OFA Training Program

Writing Application Programs for RDMA using OFA Software

Author: Rupert Dance
Date: 11/15/2011

Agenda – OFA Training Program

- Program Goals
- Instructors
- Programming course format
- Course requirements & syllabus
- UNH-IOL facilities & cluster equipment
- RDMA Benefits
- Programming course examples
- Future courses
- Course availability

OFA Training Program - Overall Goals

- Provide application developers with classroom instruction and hands on experience writing, compiling and executing an application using OFED verbs API

- Illustrate how RDMA programming is different from sockets programming and provide the rationale for using RDMA.

- Focus on the OFED API, RDMA concepts and common design patterns

- Opportunity to develop applications on the OFA cluster at the University of New Hampshire – includes the latest hardware from Chelsio, DDN, Intel, Mellanox, NetApp & QLogic

Instructors

- **Dr. Robert D. Russell**: Professor in the CS Department at UNH
  - Dr. Russell has been an esteemed member of the University of New Hampshire faculty for more than 30 years and has worked with the InterOperability Laboratory's iSCSI consortium, iWARP consortium and the OpenFabrics Interoperability Logo Program.

- **Paul Grun:** Chief Scientist for System Fabric Works
  - Paul has worked for more than 30 years on server I/O architecture and design, ranging from large scale disk storage subsystems to high performance networks. He served as chair of the IBTA's Technical Working Group, contributed to many IBTA specifications and chaired the working group responsible for creating the RoCE specification.

- **Rupert Dance:** Co-Chair of the OFA Interoperability Working Group
  - Rupert helped to form and has led both the IBTA Compliance and Interoperability and OFA Interoperability programs since their inception. His company, Software Forge, worked with the OFA to create and provide the OFA Training Program.

Programming Course Format

- Part One - Introduction to RDMA
  - I/O Architecture and RDMA Architecture
  - Address translation and network operations
  - Verbs Introduction and the OFED Stack
  - Introduction to wire protocols

- Part Two - Programming with RDMA
  - Hardware resources: HCAs, RNICs, etc
  - Protection Domains and Memory Registration keys
  - Connection Management
  - Explicit Queue Manipulation
  - Explicit Event Handling
  - Explicit Asynchronous Operation
  - Explicit Manipulation of System Data Structures

Programming Course Requirements

- Requirements
  - Knowledge of "C" programming including concepts such as structures, memory management, pointers, threads and asynchronous programming
  - Knowledge of Linux since this course does not include Windows programming

- Helpful
  - Knowledge of Event Handlers
  - Knowledge of sockets or network programming
  - Familiarity with storage protocols

Programming Course Syllabus

- Introduction to OFA architecture
  - Verbs and the verbs API
  - A Network perspective
  - RDMA Operations – SEND/RECEIVE, RDMA READ & WRITE
  - RDMA Services
  - Isolation and Protection Mechanisms
  - A brief overview of InfiniBand Management
  - A quick introduction to the OFED stack
  - Host perspective
    - Asynchronous processing
    - Channel vs. RDMA semantics

- Basic Data Structures
  - Connection Manager IDs
  - Connection Manager Events
  - Queue Pairs
  - Completion Queues
  - Completion Channels
  - Protection Domains
  - Memory Registration Keys
  - Work Requests
  - Work Completions

- Connection management basics
  - Establishing connections using RDMACM
  - RDMACM API

- Basic RDMA programming
  - Memory registration
  - Object creation
  - Posting requests
  - Polling
  - Waiting for completions using events
  - Common practices for implementing blocking wait

- Design patterns
  - Send-receive
  - RDMA cyclic buffers
  - Rendezvous

- Advanced topics
  - Work Request chaining
  - Multicast
  - Unsignaled Completions

- RDMA ecosystems
  - Native InfiniBand
  - iWARP
  - RoCE

UNH Interoperability Lab

Linux

OFA Cluster
at UNH-IOL

Windows

Thanks to AMD, Intel and OFA for the addition of 16 new nodes in October 2011
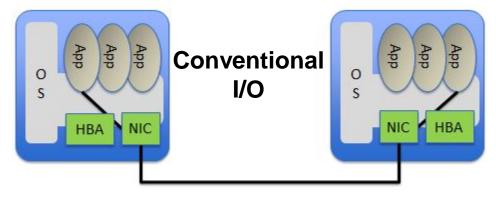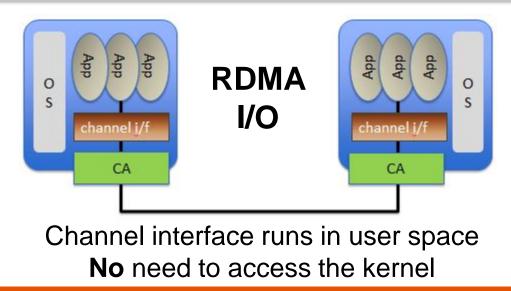
OFA Software Benefits

- Remote Direct Memory Access provides
  - Low latency – stack bypass and copy avoidance
  - Kernel bypass – reduces CPU utilization
  - Reduces memory bandwidth bottlenecks
  - High bandwidth utilization

- Cross Platform support
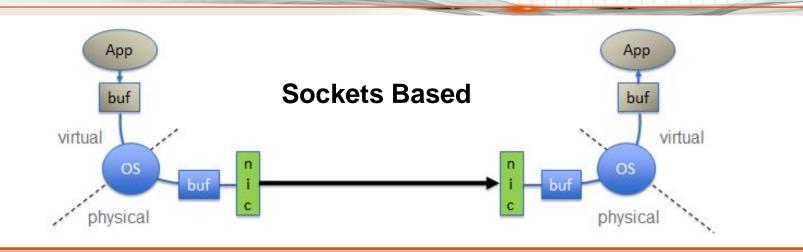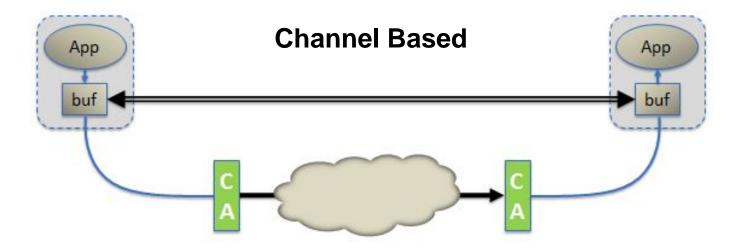 - InfiniBand
  - iWARP
  - RoCE

Conventional I/O versus RDMA I/O

OS involved in all operations

Conventional I/O

RDMA I/O

Channel interface runs in user space
No need to access the kernel

Address Translation

Sockets Based

Channel Based

Many apps, one interface, three wires

OFED – the whole picture

OPENFABRICS

SA	Subnet Administrator
MAD	Management Datagram
SMA	Subnet Manager Agent
PMA	Performance Manager Agent
IPoIB	IP over InfiniBand
SDP	Sockets Direct Protocol
SRP	SCSI RDMA Protocol (Initiator)
iSER	iSCSI RDMA Protocol (Initiator)
RDS	Reliable Datagram Service
VNIC	Virtual NIC
UDAPL	User Direct Access Programming Lib
HCA	Host Channel Adapter
R-NIC	RDMA NIC

Application Level

Diag Tools | Open SM | IP Based App Access | Sockets Based Access | Various MPIs | Block Storage Access | Clustered DB Access | Access to File Systems

User APIs

User Level MAD API

UDAPL

OpenFabrics User Level Verbs & CMA / API

User Space

SDP Lib

Upper Layer Protocol

Kernel Space

VNIC | IPoIB | SDP | SRP | iSER | RDS | NFS-RDMA RPC | Cluster File Sys

Mid-Layer

Kernel bypass

Connection Manager Abstraction (CMA)

SA Client | MAD | SMA | Connection Manager | Connection Manager

Kernel bypass

OpenFabrics Kernel Level Verbs / API

Provider

Hardware Specific Driver | Hardware Specific Driver

Hardware

InfiniBand HCA | iWARP R-NIC

Key

Common | Apps & Access Methods for using OF Stack
InfiniBand
iWARP

Programming Course Sample

- Description of the verbs
- Description of the data structures
- Preparation for posting a send operation
- Create the work request
- Gathering data from memory
- Putting gathered elements on the wire
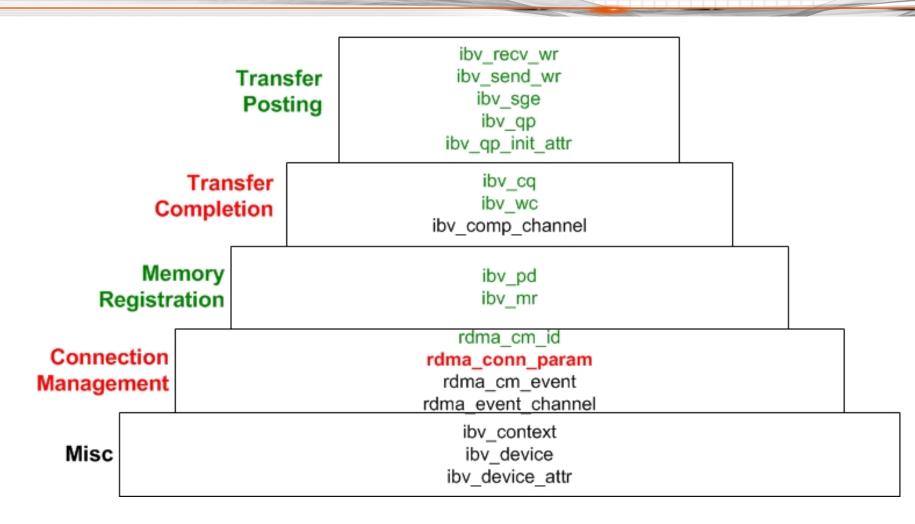- Multicast concept
- The Big Picture

Programming Course - OFED Verbs

	Setup	Use	Break-Down
Transfer Posting	rdma_create_qp	ibv_post_recv ibv_post_send	rdma_destroy_qp
Transfer Completion	ibv_create_cq ibv_create_comp_channel	ibv_poll_cq ibv_wc_status_str ibv_req_notify_cq ibv_get_cq_event ibv_ack_cq_events	ibv_destroy_cp ibv_destroy_comp_channel
Memory Registration	ibv_alloc_pd ibv_reg_mr		ibv_dealloc_pd ibv_dereg_mr
Connection Management	rdma_create_id rdma_create_event_channel	rdma_resolve_addr rdma_resolve_route **rdma_connect** **rdma_disconnect** rdma_bind_addr rdma_listen rdma_get_cm_event rdma_ack_cm_event rdma_event_str rdma_accept rdma_reject rdma_migrate_id rdma_get_local_addr rdma_get_peer_addr	rdma_destroy_id rdma_destroy_event_channel
Misc		rdma_get_devices rdma_free_devices ibv_query_devices	

Programming Course – Data Structures

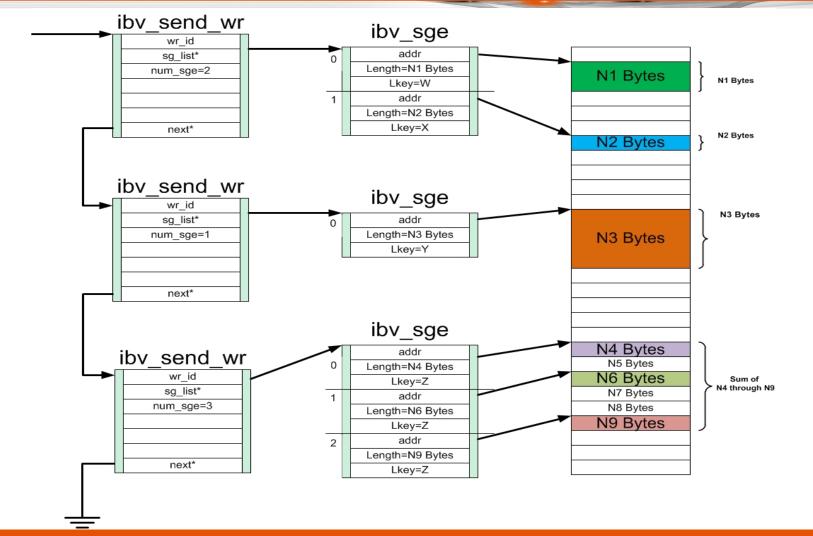**Transfer Posting**
- ibv_recv_wr
- ibv_send_wr
- ibv_sge
- ibv_qp
- ibv_qp_init_attr

Transfer Completion
- ibv_cq
- ibv_wc
- ibv_comp_channel

Memory Registration
- ibv_pd
- ibv_mr

Connection Management
- rdma_cm_id
- **rdma_conn_param**
- rdma_cm_event
- rdma_event_channel

Misc
- ibv_context
- ibv_device
- ibv_device_attr

Bottom-up client setup phase

- **rdma_create_id()** - create **struct rdma_cm_id** – identifier
- **rdma_resolve_addr()** - bind **struct rdma_cm_id** to local device
- **rdma_resolve_route()** - resolve route to remote server
- **ibv_alloc_pd()** - create **struct ibv_pd** – protection domain
- **ibv_create_cq()** - create **struct ibv_cq** – completion queue
- **rdma_create_qp()** - create **struct ibv_qp** – queue pair
- **ibv_reg_mr()** - create **struct ibv_mr** – memory region
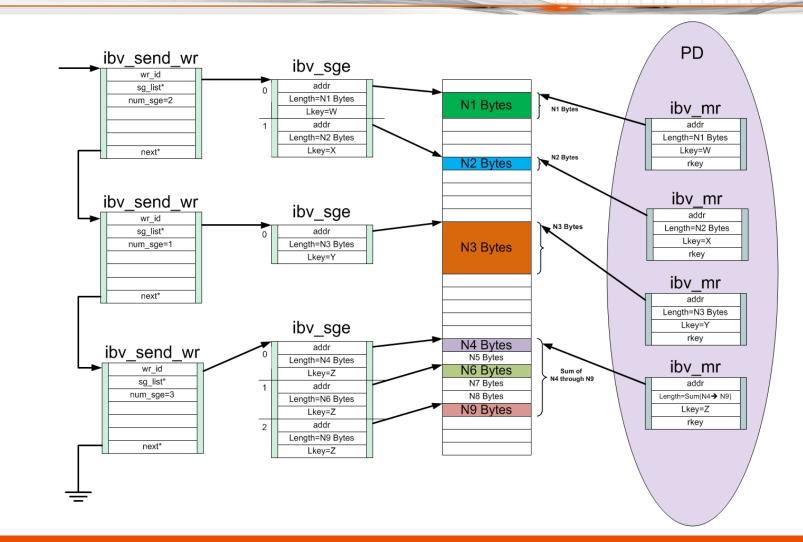- **rdma_connect()** - create connection to remote server

Creating Scatter Gather Elements
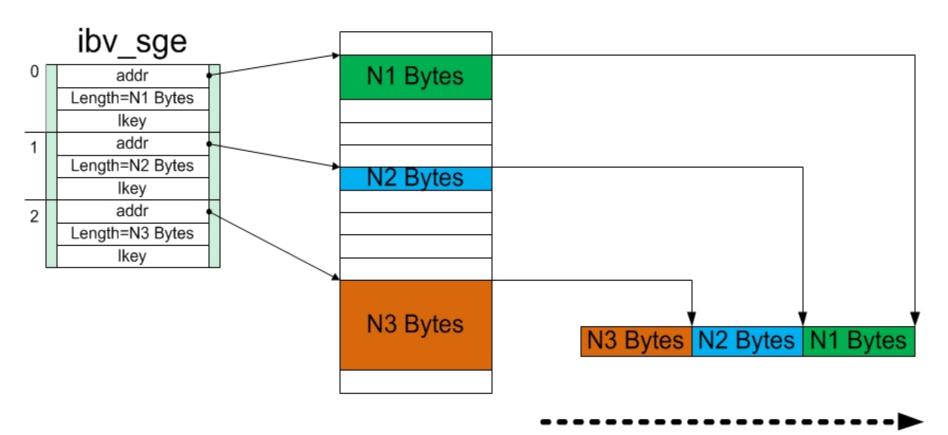
Protection Domains – Memory Regions

Gather during ibv_post_send()

Gather out of memory onto the Wire

Send Work Request (SWR)

- Purpose: tell network adaptor what data to send
- Data structure: **struct ibv_send_wr**
- Fields visible to programmer:

next	pointer to next SWR in linked list
wr_id	user-defined identification of this SWR
sg_list	array of scatter-gather elements (SGE)
opcode	**IBV_WR_SEND**
num_sge	number of elements in **sg_list** array
send_flags	**IBV_SEND_SIGNALED**

- Programmer must fill in these fields before calling **ibv_post_send()**

Posting to send data

- Verb: **ibv_post_send()**
- Parameters:
  - Queue Pair - QP
  - Pointer to linked list of Send Work Requests – SWR
  - Pointer to bad SWR in list in case of error
- Return value:

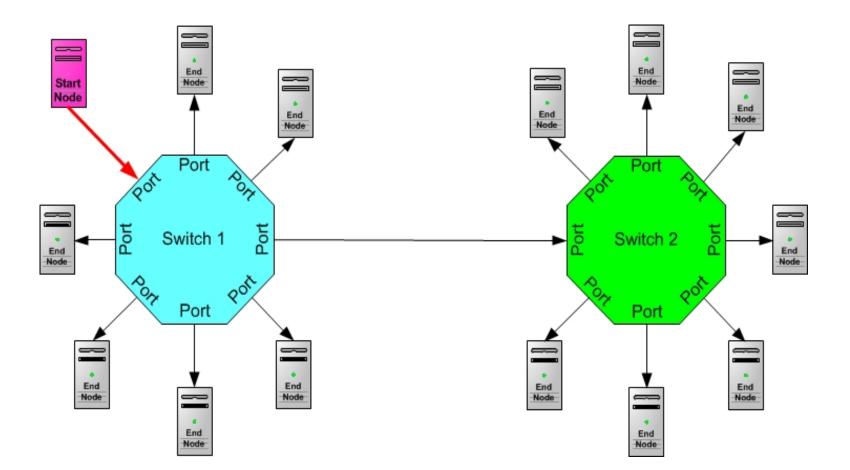 == 0 all SWRs successfully added to send queue (SQ)

  != 0 error code

Bottom-up client break-down phase

- **rdma_disconnect()** - destroy connection to remote server
- **ibv_dereg_mr()** - destroy **struct ibv_mr** – memory region
- **rdma_destroy_qp()** - destroy **struct ibv_qp** – queue pair
- **ibv_destroy_cp()** - destroy **struct ibv_cq** – completion queue
- **ibv_dealloc_pd()** - deallocate **struct ibv_pd** – protection domain
- **rdma_destroy_id()** - destroy **struct rdma_cm_id** – identifier
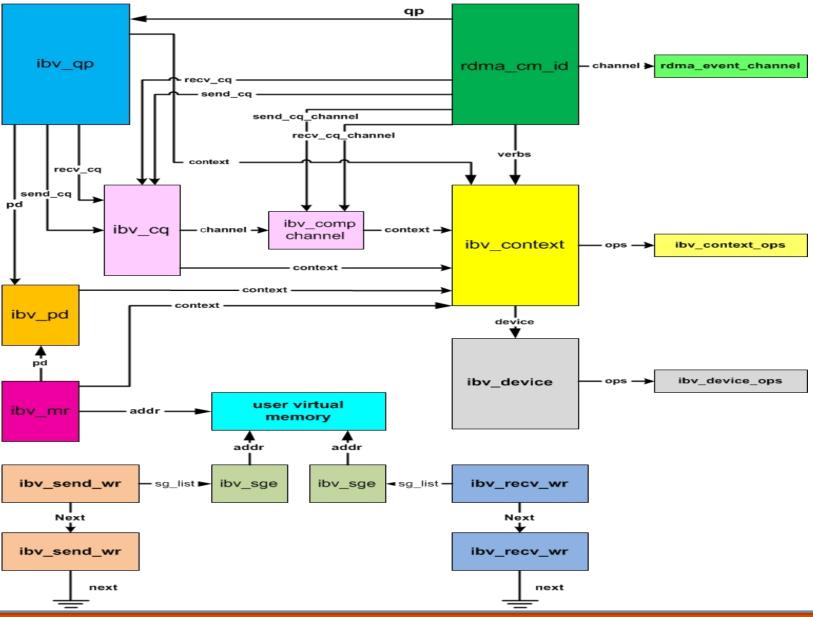
Multicast concept

Multicast

- Optional to implement in IB CAs and switches

- Uses **Unreliable Datagram (UD)** mode
  - Only **Send/Recv** operations allowed
  - Both sides must actively participate in data transfers

- Receiver must have RECV posted for next SEND

- Receiver must process each RECV completion

- Only possible with IB, not iWARP

Programming Course – The Big Picture

Future OFA Software Training Course

- System Administration
  - System configuration
  - Cluster optimization
- Advanced Programming topics
  - Kernel level programming
- ULP Training
  - MPI
  - RDS
  - SRP

OFA Programming Course Availability

- Available quarterly at the University of New Hampshire Interoperability Lab (UNH-IOL)
  - January 18-19, 2012 – OFA Programming Course
    - January 20th, 2012 – Free Ski Trip to Loon Mountain
  - March 13-14 2012 – OFA Programming Course
  - Registration: https://www.openfabrics.org/resources/training/training-offerings.html

- The course can be presented at your company location
  - Additional fees apply for travel and equipment required to support the training materials and exercises.
  - Minimum of 8 attendees required
  - Europe and Asia supported in addition to USA

- New for 2012, the course will be made available via Webinar
  - This is a 4 day course made available for developers in Asia and other countries requiring extensive travelling
  - Minimum of 8 attendees required

- For more information contact: rsdance@soft-forge.com