EMiT 2019, Direct Communication Between Distributed FPGA Resources

Joshua Lant, Javier Navaridas

Advanced Processor Technologies Group
School of Computer Science
The University of Manchester

EMiT Emerging Technology Conference 2019, Huddersfield
Overview

1. FPGAs for HPC, the need for direct communication

2. Custom Network Interface

3. Results

4. Concluding Remarks
FPGAs for HPC, the need for direct communication
FPGAs for HPC

• Typically think of GPU as goto accelerator.
• Suitable for dataflow workloads or irregular parallelism.
• FPGAs can provide exceptional performance-per-watt.
• Reduced precision and custom data types.
• Improvements in memory bandwidth are good sign.
• e.g. PCIe bus.
• Network communication through the CPU, or separate FPGA network (point-to-point only).
• New architectures, Xilinx Zynq Ultrascale+, Altera Stratix 10 (IOMMUs).
• Coherent access, tight coupling with CPU.
• Requires CPU for inter-FPGA transfer, reliable access to NIC.
• Allows communication between FPGAs without CPU involvement.
• TCP Offload Engines, non-scalable.
• Simpler solutions typically point-to-point.
• All communication is to globally addressable location, direct to memory.
• Shared-memory access to remote nodes (NUMA).
• Traditional HPC communication (RDMA).
• Same communication method for CPU as FPGA.
Custom Network Interface
Our Solution

• Custom network interface and protocol
  ▶ Addressed using geographic routing scheme.
  ▶ Upper bits are node ID, lower is local memory address.
  ▶ System bus protocol $\rightarrow$ network packet $\rightarrow$ system bus protocol.

• Novel transport layer
  ▶ Completely hardware-offloaded.
    ★ Segregated transport mechanisms (RDMA or Shared Memory).
  ▶ Connectionless (datagram) approach.
    ★ Keeps state information only about outstanding transactions, rather than persistent source-dest connections.
  ▶ End-to-end reliability.
System Design

[Diagram showing system design with components like DRAM, CPU, Accelerator, NIC, NETWORK SWITCH, STREAM TX and RX, MAC/PHY, 4x10G SFP, SHM, RDMA connections.]
## 16B Latency Results

<table>
<thead>
<tr>
<th>latency component</th>
<th>Shared memory (ACK’d)</th>
<th>RDMA (w/ notif.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cycles</td>
<td>ns</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>172</td>
<td>1101</td>
</tr>
<tr>
<td><strong>Initial write- last flit at NIC output</strong></td>
<td>24</td>
<td>154</td>
</tr>
<tr>
<td><strong>Read from RAM</strong></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>TX MAC in- RX MAC out</strong></td>
<td>59</td>
<td>378</td>
</tr>
<tr>
<td><strong>RX MAC out- Resp/Notif at TX MAC in</strong></td>
<td>21</td>
<td>134</td>
</tr>
<tr>
<td><strong>RX MAC out- Completion</strong></td>
<td>9</td>
<td>58</td>
</tr>
</tbody>
</table>
Using Distributed FPGA Resources

- SW based transport vs. HW offload
  - Software transport
    - Copy back to DRAM from Accelerator.
    - More complex control path.
    - CPU controls data movement.
  - Hardware offload
    - Low latency transfers.
    - Simple control path.
    - FPGA writes directly into remote memory.
- Block transfers to accelerator (512B-32KB).
- Implementation on ZCU102 development board.
- Transfers initiated from a user-space program.
SW Transport

Diagram showing the interaction between CPU, SRAM, ACC, DRAM, and NIC with arrows indicating the flow of control and data signals.
HW Offloaded Transport
Results
Latency Results

![Latency Results Graph]

- **Latency (us)**
- **Computation/Communication Ratio**
- **512B HW**
- **1K HW**
- **2K HW**
- **512B SW**
- **1K SW**
- **2K SW**
Data Processing Throughput

Throughput (Mbps) vs. Computation/Communication Ratio

- 32K HW
- 32K SW
- 16K HW
- 16K SW
- 8K HW
- 8K SW

Joshua Lant
EMiT 2019, Direct Communication Between Distributed FPGA Resources
Concluding Remarks
To Conclude…

- Hardware offloaded and connectionless transport is only solution to enable:
  - Direct communications
    - Disaggregating FPGA from CPU resources.
  - Tight memory coupling
    - Lower latency inter-FPGA communications
- Latency improvement of $\approx 29\%$ for small block transfers.
- Throughput improvement of $\approx 9\%$ for large block transfers.
EMiT 2019, Direct Communication Between Distributed FPGA Resources

Joshua Lant, Javier Navaridas

Advanced Processor Technologies Group
School of Computer Science
The University of Manchester

EMiT Emerging Technology Conference 2019, Huddersfield