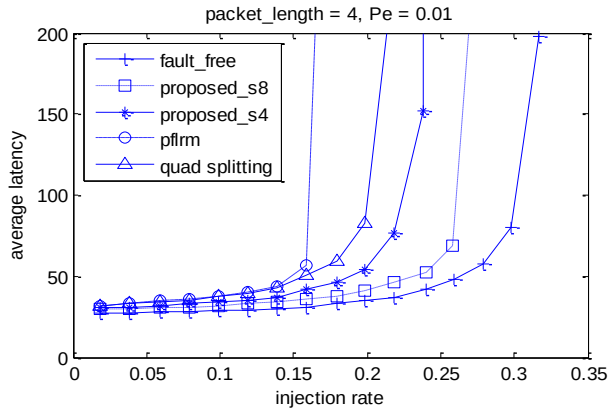
(a) fault pattern when p_e=0.001



(b) performance when p_e=0.001



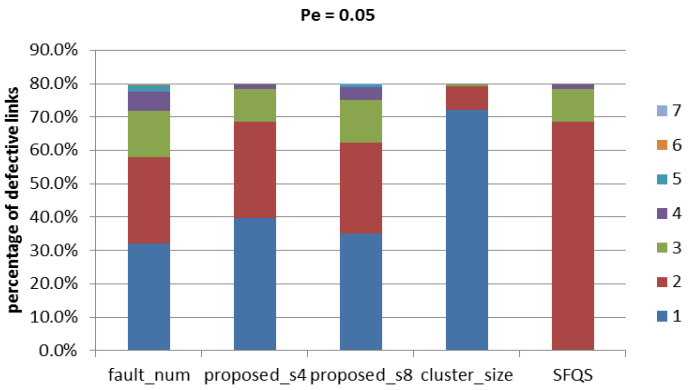
(c) fault pattern when $p_e = 0.01$



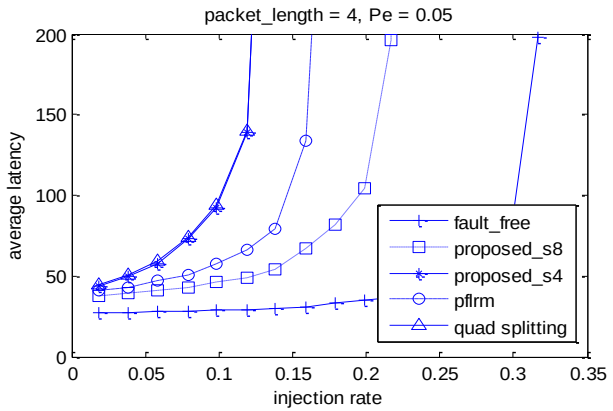
(d) performance when $p_e = 0.01$

Evaluation Results

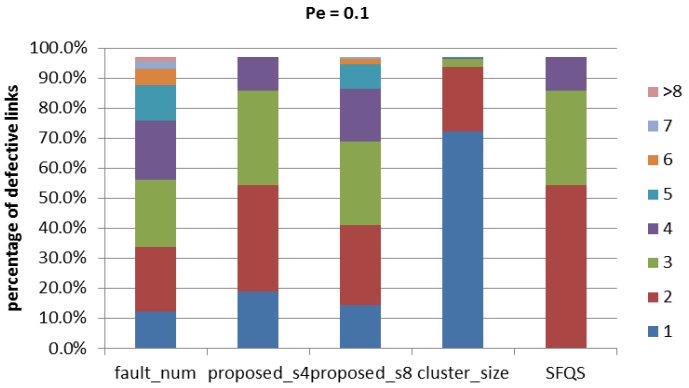
- System performance (uniform traffic)



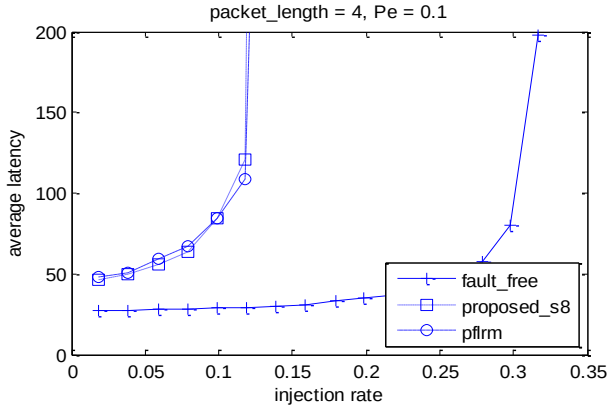
(a) fault pattern when $p_e = 0.05$



(b) performance when $p_e = 0.05$



(c) fault pattern when $p_e = 0.1$



(d) performance when $p_e = 0.1$

Conclusion

- A novel flit serialization strategy to utilize partially faulty links
- Flits and links are divided into several sections
- On a defective link, flit sections are serialized and transmitted on all faulty free link sections
- The flit serialization and deserialization modules are transparent to the other part of the router
- Link latency overhead is significantly reduced when compared with other partially faulty link usage strategies
- Area and energy overheads are reduced when compared with spare wire replacement method with slight performance degradation
- Performance of the NoCs can be degraded gracefully

THANKS