Workloads for Programmable Network Interfaces

Patrick Crowley, Marc Fiuczynski,
Jean-Loup Baer, and Brian Bershad

University of Washington
pcrowley@cs.washington.edu

WWC '99 Austin, TX

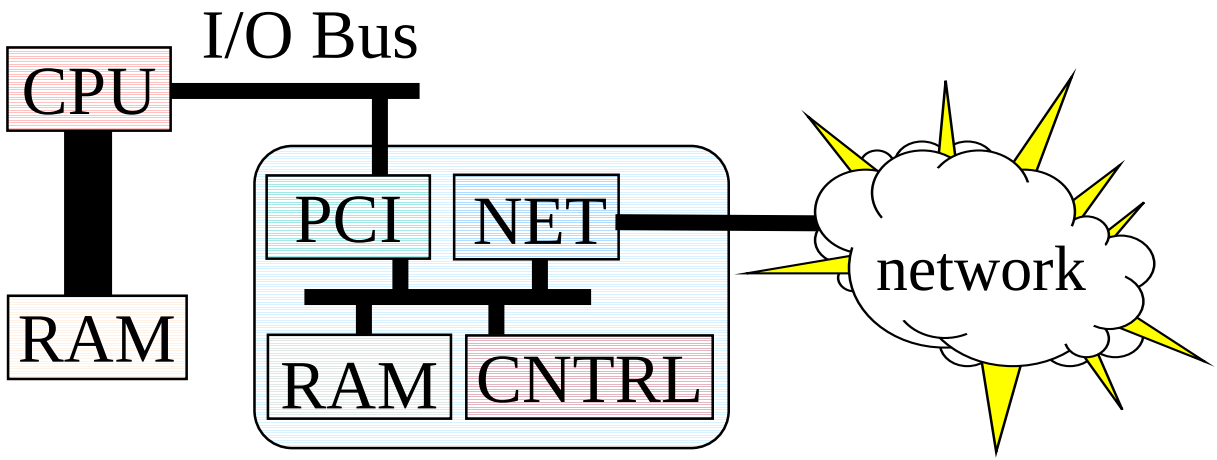
Motivation

Today's high-performance network interfaces (NIs) are programmable.

- Intel, 3Com, Alteon, Myri, etc.

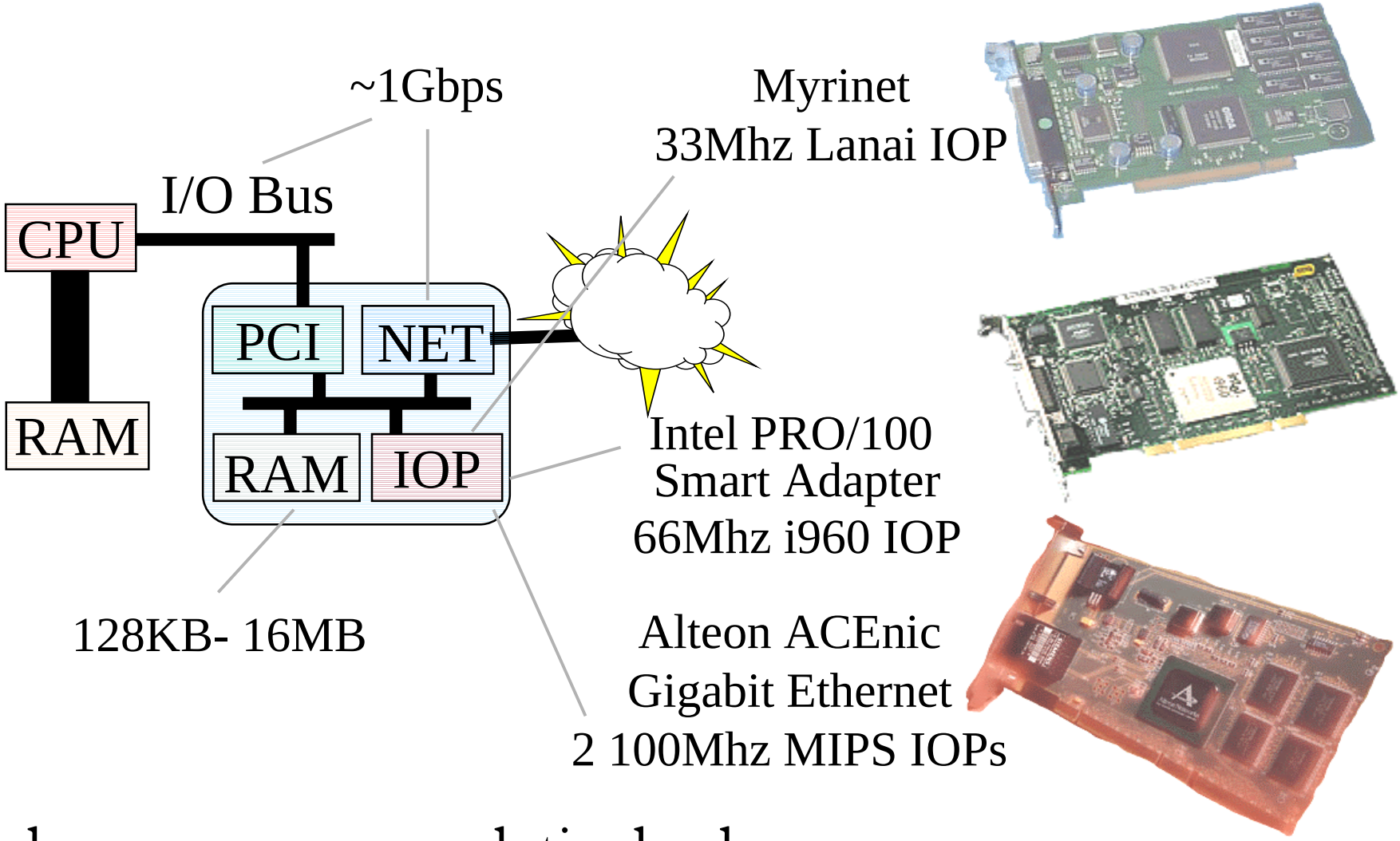
- **Question**: How should we build the next generation of programmable NIs?
  - What will they need to do?
  - What processor performance do they require?
  - What architectures provide this performance?

The old …



- Small memory, fixed function control

… and the current



- larger memory, relatively slow processor

Workloads

- *Messages/packets are the basic unit of work*
- Two types of server/router workloads
  - partial packet routines (e.g., filtering & web switching)
  - full packet routines(e.g., compression & transcoding)
- Consider one routine of each type
  - **ip4lookup:** *address-based* tree lookup, used in packet classification/filtering
  - **MD5:** computes unique signature over *full* packet, used in authentication, encryption, and compression.

Workloads

Applications	Description
Packet Classification/Filtering	Claim/forward/drop decisions, statistics gathering, and firewalling.
IP Packet Forwarding	Forward IP packets based on routing information.
Network Address Translation	Translate between globally routable and private IP packets. Useful for IP masquerading, virtual web server, etc.
TCP connection management	Traffic shaping within the network to reduce congestion.
TCP/IP	Offload TCP/IP processing from Internet/Web servers.
Web Switching	Web load balancing and proxy cache monitoring.
Virtual Private Network (VPN) IP Security (IPSec)	Encryption (DES) and Authentication (MD5)
Data Transcoding	Converting a multimedia data stream from one format to another within the network.
Duplicate Data Suppression	Reduce superfluous duplicate data transmission over high cost links.

- Two types of server/router workloads
  - partial packet routines (top six)
  - full packet routines(bottom four)

Benchmark Characteristics

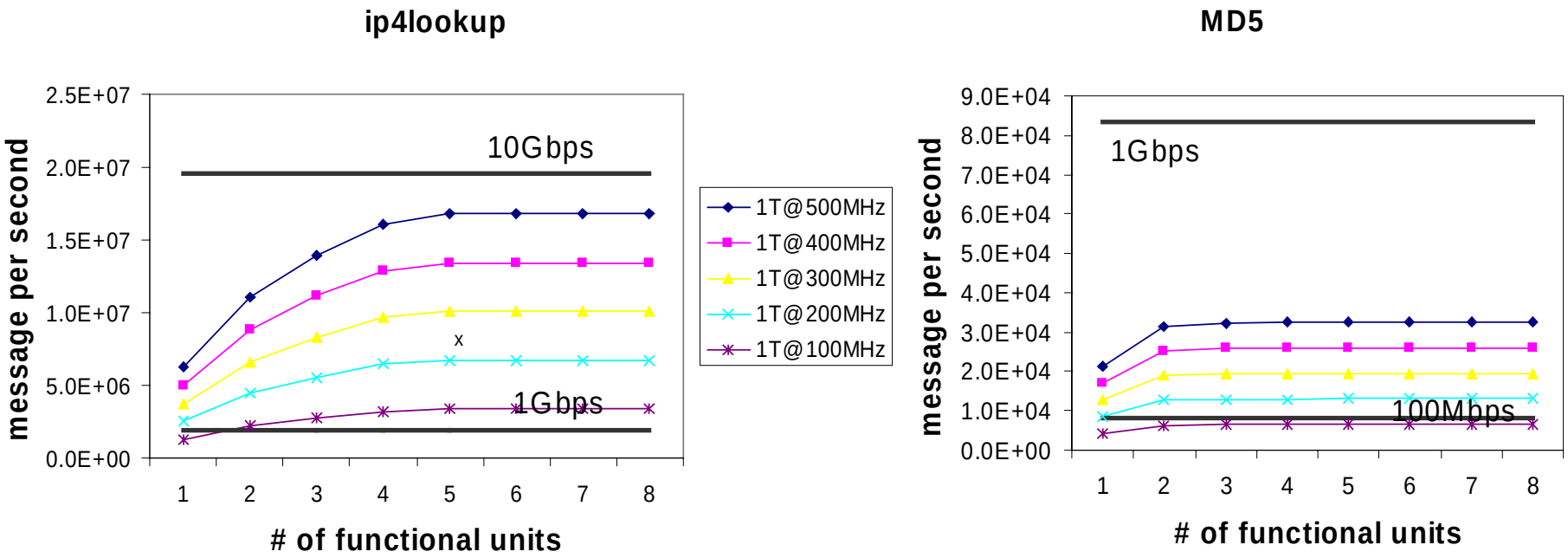
Application	Insts Executed per Message	Loads/Stores (%)	Ctrl Flow (%)	Other (%)
ip4lookup	120	18.6	12.1	69.3
MD5	23K	10.3	0.6	89.1

- 100x difference in computation per message
- Relatively few load/store & control instructions
  - neither stress memory hierarchy or branch predictor
  - Benchmarks are compute-bound
- Packets are processed independently
- *These applications must process packets at network speeds*

Experimental Architectures

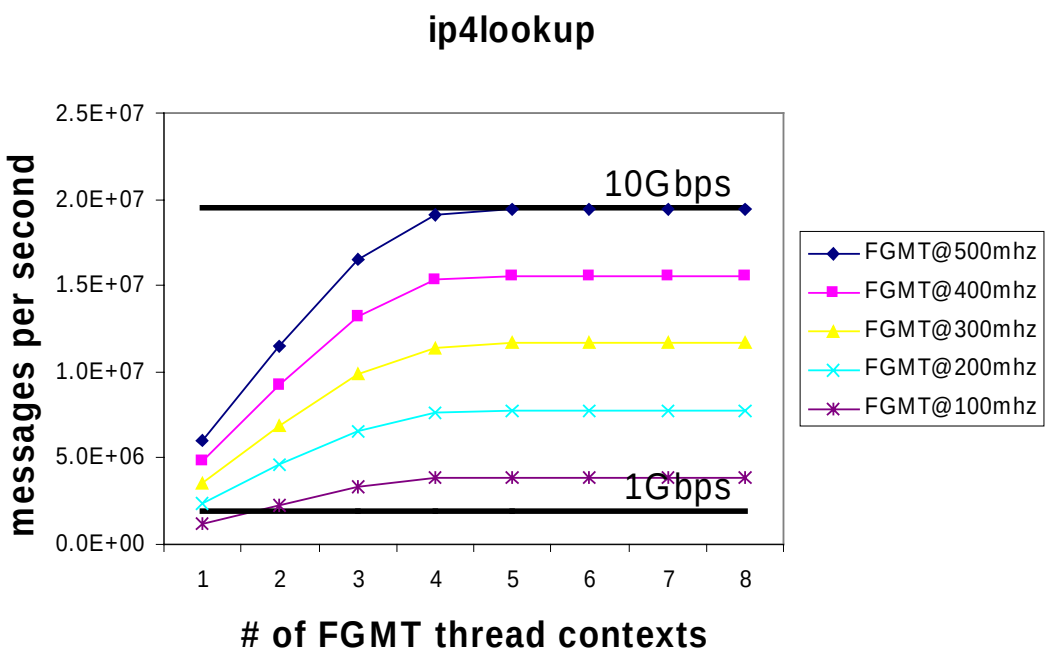
- Aggressive superscalar
  - issues 1-8 instructions per cycle, out-of-order
- Fine-grained multi-threaded (FGMT)
  - HW based thread contexts, fetch & issue instructions from one thread each cycle
- Simultaneously multi-threaded (SMT)
  - HW based thread contexts, fetch and issue from multiple threads each cycle

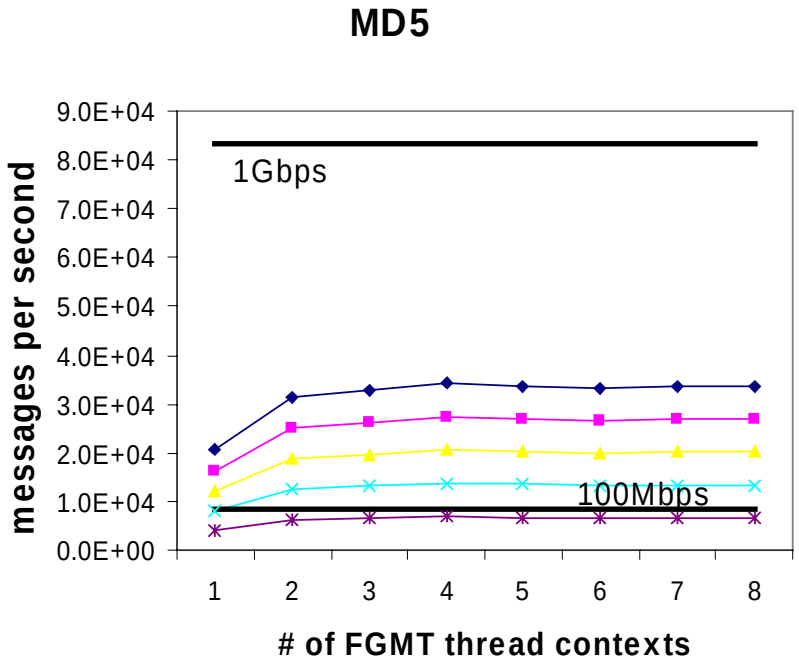
Superscalar Performance



- Performance linear with clock rate
- **ip4lookup** has more ILP
- Sustained link rates: **ip4lookup** @ 1Gbps & **MD5** @ 100Mbps

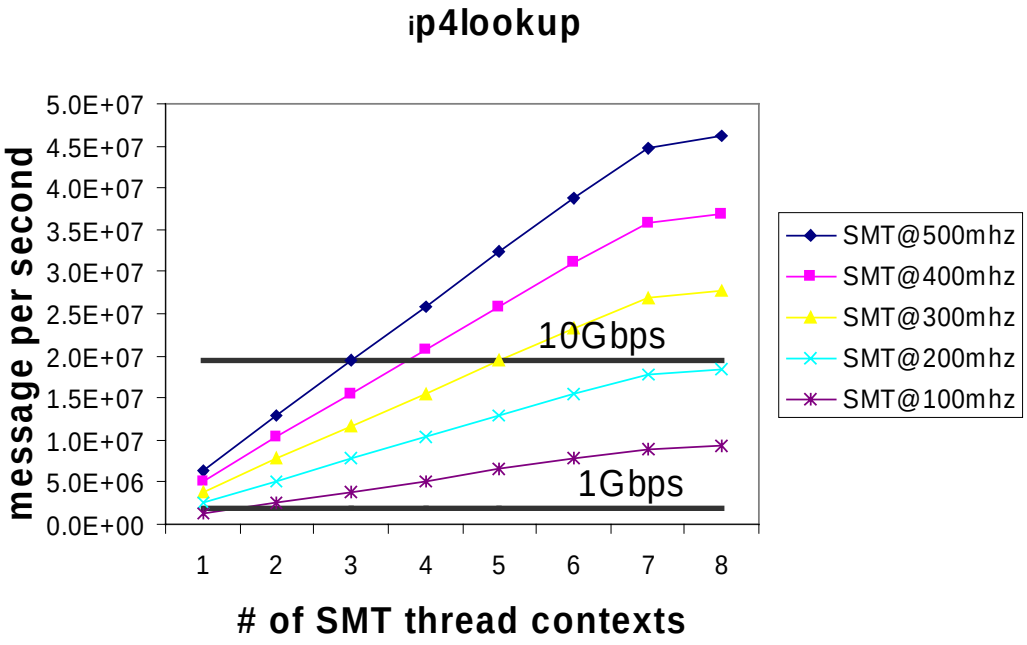
FGMT Performance

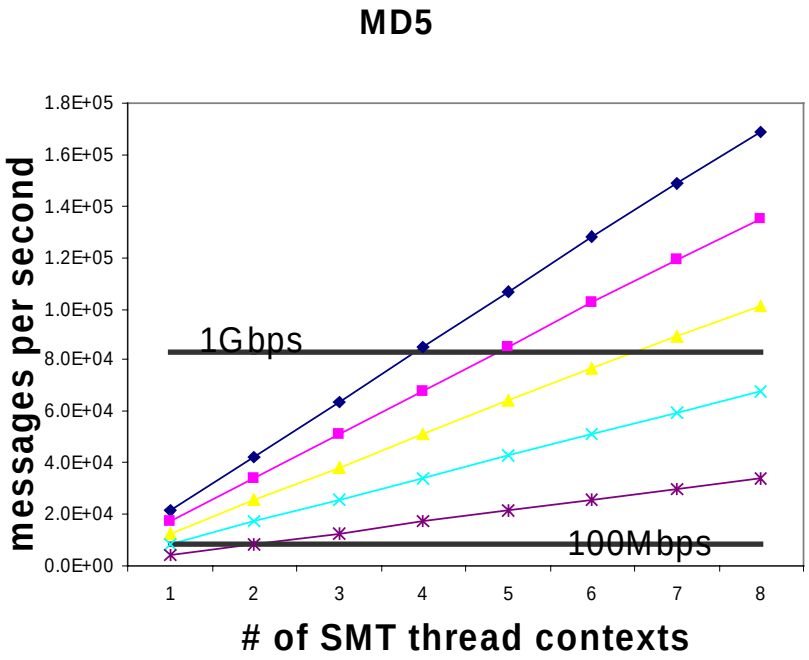




- Performance linear with clock rate
- **ip4lookup** performance slightly better than superscalar
- Sustained link rates: **ip4lookup** @ 1Gbps & **MD5** @ 100Mbps

SMT Performance





- Performance linear with clock rate
- Performance linear with # of threads
- Sustained link rates: **ip4lookup** @ 1 & 10Gbps and **MD5** @ 100Mbps & 1Gbps

Conclusions

- Workloads embarrassingly parallel at the packet level
- Aggressive superscalar and FGMT processors do not efficiently exploit packet-level parallelism.
- SMT can.

Future Work

- Investigate multiprocessor performance
- Execute a selection of workload routines simultaneously
- Consider memory contention among NI processor, host interface, and physical network interface.