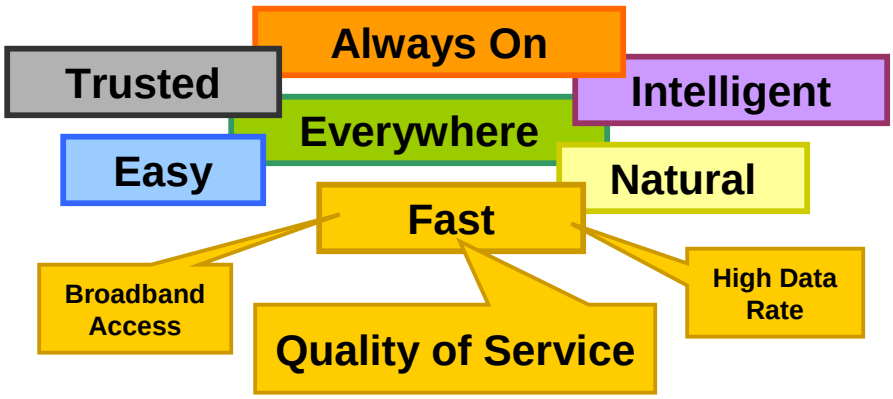
A Family of ASIC Devices for Next Generation Distributed Packet Switches with QoS support for IP and ATM

Fabio Chiussi, Alberto Brizio, Andrea Francini, Kevin Grant, Khurram Kazi, Denis Khotimsky, Santosh Krishnan, Sheng Shen, Mohammad Syed, Thomas Wasilewski

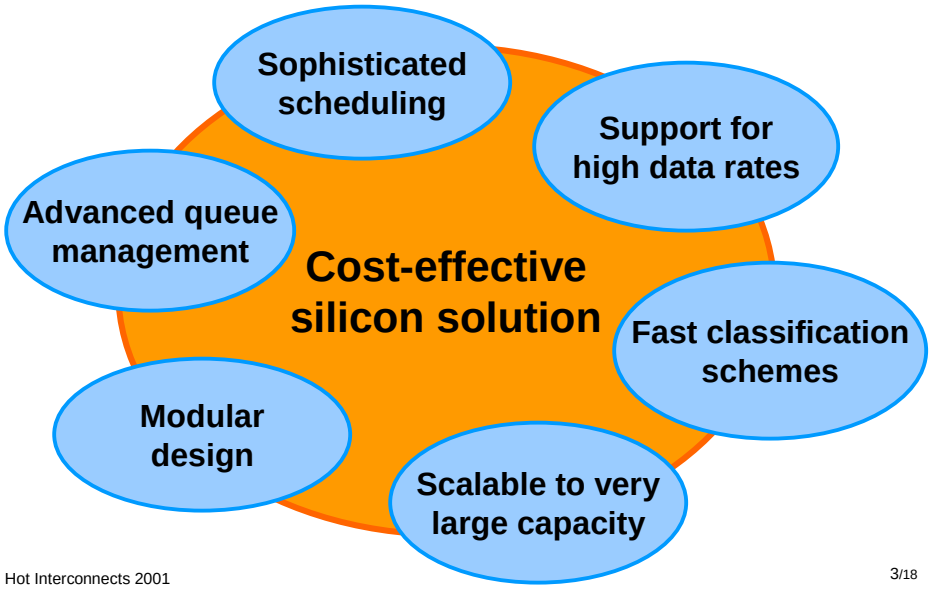
Bell Laboratories
Lucent Technologies

Next Generation Internet



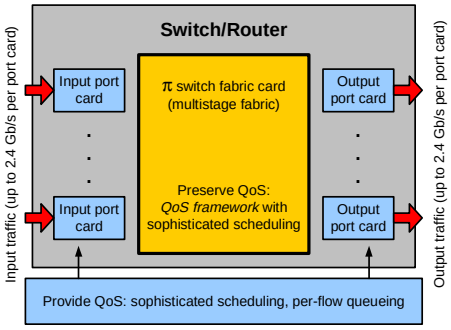
Switches and Routers must provide guarantees on bandwidth, delay, jitter

Challenges



The π family

- A complete solution offering QoS support for next generation switches and routers, built on top of a Distributed Multilayered Scheduling scheme
- DMS principles:
  - per-flow QoS in the port cards
  - per input-output pair QoS in the switch fabric (aggregation of flows)



Devices

Port Cards

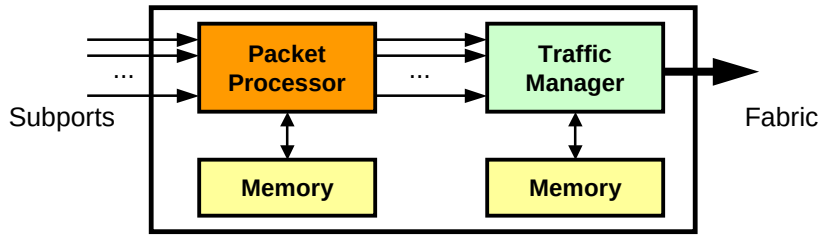
- π-IP & π-ATM
  - Protocol-specific classification engines
  - Full-duplex OC-48 line rate
- π-sched
  - Traffic manager offering per-flow queuing
  - 256,000 entries flow table
  - Half-duplex OC-48 line rate

Switch Fabric

- π-X
  - Queue management, buffering and scheduling
  - Full duplex with total raw throughput of 40 Gb/s
- π-C
  - Bufferless center stage crossbar
  - 8 or 16 input and output ports @ 4 Gb/s

Port cards

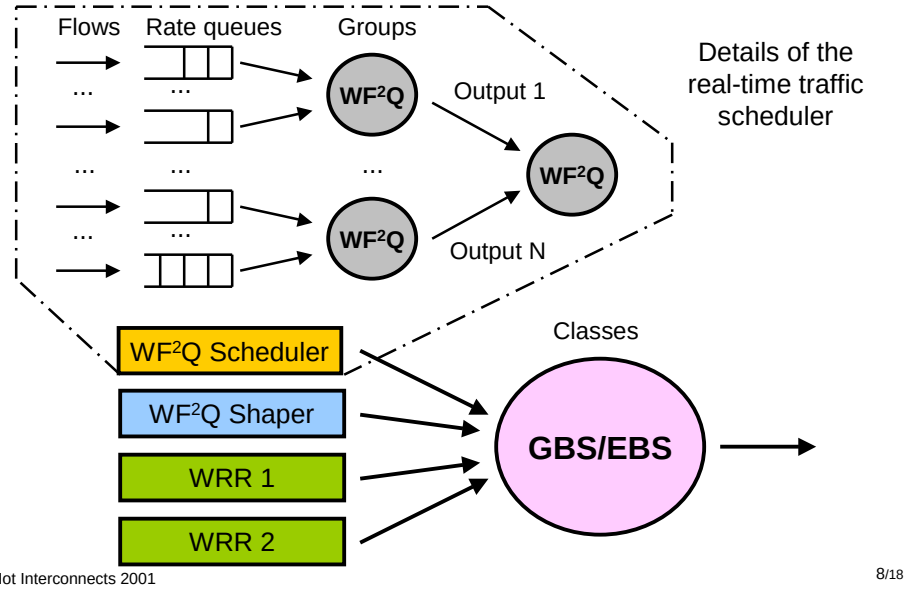
- Packet Classification
- Forwarding Decision
- Segmentation and Reassembling
- Flow-level Traffic Management (buffering, scheduling, and multiplexing/demultiplexing)



π-sched features

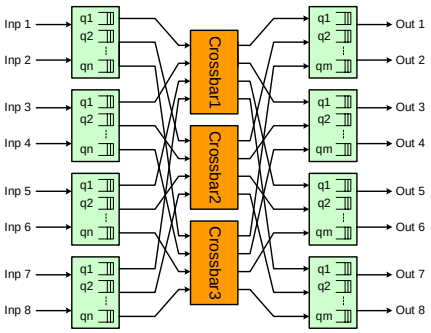
- ATM/IP traffic support
- OC-48 line rate
- Interworks with fabrics with up to 1024 input/output ports
- Support for 256 K flows (1 M in rev. 2)
- Only external memories are needed for packet buffers
- 5 M gates, 0.18 μm CMOS

π-sched scheduling structure



Multistage Switch Fabric

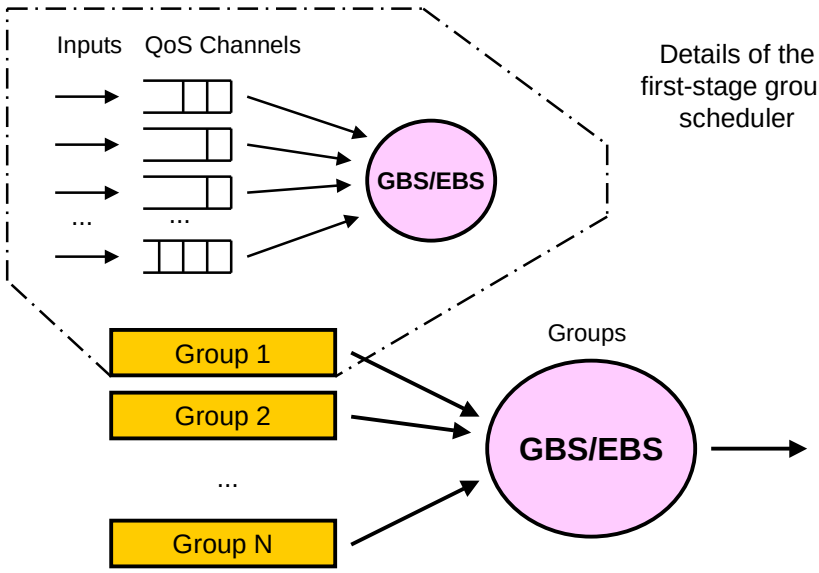
- Memory/Space/Memory
  Scalability, separation of functions
- QoS Framework for traffic isolation
  Differentiated support of traffic with stringent and non-stringent delay requirements (2 QoS channels per input/output pair)
- Scheduling
  Per-QoS channel Enhanced WF^2Q
- Concurrent Dispatching
  Multi-thread fully-distributed single-iteration crossbar arbitration scheme
- Selective backpressure
  Flow control on a per-QoS channel basis



π-x features

- ATM/IP traffic support
- Incorporates both first and third stage functionality, offering:
  - 8 ingress and 8 egress ports @ 1 Gb/s or
  - 2 ingress and 2 egress ports @ 4 Gb/s and
  - 3 input and 3 output ports @ 4 Gb/s in crossbar interface (space redundancy factor is 1.5)
- On-chip dynamically-shared buffer (SRAM) holds up to 6K packets
- 512 Mb/s LVDS connections to crossbars
- 5.5 M gates, 0.16 μm CMOS

π-x scheduling structure



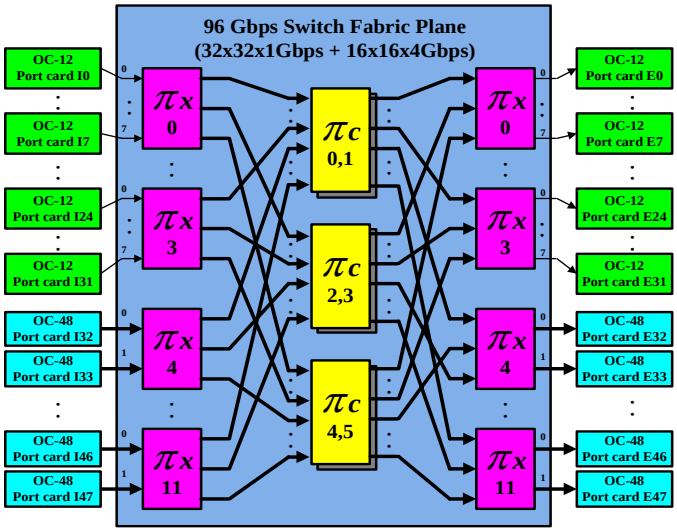
The π switch fabric: logical view

Up to 48 ports or 16 π-x modules

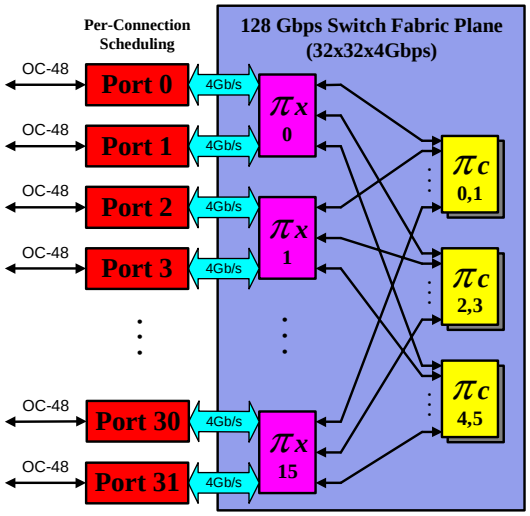
Up to 128x128 Gb/s throughput

Mix and match 1Gb/s & 4Gb/s π-x modules

Up to 1024 QoS channels per π-x module



The π switch fabric: physical view



Folded Architecture
First and third stage are built into the same device to save area, cost, power, and system complexity

Number of devices in the system

Configuration	π-x	π-c	Total
16x16 @ 4 Gbps (64 Gbps)	8	3	**11**
24x24 @ 4 Gbps (96 Gbps)	12	6	**18**
32x32 @ 4 Gbps (128 Gbps)	16	6	**22**

Design challenges...

- Algorithm related
  - Sophisticated scheduling requires large amount of state information and complex logic
  - Support for extra control paths (e.g.: backpressure)
- Speed related
  - Timestamp ordering schemes (e.g.: 13 clock cycles to find 3 minima out of 224 timestamps in π-x)
  - Aggregate traffic rate per π-x is 40 Gb/s
- Size limiting factors
  - Large number of queues to be managed
  - Transparent multicast traffic support
  - System synchronization, skew tolerance

…and solutions

- Algorithmic level
  - Simplification while maintaining strict QoS guarantees
  - Distributed crossbar arbitration
- Architectural level
  - Massive resource parallelism
  - Fast search schemes
  - Control communications piggybacked to data transfers
- Physical design level
  - Custom-designed high-speed interfaces with clock recovery for chip to chip data transfers
  - Manual placement and routing of stackable cells for critical data paths

Functional verification

- Brute-force approach?
  Fails because of the device complexity
- Formal verification methodologies?
  Still too immature for large devices
- Is it possible to bridge the two worlds?
  Expressing the original design specifications into simple rules that can be automatically enforced while simulating the behavioral code or the RTL code

\+ Limited or no manual intervention in analyzing test results

\- Still necessary to write a suite of tests

Pixan

- A tool for automating the verification effort, written in TCL
  - Uses ***event traces*** generated during the simulation to check the behavior of the device
  - Simple to write rules and extend the knowledge base
  - Can be applied on the output generated by various models of the UUT, not just RTL
  - Verifies ***every*** rule in ***each*** test, not just the ones the test is targeted at
  - Creates signatures for each test to speed up the regression phase

Conclusions

- QoS support in broadband Internet drastically increases VLSI and system complexity
- Size and complexity of the devices make it hard to thoroughly verify their functionality with both traditional and formal methods
- The π family
  - a scalable solution for next generation QoS-enabled switches and routers
  - synthesizes a number of innovations in algorithm and architecture design, physical design, and verification procedures