High-Performance Message Passing over generic Ethernet with Open-MX

Brice Goglin - 2009/11/19
Heard about convergence?

- Storage and networking already converging (FCoE, DCB, ...)
  - What about HPC?
- Will InfiniBand be the converged technology?
  - “IB won’t win because it’s not Ethernet”
- High-speed networking works over Ethernet too
  - Myricom did MXoE 3 years ago
  - Mellanox pushing RDMAoE
HPC over Ethernet, really?

- Performance problems are in the stack, not in the fabric
  - TCP over IB isn’t better than TCP over Ethernet
  - HPC over Ethernet needs the right stack
    - *aka* not TCP
- How about a HPC stack over Ethernet?
  - Look at latency, throughput, overlap, message-rate
    - Not at retransmission or congestion control
What about existing stacks?

- iWarp, RDMAoE, MXoE, ...
  - No need to spend money in expensive advanced NICs
- GAMMA?
  - I don’t want to modify the network stack
  - I don’t want to break IP drivers
Why Open-MX?

- Need support for any Ethernet hardware
- Need to keep existing stacks/drivers unmodified
  - Can coexist with IP
  - No need to patch the kernel
- Design the stack for modern hardware
  - 10G boards, ...
What’s Open-MX?

• Yet another custom stack with a custom API?
• Yet another custom MPI implementation with limited features, poor stability, ...?

• No, Open-MX is MX API/ABI compatible with MX
  • and even wire compatible
• Native support from many existing MPIs
Software stack summary

User
- Sockets

Kernel
- TCP, UDP, ...
- Open-MX Driver
- Ethernet

Hardware
- Ethernet NIC
- MX NIC
- MX Wire Protocol

MX API
- MX Lib.
- Open-MX Lib.
- MX Wire Protocol
- OS-Bypass
How do I use it?

• Build and install as usual
• Run startup script
  • Loads a Linux kernel module
  • Automatic discovery of the fabric
• It works!
• Run MPI jobs as usual with Myricom’s MX
What about OS-bypass?

- « I need OS-bypass for low latency »
- Wake up! We’re not in the 90s anymore!
  - A syscall is less than 100ns today
- Going through the OS brings some advantages
  - Resource sharing
  - Security
  - Less work in the (slow) NIC
What about zero-copy?

• Easy on the send side

• Much harder on the receive side
  • The NIC+driver decides where packets are received
    • No way to receive directly in the application buffer
  • This is where RDMA-enabled NICs are different
    • (and expensive)

➢ Open-MX has to copy once on the receive side
Efficient non-zero-copy receive stack

• Memory copies are bad?
  • Depends on the actual network throughput
    • Doesn’t matter for 1G
  • Depends on the host performance
    • Nehalem is much faster than a 10G network
• Memory copies may be offloaded on I/OAT hardware
  ➢ Overlapped offloaded receive copy for large messages
What performance may I expect?

• 10G line-rate
• Up to 5µs with high-end NICs
  – 10-15µs with regular 1 Gigabit/s NICs
Stateless offload

• Open-MX doesn’t need advanced NICs
• But may benefit from stateless offloaded features
  • Multiqueue support
  • Open-MX-aware interrupt coalescing

• Easy features give huge performance improvements
  • Much more cost-efficient than RDMA or TOE
Summary

• Works with all Ethernet hardware
• Works with all Linux kernels
• Low latency (up to 5µs)
• High throughput (10G linerate)
• Successfully runs with OpenMPI, MPICH2, Platform MPI, Intel MPI, PVFS2, and more
Thank you for your attention!


INRIA Booth #1405