RDMA enabled NIC (RNIC) Verbs Overview

Renato Recio

RNIC Verbs

- The RDMA Protocol Verbs Specification describes the behavior of RNIC hardware, firmware, and software as viewed by the host,
  - not the host software itself, and
  - not the programming interface viewed by the host.

- The behavioral description is specified in the form of an RNIC Interface (RI) and a set of RNIC Verbs:

  - A RNIC Interface defines the semantics of the RDMA services that are provided by an RNIC that supports the RNIC Verb Specification. The RI can be implemented through a combination of hardware, firmware, and software.

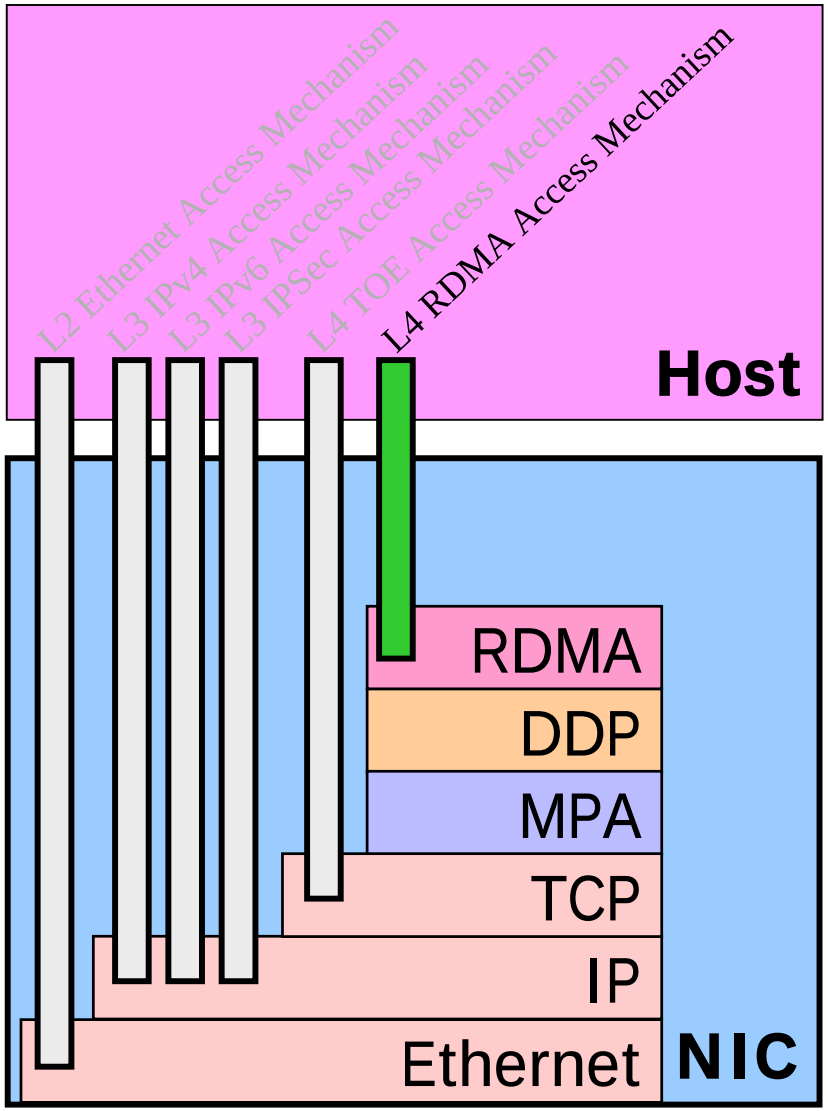
  - A Verb is an operation which an RNIC Interface is expected to be able to perform.

First Principles

- Strove to minimize the number of options in the RNIC Verbs Spec.
- Strove to minimize semantic and interface delta from existing standards (i.e. InfiniBand).
- RNIC Verbs Specification supports TCP transport.
  - Some effort was placed on provisioning for SCTP, but additional work would be needed for SCTP.
- Define a common RNIC Interface that can be used by O/Ss and appliances to access RNIC functions.
- Consumer does not directly access queue elements.
  - IB style queue model (vs VIA style queue model).

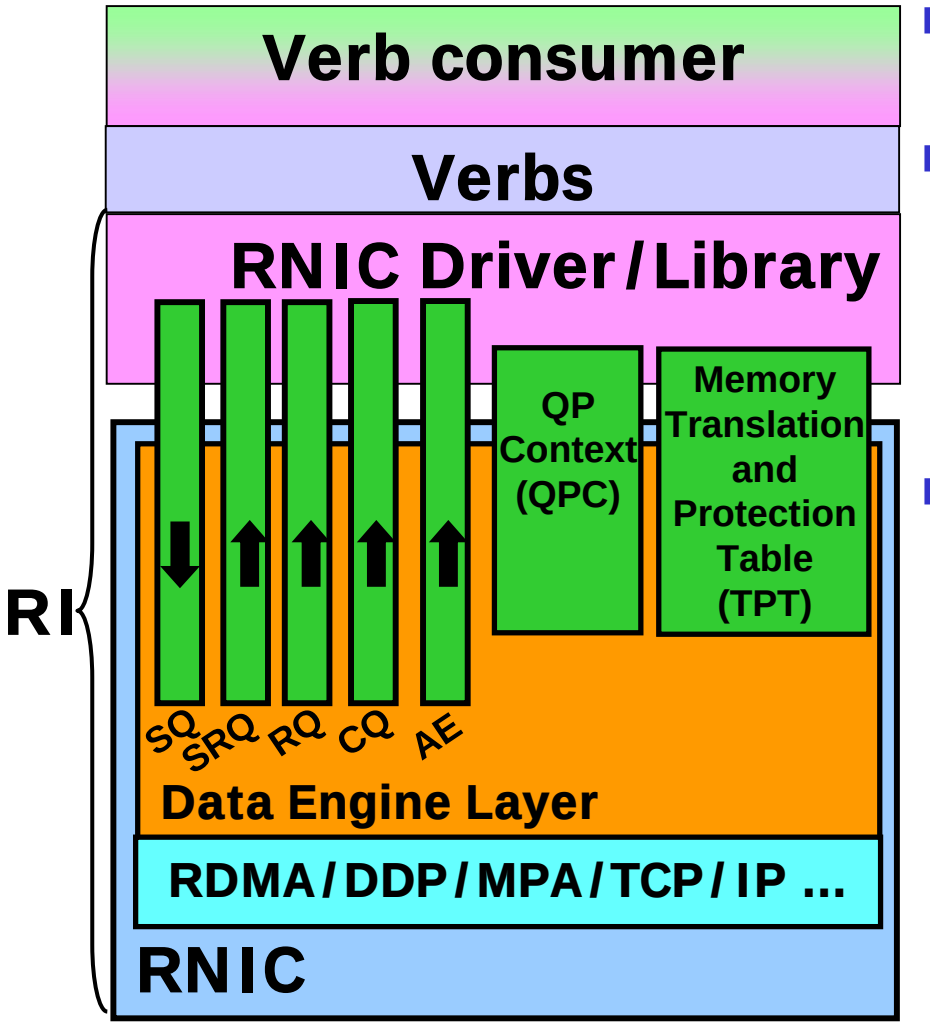
Emerging NIC Model

Example of service offered by a network stack offload NIC.



- Background on RNIC Verb scope.
  - NICs are in the process of incorporating layer 3 and layer 4 functions.
  - The scope of the RNIC Verbs Spec is Layer 4 access to RDMA functions:
    - Definition of the verbs (and their associated semantics) needed to access RDMA Protocol Layer functions.semantics.
    - Except for connection management and teardown semantics, access to other layers is not semantically defined by the RNIC verbs.

RNIC Model Overview

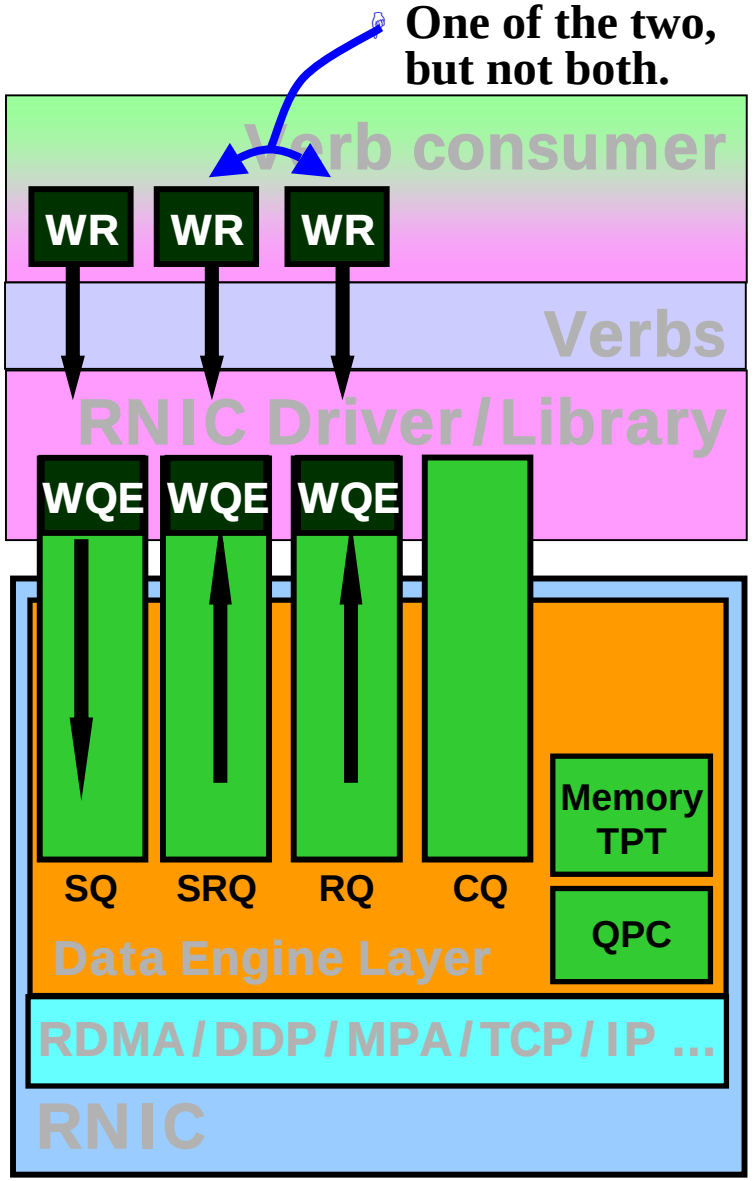


SQ – Send Queue
RQ – Receive Queue
SRQ – Shared RQ

QP – Queue Pair
QP = SQ + RQ
CQ = Completion Queue

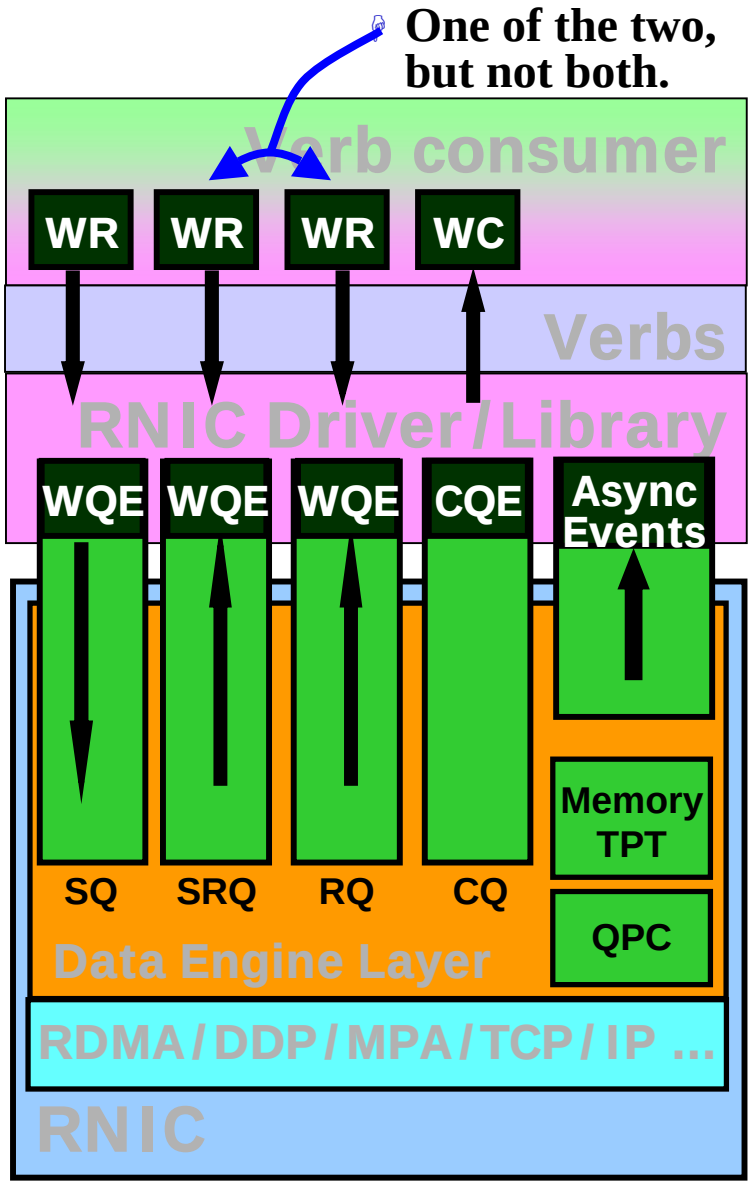
- Verb consumer – SW that uses RNIC to communicate to other nodes.
- Communication is thru verbs, which:
  - Manage connection state.
  - Manage memory and queue access.
  - Submit work to RNIC.
  - Retrieve work and events from RNIC.
- RNIC Interface (RI) performs work on behalf of the consumer.
  - Consists of software, firmware, and hardware.
  - Performs queue and memory mgt.
  - Converts Work Requests (WRs) to Work Queue Elements (WQEs).
  - Supports the standard RNIC layers (RDMA, DDP, MPA, TCP, IP, and Ethernet).
  - Converts Completion Queue Elements (CQEs) to Work Completions (WCs).
  - Generates asynchronous events.

Send/Receive Work Submission Model



- Consumer work submission model:
  - For all outbound messages, Consumer uses SQ.
  - For inbound Send message types, Consumer uses:
    - RQ, or
    - SRQ (if QP was created with SRQ association).
- Work is submitted in the form of a Work Request, which includes:
  - Scatter/gather list of local data segments, each represented by Local: STag, TO, and Length
  - Other modifiers (see specification).
- WR types:
  - Send (four types), RDMA Write, RDMA Read, Bind MW, Fast-Register MR, and Invalidate STag
- RI converts work requests into WQEs and processes the WQEs.
  - RI returns control to consumer immediately after WR has been converted to WQE and submitted to WQ.
  - After control is returned to consumer, RI can not modify WR.

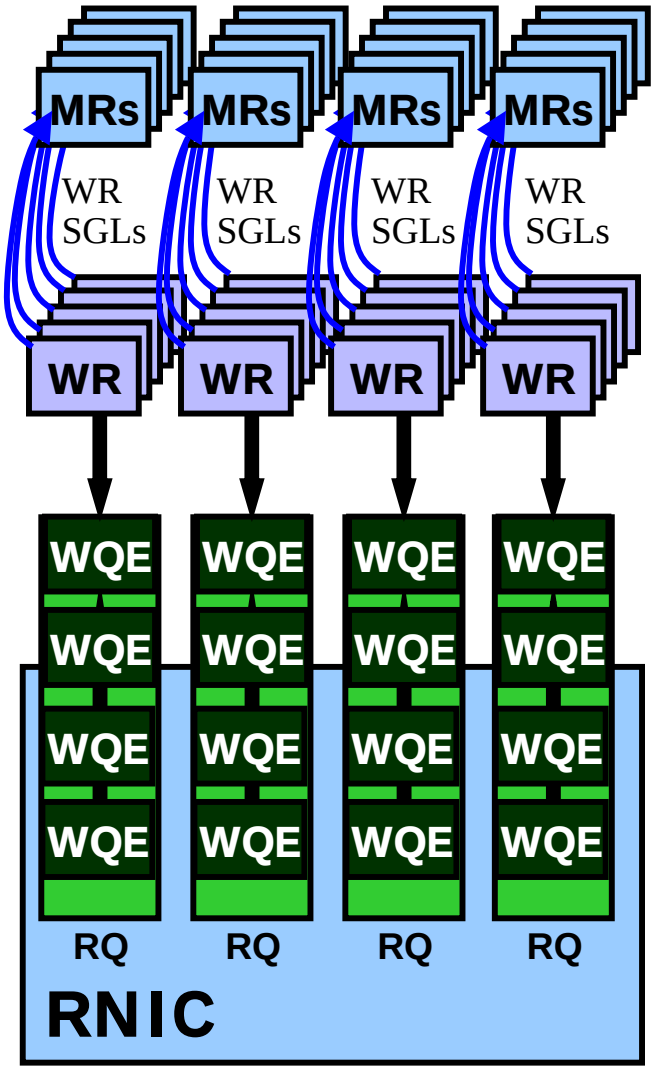
Completion and Event Model



- RI completion processing model:
  - Consumer sets Completion Event Handler.
  - WQE processing model:
    - For SQ,
      - RNIC performs operation.
      - When operation completes, if WQE was signaled (or completed in error), a CQE is generated.
    - For RQ,
      - If no SRQ is associated with QP, when operation completes, WQE is converted into CQE.
      - If SRQ is associated with QP, RNIC behaves as if WQE is pulled from SRQ and moved to RQ, and then moved from RQ to CQ when the message completes.
  - Consumer polls CQ to retrieve CQEs as WC.
- RI event processing model, includes:
  - Consumer sets Asynchronous Event Handler.
  - Asynchronous events are sent to consumer through Async Event Handler.

Motivation for Shared RQ

Send/Receive skew between connections may result in inefficient memory or wire usage.



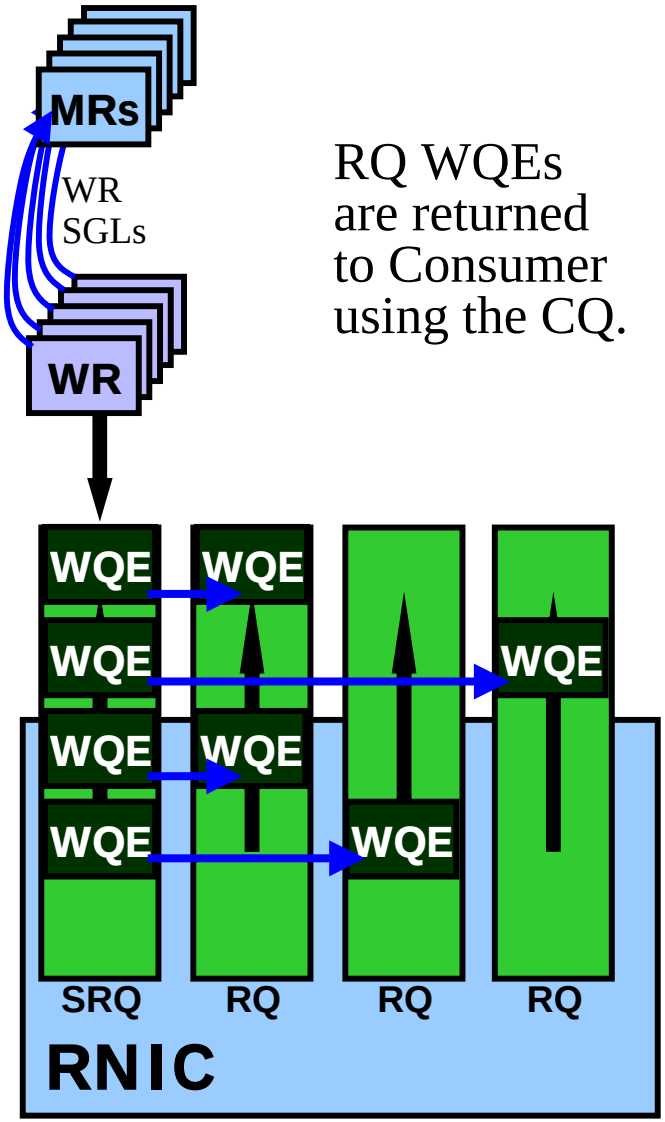
- Under traditional RQ models (e.g. VIA, IB), for each connection the Consumer posts the number of receive WRs necessary to handle incoming receives on that connection.
- If the Data Sink Consumer cannot predict the incoming rate on a given connection, the Data Sink Consumer must either:
  - A. Post the a sufficient number of RQ WQEs to handle the highest incoming rate ***for each connection***.
    - Post RQ WQE Rate >= Incoming rate
    - Where Incoming rate may equal link B/W.
  - B. Let message flow control cause the Data Source to back off until Data Sink posts more WRs.

Either approach is inefficient:

- A. Holding WQEs in RQs that are relatively inactive wastes the memory space associated with the SGE of the WQEs.
- B. Data Sink Consumer may be unaware that the RQ is starving.

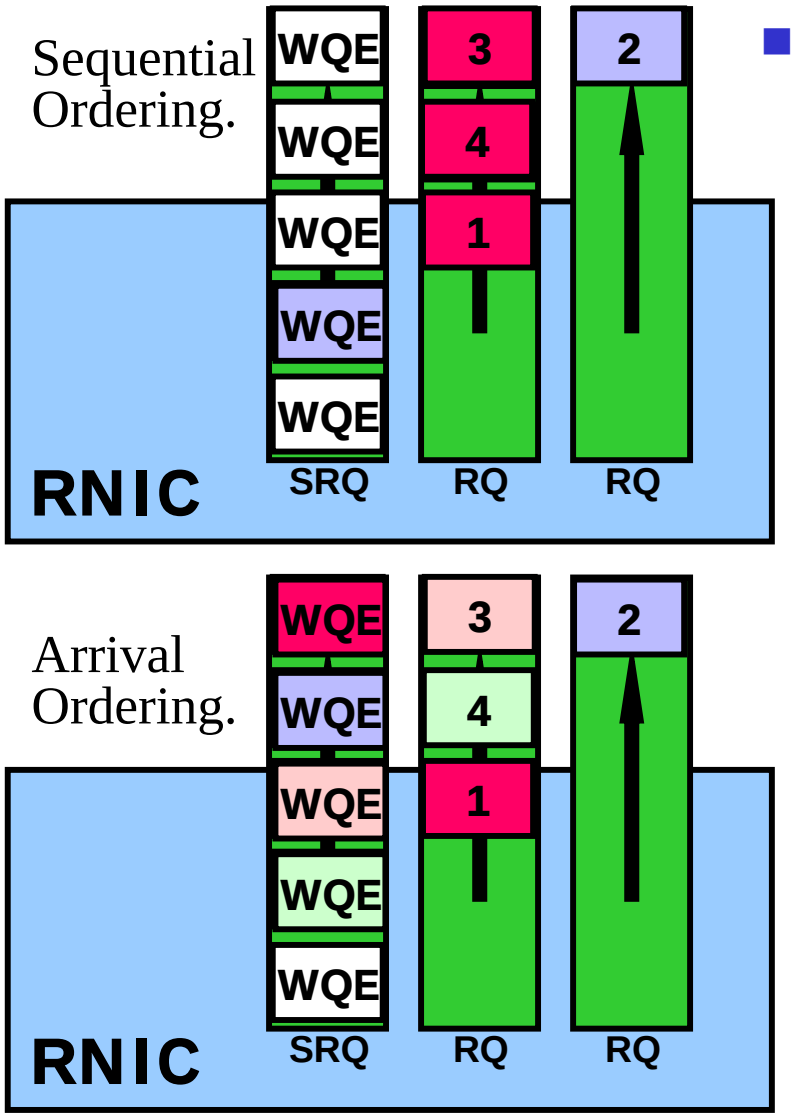
Conceptual RNIC Shared RQ WR Model

Shared RQ enables more efficient use of memory or wire.



- Some Consumers (e.g. storage subsystems) manage a fixed-size pool of buffers amongst several different transport connections to reduce buffer space requirements.
  - The SRQ model, enables these to post receive WRs to a queue that is shared by a (Consumer defined) set of connections.
- Under the SRQ model the Consumer posts receive WRs to the SRQ and, as incoming segments arrive on a given QP that is associated with the SRQ, the SRQ WQEs are transferred from the SRQ to the RQ of the given QP.
  - The maximum incoming rate is bounded by the ***link rate***:
    - Post SRQ WQE Rate =< Link B/W.

Shared RQ Processing Model



Number represents arrival order.
Color represents when WQE is dequeued.
Order, from bottom, in RQ represents MSN order.

- RNIC SRQ processing semantics:
  - On a given RQ, message WCs are returned in the order the associated messages were sent.
  - For a Send type message that arrives in order on a given RDMAP Stream:
    - Next available WQE is pulled from SRQ,
    - RNIC must behaves as if WQE is moved to RQ
      - Whether it actually does or not is implementer's choice.
  - For a message that arrives out of order, two options are allowed:
    - Sequential ordering:
      - RNIC dequeues one WQE for the incoming message plus one WQE each message with an MSN lower than the out-of-order message that doesn't already have a WQE.
    - Arrival ordering:
      - RNIC dequeues one WQE. WQEs required for messages with lower MSNs, will be dequeued when those messages arrive.

  Query RNIC has output modifier stating which of the above options is supported by the RNIC.

Work Request Types and Work Request Posting Mechanisms

	Send Queue	Receive Queue
WRs types		
Send/Receive	Send Send with SE Send with Invalidate Send with SE and Invalidate	Receive
RDMA	RDMA Write RDMA Read RDMA Read with Invalidate	
Memory	Bind Fast Register MR Invalidate Local STag	
WR Posting Attributes		
	Posted as single WR, or List of WRs	If QP is not associated with SRQ, WR posted to RQ. If QP associated with SRQ, WR posted SRQ.

Summary Error and Event Classes

- RNIC Verb Errors and Events:

SQ **Errors**,
by where detected, and how it is returned:

1) Local **Immediate,** returned before WQE posted.

Returned through CQE on associated CQ:

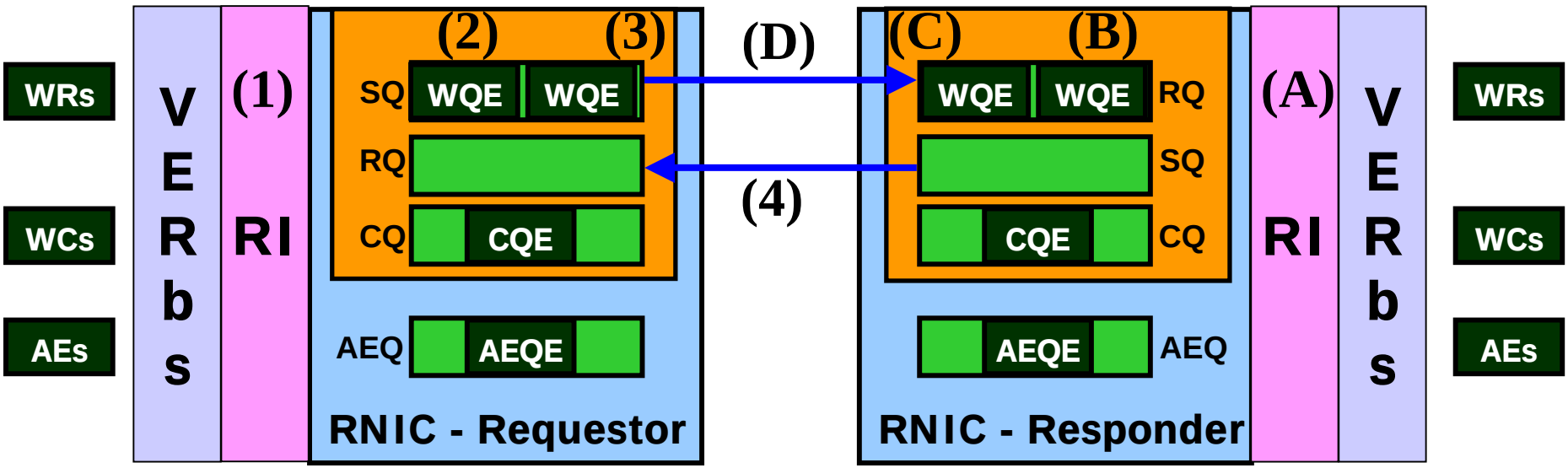
2) Local **Completion**, pre-WQE processing
3) Local **Completion**, post WQE processing
4) Remote **Completion**

RQ **Errors**,
by where detected, and how it is returned:

A. Local **Immediate,** returned before WQE posted.

Returned through CQE on associated CQ:

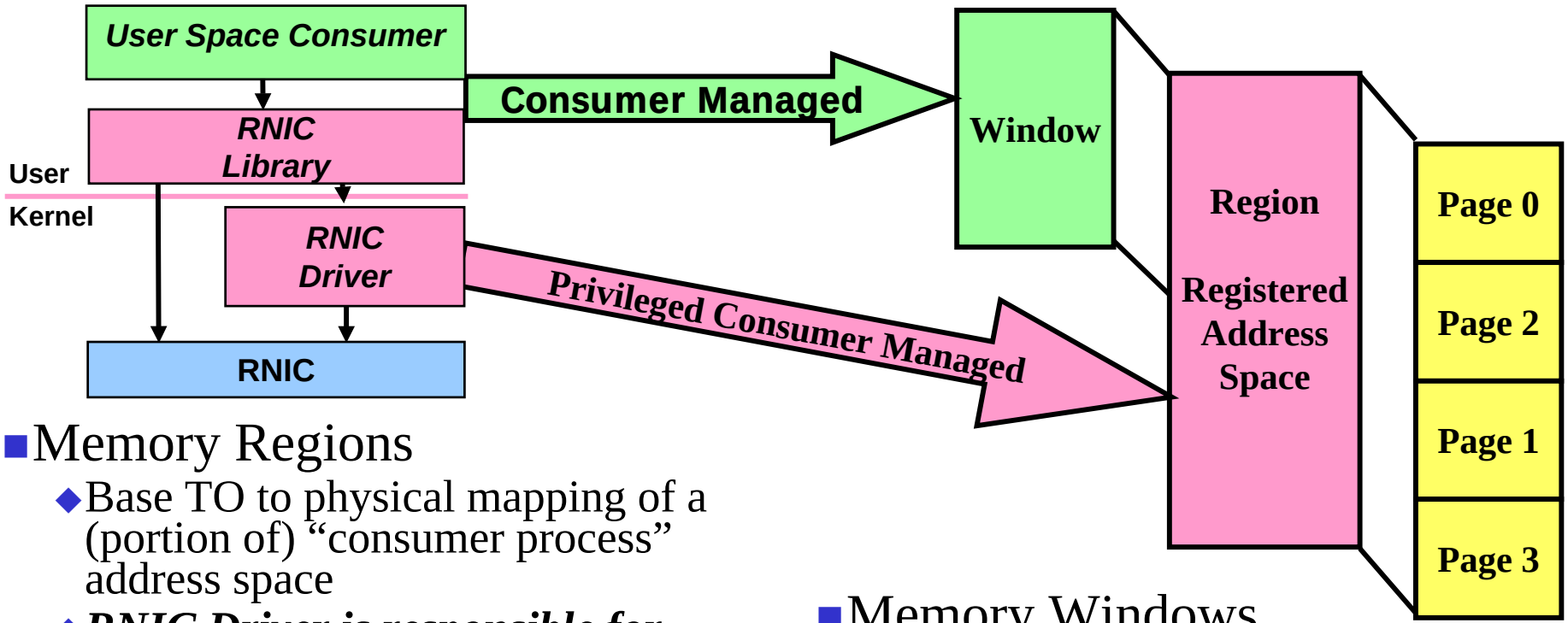
B. Local **Completion**, pre-WQE processing
C. Local **Completion**, post WQE processing
D. Remote **Completion**



Asynchronous Error and Event:

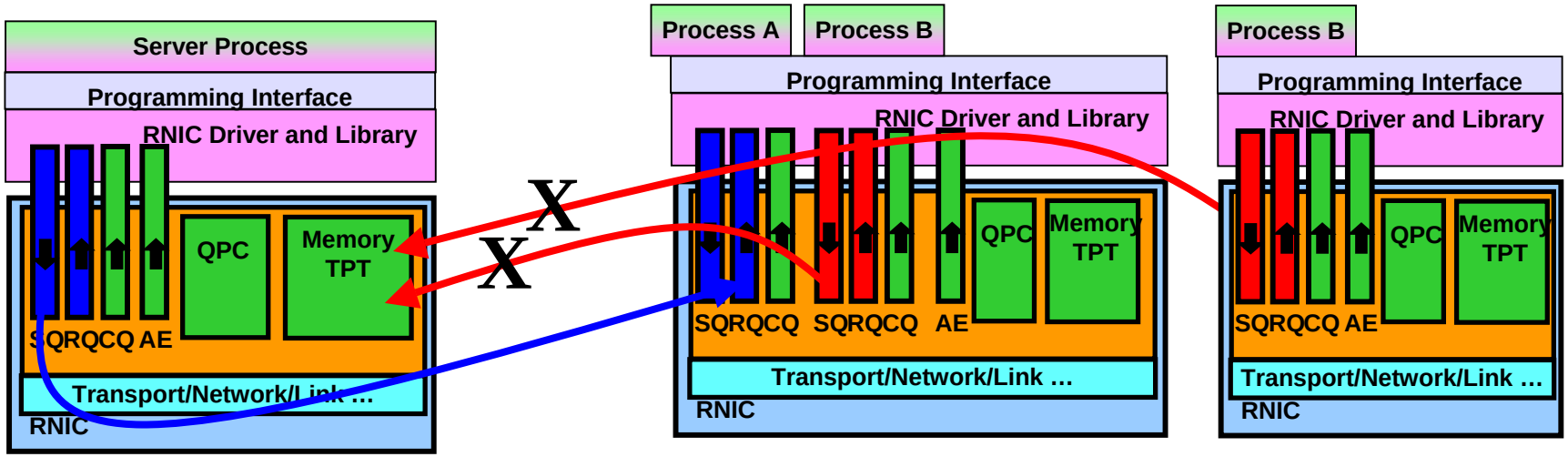
- Locally detected SQ, RQ, or RNIC errors or event that cannot be returned through CQ.
- Returned instead through RNIC's Asynchronous Event Queue.

User Space Memory Management Model



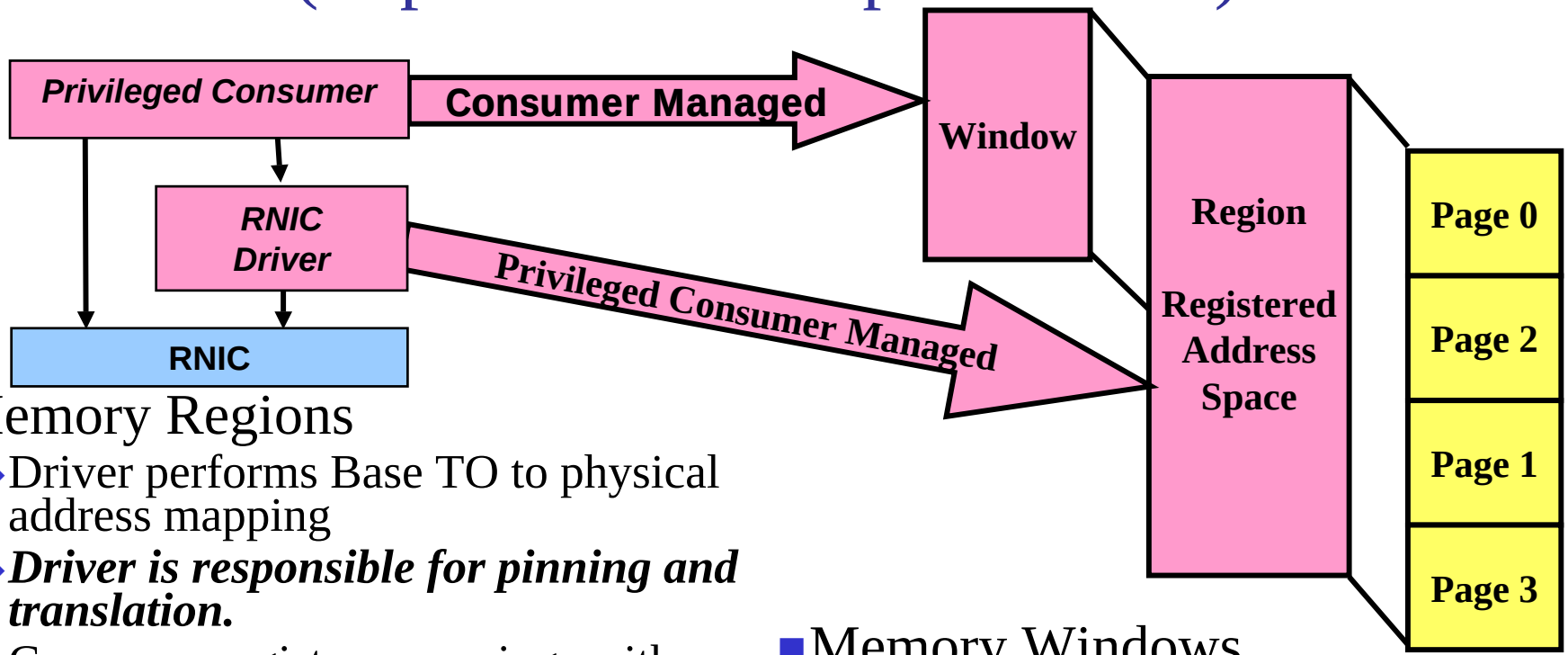
- Memory Regions
  - Base TO to physical mapping of a (portion of) "consumer process" address space
  - ***RNIC Driver is responsible for pinning and translation.***
  - Explicit registration by consumer with the RNIC Driver through RI registration mechanisms.
  - QP access to Regions managed through Protection Domains
  - QP consumers use Base TO addressing, RNIC performs Base TO to Physical mapping
- Memory Windows
  - Windows enable flexible & efficient dynamic RDMA access control to underlying Memory Regions
  - Consumer uses Send Queue to "bind" a pre-allocated Window to a specified portion of an existing Region.
  - ***QP access to Windows managed through QP ID.***

Why Associate MW with QP ID?



- Considered associating MWs with PDs, but encountered problems.
  - When a single server process communicates with multiple processes at the clients, two options are available from protecting STag access control:
    - Option 1: Use the same PD on all QPs.
      - Problem: When multiple clients, or when multiple processes running on the same client, connect to the server process, process A can access the STag exposed to process B.
    - Option 2: Use different PDs on each QP.
      - Problem: If the same region is to be exposed to multiple clients/processes, the server process would need to create multiple windows, one for each client. Each MW would consumes one Memory Translation and Protection Table entry.
  - Net:
    - PD based MW access control is not fine grained enough to support 1:N server:client models.
- RNIC MW bind and access semantics:
  - At bind time QP ID is associated with the MW (vs the PD).
  - At access time, QP ID of MW must match QP ID of QP.

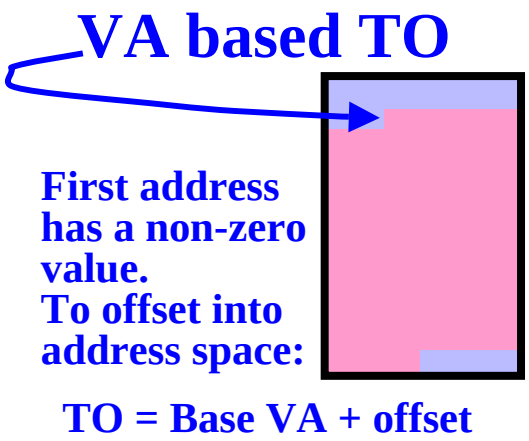
Privileged Space Memory Mgt Model
(Superset of user space model)

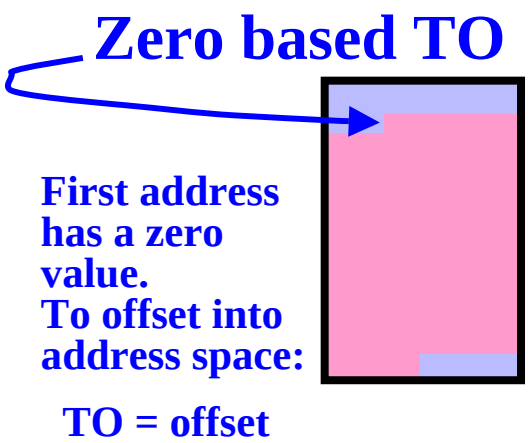


- Memory Regions
  - Driver performs Base TO to physical address mapping
  - ***Driver is responsible for pinning and translation.***
  - Consumer registers mappings with RNIC through:
    - RI registration mechanisms
    - ***Post SQ Fast-Register mechanism***
    - ***STag 0 is a special STag that requires no registration.***
  - QP access to Regions managed through Protection Domains
  - Consumer uses Base TO, RNIC performs Base TO to physical mapping

- Memory Windows
  - Consumer uses Send Queue to "bind" a pre-allocated Window to a specified portion of an existing Region.
  - ***QP access to Windows managed through QP ID.***

Terms Associated with Buffers





- Base Tagged Offset (Base TO) - The offset assigned to the first byte of a Memory Region.

  RNICs support two addressing types:

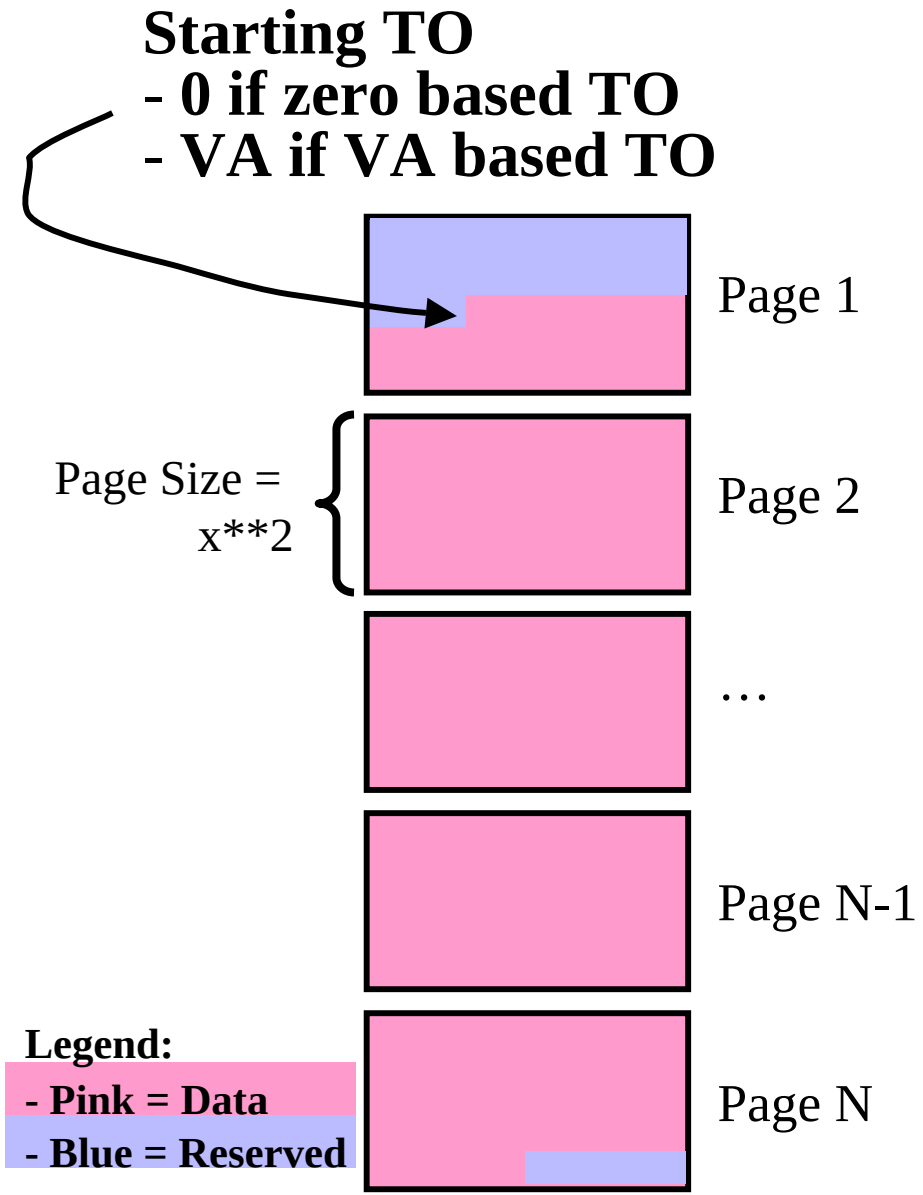
  - Virtual Address Based Tagged Offset (VA Base TO) - The Base TO of an MR or MW that starts at a non-zero TO (and the address is associated with an address of the local host's processor).

  - Zero Based Tagged Offset (Zero Base TO) - A Region or Window that starts with a TO of zero.
    - Zero Based TO – Created to meet the needs of Consumers that don't want to exchange addresses in their request/respond handshakes.
      - For example, iSCSI Extensions for RDMA don't exchange a TO in the iSCSI Command or iSCSI Response. For iSER, the TO is only used in the RDMA Write and Read operations (more will be said later on iSER).

Terms Associated with Buffers… Cont.

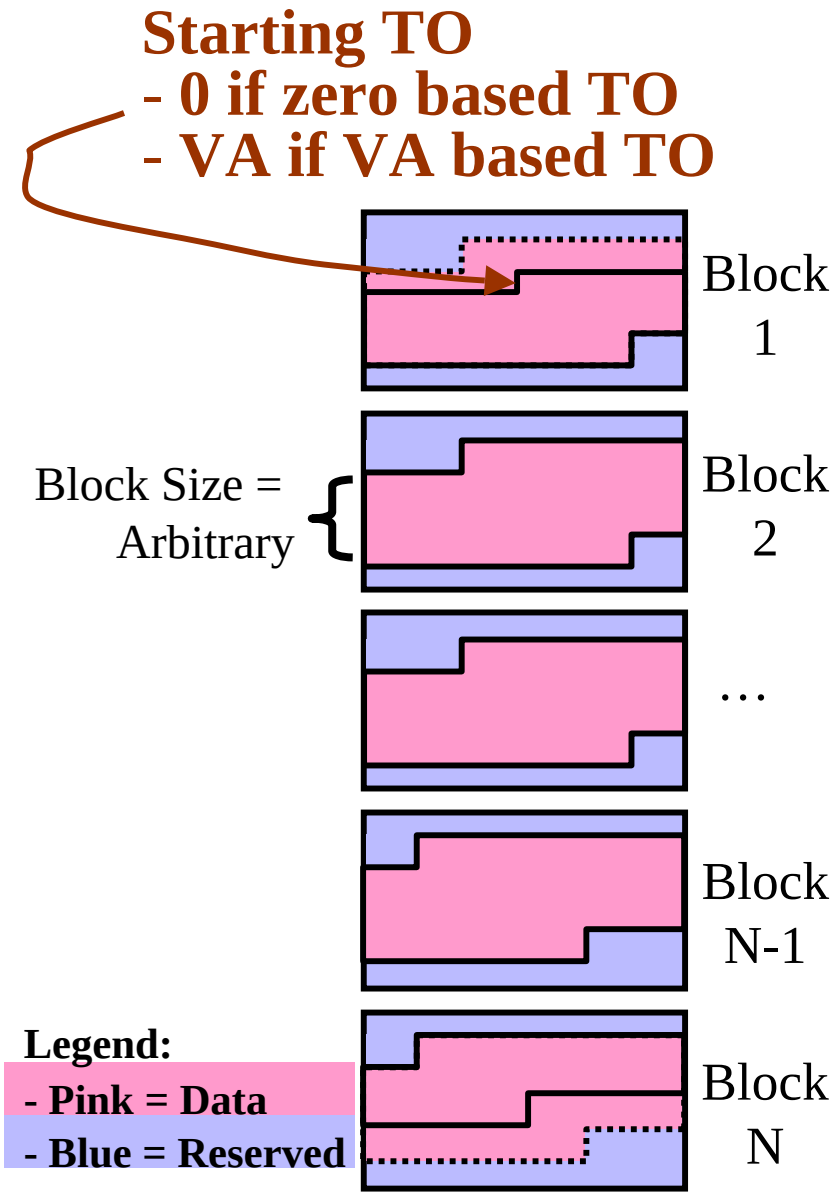
- Physical Buffer - A set of physically contiguous memory locations that can be directly accessed by the RNIC through Physical Addresses.
  - Physical Buffer List (PBL) - A list of Physical Buffers.
    - The input modifier to register Non-Shared Memory Regions.
      - Each Physical Buffer List Entry references a physical buffer.
  - RNICs support two physical buffer list types:
    - Page List - Created to support consumers that have page aligned physical buffers.
    - Block List - Created to support consumers (e.g. storage) that have control information before and/or after each block on the list and the control information is not transferred.
- First Byte Offset (FBO) - The offset into the first Physical Buffer of a Memory Region.
- Scatter/Gather List (SGL)
  - The input modifier to Post Send and Post Receive Verbs which indicate the data buffer location information.
    - Each Scatter/Gather List Entry references an existing Memory Region.

Physical Buffer List: Page List



- Page list attributes:
  - Page size:
    - Size is an integral power of 2
    - All pages have the same size
  - Data boundaries:
    - Data can start at an offset into the first page (First Byte Offset)
    - Data can end on a non-page boundary (i.e. last page may be partially filled)
  - Page (starting) addresses:
    - Must be integral number of page size.
    - Doesn't have to be contiguous.
- Page list modifiers:
  - Page size
  - FBO
  - Length
    - From FBO to last byte of data in the last page.
  - Address list

Physical Buffer List: Block List



- Block list attributes:
  - Block size:
    - Arbitrary (depends on sizes supported by RNIC)
    - All pages have the same size
  - Data boundaries:
    - Data can start at an offset into the first block (First Byte Offset)
    - Data can end at an offset into the last block (i.e. last block may be partially filled)
  - Block (starting) addresses:
    - Arbitrary.
- Page list modifiers:
  - Block size
  - FBO
  - Length
    - From FBO to last byte of data in the last block.
  - Address list

Memory Management

- Memory Region (MR) - An area of memory that the Consumer wants the RNIC to be able to (locally or locally and remotely) access directly in a logically contiguous fashion.
  - A Memory Region is identified by an STag, a Base TO, and a length.
  - A Memory Region is associated with a Physical Buffer List through the STag.

- Two types of MRs:

  - Non-Shared Memory Region - A Memory Region that solely owns the Physical Buffer List associated with the Memory Region. Specifically, the PBL is not shared and has never been shared with another Memory Region.

  - Shared Memory Region - An MR that currently shares, or at one time shared, the Physical Buffer List associated with the Memory Region. Specifically, the PBL is currently shared or was previously shared with another Memory Region.

Memory and Work Request Related Terms

- Problem addressed by Fast-Register:
  - For privileged consumer, compared to I/O transaction models over PCI, the synchronous RI-Registration mechanism is inefficient:
    - Hosts wastes CPU resources to perform registration
      - RNIC driver issues PIO Writes to create the MR.
    - Latency incurred waiting for RNIC to confirm the MR has been registered.
      - Specially if the RNIC is attached through a PCI bus.

- More efficient mechanism is needed
  - Allocate STag - A mechanism used to allocate memory registration resources for use in subsequent Fast-Registrations or RI-Reregistrations.
    - Resources that get allocated are:
      - Protection Domain, PBL, and Remote Access Enablement
  - Fast-Register - A mechanism used to register a Memory Region through the Send Queue.
    - The Fast Register Work Request uses a Memory Region STag, that is in the Invalid state, to register the PBL and access controls passed in through Post Send Queue verb.

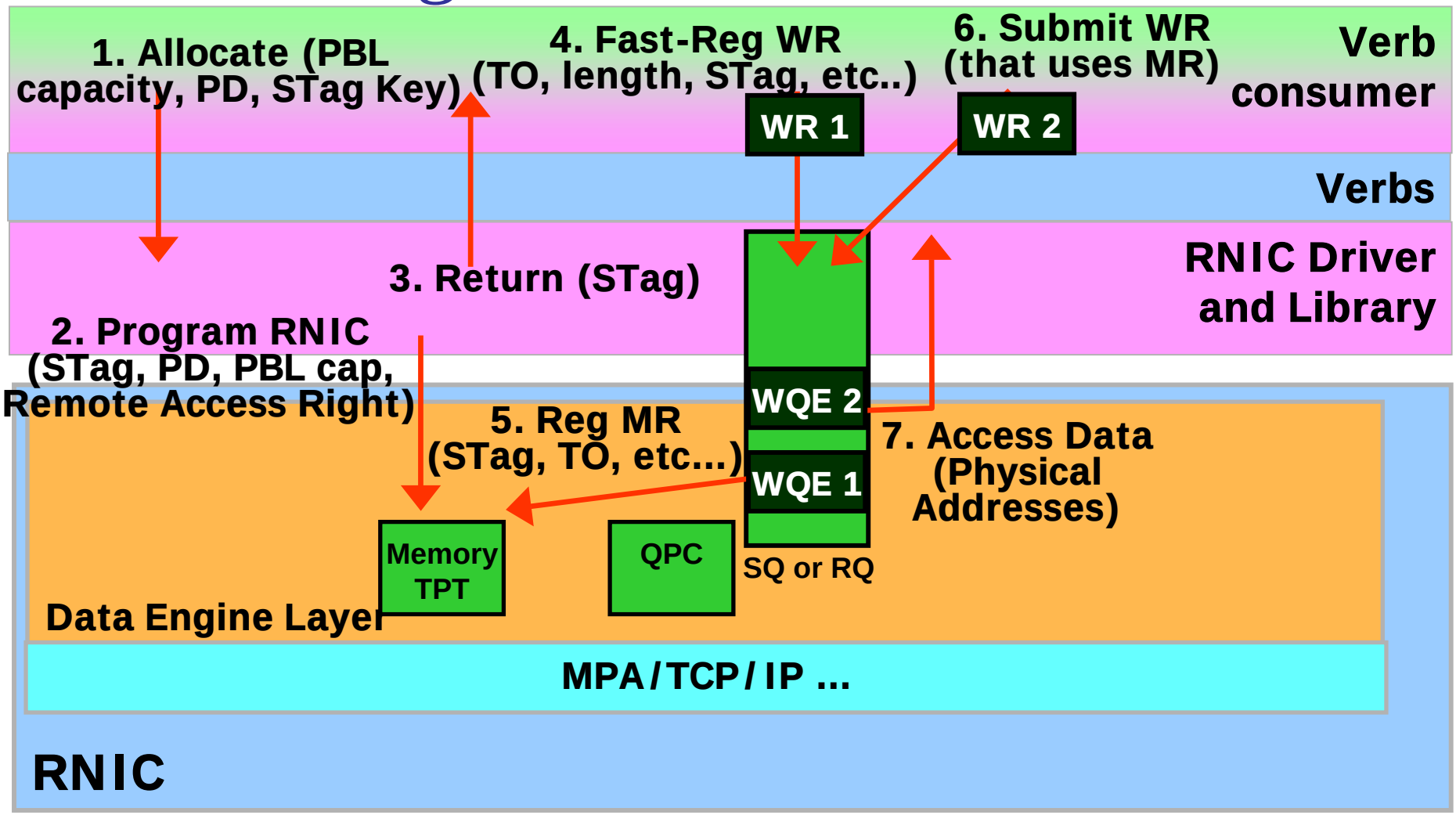
Fast-Register Consumer Usage Options

- At initialization (and possibly, at other intervals), consumer invokes Allocates STag with several size classes.
  - For example:
    - Large Size (65536 PhyPages)
    - Medium-Large (1024 PhyPages)
    - Medium (64 PhyPages)
    - Medium-Small (16 PhyPages)
    - Small (1 Physical Page)

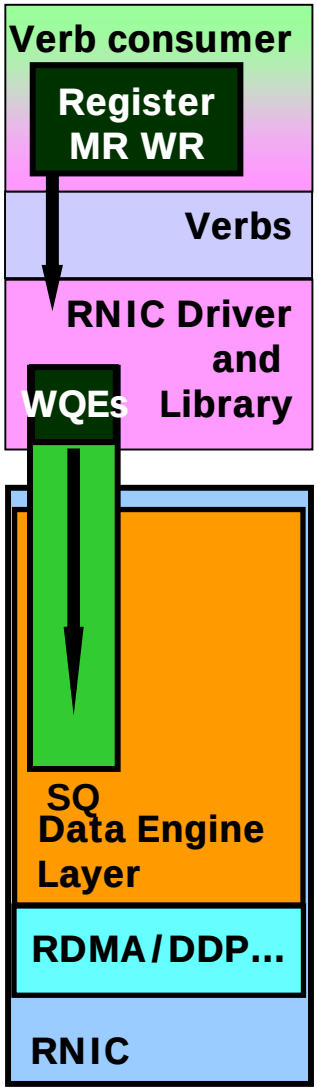
- At run-time, consumer chooses a previously allocated STag that has a size class which is higher or equal to the size of the Memory Region that is to be registered.

However, Verbs spec is silent on the above (it is up to the consumer to decide how to use the allocation).

Animated Introductory Slide Depicting 10,000 Foot Overview of Memory Region Fast-Registration and Local Access

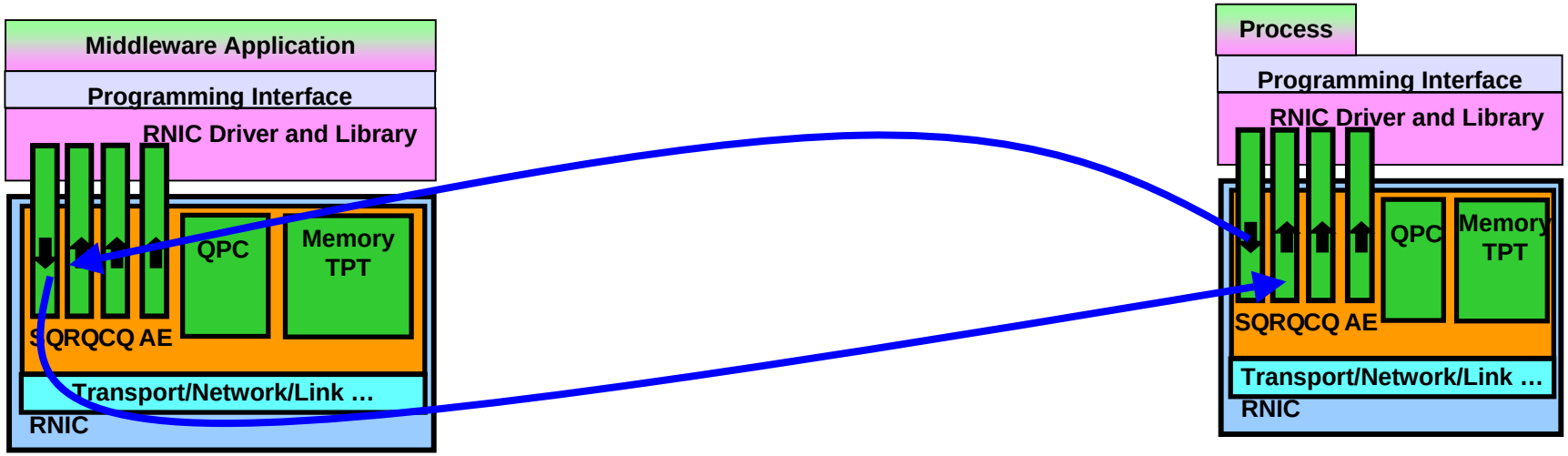


PostSend Verb WR – Register MR



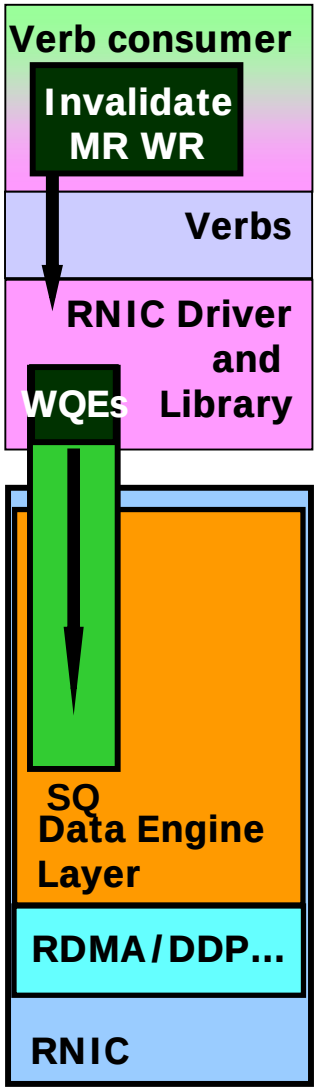
- Privileged mode operation.
- WR Type = Register MR
  - Function:
    - Registers a Memory Region using an existing Non-Shared MR STag.
    - If total # of pages <= Allocated STag PAT size and QP PD matches PD associated with the STag, then RNIC uses the Allocated STag to Register the MR (i.e. fills out Memory TPT entry). Otherwise it returns an error through the CQ.
  - Key Input Modifiers:
    - Allocated STag.
    - Addressing type
      - VA if VA based TO
    - Access rights:
      - Local Read, Local Write, Remote Read, and Remote Write
      - Bind Enabled
    - Physical Buffer List.
      - FBO
    - Length of Region to be registered in bytes and offset into first page.
  - Key Completion Results returned through CQE:
    - Invalid STag, Invalid PD, Invalid MR Length, Access Right Violation

Invalidate STag Rationale



- Problem with RI de-allocate semantics:
  - Some consumers (esp. privileged mode storage) use MRs dynamically.
    - For these consumers the MR is created, used once, and never used again.
  - RI MR de/re-registration is a synchronous, inefficient operation that incurs:
    - Host CPU resources (e.g. memory stores and PIO Writes), and latency associated with waiting for RNIC to confirm the MR has been deregistered.
- RNIC Local and Remote Invalidate semantics:
  - Assuming all STag checks pass, an incoming Send with Invalidate or Send with SE and Invalidate, invalidates access to an MR or MW.
  - An Invalidate STag Post SQ WR invalidates a local MR or MW through an asynchronous process.

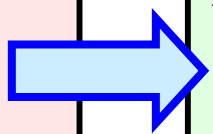
PostSend Verb WR – Invalidate STag



- WR Type = Invalidate STag
  - Function:
    - Invalidates a Local Memory Region.
    - For an STag that is associated with an MR:
      - If QP PD matches PD associated with the STag, then the RNIC disables access to the MR referenced by the STag.
    - For an STag that is associated with an MW:
      - If QP ID matches the QP ID associated with the STag, then the RNIC disables access to the MW referenced by the STag.
  - Key Input Modifiers:
    - Invalidate STag is the WR type
    - STag for the MR or MW
  - Key Completion Results returned through CQE:
    - Invalid STag
    - Invalid PD
    - MR still has MWs bound to it

Overview of MR/MW Attributes

Attribute	Region	Window
Referenced by	STag Tagged Offset Length	STag Tagged Offset Length
Access control checks	PD Access Rights Base & bounds	PD on Bind QP ID on Access Access Rights Base & bounds
PBL Types	Page Block	
Addressing Types	VA Based Zero Based STag zero	VA Based Zero Based
Access granularity	Byte	Byte



Underlying Memory Region TO base	Memory Window TO base	Valid
Zero based	Zero based	No
Zero based	VA based	No
VA based	Zero based	Yes
VA based	VA based	Yes

Summary

- The RNIC Verbs Specification
  - Provides a rich set of semantics that meet the needs of several application environments:
    - General networking,
    - Storage networking, and
    - Cluster networking.
  - Enables integration with existing RDMA APIs/KPIs, and provides extensions to those APIs/KPIs that improve performance.
  - Provides NIC Vendors with implementation flexibility, for:
    - Queue management and memory management structures, as well as, integration with other NIC layer 3 and 4 offload functions.