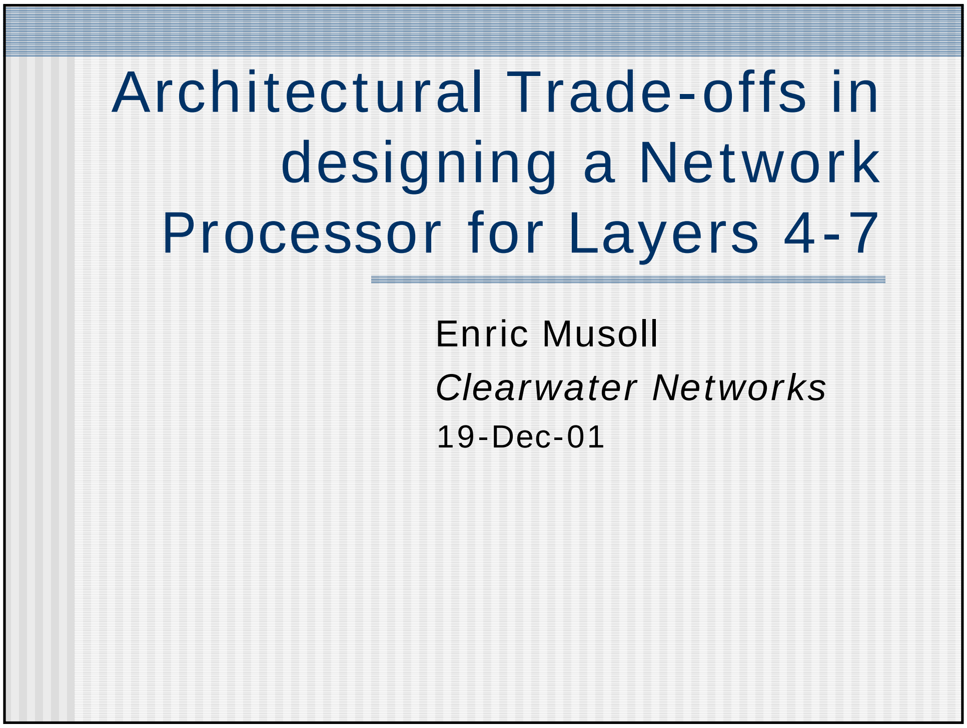
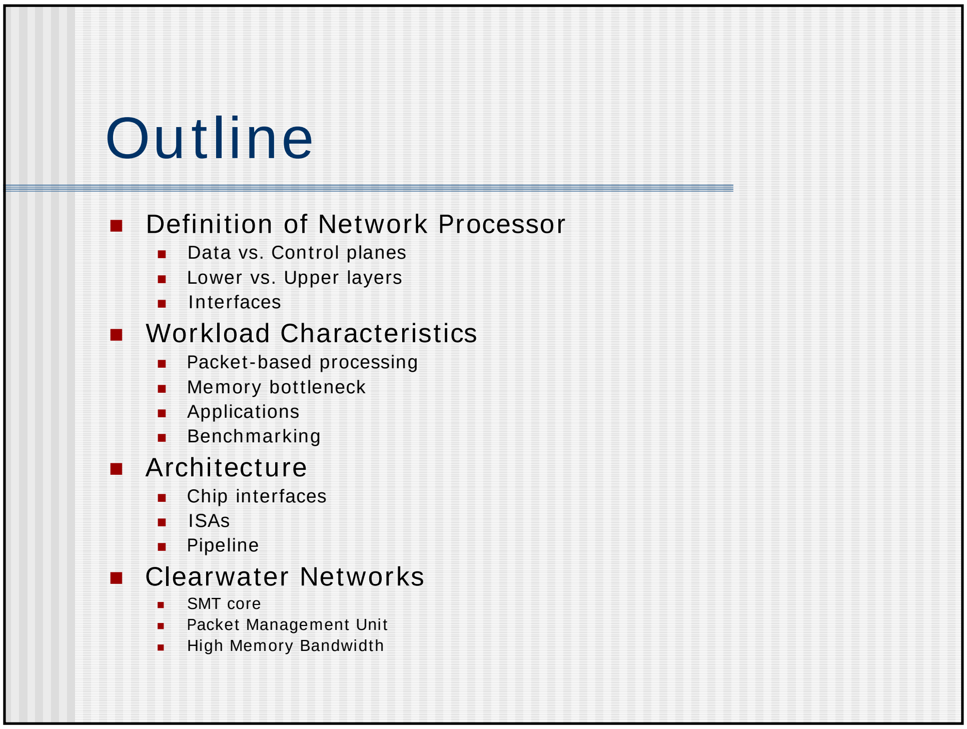


Architectural Trade-offs in designing a Network Processor for Layers 4-7

Enric Musoll

Clearwater Networks

19-Dec-01



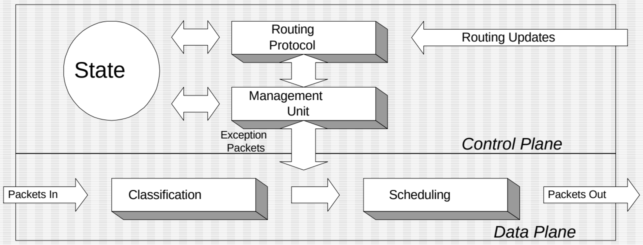
Outline

- Definition of Network Processor
  - Data vs. Control planes
  - Lower vs. Upper layers
  - Interfaces
- Workload Characteristics
  - Packet-based processing
  - Memory bottleneck
  - Applications
  - Benchmarking
- Architecture
  - Chip interfaces
  - ISAs
  - Pipeline
- Clearwater Networks
  - SMT core
  - Packet Management Unit
  - High Memory Bandwidth

Definition – Several …

- Fundamental building blocks of the internet infrastructure
  - Enable efficient processing of network cells or packets
- Several definitions depending on what dimension we are
  - Data vs. Control plane
  - Lower vs. Upper layers
  - Interfaces (bandwidth)
  - Programmability (ASIC vs. processor)

Definition – Control/Data

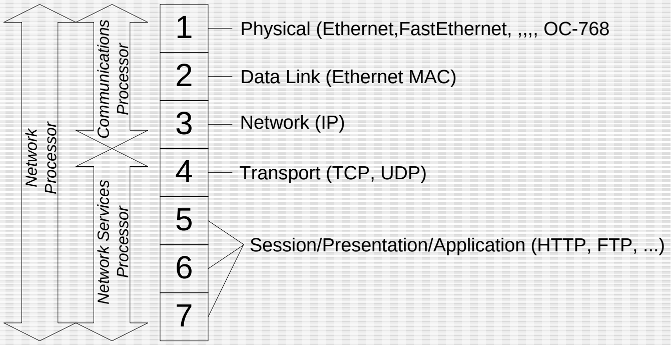


- Data Plane: processing done for each packet
- Control Plane: otherwise
- Data plane passes exception packets to control plane
  - Too complex processing for data plane
- Exception packets often result in modification of state
- Key: how often exception packets occur?

Definition – Data/Control

- Data Plane typical functions
  - Packet source/destination lookup and next-hop determination
  - QoS/CoS determination and/or enforcement
  - Packet encapsulation/de-encapsulation
  - Packet fragmentation/re-assembly
  - Maintenance of traffic statistics
- Control plan typical functions
  - Running routing protocols
  - Managing routing tables
  - Responding to management of complex inquiries (TELNET, HTTP requests,...)
  - Other system managements tasks, like host processor communications

Definition – Layers



1 Physical (Ethernet,FastEthernet, ,,,, OC-768

2 Data Link (Ethernet MAC)

3 Network (IP)

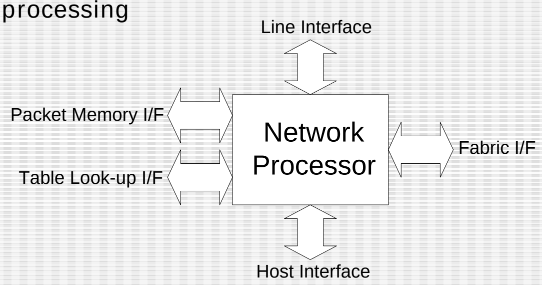
4 Transport (TCP, UDP)

5, 6, 7 Session/Presentation/Application (HTTP, FTP, ...)

- CP + NSP == NP
- CP == NP
- NSP == NP

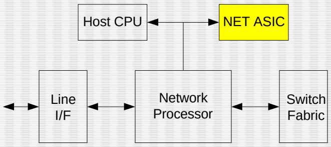
Definition - Interfaces

- Packet bandwidth determines maximum wire speed processing
- Memory bandwidth determines what can be done to meet wire speed processing

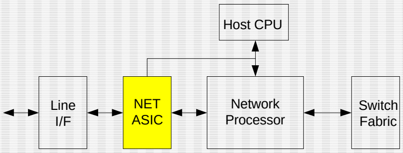


Definition - Programmability

- Stand-alone processors
  - Purpose-built NP
  - Networking-enhanced processor
- Co-processors (“NET ASICs”)
  - Configurable
  - Fixed-function engine



Look-aside configuration

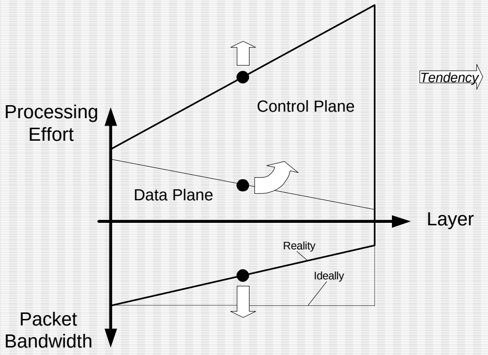


Flow-through configuration

Definition - Programmability

- Stand-alone processor
  - Fetch/Execute its own instruction set
  - Provides packet interfaces directly on chip
  - Types:
    - Purpose-built NPs
      - Execute an instruction stream in response to receiving a packet
      - Usually used in the data plane
      - High-bandwidth packet interfaces
    - Networking-enhanced processors
      - Traditional ISA
      - Networking peripherals integrated
      - Usually used in control plane and as general processors
      - Low/Medium bandwidth packet interfaces
- Co-processor
  - Offload some processing from the control-plane processor
  - They are fed instructions and/or data from the control-plane processor
  - Typically work in a request/response mode
  - Types:
    - Configurable (packet buffer manager, queuing manager)
    - Fixed-function engines (address look up, encryption, statistic gathering)

Definition - Summary

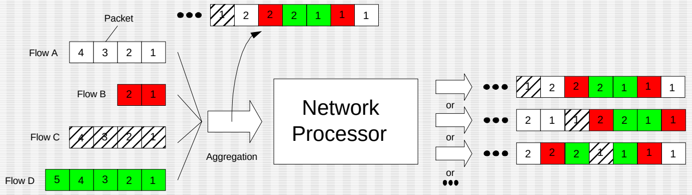


Workload - Outline

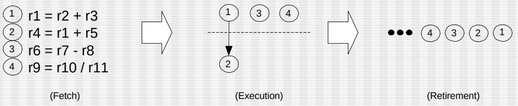
- Packet-based processing
- Memory bottleneck
- Applications
- Benchmarking

Workload – Packet based

- Internet protocol allows individual packets to be processed in any order (the receiver has the burden to put the packets back in order again)
  - Processing of individual packets is <u>fairly</u> independent
  - For performance: send the processed packets in order within the same flow

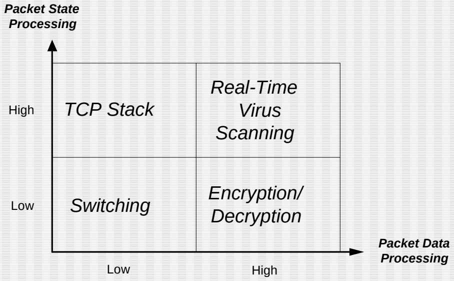


Analogy to a OOO general purpose processor:



Workload – Packet based

- Dependencies among packets within the same flow
  - NP has to maintain some kind of state
  - The more state modification, the higher the likelihood of processing stalling (like dependencies in typical pipelines)



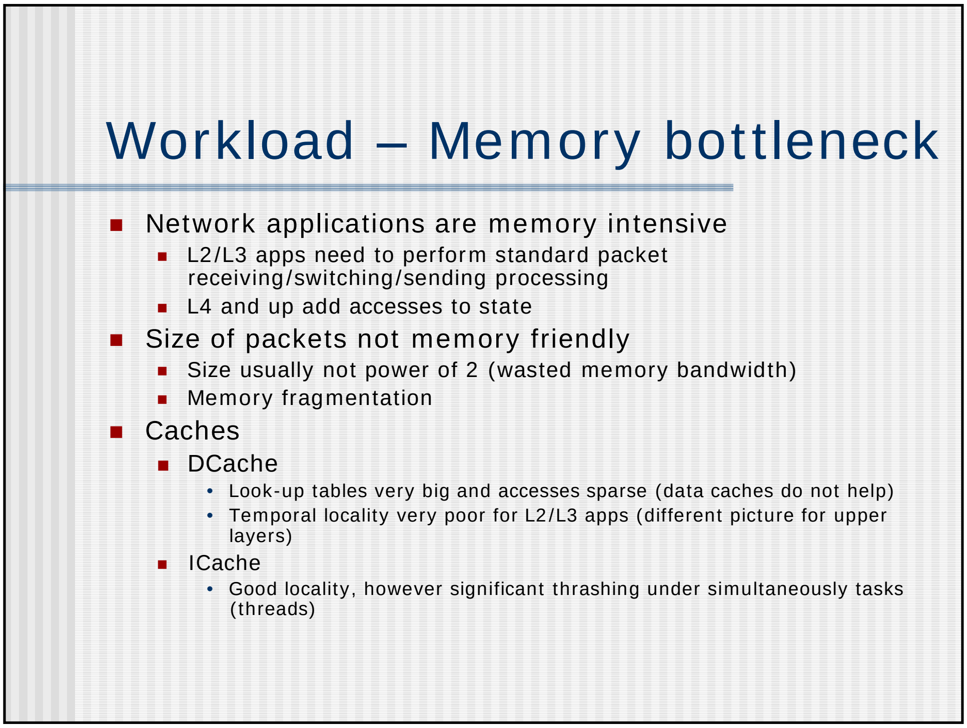
Packet State Processing	Low	High
High	TCP Stack	Real-Time Virus Scanning
Low	Switching	Encryption/ Decryption

Workload – Packet based

- Issue: time budgets per cell/packet arrival at high wire speeds
  - Budget relaxed if packets are large (except if payload is processed)

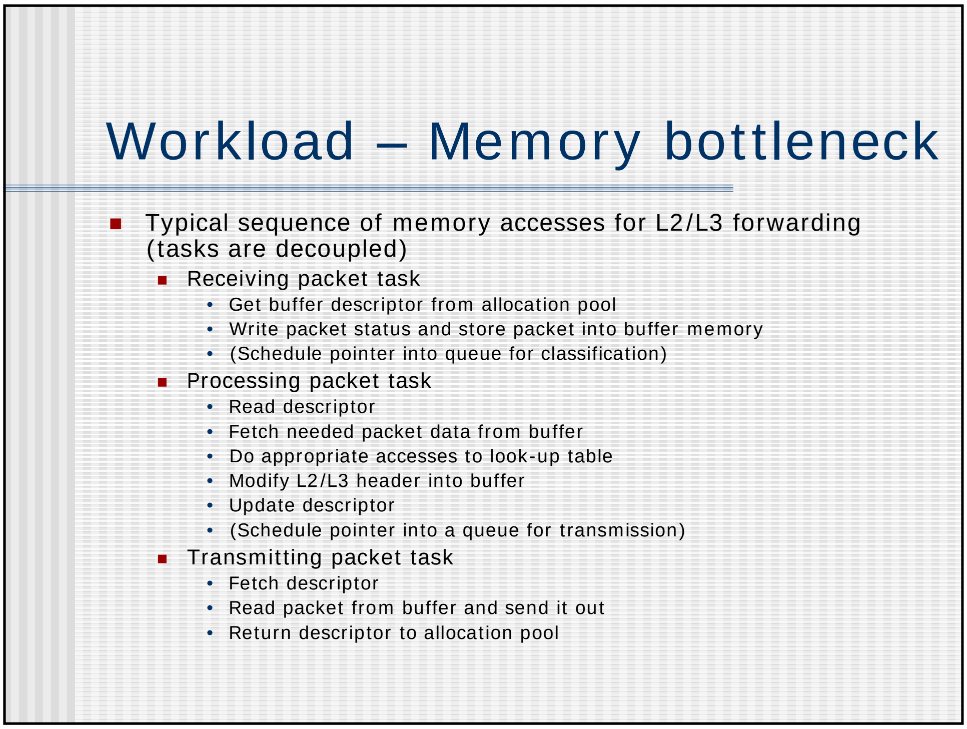
Line I/F	***Bandwidth***	***Budget (time)***	***Budget (cycles)***	
			@ 300MHz	***@ 1GHz***
Ethernet	10 Mbps	50us	15K	50K
1 Gig Ethernet	1 Gbps	500ns	150	500
10 Gig Ethernet	10 Gbps	50ns	15	50
OC-768	40 Gbps	13ns	4	13

(Assuming 64-byte packets)
(Approx. 60 RISC instructions needed for basic L3 forwarding)



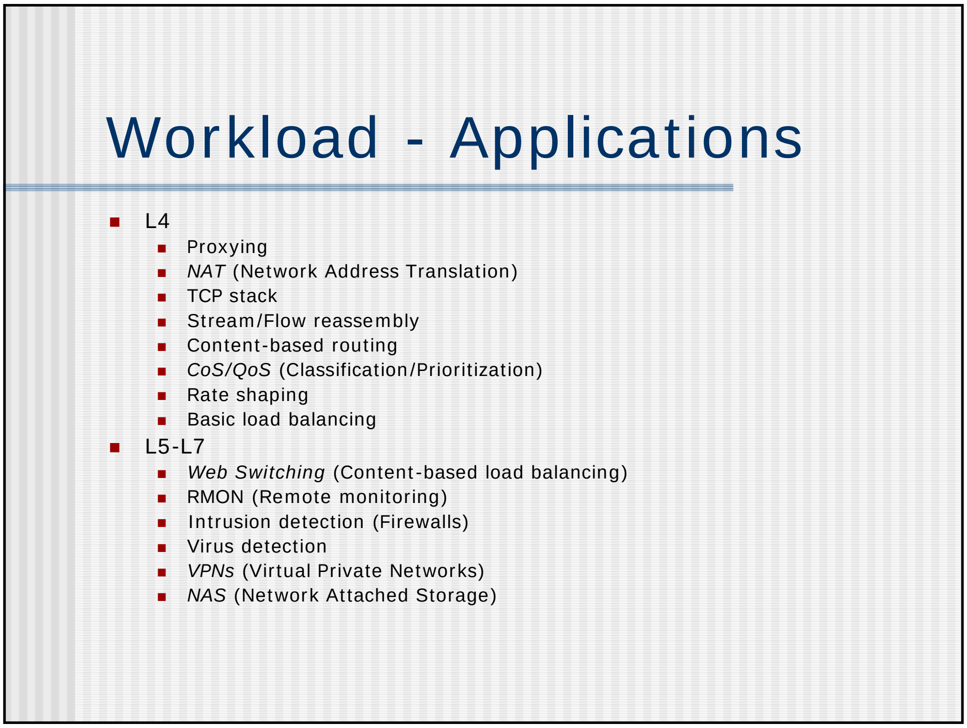
Workload – Memory bottleneck

- Network applications are memory intensive
  - L2/L3 apps need to perform standard packet receiving/switching/sending processing
  - L4 and up add accesses to state
- Size of packets not memory friendly
  - Size usually not power of 2 (wasted memory bandwidth)
  - Memory fragmentation
- Caches
  - DCache
    - Look-up tables very big and accesses sparse (data caches do not help)
    - Temporal locality very poor for L2/L3 apps (different picture for upper layers)
  - ICache
    - Good locality, however significant thrashing under simultaneously tasks (threads)



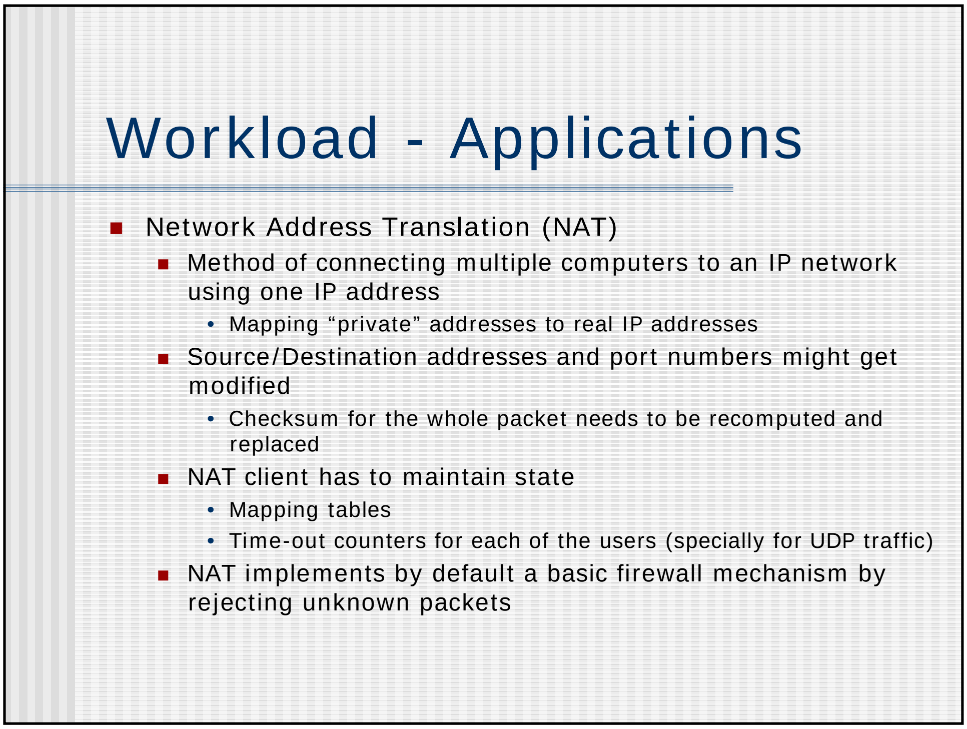
Workload – Memory bottleneck

- Typical sequence of memory accesses for L2/L3 forwarding (tasks are decoupled)
  - Receiving packet task
    - Get buffer descriptor from allocation pool
    - Write packet status and store packet into buffer memory
    - (Schedule pointer into queue for classification)
  - Processing packet task
    - Read descriptor
    - Fetch needed packet data from buffer
    - Do appropriate accesses to look-up table
    - Modify L2/L3 header into buffer
    - Update descriptor
    - (Schedule pointer into a queue for transmission)
  - Transmitting packet task
    - Fetch descriptor
    - Read packet from buffer and send it out
    - Return descriptor to allocation pool



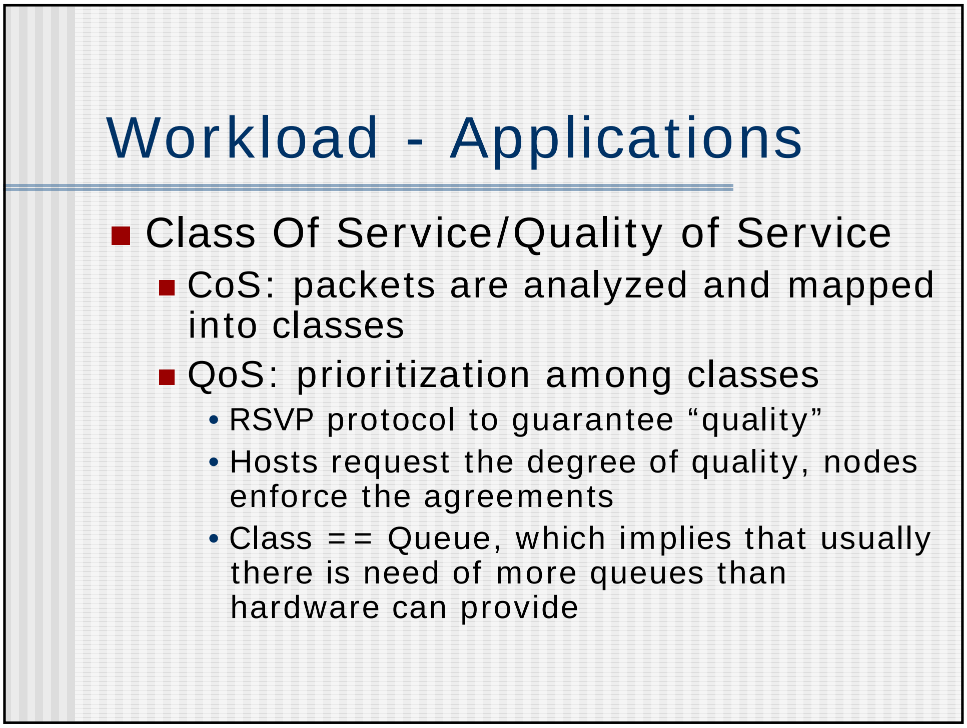
Workload - Applications

- L4
  - Proxying
  - *NAT* (Network Address Translation)
  - TCP stack
  - Stream/Flow reassembly
  - Content-based routing
  - *CoS/QoS* (Classification/Prioritization)
  - Rate shaping
  - Basic load balancing
- L5-L7
  - *Web Switching* (Content-based load balancing)
  - RMON (Remote monitoring)
  - Intrusion detection (Firewalls)
  - Virus detection
  - *VPNs* (Virtual Private Networks)
  - *NAS* (Network Attached Storage)



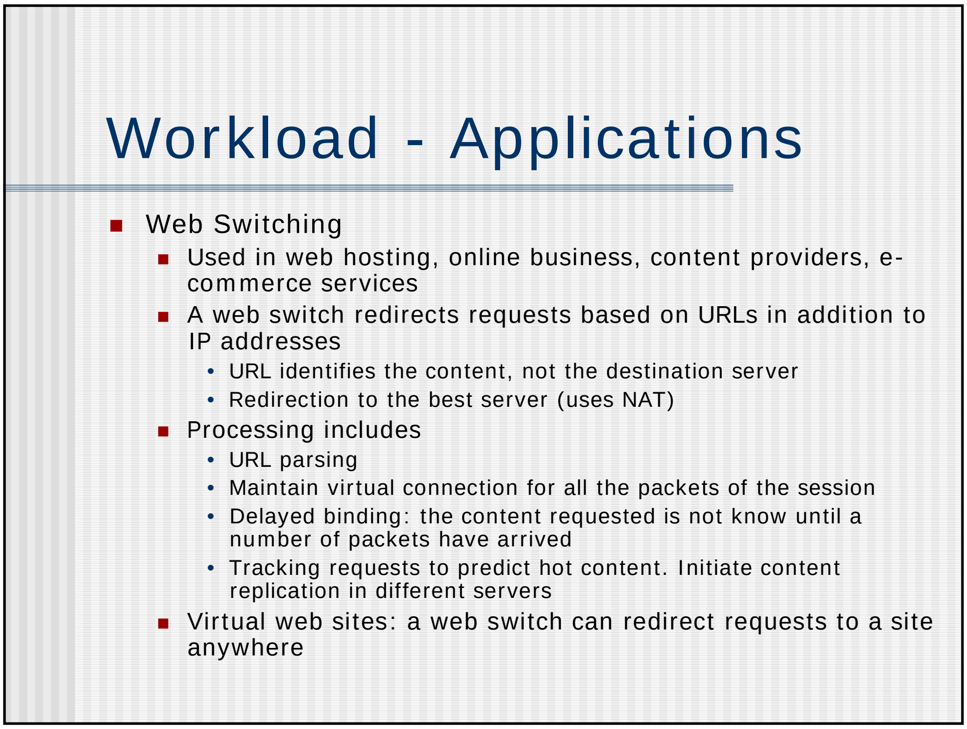
Workload - Applications

- Network Address Translation (NAT)
  - Method of connecting multiple computers to an IP network using one IP address
    - Mapping “private” addresses to real IP addresses
  - Source/Destination addresses and port numbers might get modified
    - Checksum for the whole packet needs to be recomputed and replaced
  - NAT client has to maintain state
    - Mapping tables
    - Time-out counters for each of the users (specially for UDP traffic)
  - NAT implements by default a basic firewall mechanism by rejecting unknown packets



Workload - Applications

- Class Of Service/Quality of Service
  - CoS: packets are analyzed and mapped into classes
  - QoS: prioritization among classes
    - RSVP protocol to guarantee “quality”
    - Hosts request the degree of quality, nodes enforce the agreements
    - Class == Queue, which implies that usually there is need of more queues than hardware can provide



Workload - Applications

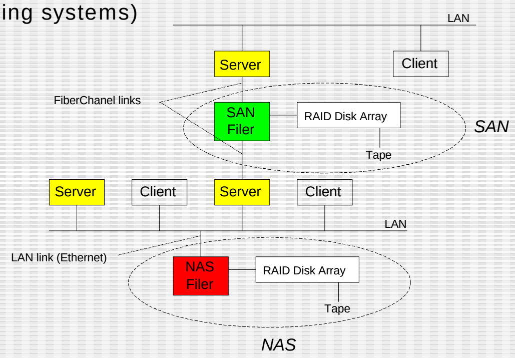
- Web Switching
  - Used in web hosting, online business, content providers, e-commerce services
  - A web switch redirects requests based on URLs in addition to IP addresses
    - URL identifies the content, not the destination server
    - Redirection to the best server (uses NAT)
  - Processing includes
    - URL parsing
    - Maintain virtual connection for all the packets of the session
    - Delayed binding: the content requested is not know until a number of packets have arrived
    - Tracking requests to predict hot content. Initiate content replication in different servers
  - Virtual web sites: a web switch can redirect requests to a site anywhere

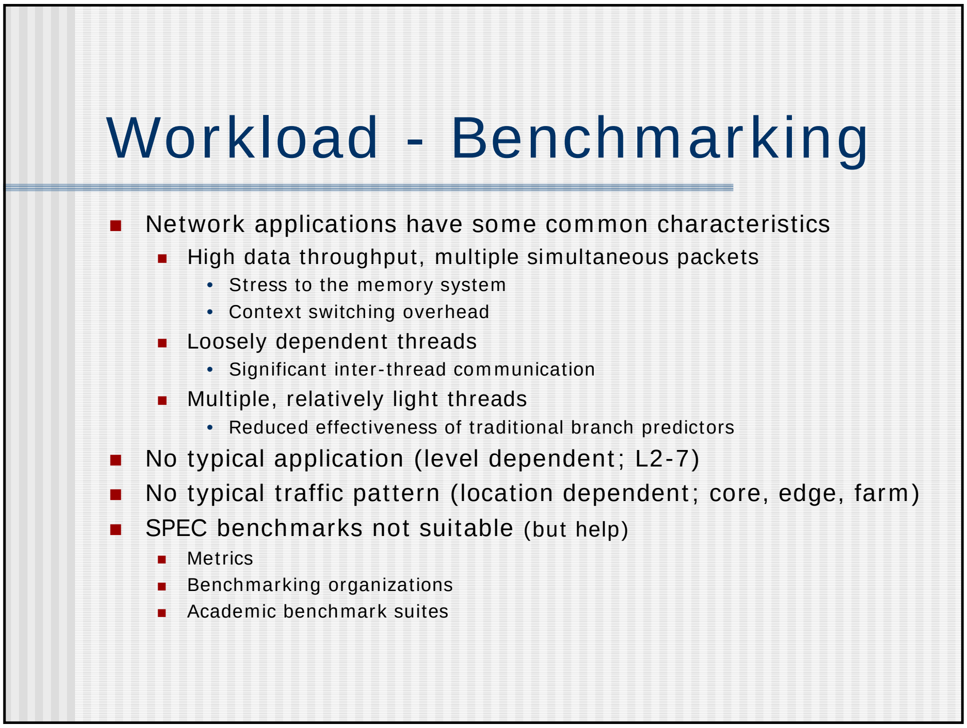
Workload - Applications

- Virtual Private Networks (VPNs)
  - Goal: to use public networks privately
  - 4 key objectives tackled by 2 mechanisms
    - IPSec
      - *Integrity* (data is not changed)
      - *Authenticity* (sender “signs” the packets); MD5
      - *Replay protection* (duplications are not welcome)
    - 3DES
      - *Confidentiality* (encryption/decryption)
  - VPN nodes at the edges need to insert/remove the variable length Authentication Header after the IP header

Workload - Applications

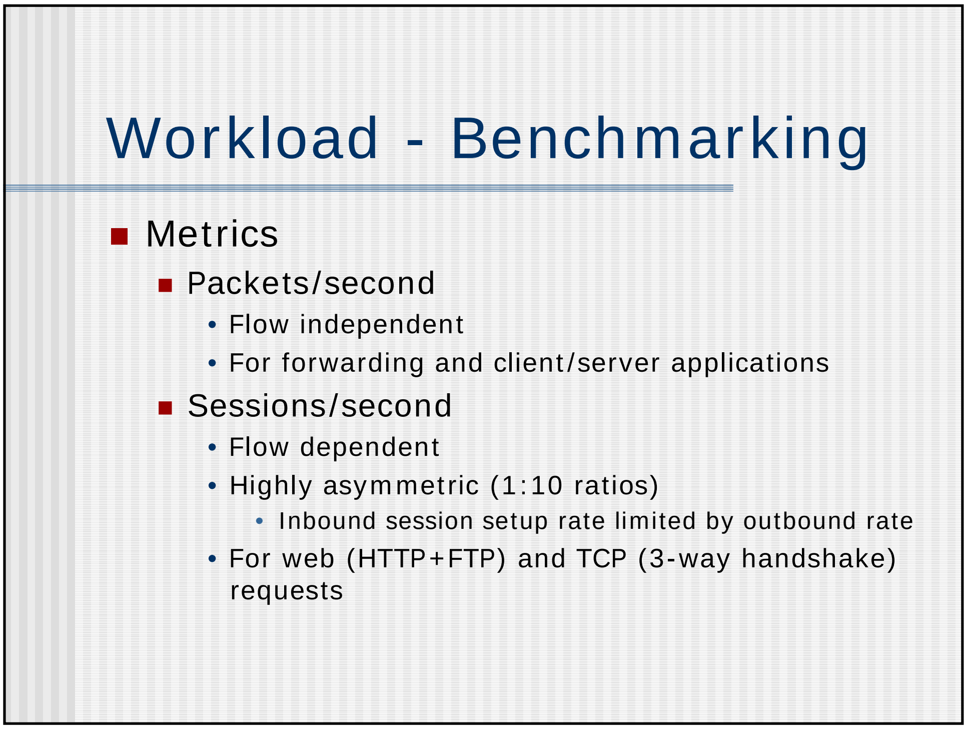
- Network Attached Storage (NAS)
  - Evolution of NFS: offload servers from file management tasks
- Platform independent (network decouples file system and servers operating systems)





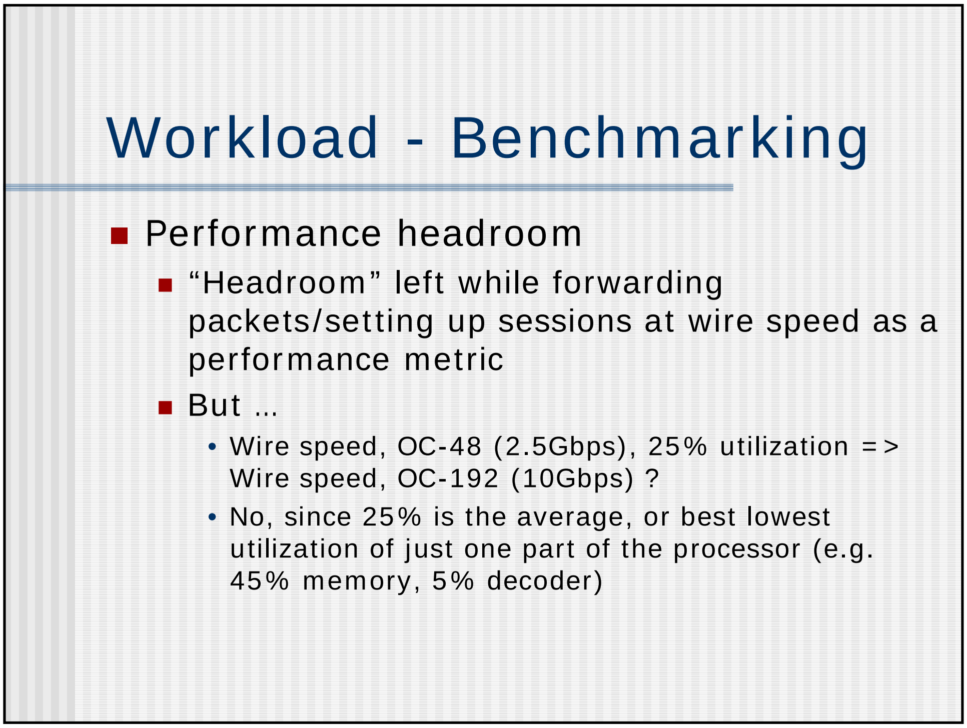
Workload - Benchmarking

- Network applications have some common characteristics
  - High data throughput, multiple simultaneous packets
    - Stress to the memory system
    - Context switching overhead
  - Loosely dependent threads
    - Significant inter-thread communication
  - Multiple, relatively light threads
    - Reduced effectiveness of traditional branch predictors
- No typical application (level dependent; L2-7)
- No typical traffic pattern (location dependent; core, edge, farm)
- SPEC benchmarks not suitable (but help)
  - Metrics
  - Benchmarking organizations
  - Academic benchmark suites



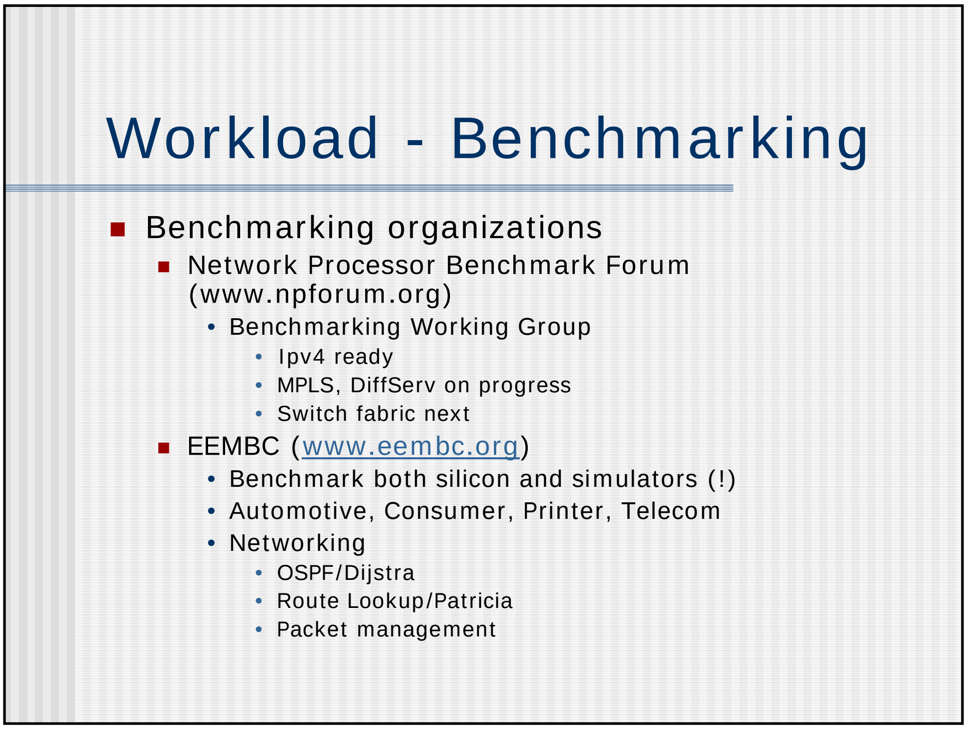
Workload - Benchmarking

- Metrics
  - Packets/second
    - Flow independent
    - For forwarding and client/server applications
  - Sessions/second
    - Flow dependent
    - Highly asymmetric (1:10 ratios)
      - Inbound session setup rate limited by outbound rate
    - For web (HTTP+FTP) and TCP (3-way handshake) requests



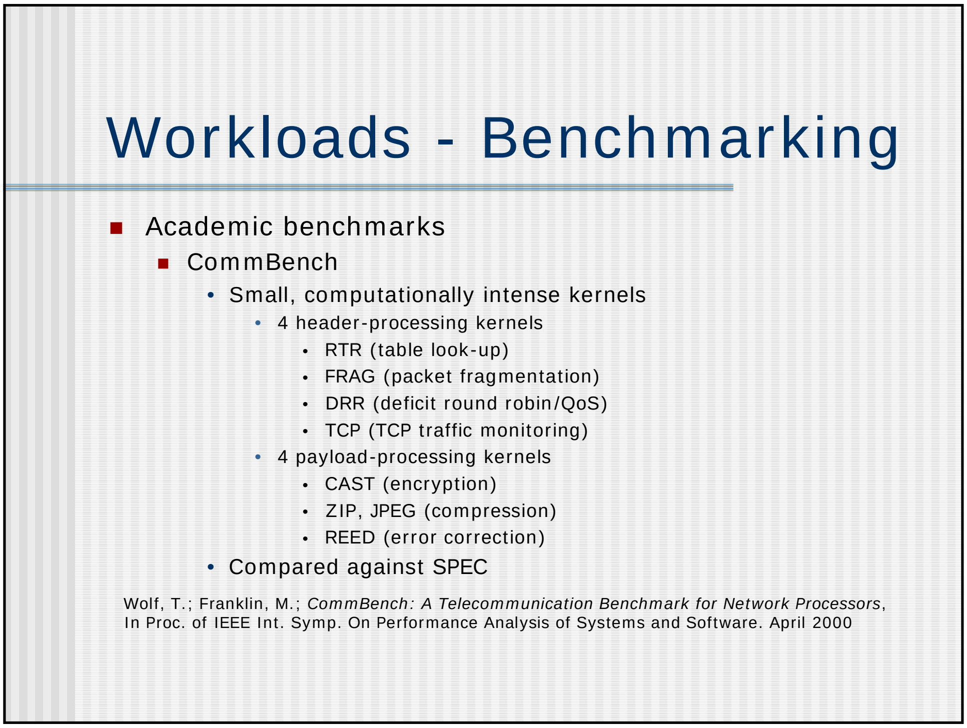
Workload - Benchmarking

- Performance headroom
  - “Headroom” left while forwarding packets/setting up sessions at wire speed as a performance metric
  - But …
    - Wire speed, OC-48 (2.5Gbps), 25% utilization => Wire speed, OC-192 (10Gbps) ?
    - No, since 25% is the average, or best lowest utilization of just one part of the processor (e.g. 45% memory, 5% decoder)



Workload - Benchmarking

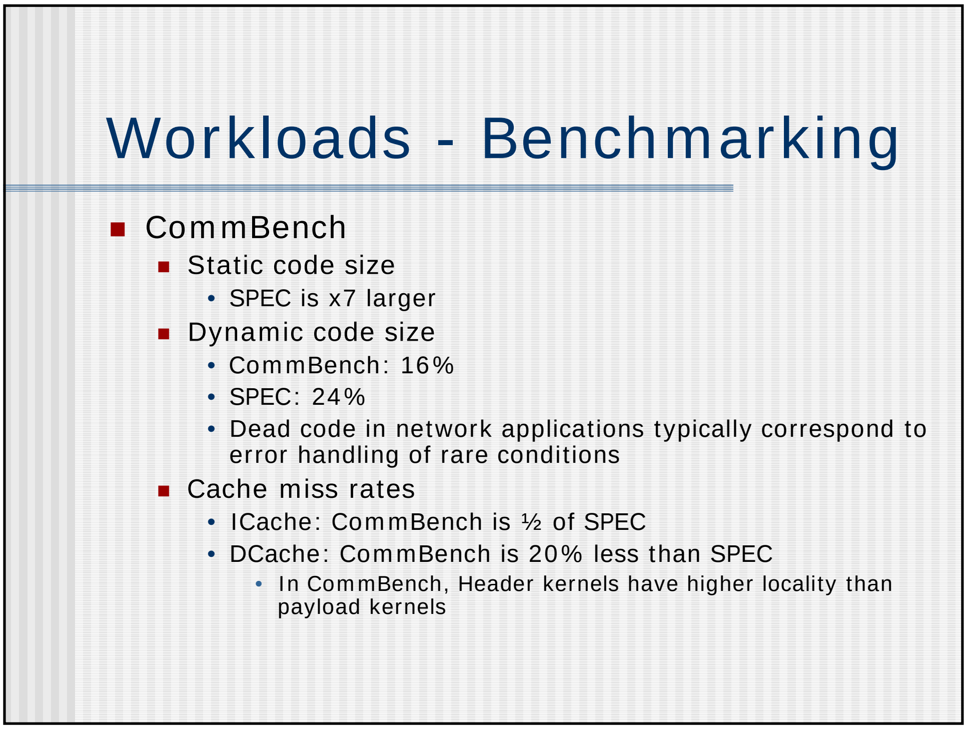
- Benchmarking organizations
  - Network Processor Benchmark Forum (www.npforum.org)
    - Benchmarking Working Group
      - Ipv4 ready
      - MPLS, DiffServ on progress
      - Switch fabric next
  - EEMBC (www.eembc.org)
    - Benchmark both silicon and simulators (!)
    - Automotive, Consumer, Printer, Telecom
    - Networking
      - OSPF/Dijstra
      - Route Lookup/Patricia
      - Packet management



Workloads - Benchmarking

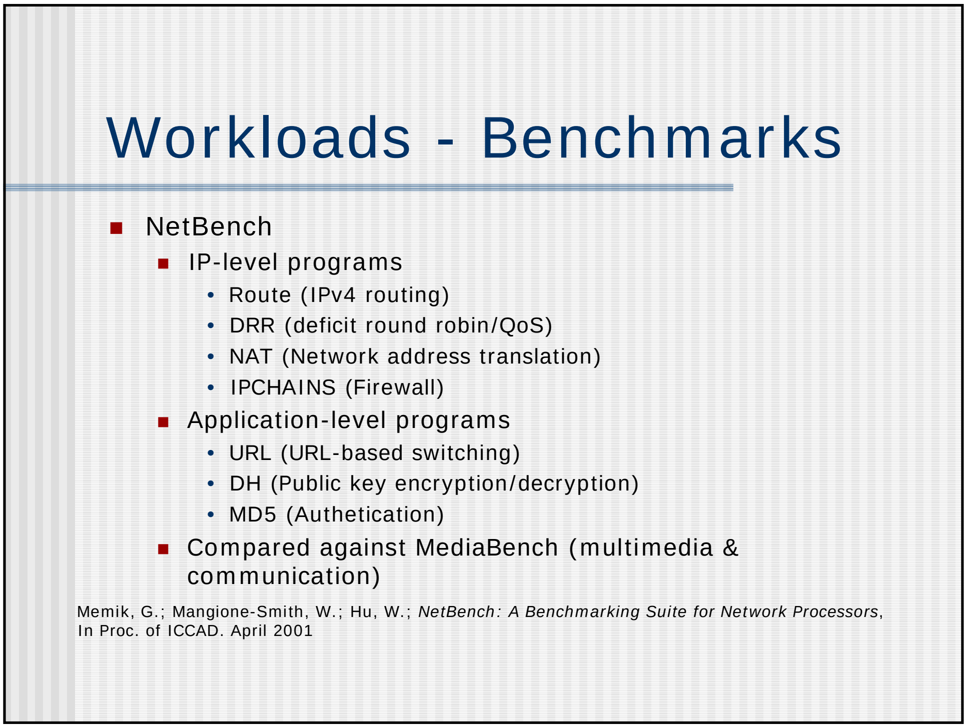
- Academic benchmarks
  - CommBench
    - Small, computationally intense kernels
      - 4 header-processing kernels
        - RTR (table look-up)
        - FRAG (packet fragmentation)
        - DRR (deficit round robin/QoS)
        - TCP (TCP traffic monitoring)
      - 4 payload-processing kernels
        - CAST (encryption)
        - ZIP, JPEG (compression)
        - REED (error correction)
    - Compared against SPEC

Wolf, T.; Franklin, M.; *CommBench: A Telecommunication Benchmark for Network Processors*, In Proc. of IEEE Int. Symp. On Performance Analysis of Systems and Software. April 2000



Workloads - Benchmarking

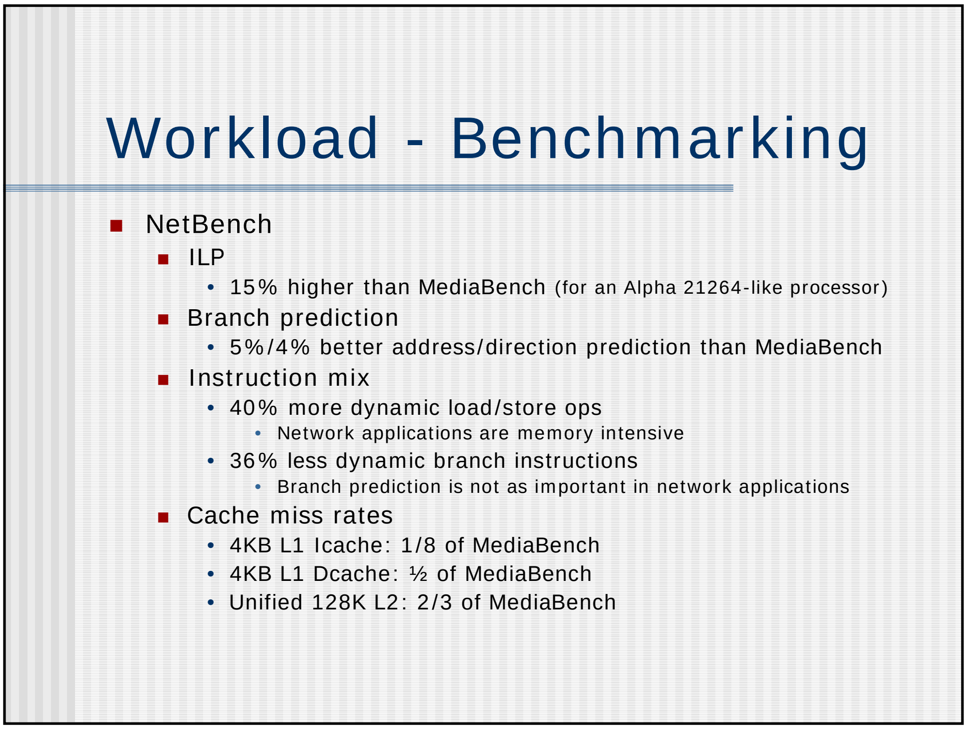
- CommBench
  - Static code size
    - SPEC is x7 larger
  - Dynamic code size
    - CommBench: 16%
    - SPEC: 24%
    - Dead code in network applications typically correspond to error handling of rare conditions
  - Cache miss rates
    - ICache: CommBench is ½ of SPEC
    - DCache: CommBench is 20% less than SPEC
      - In CommBench, Header kernels have higher locality than payload kernels



Workloads - Benchmarks

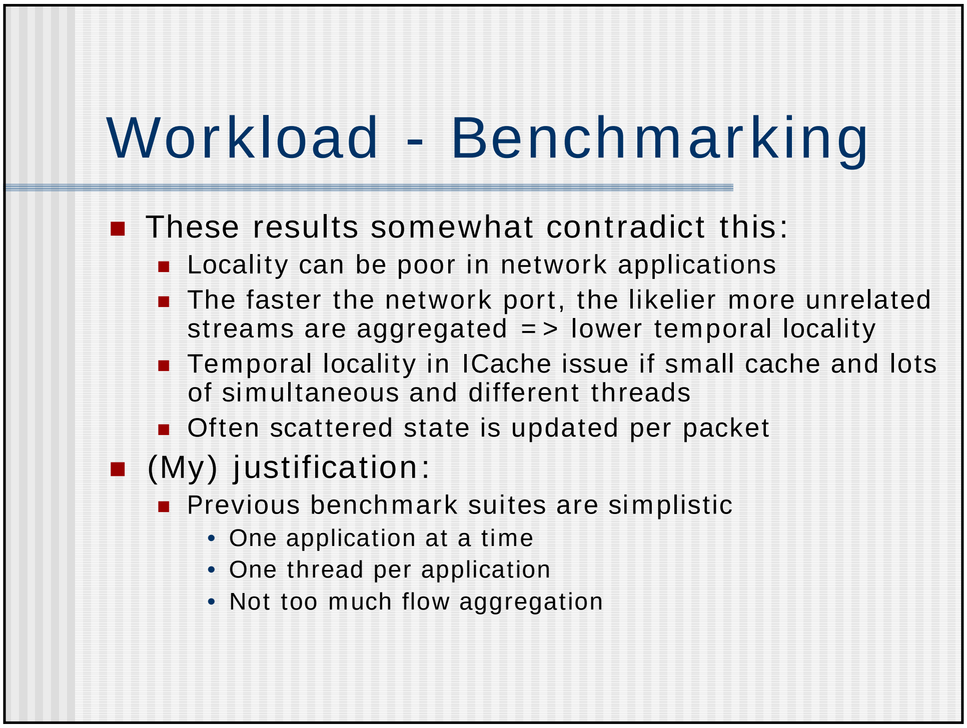
- NetBench
  - IP-level programs
    - Route (IPv4 routing)
    - DRR (deficit round robin/QoS)
    - NAT (Network address translation)
    - IPCHAINS (Firewall)
  - Application-level programs
    - URL (URL-based switching)
    - DH (Public key encryption/decryption)
    - MD5 (Authetication)
  - Compared against MediaBench (multimedia & communication)

Memik, G.; Mangione-Smith, W.; Hu, W.; *NetBench: A Benchmarking Suite for Network Processors*, In Proc. of ICCAD. April 2001



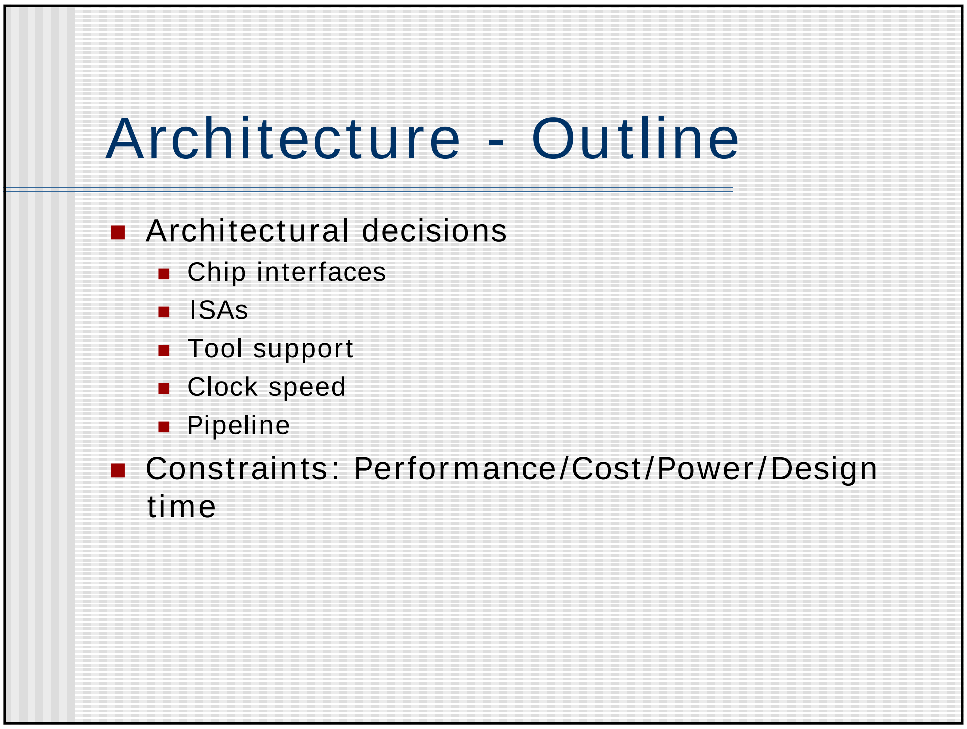
Workload - Benchmarking

- NetBench
  - ILP
    - 15% higher than MediaBench (for an Alpha 21264-like processor)
  - Branch prediction
    - 5%/4% better address/direction prediction than MediaBench
  - Instruction mix
    - 40% more dynamic load/store ops
      - Network applications are memory intensive
    - 36% less dynamic branch instructions
      - Branch prediction is not as important in network applications
  - Cache miss rates
    - 4KB L1 Icache: 1/8 of MediaBench
    - 4KB L1 Dcache: ½ of MediaBench
    - Unified 128K L2: 2/3 of MediaBench



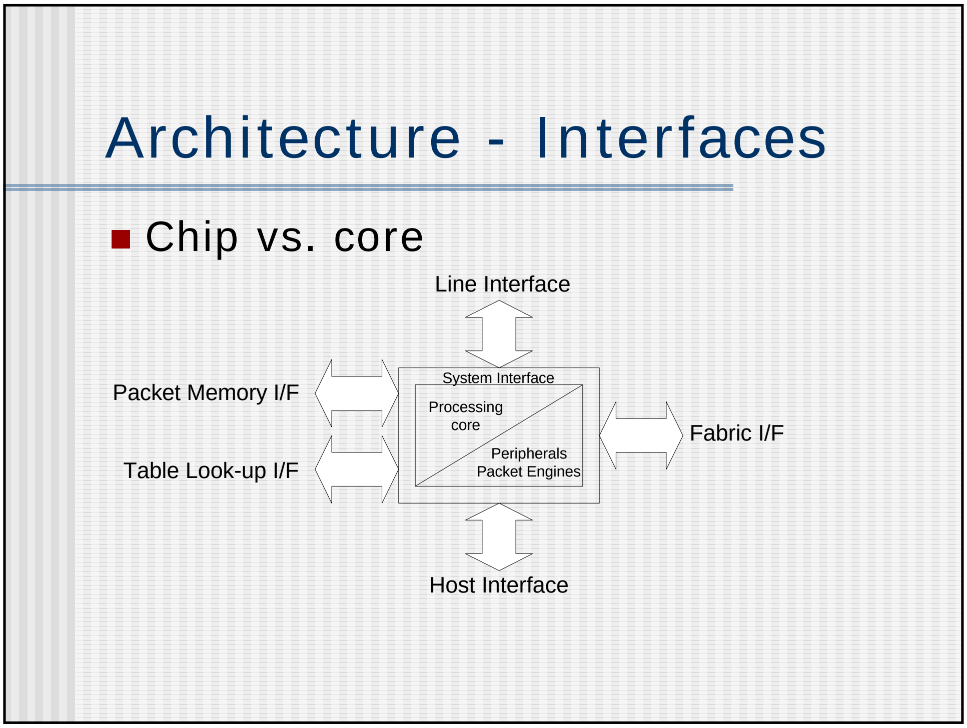
Workload - Benchmarking

- These results somewhat contradict this:
  - Locality can be poor in network applications
  - The faster the network port, the likelier more unrelated streams are aggregated => lower temporal locality
  - Temporal locality in ICache issue if small cache and lots of simultaneous and different threads
  - Often scattered state is updated per packet
- (My) justification:
  - Previous benchmark suites are simplistic
    - One application at a time
    - One thread per application
    - Not too much flow aggregation



Architecture - Outline

- Architectural decisions
  - Chip interfaces
  - ISAs
  - Tool support
  - Clock speed
  - Pipeline
- Constraints: Performance/Cost/Power/Design time



Architecture - Interfaces

- Chip vs. core

Line Interface

Packet Memory I/F

Table Look-up I/F

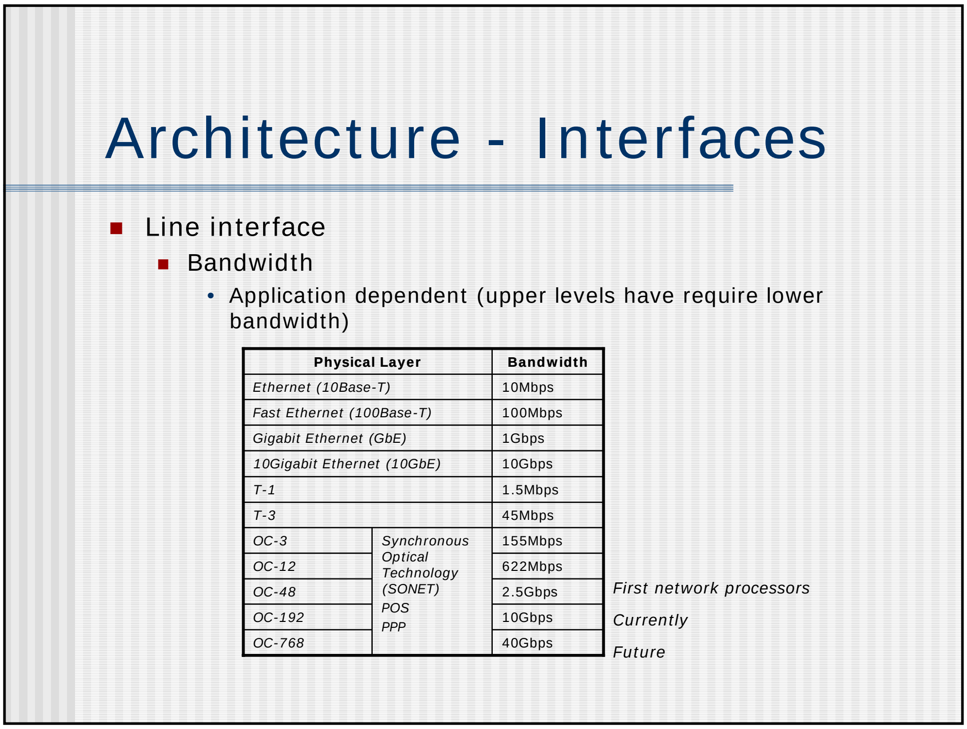
System Interface

Processing core

Peripherals Packet Engines

Fabric I/F

Host Interface



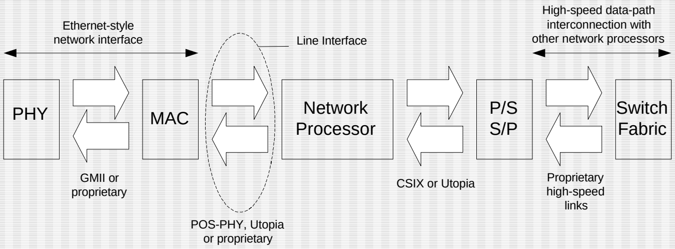
Architecture - Interfaces

- Line interface
  - Bandwidth
    - Application dependent (upper levels have require lower bandwidth)

Physical Layer		Bandwidth	
Ethernet (10Base-T)		10Mbps	
Fast Ethernet (100Base-T)		100Mbps	
Gigabit Ethernet (GbE)		1Gbps	
10Gigabit Ethernet (10GbE)		10Gbps	
T-1		1.5Mbps	
T-3		45Mbps	
OC-3	*Synchronous Optical Technology (SONET)* *POS* *PPP*	155Mbps	
OC-12		622Mbps	
OC-48		2.5Gbps	*First network processors*
OC-192		10Gbps	*Currently*
OC-768		40Gbps	*Future*

Architecture - Interfaces

- Line interface
  - Protocol: connect to external MAC or integrate?

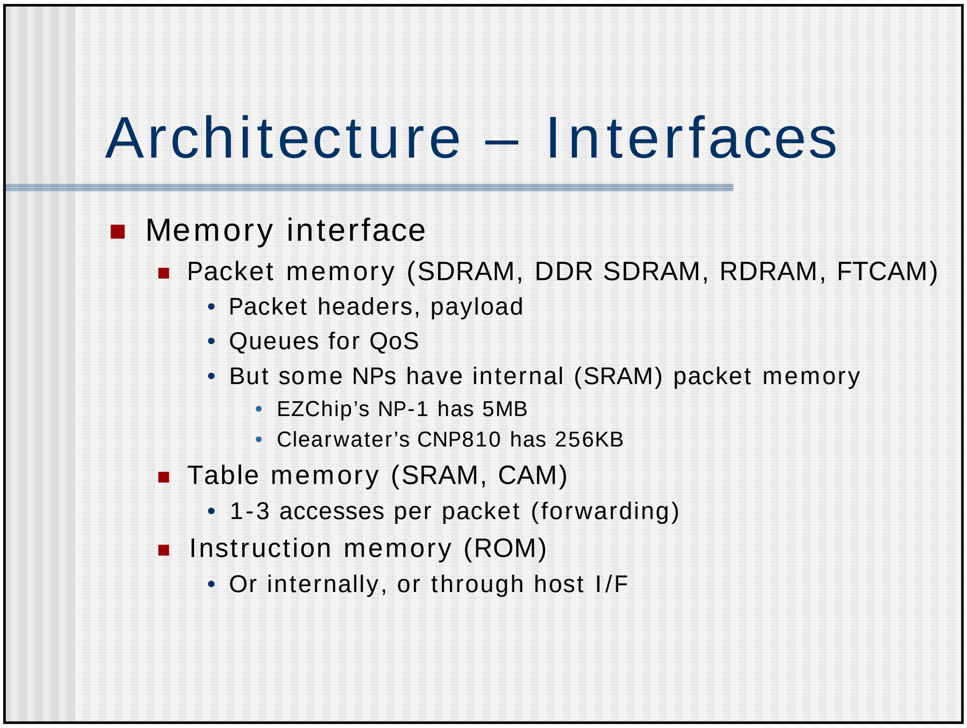

Ethernet-style network interface
High-speed data-path interconnection with other network processors
Line Interface
PHY
MAC
Network Processor
P/S S/P
Switch Fabric
GMII or proprietary
POS-PHY, Utopia or proprietary
CSIX or Utopia
Proprietary high-speed links

- Switch-fabric interface
  - Ideally narrow to reduce chip pin count
  - Not well defined yet
  - Parallel/Serial, Serial/Parallel converters
  - Bandwidth similar to line interface for routers (+25% overhead)

Architecture - Interfaces

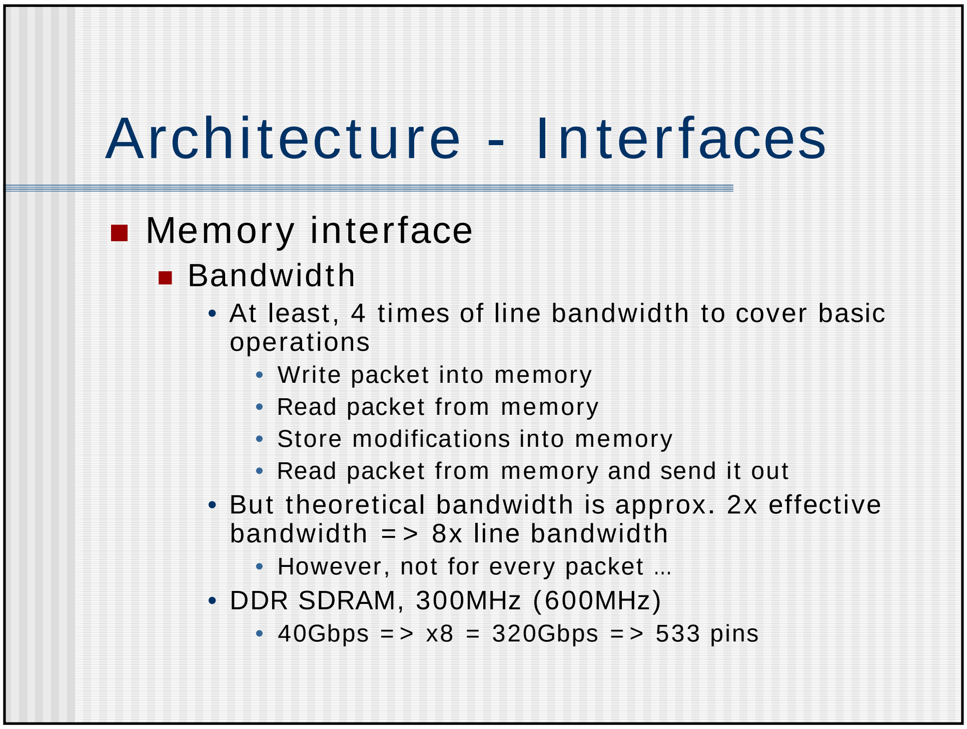
- Host interface
  - Used to connect to a general purpose processor
  - Host processor handles control-plane functions
  - Usually PCI, but others are emerging, with higher bandwidth, narrower and more scalable

Standard		**Bandwidth**
32-bit PCI	Wide	< 2Gbps
64-bit PCI		< 4 Gbps
64-bit PCI-X		< 8 Gbps
Infiniband	Narrow	> 4 Gbps
HyperTransport		> 6.4 Gbps
RapidI/O		> 16 Gbps



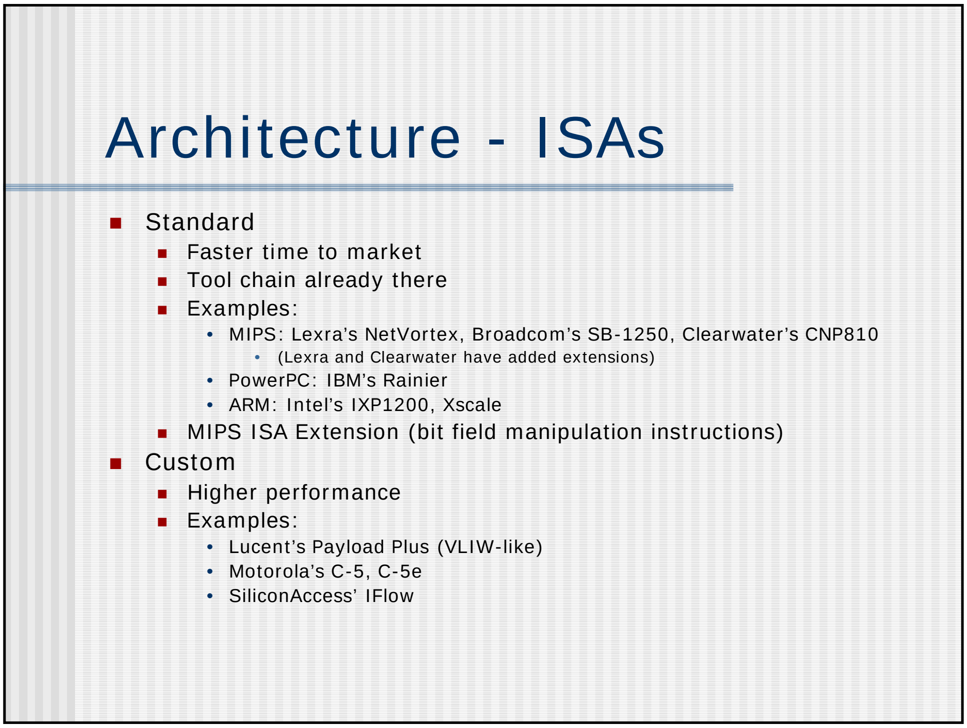
Architecture – Interfaces

- Memory interface
  - Packet memory (SDRAM, DDR SDRAM, RDRAM, FTCAM)
    - Packet headers, payload
    - Queues for QoS
    - But some NPs have internal (SRAM) packet memory
      - EZChip’s NP-1 has 5MB
      - Clearwater’s CNP810 has 256KB
  - Table memory (SRAM, CAM)
    - 1-3 accesses per packet (forwarding)
  - Instruction memory (ROM)
    - Or internally, or through host I/F



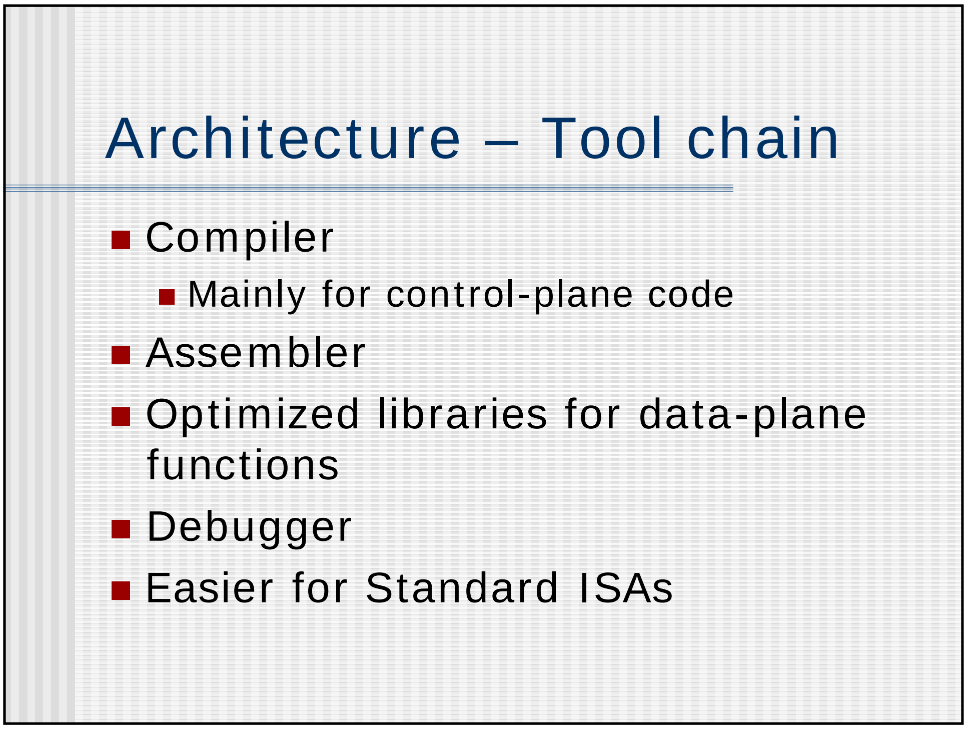
Architecture - Interfaces

- Memory interface
  - Bandwidth
    - At least, 4 times of line bandwidth to cover basic operations
      - Write packet into memory
      - Read packet from memory
      - Store modifications into memory
      - Read packet from memory and send it out
    - But theoretical bandwidth is approx. 2x effective bandwidth => 8x line bandwidth
      - However, not for every packet …
    - DDR SDRAM, 300MHz (600MHz)
      - 40Gbps => x8 = 320Gbps => 533 pins



Architecture - ISAs

- Standard
  - Faster time to market
  - Tool chain already there
  - Examples:
    - MIPS: Lexra’s NetVortex, Broadcom’s SB-1250, Clearwater’s CNP810
      - (Lexra and Clearwater have added extensions)
    - PowerPC: IBM’s Rainier
    - ARM: Intel’s IXP1200, Xscale
  - MIPS ISA Extension (bit field manipulation instructions)
- Custom
  - Higher performance
  - Examples:
    - Lucent’s Payload Plus (VLIW-like)
    - Motorola’s C-5, C-5e
    - SiliconAccess’ IFlow



Architecture – Tool chain

- Compiler
  - Mainly for control-plane code
- Assembler
- Optimized libraries for data-plane functions
- Debugger
- Easier for Standard ISAs

Architecture – Clock speed

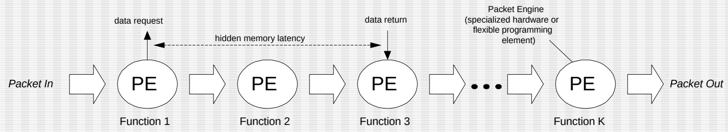
- Time to market vs. performance
  - Standard cells vs. full custome
- Memory bottlneck
  - High MHz have diminishing returns
- Examples
  - Intel's IXP1200: 166MHz
  - Clearwater's CNP810: 300MHz
  - Broadcom SB-1250: 600MHz
  - Lexra's LX8380: 420MHz
  - Intel's XScale: 1.4GHz

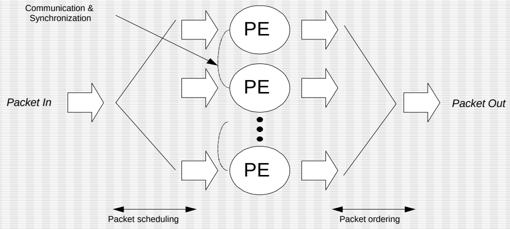
Architecture - Core

- Different implementations
  - Packet Engine + Control (PE)
  - *Context pipelining* + *Control* (CP)
  - Standard Super Scalar (SS)
  - Simultaneous Multithreading (SMT)
  - *Chip Multiprocessing* (CMP)
    - PE, CP, SS, SMT

Architecture - Core

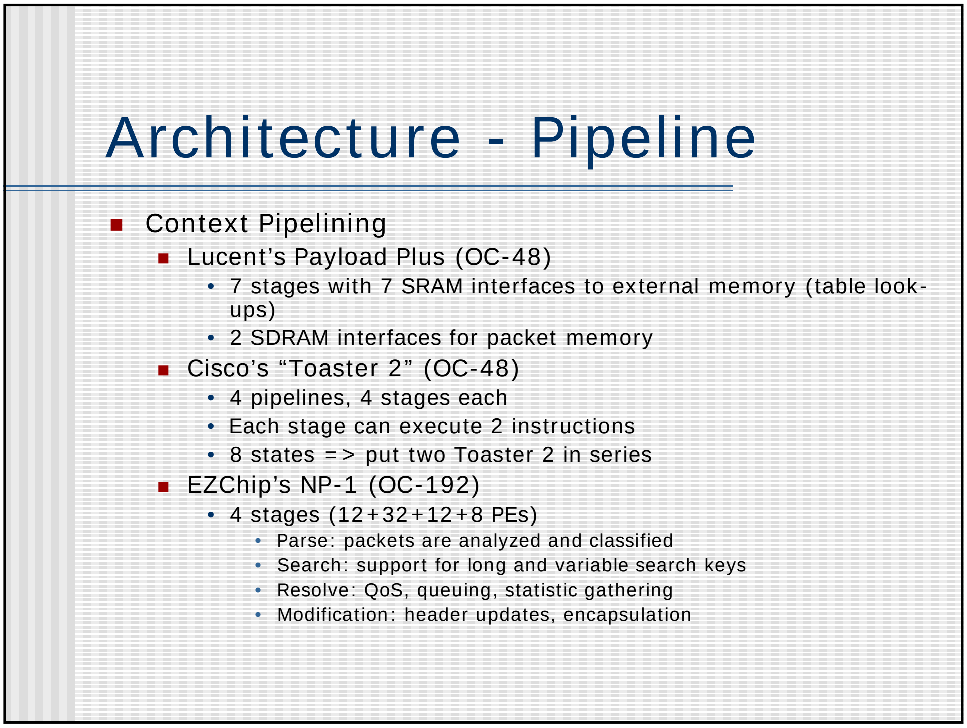
- Context pipelining vs. chip multiprocessing





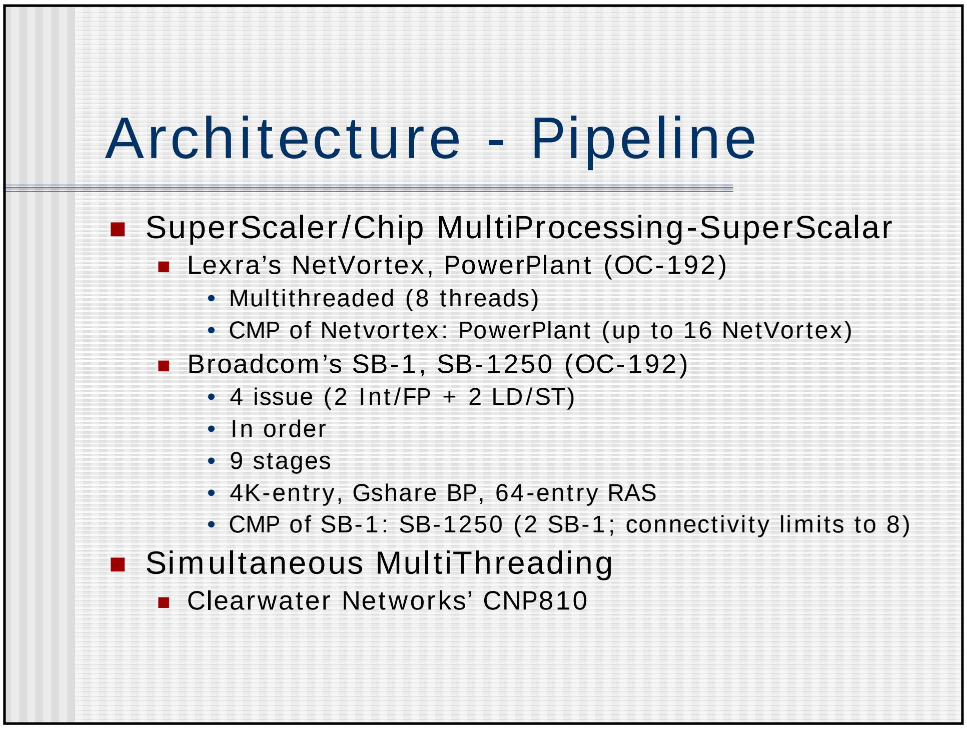
Architecture - Core

- Chip Multiprocessor with Packet Engines
  - Intel’s IXP1200 (OC12-OC-48)
    - 6 PEs
    - PE has proprietary ISA, and some level of multithreading
  - Intel’s XScale (OC-192)
    - Can also function as in CP mode
    - Almost nothing disclosed
  - IBM’s Rainier (OC-48)
    - 16 PEs divided into 8 units
    - 4 Table Look-up engines (1/2 per unit)
    - 56 co-processors (7 per unit); not programmable
      - Frame classifiers
      - Checksum
      - String movement
      - …
  - Motorola C-5e (OC-48)
    - 16 Pes with 4 threads each
    - 4 co-processors
      - Table Look-up manager
      - Queue manager
      - Fabric manager
      - Buffer manager



Architecture - Pipeline

- Context Pipelining
  - Lucent's Payload Plus (OC-48)
    - 7 stages with 7 SRAM interfaces to external memory (table look-ups)
    - 2 SDRAM interfaces for packet memory
  - Cisco's "Toaster 2" (OC-48)
    - 4 pipelines, 4 stages each
    - Each stage can execute 2 instructions
    - 8 states => put two Toaster 2 in series
  - EZChip's NP-1 (OC-192)
    - 4 stages (12+32+12+8 PEs)
      - Parse: packets are analyzed and classified
      - Search: support for long and variable search keys
      - Resolve: QoS, queuing, statistic gathering
      - Modification: header updates, encapsulation



Architecture - Pipeline

- SuperScaler/Chip MultiProcessing-SuperScalar
  - Lexra's NetVortex, PowerPlant (OC-192)
    - Multithreaded (8 threads)
    - CMP of Netvortex: PowerPlant (up to 16 NetVortex)
  - Broadcom's SB-1, SB-1250 (OC-192)
    - 4 issue (2 Int/FP + 2 LD/ST)
    - In order
    - 9 stages
    - 4K-entry, Gshare BP, 64-entry RAS
    - CMP of SB-1: SB-1250 (2 SB-1; connectivity limits to 8)
- Simultaneous MultiThreading
  - Clearwater Networks' CNP810

Architecture - Pipeline

- SS vs. FGMT vs. CMP-SS vs. SMT
  - Benchmarks
    - IPv4 packet forwarding
    - IPsec
    - 3DES
    - MD5
  - Analysis without OS overhead, single thread
  - Analysis with OS overhead, but still single thread
  - Variable number of functional units/contexts/processors (1-8)

Crowley, P.; Fiuczynski, M.; Baer, J-L.; Bershad, B.; *Characterizing Processor Architectures for Programmable Network Interfaces*, Dept. of CSE, University of Washington, Seattle, US
In Proc. of the Int. Conf. On Supercomputing. May 2000

Architecture - Pipeline

- SS vs. FGMT vs. CMP-SS vs. SMT
  - Summary of results without overhead (MPPS)

Application	**Pipeline architecture**			
	SS	**FGMT**	**CMP**	**SMT**
IPv4	12	12	*22*	21
MD5	0.35	0.41	*1.7*	*1.7*
3DES	0.033	0.033	*0.095*	*0.095*

@ 500MHz
8 FUs/Processors/Contexts

  - CMP/SMT outperform SS/FGMT 2-4 times

Architecture - Pipeline

- SS vs. FGMT vs. CMP-SS vs. SMT
  - Summary of results with overhead (MPPS)

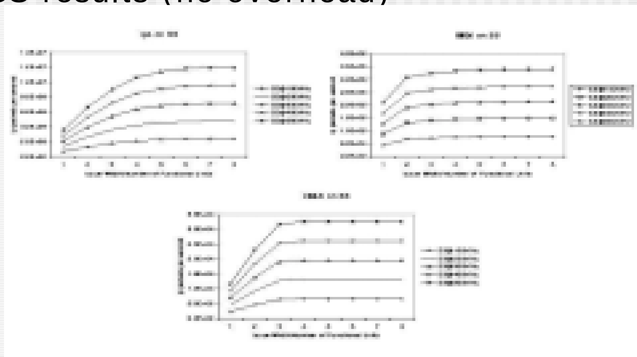
Application	Pipeline architecture			
	SS	FGMT	CMP	SMT
IPv4	0.65	*1.25*	0.6	1.15
MD5	0.25	0.35	0.65	*0.75*
3DES	0.025	0.025	0.080	*0.085*

@ 500MHz
8 FUs/Processors/Contexts

  - SMT usually outperforms the rest architectures
  - Overhead is very significant for short applications

Architecture - Pipeline

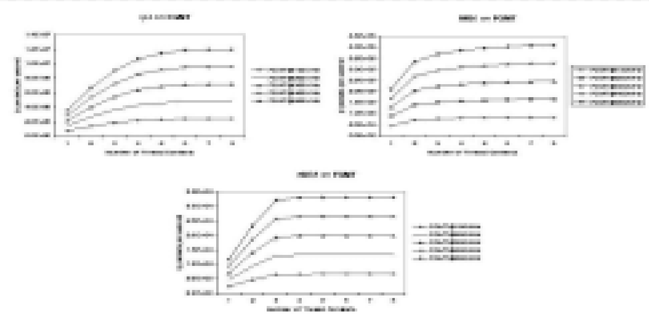
- SS results (no overhead)



  - Higher ILP in IPv4

Architecture - Pipeline

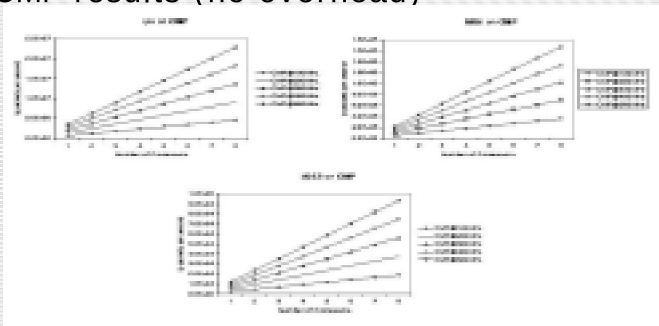
- FGMT results (no overhead)



  - Some improvement on the less compute bound one

Architecture - Pipeline

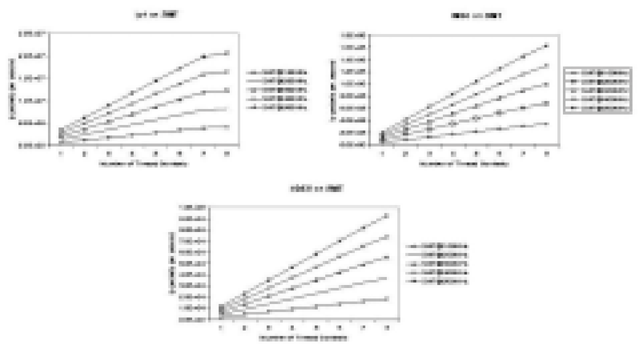
- CMP results (no overhead)



  - Single-issue cores replicated
  - Exploits thread-level parallelism => linear increase

Architecture - Pipeline

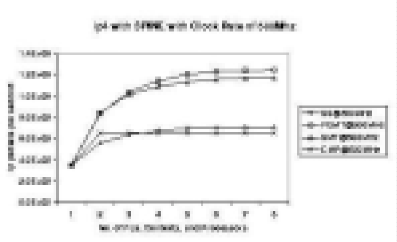
- SMT results (no overhead)



  - Exploits both ILP and thread-level parallelism

Architecture - Pipeline

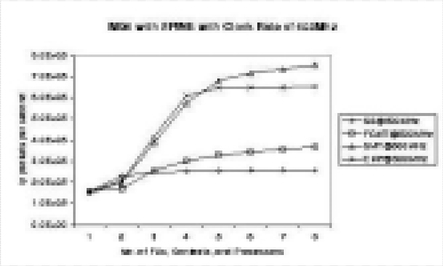
- IPv4: results with overhead



  - SMT vs. FGMT: SMT OS thread (usually idle) competing constantly for resources. FMGT only 1/8th of the time
  - SS limited by serializing processing of packets
  - CMP has only one context processing packets

Architecture - Pipeline

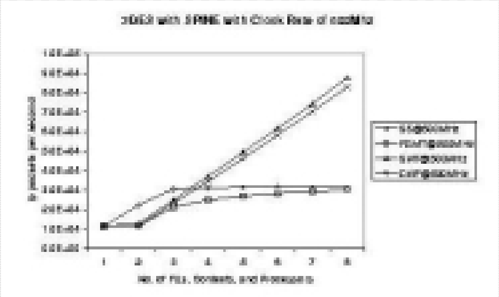
- MD5: results with overhead



  - CMP settles at the maximum performance of the OS thread
  - Lower # of FUs/Contexts => no unused resources => more effective to serialize processing

Architecture - Pipeline

- 3DES: results with overhead



  - 1/n of time FGMT gets idle (OS thread)

Clearwater Networks – Outline

- Overview
- CNP810 Network Services Processor
  - SMT core
  - NPX™ Instructions
  - PMU coprocessor
    - Packet Cache™
    - RTU
  - High-performance memory interface
  - System Implementations

Clearwater Networks

- Some slides from

  www.clearwaternetworks.com/clearwater_overview.pdf