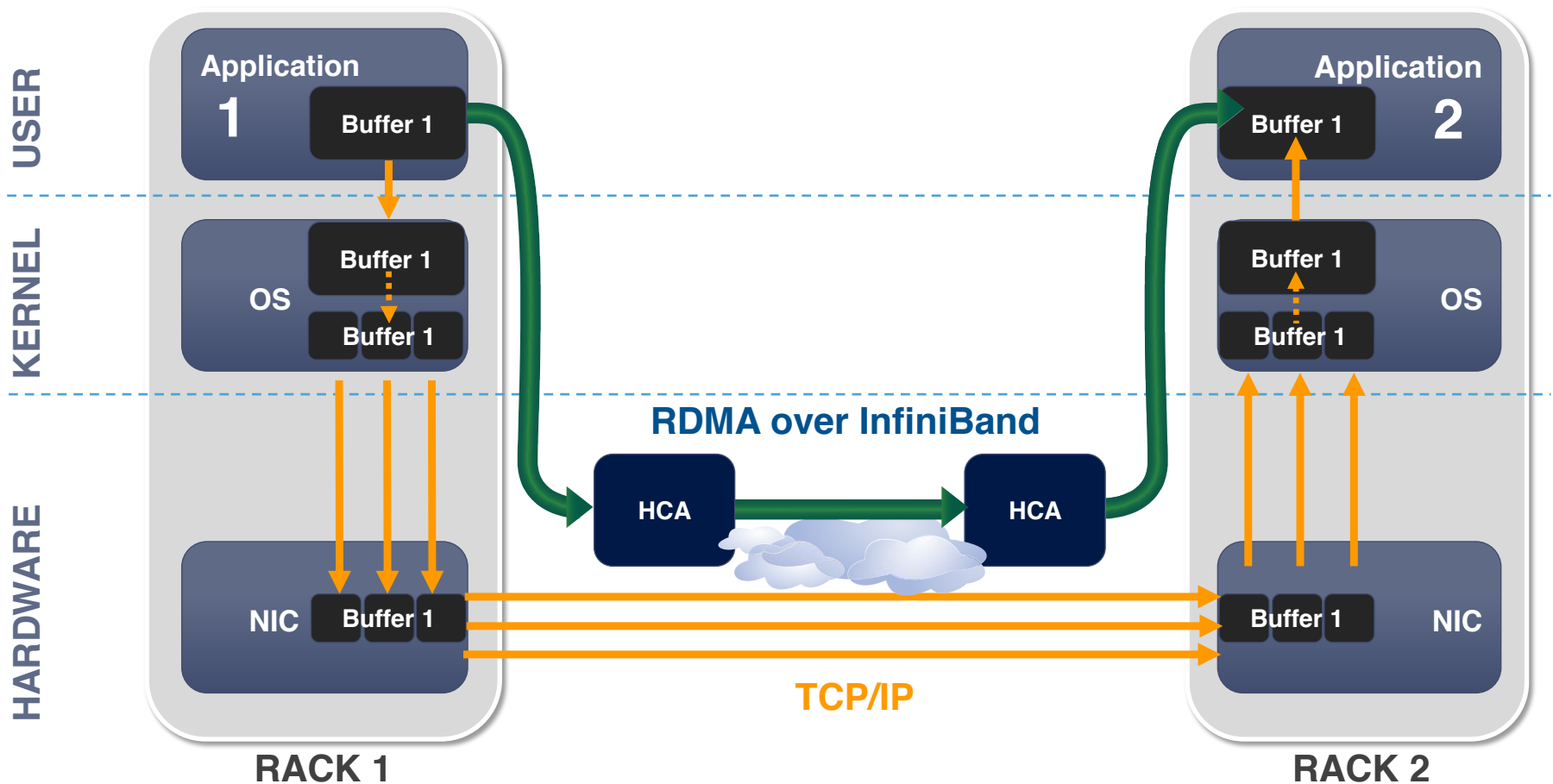


# HPC networks: Infiniband

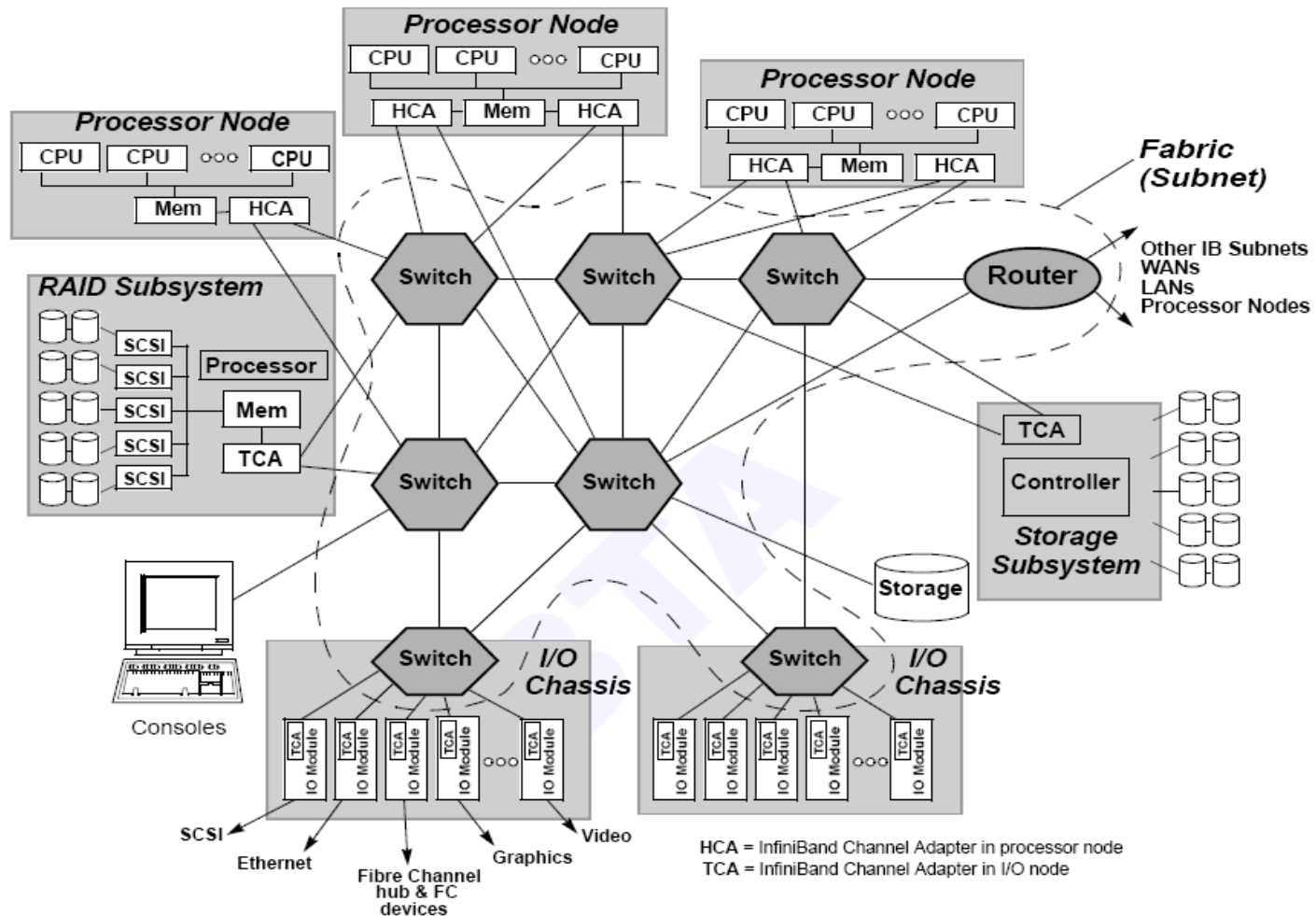
# IBA

- The InfiniBand Architecture (IBA) is an industry-standard architecture for **server I/O** and **inter-server communication**.
  - Developed by InfiniBand Trade Association (IBTA).
- It defines a **switch-based, point-to-point** interconnection network that enables
  - **High-speed**
  - **Low-latency**communication between connected devices.

# Infiniband used RDMA based Communication



- Infiniband architecture overview



# Architecture Layers

- **Software Transport Verbs and Upper Layer Protocols:**

- Interface between application programs and hardware.
- Allows support of legacy protocols such as TCP/IP
- Defines methodology for management functions

- **Transport:**

- Delivers packets to the appropriate Queue Pair; Message Assembly/De-assembly, access rights, etc.

- **Network:**

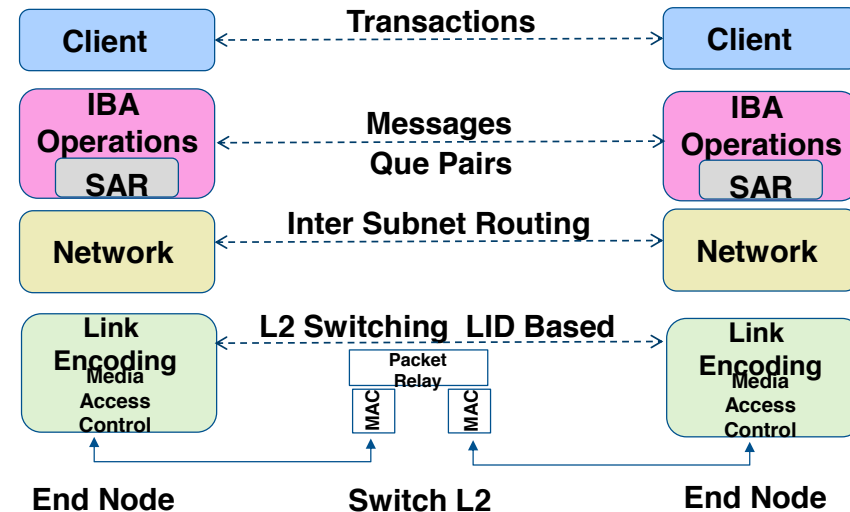
- How packets are routed between different partitions/subnets

- **Data Link (symbols and framing):**

- From source to destination on the same partition subnet Flow control (credit-based); How packets are routed

- **Physical:**

- Signal levels and frequency, media, connectors



# InfiniBand VS. Ethernet

	Ethernet	InfiniBand
Commonly used in what kinds of network	Local area network(LAN) or wide area network(WAN)	Interprocess communication (IPC) network
Transmission medium	Copper/optical	Copper/optical
Bandwidth	1Gb/10Gb	2.5Gb~120Gb
Latency	High	Low
Popularity	High	Low
Cost	Low	High

# InfiniBand Devices

- Host Channel Adapter (HCA)

- Device that terminates an IB link and executes transport-level functions and support the verbs interface



- Switch

- A device that moves packets from one link to another of the same **IB** Subnet



- Router

- A device that transports packets between **different IBA subnets**

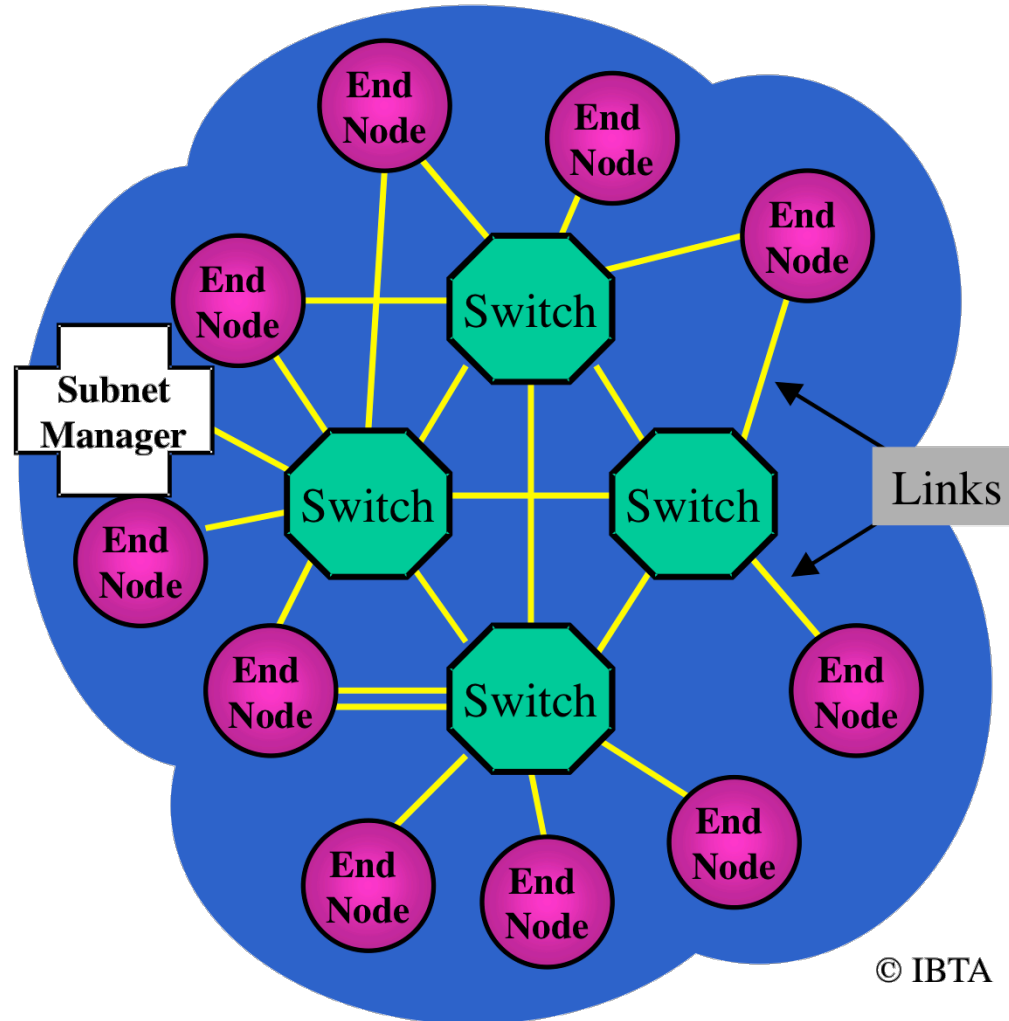


- Bridge/Gateway

- **InfiniBand** to **Ethernet**



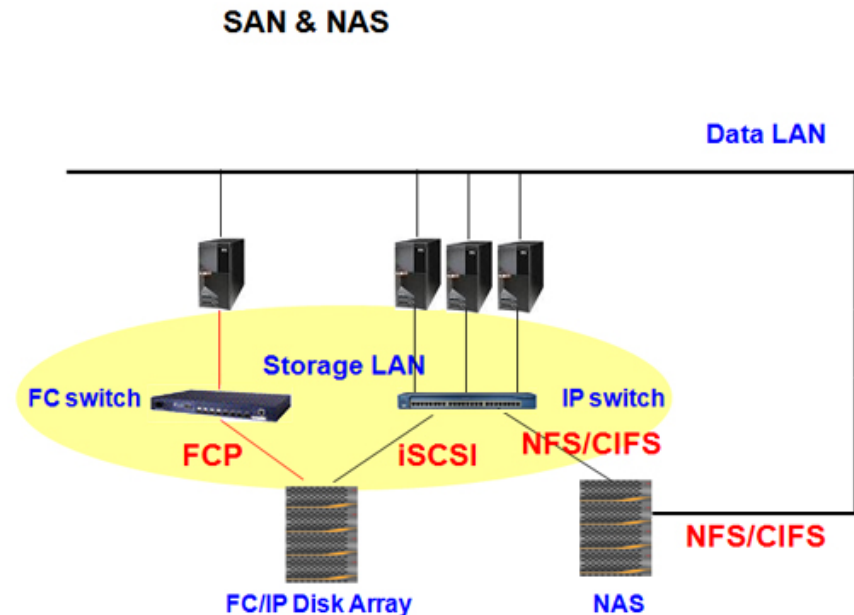
# IBA Subnet





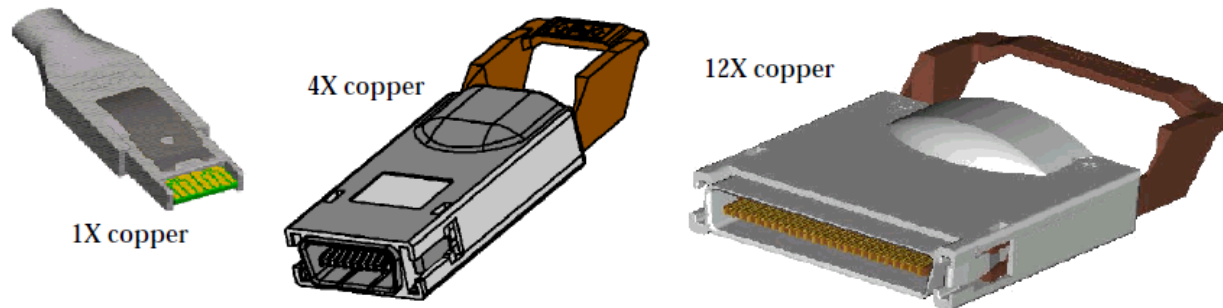
# Endnodes

- IBA endnodes are the ultimate sources and sinks of communication in IBA.
  - They may be host systems or devices.
    - Ex. network adapters, storage subsystems, etc.



# Links

- IBA links are bidirectional point-to-point communication channels, and may be either copper and optical fibre.
  - The base signalling rate on all links is 2.5 Gbaud.
    - Link widths are 1X, 4X, and 12X.



# Channel Adapter

- Channel Adapter (CA) is the interface between an endnode and a link
- There are two types of channel adapters
  - Host channel adapter(HCA)
    - For inter-server communication
    - Has a collection of features that are defined to be available to host programs, defined by verbs
  - Target channel adapter(TCA)
    - For server IO communication
    - No defined software interface

# Addressing

- **LIDs**
  - Local Identifiers, 16 bits
  - Used within a subnet by switch for routing
  - Dynamically assigned at runtime
- **GUIDs**
  - Global Unique Identifier
  - Assigned by vendor (just like a MAC address)
  - 64 EUI-64 IEEE-defined identifiers for elements in a subnet
- **GIDs**
  - Global IDs, 128 bits (same format as IPv6)
  - Used for routing across subnets

# GID: Routing across subnets

## ■ Usage

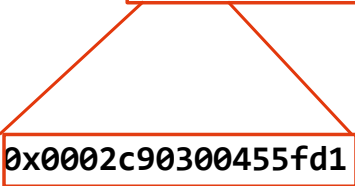
- A 128 bit field in the Global Routing Header (GRH) used to route packets between different IB subnets
- Multicast groups port identifier IB & IPOIB

## ■ Structure

- GUID- 64 bit identifier provided by the manufacturer
- IPv6 type header
- Subnet Prefix: A 0 to 64-bit:
  - Identifier used to uniquely identify a set of end-ports which are managed by a common Subnet Manager

port GUID: 0x0002c90300455fd1

default gid: fe80:0000:0000:0x0002c90300455fd1

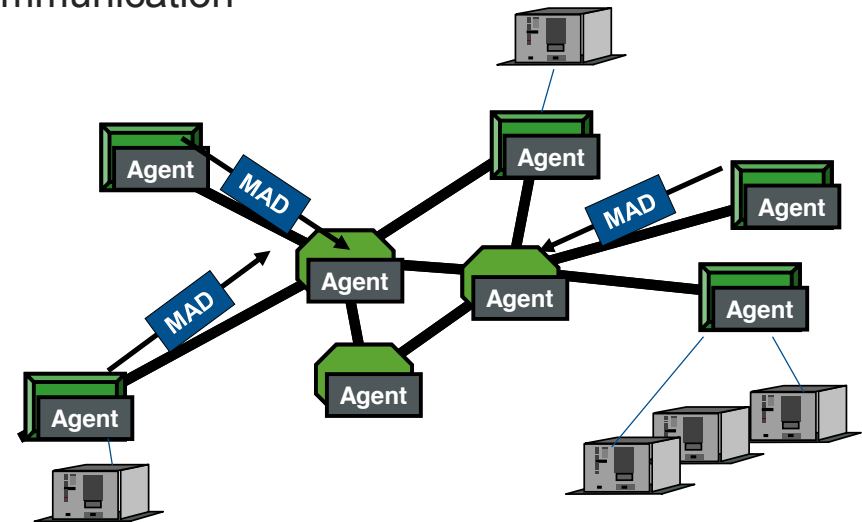


# Switches

- IBA switches route messages from their source to their destination based on routing tables
  - Configured explicitly by Subnet Manager
- Switch size denotes the number of ports
  - The maximum switch size supported is one with 256 ports
- The addressing used by switches
  - Local Identifiers, or LIDs allows 48K endnodes on a single subnet
  - A 64K LID address region is reserved for multicast addresses
  - Routing between different subnets is done on the basis of a Global Identifier (GID) that is 128 bits long

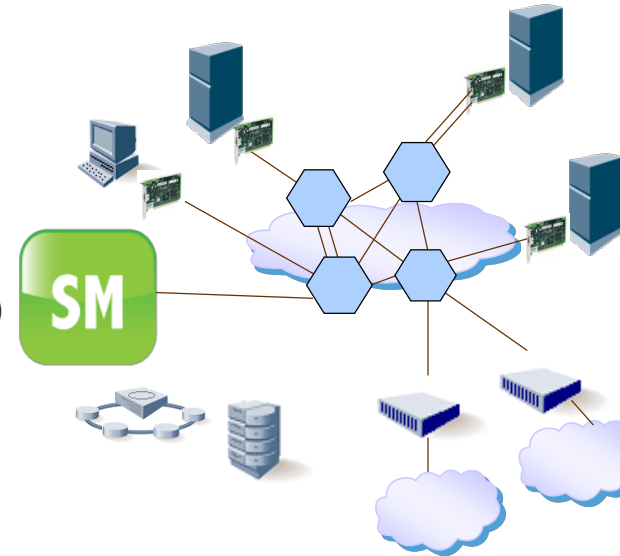
# Management Basics

- Node: any managed entity– End Node, Switch, Router
- Manager: active entity; sources commands and queries
  - The subnet manager (SM)
- Agent: passive (mostly) entity that will reside on every node, responds to Subnet Managers queries
- Management Datagram (MAD):
  - Standard message format for manager–agent communication
  - Carried in an unreliable datagram (UD)



# Subnet Manager

- Every subnet must have at least one
  - Manages all elements in the IB fabric
  - Discover subnet topology
  - Assign LIDs to devices
  - Calculate and program switch chip forwarding tables (LFT pathing)
  - Monitor changes in subnet
- Implemented anywhere in the fabric
  - Node, Switch, Specialized device
- No more than one **active** SM allowed
  - 1 Active (Master) and remaining are Standby (HA)

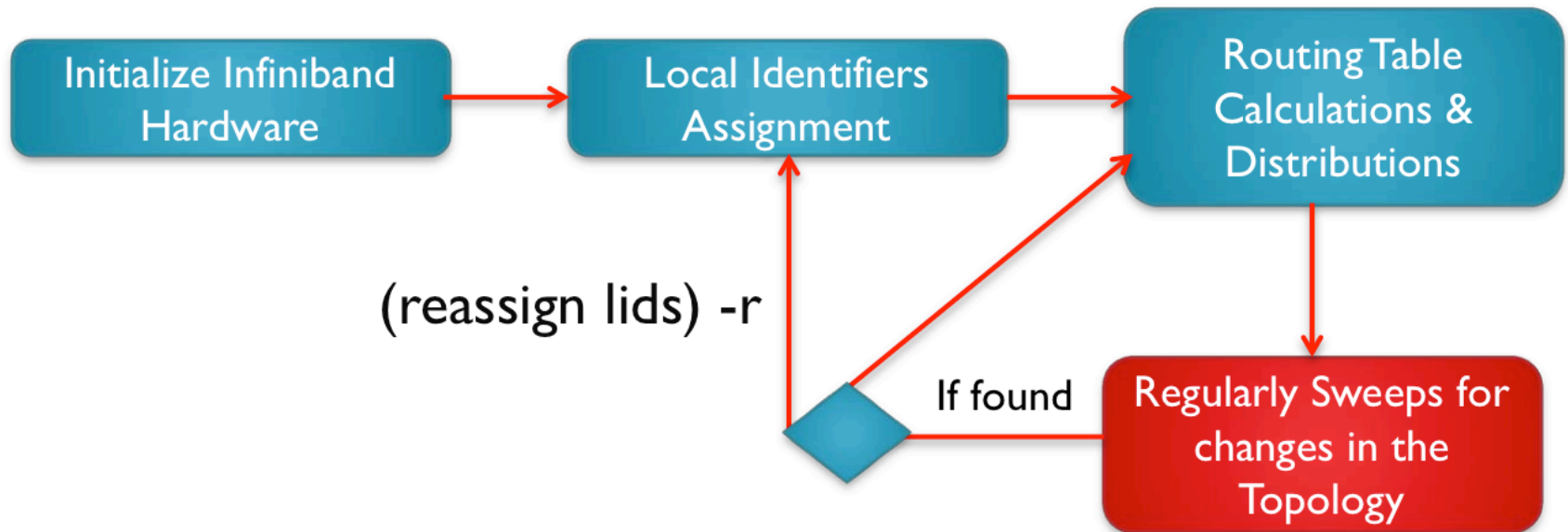


```
[root@l-supp-18 ~]# sminfo
sminfo: sm lid 44 sm guid 0x2c9030010392b, activity count 1372449 priority 14 state 3 SMINFO_MASTER
[root@l-supp-18 ~]# saquery -s
IsSM ports
PortInfoRecord dump:
```



# Subnet Management

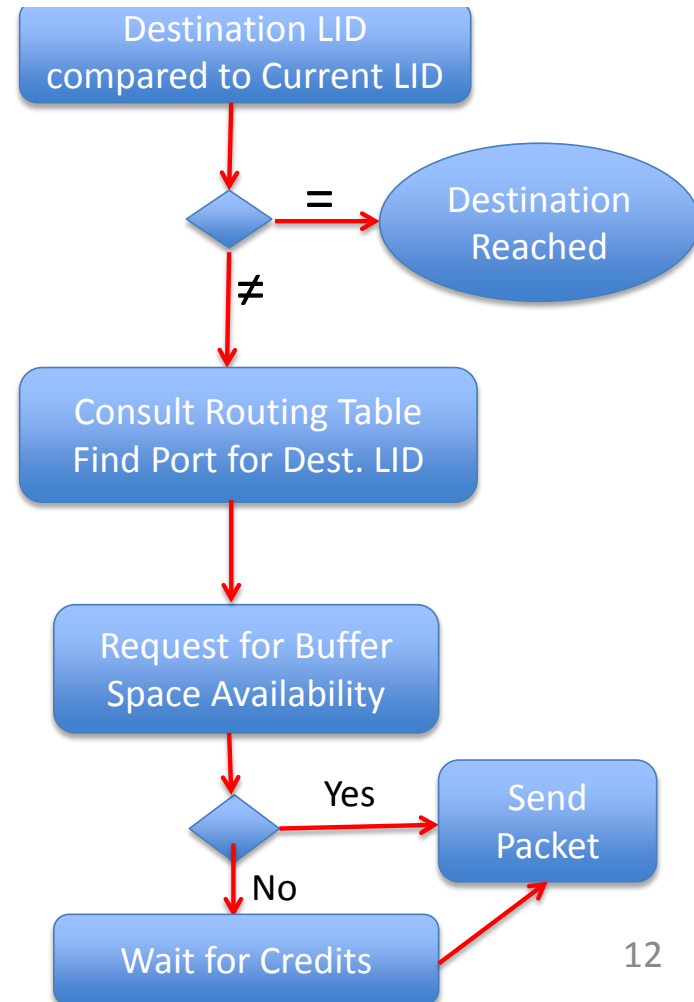
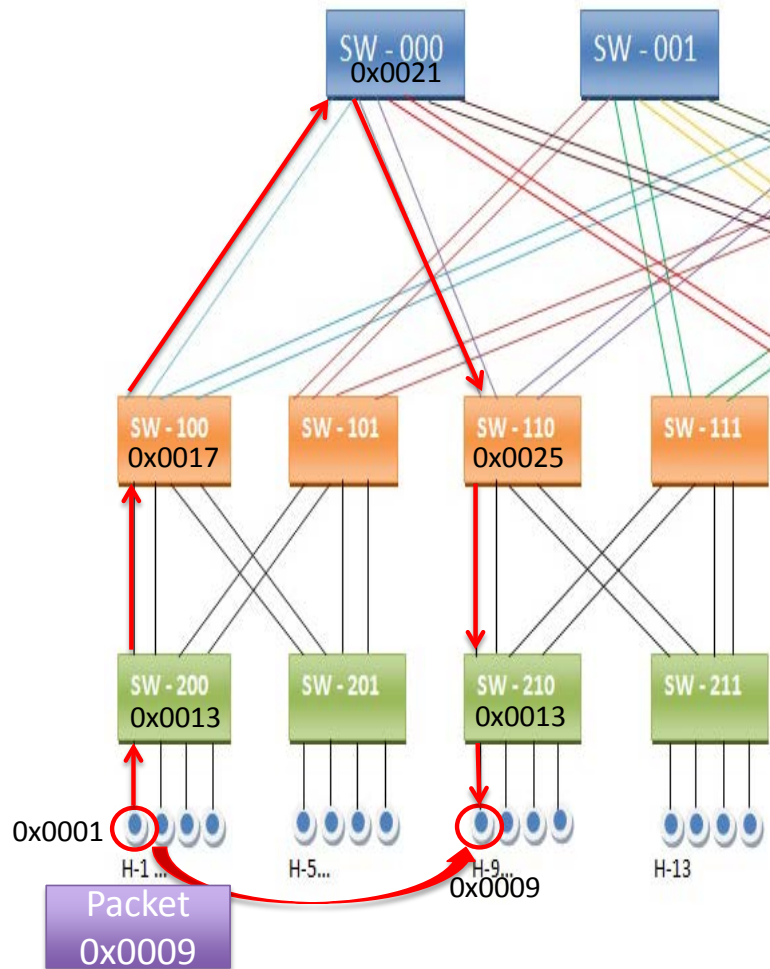
- Subnet Manager:
  - External software service running on an endhost or switch
  - OpenSM – most commonly used
  - Assigns Addresses to endhosts and switches
  - Directly configures routing tables in each switch and device



# Management Datagrams

- All management is performed in-band, using Management Datagrams (MADs).
  - MADs are unreliable datagrams with 256 bytes of data (minimum MTU).
- Subnet Management Packets (SMP) are special MADs for subnet management.
  - Only packets allowed on virtual lane 15 (VL15).
  - Always sent and receive on Queue Pair 0 of each port

# Infiniband routing



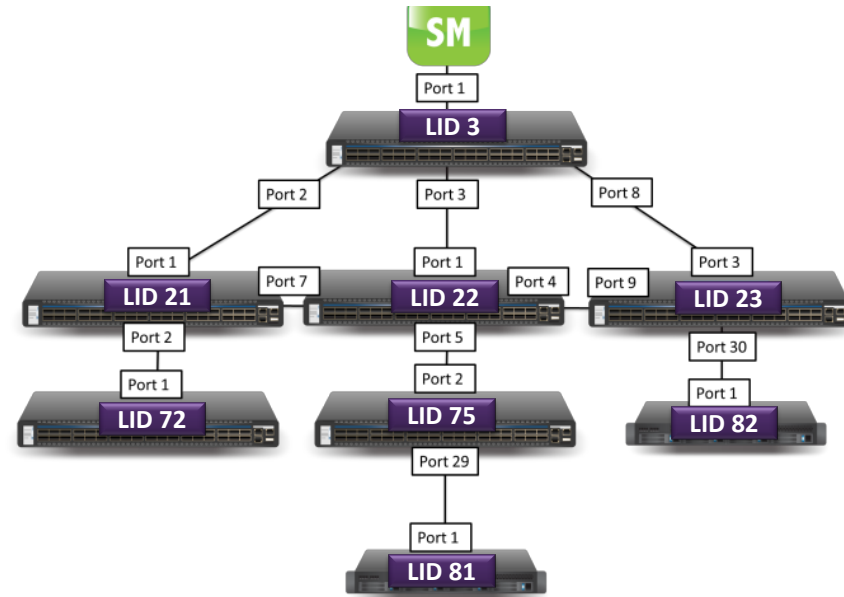
# Infiniband Routing

- After the SM finished gathering all Fabric information , including direct route tables , it assigns a LID to each one of the NODES
- At this stage the LMX table will be populated with the relevant route options to each one of the nodes
- The output of the LMX will provide the Best Route to Reach a DLID as well as the other Routes .
- The Best Path Result Will be based on Shortest Path First (SPF) algorithm

PORT D-LID				
21	1	2	3	1
22	2	1	2	1
23	3	2	1	1
75	3	2	3	2
81	4	3	4	3
82	4	3	2	2



The Dest. LID	Best Route/ exit port
21	2
22	3
23	8
75	3
81	3
82	8



# Infiniband Packet Format

- LRH: Local Routing Header :

- Source & Destination LID
- Service Level-SL
- Virtual Lane-VL
- Packet Length

- BTH: Base Transport Header

- Processed by endnodes

- ICRC: Invariant CRC

- CRC over fields that don't change

- VCRC: Variant CRC

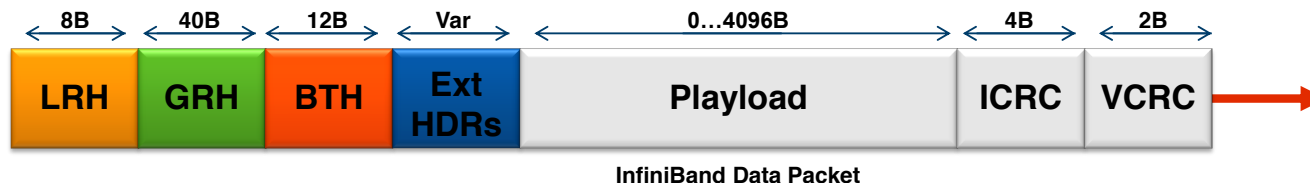
- CRC over fields that can change

- GRH: Global Routing Header

- Routes between subnets

## LFT Switch\_1

The Dest. LID	Best Route/ exit port
21	2
22	3
23	8
75	3
81	3
82	8



# Communication Service Types

Type	Establish connection ?	Transmission Reliable?	Remarks
Reliable Connection (RC)	Yes	Yes	Optional for TCAs, Mandatory for HCAs.
(Unreliable) Datagram (UD)	No	No	none
Unreliable Connection (UC)	Yes	No	Optional.
Reliable Datagram (RD)	Yes, but no one-to-one connection	Yes	Optional.
Raw IPv6 Datagram & Raw Ethertype Datagram (optional) (Raw)	No	No	Allows packets using non-IBA transport layers to traverse an IBA network

# Data Rate

- Effective theoretical throughput

	<b>SDR</b>	<b>DDR</b>	<b>QDR</b>	<b>FDR-10</b>	<b>FDR</b>	<b>EDR</b>
<b>1X</b>	2 Gbit/s	4 Gbit/s	8 Gbit/s	10.3 Gbit/s	13.64 Gbit/s	25 Gbit/s
<b>4X</b>	8 Gbit/s	16 Gbit/s	32 Gbit/s	41.2 Gbit/s	54.54 Gbit/s	100 Gbit/s
<b>12X</b>	24 Gbit/s	48 Gbit/s	96 Gbit/s	123.6 Gbit/s	163.64 Gbit/s	300 Gbit/s

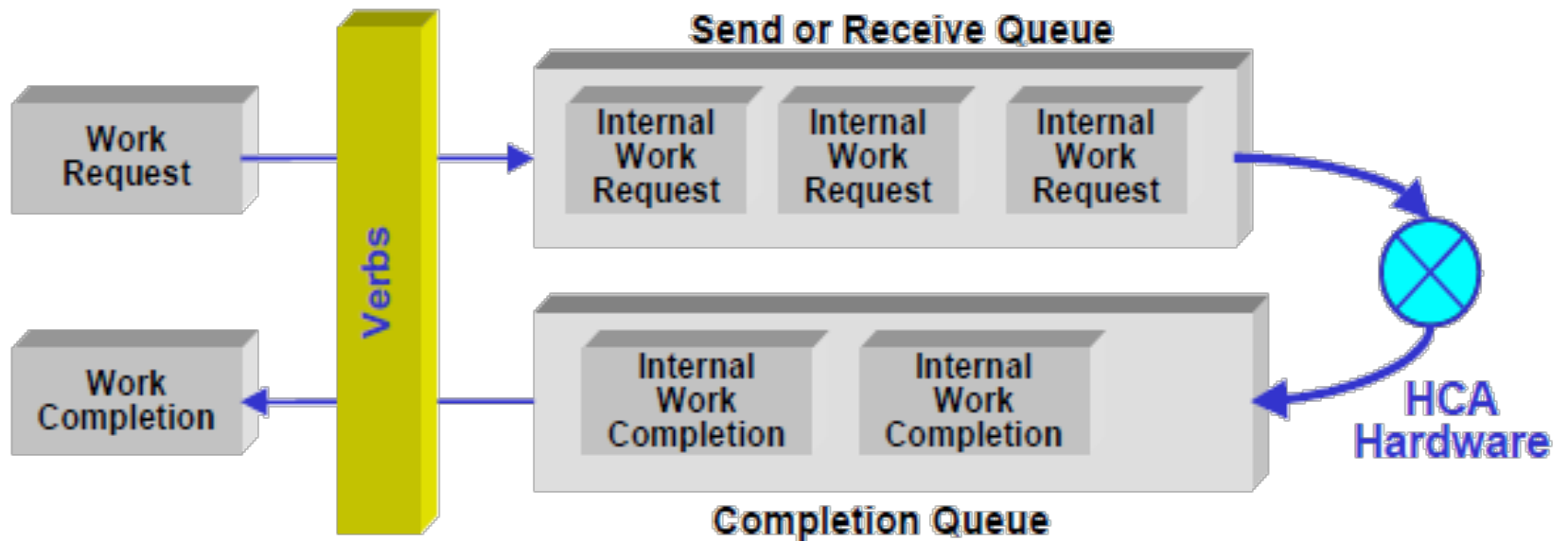
# Queue-Based Model

- Channel adapters communicate using Work Queues of three types:
  - **Queue Pair(QP)** consists of
    - Send queue
    - Receive queue
  - **Work Queue Request (WQR)** contains the communication instruction
    - It would be submitted to QP.
  - **Completion Queues (CQs)** use Completion Queue Entries (CQEs) to report the completion of the communication



# Queue-Based Mode

© IBTA



# Access Model for InfiniBand

- Privileged Access
  - OS involved
  - Resource management and memory management
    - Open HCA, create queue-pairs, register memory, etc.
- Direct Access
  - Can be done directly in user space (OS-bypass)
  - Queue-pair access
    - Post send/receive/RDMA descriptors.
  - CQ polling

# Access Model for InfiniBand

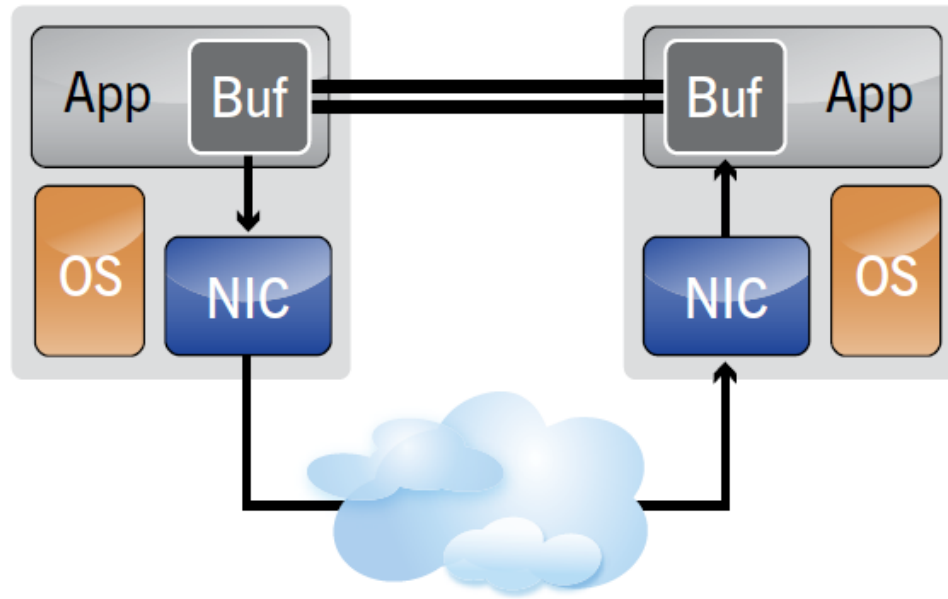
- Queue pair access has two phases
  - Initialization (privileged access)
    - Map doorbell page (User Access Region)
    - Allocate and register QP buffers
    - Create QP
  - Communication (direct access)
    - Put WQR in QP buffer.
    - Write to doorbell page.
      - Notify channel adapter to work

# Access Model for InfiniBand

- CQ Polling has two phases
  - Initialization (privileged access)
    - Allocate and register CQ buffer
    - Create CQ
  - Communication steps (direct access)
    - Poll on CQ buffer for new completion entry

# Memory Model

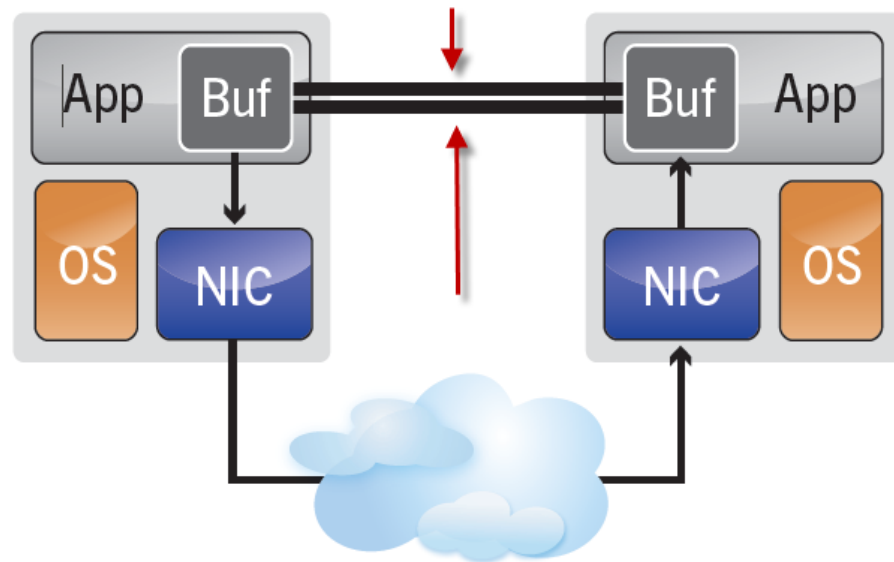
- Control of memory access by and through an HCA is provided by three objects
  - Memory regions
    - Provide the basic mapping required to operate with virtual address
    - Have R\_key for remote HCA to access system memory and L\_key for local HCA to access local memory.
  - Memory windows
    - Specify a contiguous virtual memory segment with byte granularity
  - Protection domains
    - Attach QPs to memory regions and windows



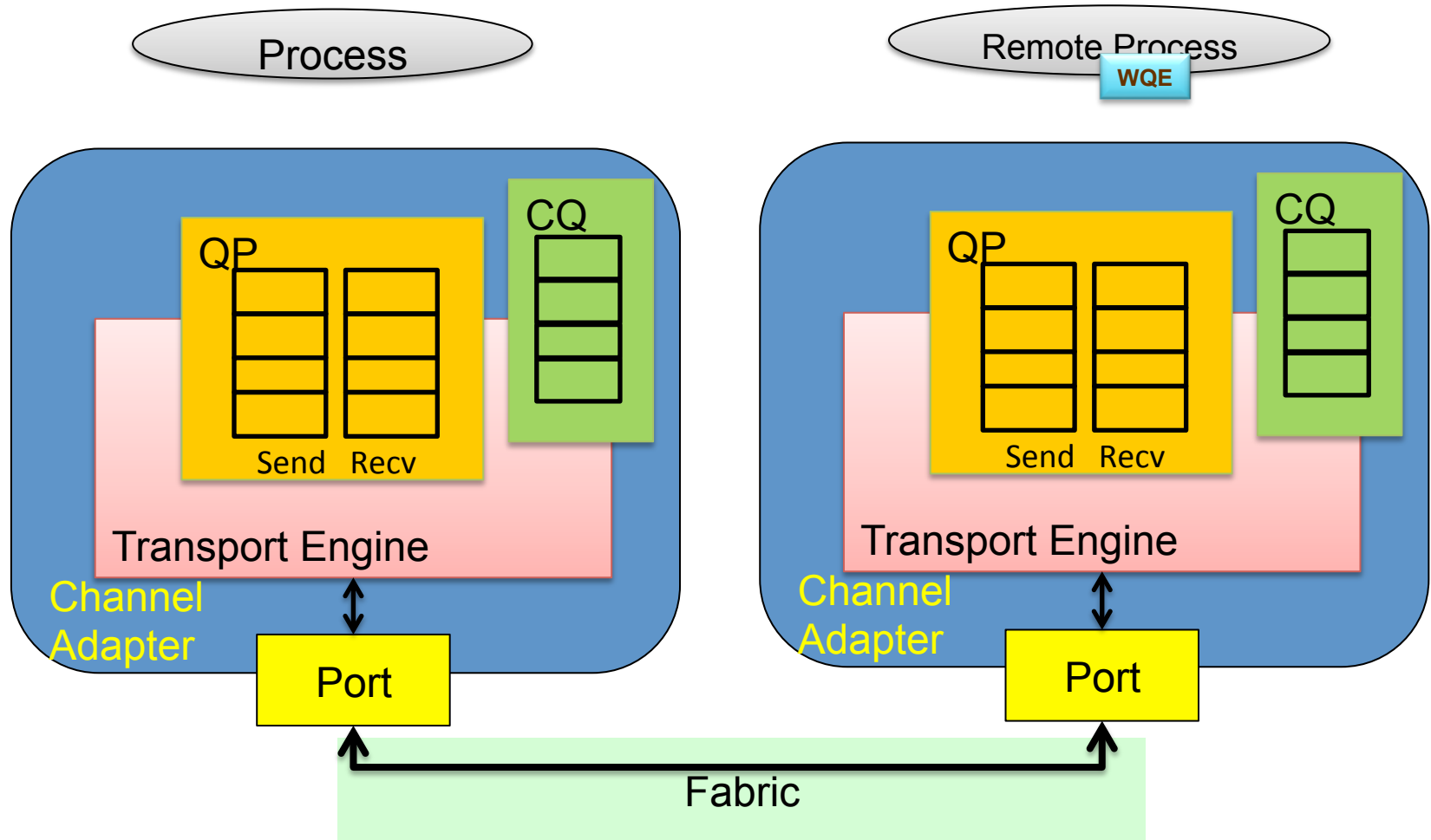
- *InfiniBand creates a channel directly connecting an application in its virtual address space to an application in another virtual address space.*
- *The two applications can be in disjoint physical address spaces – hosted by different servers.*

# Communication Semantics

- Two types of communication semantics
  - **Channel** semantics
    - With traditional send/receive operations.
  - **Memory** semantics
    - With RDMA operations.

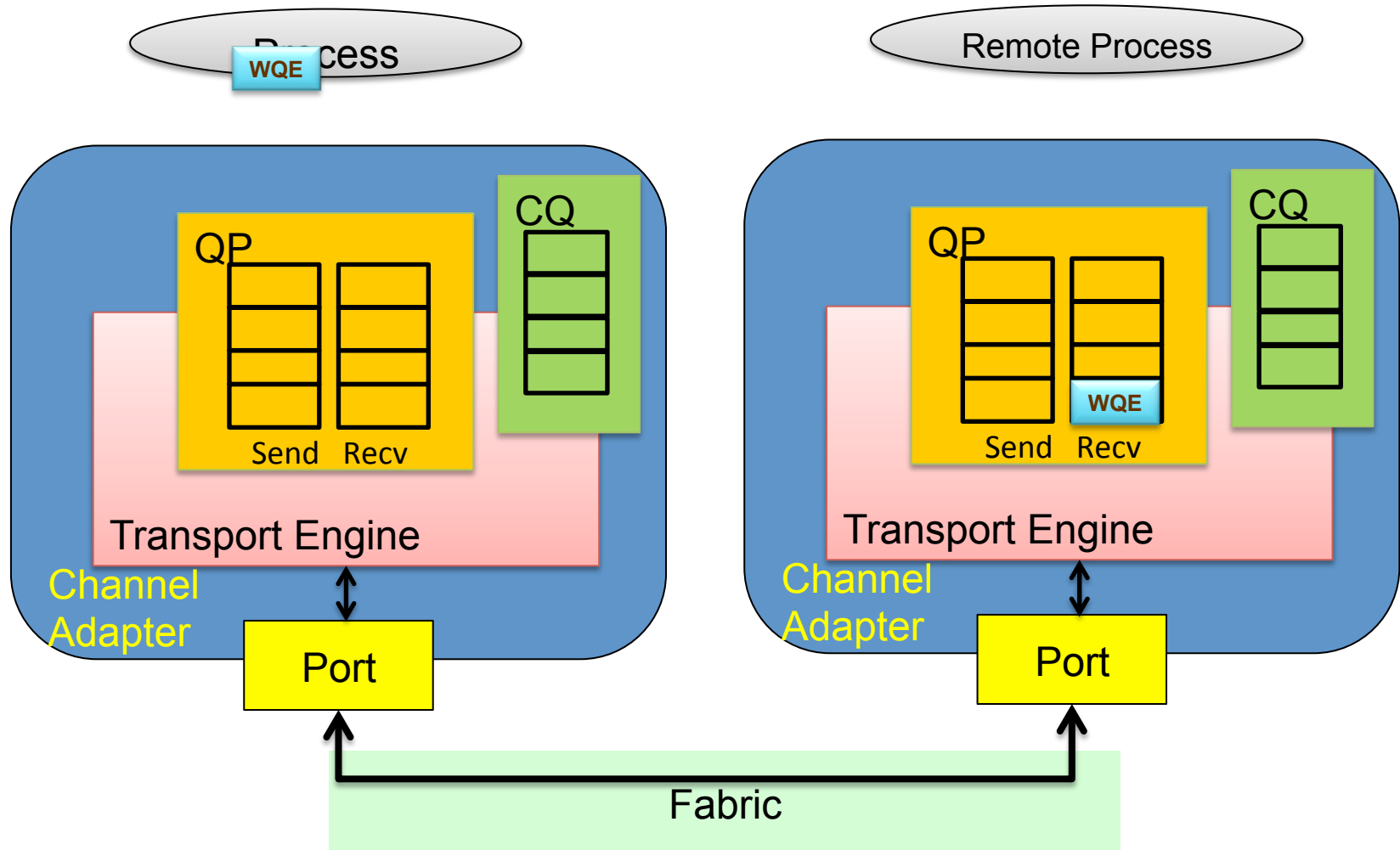


# Send and Receive

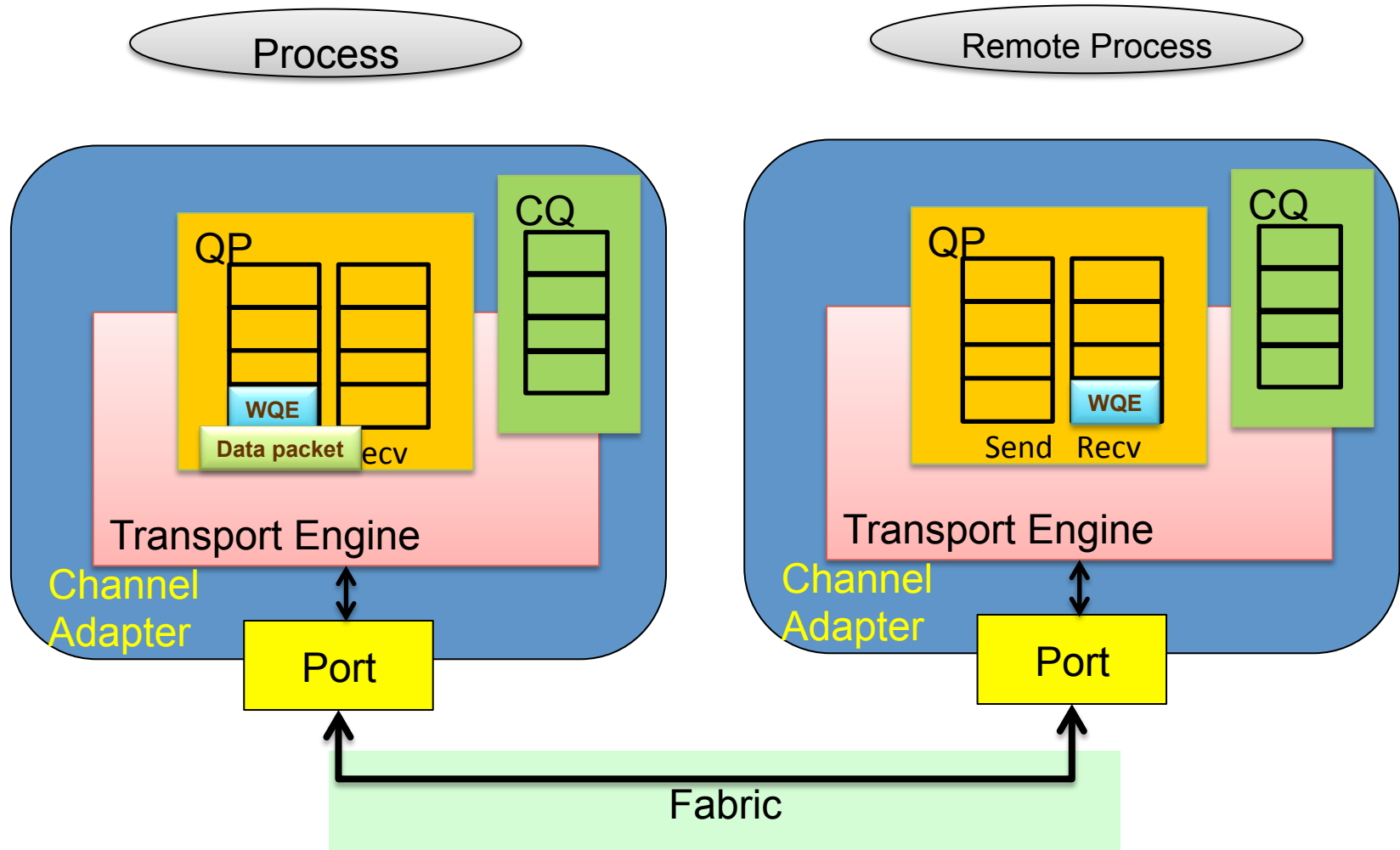




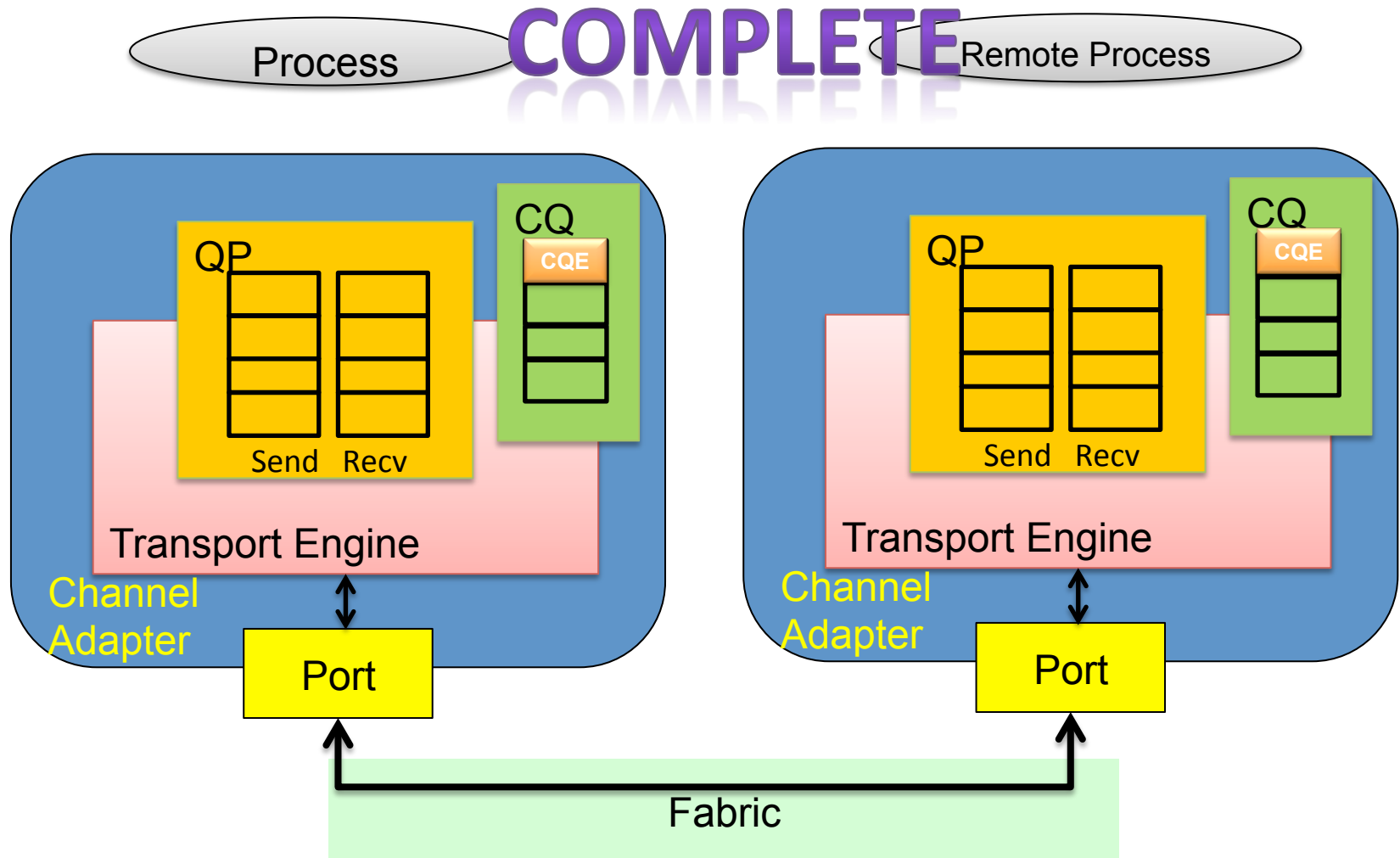
# Send and Receive



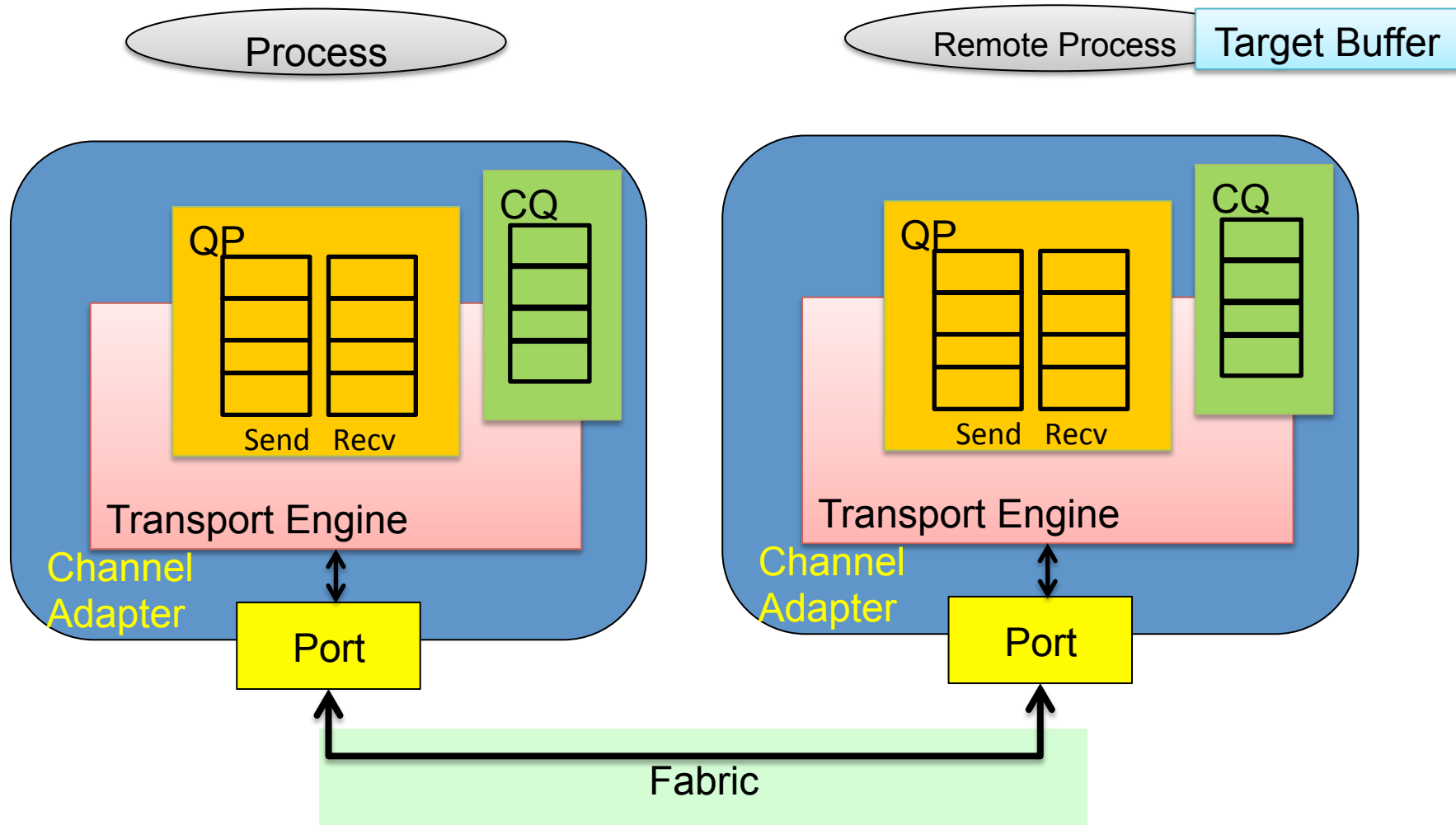
# Send and Receive



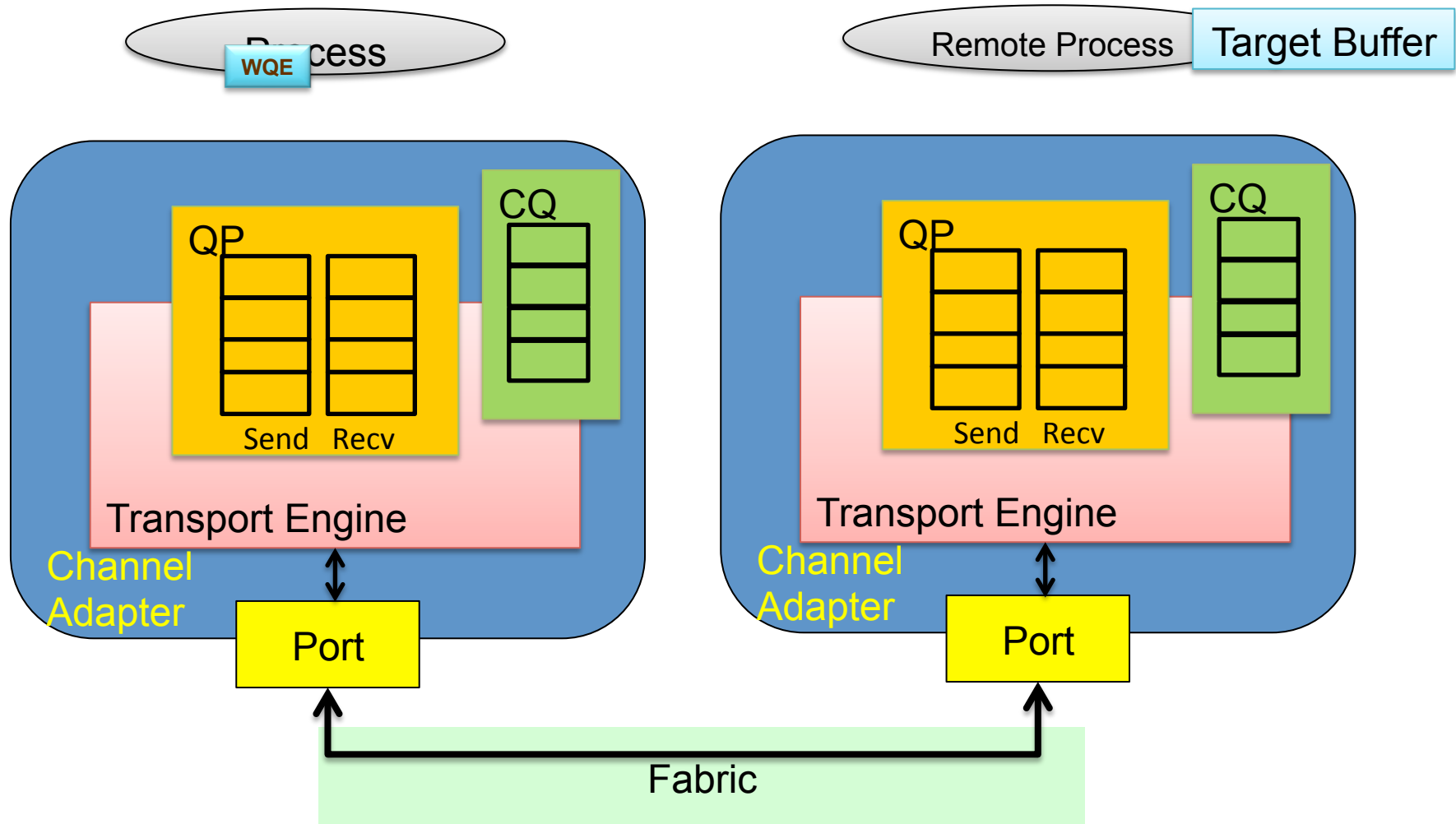
# Send and Receive



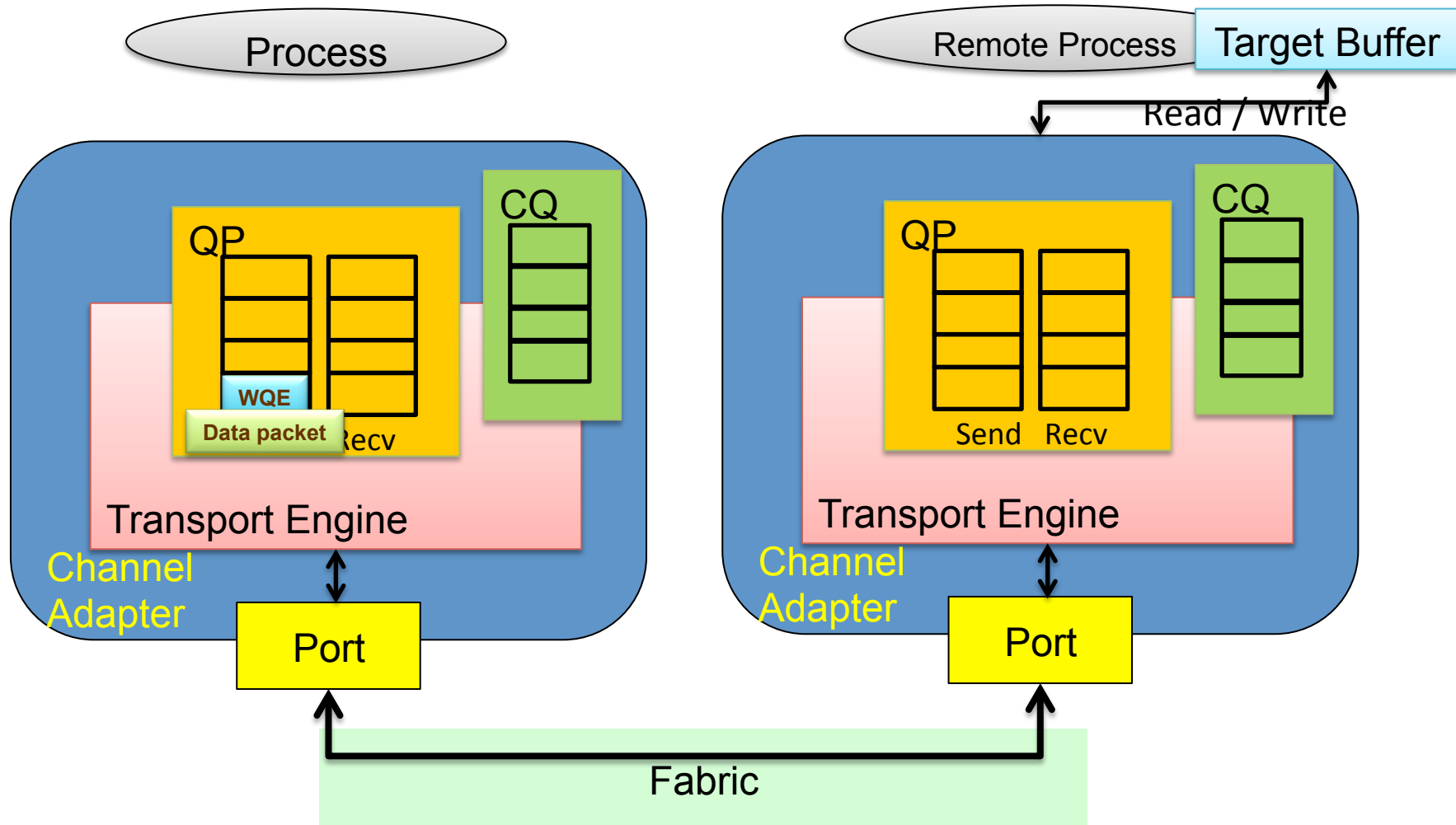
# RDMA Read / Write



# RDMA Read / Write



# RDMA Read / Write



# RDMA Read / Write

