Summary

• User level network interface works and performs well!
  —  supports legacy protocols (TCP/IP) as well as novel ones (UNAM).
  —  need better resource management.

• UNAM adds very little overhead.
  —  need to investigate flow control further.

• Split-C performance comparable to MPPs like CM-5 and Meiko CS-2.
  —  need to make it robust enough to be usable by others.

• TCP/IP performance order of magnitude better than traditional architectures.
  —  need to figure out what user-level IP means.
TCP/IP: Performance over U-Net

- **TCP round trip latency:**
  U-Net: 200 - 300μs vs Fore ATM: 1200 - 1400μs — order of magnitude better.

- **TCP bandwidth:**
  — U-Net: full 15.2 Mbytes/sec for 2Kbyte message size with 8Kbyte window.
  — Fore ATM: 7Mbytes/sec for 8Kbyte message size with 64Kbyte window.
Split-C: Application benchmarks

- executions on 8 processors, normalized to the CM-5.
- ATM cluster beats Meiko CS-2 on CPU but is slightly worse in bandwidth.
- ATM cluster and Meiko CS-2 better than CM-5 in bulk message case.
- ATM cluster worse than CM-5 in the small message case.
Split-C: the Model and Implementation

- Split-C programs consist of:
  - one thread/processor.
  - global pointers.
  - communication among threads via shared data.

- implementation:
  - dereferencing global pointers generate appropriate active message calls.
UNAM: Micro-benchmarks

- round trip time for small messages: 66µs.
- maximum bandwidth achieved: 13.6 MB/sec using 3Kbyte messages.
- working on getting full bandwidth.
UNAM: the Flow Control

- window based flow control.
- four types of messages: request, reply, ack and nack.
- acknowledgements piggybacked whenever possible.
UNAM: Implementing GAM over U-Net

- user-level library, implements the GAM 1.1 specification from Berkeley.
- per channel request and reply queues.
- flow control and reliable message delivery using a windowing protocol.
GAM: the Architecture

- assumes SPMD model.
- small messages — upto 4 words of data + handler.
  bulk xfers — store and get.
- request and reply messages distinct.
- best-effort delivery.
U-Net: Memory Requirements

- currently: fixed number of pages pinned to physical memory for all endpoints.
- pages for communication segments and queues all mapped to DMA space.
- size of DMA space limits number of distinct applications running at one time.
- how to get around this?
• roundtrip time: 60µs across the switch.

• full bandwidth (~15.2MB/sec) achieved for PDU sizes as low as 1Kbytes.
U-Net: The Implementation

- send and free buffer queues on i960, receive queue on host.
- all queues and communication segment mapped to DMA-space.
- DMA done using the “fly-by” feature.
- optimized for single cell messages.
• push pointers to transmit data buffers onto send queue (out fifo) for sending.

• receive queue (in fifo) entries contain small msgs or pointers to large msgs.

• connections opened via kernel.

• U-Net checks incoming endpoint addresses and demuxes.
U-Net: The Architecture

- Remove kernel from the critical path.
- Virtualize network using a message mux/demux mechanism.
- Simplify buffering strategies.
- Queue-based send and receive models.
- Concept of endpoint as unit of addressing and protection.
Outline of the talk

• U-Net solution: architecture, implementation and performance.

• UNAM: an active message implementation over U-net.

• Split-C over UNAM.

• Performance of TCP/IP over U-Net

• Summary
Motivation for U-Net

• End-to-end high-bandwidths and low latencies not observed even when network fabric is capable.

• Generalized buffering strategies in kernel (e.g. mbufs) at the core of problems.

• So what is the solution?
TCP/IP performance on ATMs: a brief look

- no significant improvement in latencies in spite of faster network fabric.

- full bandwidth not achieved even at packet sizes of 8Kbytes.
U-Net and UNAM: bringing parallel and distributed computing closer

Anindya Basu

Joint work with Vineet Buch, Werner Vogels, and Thorsten Von Eicken

http://www.cs.cornell.edu/Info/Projects/ATM

April 12, 1995