Network Processor Programming

Hao Che
Department of Computer Science and Engineering
University of Texas at Arlington
USA

Two New Challenges in Router Design

1. Providing circuit switch like high availability: 5 9s (i.e., 99.999%) uptime

- Focused on control card high availability: number one R&D research expenditure reported for year 2002
- Two solutions (2002): *Nonstop Forwarding* (Cisco, Juniper, etc.) and *Nonstop Routing* (Alcatel, Avici, etc.)
- Largely failed to deal with software failures which constitute more than 60% failure situations
- A research area the networking research community may play an important role

This talk will focus on the second challenge →

Two New Challenges in Router Design

2. Network Processor Programming: A key differentiator of vendor solutions

Traditional routers were built based on:

- General purpose micro-processors: low speed
- Purposely built ASICs: limited programmability

Network Processor: a new species of microprocessor optimized for packet processing. The best of the two worlds

- Fully programmable
- Multi-gigabit processing speed

Programming network processors and managing their resources pose new challenges

Applications

- **RFC1812 Compliant IP Packet Processing**
  - L2/L3 header processing
  - IP forwarding table lookup
  - Access control list
  - TTL update
  - Checksum update
  - L2 encapsulation
  - Packet fragmentation/reassembly
  - IP Options field parsing
  - ICMP packet processing
- **Advanced IP Packet Processing**
  - MPLS: FEC to tunnel mapping, label popping/pushing, label swapping
  - DiffServ: traffic conditioning including metering, policing, marking/remarking
  - Policy/firewall filteirng
  - L3 VPN
  - NAT
  - ATM/FR interworking

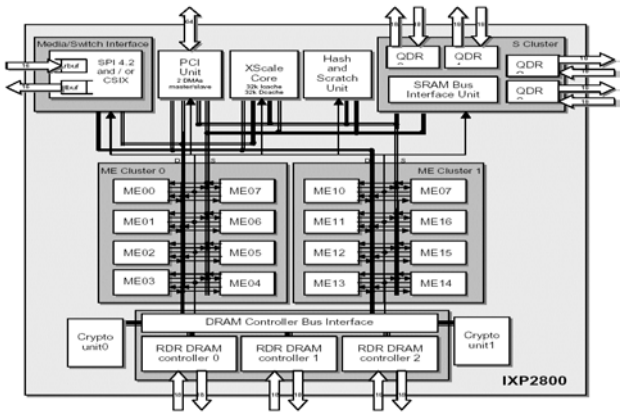
Applications

- **Challenging Applications**
  - IPSec
  - Flow classification
  - H323/SIP/dynamic NAT
  - MPLS pseudo-wire and L2 VPN
  - Active networking

A daunting, ever growing list of applications to be processed within 40ns at OC-192 POS line rate!

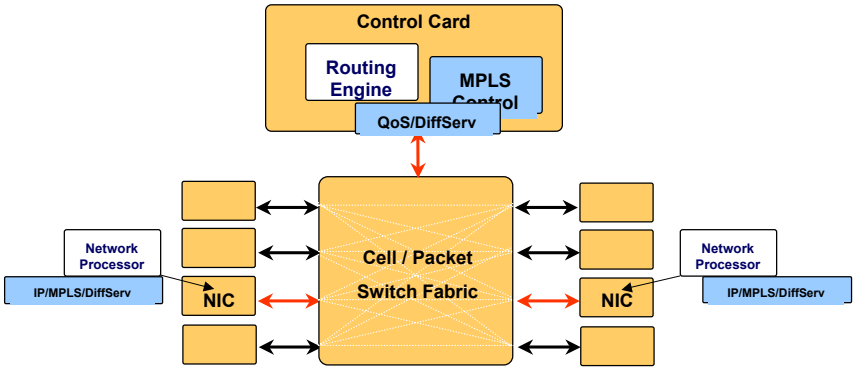
Network Processor: what's inside?

- **Micro-engines (MEs) can be configured to work in pipeline and/or parallel**
- **Each ME can be configured to support multiple number of threads**
- **Thread execution may be scheduled in various ways**
- **All the MEs share the same external resources**



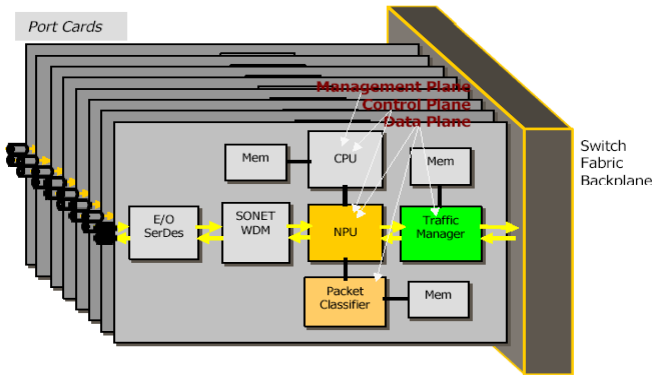
System Level View

- Fast Data Path: ingress NIC/NPU → egress NIC/NPU
- Global Slow Data Path: ingress NIC/NPU →Control Card → egress NIC/NPU



Card Level View

- Local Slow Data Path: NPU → Local CPU → NPU → Switch Fabric



Source: "Network Processors and Coprocessors for Broadband Network Applications," T. A. Chu, ACORN Networks

Important Issues

1. Mapping applications to specific network processor configurations

- Existing research focuses on mapping one or a few data path flows to a specific configuration, mostly based on cycle-accurate simulation. In practice, however, a network processor may have to process a mixture of a good number of data path flows
- A systematic approach is needed. We are
  - building a model to capture the important critical data path functions for various data path flows
  - building a library of modeled data path flows
  - building a generic NP model
  - developing a simulation tool to allow fast performance analysis of various mappings

Important Issues

2. Mapping applications to various router components:

- Existing research solely focuses on mapping applications to an isolated NPU and lacks of system level view
- Mapping applications to NPU should be an integral part of system level architecture design: E.g., running ARP locally in each NIC → L2 encapsulation in egress NPU → IP forwarding/policy table lookup in egress → outgoing packet loss in egress NPU if the thread scheduler has given higher priority to ingress packets
- Need a system level framework

Important Issues

3. Resource contention: database update vs database access

- Due to the fact that *all* MEs share the same external memory resources, the traditional approach, i.e., *locking* the memory for database update, can significantly impact the data path processing performance. E.g., locking TCAM for per rule update can lead to the loss of hundreds of packets at OC-192 line rate
- We designed a general TCAM database update algorithm, which allows zero impact on TCAM lookup, without locking

Thanks!!!

FYI: If interested, please visit:
http://crystal.uta.edu/~hche
for recent papers we wrote on issues related to TCAM coprocessor resource management. Please also stay tuned for more results on NPU/router design issues.