Exploiting the ExaNeSt Communication Primitives for a High Performance MPI Library

A. Psistakis, M. Asiminakis, P. Xirouchakis, M. Gianioudis, P. Peristerakis, F. Chaix, M. Ploumidis, V. Papaefstathiou, N. Chrysos, M. Katevenis

Foundation for Research and Technology - Hellas (FORTH)
Goals

• Functionality contributed by FORTH within ExaNeSt
  • Communication primitives
    • Hw + sw
• Functionality employed by ExaNeSt, contributed by ExaNoDe
• Demonstrate above functionality
  • MPI library
    • Partial implementation of the MPI standard
    • Contributed by FORTH
Overview

• Prototype employed
• Transfers between local and remote virtual addresses
• User-level zero-copy RDMA
  • Virtual addresses
• User-level low-latency atomic message delivery
• UnimemMPI
Overview

• Prototype employed
  • Transfers between local and remote virtual addresses
  • User-level zero-copy RDMA
    • Virtual addresses
  • User-level low-latency atomic message delivery
  • UnimemMPI
Development Prototype
Overview

• Prototype employed

• Transfers between local and remote virtual addresses

• User-level zero-copy RDMA
  • Virtual addresses

• User-level low-latency atomic message delivery

• UnimemMPI
Transfers between local and remote VAs
Using remote virtual addresses

- Access to remote virtual addresses
  - Unimem architecture
- Virtual address in Xilinx Linux Kernel = 39 bits
- Routing data add to remote VAs = 8 bits
- Restriction (due to Xilinx Kernel, not Unimem)
  - $\forall$ Process Virtual address $\in [0, 2^{31} - 1]$
- Workaround
  - Patch mmap() system call
  - Addresses in stack
    - Mackecontext, swapcontext, getcontext
Exploiting System MMU 1/2

• Configuration and programming of system MMU (a.k.a IOMMU)
  • zDMA, PL
• 16 contexts banks
• Kernel module patch
  • Context: process page table
• Local and remote virtual address translation
SMMU: protected access to remote VAs

- Processes of the same application
  - Same protection domain
    - Protection domain ID
- Sender side:
  - FORTH’s NI at Programmable logic (PL)
    - Add PDID information to transaction
- Receiver side:
  - FORTH’s NI: extract PDID
  - PDID -> stream ID
  - Stream ID -> SMMU context bank
  - Context bank = translation context (process page table)
Overview

• Prototype employed
• Transfers between local and remote virtual addresses
• **User-level zero-copy RDMA**
  • Virtual addresses
• User-level low-latency atomic message delivery
• UnimemMPI
User level zero copy RDMA transfers

• Employ low power domain zDMA in PS
  • Cache coherent memory accesses
  • Eight independent channels

• Kernel module by FORTH
  • Each process: exclusive access to single zDMA channel
  • Associate PDID to zDMA channel
  • Control of zDMA from user space
    • Kernel space involved only at init time
    • Not involved in actual transfers
      • Zero-copy transfers

• Virtual addresses
  • No memory pinning
Overview

• Prototype employed
• Transfers between local and remote virtual addresses
• User-level zero-copy RDMA
  • Virtual addresses
• User-level low-latency atomic message delivery
• UnimemMPI
User-level low-latency atomic message delivery
1/2

• FORTH’s contribution in ExaNoDe
• Two hardware blocks
  • Virtualized mailbox
  • Virtualized packetizer
• Kernel modules
• User-space library
  • Expose to user-space code
• Virtualized mailbox
  • 256bits messages, 64 interfaces
• Virtualized packetizer
  • 256 bits messages, 64 interfaces, exploit AXI Burst capability
User-level low-latency atomic message delivery 2/2

• Kernel module
  • Configure virtualized packetizer/mailbox
  • Process/thread: attach one/64 virtual packetizer/mailbox

• User-level atomic message send/recv
  • Packetizer 256 bits message
  • Destined to remote process’s mailbox
Overview

• Prototype employed
• Transfers between local and remote virtual addresses
• User-level zero-copy RDMA
  • Virtual addresses
• User-level low-latency atomic message delivery
• UnimemMPI
MPI libraries in ExaNeSt

- MPI From BSC
  - MoU with ExaNoDe

- UnimemMPI
  - Contributed by FORTH
  - Debug/optimize performance
    - Software libraries
    - HW
  - Preliminary performance results
    - Other than micro-benchmarks

Manolis Ploumidis (FORTH), HiPEAC (Jan 2018, Manchester)
UnimemMPI overview 1/2

• Partial implementation of the MPI standard
  • Almost all point-to-point primitives
  • Collectives
    • Delegated to slightly modified MPICH library
• Point-to-point related functionality under-testing
  • Traffic through derived data types
    • Copy....
  • Persistent requests
  • MPI_Cancel, MPI_Mprobe, MPI_Improbe, MPI_Mrecv, MPI_Imrecv
UnimemMPI overview 2/2

MPI application

yes

Is pt2pt?

no

UnimemMPI

Unimem library: zDMA, Vpacketizer, VMbox

Mpch-3.2 (modified)

User Space: application

User Space: libraries

Kernel Space

Kernel

TCP
IP
MAC

Vmox k
Vpktzr k

DMA Channel m

Virtualized Mbox
Virtualized Pktzr

zDMA
SMMU
Eth

PS
PL

HW

Manolis Ploumidis (FORTH), HiPEAC (Jan 2018, Manchester)
UnimemMPI messaging protocol

Sender

msg envelope
Virtual Pcktizr N:
@Virt MboxN

RDMA read:
Prefix::Buf_S → Buf_R

ack
Virtual Pcktizr M:
@Virt MboxM

Receiver

Poll
VirtMboxN

Manolis Ploumidis (FORTH), HiPEAC (Jan 2018, Manchester)
UnimemMPI preliminary results 1/2

• Micro-benchmarks
  • Packetizer to remote mailbox message ~= 2 usec
  • zDMA-read (8 bytes) ~= 3 usec
  • zDMA-read (4096 bytes) ~= 18.3 usec

• MPI ping-pong test
  • UnimemMPI ping-pong = 4 mailbox messages + 2 zDMA-read ops

<table>
<thead>
<tr>
<th>Ping, pong msg size</th>
<th>UnimemMPI</th>
<th>MPICH(TCP sockets)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 bytes</td>
<td>12.5 usec</td>
<td>279.6 usec</td>
</tr>
<tr>
<td>4096</td>
<td>46.0 usec</td>
<td>1207.1 usec</td>
</tr>
</tbody>
</table>
UnimemMPI preliminary results 2/2

- Preliminary results
  - LAMMPS application
    - State of the art molecular dynamics code
    - Cooperation with eXact lab
    - Rhodopsin problem
    - Three OMP threads per run

<table>
<thead>
<tr>
<th>Num of nodes</th>
<th>Timesteps</th>
<th>Unimem Wall Timesteps/s</th>
<th>MPICH (TCP sockets) Timesteps/s</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>600</td>
<td>1.599</td>
<td>1.542</td>
<td>3.6%</td>
</tr>
<tr>
<td>3</td>
<td>900</td>
<td>2.103</td>
<td>1.982</td>
<td>6.1%</td>
</tr>
<tr>
<td>4</td>
<td>1200</td>
<td>2.575</td>
<td>2.409</td>
<td>6.9%</td>
</tr>
<tr>
<td>6</td>
<td>1800</td>
<td>3.205</td>
<td>2.889</td>
<td>10.9%</td>
</tr>
</tbody>
</table>
Ongoing and future work

• Ongoing
  • Move from AXI-based prototype to ExaNet
  • Support missing pt2pt primitives
  • Allow remote page faults
  • Employ new mailbox
    • Larger messages, higher performance

• Future work
  • MPI messaging protocol = rdma-write based