

---

# Design and Implementation of MPI on Portals 3.0

**Ron Brightwell and Rolf Riesen**

**Sandia National Labs**

**Scalable Computing Systems Department**

**[{bright,rolf}@cs.sandia.gov](mailto:{bright,rolf}@cs.sandia.gov)**

**Arthur B. Maccabe**

**University of New Mexico**

**Computer Science Department**

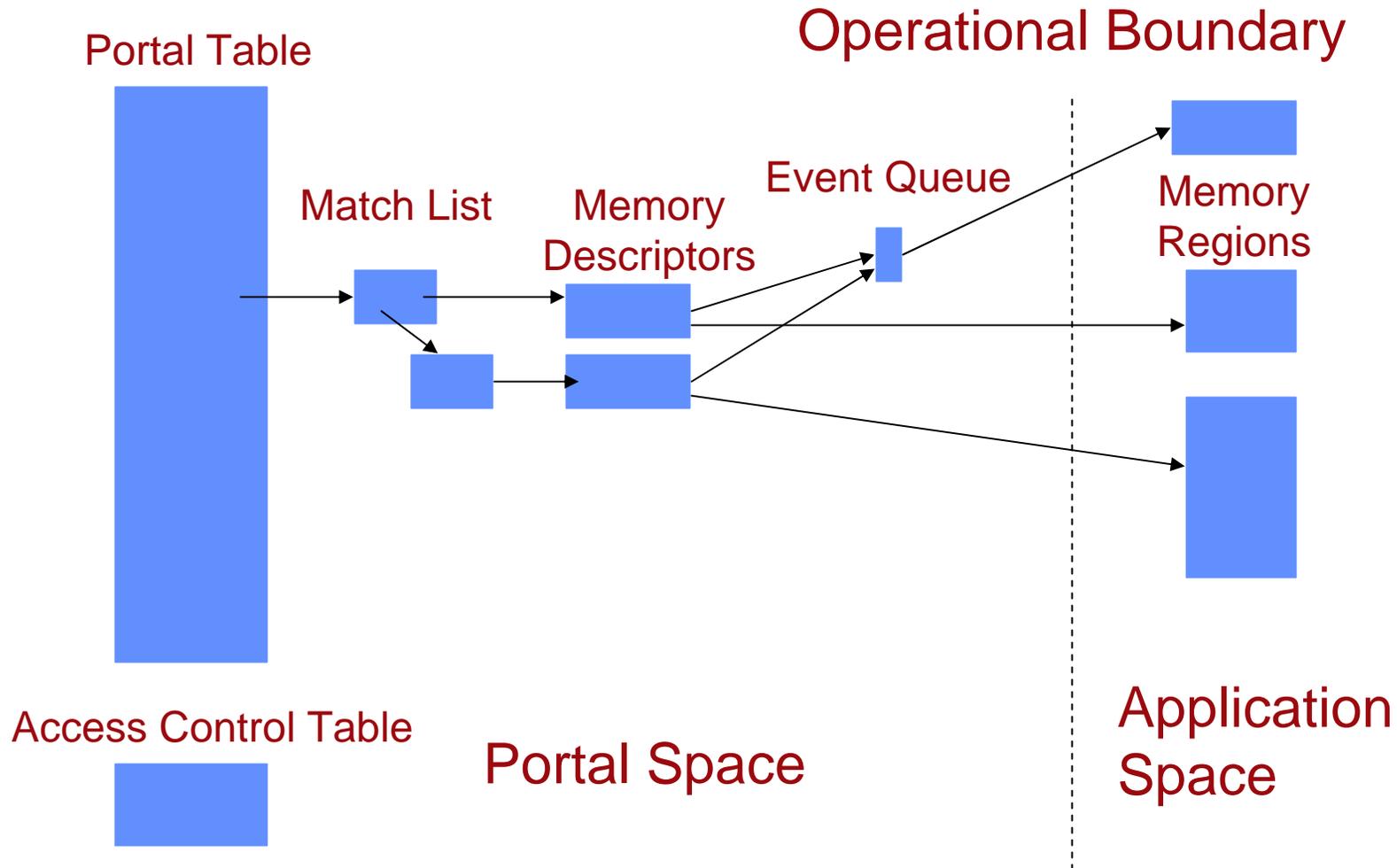
**[maccabe@cs.unm.edu](mailto:maccabe@cs.unm.edu)**

## Portals 3.0

---

- **New API for functionality of Portals 2.0 on ASCI/Red**
- **Intended to provide application bypass capabilities for programmable NICs**

# Portals Addressing



# Portals 3.0 Objects

---

- **Match entry criteria:**
  - Process id (nid/pid,gid/rid)
  - Match bits (64 bits)
  - Ignore bits – mask off unimportant match bits
- **Memory descriptor options**
  - Respond to gets
  - Respond to puts
  - Sender or receiver managed
  - Truncate
  - Generate an acknowledgement
  - Unlink when threshold is 0

# MPI Protocols

---

- **Short protocol**
  - Eager send
  - Buffer unexpected messages (messages for which there is no matching posted receive) at the receiver
  - Unexpected messages are copied when the receive is posted
- **Short synchronous protocol**
  - Same as short protocol, but waits for acknowledgment
- **Long protocol**
  - Eager send
  - Buffer unexpected messages in-place at the sender
  - Unexpected messages are pulled from the sender when the receive is posted
- **Ready protocol**
  - Same as short protocol for both short and long messages, but no receive-side buffering

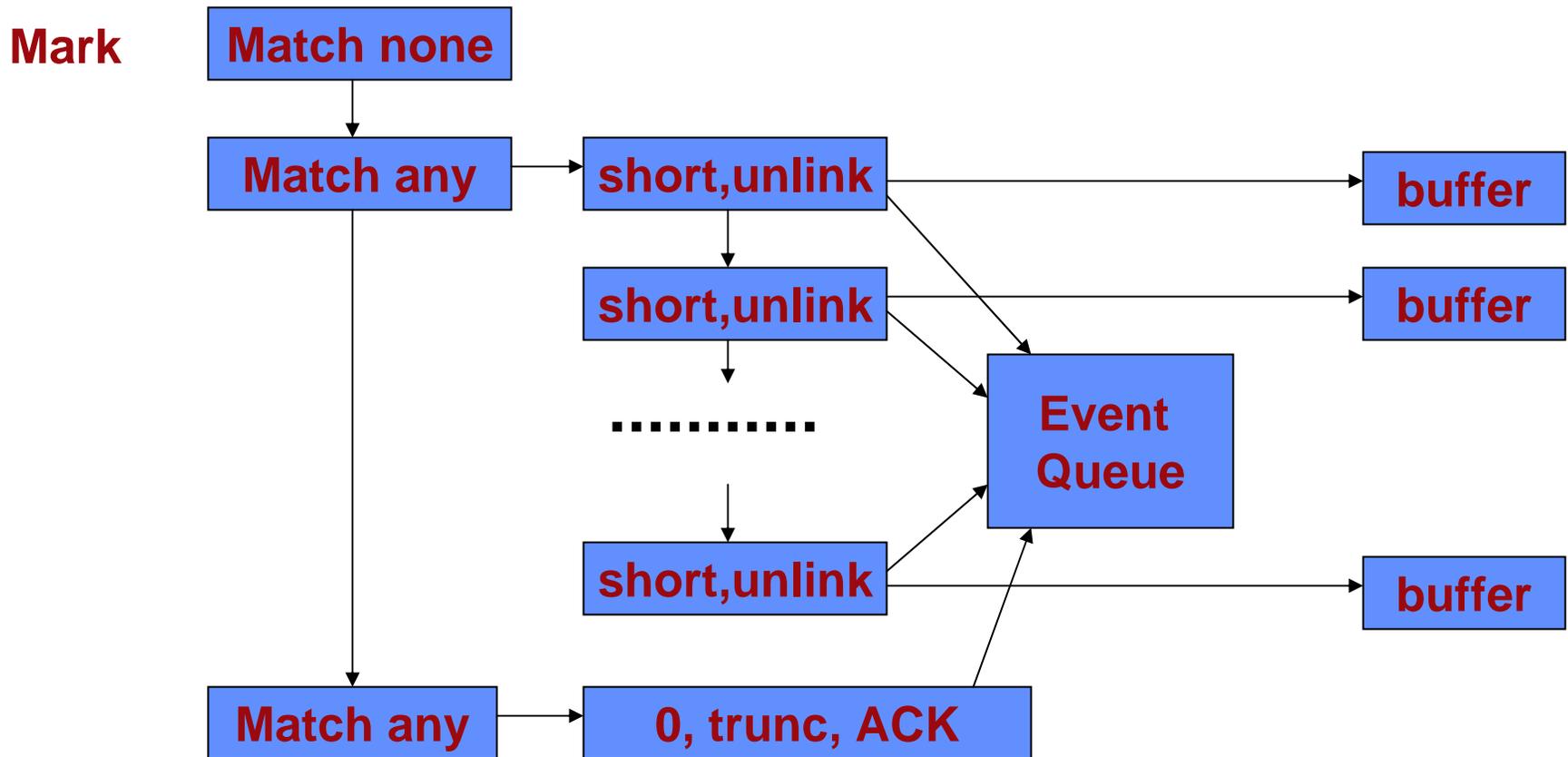
# Flow Control

---

- **Portals flow control**
  - Drop messages that receiver is not prepared for
  - There are no unexpected messages
- **Long eager protocol might waste network resources**
  - Good performance for well-behaved apps
  - Network resources belong to the app
  - Doesn't matter on ASCI/Red, network can handle it
  - Doesn't matter on Cplant<sup>TM</sup>, RMPP takes care of it

# Previous Strategy for Unexpected Messages

Pre-posted

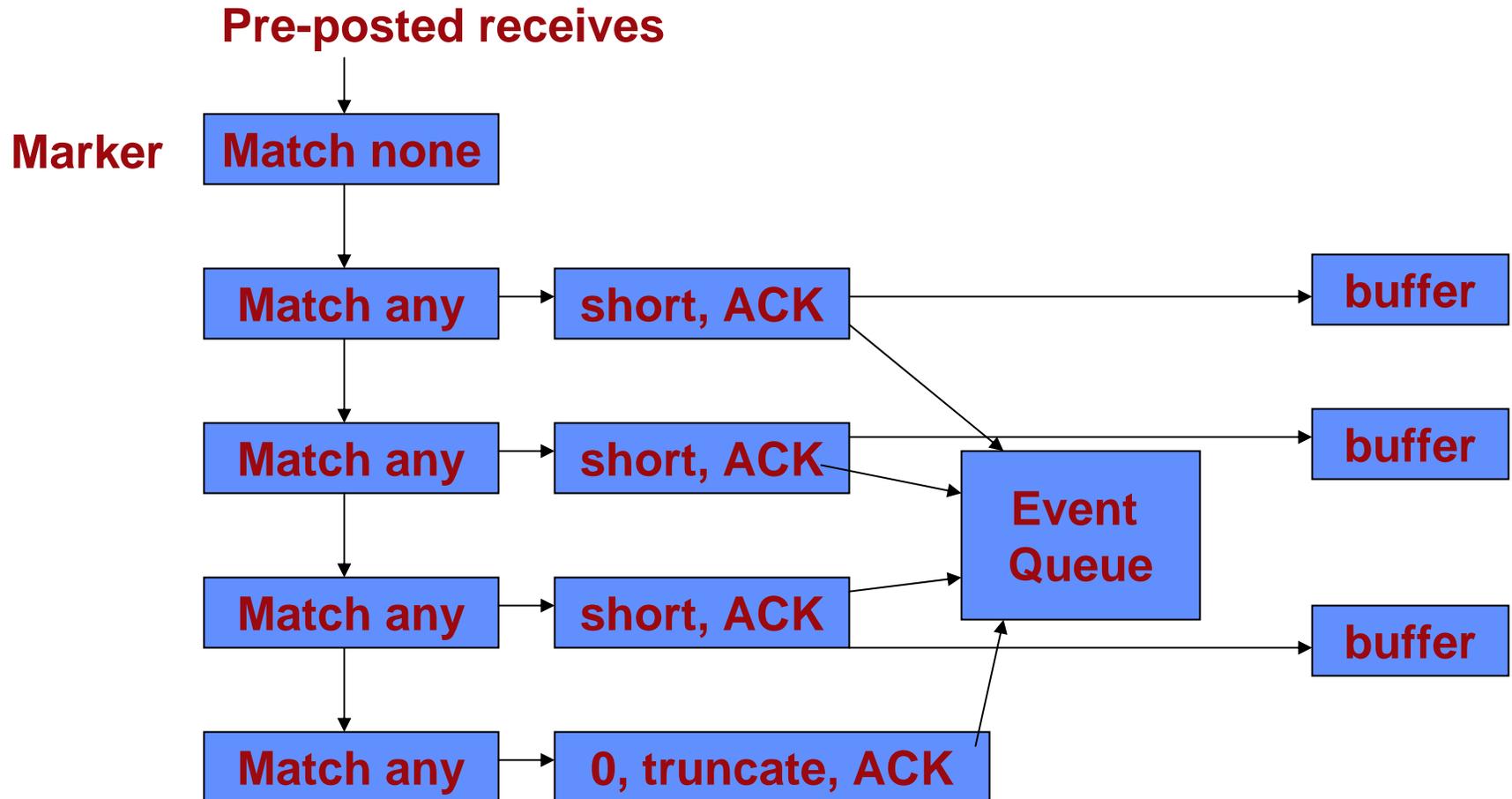


# Limitations

---

- **Limited number of unexpected messages allowed due to kernel memory resources**
- **Any size unexpected message consumes an unexpected message slot, even zero-length**
- **Unexpected message limit based on count rather than size**
- **Consumes a significant amount of Portals resources**
  - 1025 memory descriptors

# Current Strategy



# Advantages

---

- **More efficient use of unexpected message memory**
  - A zero-length message doesn't consume any memory
  - Limitation becomes space rather than count
- **Uses only a few Portals resources**
  - Four memory descriptors versus 1025