Writing RDMA applications on Linux

Roland Dreier <rolannd@cisco.com>
RDMA?
RDMA:

Remote DMA
RDMA:
Remote Direct Memory Access
RDMA:
Remote Direct Memory Access
one-sided operations
RDMA: Remote Direct Memory Access
one-sided operations
get/put semantics
RDMA:
Remote Direct Memory Access
one-sided operations
get/put semantics
direct data placement
RDMA:
Remote Direct Memory Access
...but wait, there's more...
RDMA:
Remote Direct Memory Access
Asynch work/completion queues
RDMA: Remote Direct Memory Access
Asynch work/completion queues
Kernel bypass
RDMA:

InfiniBand
RDMA:
InfiniBand
iWARP
RDMA Verbs

Subtitle
Verbs?
Verbs:

not quite an API
Verbs:

not quite an API;

“abstract definition of functionality”
Verbs:

resources (objects)

operated on by

verbs (functions)
Verbs: create object
Verbs:

create object

destroy object
Verbs:
create object
destroy object
more interesting things...
Objects:
Objects:

device context
Objects:

queue pair (QP)
Objects:

queue pair (QP)

send queue & receive queue
Objects:

queue pair (QP)

post send
post receive
modify state
Objects:

completion queue (CQ)
Objects:

completion queue (CQ)

work request completions reported as CQ entries
Objects:

completion queue (CQ)

poll CQ

request notification
Objects:

completion channel
Objects:

memory region (MR)
Objects:

protection domain (PD)
Objects:

shared receive queue (SRQ)
Objects:

shared receive queue (SRQ)

multiple QPs can share a receive queue

cleaning up is a little tricky
Objects:

shared receive queue (SRQ)

post receive
Objects:

address handle (AH)
memory window (MW)
Work processing:
requests from WQs get executed
completions are reported to CQs
mostly things stay in order
Linux & RDMA
librdmacm
librdmacm

Linux library to abstract connection setup
librdmacm

Linux library to abstract connection setup

same code runs on IB and iWARP
librdmacm

mimics TCP socket model
librdmacm

mimics TCP socket model

“cm_id” is socket analog
librdmacm

mimics TCP socket model

“cm_id” is socket analog

IP addressing used even on InfiniBand
librdmacm

mimics TCP socket model

“cm_id” is socket analog

IP addressing used even on InfiniBand

additional address/route resolution steps
librdmacm

events reported through “channels”
librdmacm

events reported through "channels"

rdma_create_event_channel()

rdma_get_cm_event()

rdma_ack_cm_event()
librdmacm

active connection steps
librdmacm

active connection steps

rdma_resolve_addr()

rdma_resolve_route()

rdma_connect()
librdmacm

passive connection steps
librdmacm

passive connection steps

rdma_bind_addr()

rdma_listen()

rdma_accept()
libibverbs
libibverbs

Linux implementation of

RDMA verbs
libibverbs

Loads device-specific drivers for hardware support
libibverbs

Loads device-specific drivers for hardware support

IB: libmthca, libmlx4, libipathverbs, libehca

iWARP: libcxgb3, libamso
libibverbs

Loads device-specific drivers for hardware support

IB: libmthca, libmlx4, libipathverbs, libehca

iWARP: libcxgb3, libamso
libibverbs

Creating QP can be confusing

rdma_create_qp() vs. ibv_create_qp()

all those parameters!
libibverbs

Posting work requests is tricky too

send opcodes

iWARP doesn't have immed data or atomics

signaled and unsignaled completions
Q and A