A Novel Approach for Network on Chip Emulation

Nicolas Genko, LSI/EPFL Switzerland
David Atienza, DACYA/UCM Spain
Giovanni De Micheli, LSI/EPFL Switzerland
Luca Benini, DEIS/Bologna Italy
José Mendias, DACYA/UCM Spain
Roman Hermida, DACYA/UCM Spain
Francky Catthoor, IMEC Belgium
Outline

• Introduction
• General Approach
• Applications
• Results
• Conclusion
Motivation -- NoCs

• Provide a *structured methodology* for realizing on chip communication schemes
  – Modularity
  – Flexibility
• Overcome the *limitations of busses*
  – Performance and power do not scale up
• Support *reliable operation*
  – Layered approach to error detection and correction
Motivation -- NoC Emulation

• NoCs are designed for:
  – On-chip multiprocessing (regular networks)
  – Specific applications (ad hoc networks)

• Design tools:
  – Synthesis: create NoC circuitry from architectural templates (e.g., Xpipes)
  – Analysis: validate functionality and performance
    • Software simulation (cycle accurate)
    • Emulation with Field Programmable Gate Arrays (FPGAs)
Previous work

• NoC software simulation:
  – High level models in C/C++ [H.-Sheng et al; Kolso et al]
  – Evaluate latency NoCs [Siguenza et al; Angiolini et al]
  – Evaluate throughput NoCs [Wiklund et al; Pestana et al]

• NoC implementation on FPGAs:
  – For functional validation [Marescaux et al; Moraes et al]
  – Show effectiveness NoCs [Kumar et al; Pinto et al]
  – Validate NoCs features [Brebner et al; Zeferino et al]
NoC Emulation on FPGA

- Emulation on FPGA enables functional and performance validation of NoC based systems
  - Accurate execution model
  - Probing for profiling and gathering of statistics
- The emulation can achieve important speedups compared to cycle accurate simulation:
  - Up to four orders of magnitude faster
  - Real inputs with millions of packets can be used
Outline

• Introduction
• General Approach
• Applications
• Results
• Conclusion
General Approach

• A platform which instantiate a NoC on FPGA with modules for emulation:
  – Traffic generators & receptors
  – NoC switches
  – Traffic analyzers
  – Network interfaces (NIs) to cores can be included

• A system which is controlled by a processor
  – The processor configures and controls the traffic pattern to be emulated and analyzes the statistics provided by the platform
Development board
Xilinx XUP

- Power
- Serial interface
- Virtex-II Pro FPGA
  - 2 Power PC Cores
  - 3 M programmable gates

Programming cable
Emulated NoC Architectures

- Processor linked to each system component
  - Monitor
  - Traffic generators
  - Traffic receptors
  - Traffic analyzers
- Two architectures:
  - Network of switches
  - Network with switches and network interfaces
The NoC Architectural Flavour

**Open Core Protocol (OCP)**

**Network Protocol**
- Transmit
  - Access routing tables
  - Assemble packets
  - Split into flits
- Receive
  - Synchronize
  - Drop routing information

```
TAIL PAYLOAD HEADER
FLIT ... FLIT FLIT FLIT
```
Outline

• Introduction
• General Approach
• Applications
• Results
• Conclusion
Architecture 1-- Network of Switches

- **A Processor** (PowerPC): Orchestrates the process and access each component independently
- **A Monitor**: Displays on the PC screen the information extracted
- **The Emulation Platform**:  
  - Traffic generators  
  - Traffic receptors  
  - Network of switches
Emulation of a Network of Switches

• Several types of traffic:
  – Stochastic traffic:
    • Uniform model
    • Burst model (with a two state Markov chain)
  – Trace-driven traffic (real workload)

• Several types of statistics:
  – Measurement of latency of packets
  – Congestion counter (not-acknowledged flits)

• Routing policy evaluation:
  – The routing policy is programmed by software
  – Evaluation of many routing policies without re-synthesis
Architecture 2-- NoCs with interfaces

- **Common components:**
  - Monitor
  - Processor

- **Additional components:**
  - Traffic analyzers
  - NIs to cores

- Slave core receptiveness
  - Modeled by a two-state (on/off) Markov chain

- Traffic analyzers monitor network links activity and interface behavior
Emulation of NoC with interfaces

- Trace-based traffic:
  - Master cores generate traffic according to traces provided by the processor from real applications

- Statistics generated by this platform:
  - Master cores measure average operation execution time
  - Slave cores measure packets latency through the NoC
  - Traffic analyzers measure ACK & NACK activities on links

- Main use of emulation platform:
  - Tuning of a NoC for a specific application
  - Latency analysis for application-specific NoC
Outline

• Introduction
• General Approach
• Applications
• Results
• Conclusion
# FPGA Reports

<table>
<thead>
<tr>
<th>Emulation Architecture</th>
<th>Xilinx Slices</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Emulation of a Network of switches</td>
<td>7387 slices (47%) (6 switches + 4 traffic generators + 4 traffic receptors)</td>
<td>50 MHz</td>
</tr>
<tr>
<td>2. Emulation of a NoC with NIs</td>
<td>7914 slices (51%) (4 switches + 4 master cores + 4 slave cores)</td>
<td>50 MHz</td>
</tr>
</tbody>
</table>
Speed comparison in cycle-accurate NoC environments

<table>
<thead>
<tr>
<th>Simulation mode</th>
<th>Speed (cycles/sec)</th>
<th>Simulation time For 16 Mpackets</th>
<th>Simulation time For 1000 Mpackets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verilog (ModelSim)</td>
<td>3.2K</td>
<td>13h53’</td>
<td>36 days 4h</td>
</tr>
<tr>
<td>SystemC (MPARM)</td>
<td>20K</td>
<td>2h13’</td>
<td>5 days 19h</td>
</tr>
<tr>
<td>Our emulation architectures</td>
<td>50M</td>
<td>3.2 sec</td>
<td>3’20”</td>
</tr>
</tbody>
</table>
Emulation Network of Switches

- Example of statistics:
  - Average latency of packets
- Parameters of the emulation – Burst traffic:
  - Average number of packets/burst
  - Average number of flits/packet
Emulation NoC with NIs

- **Statistics:**
  - Ratio Ack/(Ack+Nack).
  - Average latency of packets on the NoC

- **Emulation parameters:**
  - OCP activity
  - Average number of R/Ws per burst

![Graph showing Avg Latency (Clk) and Ratio Ack/(Ack+Nack) against OCP Load (%)](image)
Outline

• Introduction
• General Approach
• Applications
• Results
• Conclusions
Conclusions

• Mixed HW/SW framework that helps designers to design and validate ad-hoc NoCs

• Two architectures:
  – Emulation of a network of switches.
  – Emulation of a complete NoC with OCP-compliant interfaces

• The FPGA emulation enables to tune NoC parameters with realistic inputs (experiments based on traces from real applications with millions of packets):
  – Topology efficiency
  – Routing policies
  – Latency effects
  – OCP traffic pattern influence
Thank you