Centralized Traffic Monitoring for online-resizable Clusters in Networks-on-Chip

Philipp Gorski, Dirk Timmermann
Institute of Applied Microelectronics and Computer Engineering
University of Rostock, Germany
{philipp.gorski2, dirk.timmermann}@uni-rostock.de
Outline

• Introduction
  – Networks-on-Chip (NoC)
  – Why Traffic Monitoring?

• Traffic Monitoring
  – Basic Concept
  – Dual NoC Infrastructure
  – Hardware/Software-based Clustering
  – Sensing and Flow
  – Experimental Results

• Outlook & Future Work
Introduction – Networks-on-Chip

• **Networks-on-Chip (NoC)**
  – Packet-based and globally asynchronous communication on-chip
  – Replacement of bus-based interconnections $\rightarrow$ Scalability & Parallelism

• **Basic elements of the NoC:**
  – **Ipcore** = Computational resource that communicates via the NoC
  – **Network-Interface (NI)** = Connection of Ipcore and NoC for reception/transmission of packets
  – **Router (R)** = Switching units that lead packets through the NoC from source to destination Ipcore
  – **Link** = Bidirectional point-to-point connections between Routers
  – **Topology** = Connection Graph of Ipcores, Routers and Links

• **Scalable on-chip communication for Chip Multiprocessor (CMP) Systems**
Introduction – Networks-on-Chip

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

- XY-Routing, Wormhole Switching, REQ/ACK Flow Control, Input Buffering and Round Robin Arbitration
Introduction – Why Traffic Monitoring?

- **Increasing parallelism** in CMP (64, 128, 256, 512, ... Ipcores)
- **Communication becomes dominant** for performance and energy consumption
- **Runtime-based management** mechanisms **need** current **traffic information**
  - Application Mapping / Workload Management
  - Adaptive Routing / Traffic Management
  - Application Profiling / Debugging
  - Wear-out and Degradation Management
- **To achieve cooperative interaction of these mechanisms a common information base is needed**
Outline

• Introduction
  – Networks-on-Chip (NoC)
  – Why Traffic Monitoring?

• Traffic Monitoring
  – Basic Concept
  – Dual NoC Infrastructure
  – Hardware/Software-based Clustering
  – Sensing and Flow
  – Experimental Results

• Outlook & Future Work
Traffic Monitoring – Basic Concept

• Design criteria:
  – Monitoring inside a Region of Interest ➔ Isolated workload fractions
  – Monitoring with a Resolution of Interest ➔ Timing & Accuracy
  – Reuse of Information/Infrastructure ➔ Mapping, Routing, Profiling

• Flexible hardware/software solution
  – Counter-based activity sensing/aggregation in hardware ➔ Simple Events
  – Monitoring data evaluation and management in software

• Centralized collection of all link- and pathloads inside a defined Region
  – All loads scaled to 0-100% with resolution of 1, 2 or 4% in hardware (ready to use)

• Adjustable timing for each Region/Cluster
  – 1000 to 4000000 clock cycles for full data set
Traffic Monitoring – Basic Concept

- Research scope → More software-defined and cooperative runtime optimization
Traffic Monitoring – Dual NoC Infrastructure

- **Data-NoC** for application data
- **System-NoC** for monitoring
- **Minimal** design and runtime interferences
- **IP-Cores** now has two NIs
  - Data-NoC Interface (DNI)
  - System-NoC Interface (SNI)
- **Smallest management unit** is the **CELL/TILE**
- **Two types of CELLS**
  - Master (CPU)
  - Slave
- **Full reuse!**

**Diagram Notes:**
- Data-NoC
- System-NoC
- Master Cores
- R: Router Node
- DNI: Data-NoC Network Interface
- SNI: System-NoC Network Interface
- NX CELLS
- NY CELLS
Traffic Monitoring – HW/SW-based Clustering

- Rectangular shaped group of CELLS
  - Lower Left Corner (LLC)
  - Upper Right Corner (URC)
  - At least one Master CELL
- No overlapping of Clusters
- One Master CELL per Cluster runs monitoring software
- Each CELL monitors own loads
  - Cluster paths
  - Router links
- Hardware extensions at each CELL needed
- Maximum size is $N_{\text{Clmax}}$ [14, 64]
Traffic Monitoring – Sensing and Flow

Stage 1
- Data-NoC
- Link_BUSY Signals (Data-NoC)

Stage 2
- System-NoC

Stage 3
- IP-CORE
- EAP-CI
- Input Buffer @ DNI
- Output Buffer @ DNI
- Output Buffer @ SNI
- Input Buffer @ SNI
- Event Aggregation Point (Counter Array)

- Depacketization Unit @ DNI
- Packetization Unit @ DNI
- Packetization Unit @ SNI
- Depacketization Unit @ SNI

- DNoC Output Port (CORE)
- DNoC Input Port (CORE)
- SNoC Input Port (CORE)
- SNoC Output Port (CORE)
Traffic Monitoring – Sensing and Flow

• Hierarchical aggregation structure of 3 interacting stages:

  – **Stage 1 – Traffic Sensors**: Activity sensing of links and paths at CELLs
    
    • Counting of activity clock cycles as basic events
    • $N_{\text{Clmax}}$ **Path Sensors** uses ACK-signal of DNI for destinations inside region $\rightarrow$ **REAL LOAD!**
    • 5 **Link Sensors** uses lock signals of arbiter device $\rightarrow$ **REAL LOAD + CONGESTION!**
    • Counting of active clock cycles until overflow happens at counter

  – **Stage 2 – SNI-Extension**: Periodic checking and reporting at CELLs
    
    • Finite state machine periodically checks Traffic Sensors for overflows $\rightarrow$ **Sensor Check Period**
    • Generates and transmits monitoring packet to Master if overflows happened

  – **Stage 3 – Event Aggregation Point**: Master counts reported overflows
    
    • Each Traffic Sensor has a corresponding overflow counter at the Event Aggregation Point (EAP)
    • Event Aggregation Point only implemented at Master CELLs
    • Periodic access of counter values via Core Interface (CI) $\rightarrow$ **Monitoring Cycle**
Traffic Monitoring – Sensing and Flow

• Stage 1 – Traffic Sensors

\[
cnt_j(t_{i+1}) = \begin{cases} 
  cnt_j(t_i) + 1 & (s_{ctrl} = 1) \land cnt_j(t_i) < b_{T-MODE} \\
  cnt_j(t_i) & (s_{ctrl} = 0) \\
  0 & (s_{ctrl} = 1) \land cnt_j(t_i) = b_{T-MODE}
\end{cases}
\]

Activity Timing in Clock Cycles

\[t_{i+1} = t_i + t_{clk}\]

Implementation:
• 12-bit counter
• 12-bit comparator
• ENABLE = \(s_{ctrl}\)
• Six \(b_{T-MODE} [64 – 2048]\)
• OFG = Overflow bit
• \(j < (N_{CLmax} + 5)\) per CELL
Traffic Monitoring – Sensing and Flow

- Stage 2 – SNI-Extension

**FSM: Sensor Check Period**

\[ t_{sp} = b_{T-MODE} \cdot t_{clk} \]

\[ \text{check}(t_{sp}) = \sum_{j=0}^{N_s-1} OFG_j(t_{sp}) > 0 \]

**Implementation:**

- Check via Xoring all OFG
- Read & Reset all OFG
- Packet composition via FSM
- \(b_{T-MODE}\) set during Cluster Setup
- \(b_{T-MODE}\) equal at all Cluster CELLS
Traffic Monitoring – Sensing and Flow

**Stage 3 – Event Aggregation Point**

\[
load_j(t_{sp+1}) = \begin{cases} 
load_j(t_{sp}) + 1 & (OFG_j(t_{sp+1}) = 1) \land (rst = 0) \\
load_j(t_{sp}) & (OFG_j(t_{sp+1}) = 0) \land (rst = 0) \\
0 & rst = 1 
\end{cases}
\]

**Implementation:**
- 7-bit counter GROUPs
- \((N_{CL\text{max}} + 5)\) counter per GROUP
- \(N_{CL\text{max}}\) GROUPs (for each CELL)
- \(N_{CL\text{max}} = [16, 64]\) analysed
- Event Buffer stores incoming packets
- Vector order = Counter order
- Periodic read and reset by software
- Intermediate read by software
- Direct Access via Core Interface (CI)
- Scaling of loads via Access Timing!
Traffic Monitoring – Sensing and Flow

• Load Scaling via Access Timing
  – Stepping $k_s = \{1, 2, 4\} \rightarrow sload = 0:k_s:100$ in % of max BW per Link/Path
  – $N_{SP} =$ # of Sensor Periods $t_{sp} = f(t_{clk}, b_{T-MODE})$ per Monitoring Cycle
  – $T_{MC} =$ Monitoring Cycle

\[
N_{SP} = \frac{100}{k_s} \\
T_{MC} = N_{SP} \cdot t_{sp}
\]

\[
sload_j = \begin{cases} 
load_j & k_s = 1 \Rightarrow N_{SP} = 100 \\
load_j << 1 & k_s = 2 \Rightarrow N_{SP} = 50 \\
load_j << 2 & k_s = 4 \Rightarrow N_{SP} = 25 
\end{cases}
\]

• $\min(b_{T-MODE}) = f(\text{Clustersize})$
  – Traffic Monitoring $\rightarrow$ Many-to-One Pattern
  – Condition: (Injected BW in Cluster $<$ Receivable BW at Master)
    • 16 CELLs $\rightarrow \min(b_{T-MODE}) = 128$
    • 64 CELLs $\rightarrow \min(b_{T-MODE}) = 1024$
  – If condition met $\rightarrow sload \pm e_{\text{max}}$ with $e_{\text{max}} \leq 2 \cdot k_s$!!!
Traffic Monitoring – Sensing and Flow

- **Timing of Traffic Monitoring offers two options:**
  - Adjustment of $b_{T-MODE}$ → At all Cluster CELLs
  - Adjustment of Stepping $k_s$ → At Master CELL only
  - Tradeoff: Effort vs. Accuracy

<table>
<thead>
<tr>
<th>$b_{T-MODE}$</th>
<th>$k_s=1$</th>
<th>$k_s=2$</th>
<th>$k_s=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>6.4</td>
<td>3.2</td>
<td>1.6</td>
</tr>
<tr>
<td><strong>128</strong></td>
<td><strong>12.8</strong></td>
<td><strong>6.4</strong></td>
<td><strong>3.2</strong></td>
</tr>
<tr>
<td>256</td>
<td>25.6</td>
<td>12.8</td>
<td>6.4</td>
</tr>
<tr>
<td>512</td>
<td>51.2</td>
<td>25.6</td>
<td>12.8</td>
</tr>
<tr>
<td><strong>1024</strong></td>
<td><strong>102.4</strong></td>
<td><strong>51.2</strong></td>
<td><strong>25.6</strong></td>
</tr>
<tr>
<td>2048</td>
<td>204.8</td>
<td>102.4</td>
<td>51.2</td>
</tr>
</tbody>
</table>
Traffic Monitoring – Experimental Results

- 45nm hardware synthesis via Synopsys Design Compiler
  - Estimation of hardware overhead
- SystemC-based cycle accurate NoC simulation
  - Measurement of absolute traffic monitoring error (max & avrg)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoC</td>
<td>SNoC</td>
</tr>
<tr>
<td>Linkwidth ( w_L )</td>
<td>8-, 16-bit</td>
</tr>
<tr>
<td>Clock Rate ( t_{clk} )</td>
<td>1ns</td>
</tr>
<tr>
<td>Port Buffer Depth</td>
<td>1 flit</td>
</tr>
<tr>
<td>( b_{T-MODE} )</td>
<td>64 – 2048 (Simulation: 128, 1024)</td>
</tr>
<tr>
<td>Cluster Size ( N_{Clmax} )</td>
<td>16 and 64</td>
</tr>
<tr>
<td>Cluster Shape</td>
<td>4×4, 8×2 and 16×4, 8×8</td>
</tr>
<tr>
<td>Monitor Position</td>
<td>Lower Left Corner (LLC)</td>
</tr>
<tr>
<td>Technology</td>
<td>45nm (Nangate FreePDK45)</td>
</tr>
</tbody>
</table>
Traffic Monitoring – Experimental Results

- Hardware Overhead of the Traffic Monitoring in 45nm
  - \( A_{\text{normal}} \) = Area(\( \text{Router}_{\text{SNoC}}, \text{Links}_{\text{SNoC}}, \text{Traffic Sensors}, \text{SNI-Extension} \))
  - \( A_{\text{master}} \) = Area(\( A_{\text{normal}}, \text{EAP} \))
  - \( A_{\text{tile}} \) = 3mm x 3mm (estimate 45nm Intel SCC)

<table>
<thead>
<tr>
<th>( N_{\text{CLmax}} )</th>
<th>16 CELLS</th>
<th>64 CELLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNoC Linkwidth</td>
<td>8-bit</td>
<td>16-bit</td>
</tr>
<tr>
<td>( A_{\text{master}}/A_{\text{tile}} )</td>
<td>0.51%</td>
<td>0.66%</td>
</tr>
<tr>
<td>( A_{\text{normal}}/A_{\text{tile}} )</td>
<td>0.29%</td>
<td>0.44%</td>
</tr>
</tbody>
</table>

- EAP and Traffic Sensors dominate the hardware overhead
- But area estimates remain inside a feasible range!
Traffic Monitoring – Experimental Results

• Full system simulation with mixed workloads

  – Random Uniform Traffic Pattern
    • Balanced loads and high overall utilization
    • Packetsize = rand(5,15) Flit
    • Destination = rand(0, Nx·Ny - 1)

  – Random Task Graphs with sequential mapping
    • 7 to 70 Tasks per Graph
    • 2 to 10 Graphs per Workload
    • Packetsize = rand(5,50) Flit

  – Each Workload simulated 10 times with 10 full Monitoring Cycles for each Cluster shape (4x4, 8x2, 8x8, 4x16) and stepping (1, 2, 4)
  – Logging of average and maximum errors for pathloads (PL) and linkloads (LL)
Traffic Monitoring – Experimental Results

- Max. error in 4x4 CELL Cluster @ Random: \( \min(b_{T-MODE}) = 128 \)
Traffic Monitoring – Experimental Results

- Max. Error all 16 CELL Cluster Scenarios: \( \min(b_{T-MODE}) = 128 \)
Traffic Monitoring – Experimental Results

- Max. Error all 64 CELL Cluster Scenarios: $\min(b_{T-MODE}) = 1024$

![Graph showing absolute error maximum and maximum average error for different cluster scenarios.](image)
Traffic Monitoring – Experimental Results

- Pseudoload 4x4 CELL Cluster Scenarios: $\min(b_{T\text{-MODE}}) = 128$

![Diagram showing link load and average injected load per core]

- $k_s = 1$

- Pseudoload → Congestion
- Pseudoload → External traffic
Outline

• Introduction
  – Networks-on-Chip (NoC)
  – Why Traffic Monitoring?

• Traffic Monitoring
  – Basic Concept
  – Dual NoC Infrastructure
  – Hardware/Software-based Clustering
  – Sensing and Flow
  – Experimental Results

• Conclusion & Future Work
Conclusion & Future Work

• Flexible HW/SW traffic monitoring proposed
  – All path- and linkloads inside a resizable region at a single entity
  – Adjustable timing and accuracy
  – Hardware overhead and achievable Monitoring Cycles are feasible

• Next steps and investigations → Runtime Mechanisms
  – Workload/Application Profiling (Rent’s Rule)
    • Communication Distributions/Probabilities
    • Execution Phase Detection
    • Combination with Performance Counter Data
  – Flow-based Traffic Management
    • Path adaptations inside Clusters
THANK YOU!