Network Processor: Architecture and Applications

Yan Luo
Yan_Luo@uml.edu
http://faculty.uml.edu/yluo/

Outline

- Overview of Network Processors
- Network Processor Architectures
- Applications
- Case Studies
  - Wireless Mesh Network
  - a Content-Aware Switch
- Conclusion

Packet Processing in the Future Internet

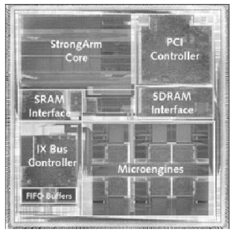
Future Internet

More packets
&
Complex packet processing

ASIC

+

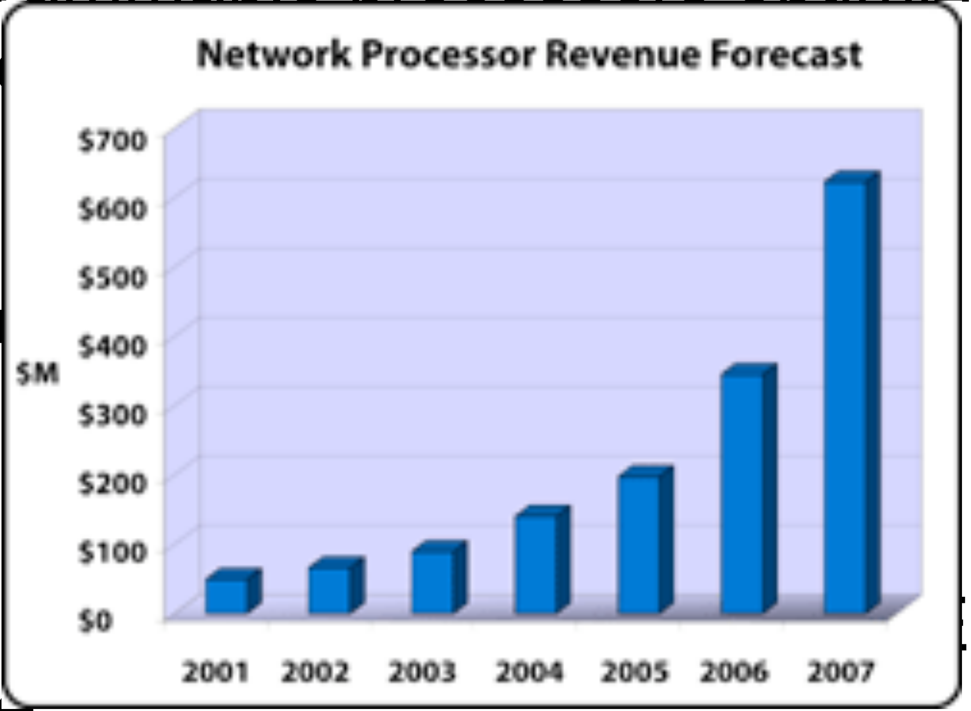
General-Purpose Processors



- •High processing power
- •Support wire speed
- •Programmable
- •Scalable
- •Optimized for network applications
- • …

What is Network Processor ?

- Programmable processors optimized for netw processing
  - H
  - Pr
  - O
- Main zchip, Agere

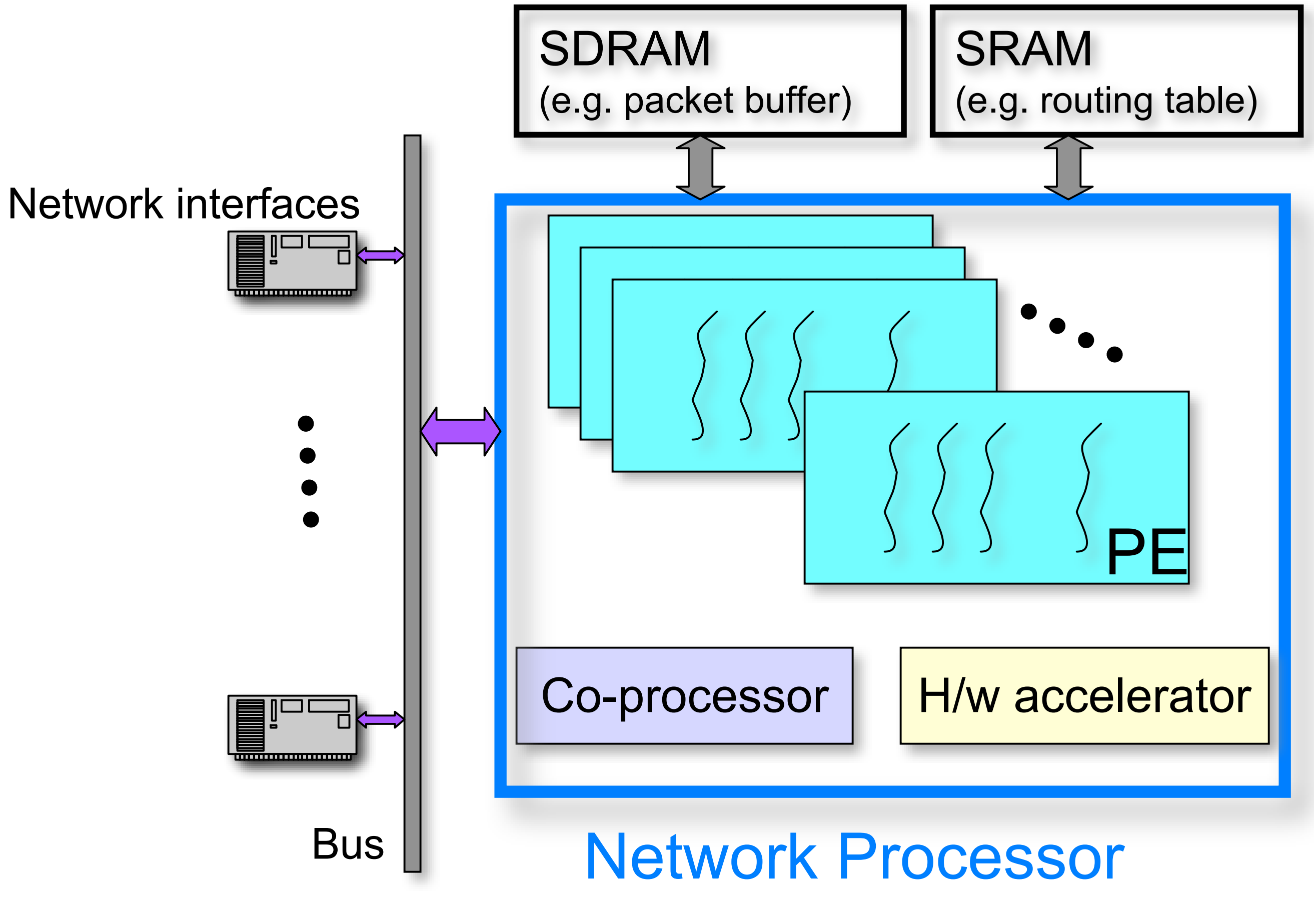


Semico Research Corp. Oct. 14, 2003

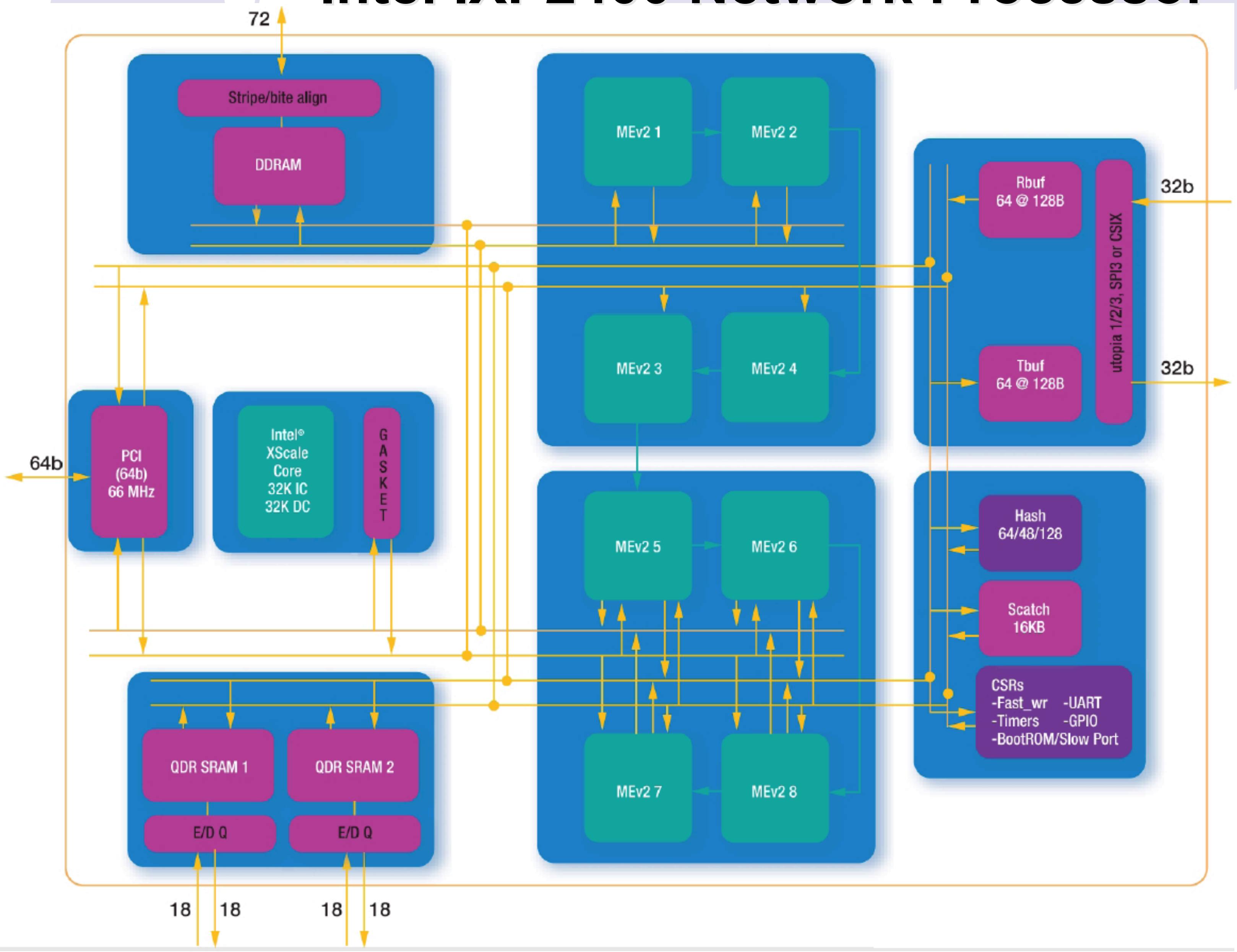
Commercial Network Processors

Vendor	**Product**	**Line speed**	**Features**
AMCC	nP7510	OC-192/ 10 Gbps	Multi-core, customized ISA, multi-tasking
Intel	IXP2850	OC-192/ 10 Gbps	Multi-core, h/w multi-threaded, coprocessor, h/w accelerators
Hifn	5NP4G	OC-48/ 2.5 Gbps	Multi-threaded multiprocessor complex, h/w accelerators
EZchip	NP-2	OC-192/ 10 Gbps	Classification engines, traffic managers
Agere	PayloadPlus	OC-192/ 10 Gbps	Multi-threaded, on-chip traffic management

Typical Network Processor Architecture



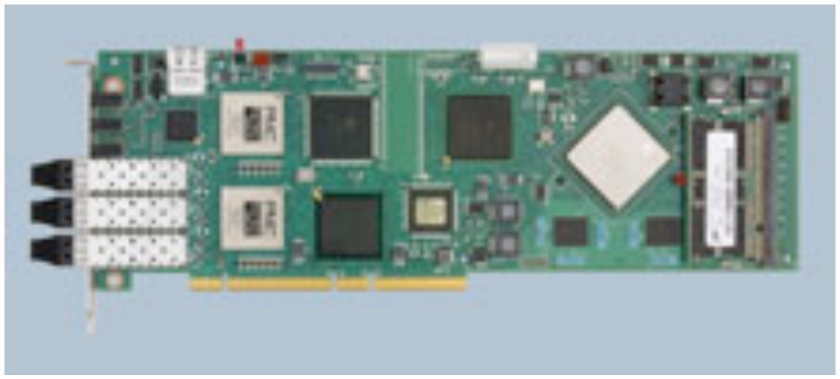
Intel IXP2400 Network Processor



Snapshots of IXP2xxx Based Systems

ADI Roadrunner Platform

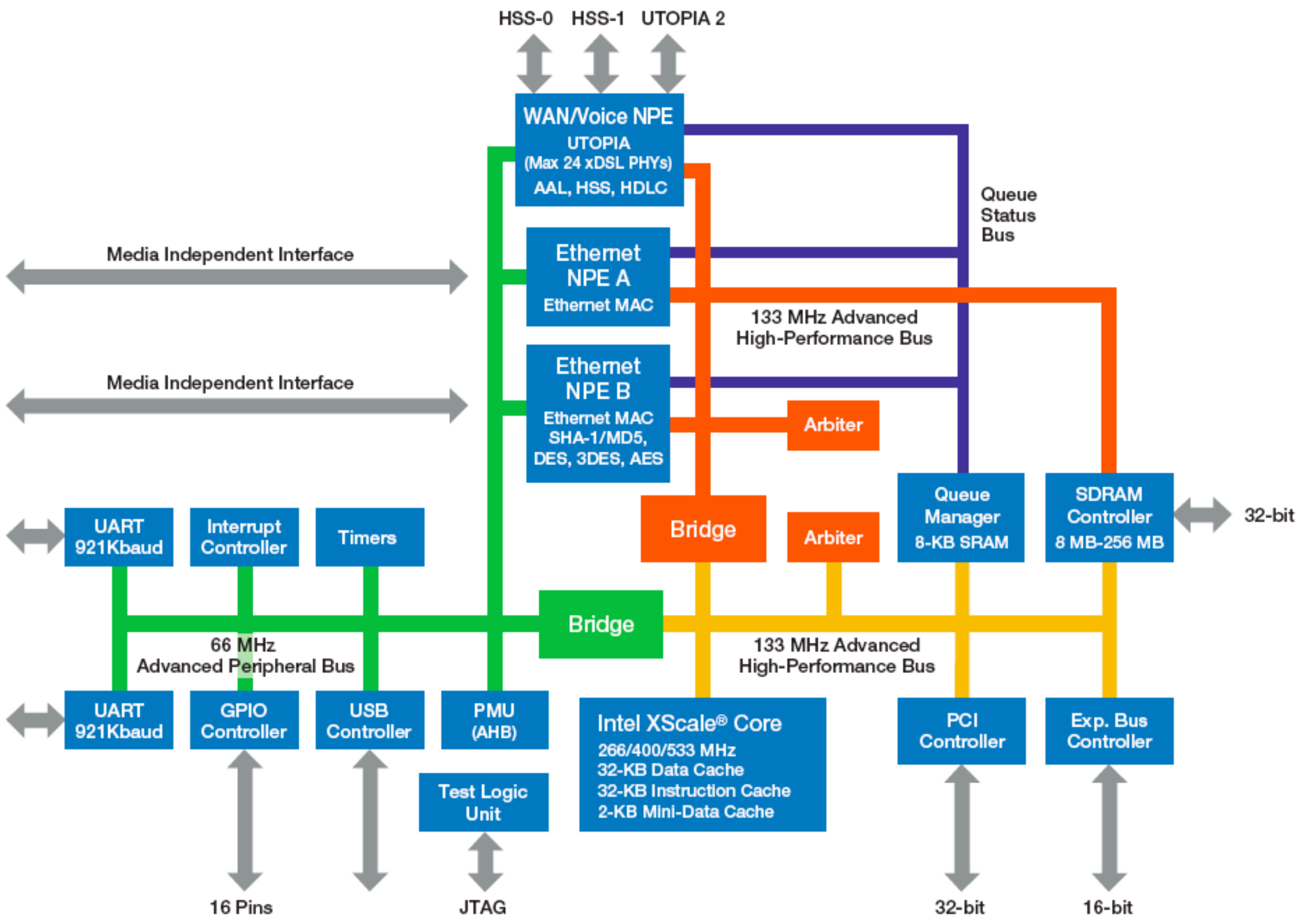
- IPv4 Forwarding/NAT
- Forwarding w/ QoS / DiffServ
- ATM RAN
- IP RAN
- IPv6/v4 dual stack forwarding



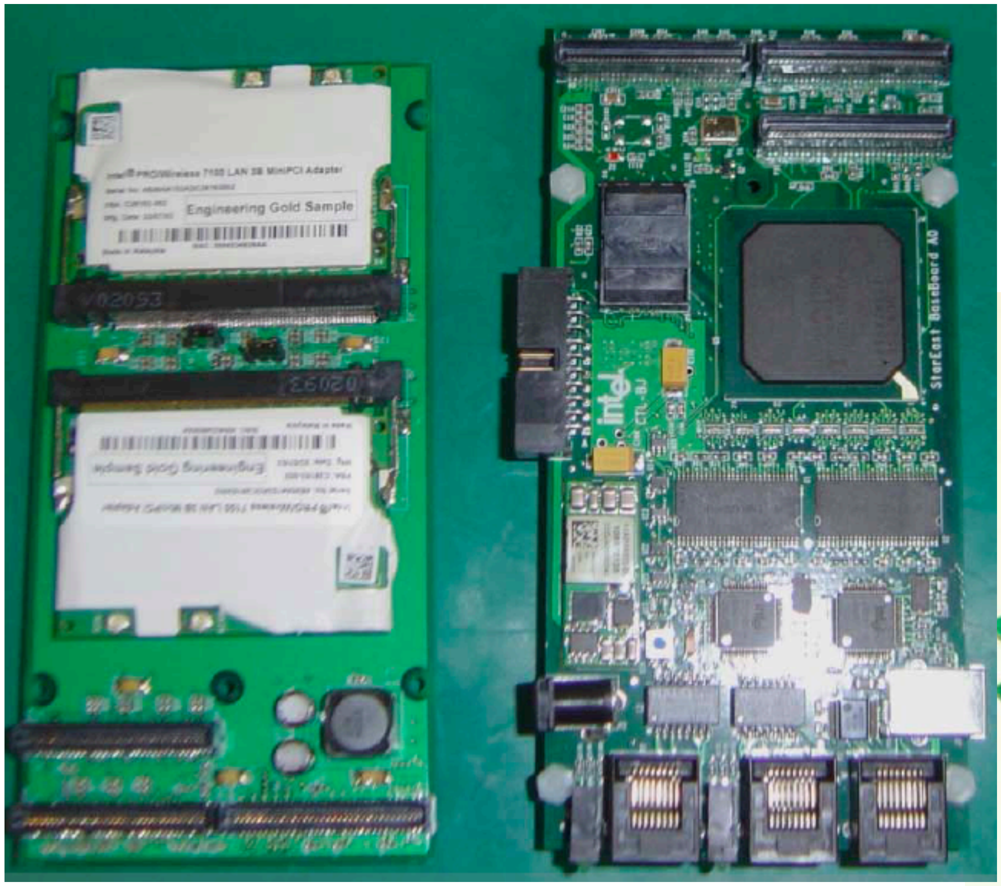
Radisys ENP2611 PCI Packet Processing Engine

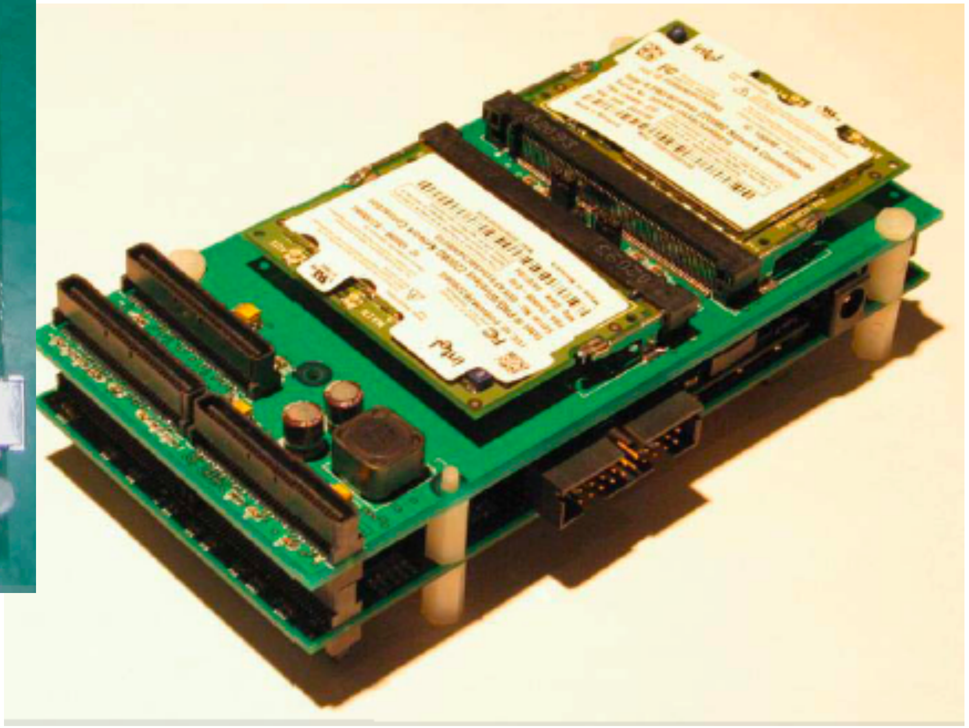
- multiservice switches,
- routers, broadband access devices,
- intrusion detection and prevention (IDS/IPS)
- Voice over IP (VoIP) gateway
- Virtual Private Network gateway
- Content-aware switch

Intel IXP425 Network Processor

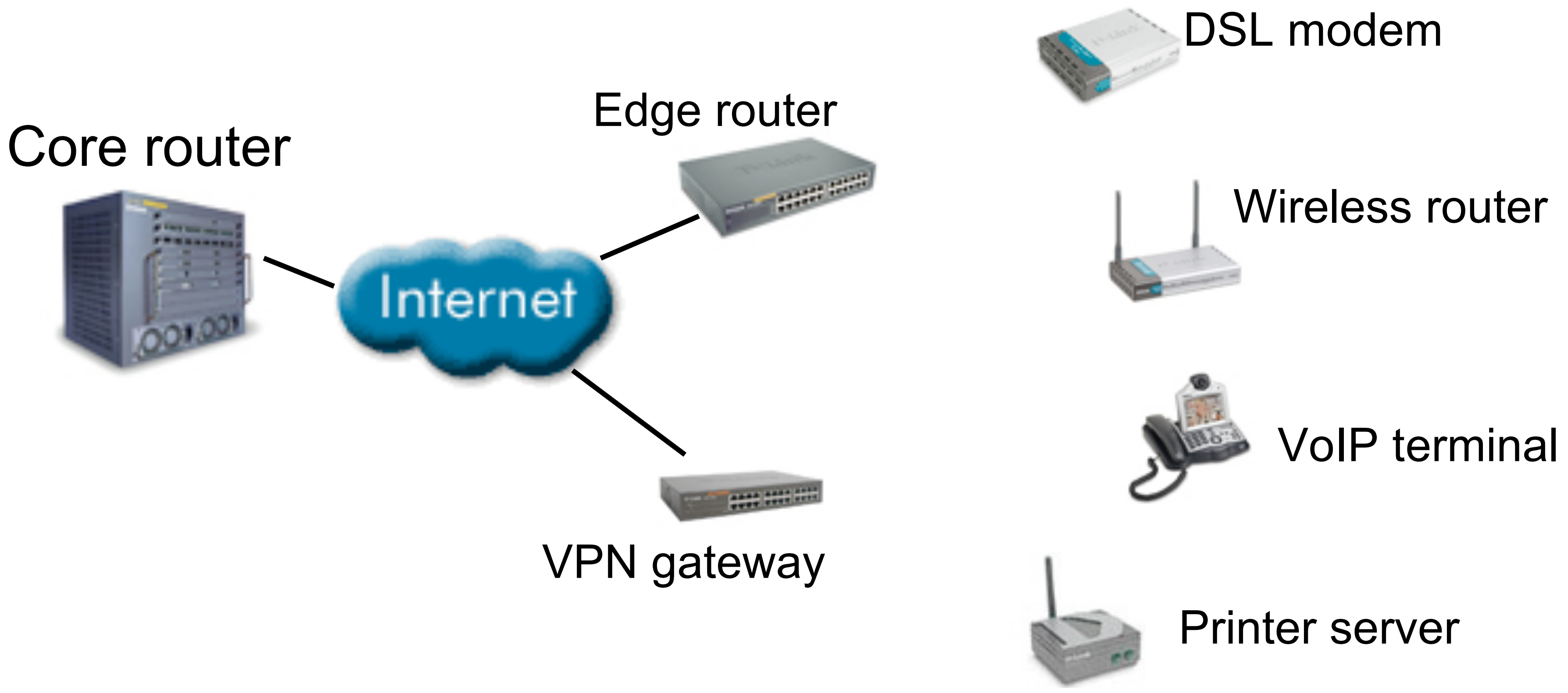


StarEast: IXP425 Based Multi-radio Platform

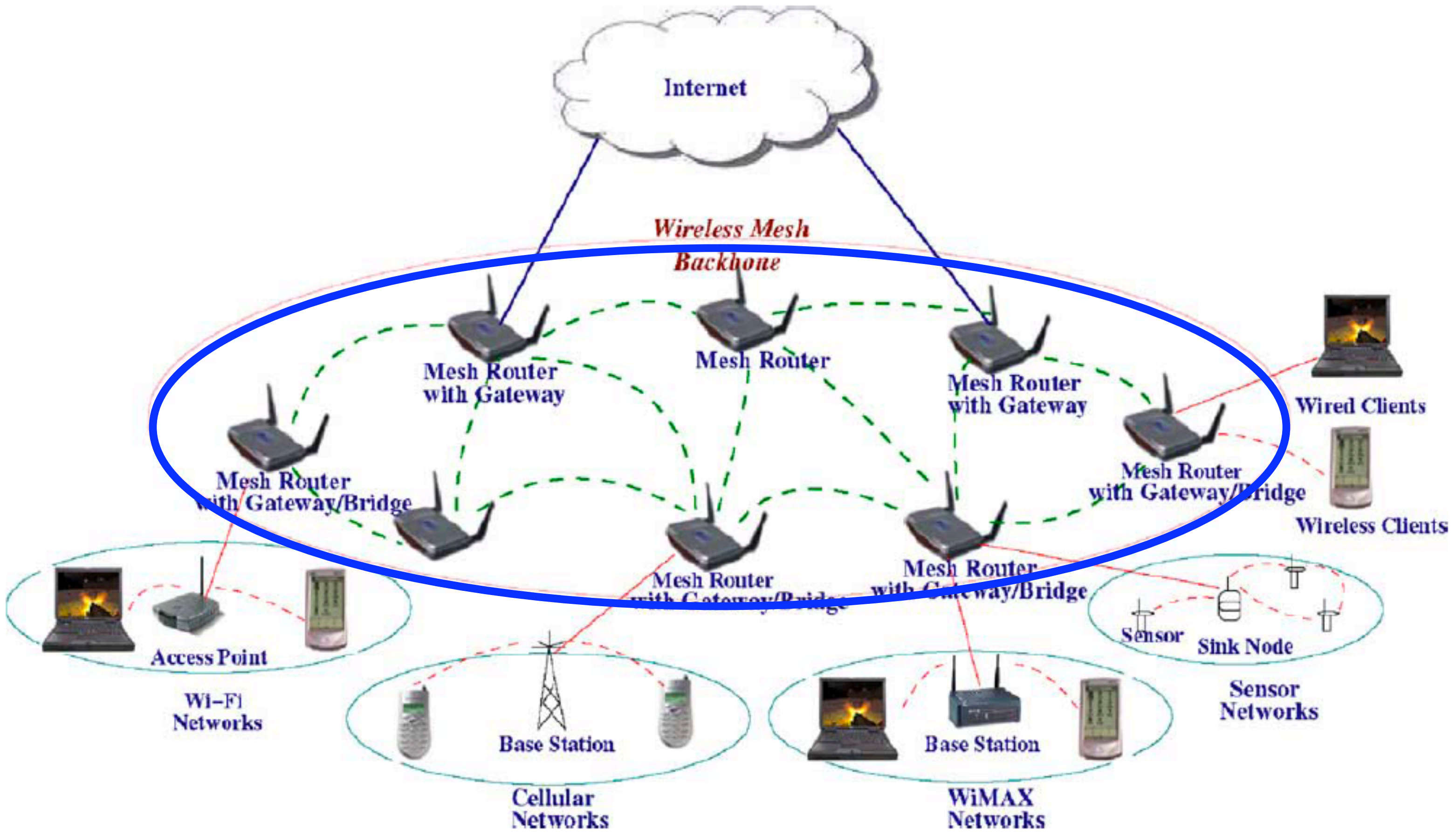




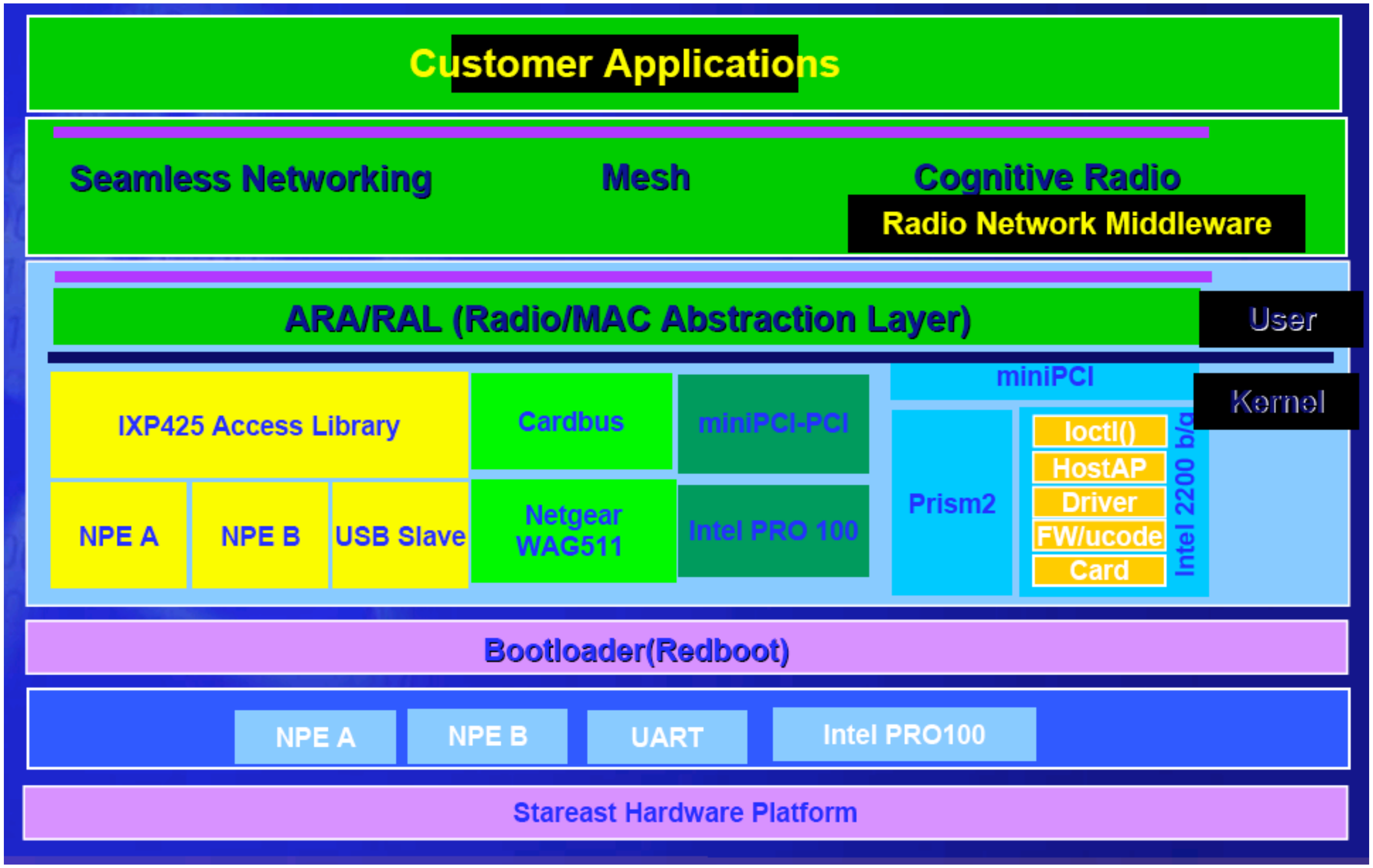
Applications of Network Processors



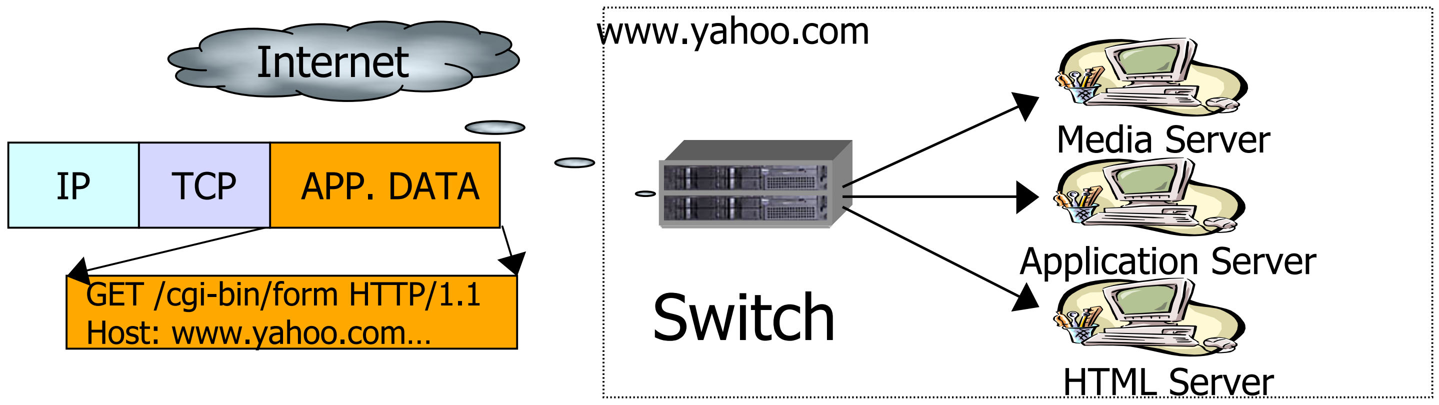
Case Study 1: Wireless Mesh Network



Software Stack on StarEast



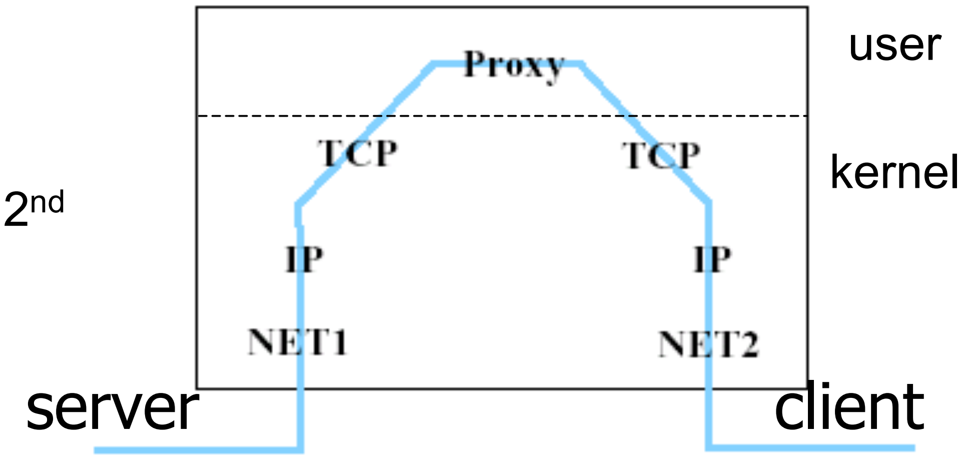
Case Study 2: Content-aware Switch



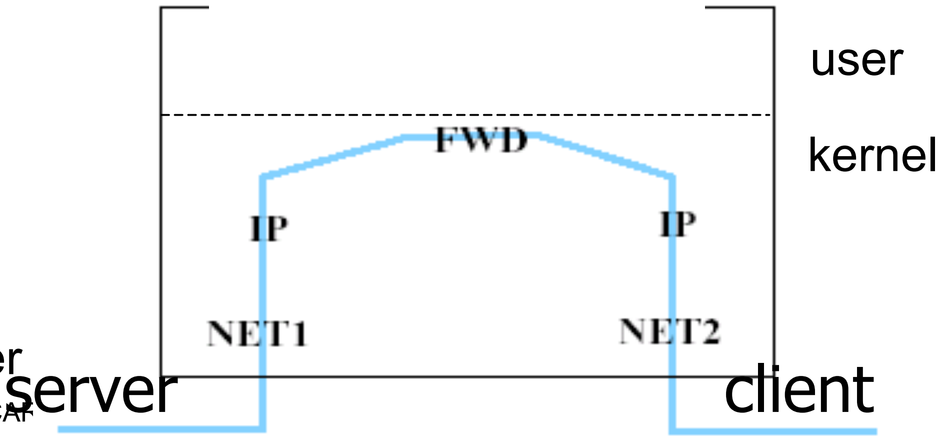
- Front-end of a Web cluster, only one Virtual IP
- Route packets based on Layer 5 information
  - Examine application data in addition to IP& TCP
- Advantages over layer 4 switches
  - Better load balancing: distributed based on content type
  - Faster response: exploit cache affinity
  - Better resource utilization: partition database

Mechanisms to Build a Content-aware Switch

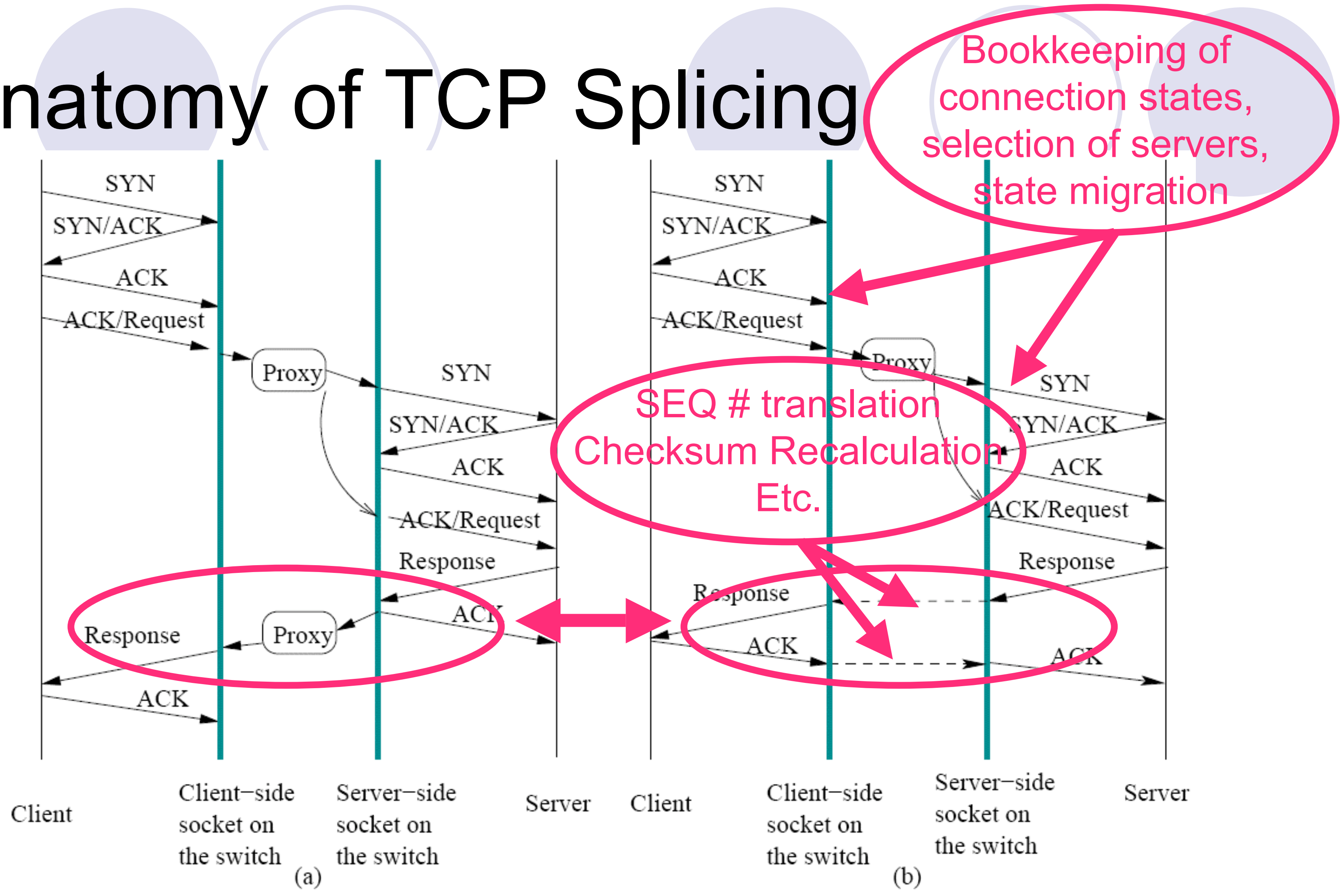
- TCP gateway
  - An application level proxy
  - Setup 1st connection w/ client, parses request →server, setup 2nd connection w/ server
  - Copy overhead



- TCP splicing
  - Reduce the copy overhead
  - Forward packet at network level between the network interface driver and the TCP/IP stack
  - Two connections are spliced together
  - Modify fields in IP and TCP header



Anatomy of TCP Splicing

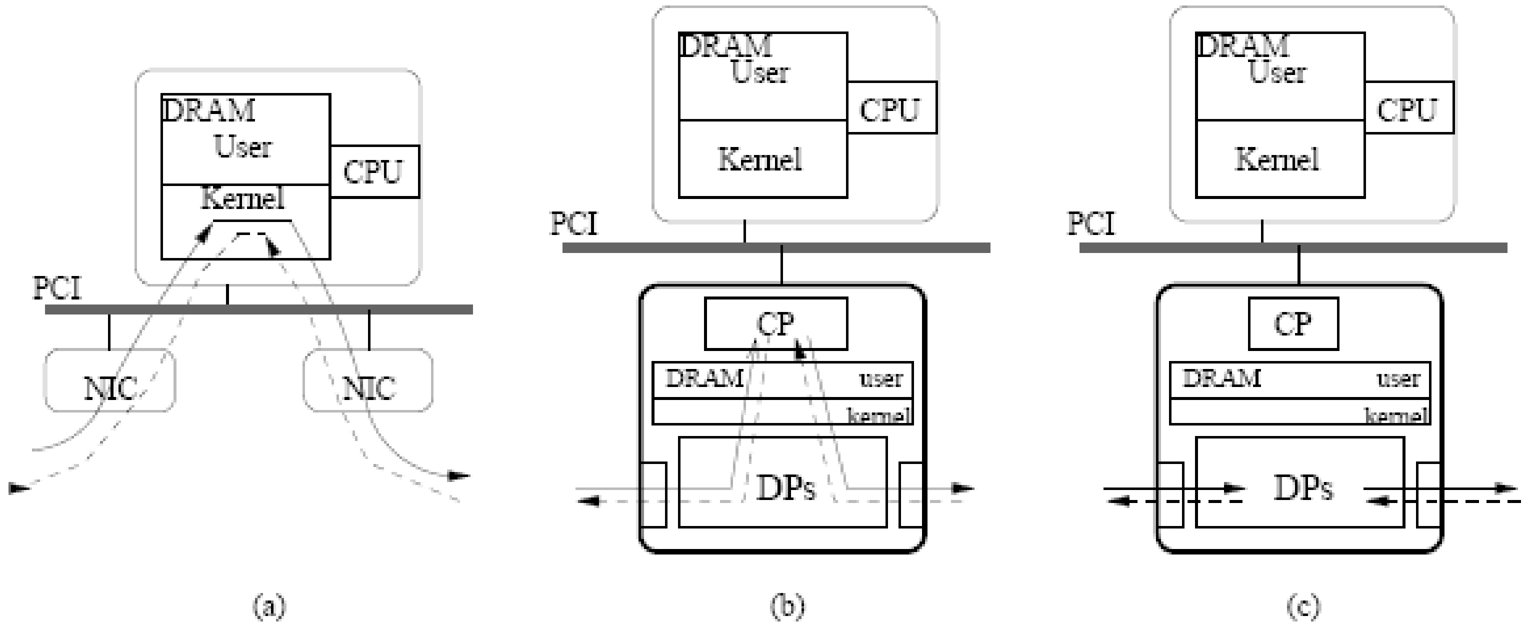


Bookkeeping of connection states, selection of servers, state migration
SEQ # translation Checksum Recalculation Etc.
SYN
SYN/ACK
ACK
ACK/Request
Proxy
Response
Client
Client−side socket on the switch
Server−side socket on the switch
Server
(a)
(b)

Without TCP Splicing

With TCP Splicing

Design Options



•Option 0: GP-based (Linux-based) switch

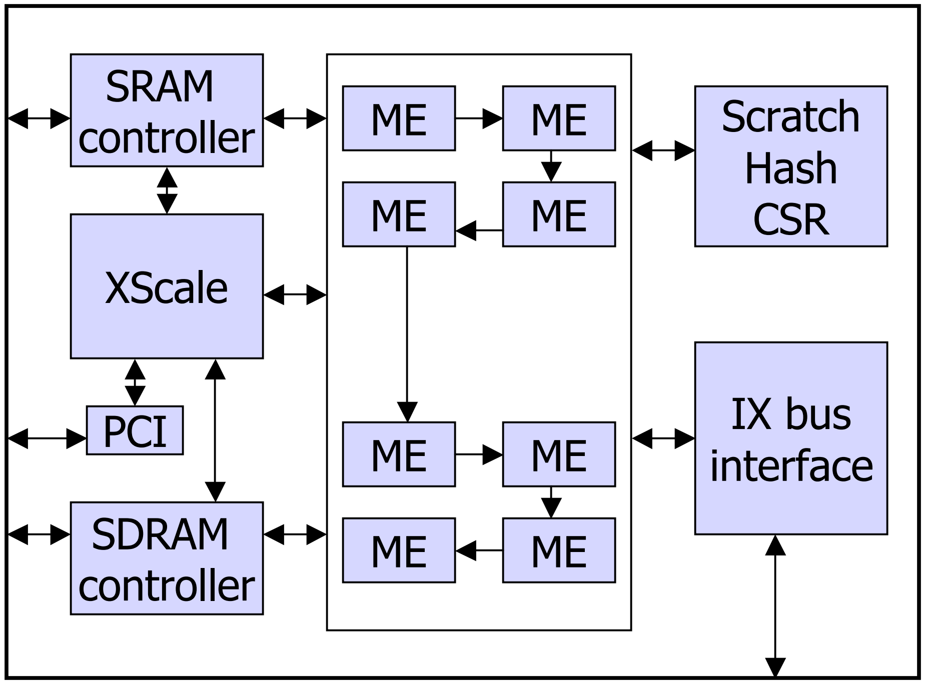
•Option 1: CP setup & and splices connections, DPs process packets sent after splicing

Connection setup & splicing is more complex than data forwarding

Packets before splicing need to be passed through DRAM queues

•Option 2: DPs handle connection setup, splicing & forwarding

IXP 2400 Block Diagram



- XScale core
- Microengines(MEs)
  - 2 clusters of 4 microengines each
- Each ME
  - run up to 8 threads
  - 16KB instruction store
  - Local memory
- Scratchpad memory, SRAM & DRAM controllers

Resource Allocation

SRAM (8MB)

- Client side CB list
- Server side CB list
- server selection table
- Locks

- Client-side control block list
  - record states for connections between clients and SpliceNP, states after splicing
- Server-side control block list
  - record states for connections between server and SpliceNP

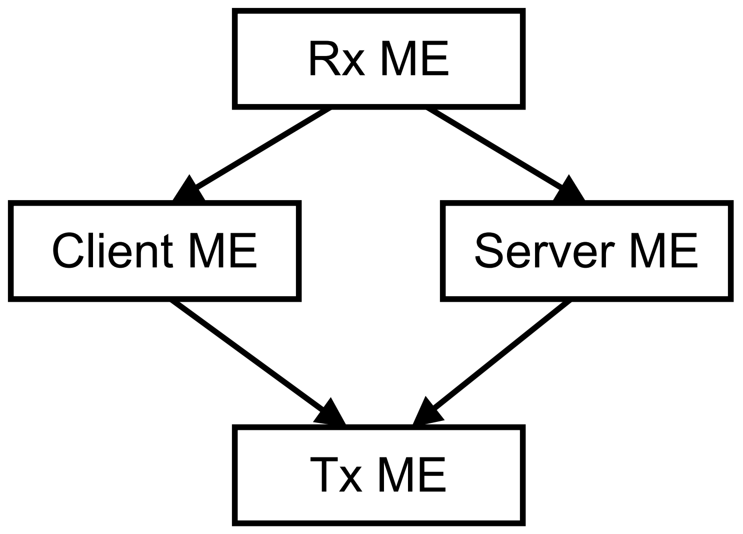
DRAM (256MB)

Packet buffer

Scratchpad (16KB)

Packet queues

Microengines



Comparison of Functionality

- *A lite version of TCP due to the limited instruction size of microengines.*

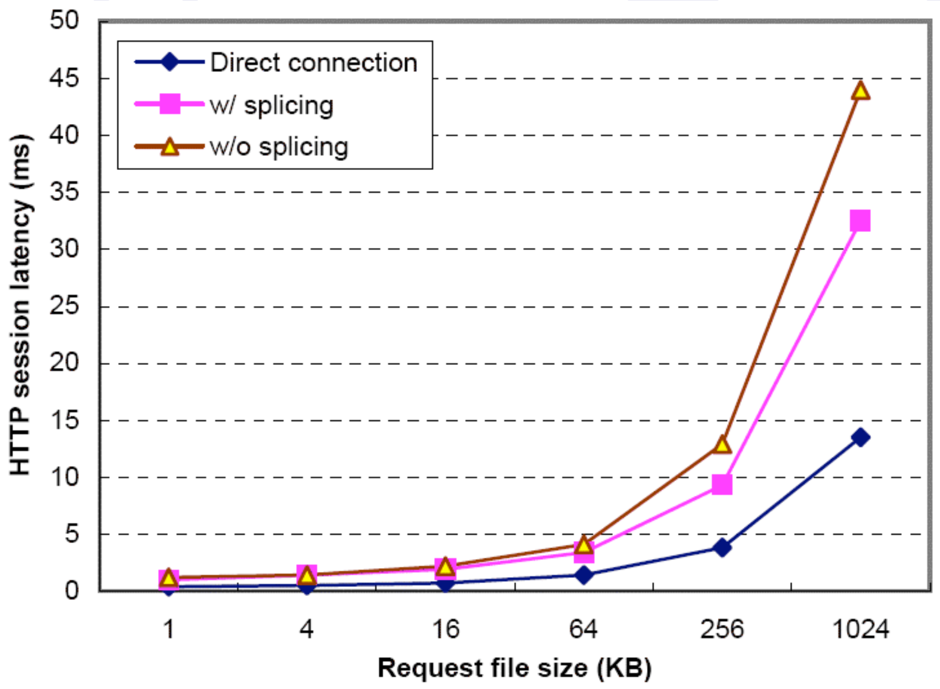
Processing a SYN packet

Step	Functionality	TCP	Linux Splicer	SpliceNP
1	Dequeue packet	Y	Y	Y
2	IP header verification	Y	Y	Y
3	IP option processing	Y	Y	N
4	TCP header verification	Y	Y	Y
5	Control block lookup	Y	Y	Y
6	Create new socket and set state to LISTEN	Y	Y	No socket, only control block
7	Initialize TCP and IP header template	Y	Y	N
8	Reset idle time and keep-alive timer	Y	Y	N
9	Process TCP option	Y	Y	Only MSS option
10	Send ACK packet, change state to SYN_RCVD	Y	Y	Y

Experimental Setup

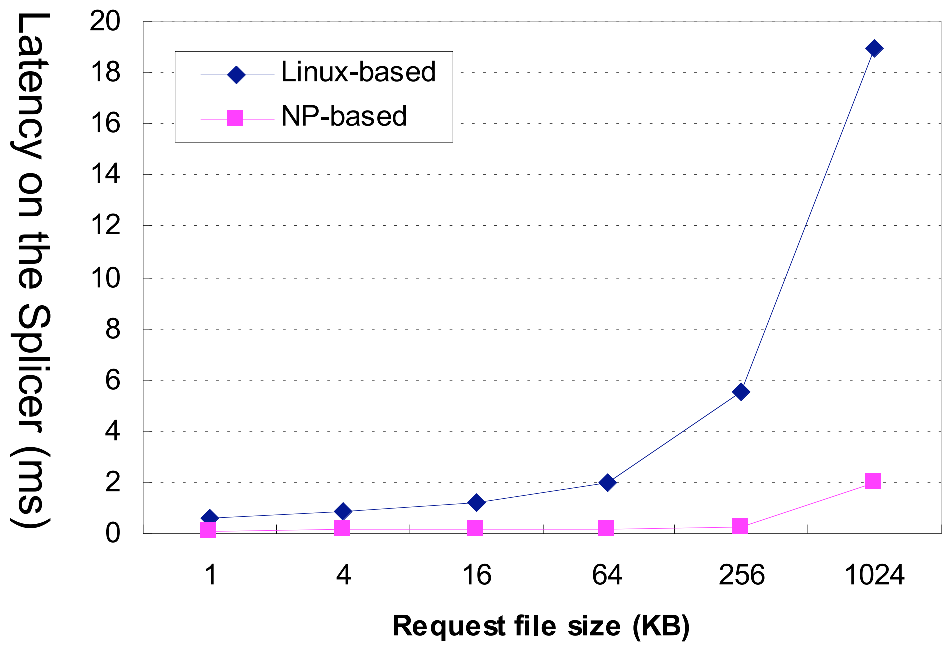
- Radisys ENP2611 containing an IXP2400
  - XScale & ME: 600MHz
  - 8MB SRAM and 128MB DRAM
  - Three 1Gbps Ethernet ports: 1 for Client port and 2 for Server ports
- Server: Apache web server on an Intel 3.0GHz Xeon processor
- Client: Httperf on a 2.5GHz Intel P4 processor
- Linux-based switch
  - Loadable kernel module
  - 2.5GHz P4, two 1Gbps Ethernet NICs

Latency on a Linux-based TCP Splicer



- Latency is reduced by TCP splicing

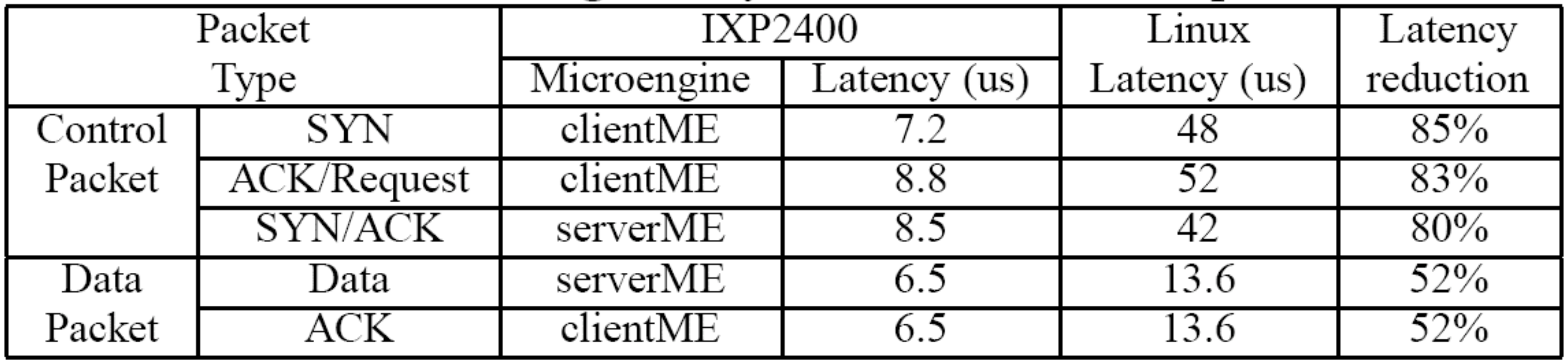
Latency vs Request File Size



- Latency reduced significantly
  - 83.3% (0.6ms → 0.1ms) @ 1KB
- The larger the file size, the higher the reduction
  - 89.5% @ 1MB file

Comparison of Packet Processing Latency

Table 5: Processing latency for control and data packets

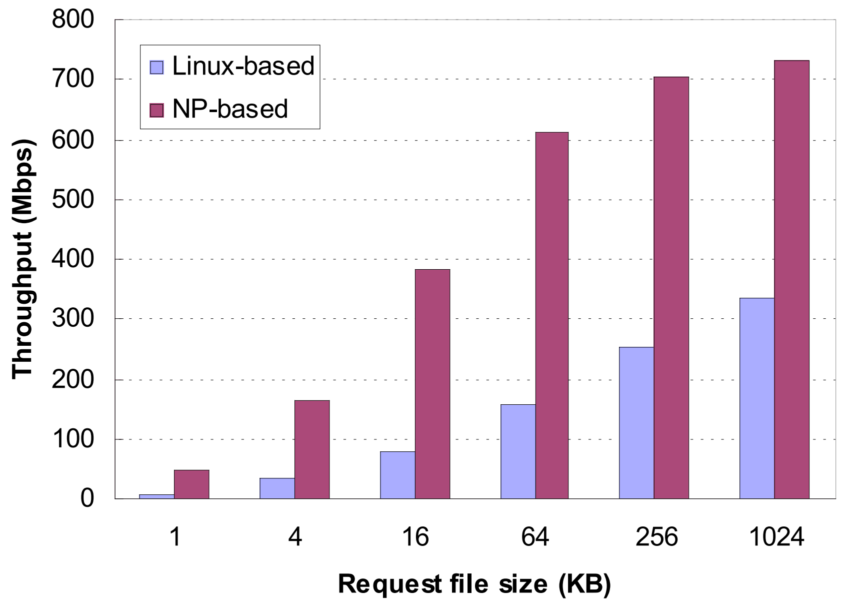


Packet Type		IXP2400		Linux	Latency
		Microengine	Latency (us)	Latency (us)	reduction
Control Packet	SYN	clientME	7.2	48	85%
	ACK/Request	clientME	8.8	52	83%
	SYN/ACK	serverME	8.5	42	80%
Data Packet	Data	serverME	6.5	13.6	52%
	ACK	clientME	6.5	13.6	52%

Analysis of Latency Reduction

Linux-based	NP-based
Interrupt: NIC raises an interrupt once a packet comes	polling
NIC-to-mem copy Xeon 3.0Ghz Dual processor w/ 1Gbps Intel Pro 1000 (88544GC) NIC, 3 us to copy a 64-byte packet by DMA	No copy: Packets are processed inside without two copies
Linux processing: OS overheads Processing a data packet in splicing state: 13.6 us	IXP processing: Optimized ISA 6.5 us

Throughput vs Request File Size



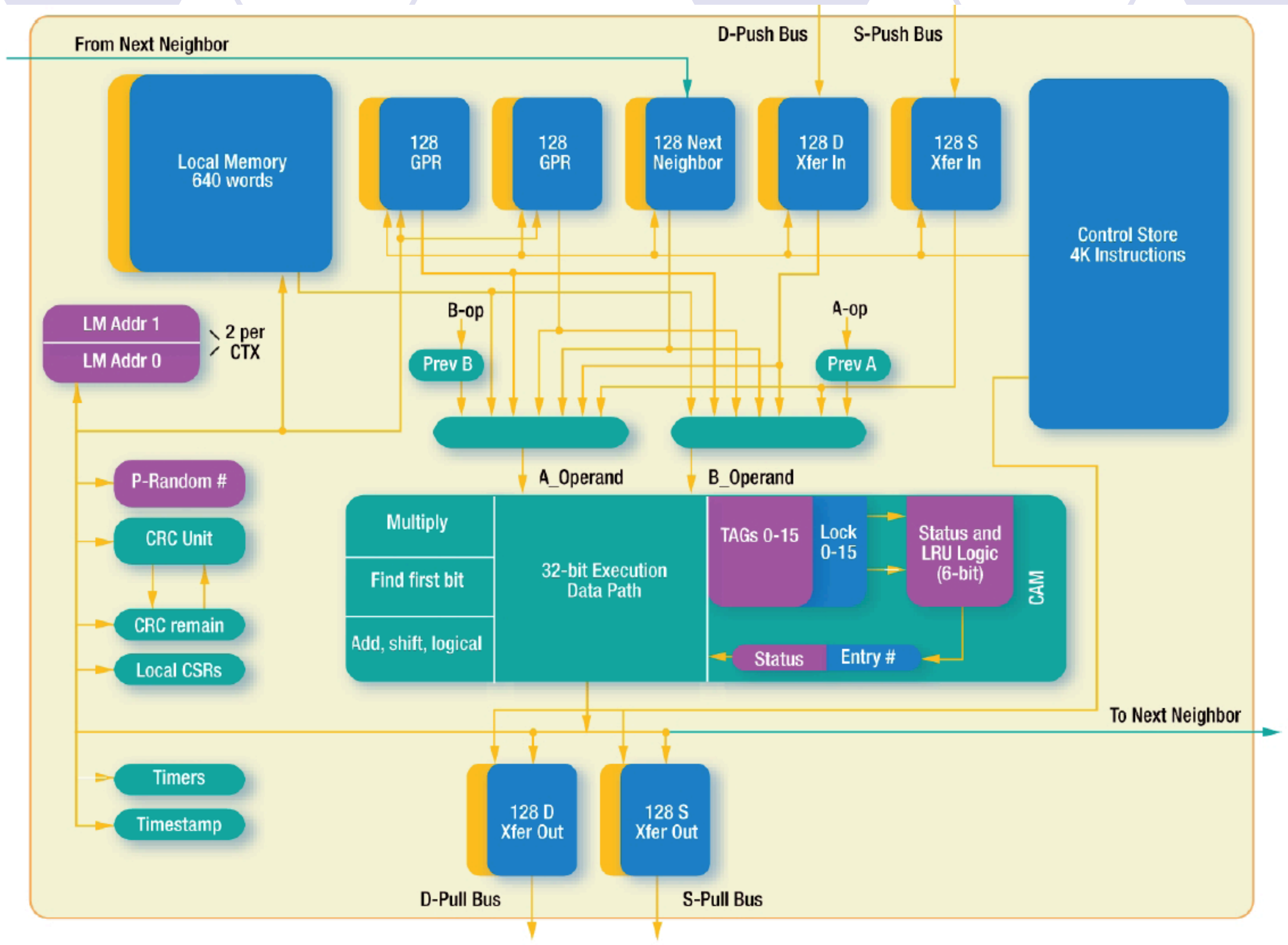
- Throughput is increased significantly
  - 5.7x for small file size @ 1KB, 2.2x for large file @ 1MB
- Higher improvement for small files
  - Latency reduction for control packets > data packets
  - Control packets take a larger portion for small files

Conclusion

- Network Processor combines high-performance packet processing and programmability
- A large variety of NP applications
- Efficient resource utilization is challenging

Thank you !

Microengine


From Next Neighbor
D-Push Bus
S-Push Bus
Local Memory 640 words
128 GPR
128 GPR
128 Next Neighbor
128 D Xfer In
128 S Xfer In
Control Store 4K Instructions
LM Addr 1
LM Addr 0
2 per CTX
B-op
A-op
Prev B
Prev A
A_Operand
B_Operand
P-Random #
CRC Unit
CRC remain
Local CSRs
Multiply
Find first bit
Add, shift, logical
32-bit Execution Data Path
TAGs 0-15
Lock 0-15
Status and LRU Logic (6-bit)
CAM
Status
Entry #
To Next Neighbor
Timers
Timestamp
128 D Xfer Out
128 S Xfer Out
D-Pull Bus
S-Pull Bus