OmniXtend: Scalability and LPC

Jaco Hofmann, Tu Dang, Dejan Vucinic
Why do we need memory disaggregation?

- Space in racks is limited
- Memory utilization is low
- Fast shared memory
- Processor only nodes

![Diagram showing memory disaggregation concept](image)
OmniXtend Overview

- OmniXtend is based off TileLink
  - TileLink is an open, coherent bus used to connect Cores with Memory

OmniXtend enhances TileLink and serializes it over Ethernet
An OmniXtend Compute Node

High-level view of each compute node
Compute Node Address Space

High-level view of physical address space

- **Local MMIO (Non-cacheable)**
  - Local MMIO space always maps to a node’s own MMIO devices

- **Global MMIO (Non-cacheable)**
  - MMIO devices of other nodes, will be mapped in Global MMIO
  - Node’s own MMIO devices can also be mapped in Global MMIO space

- **Global RAM (Cacheable)**
  - At reset time, node’s own RAM will be mapped to different part of Global RAM space
  - RAM from other nodes can be mapped in Global RAM space

- **Local RAM (Cacheable but not shared)**
  - Local RAM is only accessible to local node
OmniXtend 1.0.3 to 1.1
OmniXtend 1.0.3 Features

• What does OmniXtend provide right now?
• Cached, Uncached and Coherent Accesses
• Flow Control
• Out-of-sequence/dropped packet detection and handling
OmniXtend 1.0.3 Scalability Concerns

• OmniXtend requires a statically set up system
  – Resend/Flowcontrol mechanisms require state for each communication pair
    • 10s of sessions using SRAM, 100s to 1000s in DRAM with latency penalty

• Permanent connection between all participants is not necessary
OmniXtend 1.1 Dynamic Connections

• Goal: Connection establishment and termination based on existing OX mechanisms

<table>
<thead>
<tr>
<th>VC (3)</th>
<th>Type (4)</th>
<th>Res (3)</th>
<th>Sequence_number (22)</th>
<th>Sequence_number_ack (22)</th>
<th>ACK (1)</th>
<th>Channel (3)</th>
<th>Credit (5)</th>
</tr>
</thead>
</table>

• Three new message types indicated by OX header field
  – Establish Connection -> Starts with Sequence Number 0
  – Terminate Connection -> Indicate end of connection
  – ACK only
Connection Establishment

• Utilizes existing fault tolerance mechanisms
  – Retry until success if communication partner does not answer

• In the best case: Zero additional latency

• No changes to the protocol after the first packet
Connection Termination

• Both parties can initiate connection termination
  – Termination can be delayed if necessary

• Termination can only be approved if there are no outstanding TileLink transactions

• Permissions for cache lines:
  – must be returned in a probe-based cache system
  – can be kept in directory-based cache systems
    • Requires connection reestablishment for permission changes
Canals: Messages bypassing the resend logic

• Messages outside the fault tolerance mechanisms

• First canal message type: ACK only
  – A message that contains only an ACK for a previous message
  – Avoids congestion of the resend buffers

• Avoids a potential deadlock in high throughput, high latency scenarios
  – Both parties have full resend buffers and cannot send another ACK
  – Resend buffers remain full -> Deadlock
OmniXtend Lowest Point of Coherence

• Fully OmniXtend 1.1 compatible LPC for FPGA
• Written in Bluespec (Open-Source Compiler available)
• Designed for 10Gbit/s Ethernet
• Supports a variety of Xilinx FPGAs (using TaPaSCo for bitstream generation)
• Will be released as source and Verilog under Apache 2.0 license at Github
• Includes software implementation of the requester and full system simulation
Demonstration

Software Requester 1

Software Requester 2

Software Requester 3

Ethernet Switch

FPGA LPC
Video