

Limits to Low-Latency Communication on High-Speed Networks

Chandramohan Thekkath and
Henry Levy, UW

Overview

- Understand performance of basic packet exchange in different scenarios
 - Three different network types:
 - 10 Mbps Ethernet
 - high bw (100Mbps) token-ring FDDI
 - 140Mbps ATM w/ 53B cells
 - Different network controllers:
 - Ethernet/DEC: 128kB on-board memory, pkt desc.
 - Ethernet/SPARC: DMA in both directions, scatter-gather
 - FDDI/DEC: DMA on receive side, no scatter-gather
 - ATM/DEC: FIFOs, one in each dir
- Understand how much extra an RPC service must do on top of that – what's are limiting factors in delivering low-latency communication services

Contributions

- Structuring low-latency RPC or communication services has to be done in close consideration of underlying networking hardware and protocols used
- Some thoughts on requirements for future NICs
- Experimental results: high bandwidth networks don't always meet low-latency requirements

Baseline Performance

Component	Round-Trip Time (μ seconds)							
	Packet Size in bytes (send/rcv)							
	Ethernet (DEC)		Ethernet (Sparc)		FDDI (DEC)		ATM (DEC)	
	60/60	1514/1514	60/60	1514/1514	60/60	1514/1514	53/53	1537/1537
Time on the Wire	115	2442	115	2442	13	245	6	176
Controller Latency	51	53	89	103	97	230	16	161
Control/Data Transfer	40	600	6	6	40	253	17	458
Vectoring the Interrupt	25	25	12	12	25	25	25	25
Interrupt Service	26	26	42	42	92	140	9	20
Sum of Components	257	3146	264	2605	267	893	73	840
Measured Round Trip Time	253	3137	263	2611	263	894	73	746

Notes:

- For DMA data transfer is in Controller latency; Cache must be flushed – in Interrupt Service;
- For ATM no fragmentation/reassembly costs include (1065usec); also switching overheads not included; pipelining effects not included in Sum of Components but are in Measured RTT;

Observations:

- High bandwidth doesn't translate to low latency for small packets
- For DMA NIC Eth/SPARC – overlap of data transfer over bus and data transfer to network – $89+6 \sim 103+6$
- Use of FIFO simplifies Interrupt Service – fewer Mm accesses compared to packet desc.

Low-Latency RPC design

- Follows basic RPC design
- Optimized for the common case:
 - Single-packet transfer between machines with same byte order on same LAN
 - No transport protocol, directly network packets
 - Reliability at RPC-level... RPC response == ACK
 - Decision can be made at bind time
- Main overheads due to
 - Data copying/marshaling
 - Control transfer
 - Protocol processing
- => specialize design to leverage capabilities of NIC/network

Data copying / Marshaling

- Objective: minimize data copying
- For DMA
 - For scatter-gather copy data from user space, header from kernel
 - Must be able to map user buffer into DMA region
 - If data cannot be placed directly in user space, consider copy vs. mapping trade-offs
 - What about alignment? What about demux? What about limits on segment size or number?
- For PIO
 - “synthesize” system calls in kernel which implement marshalling code
 - Pass data descriptor to kernel, kernel knows how to handle primitive data types
 - Alternative to map packet buffer space / FIFO into userspace – too relaxed protection

Control Transfers

- Defer blocking
 - spin for short period
 - Rely on thresholds, hints or heuristics...

Protocol Processing

- Optimize for destination within same LAN
 - Decision made at bind time
 - Raw network packets
 - Only hardware-supported checksums
 - RPC communication used for ACKs
 - Preallocate and Reuse headers
 - If buffer must be copied
 - overlap copy with network transfer to remove copy costs from critical path
 - copy-on-write to conserve space

Experimental Data

Activity	RPC Time (μ seconds)							
	Ethernet (DEC)		Ethernet (Sparc)		FDDI (DEC)		ATM (DEC)	
	Minus	MaxArg	Minus	Maxarg	Minus	Maxarg	Minus	MaxArg
Client Call	28	145	59	137	46	173	25	159
Controller Latency	26	27	45	54	48	82	8	44
Time on the Wire for Call	58	1221	58	1221	4	122	3	88
Interrupt Handling on Server	25	25	27	27	56	70	17	20
Server Packet Receipt	39	470	59	169	42	42	29	347
Server Reply	27	27	46	46	36	36	25	25
Controller Latency	25	25	44	44	49	82	8	44
Time on the Wire for Reply	57	57	57	57	5	5	3	3
Interrupt Handling on Client	26	26	27	27	56	56	17	17
Client Reply Receipt	29	29	49	49	29	29	29	29
Total Attributed Time	340	2052	471	1831	371	697	164	776
Measured Time	340	2070	496	1997	380	693	170	675

- Diff OS/VMM and hardware for two sets of Ethernet numbers
- Simpler ATM controller good for small packets, bad from large ones because of software segmentation/reassembly (although can be overlapped with transmission)... maybe if in hardware it will be better
- High BW for FDDI good for large data, complex controller bad for small packets
- Synchronous nature of measurements – hard to see benefits of overlap with DMA ops
- Numbers ignore demux overheads

NIC design considerations

- PIO vs. DMA – there will be a breakeven point
- Cache invalidation/coherency issues
 - In paper cache flush – cost increases for larger data
 - Alternative cache coherency for DMA ops
 - Ideally, cache injection
- Direct data placement in correct address space
 - Copy vs. remap
 - Pass memory descriptors in NIC hardware
 - what about demux and header?
 - what about address translation?
- FIFO vs. packet descriptor
 - FIFO simpler, good for small data; hard to share, map...
 - Packet memory also limited, must be managed too
 - Page alignment issues – e.g., in paper to double map each 2k NIC page into each 4k host page
- Network consideration
 - Network datagram size issues – avoid fragmentation/reassembly, interrupts... (paper uses interrupt per x ATM cells to deliver entire RPC data)
- Other factors?