A Distributed Crossbar Switch Scheduler for On-chip Networks

Kangmin Lee, Se-Joong Lee, and Hoi-Jun Yoo

Semiconductor System Lab., Dept. of EECS, Korea Advanced Institute of Science and Technology
Outline

• Motivation
  – Crossbar Switch & Scheduler

• Proposed Distributed Scheduler
  – Description of the proposed algorithm
  – Implementation
  – Comparison w/ Round-Robin Scheduler
    • Area & Latency

• Conclusion
Motivation

• On-chip crossbar switch examples
  – ALPHA 21364 : 1.2GHz 7 x 7 Crossbar Router
  – KAIST BONE : 800MHz 4 x 4 Crossbar Switch

• Crossbar switch becomes a crucial component for on-chip communications in SoC.
Motivation (Cont’d)

- **Costs of a Scheduler**
  - **Area**
    - Up to 50% of total crossbar switch
    - Schedulers do not scale down by serialization
  - **Latency**
    - $t_{CLK} = t_{Schedule} + t_{Fabric}$
  - **Power**
    - < 10% of total switch power

→ High-speed & Area-efficient Scheduler is necessary.
Conventional Scheduler

• Round-Robin Algorithm
  – Uses a centralized rotating priority
    → Programmable Priority Encoder is a key component
    → PPE is implemented with two Simple Priority Encoder.
  – Area & Latency Complexity: $O(N)$, $N$ is a # of input ports
Proposed Scheduler

• A Distributed Algorithm: **NAMOO**
  – Binary-tree connected MACROS (MC)
  – Each MACRO selects one of two requests
  – Uses distributed 1bit priorities: a preference bit (PB)
  – Latency Complexity: $O(\log_2 N)$

![Diagram of 8 ports NAMOO Scheduler]

0: MC prefers a UPPER port
1: MC prefers a LOWER port

CICC 2003
Proposed Scheduler (Cont’d)

• An example of NAMOO’s algorithm

(1) Granting Operation

(2) Updating Operation

• Finally, the granted port (#1) → the lowest probability to be granted.
Implementation

- Granting Path

→ R-S latch holds the preference bit
Implementation (Cont’d)

- Preference Bit Updating

→ MC inverts its PB along the feedback
Advantages

• Area Comparison
- Same topology as Mux-Tree Crossbar Fabric: Binary-TREE
- MCs are distributed over the switch fabric. \(\rightarrow\) Area reduction

55% Reduction of scheduler area
31% Reduction of total switch area

8x8 Crossbar Fabric
RR-scheduler

221x226 um²

8x8 Crossbar Fabric
RR-scheduler

321x226 um²

8x8 crossbar w/ round-robin scheduler
8x8 crossbar w/ NAMOO scheduler

CICC 2003
Advantages (Cont’d)

• Latency Comparison
  – $\log_2(N) \times \text{MC\_delay}$
  – Wiring delay of grant signals is **reduced**.
  – Balanced Fan-out

![Diagram comparing Conventional Mux-Tree based Crossbar with Round-Robin and NAMOO](image)
Advantages (Cont’d)

• Latency Comparison

<table>
<thead>
<tr>
<th>N</th>
<th>R.R.</th>
<th>NAMOO</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0.92ns</td>
<td>0.61ns</td>
<td>34%</td>
</tr>
<tr>
<td>8</td>
<td>1.37ns</td>
<td>0.71ns</td>
<td>48%</td>
</tr>
<tr>
<td>16</td>
<td>1.76ns</td>
<td>0.93ns</td>
<td>47%</td>
</tr>
</tbody>
</table>

N = # of input ports

• HSPICE Simulation @ 0.18um Technology
Application to HDTV

- Bus Architecture → On-Chip Network Architecture

On-Chip Network Architecture
Application to HDTV (Cont’d)

Input B/W per Port

- negligible
- 1.5Gbps
- 0.24Gbps
- 0.18Gbps

Simulation Environments
- C++ high-level models
- NAMOO can speed up its clock frequency

Results
- Waiting time is reduced by its speed-up

Avg. packet waiting time in input queues

- 500MHz Same CLK
- 500MHz (RR)
- 1GHz (NAMOO)

Speed-Up
Conclusion

• A Distributed Scheduler is proposed
  – Structure: Binary Tree-connected MACROs
  – 55% area reduction
  – 48% computing delay reduction
    than conventional round-robin scheduler

• The Proposed scheduler is area-efficient and
  shows higher-performance for the On-chip Networks
Supplementary: Disadvantage

- NAMOO generally doesn’t guarantee Fairness.

<Example>
Request w/ same bandwidth

<table>
<thead>
<tr>
<th>Port #</th>
<th>NAMOO</th>
<th>R.R.</th>
</tr>
</thead>
<tbody>
<tr>
<td># 1</td>
<td>12.5%</td>
<td>25%</td>
</tr>
<tr>
<td># 2</td>
<td>12.5%</td>
<td>25%</td>
</tr>
<tr>
<td># 4</td>
<td>25%</td>
<td>25%</td>
</tr>
<tr>
<td># 6</td>
<td>50%</td>
<td>25%</td>
</tr>
</tbody>
</table>

• What if we know the port bandwidth in advance?
  → By assigning larger bandwidth to #6 or #4,
  Larger bandwidth port gets more grants.
  → Lower packet latency!
Supplementary: Application to HDTV (Cont’d)

- Simulation Results (Cont’d)
  - Packet Waiting Time Distribution

- Less deviation is achieved in NAMOO scheduler
  → More predictable waiting time