Universität Rostock

Traditio et Innovatio

# I. RedNoCs: A Runtime Configurable Solution for Cluster-based and Multi-objective System Management in Networks-on-Chip

# II. Centralized Adaptive Source-Routing for Networks-on-Chip as HW/SW-Solution with Cluster-based Workload Isolation

Philipp Gorski[1], Claas Cornelius[1], Dirk Timmermann[1], Volker Kühn[2]
Institute of Applied Microelectronics and Computer Engineering[1]
Institute of Communication Engineering[2]
University of Rostock, Germany
{philipp.gorski2, claas.cornelius, dirk.timmermann, volker.kuehn}@uni-rostock.de

# Overview/Structure

- **Introduction**
  - Challenges for Many-Core Platforms
  - Basics of Networks-on-Chip (NoC)
- **RedNoCs**
  - Basic Concept
  - Hardware/Software-based Clustering
  - Traffic Monitoring
  - Temperature Monitoring
  - Experimental Evaluation and Results
- **Centralized Adaptive Source-Routing**
  - Basic Concept
  - Adaptive Centralized XY/YX-Routing
  - Evaluation Model
  - Experimental Evaluation and Results
- **Outlook & Future Work**
- **References**

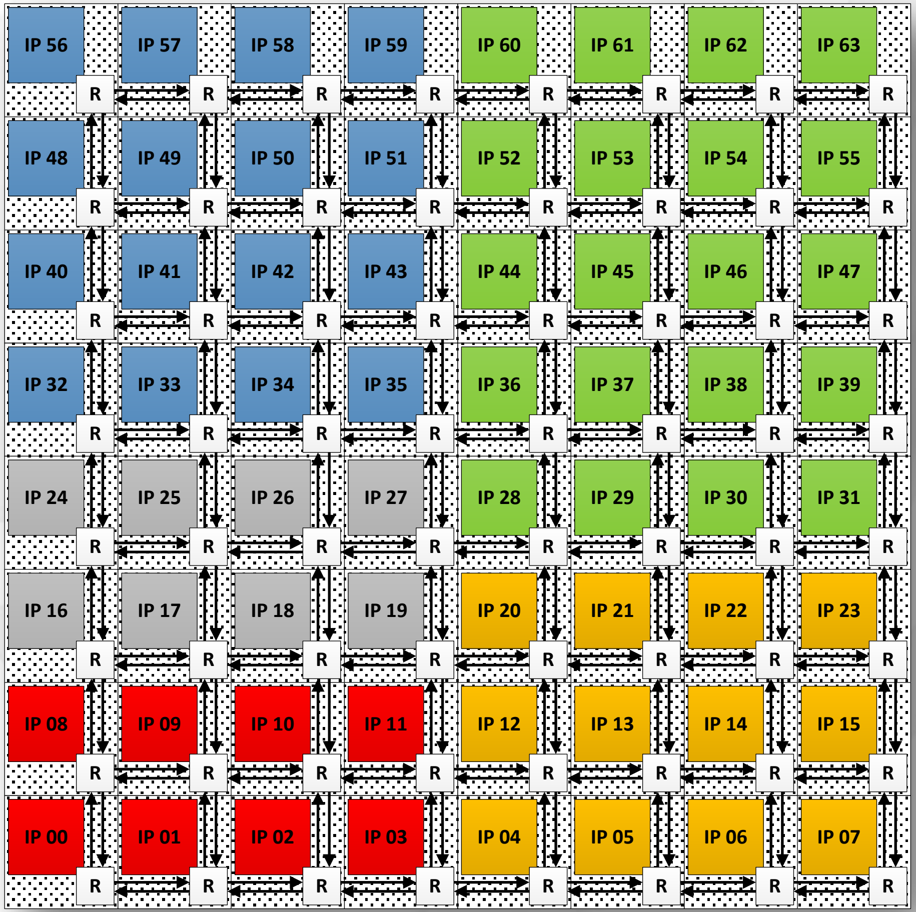
# Introduction

- **Challenges for Many-Core-Platforms?**
  - Increasing number of on-chip Resources, but with specific integrated Application Domains
    - General Purpose Processing, Graphic Processing, Multimedia (Codecs + DSPs), Cryptography, Physical/Scientific Models, Connectivity
  - Each Application Domain has own Characteristics
    - Computational/Algorithmic Structure and Loads
      - Serial, Functional Parallelism (Pipeline Stages), Data Parallelism, Mixed
      - Fine-grained (Tasks) and/or coarse-grained (Threads) computations
    - Communicational Traffic Loads and Timing
      - Periodical/Random transfers of small/big data blocks
    - Characteristics may evolve with algorithmic progress or lifecycle
  - Dynamic System Workload Compositions and User Scenarios
    - Variable Mix of different Application Domains and Loads
  - **System Management @ Runtime becomes indispensable!**

Universität Rostock Traditio et Innovatio

# Introduction

- Examplary Floorplan




Graphic Processor

General Purpose Processor

Multimedia Proceesor

Cryptography Processor

Connectivity

**@DESIGNTIME**

Optimal Integration of Application Domains and Platform Features under consideration of Multiple Workload Scenarios

→ *System Composition*

**@RUNTIME**

Optimal System Operations for current global Workload and/ or dedicated Workload Fractions

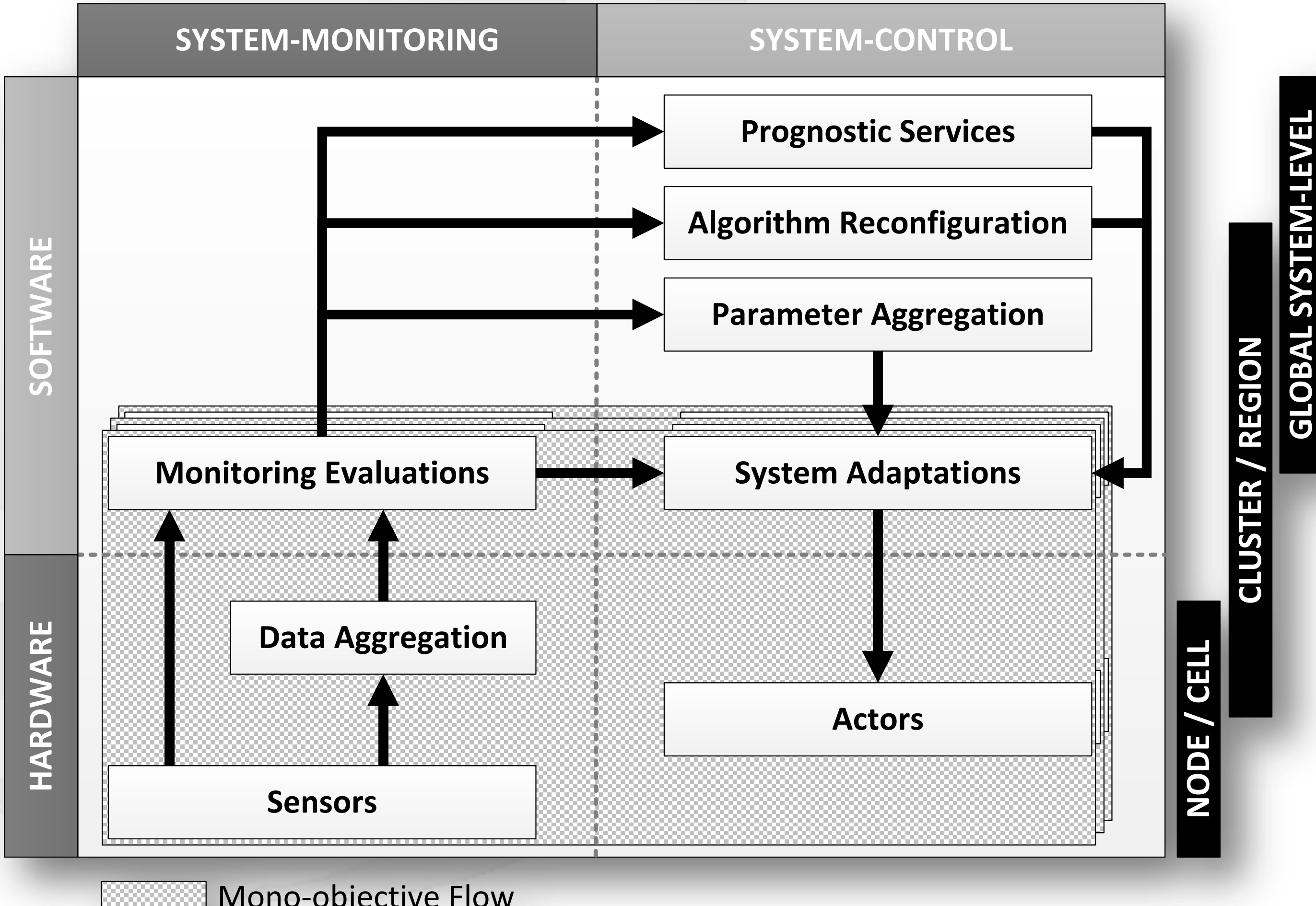
→ *System Management*

# Introduction

- **General Tasks of Runtime-based System Management**
  - Workload Coordination (Mapping/Routing)
  - Energy/Power Configuration (Clocking/Supply-Voltage)
  - Monitoring (Traffic/Temperature/Power/Timing)
  - Reliability (Computation/Communication)
  - Maintenance and Tests (Wear-Out/Aging)
- **Singular Tasks of System Management are highly interdependent and/or complementary with high synergistic potential**
  - i.e. Mapping→Routing→Power→Reliability
- **Implemented mechanisms may work with different spatial scopes**
  - Centralized/Global, Distributed/Local, Hybrid
- **System Management need to be designed as dynamically configurable and multi-objective solution**
  - Specific Management-Task configuration depends on current system workload composition
- **Flexibilty and adaptivity of System Management processes through Hardware/Software-CoDesign**
  - Hardware-Modules : Local Sensors, Aggregators and Switches
  - Software-Modules : Global Evaluation and Decision Logic

# Introduction

- System Management → Monitoring and Control

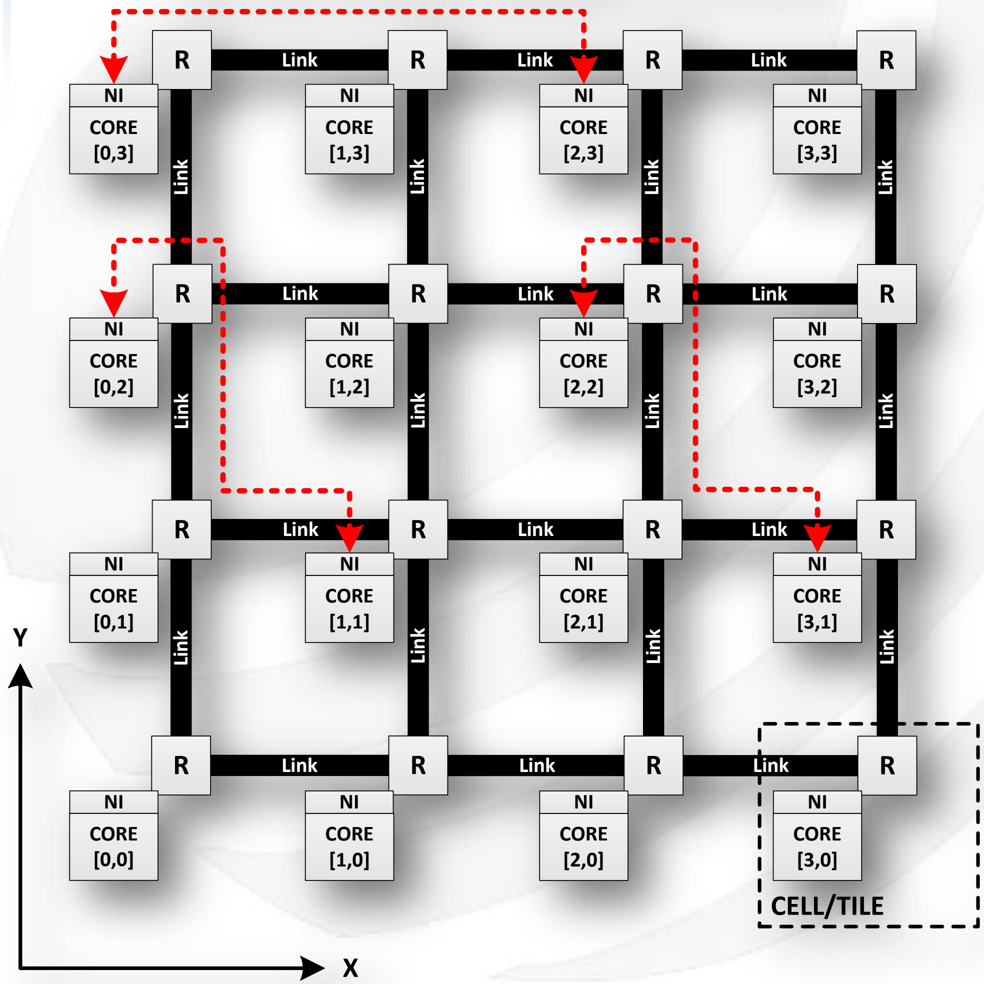


# Introduction

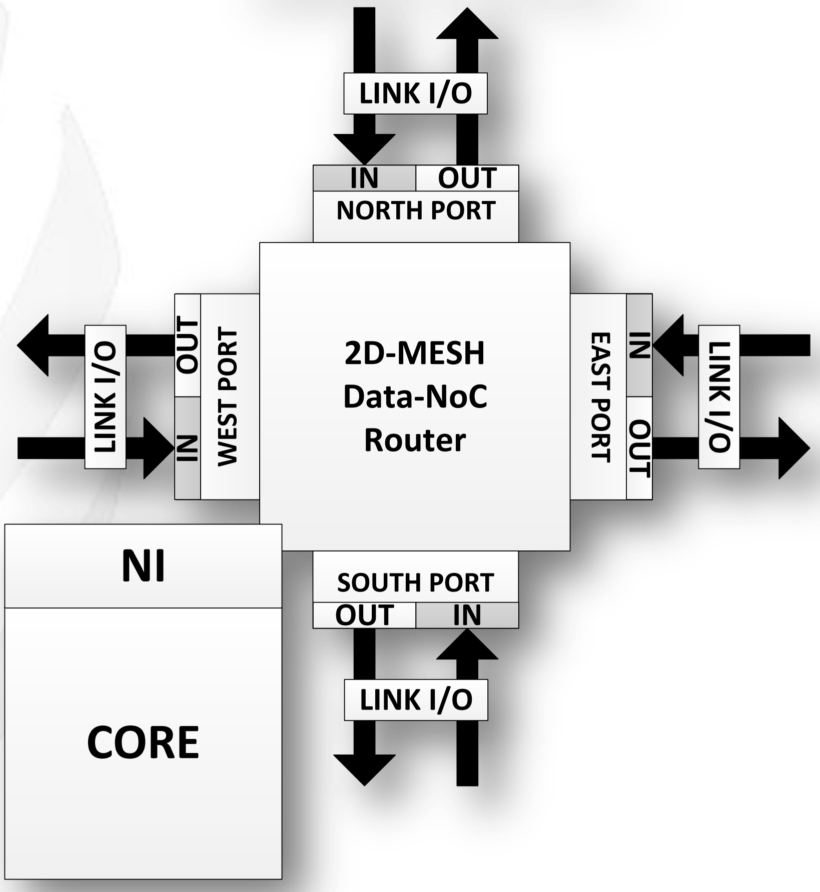
- **Networks-on-Chip (NoC)**
  - Packet-based and globally asynchronous communication on-chip
  - Replacement of conventional bus-based interconnections → Scalability
  - Inherent parallel communication support
- **Basic elements of the NoC are**
  - **IP-Core** = Computational resources that communicates via the NoC
  - **Network-Interface (NI)** = Connection unit of IP-Core and NoC for reception/transmission of packets
  - **Router (R)** = Switching units that lead packets through the NoC from source to destination node
  - **Link** = Bidirectional point-to-point connections transmitting N-Bit datawords in parallel between Routers
  - **Topology** = Connection Graph of Routers and Links
- **Scalable on-chip communication for Many-Core Systems**

# Introduction

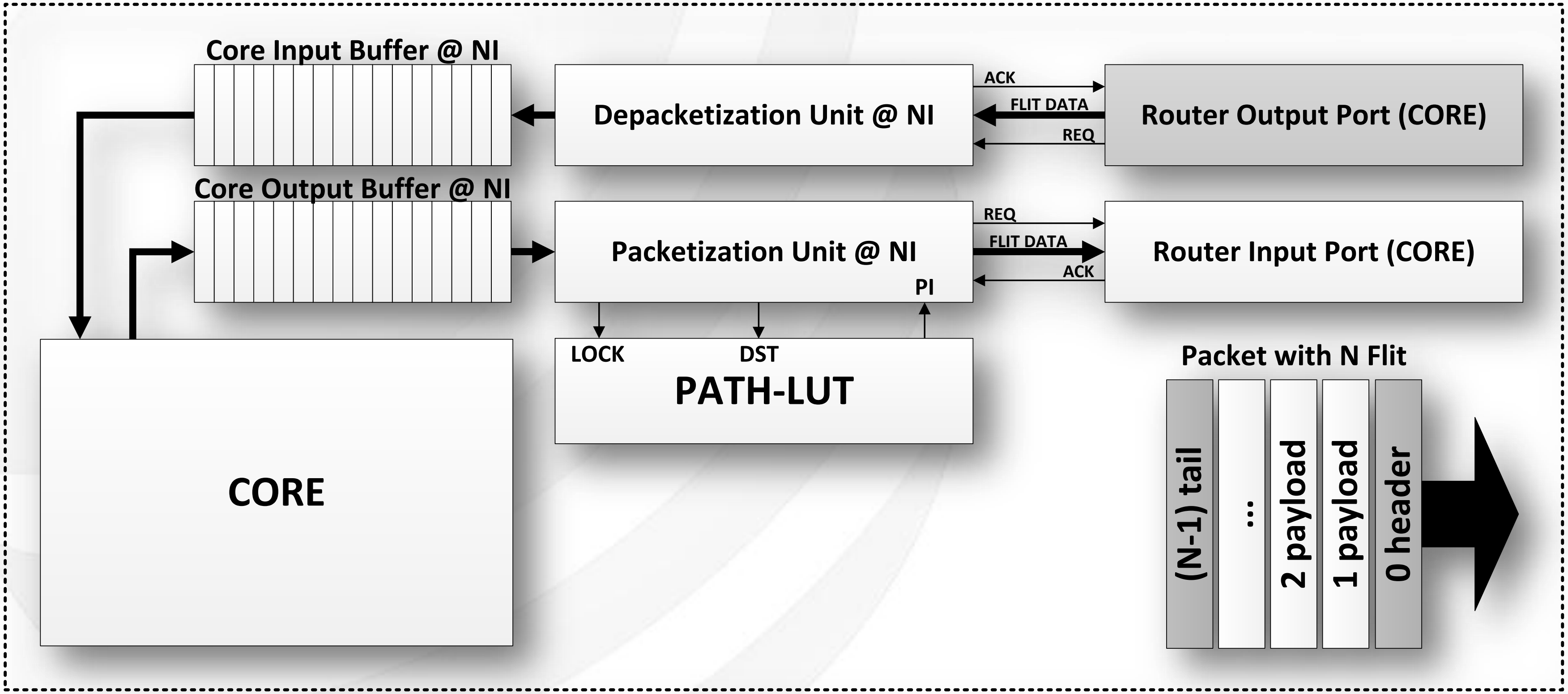
**NoC with 2D-MESH TOPOLOGY** ($N_x$=4, $N_y$=4)



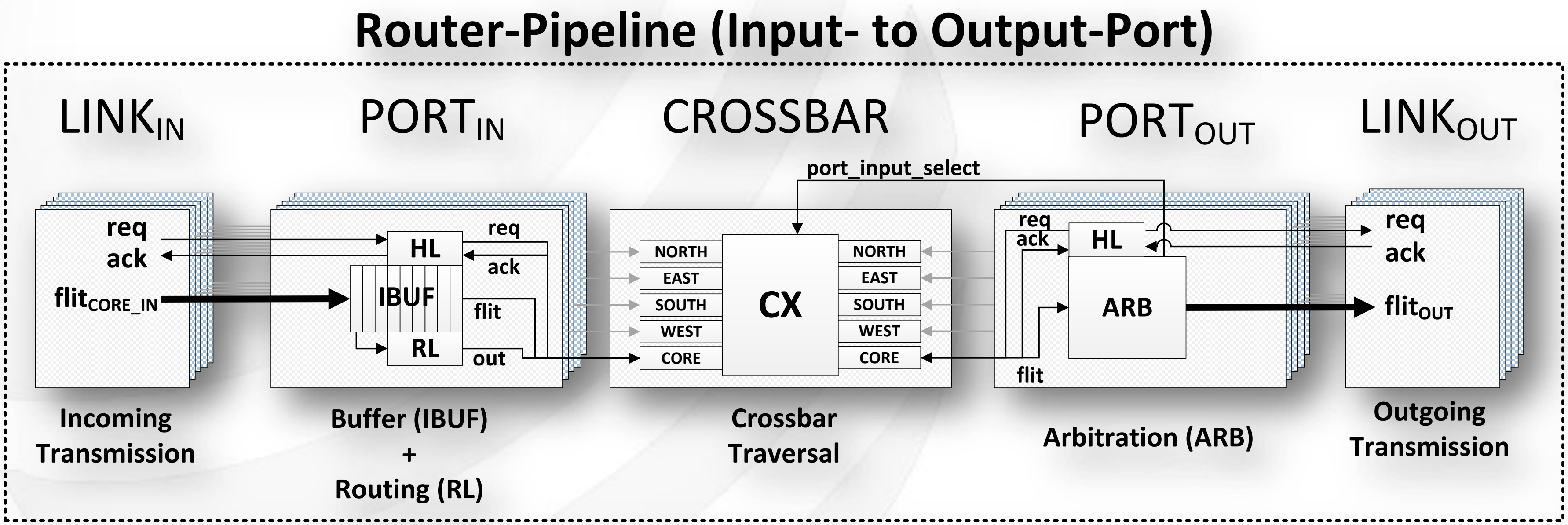

*LINKWIDTH = 8, 16, 32, 64 , …*



# Introduction



# Introduction



# Overview/Structure

- **Introduction**
  - Challenges for Many-Core Platforms
  - Basics of Networks-on-Chip (NoC)
- **RedNoCs**
  - Basic Concept
  - Hardware/Software-based Clustering
  - Traffic Monitoring
  - Temperature Monitoring
  - Experimental Evaluation and Results
- **Centralized Adaptive Source-Routing**
  - Basic Concept
  - Adaptive Centralized XY/YX-Routing
  - Evaluation Model
  - Experimental Evaluation and Results
- **Outlook & Future Work**
- **References**

# RedNoCs – Basic Concept

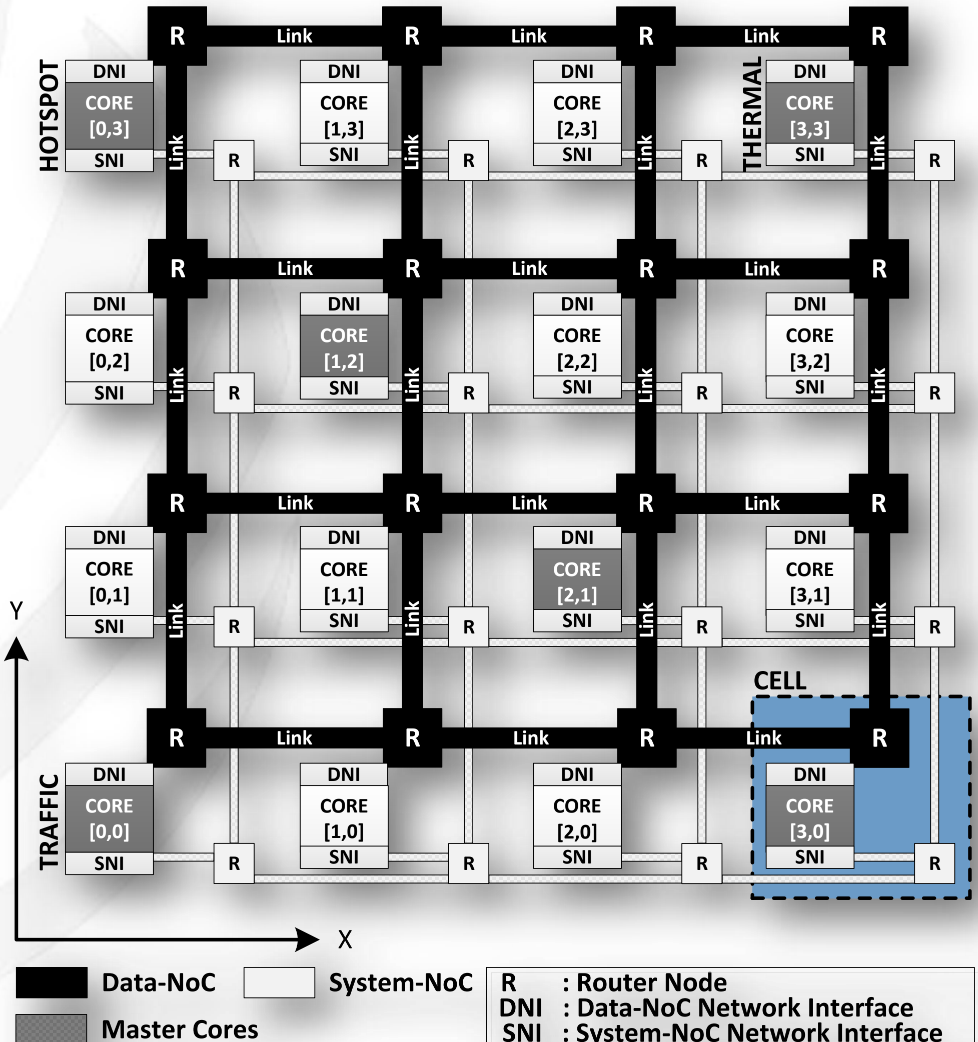
- **Transport infrastructure of System Management information is one of the major critical issues**
  - Should not interfere with Application Data transfers
  - Should be open for multi-objective utilization over various management mechanisms
    - Global/Centralized, Distributed/Local and Hybrid
  - Should transfer System Management Data in time to the desired destination
- **Two major solution approaches can be classified**
  - Shared Infrastructure Solutions (SIS)
  - Exclusive Infrastructure Solutions (EIS)

# RedNoCs – Basic Concept

- In SIS the Application and System Management Data utilize the same NoC
- Bandwidth guarantees for System Management Data through Time-Division-Multiplex (TDM)
  - Priority- or credit-based time slots
  - Virtual-Channels for each traffic category
- **But both traffic categories have different constraints and requirements:**
  - Typical traffic pattern
    - *Applications* → Random or hotspot communication
    - *System Management* → Multicast, N:1 and/or 1:N
  - Typical dataword and thus needed link sizings
    - *Applications* → 32, 64 or 128 Bit
    - *System Management* → 8 – 16 Bit
  - Typical bandwidth demands
    - *Applications* → Gigabyte
    - *System Management* → kByte – Mbyte
- **SIS does not support optimization for both categories → EIS picked**

# RedNoCs – Basic Concept

- **RedNoCs = Red**undant **NoCs**
- **Data-NoC** for Application Data
- **System-NoC** for System Data
- **Both NoCs can be optimized and tested independently**
- IP-Cores now has two NIs
  - Data-NoC Interface (DNI)
  - System-NoC-Interface (SNI)
- **Smallest System Management Unit is the CELL**
- Two types of CELLs
  - Master and Slave
- Master-CELLs run CLUSTER and System Management Agents



# RedNoCs - HW/SW-based Clustering

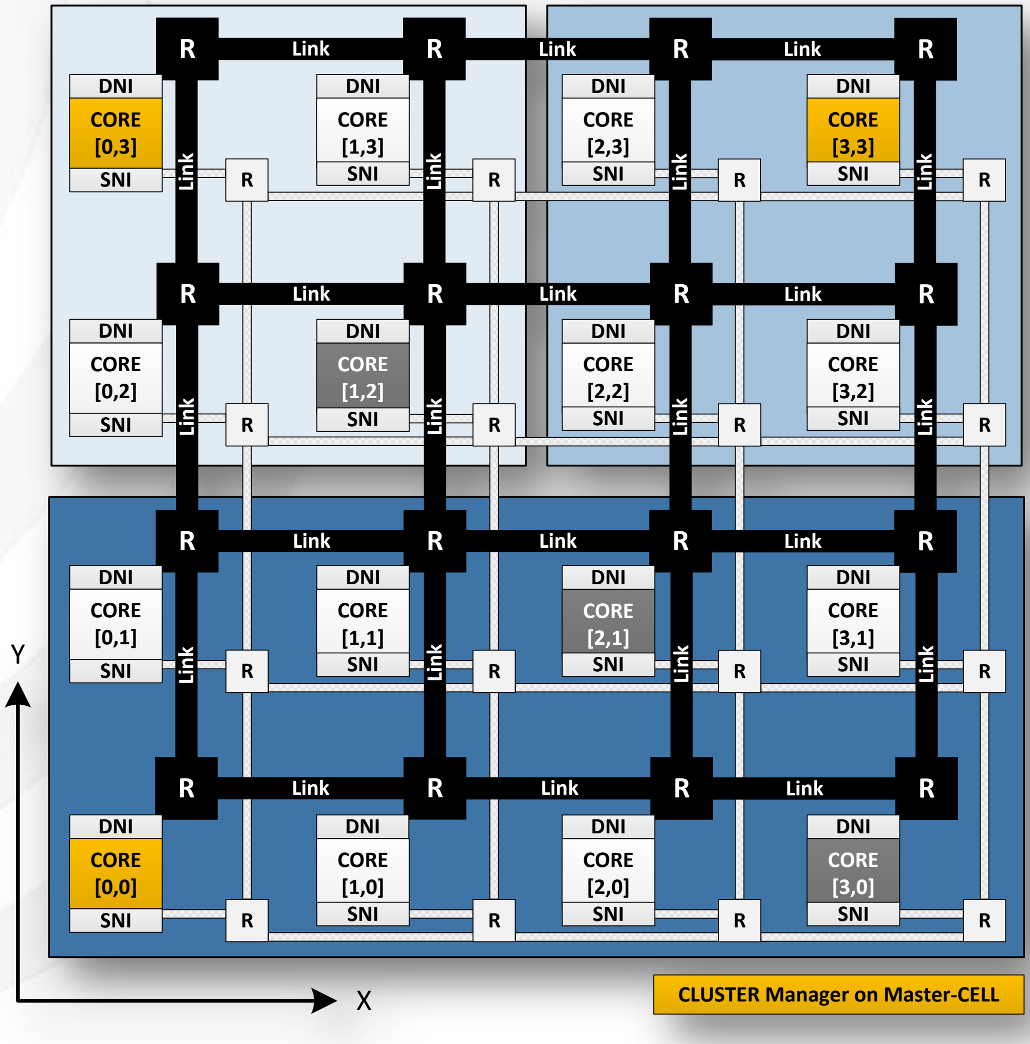
- **As part of RedNoCs a dynamic runtime-based Clustering Concept was defined**
  - The **CLUSTER is rectangular shaped group of CELLs** inside the NoC with a specific context (i.e. Temperatur or Traffic Monitoring)
  - CELLs can be part of different CLUSTERs if they have another context
  - **CLUSTERs will be planed/shaped by global software agents** (i.e. Application Mapper or Power Manager)
  - **SW-based CLUSTER agents realize the building, operation and maintenance**
  - CLUSTER agents aggregates information for global management agents
- **Clustering is completely SW-based**, flexible and extensible for new management mechanims
- **CLUSTER agents can migrate between Master-CELLs**
- The **closed rectangular shaping supports application isolation** for monitoring and control of the assigned workload of each CLUSTER

# RedNoCs - HW/SW-based Clustering

After CLUSTER shaping of global agent is done and send to CLUSTER agent:

1. **Multicast packets from CLUSTER agent to all Slave-CELLs**
   - CLUSTER-Request
   - Context configuration
2. **Collection of Slave-CELLs response packets**
3. **CLUSTER is established**

CLUSTER updates are similar!

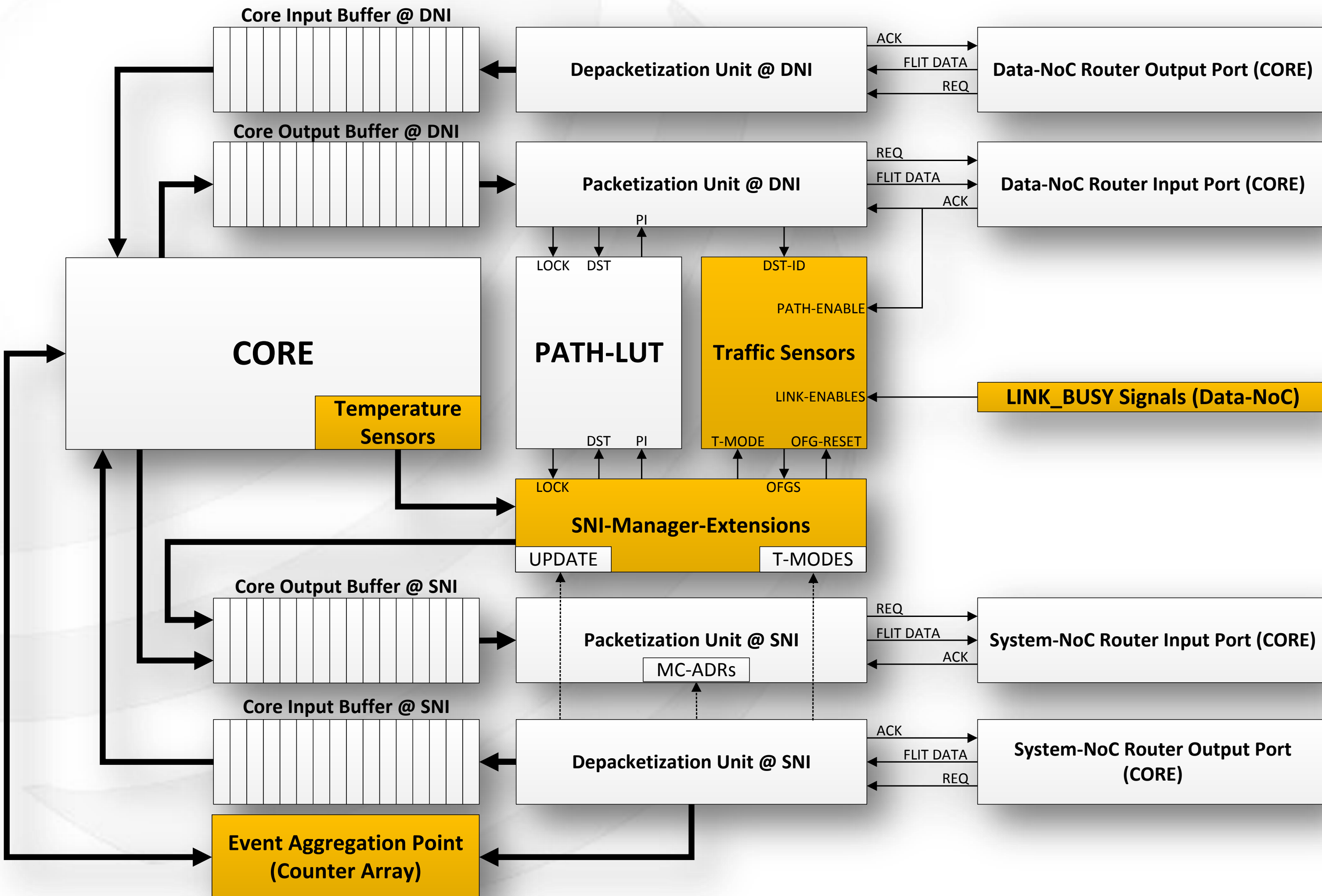


# RedNoCs – Traffic Monitoring

- **Cluster-based Monitoring of Link Loads and Path-based Loads at the IP-Cores**
- Configurable Sensor Capturing Periods and Data Evaluation Cycles for each CLUSTER
- Essential System Management Mechanism for Application Mapping and Adaptive Routing
- **3 hierarchical Levels of Monitoring Information Flow**
  - **LINK/PATH-Level** → Load Sensing
  - **CELL-Level** → Load Reporting to CLUSTER agent
  - **CLUSTER-Level** → Load Aggregation and Evaluation

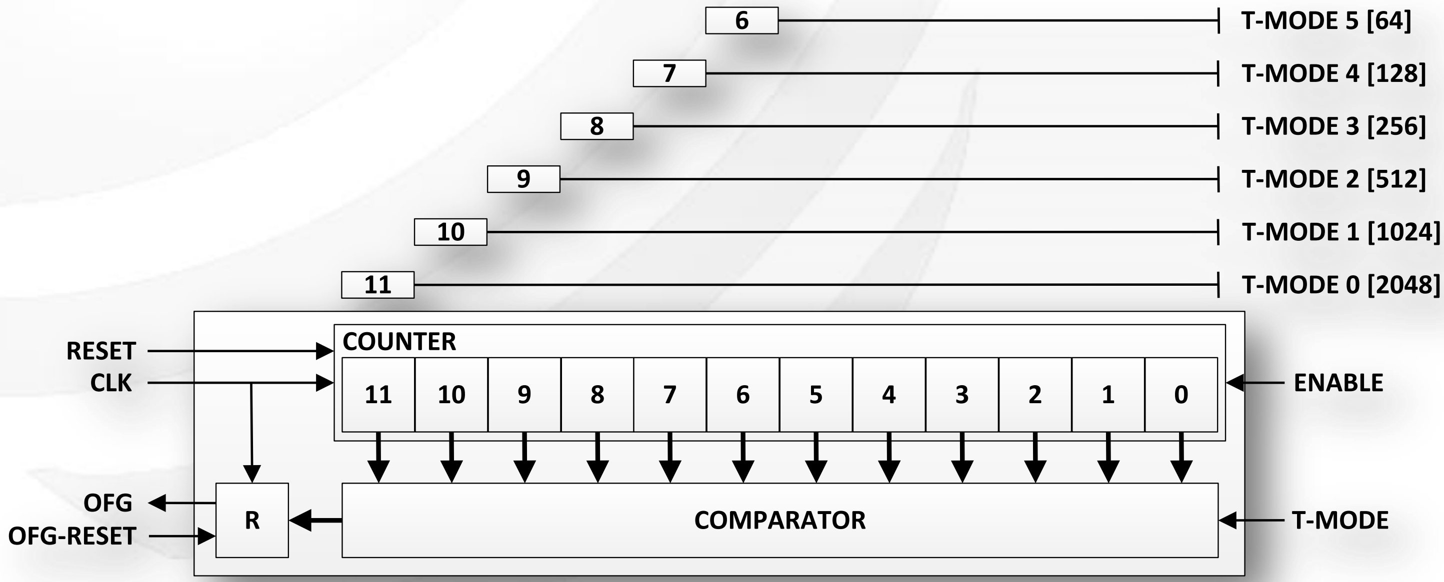
Universität Rostock
Traditio et Innovatio

# RedNoCs – Traffic Monitoring



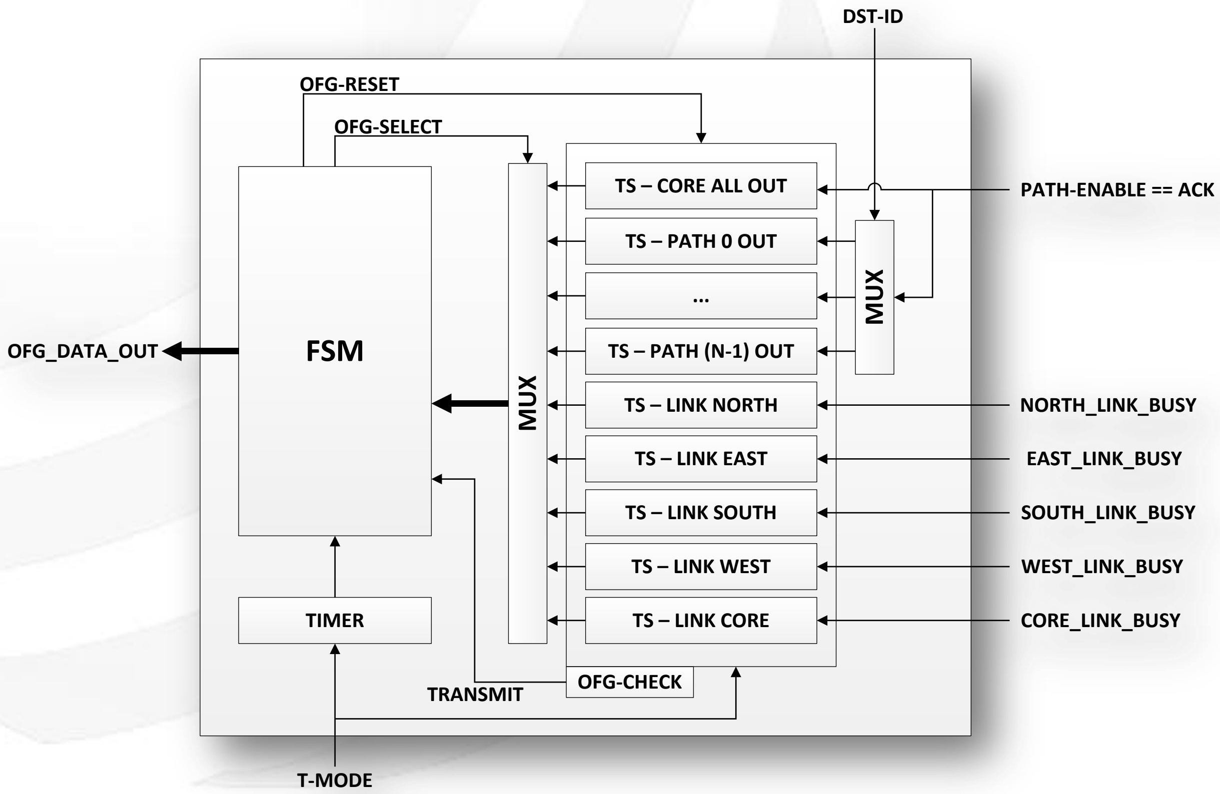
# RedNoCs – Traffic Monitoring

- **LINK/PATH-Level → Traffic Sensors**
  - **Basic Load Sensor = External Triggered Counter + Configurable Comparator**
    - Connected to each Router Output to count busy cycles of links
    - Conneted to DNI to count injected Flits for single Routing Paths
  - **T-MODE defines the counter size** (max # of active counted clock cycles)
  - **T-MODE is set by CLUSTER agent** and equal for all sensors inside the CLUSTER
  - **Comparator raises Overflow Flag (OFG) if T-MODE is reached**



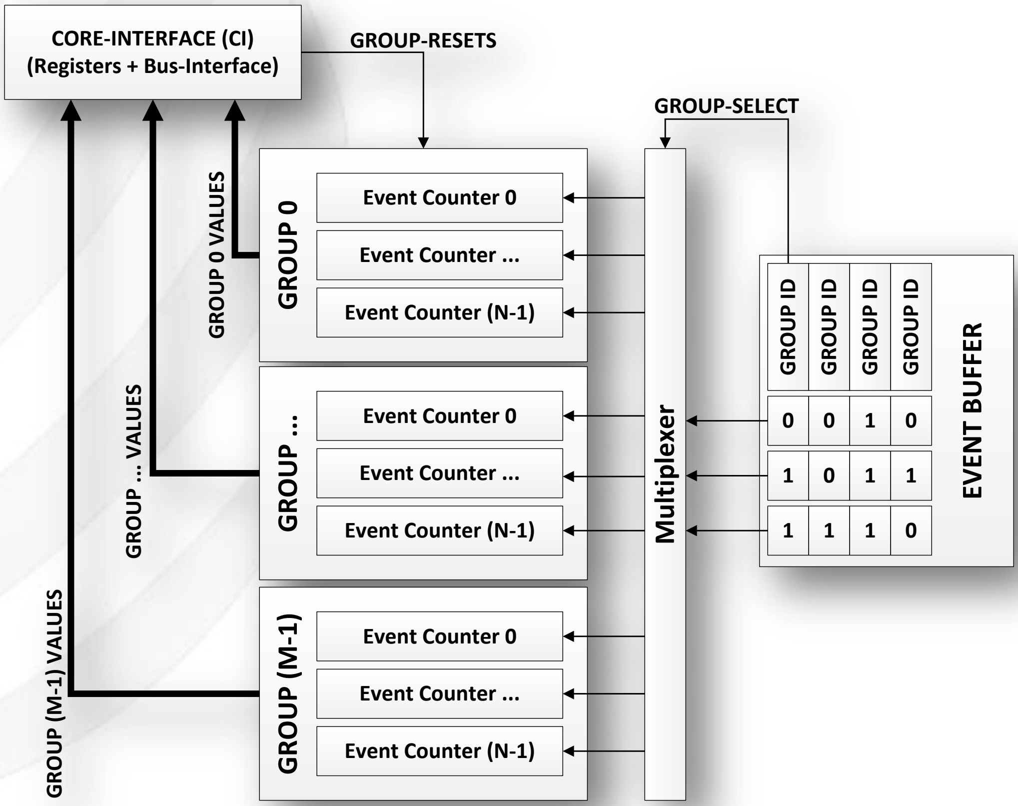
# RedNoCs – Traffic Monitoring

- **CELL-Level → Load Reporting**
  - SNI-Manager-Extension checks every T-MODE period if OFGs occured
  - CHECK = TRUE: Send Monitoring Packet to CLUSTER agent + OFG Reset
  - CHECK = FALSE: Go on with next T-MODE period



# RedNoCs – Traffic Monitoring

- **CLUSTER-Level →Load Aggregation**
  - **Each Master-CELL has an Event-Aggregation-Point** (EAP - Groups of 7-Bit Counter)
  - **EAPs counts Overflow Events for each Traffic Sensor** (OFG = 1 → Event Counter ++)
- **Event Counter organized in Groups**
  - Group ID -> CELL ID/ADRESS
  - Event Counter -> Traffic Sensor
- **EAP counter values direct accessible via CI**
- **EAP save computations**
- **Unused EAPs reusable for:**
  - Online Profiling
  - Execution Monitoring

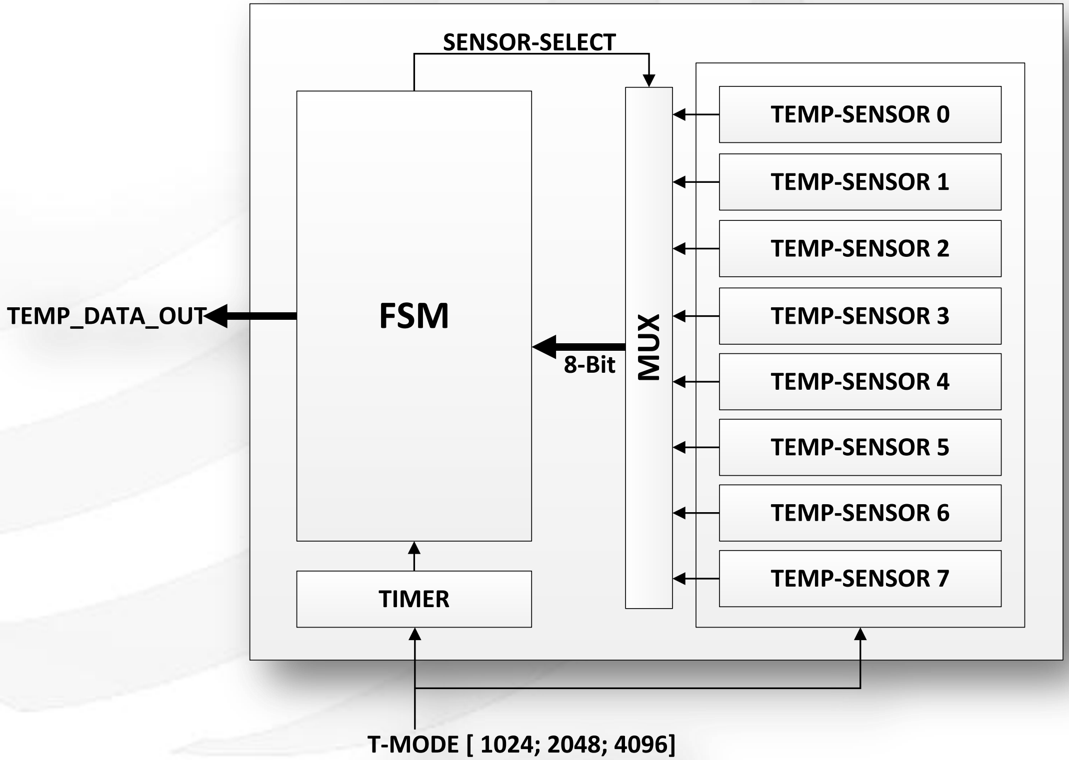


# RedNoCs – Traffic Monitoring

- **CLUSTER-Level → Load Evaluation**
  - **CLUSTER agent captures Event Counter after N full T-MODE periods** (=Monitoring Cycle) **via CI**
    - Afterwards Event Counter will be reset for next Monitoring Cycle
  - **Captured Counter Values are scaled Loads from 0 – 100% if**
    - **N = 100/S with S = OFG Weight**
      - S = 1 → Monitoring Cycle = 100*T-MODE with 1% Scale Resolution
      - S = 2 → Monitoring Cycle = 50*T-MODE with 2% Scale Resolution
      - S = 4 → Monitoring Cycle = 25*T-MODE with 4% Scale Resolution
  - **No additional computations by CLUSTER agent to generate final Load Statistics**
  - **Monitoring Cycles and thus Traffic Monitoring are fully configurable in timing and accuracy due to current Workload of the CLUSTER**
  - T-MODE ~ ns → Monitoring Cycle ~ µs
  - **CLUSTER agent enabled to observe progress inside single Monitoring Cycles via snapshots**

# RedNoCs – Temperature Monitoring

- Each IP-Core has integrated temperature sensors that produce 8-Bit data
- Sensor data will be read out every T-MODE period and transmitted to the CLUSTER agent via SNI-Manager-Extension
- Needed T-MODE period and sensor basics taken from solution in [1]



# RedNoCs – Evaluation and Results

- **Setup for Simulation and ASIC-Synthesis**
  - System-NoC has a minimal design to generate low hardware overhead
  - Full System Simulation via SystemC [2] Hardware Modeling at Worst-Case
  - Hardware Synthesis using Synopsys Design Compiler® and 45nm Nangate Library [3]

| Parameter | Value |
|---|---|
| Clock Rate | **1 ns** |
| Topology | **2D-Mesh** |
| NoC-Size | **8 x 8** |
| Cluster Sizing | **4 x 4, 8 x 2** (=16 Cores) |
| Linkwidth (net/gross) | **8-Bit / 11-Bit** |
| Input Buffer Depth @ Router | **1 Flit** (minimal) |
| # of Master-CELLs | **32** (=50%) |
| Temperature Sensors per CELL | **8** |
| Traffic Sensors per CELL | **21** (5 Link and 16 Path Sensors) |
| 7-Bit Counter per EAP | **336** (16 Groups with 21 Counter) |
| Core-2-Core Communcation | **25 -50 Mbyte/s each Core** |
| Core-2-Core Traffic Pattern | **random, transpose, hotspot** (H=20%) and **bit complement** |

# RedNoCs – Evaluation and Results

- **Tables show the achieved packet latency results with configurations that not lead to deadline misses of T-MODE periods**
- TRC T-MODE = 256/512 ns → Traffic Monitoring Cycles = 25,6/51,2 µs (S=1)
- TMP T-MODE = 2,048 µs
- CLUSTER agent were positioned at the lower left and upper right CELLs
- RedNoCs allows three different System Management mechanisms to communicate

| 4x4 CLUSTER | Parameter Configurations | | | Packet Latencies [# of clock cycles] | | | |
|---|---|---|---|---|---|---|---|
| C2C Pattern | TMP T-MODE | C2C IR | TRC T-MODE | TRC | TMP | C2C | UPDATE |
| hotspot | 2048 | 25 Mbyte | 256 | 89,1 | 167,7 | 69,5 | 1000 |
| uniform | 2048 | 50 Mbyte | 256 | 93,8 | 172,2 | 69,4 | 576 |
| bit comp | 2048 | 50 Mbyte | 256 | 92,1 | 171,8 | 60,3 | 610 |
| transpose | 2048 | 50 Mbyte | 256 | 92,6 | 173,7 | 66,8 | 610 |

| 8x2 CLUSTER | Parameter Configurations | | | Packet Latencies [# of clock cycles] | | | |
|---|---|---|---|---|---|---|---|
| C2C Pattern | TMP T-MODE | C2C IR | TRC T-MODE | TRC | TMP | C2C | UPDATE |
| hotspot | 2048 | 25 Mbyte | 512 | 83,4 | 133,1 | 45,9 | 1003 |
| uniform | 2048 | 25 Mbyte | 256 | 74,7 | 140,4 | 41,8 | 795 |
| bit comp | 2048 | 25 Mbyte | 256 | 74,2 | 140,1 | 40,9 | 767 |
| transpose | 2048 | 25 Mbyte | 256 | 73,6 | 140,5 | 40,8 | 779 |

# RedNoCs – Evaluation and Results

- **Hardware Synthesis shows that overhead of additional System-NoC is feasible**

| | Total Cell Area @ 45nm ASIC | |
|---|---|---|
| **Design Component** | *CELL [µm²]* | *8 x 8 NoC [mm²]* |
| System-NoC Router | 3285,9 | 0,194 |
| Traffic Monitoring SNI-Extension | 1511,94 | 0,097 |
| Temperature Monitoring SNI-Extension | 851,99 | 0,055 |
| Traffic Sensors | 2720,34 | 0,174 |
| Event-Aggregation-Points | 22230,68 | 0,711 |
| **All Components** | **30600,85** | **1,231** |
| | *Relative to the Area of Intels SCC [4] in % ->* | **0,217** |

# RedNoCs – Evaluation and Results

- **What was not part of this presentation ?**
  - *State-of-the-Art overview*
  - *Details about optimization strategies used for RedNoCs*
  - *Details about the Clustering*
- **But you will find it in the paper!**
- **Now we go on with an scenario of applied RedNoCs for dynamic Routing**

# Overview/Structure

- **Introduction**
  - Challenges for Many-Core Platforms
  - Basics of Networks-on-Chip (NoC)
- **RedNoCs**
  - Basic Concept
  - Hardware/Software-based Clustering
  - Traffic Monitoring
  - Temperature Monitoring
  - Experimental Evaluation and Results
- **Centralized Adaptive Source-Routing**
  - Basic Concept
  - Adaptive Centralized XY/YX-Routing
  - Evaluation Model
  - Experimental Evaluation and Results
- **Outlook & Future Work**
- **References**

# Centralized Adaptive Source Routing

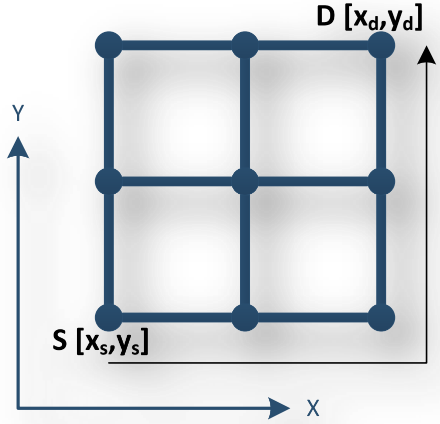
- Adaptive Routing in NoCs is the major mechanism to create balanced traffic loads and avoid/solve congestions → Optimization!
- Many algorithms were presented over the last years that mainly differ on three aspects:
  - The location of path updates
    - **Distributed** → Each router decides about path updates of the current packet
    - **Centralized** → One central node calculates all path updates for all ip-cores/routers
  - The scope of evaluated traffic monitoring data
    - **Local/Regional** → Aggregated traffic statistics of neighboring router will be used for update calculations
    - **Global** → Traffic statistics from all relevant nodes in the NoC are known and used for update calculations
  - The scope of allowed path adaptations
    - **Minimal** → The exploration of longer paths than minimal one is forbidden
    - **Non-Minimal** → The exploration of indirections is allowed
- Mainly all runtime algorithms proposes a full hardware implementation
- This work proposes a hardware/software-combination based on the Traffic Monitoring and Clustering of RedNoCs

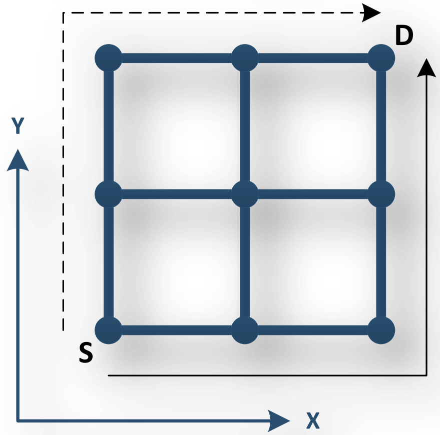
# Centralized Adaptive Source Routing

- **This work was inspired by ATDOR** [5]
  - First fully suitable and centralized adaptive source routing with global traffic consideration
  - Simple and effective update strategy with successive optimization
  - Monitoring and path evaluation are purely implemented in hardware
- **The following changes were realized**
  - Centralized adaptive source routing is applied to RedNoCs CLUSTER with individual configuration
  - Path evaluation is realized as software agent at CLUSTER-Level
  - Path updates will be distributed via System-NoC
- **Comparision of own routing with two different algorithms**
  - XY and O1TURN [6]
- **Simulation for different traffic characteristics**

# Centralized Adaptive Source Routing

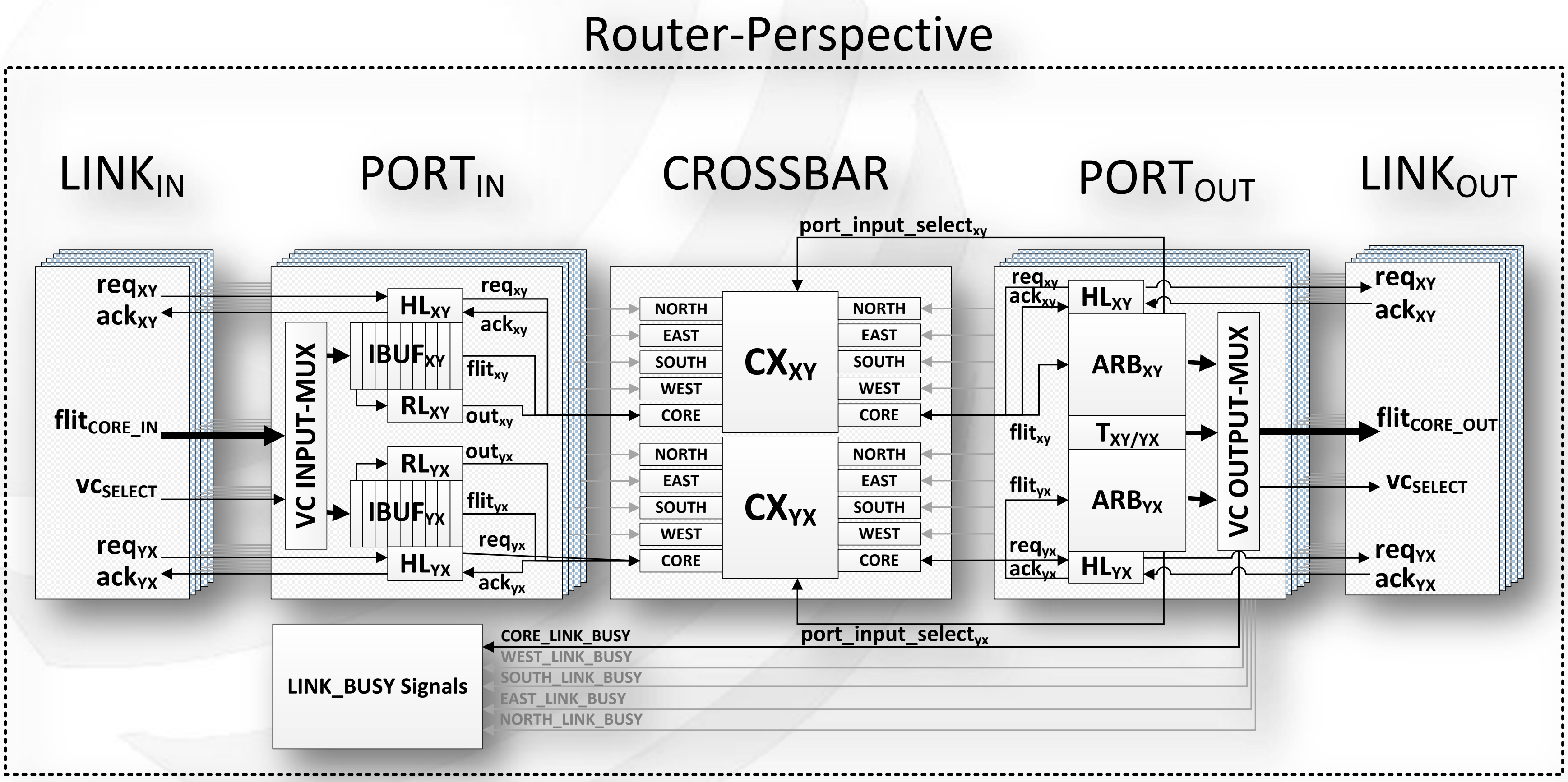
- **Exploration of XY- and YX-Path for minimal load sum of passed links**
  - Feasible computational costs
  - Low hardware overhead
    - Direction Bit [XY=0;YX=1] needed
    - One register per entry at Path-LUT to store Direction Bit
    - Two Virtual-Channels (VCs) for XY and YX traffic needed to avoid deadlocks
  - Minimal path routing
  - Static Virtual-Channel assignment at NI
- **Consideration of global traffic statistics**
- **Updates applied at Path-LUT of DNI**
- **Each CLUSTER has own timing configurations**
  - Dynamic updates at runtime possible
  - Isolation of workload fractions
- **Software agent allows flexible adaptation of path update strategy**



# Centralized Adaptive Source Routing

- **Router-Pipeline for 2 Virtual-Channels (XY and YX)**




IBUF : Port Input Buffer
HL : Handshake Logic
RL : Routing Logic
ARB : Port Output Arbitration Logic
T : Virtual Channel Toggler
CX : Multiplexer Crossbar

# Centralized Adaptive Source Routing

## Pseudo-Code of path update evaluation

**ENTRY:** Finalized Traffic Monitoring Cycle → **Traffic statistics for all paths and links available**

```
for all x < N_X do
   for all y < N_Y do
      for all path_entries at core(x,y) do
         if (path_load > 0) && (no_shared_dimensions) then
            remove_path_load_from_statistics();
            calculate_load_sum_XY(); → Sum of link loads at XY path
            calculate_load_sum_YX(); → Sum of link loads at YX path
            compare_and_select();  → Path with smaller load becomes selected
            add_path_load_to_statistics();
         end if;
         if (# of updates > 0) then
            generate_path_update_packet();
         end if;
      end for;
   end for;
end for;
```

# Centralized Adaptive Source Routing

- **Setup for Simulation and ASIC-Synthesis**
  - Similar to RedNoCs but more Data-NoC design and traffic parameter

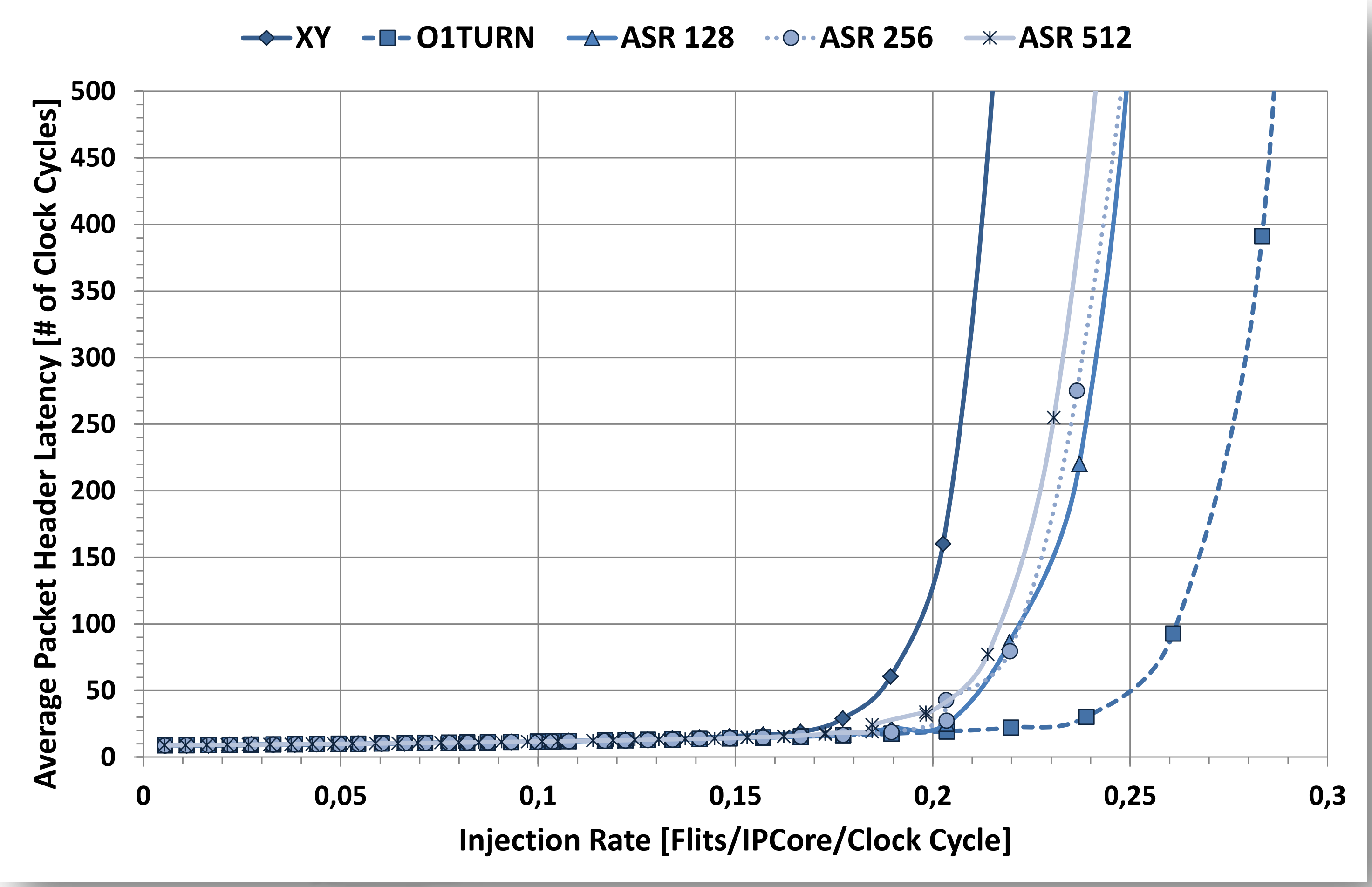
| Parameter | Value |
|---|---|
| NoC Clock Rate | **1 ns** |
| Master-CELL CPU Clock Rate | **0.5 ns** |
| NoC Topology | **2D-Mesh** |
| Synthesized NoC-Size | **8 x 8** |
| Simulated Cluster Size | **16 CELLS at 4 x 4 spatial shape** |
| Data-NoC Input Buffer Depth | **4 Flit** |
| System-NoC Input Buffer Depth | **1 Flit** |
| Data-NoC Linkwidth | **66 Bit** |
| System-NoC Linkwidth | **11 Bit** |
| Average Link wire length [7] | **0.83 mm** |
| Wire width/spacing [7] | **140/140 nm** |
| # of I/O-Links in 8x8 NoC | **224** |
| # of Master-CELLs in NoC | **32** (=50%) |
| Traffic Sensors per CELL | **21** |
| 7-Bit Counter per EAP | **336** (=16*21) |
| Simulated Data-NoC Packet Sizes | **2 - 9 Flit** (uniform) |
| Traffic Pattern | **random**, **transpose**, **hotspot** (H=20%) and **bit complement** Hotspot Position = CORE [0,0] |

# Centralized Adaptive Source Routing

- **Data-NoC simulation were done for two different traffic constellations:**
  - **Homogeneous:** All IP-Cores inject packets with the same injection rate / bandwidth
  - **Heterogeneous:** All IP-Cores inject packets with a random injection rate / bandwidth selected from a given interval
- **Selection of packet destinations depends on the simulated traffic pattern**
- **Proposed solution was simulated for different T-MODEs (128, 256, 512)**
- Presented results are average values from at least 100 simulation runs per parameter variation
- Results show variation of improvements for simulated traffic constellations
- Average number of calculated path updates indicates convergence

# Centralized Adaptive Source Routing

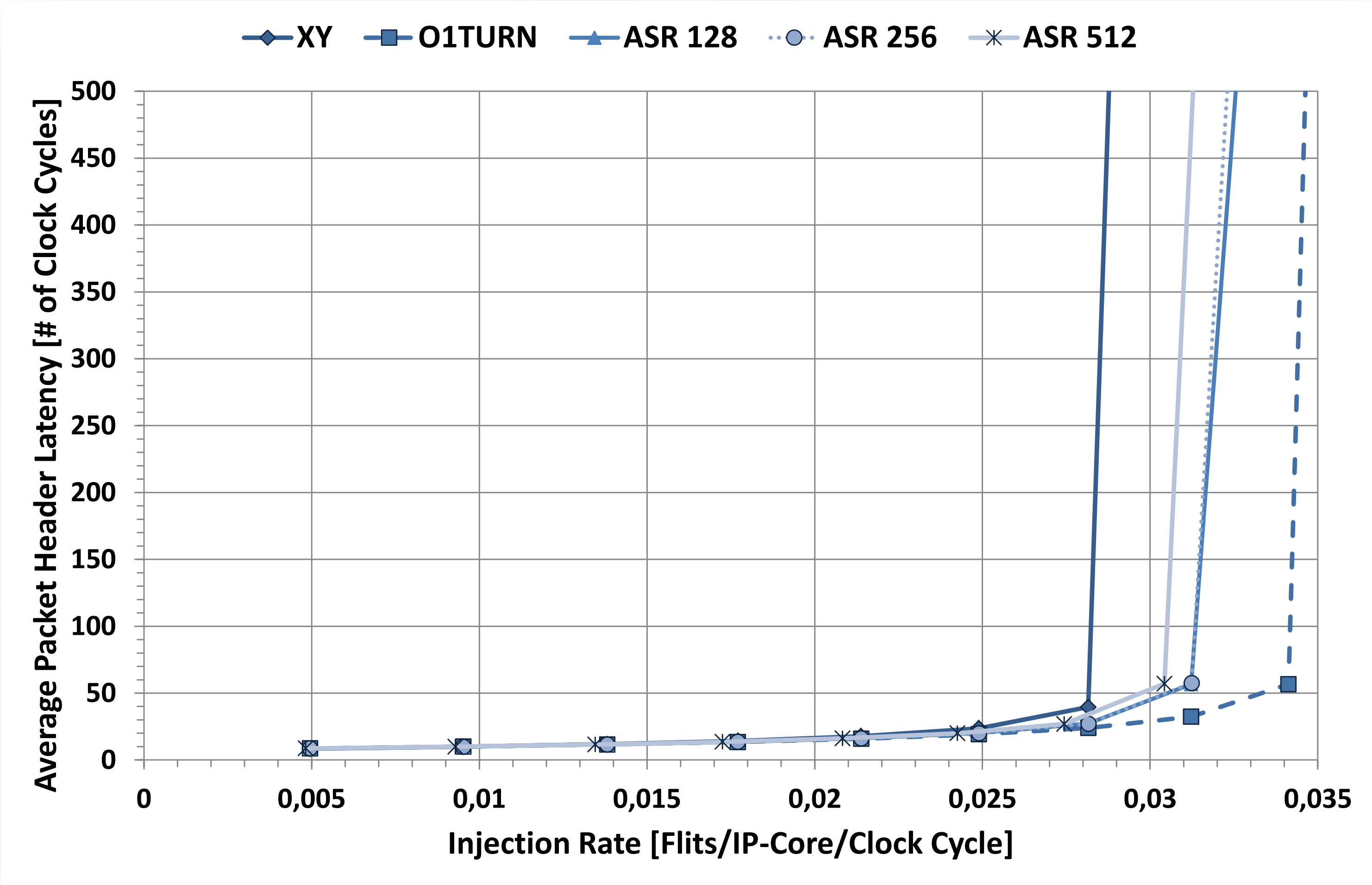
- Homogeneous random traffic pattern results



Universität Rostock
Traditio et Innovatio

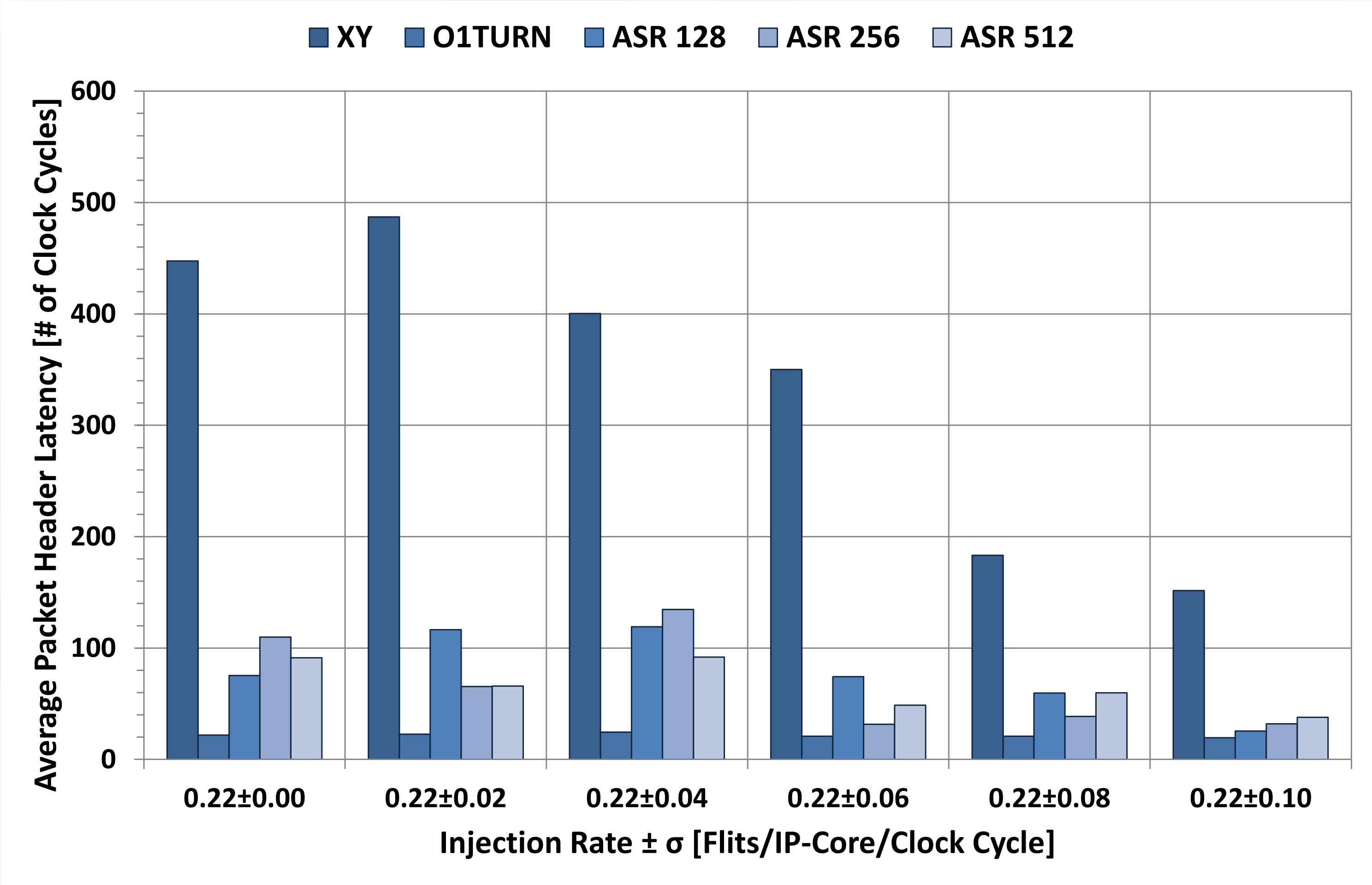
# Centralized Adaptive Source Routing

- Homogeneous hotspot traffic pattern results



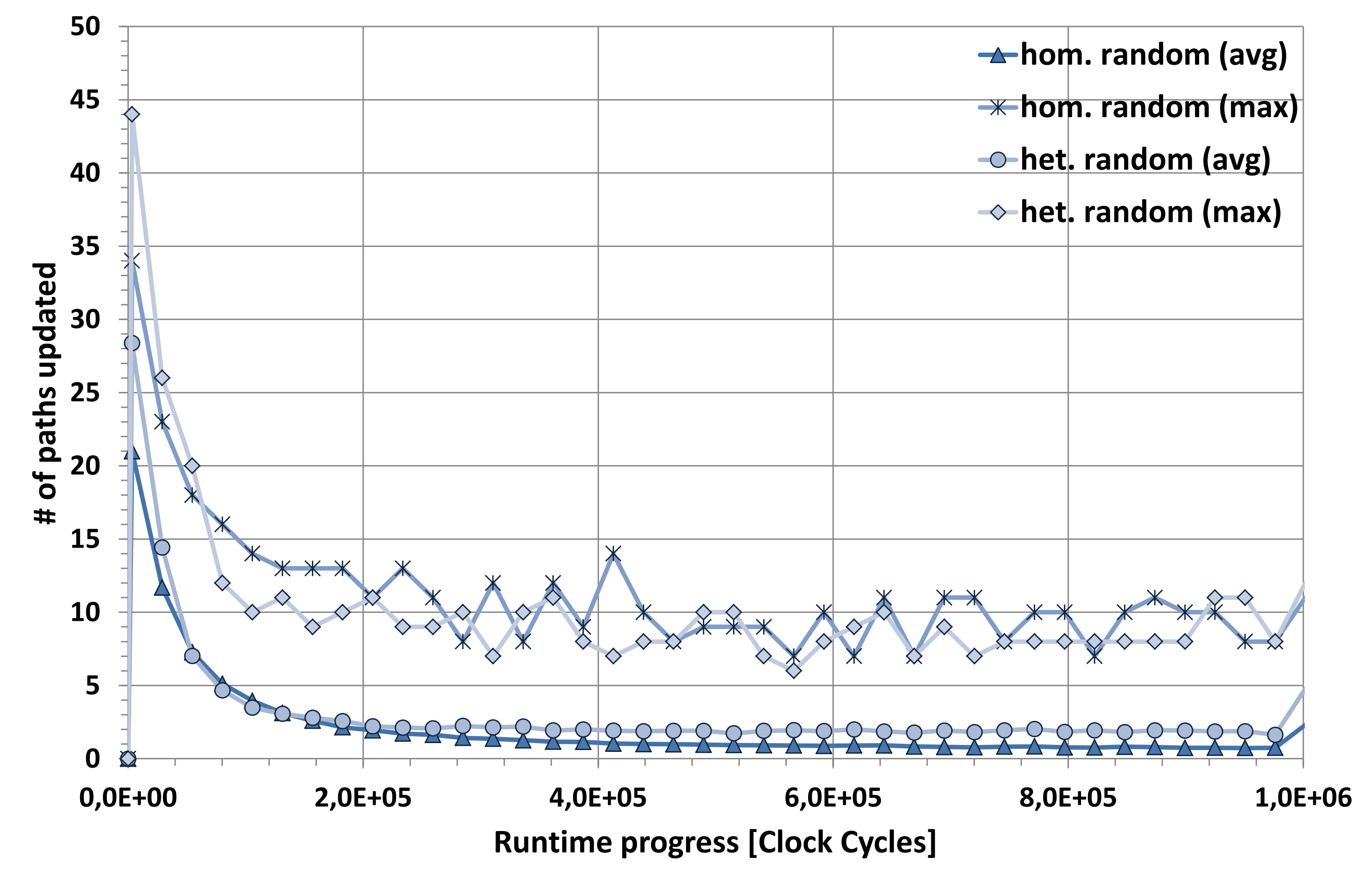
# Centralized Adaptive Source Routing

- Heterogeneous random traffic pattern results



# Centralized Adaptive Source Routing

- Path update statistics for different simulation run configurations



# Centralized Adaptive Source Routing

- Runtime profiling results on 32-Bit PowerPC 440 in Xilkernel

| NoC Size | Maxhop | PREPARE/UPDATE | XY/YX |
|---|---|---|---|
| 4 x 4 | - | 36 | 132 |
| 4 x 4 | 4 | 34 | 118 |
| 4 x 4 | 3 | 32 | 104 |
| 8 x 8 | - | 46 | 174 |
| 8 x 8 | 8 | 42 | 157 |
| 8 x 8 | 4 | 34 | 119 |
| 8 x 8 | 3 | 32 | 105 |

- 45nm Nangate OpenPDK Hardware Synthesis results

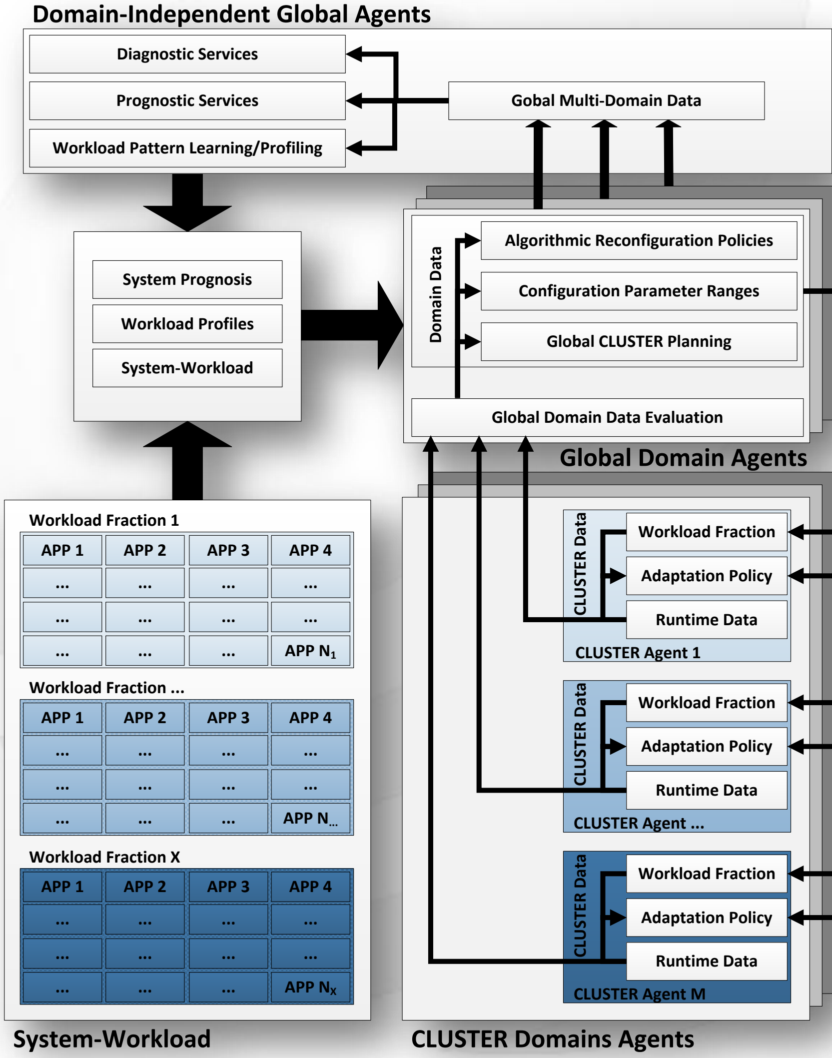
| | Total Cell Area @ 45nm ASIC | |
|---|---|---|
| **Design Component** | *CELL [µm²]* | *8x8 NoC [mm²]* |
| SYSTEM-NOC ROUTER | 2783.69 | 0.178156 |
| SYSTEM-NOC LINK | 1510,6* | 0.3383744 |
| DATA-NOC ROUTER | 40174.51 | 2.571169 |
| DATA-NOC LINK | 14641.2* | 3.279628 |
| SNI TRAFFIC LOAD EXT. | 1511.94 | 0.096764 |
| TRAFFIC SENSORS | 2720.34 | 0.174102 |
| AGGREGATION POINT | 22230.68 | 0.711382 |
| **SUM OF ALL UNITS** | **85572.96** | **7.3495754** (< 2%) |

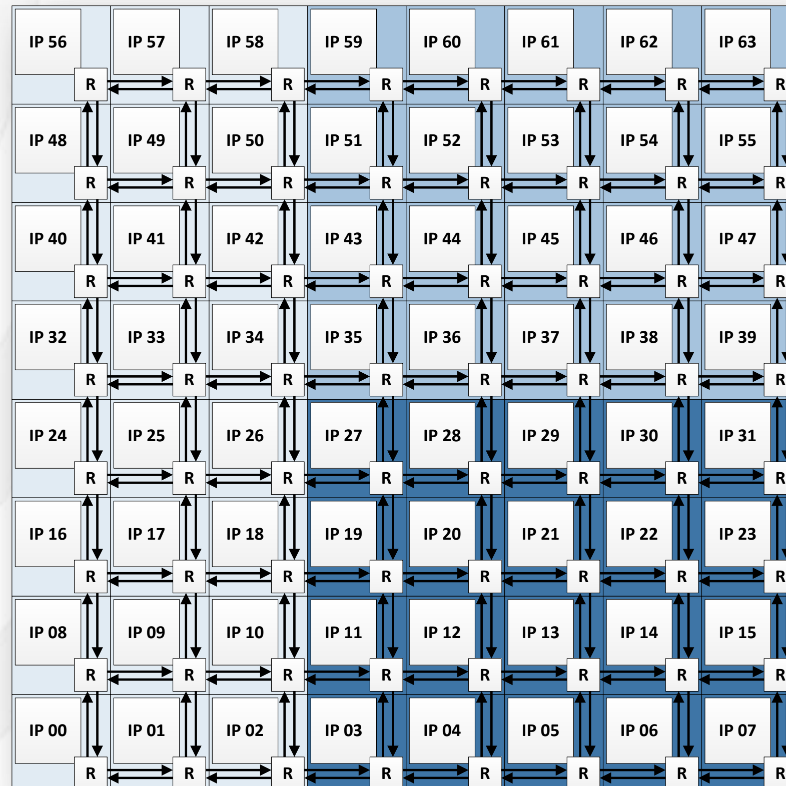
**area of a single I/O NoC-Link*

# Overview/Structure

- **Introduction**
  - Challenges for Many-Core Platforms
  - Basics of Networks-on-Chip (NoC)
- **RedNoCs**
  - Basic Concept
  - Hardware/Software-based Clustering
  - Traffic Monitoring
  - Temperature Monitoring
  - Cluster Management
  - Experimental Evaluation and Results
- **Centralized Adaptive Source-Routing**
  - Basic Concept
  - Adaptive Centralized XY/YX-Routing
  - Evaluation Model
  - Experimental Evaluation and Results
- **Outlook & Future Work**
- **References**

# Outlook & Future Work



# THANK YOU!

# Reference


[1] J. Zhao, S. Madduri, R. Vadlamani, W. Burleson, and R. Tessier, "A Dedicated Monitoring Infrastructure for Multicore Processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 6, pp. 1011–1022, Jun. 2011.

[2] IEEE 1666: SystemC language – 2011, http://www.systemc.org/

[3] NanGate FreePDK45 Generic Open Cell Library, https://www.si2.org/openeda.si2.org/projects/nangatelib

[4] J. Howard et. al., "A 48-Core IA-32 Processor in 45 nm CMOS Using On-Die Message-Passing and DVFS for Performance and Power Scaling", IEEE JOURNAL OF SOLID-STATE CIRCUITS, VOL. 46, NO. 1, JANUARY 2011

[5] R. Manevich, I. Cidon, A. Kolodny, I. Walter, and S. Wimer, "A Cost Effective Centralized Adaptive Routing for Networks-on-Chip," *2011 14th Euromicro Conference on Digital System Design*, vol. 9, no. 2, pp. 39–46, Aug. 2011.

[6] M. H. Cho, M. Lis, K. S. Shim, M. Kinsy, and S. Devadas, "Path-based, randomized, oblivious, minimal routing," in *Proceedings of the 2nd International Workshop on Network on Chip Architectures - NoCArc '09*, 2009, p. 23

[7] C. Hernandez, F. Silla, and J. Duato, "A methodology for the characterization of process variation in NoC links," in *Design, Automation & Test in Europe Conference & Exhibition (DATE'10)*, 2010, pp. 685–690