Distributing OS Functionality to Enhance Application Performance

March 2002

Arthur B. Maccabe
maccabe@cs.unm.edu

Scalable Systems Lab
Computer Science Department
The University of New Mexico
Where you place functionality matters

- OS Bypass in MPI Implementations
- OS Offload, a different perspective
- Application Offload
- Double buffering benchmark
- Application Bypass benchmark
Basic MPI Stack

- Memory copies
  - within the application
  - in kernel stack
  - between NIC and kernel
- Latency
- Interrupt pressure
- Overhead
OS Bypass

- Memory copies
  - between NIC and application
  - within application

- Latency

- Interrupt pressure

- Overhead
Different Perspective

You never really bypass the OS

A bit of the OS goes onto the NIC
Why not offload part of the Application?

Just enough to decide where to put messages

matching
Double Buffering

- Overlap communication with processing
- Latency hiding
Double Buffering

Producer

\[
\text{for( i = 0 ; i < n-1 ; i++ ) } \\
\{ \\
\text{fill A; wait CTS A; } \\
\text{isend A; } \\
\text{fill B; wait CTS B; } \\
\text{isend B; } \\
\} 
\]

Consumer

\[
\text{ireceive A; isend CTS A; } \\
\text{ireceive B; isend CTS B; } \\
\text{for( i = 0 ; i < n ; i++ ) } \\
\{ \\
\text{wait A; sum A; } \\
\text{ireceive A; isend CTS A; } \\
\text{wait B; sum B; } \\
\text{ireceive B; isend CTS B; } \\
\} 
\]
Double Buffering Performance
Why is Portals Better?

Long message protocol

- RTS
- MPI Match
- CTS
- Data
- Data
- Data
- Requires Application Intervention
- Handled by OS Bypass

Distributing OS Functionality to Enhance Application Performance – p.10/18
It’s even better than it looks! Portals Bandwidth

poll interval: 10000

- 'MPICH-GM' using 1:3
- 'Portals' using 1:3
Why is Bandwidth so Bad?

The current implementation uses the kernel for everything
Portals on the NIC

![Graph showing time vs message size](image)
Post-Work-Wait

```
ireceive();  delay (work)  wait();
```

work with MH  wait time

total time

**Time as a Function of Work Interval (MPICH 1.2.4/GM 1.4, 310KB Message)**

**Time as a Function of Work Interval (Portals 3.0, 310KB Message)**
Portals Implementation Strategy

Application

Portals Library

local structures

remote forward

remote forward

OS

NIC
Placing OS Functionality

- **Host processor**
  - Supervisor mode
  - User mode

- **Co-processor**
  - Compute co-processor (threads)
  - Message co-processor (NIC)

- **Server node**
  - File server
  - TCP (socket) server
Conclusions

- It’s easy to design low-level protocols, the trick is effectively supporting higher level protocols.
- Latency hiding is critical for applications.
- Placement of functionality matters.
  - liberating perspective
  - many other opportunities.
Acknowledgements

Supported by Sandia National Labs and CSRI

- Bill Lawry and Riley Wilson
- Ron Brightwell and Rolf Riesen
- Patricia Gilfeather, Edgar Leon, Dennis Lucero, Carl Sylvia, and Wenbin Zhu