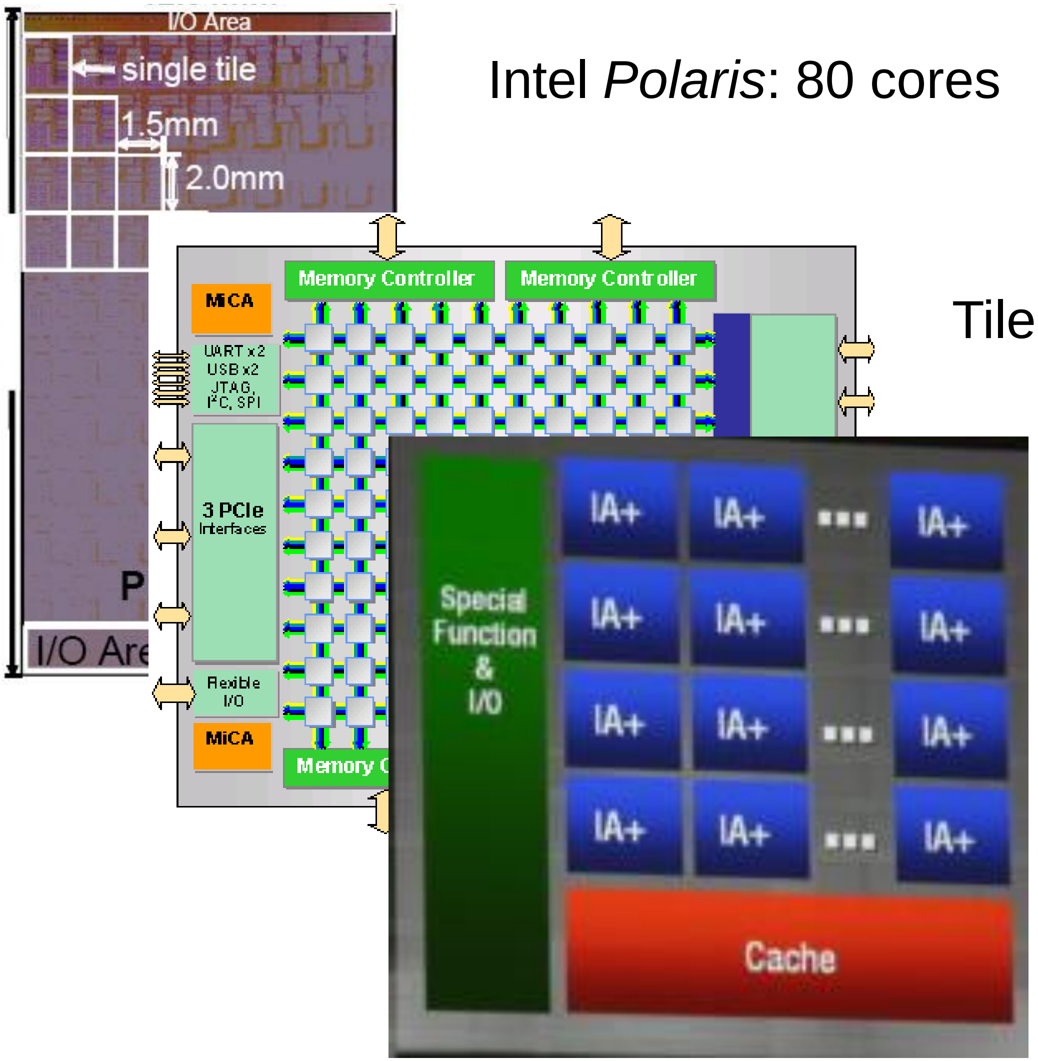
Preemptive Virtual Clock: A Flexible, Efficient and Cost-effective QOS Scheme for Networks-on-Chip

B. Grot, S. W. Keckler *(UT-Austin)*

O. Mutlu *(CMU)*

The Many-core Generation



Intel *Polaris*: 80 cores

Tilera *Tile-Gx*: 16-100 cores

Intel *Larrabee*: 16-64 cores

Why on-chip QOS?

- Shared on-chip resources require QOS support for fairness, service differentiation, performance, etc.
  - Memory controllers
  - Cache banks
  - Specialized accelerators
  - *On-chip network*
- End-point QOS solutions are insufficient
  - Data has to traverse the on-chip network, a shared resource
  - Need QOS support at the interconnect level

NOC QOS Desiderata

Feature	*PVC*
Fairness	✓
Isolation of flows	✓
Efficient BW utilization	✓
Low overhead: delay, area, energy	✓
Flexible BW allocation	✓

Outline

- Prior Art
  - Conventional network QOS schemes
  - On-chip network QOS
- Preemptive Virtual Clock
  - Bandwidth allocation
  - QOS particulars
  - Microarchitectural details
- Evaluation methodology
- Experimental results
- Summary

Conventional QOS Disciplines

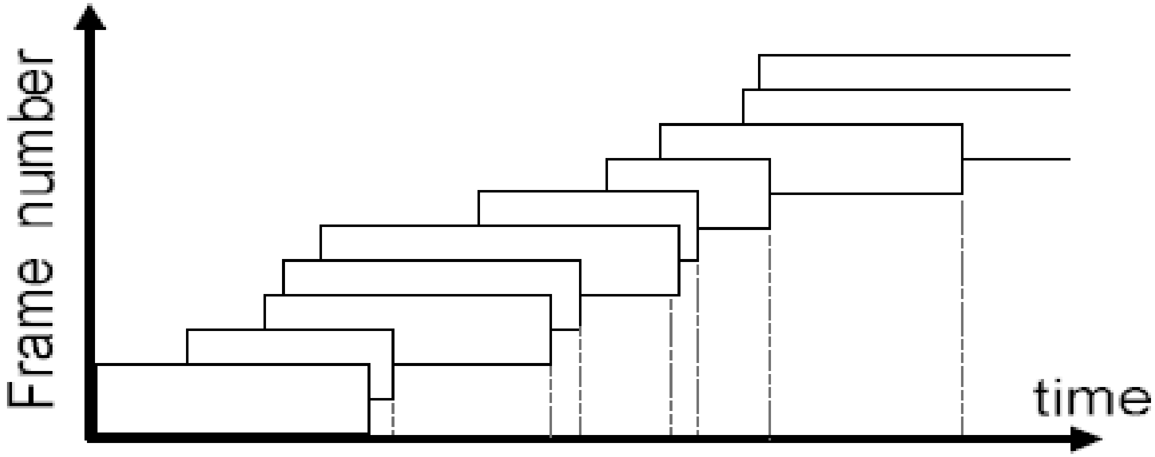
- Fixed schedule
  - Pros: algorithmic and implementation simplicity
  - Cons: inefficient BW utilization; per-flow queuing
  - Example: Round Robin
- Rate-based
  - Pros: fine-grained schedule control; efficient
  - Cons: complex scheduling; per-flow queuing
  - Example: Weighted Fair Queuing (WFQ) [SIGCOMM ‘89]
- Frame-based
  - Pros: good throughput at modest complexity
  - Cons: throughput-complexity trade-off; per-flow queuing
  - Example: Rotating Combined Queuing (RCQ) [ISCA ’96]

Conventional QOS Disciplines

Per-flow queuing

- Area overhead
- Energy overhead
- Delay overhead
- Scheduling complexity

On-chip QOS: *Globally Synchronized Frames*[+]

- Key contribution: move much of the buffer overhead and scheduling complexity into the source nodes
- Overview
  - Frame-based approach
  - Fixed number of injection slots per source in each frame
  - Multiple frames in flight
  - Barrier network detects frame completion
- 

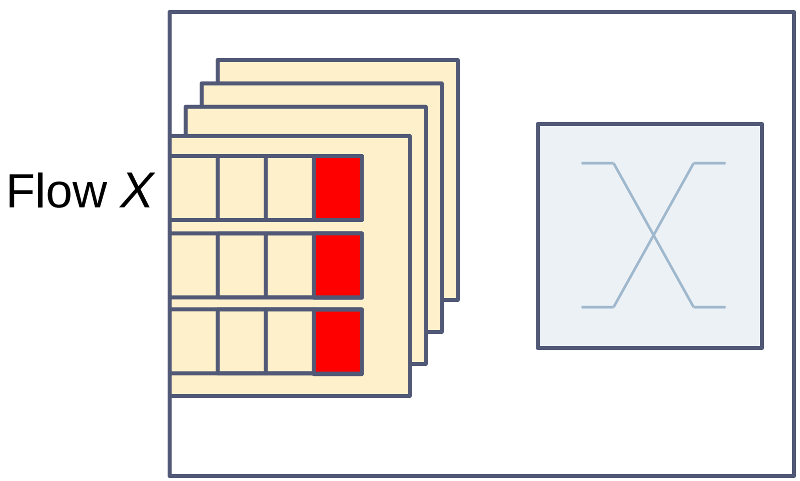
[+] J. Lee, et al. ISCA 2008

Preemptive Virtual Clock (PVC)

- Goal: high-performance, cost-effective mechanism for fairness and service differentiation in NOCs.
- Full QOS support
  - Fairness, prioritization, performance isolation
- Modest area and energy overhead
  - Minimal buffering in routers & source nodes
- High Performance
  - Low latency, good BW efficiency
- Flexible network provisioning
  - Per-application or per-VM bandwidth allocation independent of the core/thread count

PVC: Scheduling

- Combines rate-based and frame-based features
- Rate-based: evolved from *Virtual Clock* [SIGCOMM '90]
  - Routers track each flow's bandwidth consumption
  - Cheap priority computation
    - *f (provisioned rate, consumed BW)*
    - Problem: history effect

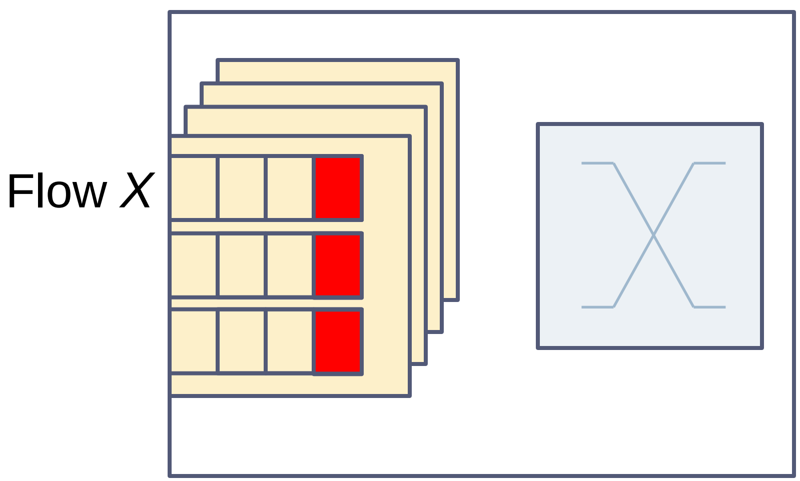


PVC: Scheduling

- Combines rate-based and frame-based features
- Rate-based: evolved from *Virtual Clock* [SIGCOMM '90]
  - Routers track each flow's bandwidth consumption
  - Cheap priority computation
    - f *(provisioned rate, consumed BW)*
    - Problem: history effect
- Framing: PVC's solution to history effect
  - Frame rollover clears all BW counters
  - Fixed frame length
  - Packets not bound to any particular frame

PVC: Scheduling

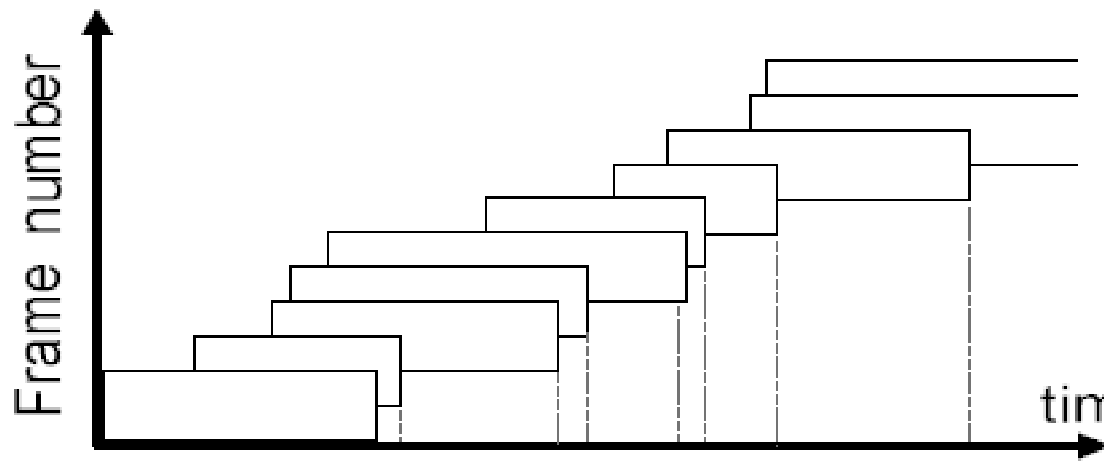
- Combines rate-based and frame-based features
- Rate-based: evolved from *Virtual Clock* [SIGCOMM '90]
  - Routers track each flow's bandwidth consumption
  - Cheap priority computation
    - *f (provisioned rate, consumed BW)*
    - Problem: history effect



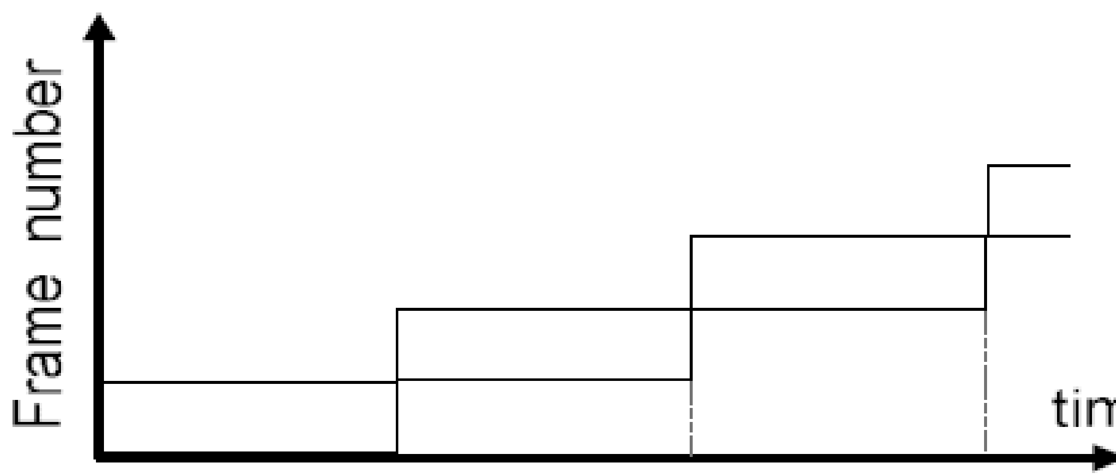
Frame roller
- BW counters reset
- Priorities reset

Framing: GSF vs PVC

GSF

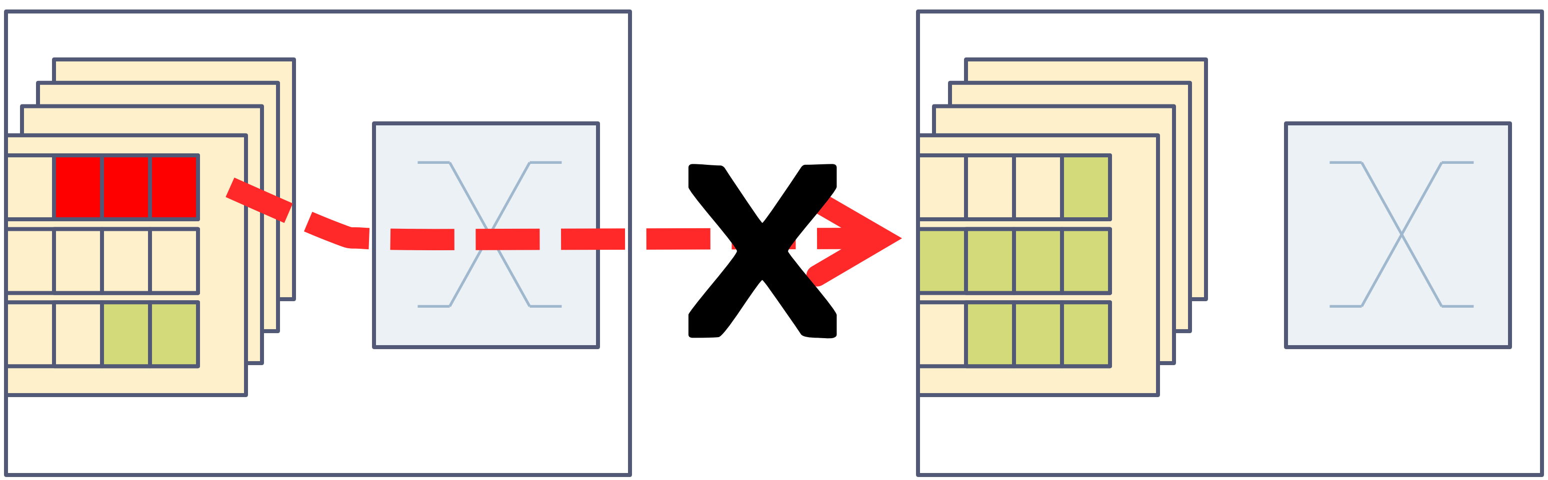


PVC



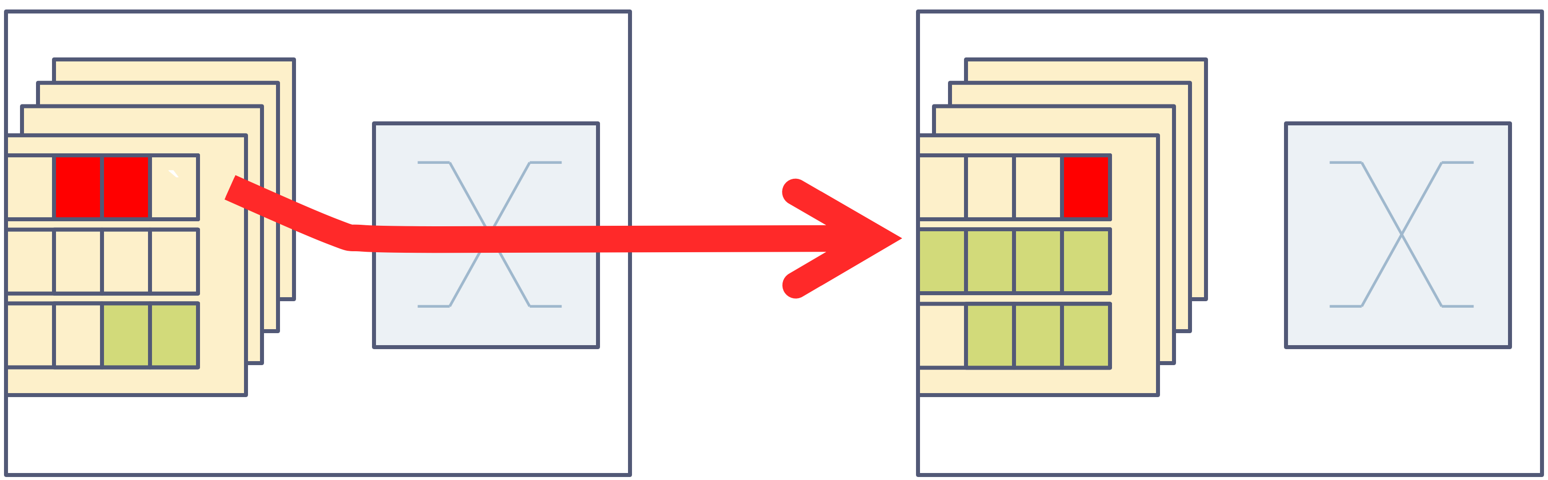
PVC: Freedom from Priority Inversion

- PVC: simple routers w/o per-flow buffering and no BW reservation
  - Problem: high priority packets may be blocked by lower priority packets (*priority inversion*)



PVC: Freedom from Priority Inversion

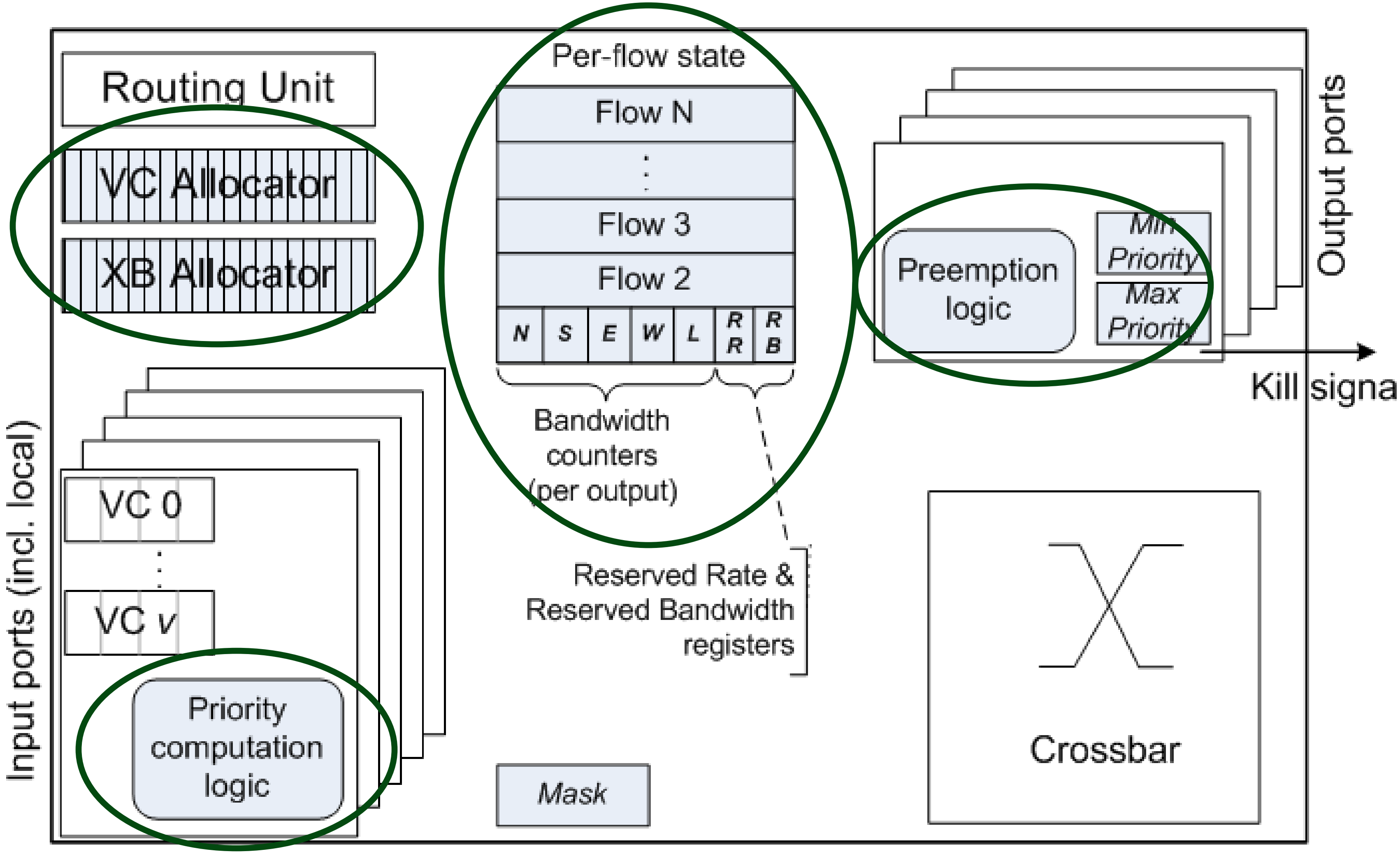
- PVC: simple routers w/o per-flow buffering and no BW reservation
  - Problem: high priority packets may be blocked by lower priority packets (*priority inversion*)
- Solution: *preemption* of lower priority packets



PVC: Preemption Recovery

- Retransmission of dropped packets
- Buffer outstanding packets at the source node
- ACK/NACK protocol via a dedicated network
  - All packets acknowledged
  - Narrow, low-complexity network
  - Lower overhead than timeout-based recovery
  - 64 node network: 30-flit transaction buffer sufficient

PVC: Router Modifications

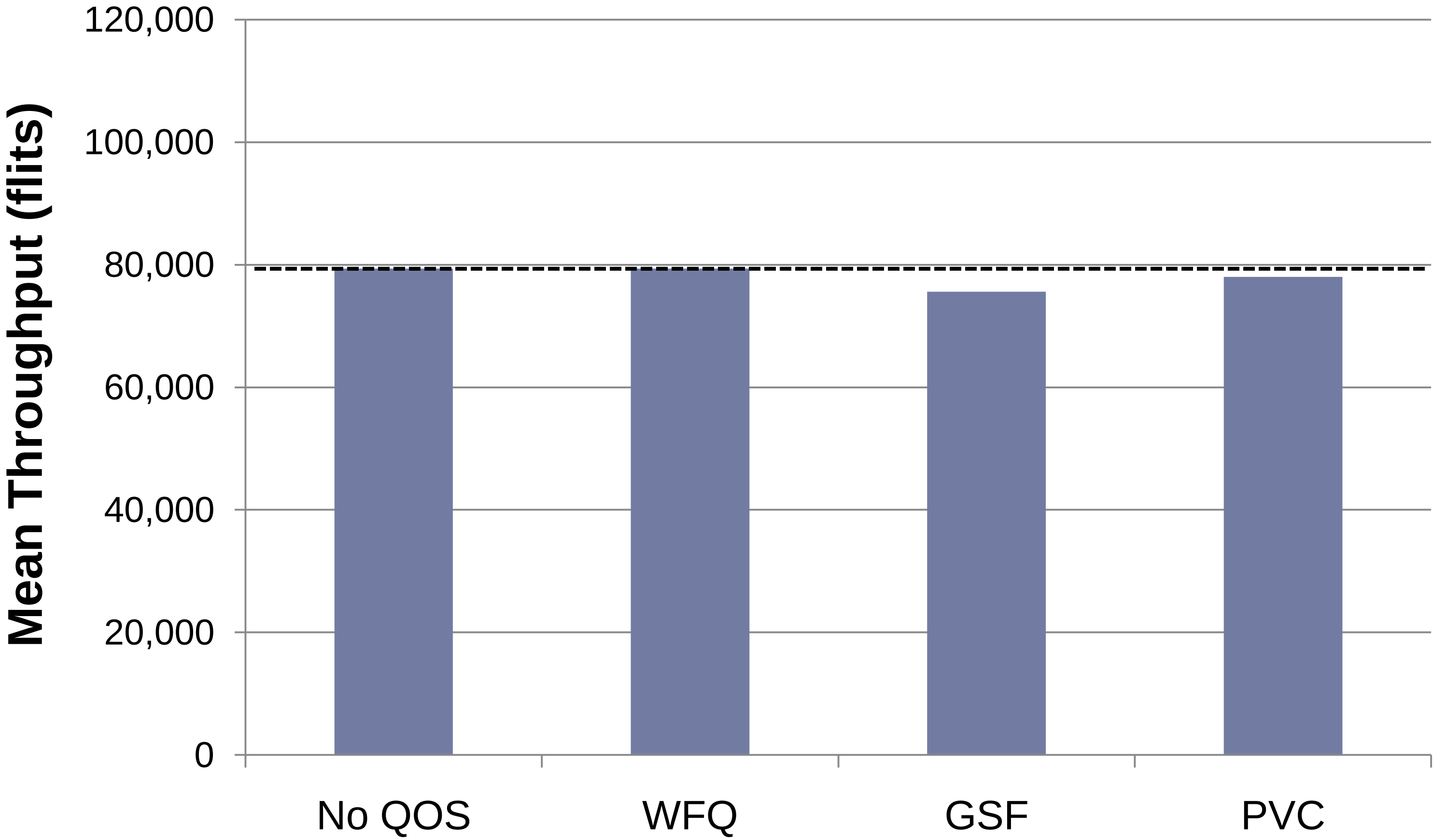


Evaluation Methodology

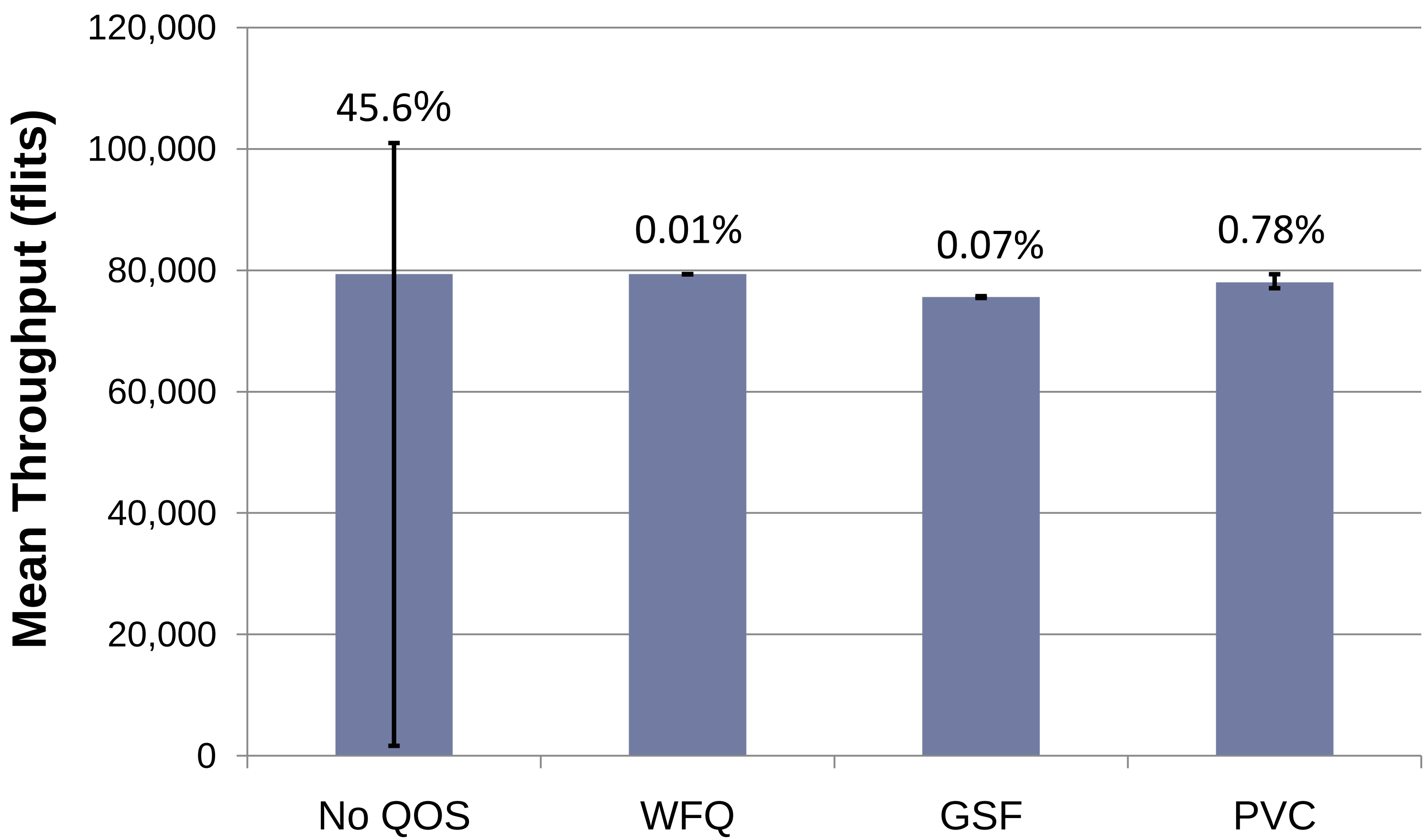
Network	64 nodes+, 16 byte link width, XY DOR routing
Synthetic experiments	*hotspot* and *uniform random*; 1- and 4-flit packets
PARSEC experiments	*blackscholes, fluidanimate, vips*; sim-medium data set
Baseline network (no QOS)	6 VCs per network port, 5 flits/VC, 1 injection VC, 2 ejection VCs; 3-cycle router pipeline
WFQ network	Per-flow queuing at each router node: 64 queues, 5 flits/queue
GSF network	2K slots/frame, 6 frames in-flight, 8 cycle frame reclamation delay; Router config: same as baseline, but 1 VC reserved
PVC network	50K cycles/frame, 30 flit source transaction buffer; Router config: same as baseline, but 1 VC reserved

+ Select results for 256 nodes in the paper

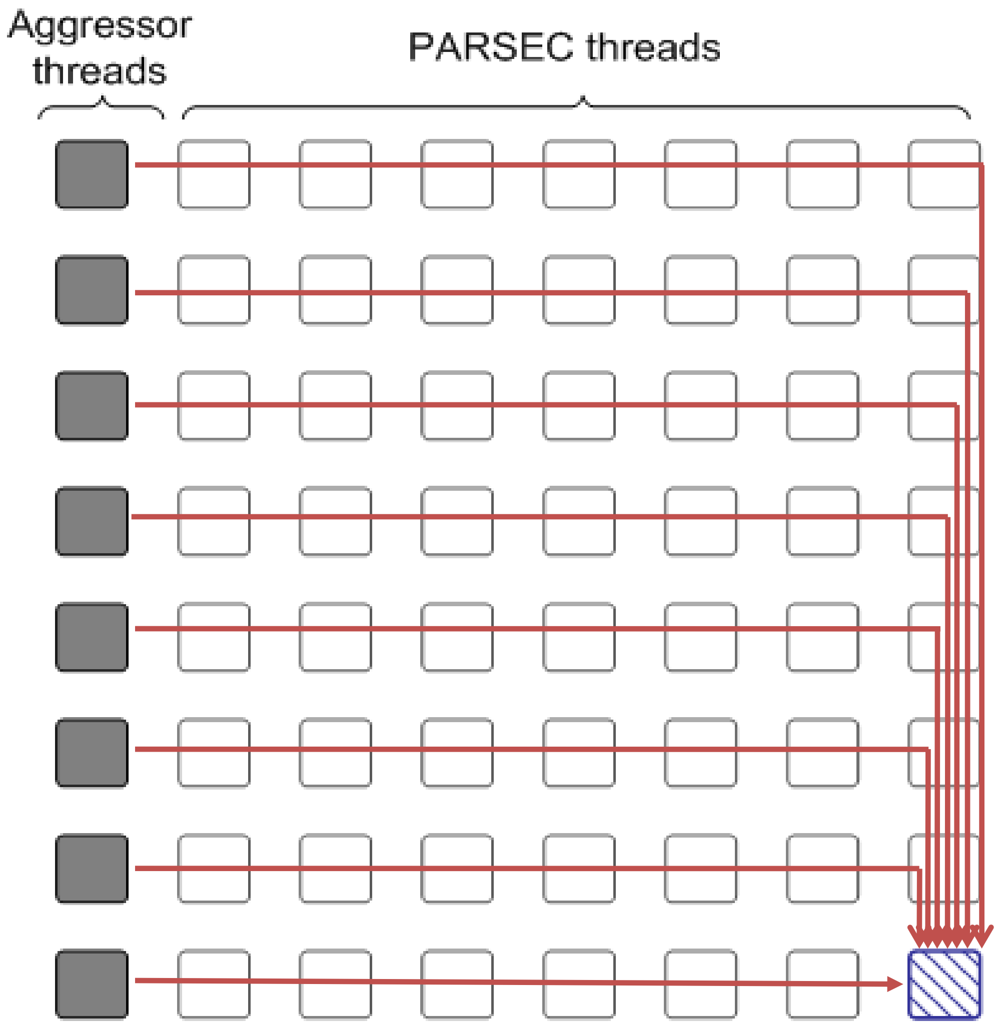
Throughput & Fairness (*hotspot* traffic)



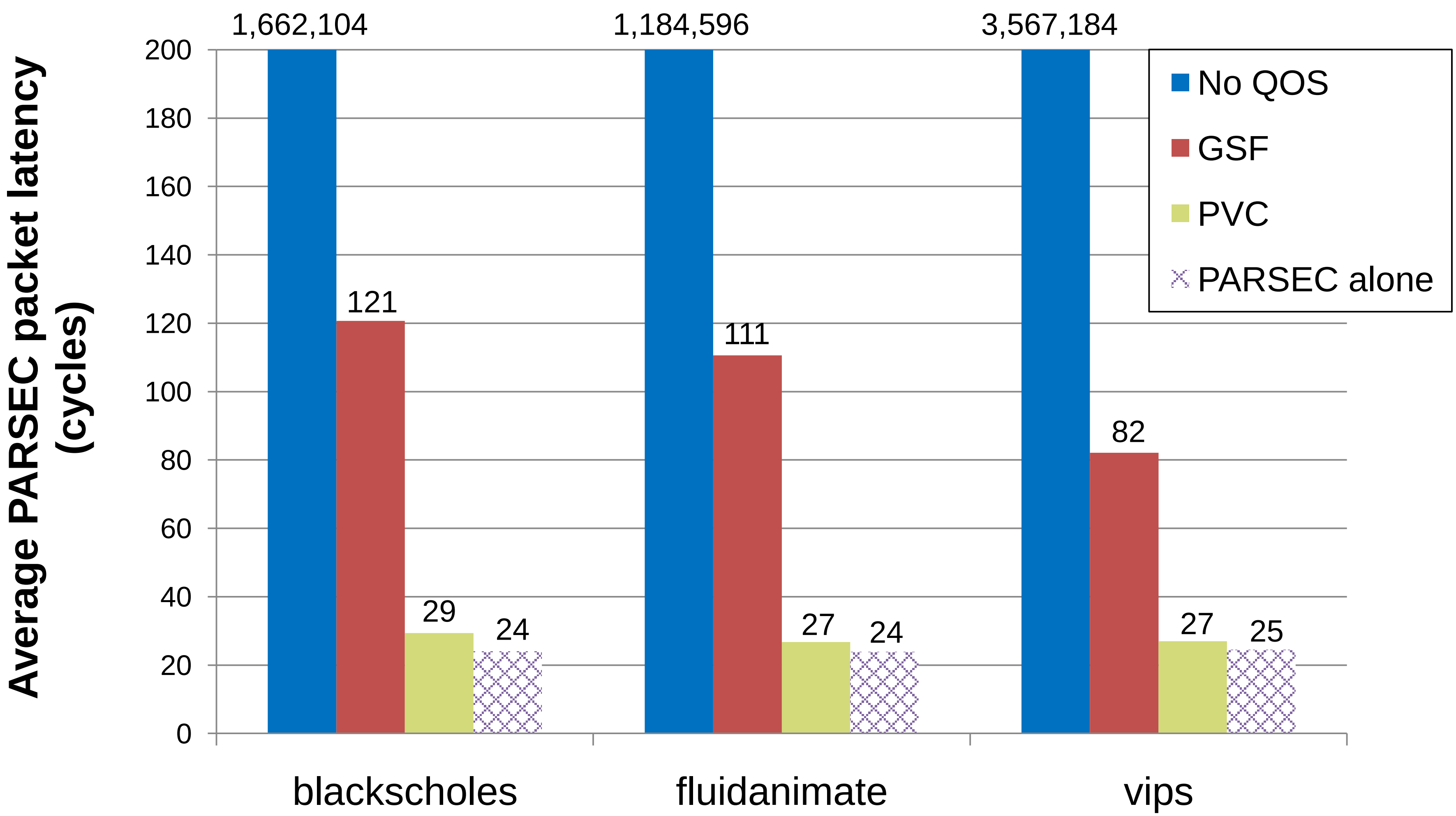
Throughput & Fairness (*hotspot* traffic)



Performance Isolation



Performance Isolation



PVC Summary

- Full QOS support
  - Fairness & service differentiation
  - Strong performance isolation
- High performance
  - Complexity-effective routers → low latency
  - Good bandwidth efficiency (12% thruput loss on *Unif. Random*)
- Modest area and energy overhead
  - 3.4 KB of storage per node (1.8x baseline)
  - Up to 18% energy overhead over baseline (*Uniform Random*)
- Flexible network provisioning
  - Aggregate multiple threads into a single flow

PVC Summary

- Full QOS support
  - Fairness & service differentiation

Low-cost
high-performance
QOS for NOCs

PVC

ndom)

ndom)

- Flexible network provisioning
  - Aggregate multiple threads into a single flow