A QoS-enabled Packet Scheduling Algorithm for IPSec Multi-Accelerator Based Systems

Alberto Ferrante and Vincenzo Piuri

DTI, University of Milan

{ferrante, piuri}@dti.unimi.it

Fabien Castanier

AST, ST Microelectronics

fabien.castanier@st.com

Presentation Outline

1. IPSec;
2. The scheduling algorithm;
3. Model for simulations and results;
4. Architectural enhancements;
5. Conclusions and Future Work.

IPSec

The scheduling algorithm

Model For Simulations and Results

Architectural Enhancements

Conclusions and Future Work

IPSec

- Is a suite of protocols
  - adding security at IP (network) level;
- makes extensive use of cryptographic functions:
  - it is resource consuming.

IPSec importance

- it is included
  as security mechanism in IPv6;
- it is widely used in present VPNs.

Goals

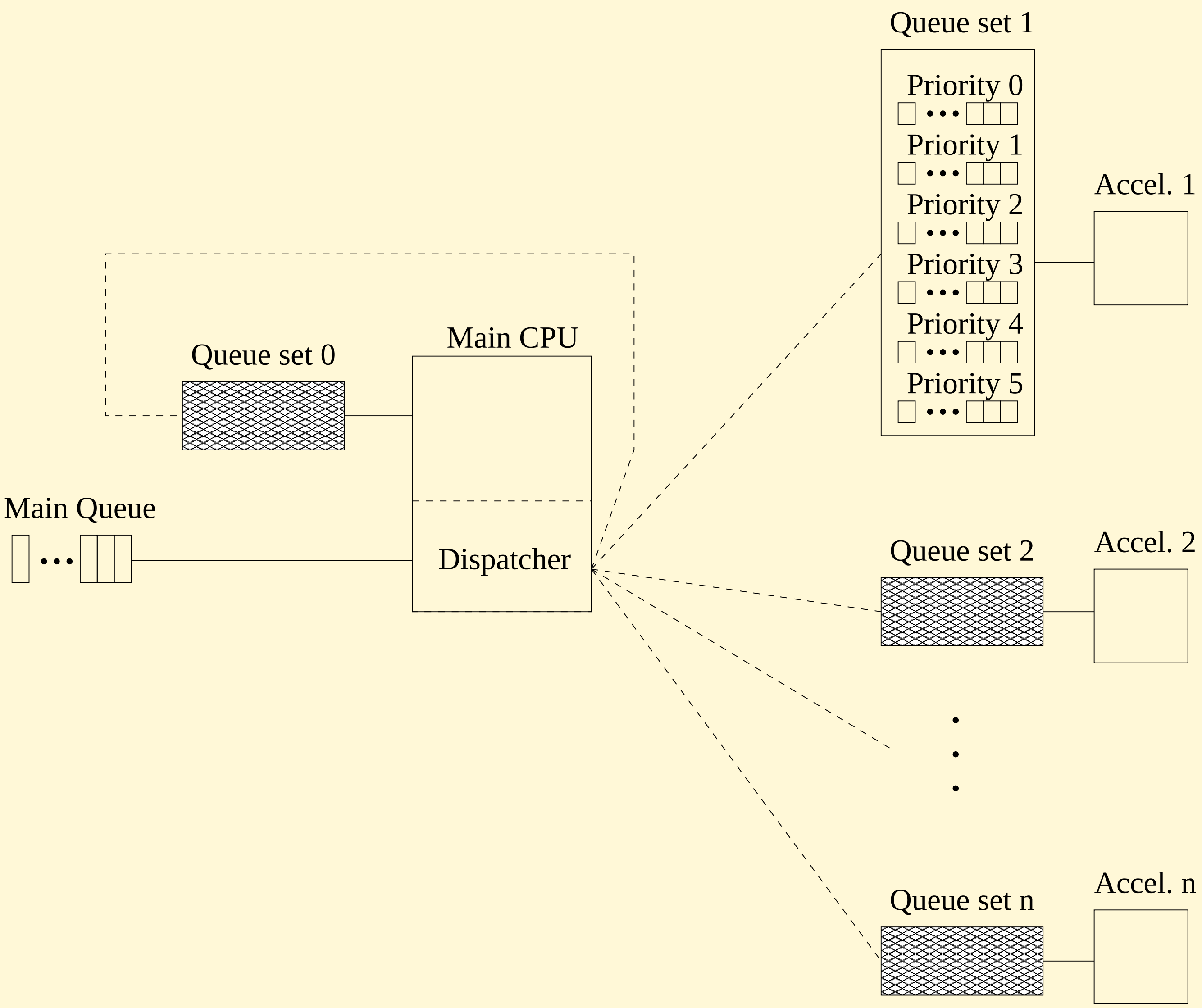
- To obtain a scheduling algorithm being able to:
  - schedule packet processing between N crytpo-accelerators;
  - schedule packets also to a software implementation of the cryptographic algorithms;
- support QoS;
- minimize latency obtaining high throughput.

Assumptions

The scheduling algorithm relies heavily on two facts:

- processing time of each packet is known in advance;
- each packet can be processed independently from the others.

Scheduling Algorithm (1)



Scheduling Algorithm (2)

- Each received packet is processed by the scheduler that:
  - selects a set of suitable processors;
  - computes the finishing time for each of the processors;
  - allocates the packet to the processor with lowest finishing time.

Packet Processing

- Packets in priority queues are processed accordingly to a modified version of the Weighted Fair Queuing (WFQ) policy:
  - each packet need to be considered as an atomic unit.

$$F_p = \frac{p+1}{\sum_{l=1}^{P} l}$$

Finishing Time

- $finishing\_time = waiting\_time + processing\_time$;
- due to priorities the finishing time can only be estimated;
- two parameters are added to allow tuning CPU load:
  - α_0 is a multiplicative constant;
  - β_0 is an additive constant.

Predictions On Packets

IPSec

The scheduling algorithm

Model For Simulations and Results

Architectural Enhancements

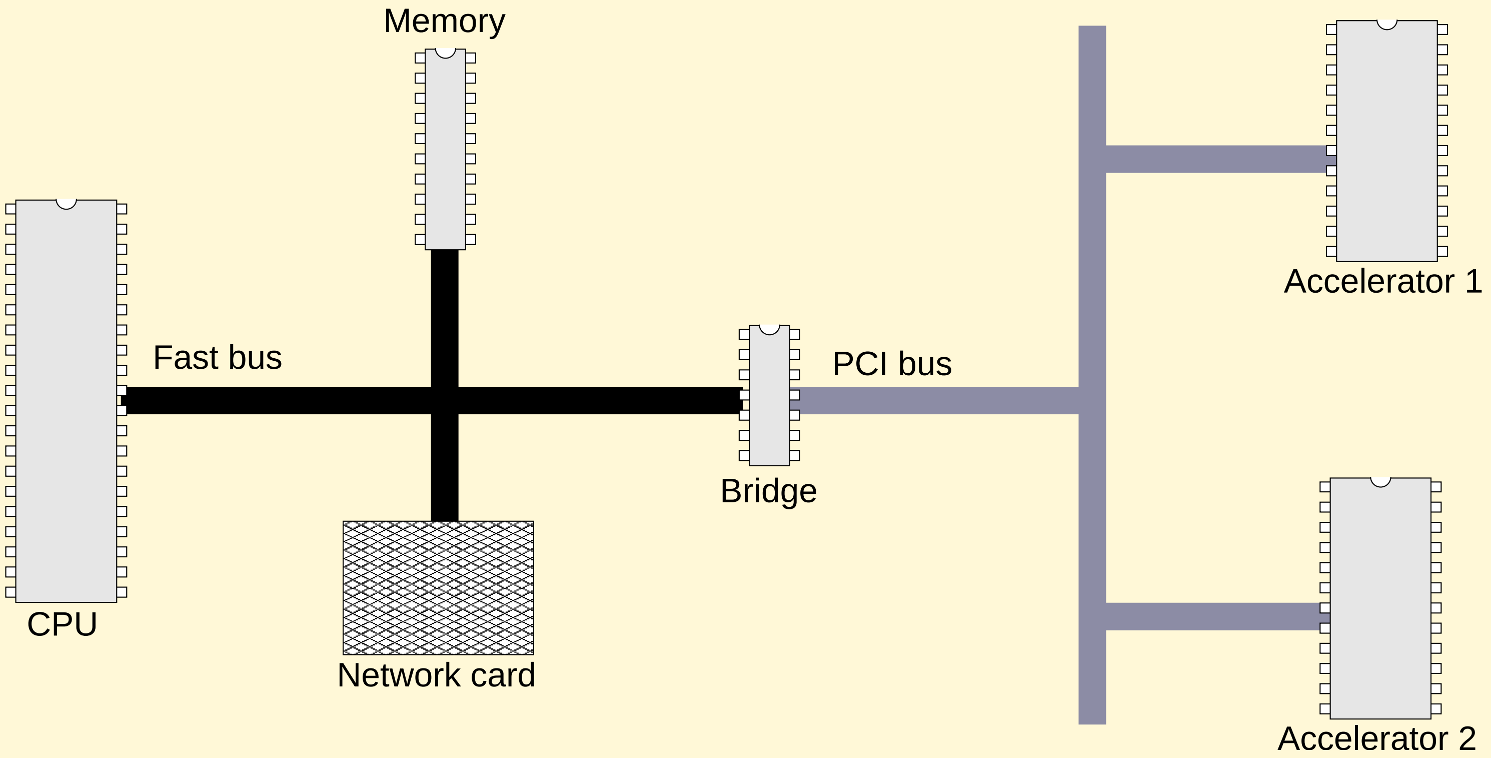
Conclusions and Future Work

- A k-step moving average:
  - is used to evaluate:
    - the number of packets that are in each queue;
    - their average processing time;
  - values can be computed:
    - each time one of the queues is modified *(packet average)*;
    - each l round robin cycles *(round robin average)*.

Scheduler in Practice

- Waiting time for a processor is computed:
  - each time the corresponding queue set is modified;
  - or each l round robin cycles;
- finishing time for a processor is computed each time a packet needs to be scheduled;
- In each scheduling operation, at most $N + 1$ comparisons are needed.

Reference Architecture



Data transfers to and from the accelerators are performed in DMA mode.

Model For Simulations (1/2)

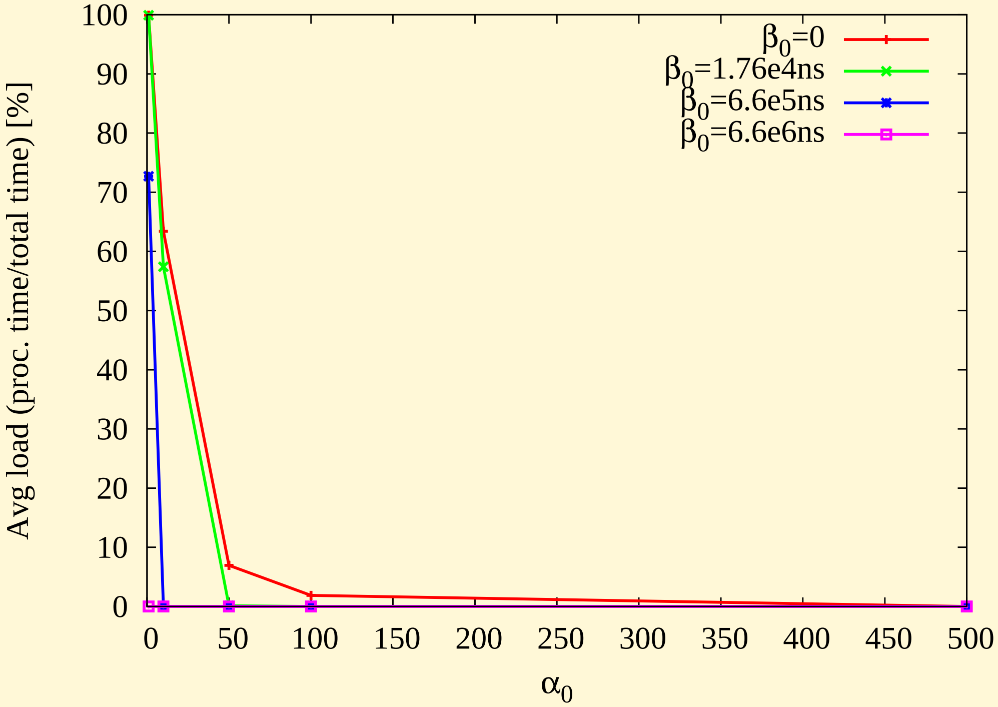
- It models the main parts of the system;
- in accelerators AES encryption is only considered;
- the only form of synchronization considered is bus contention:
  - accesses to memory are faster;
  - model not done to really measure performance;

Model For Simulations (2/2)

- It has been implemented in functional SystemC;
- Simulation inputs are taken from files provided on ITA website:
  - 1mln of packets were considered in each simulation.

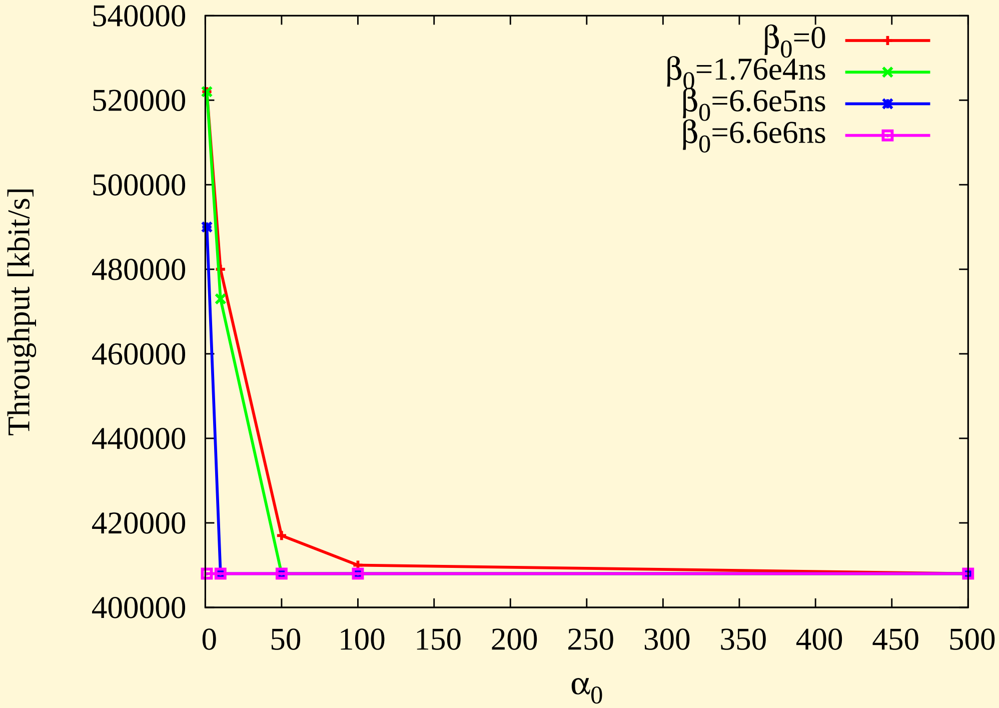
Results (1/3)

CPU Load



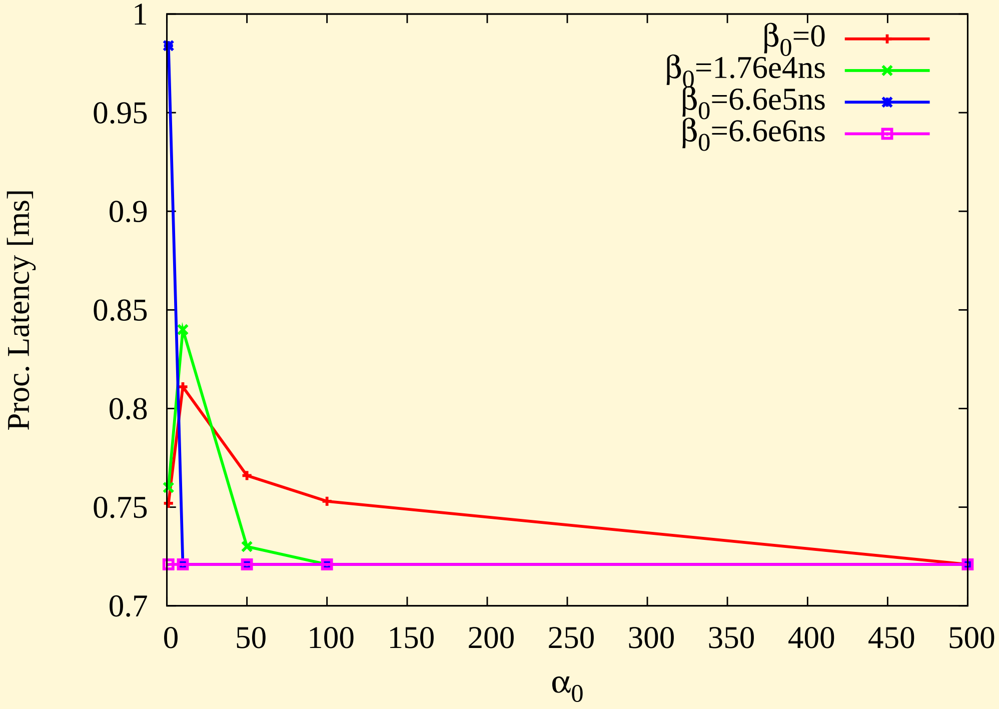
- Required bandwidth: 1Gbit/s;
- packet average case; window size: 1;
- number of accelerators: 2.

Throughput



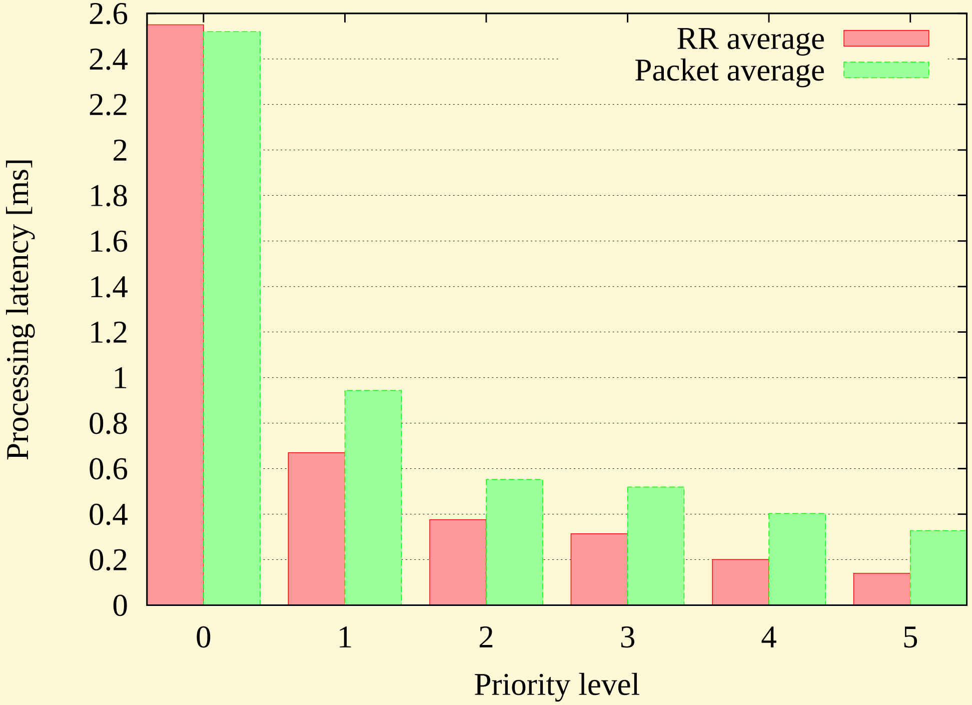
Results (2/3)

Average processing latency

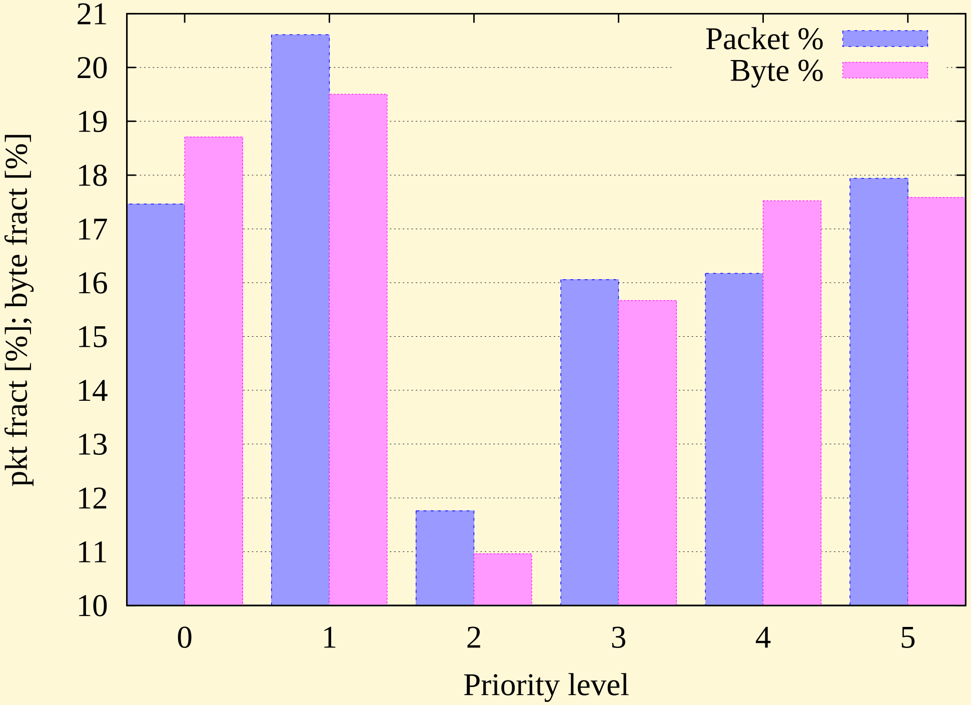


Results (3/3)

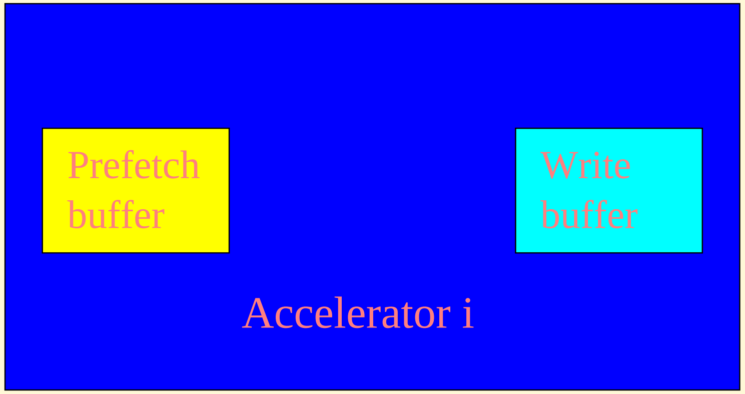
Average processing latency



Packet distribution among priority levels



Architectural Enhancements



IPSec

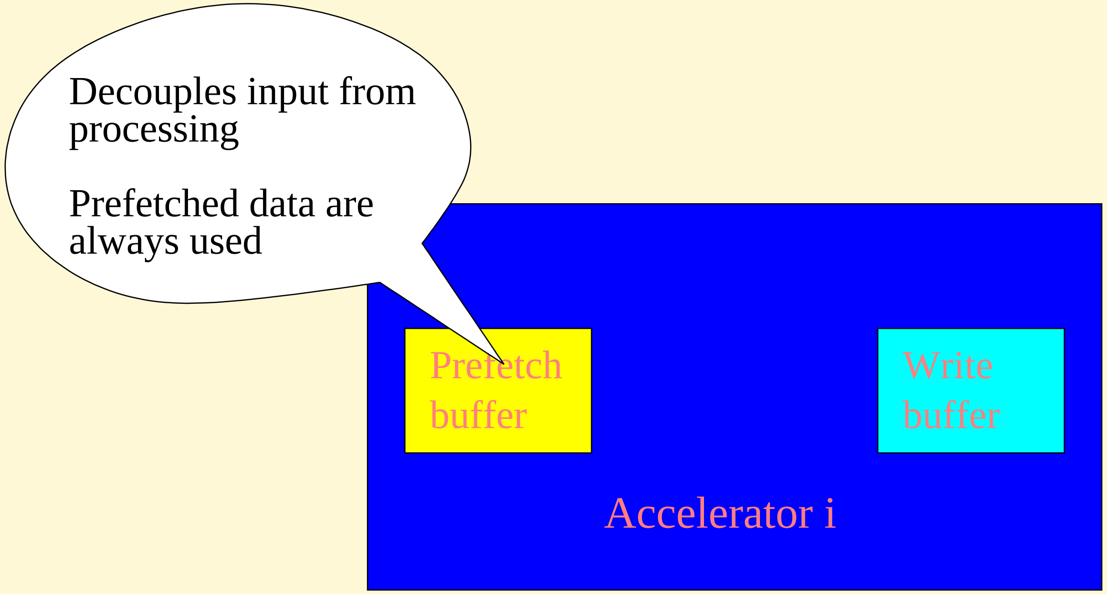
The scheduling algorithm

Model For Simulations and Results

Architectural Enhancements

Conclusions and Future Work

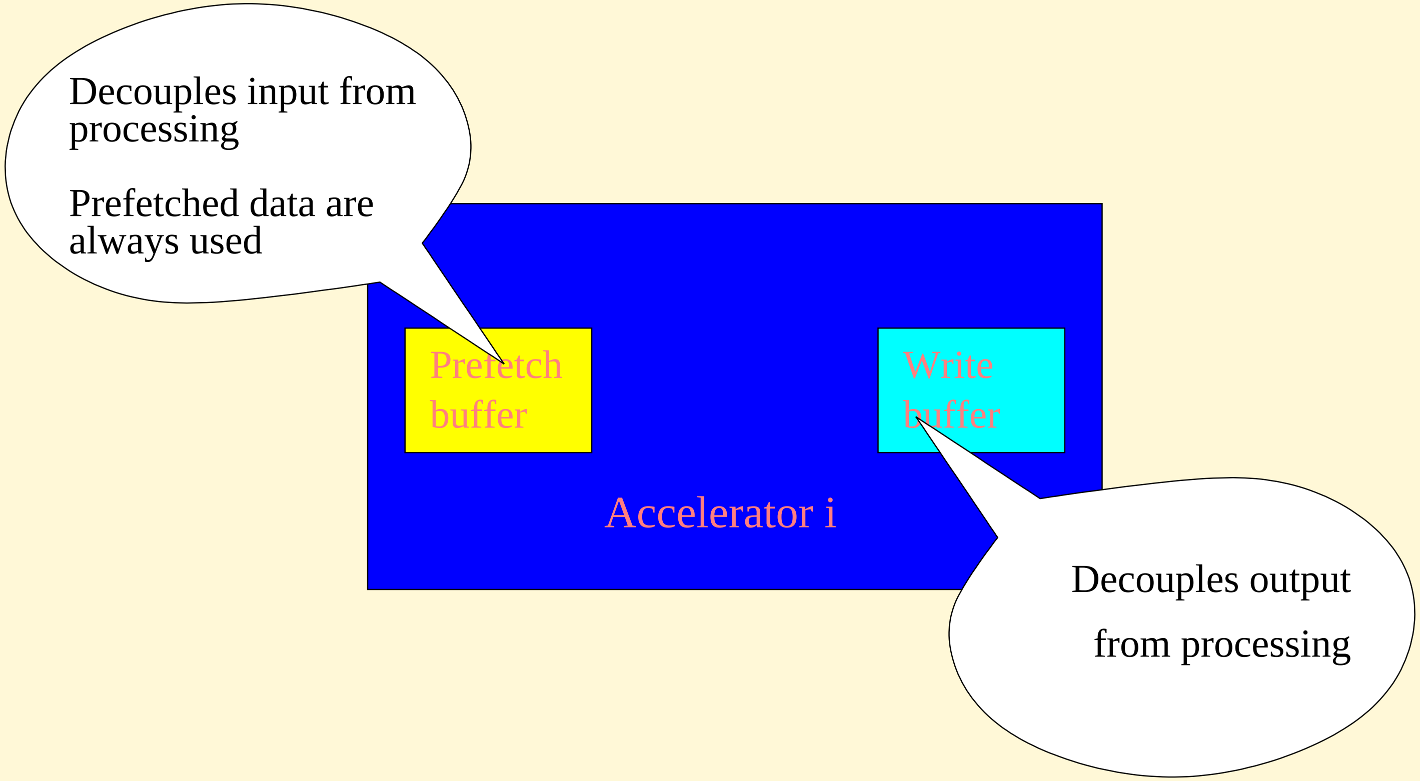
Architectural Enhancements



Decouples input from processing

Prefetched data are always used

Architectural Enhancements



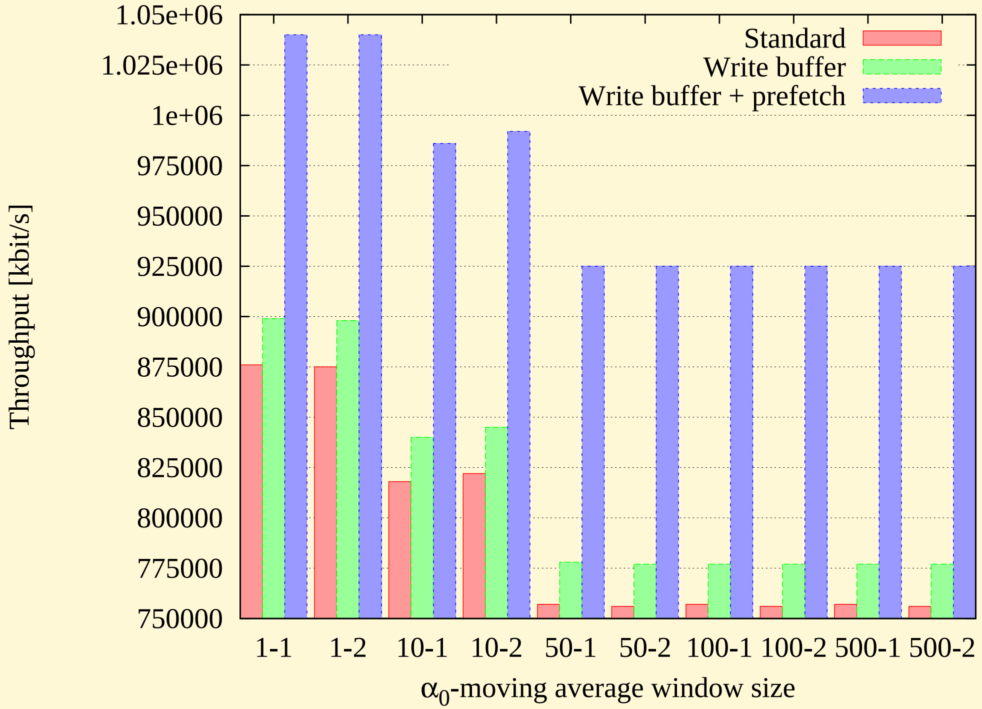
Decouples input from processing

Prefetched data are always used

Decouples output from processing

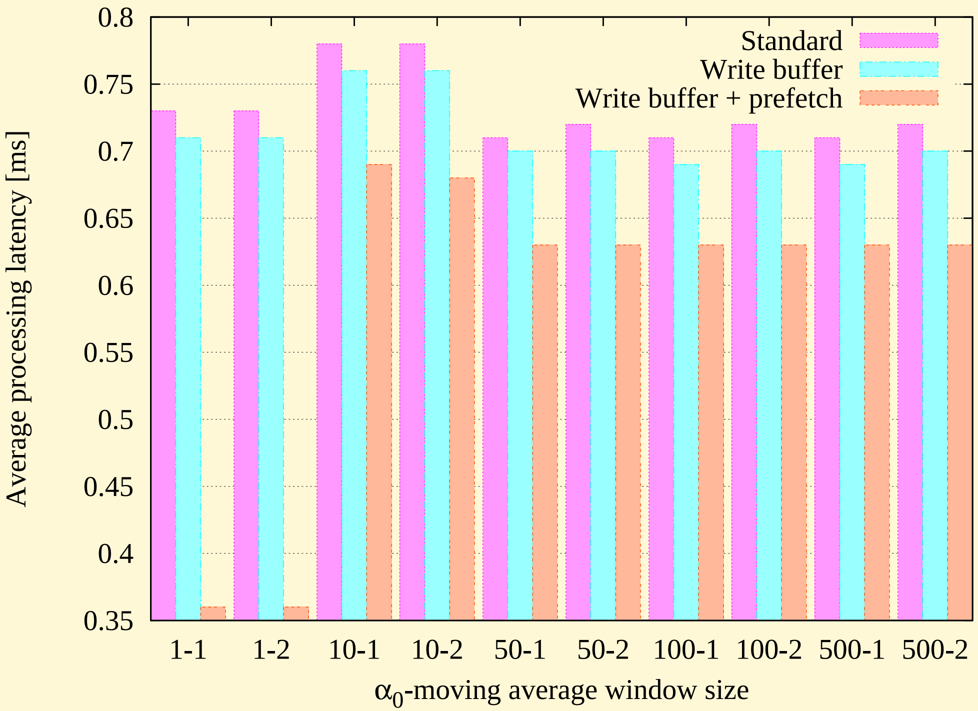
Results

Throughput



- Required bandwidth: 1Gbit/s;
- packet average case;
- number of accelerators: 4;
- $\beta_0 = 1.76 * 10^4$

Processing Latency



Conclusions (1/2)

- We have obtained an algorithm that:
  - is able to distribute IPSec packet processing over multiple processors;
  - supports QoS;
- We have shown that the algorithm works as desired.

Conclusions (2/2)

The scheduling algorithm
is only useful when:

- more than one accelerator is present:
  - having multiple accelerators may allow for scalability at “low” price;
- the system is overloaded:
  - QoS support is provided;
  - the CPU can help processing short peaks over the supported bandwidth.

Future Work

- Test the algorithm in a real system.