

# Network Processors

Nevin Heintze  
Agere Systems

## Network Processors

---

- What are the packaging challenges for NPs?

**Caveat: I know very little about packaging.**

## Network Processors

---

- ~~What are the packaging challenges for NPs?~~
- *What is a network processor?*

## Network Processors

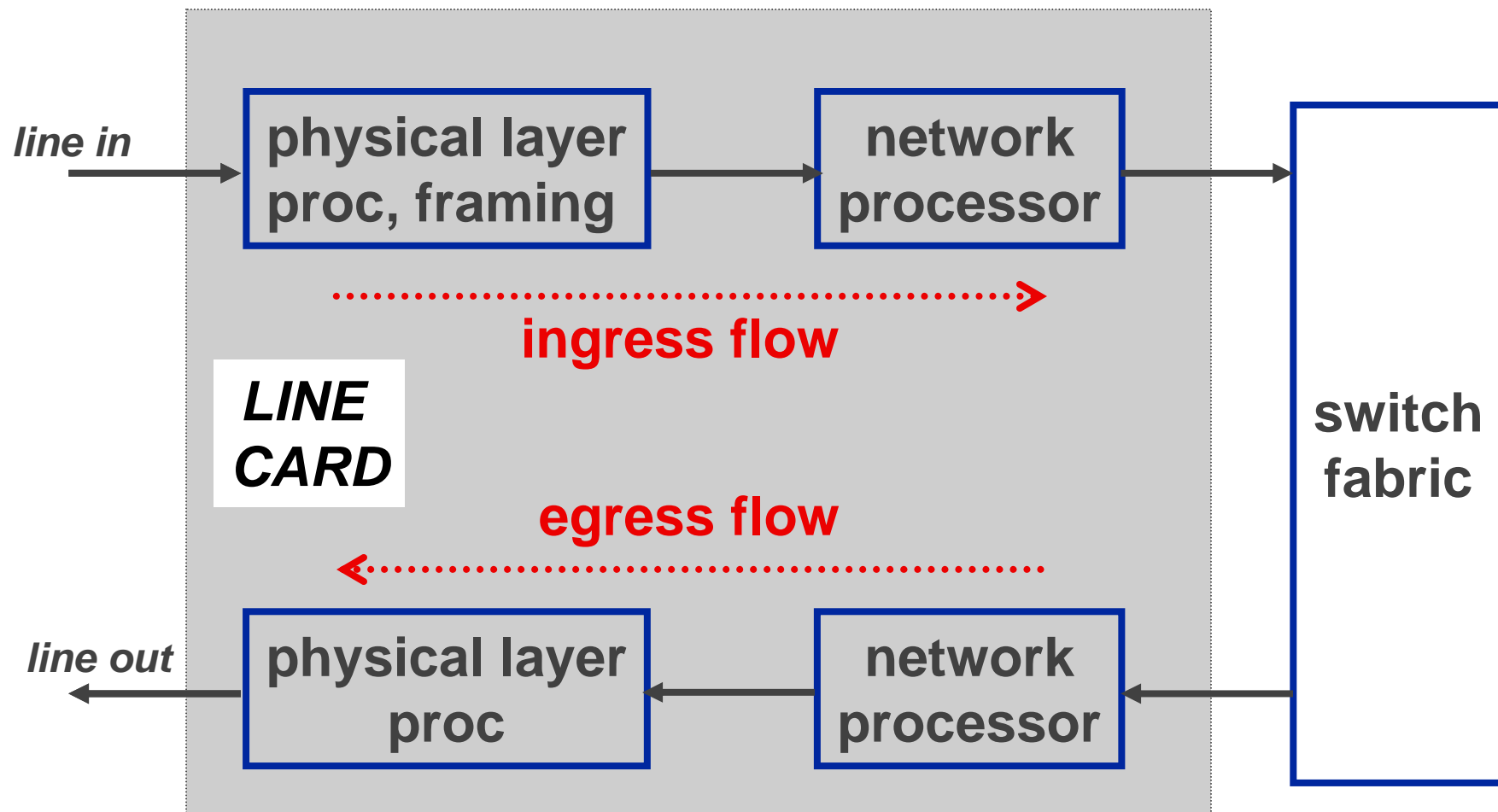
---

- What are the packaging challenges for NPs?
- *What is a network processor?*

### Plan


- applications **What?**
- architecture **How?**
- packaging **The consequences ...**

## Network node: router, DSLAM, ...



## History

---

- 80's: early routers were souped-up Unix boxes
  - couldn't keep up with network growth
- 90's: 'fast path' ASICs
  - offload the most compute intensive tasks
    - routing/classification, policing, traffic management, etc.
  - Unix box controls setup of the ASIC and any tasks that the fast path cannot handle
  - but: standard limitations of ASICs
- 00's: programmable fast-path ASICs  NPs

## The NP “value-proposition”

- multi-protocol support
- quicker turn-around
  - evolving protocols
- complex protocols
  - hard to get right in an ASIC

## **What is a Network Processor?**

---

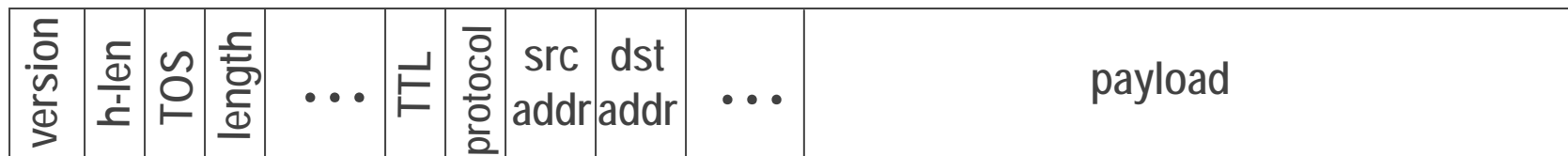
- **No simple answer**
- **NPs still in their infancy**
- **Many diverse approaches competing for survival**

## An Easier Question ...

### *What do Network Processors do?*

- **Classification**
  - look at headers and figure out what to do with this packet
  - routing, TCP flow classification, etc.

e.g. IPv4



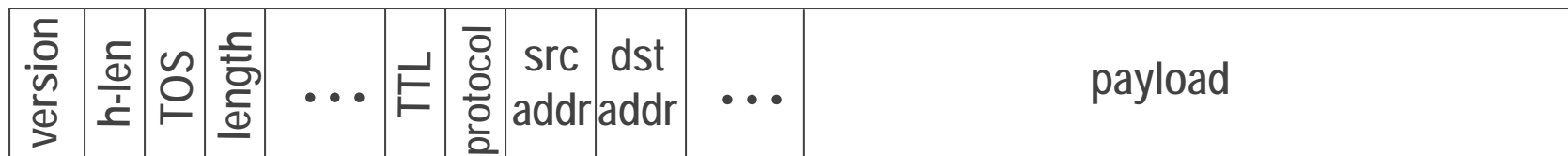
  
**check version = 4**

## An Easier Question ...

### *What do Network Processors do?*

- **Classification**
  - look at headers and figure out what to do with this packet
  - routing, TCP flow classification, etc.

e.g. IPv4



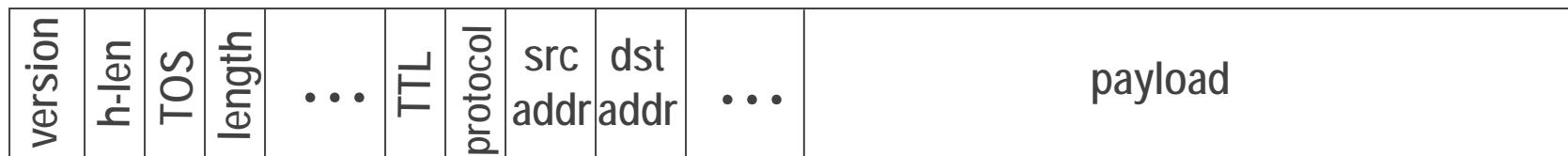
↑  
**check time-to-live > 0**

## An Easier Question ...

### *What do Network Processors do?*

- **Classification**
  - look at headers and figure out what to do with this packet
  - routing, TCP flow classification, etc.

e.g. IPv4



↑  
**check transport-layer protocol e.g. UDP, TCP, ICMP**

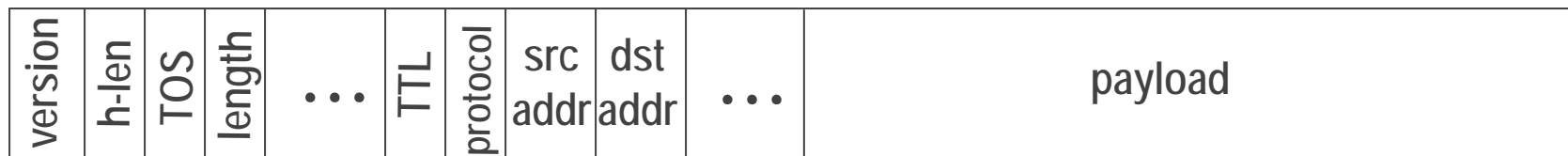
## An Easier Question ...

### *What do Network Processors do?*

---

- **Classification**
  - look at headers and figure out what to do with this packet
  - routing, TCP flow classification, etc.

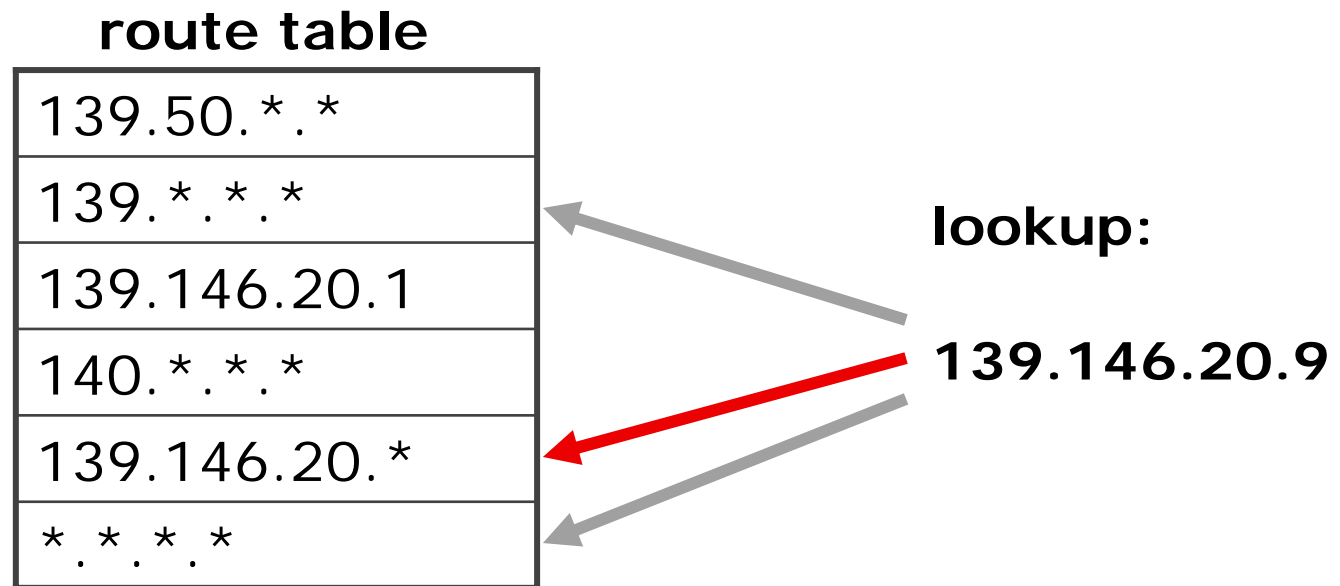
e.g. IPv4



↑  
**lookup route table to find “next-hop”**

## Routing: Longest-Prefix-Match

---



Routing tables can be big

- 100K routes, 1M routes soon
- caching "not useful"

## Two Ways to Implement Routing

- **Ternary CAM (TCAM)**
  - simple architectural solution
  - fast/simple to update and lookup; modest I/O
  - But: expensive, power & area hungry
- **Tries**
  - use commodity DRAM
    - cheaper, less power, etc. etc.
  - multiple memory reads per route lkp
    - high I/O needs
  - software support for updates

## Back to: What to NPs do?

---

- **Traffic management**
  - which pkt to send next?
  - traffic shaping
  - packet buffering (up to 100ms of traffic)
  - queue management

## Other typical NP operations

---

- packet modification
  - update header fields e.g. TTL, new MPLS label
  - update checksums
  - add new headers (encapsulation)
- policing: is this flow sending too many pkts?
- statistics
- segmentation and reassembly
- security
  - access control lists, firewalls
  - encryption
- 'deep classification' – URL switching, virus detection

## Example 1: Simple IP Router

---

- reassemble packet (framers are block-based)
- inspect headers
  - check protocol version field
  - lookup src ip in route table to find next-hop
  - check access control lists (ACLs) to check if packet is allowed
- statistics: packet counts
- queue packet and send to switch fabric
- update Time-To-Live field
- segment pkt into blocks and send to fabric

## Example 2: Stateful Firewall

---

In addition to IP Routing tasks:

- use TCP port numbers, in addition to src and dst IP addresses to identify what TCP connection this packet belongs to
- use the TCP flags to update the state of this packet's connection
- more sophisticated ACLs using TCP state

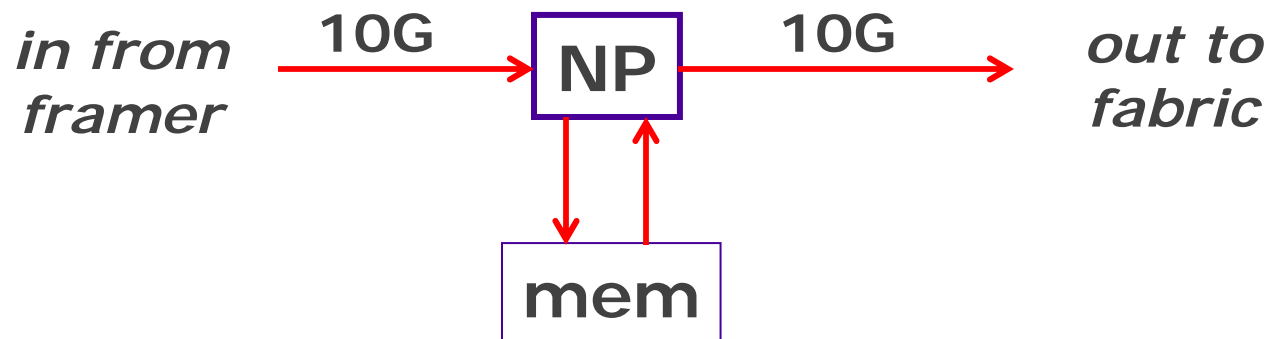
## Architectural Requirements

---

- **Lots of compute cycles**
  - route table lookups
  - updates to state
  - updates to packet
  - queue management
  - algorithms: traffic shaping & management
  - in excess of 1000 "RISC" cycles/pkt, excluding TM
  - assuming 1000 cycles/pkt and 40Byte pkts
    - **0.5 GOPS@155M, 7.8GOPS@2.5G, 31GOPS@10G**
- **Memory bandwidth & I/O .....**

## I/O Requirements: The Packet Path @ 10G

---



- need to get pkt in and out of memory
- need to deal with IP packets of arbitrary size
  - e.g. 65 byte pkts with 64 byte mem blocks
  - this inefficiency leads to a b/w multiplier of x2-x4
- typical design: external DRAM; 30-60G b/w

## I/O Requirements II

---

- Queues: need fast enqueue/dequeue
  - reassembling packets from framer blocks
  - pkt queues for traffic management
  - typical design: SRAM; ~10G b/w for 10G NP
- Classification: e.g. route table lookup
  - typical: 64-128 bits per packet
    - more for “deep-classification”, nested protocols
  - TCAM: lookaside interface
  - DRAM
    - 10-20G b/w off-chip DRAM
    - banking games can add x2 or x4

## I/O Requirements III

---

- Policing and traffic management
  - can tolerate latency
    - architecture choices
  - need: 20-40G b/w
- Lookaside/co-processor interfaces
  - classification, voice, security
  - these tend to be power-hungry e.g. SERDES

## I/O Summary @ 10G

---

- DRAM: 50-200G (RLDRAM,FCRAM,DDR DRAM)
- SRAM: 10-40G (QDR SRAM)
- I/O: 30-60G (POS-PHY, SPI-4) [SERDES]

**This is a lot of bandwidth!**

**... and this only covers ingress!**

## I/O Summary @ 10G

---

- **DRAM: 50-200G (RLDRAM,FCRAM,DDR DRAM)**
- **SRAM: 10-40G (QDR SRAM)**
- **I/O: 30-60G (POS-PHY, SPI-4) [SERDES]**

Critical NP Features That Impact I/O Needs:

- guaranteed line rate processing (40B+ pkts)
- support for ATM (ATM reassembly)
- classification capabilities; policing capabilities
- number of TM queues
- packet buffering capacity (up to 100ms)
- number of packet reassembly queues

## Consequences: High Pin Count @ 10G

- 700-1000 signal pins per device (NP, TM)
  - the bulk of these are for memory
  - I/O also important
- 1400+ total pins

## Summary

---

- Limited general agreement on NP architecture.
- Networking applications define NPs.
- NP architecture largely driven by memory interfaces (followed by pkt I/O interfaces).
- High pin counts, esp. at 5G, 10G
- High speed serial interfaces are increasingly important
  - tradeoff: pins for power/complexity

## The Future

---

### Many ways to address pin count issues

- High speed SERDES
  - need better models of package parasitics
- Multi-chip modules
  - combine NP and memory
  - reduce needs for external (off-module) memory
  - configurability?