SWIFT

A Transparent and Flexible communication Layer for PCIe-coupled Accelerators and (Co-)Processors

05/19/2014

Simon Pickartz, Pablo Reble, Carsten Clauß, and Stefan Lankes
Today’s HPC Systems

- Homogeneous hardware landscape
- Separate computer systems connected to increase the aggregated compute power
- Different interconnects for LAN and SAN
- RDMA capabilities

→ For portability concerns one standardised interface to layers on top is sufficient (e.g. uDAPL)
Tomorrow’s HPC Systems

- Intra-rack network connecting
  - Hosts
  - I/O devices
  - Storage

- Heterogeneous compute nodes
  - CPUs
  - GPUs
  - Accelerators
  - Etc.

- Peer-to-peer communication
- RDMA and RMA capabilities
- Still different interconnects

→ Computer systems connected to share resources and to increase the aggregated compute power
Agenda

- Socket Wheeled Intelligent Fabric Transport (SWIFT)
- Hardware
- Results
- Outlook
SWIFT – Requirements

- Support for heterogeneous network landscapes
- High portability
- Supply of different programming models
- Consideration of the hardware’s RDMA and RMA capabilities
- High performance
SWIFT – Requirements

- Support for heterogeneous network landscapes
  → A transparent solution is targeted
- High portability

- Supply of different programming models

- Consideration of the hardware’s RDMA and RMA capabilities

- High performance
SWIFT – Requirements

- Support for heterogeneous network landscapes
  - A transparent solution is targeted
- High portability
  - Hardware abstraction
- Supply of different programming models
- Consideration of the hardware’s RDMA and RMA capabilities
- High performance
Support for heterogeneous network landscapes
   → A transparent solution is targeted

High portability
   → Hardware abstraction

Supply of different programming models
   → Service-oriented, SPMD, etc.

Consideration of the hardware’s RDMA and RMA capabilities

High performance
SWIFT – Requirements

- Support for heterogeneous network landscapes
  → A transparent solution is targeted
- High portability
  → Hardware abstraction
- Supply of different programming models
  → Service-oriented, SPMD, etc.
- Consideration of the hardware’s RDMA and RMA capabilities
  → Offer one-sided communication primitives
- High performance
SWIFT – Requirements

- Support for heterogeneous network landscapes
  - A transparent solution is targeted
- High portability
  - Hardware abstraction
- Supply of different programming models
  - Service-oriented, SPMD, etc.
- Consideration of the hardware’s RDMA and RMA capabilities
  - Offer one-sided communication primitives
- High performance
  - Low latencies and high data rates
SWIFT – Basic Concepts

- Topology
  - Hosts
  - Nodes
  - Endpoints

- Communication modes
  - Asynchronous signaling via mails
  - Non-blocking two-sided communication
  - One-sided communication (including atomics)

- Gateway-based fabric connection
Topology

Fabric 0

Fabric 1

Fabric 2

Fabric 3
Topology

SWIFT Network
Layered Architecture

- **Application layer**
  - Higher level libraries (e.g. MPI)
  - Parallel applications
  - Service-oriented apps

- **SWIFT layer**
  - Routing
  - Topology
  - Messaging services

- **Device layer**
  - Hardware abstraction
  - Optimization

→ Well-defined interfaces to layers above and below
SWIFT Device

- Small interface (around 20 prototypes only)
- Administration module
  - Constructor and destructor
  - Automatic discovery of fabric nodes
- Channel module
  - Bi-directional FIFO channel
  - Fixed channel size
  - Asynchronous connection establishment via `create()` and `connect()`
  - Three transfer modes: PIO, DMA, and AUTO

→ High portability
Connection Setup

- A distributed *Directory Service* (DS)
  - Dedicated process for holding topology information
  - Automatically maintains connections to other DS on neighbor hosts
  - Manages node IDs for the local nodes
  - No single point of failure

- On-demand connection setup via DS
  - Direct communication between nodes on different hosts
    - Minimization of the hop count
  - Automatically connect to destination DS if necessary
  - A bit of proactivity
Connection Setup

Host #0

A

DS

4

1

Host #1

B

DS

2

3

05/19/2014 | ACS Automation for Complex Power Systems | 11
Simon Pickartz
Connection Setup

Host #0

DS

A

Host #1

DS

B
Connection Setup

![Diagram showing connection setup between Host #0 and Host #1 with labeled connections 1 to 4.]

- **Host #0**: A to DS, DS to A
- **Host #1**: B to DS, DS to B, DS to B

Connections:
1. A to DS
2. DS to B
3. B to DS
4. DS to A
Connection Setup

Host #0

A

DS

B

DS

Host #1
The ACS Cluster

Host #0

Intel Xeon (E5-2650) #0

Intel Xeon (E5-2650) #1

Intel Xeon Phi #0

Intel Xeon Phi #1

Host #1

Intel Xeon (E5-2650) #0

Intel Xeon (E5-2650) #1

Intel Xeon Phi #0

Intel Xeon Phi #1

Dolphin IXH610 PCIe Fabric
Mapping SWIFT onto the Cluster

SCIF Fabrics

SISCI Fabric
SWIFT Overhead – Throughput

Host-to-Phi (RMA)

Throughput in MiB/s

Size in Byte

Device
SWIFT
MPI
TPP
Multi-Hop PingPong – Latency

Latency in $\mu$s

1-Hop

2-Hop

3-Hop

Real

Phi-Host

Host-Host

0

5

10

15

05/19/2014 | ACS Automation for Complex Power Systems | 17
Simon Pickartz
Multi-Hop PingPong – Throughput

- Latencies accumulate
- Average throughput equals that of the bottleneck link

\[
\begin{align*}
A & \quad t_X \quad B \\
B & \quad t_Y \quad C \\
C & \quad t_Z \quad D
\end{align*}
\]
Multi-Hop PingPong – Throughput

Throughput in MiB/s

Size in Byte

0 200 400 600 800

64 512 4 k 32 k 256 k 2 M 16 M

1-Hop

2-Hop

1-Hop

2-Hop

05/19/2014 | ACS Automation for Complex Power Systems | 19
Simon Pickartz
Multi-Hop PingPong – Throughput

![Graph showing throughput in MiB/s vs size in Byte for 1-Hop, 2-Hop, and 3-Hop scenarios. The x-axis represents size in Byte: 64, 512, 4k, 32k, 256k, 2M, 16M. The y-axis represents throughput in MiB/s.]

- **1-Hop**
- **2-Hop**
- **3-Hop**
Multi-Hop PingPong – Throughput

- Latencies accumulate
- Average throughput equals that of the bottleneck link

![Diagram]

A → B → C → D

\( t_X \) \( t_Y \) \( t_Z \)
Multi-Hop PingPong – Throughput

- Latencies accumulate
- Average throughput equals that of the bottleneck link

\[ t_{BA} \quad t_{CB} \quad t_{DC} \]

\[ t_{AB} \quad t_{BC} \quad t_{CD} \]

→ Asymmetric links
→ Average throughput corresponds to the harmonic mean of the two bottleneck links
What we have . . .

- Support for heterogeneous network landscapes
- High portability

- Supply of different programming models

- Consideration of the hardware’s RDMA and RMA capabilities

- High performance
What we have . . .

- Support for heterogeneous network landscapes
- High portability
  - Device abstraction
  - Three devices: SCIF, SISCI, and SHMEM
- Supply of different programming models

- Consideration of the hardware’s RDMA and RMA capabilities

- High performance
What we have . . .

- Support for heterogeneous network landscapes
- High portability
  - Device abstraction
  - Three devices: SCIF, SISCI, and SHMEM
- Supply of different programming models
  - Three-layered topology
  - Automatic node ID assignment
- Consideration of the hardware’s RDMA and RMA capabilities

- High performance
What we have . . .

- Support for heterogeneous network landscapes
- High portability
  - Device abstraction
  - Three devices: SCIF, SISCI, and SHMEM
- Supply of different programming models
  - Three-layered topology
  - Automatic node ID assignment
- Consideration of the hardware’s RDMA and RMA capabilities
  - One-sided communication
  - Zero-copy forwarding
- High performance
What we have...

- Support for heterogeneous network landscapes
- High portability
  - Device abstraction
  - Three devices: SCIF, SISCI, and SHMEM
- Supply of different programming models
  - Three-layered topology
  - Automatic node ID assignment
- Consideration of the hardware’s RDMA and RMA capabilities
  - One-sided communication
  - Zero-copy forwarding
- High performance
  - Good latency results (asynchronous signaling)
  - Promising multi-hop throughput results
What we need...

- DMA over the whole platform
  - Implementation of `swift_put()`/`_get()` (w.i.p.)
- Multicast or PUB/SUB communication mode
- Dynamic routing (e.g. Bellman-Ford)
Thank you for your kind attention!

**Simon Pickartz** – spickartz@eonerc.rwth-aachen.de

Institute for Automation of Complex Power Systems  
E.ON Energy Research Center, RWTH Aachen University  
Mathieustraße 10  
52074 Aachen

[www.eonerc.rwth-aachen.de](http://www.eonerc.rwth-aachen.de)
Asymmetric Channels

- time to send a message of length $L$ from A to B and back

$$T = \frac{L}{t_{AB}} + \frac{L}{t_{BA}}$$

- resulting throughput

$$t_{res} = \frac{L}{T} = \frac{2L}{t_{AB} + t_{BA}} = 2 \cdot \frac{t_{AB} \cdot t_{BA}}{t_{AB} + t_{BA}}$$
RDMA Results

Phi-to-Phi

Throughput in MiB/s

Size in Byte

Device
SWIFT (host-routed)
MPI (host-routed)