Hardware signaling paradigm for resource reservation

Israel Cidon
Department of Electrical Engineering
Technion

Short EE networking history

- Software packet switch – general purpose
  - Up to 1000s packets per second
- Routers built as parallel machines
  - Up to 10,000s packets per second
- Special hardware implementations
  - > 100,000,000 packets per second

Next generation services gaps

- Guaranteed, limited time p-t-p connection
- Large RT conferencing
- Video on demand - pay per view/listen
- Guaranteed large downloads
- RT Webcast
- **Need user differentiation of valued vs. low significance traffic, not flow specs**
- **Gap in QoS support – not in raw throughput**

Good old ways for QoS Support

- Reserve per connection for call duration
  - Requires massive signaling
  - Selection of available path
  - Statefull – soft-state increases CPU heat
  - More calls per second than ever imagined

Practice – limit call demand to protect computation resources using aggregation

Modern QoS Support

- Over-provisioning
  - Cost, lack of differentiation, garbage dominates
- Network wide QoS classes
  - Coarse, lack of scalability, no real-time
- Signaling aggregation - MPLS LSPs or ATM VPs
  - Quasi-static, aggregated flows only (VPN)
  - Save signaling & routing CPU cycles - adds complexity

Missing - user preference, instant billing…

Depends on low value BW fillers

Complex to understand and manage

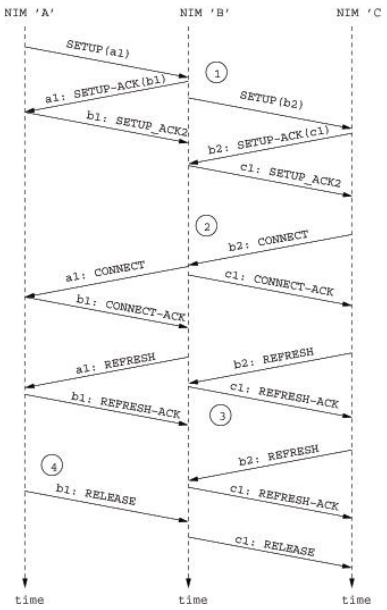
Possible brute force solution?

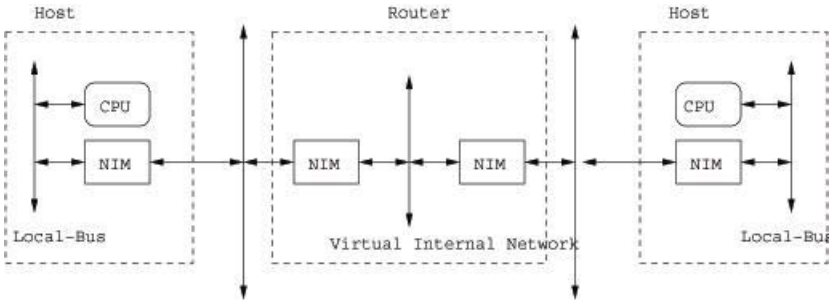
- Hardware implementation of old model
  - Hardware generated signaling
  - Hardware assisted QoS routing
- Implementation cost optimization
  - Save complexity and space not CPU cycles
- Do we need hardware optimized architecture?
  - Not essential in ATM but help understanding problems
- How we deal with routing?
  - Source routing – caching, pre-computation
  - QoS based destination routing
  - Invest in accuracy or apply multiple path reservation

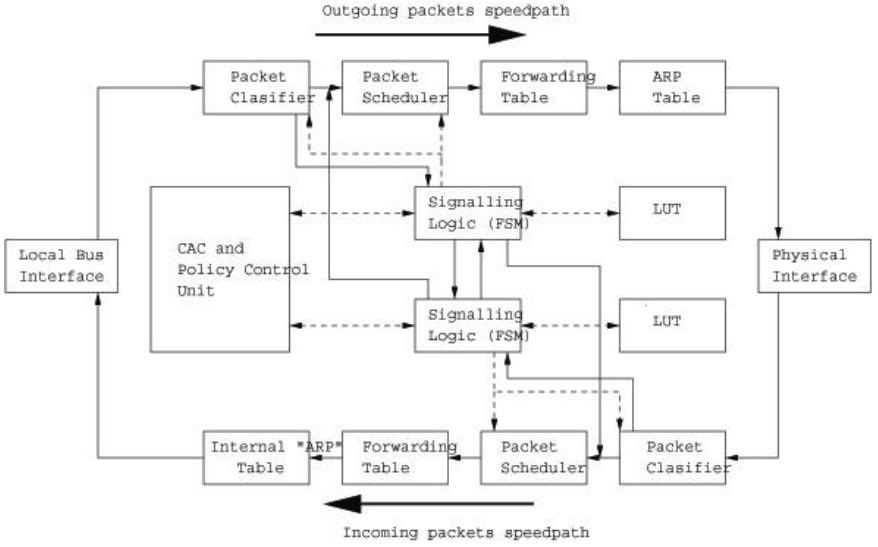
The KISS Architecture*

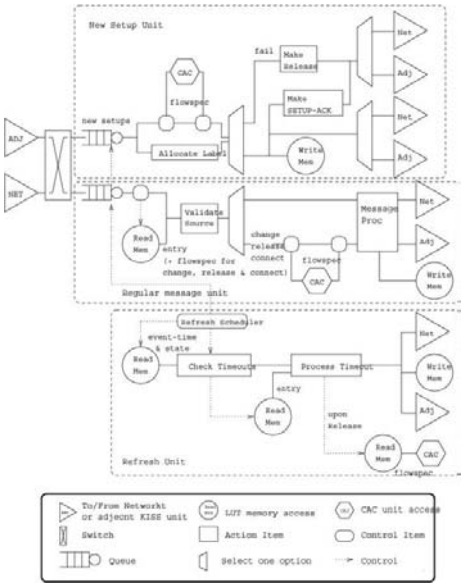
- Tailored around IP loose source routing
  - Support migration
  - Use route caching
  - Optimized for unicast
- Simple reservation, CAC and state structure
  - Random access
  - Soft-state protection

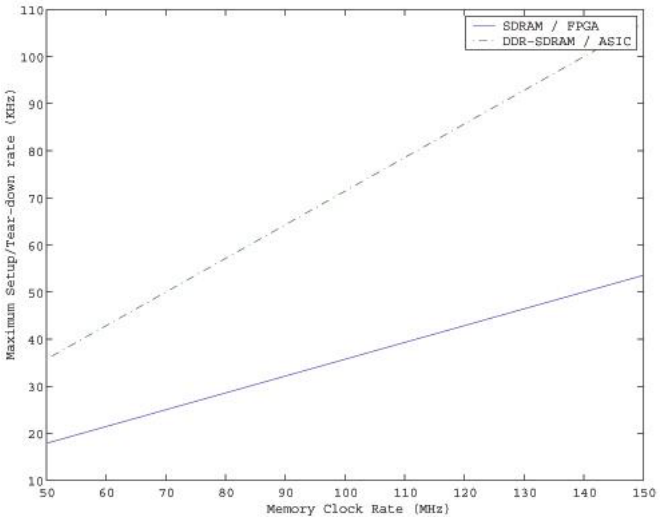
* *Joint work with Dan Gluskin*




Host
Router
Host
CPU
NIM
NIM
NIM
CPU
NIM
Local-Bus
Virtual Internal Network
Local-Bu


Outgoing packets speedpath
Packet Clasifier
Packet Scheduler
Forwarding Table
ARP Table
Signalling Logic (FSM)
LUT
Local Bus Interface
CAC and Policy Control Unit
Physical Interface
Signalling Logic (FSM)
LUT
Internal "ARP" Table
Forwarding Table
Packet Scheduler
Packet Clasifier
Incoming packets speedpath


New Setup Unit
fail
Make Release
Net
Adj
CAC
flowspec
Make SETUP-ACK
new setups
Allocate Label
Write Mem
ADJ
NET
Validate Source
Message Proc
Read Mem
entry
(+ flowspec for change, release & connect)
change/ release/ connect
flowspec
CAC
Regular message unit
Refresh Scheduler
event-time & state
Check Timeouts
Process Timeout
upon Release
Refresh Unit
To/From Network or adjecnt KISE unit
LUT memory access
CAC unit access
Switch
Action Item
Control Item
Queue
Select one option
Control


SDRAM / FPGA
DDR-SDRAM / ASIC
Maximum Setup/Tear-down rate (KHz)
Memory Clock Rate (MHz)
10
20
30
40
50
60
70
80
90
100
110
50
60
70
80
90
100
110
120
130
140
150

60 Ways to grab this BW

- Hardware based signals further exploited
  - Fast multi-way reservation and release
  - Soft-state prevents resource locking
  - Relaxed accuracy and timely BW tracking
- Hardware supported QoS routing protocol
  - Side by side not extending current routing protocols
  - Use hardware - extensive messaging link-state
  - Hardware to assist with distance vector
  - Hardware to assist with path computation

Back to basics

- Is QoS real obstacle or BW is damn cheap?
- Where are the HW/SW boundaries
  - Can anything be mapped to a NP
- Is full convergence essential?
  - Is IP the last network architecture?
  - Fast connection network can be separated
- Should standards define research?