Outline

- Announcement(s)
  - send me e-mail about composition of project groups

- Last time: Parallel machines built around commodity processing cores
  - system integration issues
    - memory vs. I/O buses, remote memory access models
  - case studies
    - Cray T3E, SGI Power Challenge, SGI Cray Origin 2000

- This time: Processor-network interfaces for clusters (on the I/O bus)
  - requirements and challenges
  - case studies
    - Myrinet
    - Hamlyn
    - SHRIMP: Yuanyuan
    - Memory Channel: Jian
Processor-Network Interfaces for Clusters

• Current situation (e.g., Ethernet)
  – communication is an OS level service
    • 3 layers: application interface, network protocol, network driver
    • aggressive implementations still incur latency of ~60µs
  – protocols are overly general
    • designed to handle lossy networks, node faults
  – all of the cost is in software!

• Desired situation
  – user-level access to network interface
  – support for multiprogramming (protection)
  – hardware network performance is available to the application
    • low latency, high bandwidth
    • low overhead: communication/computation overlap
    • ideally, should match performance of custom parallel machines

Challenges

1. cost of accessing I/O bus
   0.4µs for read
   0.1µs for write

2. visibility of memory transactions on I/O bus

3. support for multiprogramming (protection)

4. consistency of virtual-to-physical memory mappings

5. notification of message arrival
Case Studies

<table>
<thead>
<tr>
<th></th>
<th>I/O bus access costs</th>
<th>Visibility of memory transactions</th>
<th>Protection</th>
<th>Consistency of virtual-to-physical mappings</th>
<th>Notification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Myrinet</td>
<td>PIO for small amounts of data, DMA otherwise</td>
<td>none</td>
<td>pinned pages</td>
<td>optional</td>
<td>interrupts</td>
</tr>
<tr>
<td>Hamlyn</td>
<td>slots with unique keys</td>
<td>I/O space stores</td>
<td>virtual memory</td>
<td>none</td>
<td></td>
</tr>
<tr>
<td>Memory Channel</td>
<td>snoop memory bus</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRIMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Myrinet Processor-Network Interface

- no protection
- buffer management done at a higher level
Hamlyn

incoming packets

Hamlyn Interface for Sender-based Memory Management

dest ID
slot, metadata indices
protection key
(64 bits)
packet offset
packet length
delta (pkt counting)

metadata area

slot, metadata indices
protection key, msg area/bound

slot index, offset in packet

message areas

Hamlyn: Sending a Message

① processor reclaimssend buffer
② interface signals completion
① Processor writes to terminus queue

completion-of-send pointers

send buffer areas

per-process send termini

63 entries

8 base/bound registers
Hamlyn: Receiving a Message

\[
\text{accum} \leftarrow \text{delta (from packet)}
\]

\[\text{all packets have arrived when } \text{accum} = 0\]

- user metadata (60 bytes)
- packet accumulator

\[
\text{metadata area}
\]

- slots
- offset in packet

\[
\text{NQCB entries}
\]

\[
Y(=2^n) \cdot \text{npackets} \quad \ldots \quad 1 
\]

Hamlyn: Performance and Limitations

- **Prototype:** Myrinet, 100 MHz PA-RISC workstations
  - short message latency: 12.7µs
    - DMA (3.3µs), LANai (6.7µs), switch (0.5µs)
    - (overhead): host I/O writes (1.4µs), protocol software (0.8µs)
  - long message latency: 17.4µs
    - costs for accessing send buffers, notification
  - bandwidth: limited by I/O bridge (32 MB/s)
    - 20 MB/s for 256-byte messages
    - 28 MB/s for 1024-byte messages
- **Limitations**
  - sender-based memory management
  - application level buffer management
  - static resource allocation of
    - metadata structures
    - notification queue
Lecture Summary

• I/O-bus based processor-network interfaces
  – Myrinet, Hamlyn, SHRIMP, Memory Channel
  – common features
    • user-level access to network + support for multiprogramming
    • good performance
      – ~10µs latency, close to peak bandwidth (limiting factor is I/O bridge)
      – overhead is a few instructions
  • However, communication semantics are very low-level
    – sender has to be aware of message’s final destination
      • higher level (software) responsible for
        – N-to-1 merge
        – buffer management and reclamation
      – separation of data transfer from notification
  • Need for software layer(s) which provide more useful semantics

Next Lecture

• Low-level messaging layers
  – Fast Messages: Efficient Portable Communication for Workstation Clusters and MPPs, Pakin et al.
  – Effects of Communication Latency, Overhead, and Bandwidth in a Cluster Architecture, Martin et al.