Enabling Communication with FPGA-Based Network-Attached Accelerators for HPC Workloads

Workshop on Heterogeneous High-Performance Reconfigurable Computing

Steffen Christgau\textsuperscript{1}, Dylan Everingham\textsuperscript{1}, Florian Mikolajczak\textsuperscript{2}, Niklas Schelten\textsuperscript{3}, Bettina Schnor\textsuperscript{2}, Max Schroetter\textsuperscript{2}, Benno Stabernack\textsuperscript{2,3}, Fritjof Steinert\textsuperscript{2,3}

\textsuperscript{1}Zuse Institute Berlin, \textsuperscript{2}University of Potsdam, \textsuperscript{3}Fraunhofer Heinrich-Hertz-Institute
Detach FPGA from host computer, attach directly to network instead → **Network Attached Accelerators** (NAAs) in computing environments → NAAICE project

- Overall goal: enable scalable, flexible and energy-efficient HPC with FPGA-based NAAs
FPGA Framework

- **NAA foundation:** **RDMA-capable FPGA framework**, developed by Fraunhofer HHI
  - Application-independent shell, enabling communication via UDP/IP/Ethernet
  - Supported RDMA protocol: **RoCEv2**
  - Multiple accelerators roles/sockets (reconfigurable)

- Employed hardware: Bittware IA-840F board with Agilex 7 AGF027 FPGA, < 5% usage
Communication with NAAs

• Communication model: **asynchronous remote procedure calls**
  - Make use of FPGA framework’s RDMA capabilities:
    - Connection Management → Reliable Connection
    - RDMA write (no RDMA read support)
  - Use case: long-running offloaded operations → asynchronous by design

• **Challenges:**
  - Management of remotely accessible memory
  - Communication protocol for RPCs
  - Make it usable from application → API design
Solving Communication Challenges

• Memory Region Setup Protocol (MRSP)
  ■ Memory region = remotely accessible memory chunk → exchange of metadata required
  ■ MRs for input and output parameters
  ■ Advertisement of metadata from both sides using InfiniBand Send → rkeys known on both sides
  ■ Symmetric memory regions between host and NAA → allows for exchange of inputs and outputs

• Performing the RPC
  ■ Transfer parameters via RDMA write ("put")
  ■ Start computation with RDMA write with immediate
  ■ Result transfer + completion notification: Write + Immediate.
Performance Evaluation: MRSP

Evaluation of MRSP Overhead

- Testbed: Xeon 4114 host(s)/ConnectX-5 MCMX as NAA client, switched 100 Gbps connection
- Comparison between software and FPGA implementation of NAA (FPGA freq.: 340 MHz)
- Non-negligible initialization overhead due to host operations (NAA–NAA: 4 µs)

![MRSP Scaling for Single MR](image-url)
Performance Evaluation: Latency and Bandwidth

Evaluation of RPC call overhead → transfer of input data to NAA

- Minimal latency: 4.6 $\mu$s (software) vs 2.95 $\mu$s (FPGA) for 1 B data
- Maximum bandwidth 90.87 Gbps → close to theoretical maximum (92.5 Gbps)

**Latency for Single MR**

![Latency Graph](image)

**Bandwidth for Single MR**

![Bandwidth Graph](image)
Application Programming Interface

- Goal: low usage barrier
- middleware on top ibverbs/Linux RDMA stack
- handle-based, asynchronous design

HPC application
(further abstractions)

<table>
<thead>
<tr>
<th>Middleware</th>
<th>Increasing abstraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAA RPC API</td>
<td></td>
</tr>
<tr>
<td>low-level RDMA API</td>
<td></td>
</tr>
<tr>
<td>rdma-core (ibverbs)</td>
<td></td>
</tr>
<tr>
<td>RoCEv2</td>
<td></td>
</tr>
</tbody>
</table>

NAA RPC API pseudo code example

```c
double *a = ..., *b = ..., *c = ...;
naa_param_t params[] =
    {{a, N * sizeof(*a)},
     {b, N * sizeof(*b)},
     {c, N * sizeof(*c)}};

// Instantiate an NAA connection.
naa_handle naa;
naa_create(FNCODE, &params, 3, &naa);

// Invoke the NAA routine.
naa_param_t in_param[] = {params[0], params[1]};
naa_param_t out_param[] = {param[2]};
naa_invoke(&in_params, 2, &out_params, 1, &naa);

int flag = 0;
while (!flag) {
    naa_test(&naa, &flag, ...);
    // Do other work while waiting on the NAA
}
```
Summary

• Project’s goal: enable flexible and scalable usage of network-attached FPGAs in HPC context
• Successfully demonstrated efficient RDMA-based communication with NAA
• Easy-to-use API with potential for further abstractions

Thanks for your attention! Questions?!

This project is sponsored by the German Federal Ministry of Education and Research (grant # 16ME0622K, 16ME0623, 16ME0624).

NAAICE project website: greenhpc.eu