A Parallel-Polled Virtual Output Queued Switch with a Buffered Crossbar

Kenji Yoshigoe and Kenneth J. Christensen
Department of Computer Science and Engineering
University of South Florida
Tampa, Florida 33620
{kyoshigo, christen}@csee.usf.edu

This material is based upon work funded by the National Science Foundation under grant no. 9875177

Agenda

- Introduction
- Previous work in input buffered switches
- The PP-VOQ switch
- Simulation evaluation
- Adding support for priority
- Summary and future work
Introduction

- Scalability of output buffered switches is limited by memory bandwidth
  - Buffer memory bandwidth must be \( N \times \) link speed
    - For \( N \) = number of input ports
- Input buffered switches offer a solution
  - Require only a memory bandwidth equal to link speed
  - Scalability is limited by switch matrix scheduling
- Efficient support of variable length packets is of interest
  - For “native” switching of IP packets or Ethernet frames

Previous work in input buffered switches

- Input buffered switches suffer from head of line blocking
  - “Virtual output queues” (Tamir et al. [13]) solve HOL problem

We must schedule inputs and outputs (e.g., for a maximum or maximal match).
Previous work in input buffered switches continued

• Existing class of scheduling algorithms...
  - Parallel request-grant-accept cycles for input and output ports

• Parallel Iterated Matching (PIM) from Anderson et al. [1]
  - Requires fast generation of random numbers

1) Each unmatched input port sends a request for each non-empty VOQ to output ports

2) Each unmatched output port randomly selects a request and returns a grant to the selected input port

3) Each unmatched input port randomly selects a grant and return an accept to the output port (this is a match)

• iSLIP from McKeown [9] improves on PIM
  - Uses rotating counters that "slip" out of sync
  - Does not require fast random number generation

• Many variants of iSLIP now exist
  - Much work has been done in the area

Common to all existing scheduling methods is the need for internal fixed length packets (cells)
Previous work in input buffered switches continued

- Variable length packets supported by using cells internally
  - Segmentation at input port
  - Reassembly at output port

Need speed-up given cells with packet fragments
Have complexity of segmentation and reassembly
Must deliver cells in order (for correct reassembly)

The PP-VOQ switch

- Want to natively support variable length packets
- Must be fair by (at least) packet transmission opportunity
  - May not starve a VOQ or input port
- Must be fast and low in complexity
  - Scalable to large number of ports

We note...
- Constraint on a crossbar is I/O pin count and not gate count
- Stephens and Zhang [13] put buffers within crossbar
The PP-VOQ switch continued

- PP-VOQ switch with buffered crossbar

Feedback

Buffered crossbar

Polling

Classifier

Input port 1

1, 1

1, 2

1, 3

Input port 2

2, 1

2, 2

2, 3

Input port 3

3, 1

3, 2

3, 3

Out 1

Out 2

Out 3

Buffering is 1500 bytes per cross point

- 16x16 = 375 Kbytes
- 32x32 = 1500 Kbytes
- 64x64 = 6000 Kbytes

New

The PP-VOQ switch continued

- Polling of VOQ buffers in each input port...

- Rough algorithm (for each input buffer i, i = 0, ..., N-1)
  - Does not describe feedback from crossbar to input ports

```java
j = 0
While(true)
    j = (j + 1) % N
    if ((VOQ[i][j] is non-empty) && (CB[i][j] is empty))
        Wait for any current transfer to complete
        Start transfer of packet to CB[i][j]

One full poll time must be less than shortest frame transmission time
- 51.2 nanoseconds for 64 bytes at 10-Gbps
```
The PP-VOQ switch continued

- Polling of crossbar buffers for each output port...
- Rough algorithm (for each output buffer \( j, j = 0, \ldots, N-1 \))

\[
i = 0 \\
\text{While(true)} \\
i = (i + 1) \mod N \\
\text{if (CB[i][j] contains a packet)} \\
\text{Wait for any current transmit to complete} \\
\text{Start transmit of packet from CB[i][j]}
\]

One full poll time must be less than shortest frame transmission time
\( \gg 51.2 \) nanoseconds for 64 bytes at 10-Gbps

Simulation evaluation

- Simulation models built of output buffered, iSLIP, and PP-VOQ
  - Used CSIM18 function libraries
  - Models are freely available (need CSIM18 license)
- Traffic models
  - Bernoulli and Interrupted Bernoulli (IBP) arrivals of cells
  - Poisson arrivals of variable length packets
- Variable length packets pulled from empirical distribution
  - From 5 million collected packet lengths on USF backbone
    \( \gg \) "USF distribution"
Simulation evaluation continued

- Histogram of “USF distribution” of packet lengths

![Histogram of packet lengths](image)

- Need speed-up if use internal cells (e.g., in an iSLIP switch)

\[
\text{Speed-up} = \frac{\text{packet length}}{\text{cell length}} \cdot \frac{\text{cell length}}{\text{packet length}}
\]

- For 364.7 byte mean length of “USF distribution” packets and 64-byte cell size...
  - Minimum required speed-up = 1.05
Simulation evaluation continued

• For all experiments...
  - 16 port switch and infinite size buffers
  - For iSLIP used 4 iterations and 64-byte internal cell

• Experiment #1
  - Bernoulli arrival of cells to uniformly selected output port

• Experiment #2
  - Same as experiment #1 except IBP arrivals

• Experiment #3
  - Poisson arrivals of "USF distribution" packets
  - 1x, 1.05x, and 2x speed-ups for iSLIP

Simulation evaluation continued

• Results for experiment #1

   ![Simulation Results](image)

   - Validates with iSLIP and output buffered results (McKeown [8])
Simulation evaluation continued

• Results for experiment #2

Burstiness makes input and output switches the "same"

Simulation evaluation continued

• Results for experiment #3

Note the effects of speed-up on iSLIP performance
Adding support for priority

- Priority is needed for isolation of traffic classes
  - Prelude to QoS scheduling at input ports
- Multi-level polling in PP-VOQ
  - $P \times N$ VOQs per input port ($P$ = # of priority levels)
  - Poll "high-priority" as long as high-priority packets queued
- iSLIP two-level priority implemented per McKeown description
- Experiment defined to evaluate priority
  - Reproduce experiment #3 and add a "CBR" stream of 1500 byte packets from input port 0 to output port 0
  - Measure delay of CBR packets (w/ and w/out priority)

Adding support for priority continued

- Results for priority experiment

See paper for more details
Summary and future work

- New VOQ switch for native support of variable length packets
  - Parallel polling in input and output ports
  - Buffering at crossbar cross points
  - Better cell switching performance than iSLIP

- Future work
  - Investigate feasibility of buffering within a crossbar
  - Prove fairness and stability
  - Bound switch delay
  - Explore support for flow-level scheduling at input port
    - Can be done if switch delay is small and bounded???