High-Performance Custom Computing with FPGA Cluster as an Off-loading Engine

Takaaki Miyajima, Tomohiro Ueno, Atsushi Koshiba, Jens Huthmann, Kentaro Sano, and Mitsuhisa Sato
Centre for Computational Science (R-CCS), RIKEN, Japan, e-mail:{takaaki.miyajima,Kentaro.sano}@riken.jp

Motivation

Background

- Custom computing with FPGAs is becoming more attractive since..
  - Moore's low is slowing-down.
  - Data-path, memory subsystem, and network are customizable.
  - FPGAs expected to scale the performance for tasks that CPUs cannot do. e.g. stencil computation, FFT, or string matching.
  - Combination of CPU and FPGA will cover wider applications.

Final Goal: System-wide Spatial Custom Computing

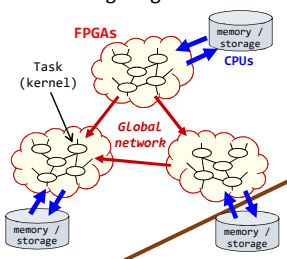
- Configure custom data-paths & custom connections in space
  ≒ HPC system with FPGA cluster as an off-loading engine

Challenges and objectives

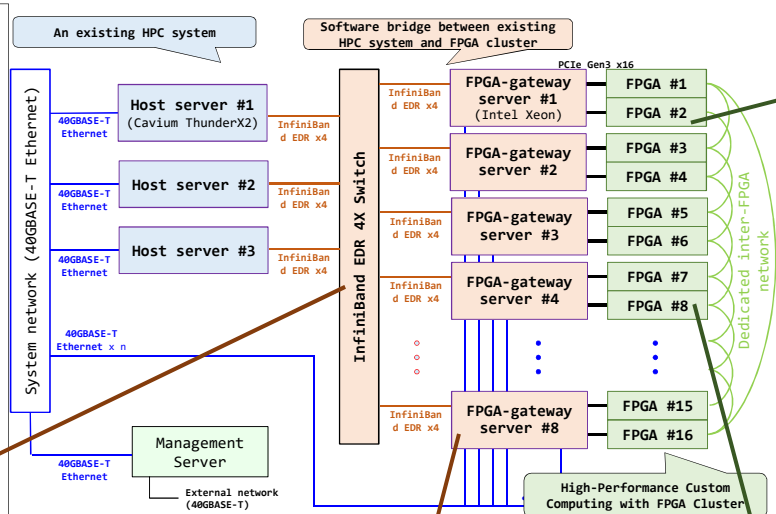
- How can an existing HPC system be extended with FPGA cluster?
- How can we achieve high-performance and scalable custom computing?

On-going work

- Evaluation of programming and task off-loading system for custom computing.
- Performance evaluation of benchmark applications.



System-wide Spatial Custom Computing



Architectural overview of our system

High-performance & scalable custom computing

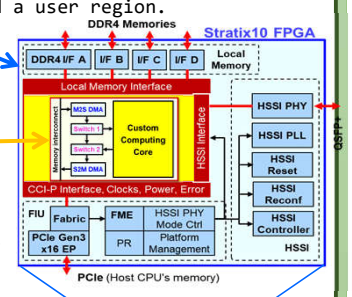
Provides both performance and usability

FPGAs are difficult to use since the users are responsible for everything such as memory subsystem or network. But common functions can be fixed and provided for usability.

"AFU Shell": an overlay for FPGA

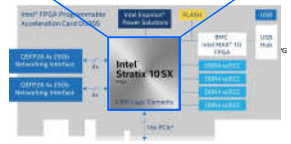
It improves usability while keeping the performance by separating fixed functions and a user region.

- **Outer-side: Fixed function**
  - PCIe, DMA, and network functions are provided.
  - SW API is also provided.
  - Off-the-shelf functions.
- **Inner-side: User region**
  - Custom data-path or memory-subsystem are mapped here.
  - The users can focus on their target computation :)



Intel PAC with Stratix10 (14nm)

- FPGA for data center
- 2753K LEs, 229 Mb BRAMs
- 5760 FP DSPs (8TFlops in SP)
- 8GB DDR4 x 4ch
- PCIe Gen3 x16
- 2x QSFP28 (100Gb/s)



Intel PAC with Stratix10

Extension of existing HPC system with FPGA cluster

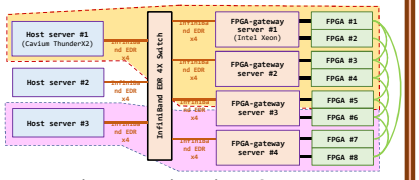
Loosely-coupled Heterogeneous HPC system

FPGA cluster are connected to an existing HPC system through a network as an off-loading engine.

- ✓ Easy extension of existing machines with FPGAs
- ✓ Intermediate network to provide high flexibility
- x Higher latency between CPUs and FPGAs

Such system requires functions that...

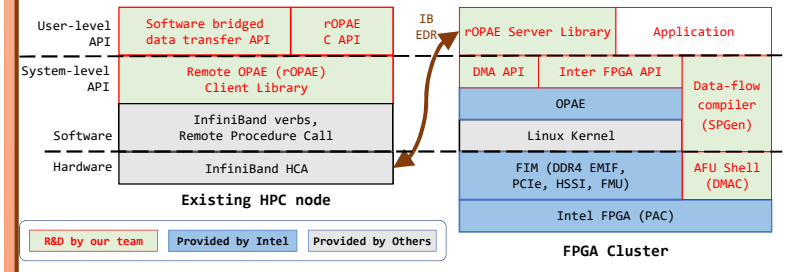
- Enables host servers to access remote FPGAs
- Operates remote FPGAs transparently and efficiently
- Makes existing HPC node can handle arbitrary FPGAs
- Decouples a 1:1 ratio of host servers and FPGAs



Arbitrary assignation of FPGAs

A system stack which meets the requirement

To meet above described requirements, we are researching management APIs, bridging software, dataflow compiler and control API. ***Remote OPAE*** and ***remote data transfer*** are developed as a bridge.



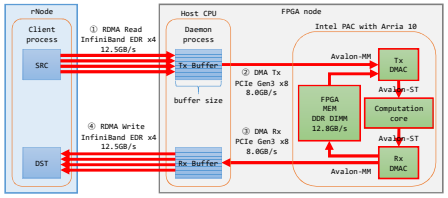
Remote OPAE and remote data transfer

Remote OPAE (rOPAE) enables host servers to operate arbitrary FPGAs remotely. It also provides remote data transfer function among multiple FPGAs from host servers.

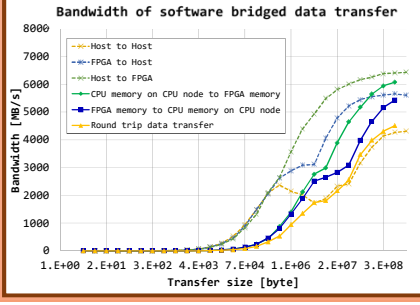
- An extension of Intel Open Programmable Acceleration Engine
- Server and client program
- Single host server can handle arbitrary number of FPGAs
- Stream computation task is off-loaded from host server to FPGAs
- Open source program for any users (to be determined)

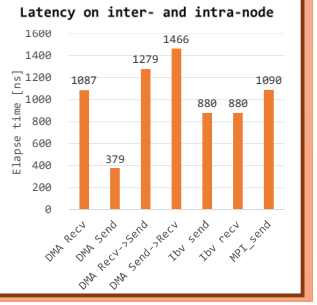
Evaluation of remote data transfer between host server and FPGA

Special remote data transfer function is developed to transfer data among FPGA and host servers through host CPU. **Infini-Band verbs's RDMA protocol** and **zero-copy DMA** are used. Bandwidth and latency are evaluated and shown as below.



Preliminary evaluation environment



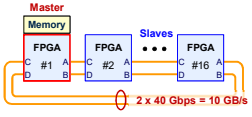


Dedicated inter-FPGA network for smaller latency and higher throughput

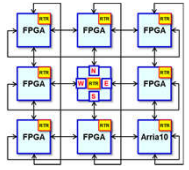
Three networks for multiple FPGAs have their own trade-off among latency, throughput, and usability.

(a) 1D Ring network without router (Done)

- Some stencil apps are evaluated.
- Stream computing w/ multiple FPGAs.
- Deeply pipelining for scalability with sufficiently large data stream.
- x NW Throughput becomes a bottle neck.



(b) 2D torus network with router (On-going)



- Our system consists of a router, a flow controller, a serial transceiver, and a remote DMA controller.
- ✓ Pros) Lower flexibility. It works best if target applications or communication patterns are static and known a priori
- x Cons) Inflexibility, More resource, Difficulty to catch up

(c) Ethernet-based network with switch (On-going)

- Our system will compatible with Ethernet.
- ✓ Pros) Higher flexibility. It works well even if target applications or communication patterns are not known priori.
- x Cons) Big overhead, Higher and variable latency, Difficulty in flow-control