Outline

• Motivation
• PCIe Overview
• Non-Transparent Bridges
• Device Lending
Distributed applications may need to access and use IO resources that are physically located inside remote hosts.
Software abstractions simplify the use and allocation of resources in a cluster and facilitate development of distributed applications.

- **Control + Signaling + Data**

- **rCUDA**
- **CUDA-aware Open MPI**
- **Custom GPUDirect RDMA implementation**
- ...
Local resource

Application

CUDA library + driver

Remote resource using **middleware**

Application

CUDA – middleware integration

Middleware service

Interconnect transport (RDMA)

Interconnect

Middleware service/daemon

CUDA driver

PCle IO bus

PCle IO bus
In PCIe clusters, the same fabric is used both as local IO bus within a single node and as the interconnect between separate nodes.
PCle Overview
PCIe is the dominant IO bus technology in computers today, and can also be used as a high-bandwidth low-latency interconnect.
Memory reads and writes are handled by PCIe as transactions that are packet-switched through the fabric depending on the address.

- Upstream
- Downstream
- Peer-to-peer (shortest path)
IO devices and the CPU share the same physical address space, allowing devices to access system memory and other devices.

- Memory-mapped IO (MMIO / PIO)
- Direct Memory Access (DMA)
- Message-Signaled Interrupts (MSI-X)
Non-Transparent Bridges
Remote address space can be mapped into local address space by using PCIe Non-Transparent Bridges (NTBs).

<table>
<thead>
<tr>
<th>Local</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xf000</td>
<td>0x9000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Using NTBs, each node in the cluster take part in a shared address space and have their own “window” into the global address space.
Device Lending
A remote IO device can be “borrowed” by mapping it into local address space, making it appear locally installed in the system.
By intercepting DMA API calls to set up IOMMU mappings and inject reverse NTB mappings, physical location is completely transparent.

```
dma_addr = dma_map_page(0x9000);
```

<table>
<thead>
<tr>
<th>Local</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xf000</td>
<td>0x5000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Use addr 0xf000
Borrowed remote resource

Resource appears local to OS, driver, and app

Unmodified local driver (with hot-plug support)

Hardware mappings ensure fast data path

Works with any PCIe device (even individual SR-IOV functions)
**Borrowed** remote resource

Local

- Application
  - CUDA library + driver
  - PCIe IO bus
  - PCIe NTB interconnect

Remote

- Application
  - CUDA driver
  - Middleware service/daemon
  - Interconnect transport (RDMA)
  - Middleware service/daemon
  - Interconnect transport (RDMA)
  - PCIe IO bus

Remote resource using middleware

- Application
  - CUDA – middleware integration
  - Middleware service
  - Interconnect transport (RDMA)
  - PCIe IO bus
  - PCIe NTB interconnect
Local resource

Borrowed remote resource

Application

CUDA library + driver

PCIe IO bus

PCIe NTB interconnect

Remote

Local resource

Application

CUDA library + driver

PCIe IO bus
Device-to-host memory transfer

- **Device-to-host memory transfer graph**
  - X-axis: Transfer size
  - Y-axis: Gigabytes per second (GB/s)
  - Lines:
    - Green: `bandwidthTest (Local)`
    - Orange: `bandwidthTest (Borrowed)`
    - Gray: PXH830 DMA (GPUDirect RDMA)

**System specifications**
- **GPU**: Quadro P400
- **Nvidia driver**: Version 375.26 (Centos 7)
- **CPU**: Xeon E5-1630 3.7 GHz
- **Memory**: DDR4 2133 MHz

**Notes**
- 1. Nvidia CUDA 8.0 Samples `bandwidthTest`
- 2. GPUDirect RDMA benchmark using Dolphin NTB DMA

Using Device Lending, nodes in a PCIe cluster can share resources through a process of borrowing and giving back devices.
EIR – Efficient computer aided diagnosis framework for gastrointestinal examination

http://mlab.no/blog/2016/12/eir/
Moving forward

• Strategy-based management
• Fail-over mechanisms
• VFIO and other API integration (“SmartIO”)
• Borrowing vGPU functions
Thank you!

Selected publications

“Device Lending in PCI Express Networks”
ACM NOSSDAV 2016

“Efficient Processing of Video in a Multi Auditory Environment using Device Lending of GPUs”
ACM Multimedia Systems 2016 (MMSys’16)

“PCIe Device Lending”
University of Oslo 2015

Device Lending demo and more
Visit Dolphin in exhibition area (booth 625)

jonassm@simula.no