

# GMDS: Hardware implementation of novel real output queuing architecture

R. Arteaga, F. Tobajas, R. Esper-Chain, V. de Armas and Roberto Sarmiento  
Institute for Applied Microelectronics (IUMA). DIEA. University of Las Palmas de Gran Canaria  
Campus Universitario de Tafira, S/N. 35017. Las Palmas de Gran Canaria, Spain  
rarteaga, tobajas, esper, armas, roberto@iuma.ulpgc.es

## Abstract

*In this paper, a real output queuing switch prototype implementation is presented. This implementation is based on a novel high speed multidrop backplane and a general purpose line card which includes a Virtex-II 6000 FPGA. This switch is named GMDS (Gigabit MultiDrop Switch) and its main features are the switch matrix replacement by the multidrop backplane -increasing system reliability-, variable length packet switching support -avoiding bandwidth efficient loss-, multiple output queuing structure for supporting QoS (Quality of Service) and a minimum speedup.<sup>1</sup>*

## 1 Introduction

Switching packets architectures can be classified in many ways. Usually time-division or space-division, single stage or multiple stage, blocking or non-blocking, and single path or multiple path architecture classification [1] can be considered. However, almost all of them suffer from output port contention which is the main source of the performance degradation. In order to improve the performance, buffering strategies must be employed.

The output queuing (OQ) switch allows all incoming packets to arrive at the output port. This architecture can achieve 100% throughput because there is no HOL (Head Of Line) blocking. However, if all the packets are destined to the same output port, in a  $N \times N$  output queuing switch, memory speed will be  $N$  times faster than external link speed, limiting switch scalability. However, memory speed can be reduced taking into account that it is not usual that all the incoming packets ask for the same output port. So, this policy introduces loss in the switch architecture, relaxing memory speed-up. This concept is called the knockout principle [2].

---

<sup>1</sup>This work was partially supported by the Spanish Ministry of Education and Science (MEC) under project ENDIVIA TEC2005-08138-C02-01/MIC.

The input queuing switch suffers from HOL blocking problem and limits the throughput to 58.6% for uniform traffic [3]. In order to increase the switch's throughput some techniques can be used such as window policy [3], internal speedup and VOQ (Virtual Output Queuing) [4]. However, VOQ switches require a fast and intelligent arbitration mechanism. Since the HOL packets of all  $N^2$  logical queues in the input buffers need to be arbitrated in each time slot, becoming the bottleneck of the switch.

In shared memory switches, a memory is shared by all the input and output ports storing all the packets. Since memory is shared among all the output ports, a more efficient use of memory space can be achieved as compared to output queuing. However, memory should operate  $2 \times N$  times faster than the external link speed and requires a complicated control mechanism.

Previous architectures are not suitable in order to satisfy the requirements of high switching capacity and high performance at the same time. However, combining buffering techniques, introducing speedup and increasing hardware resources can be made possible to reach a trade-off between performance and hardware requirements in order to emulate output queuing performance.

Some examples with input-output buffering were presented, such as CCF (Critical Cell First) or LIHP (Last In, Highest Priority) based on an implementation of priority lists [5]. Another algorithm is LOOFA (Lowest-Output-Occupancy-Cell-First Algorithm) [6]. This input-output-queued architecture use a speedup of 2 and provides 100% throughput. Others researchers [7] have studied CICQ (Combined Input Crossbar Queued) architectures proposing different scheduling algorithms in order to emulate OQ performance. These architectures require a VOQ buffering technique as well as limited internal memory. In these cases, speedup is also necessary.

However, researchers tried to resolve the speedup issue replicating different elements. In the MOQ (Multiple Output Queuing) architecture [8], a switch with performance similar to an OQ switch and comparable hardware resources to a VOQ architecture is presented. This archi-

tures does not need any speedup. In the MIOQ (Multiple Input/Output Queued) architecture [9], Lee and Sheo show theoretically that MIOQ switch can match the output queuing exactly with no speedup of any component using two parallel switches.

In this paper, a real output queuing switch named GMDS (Gigabit MultiDrop Switch) is presented. The main features of this architecture are introduced in the next section. In Section 3, a deep module description is presented. Hardware resources needed by its implementation are detailed in Section 4. Finally, performance and main conclusions are presented.

## 2 GMDS overview

The reference architecture presented in this paper is a real output queuing switch. This architecture is based on multidrop transmission, specifically in a high speed multidrop backplane avoiding the need of a switch fabric. In Fig 1, an implementation of a  $4 \times 4$  switch based on the GMDS architecture is presented.

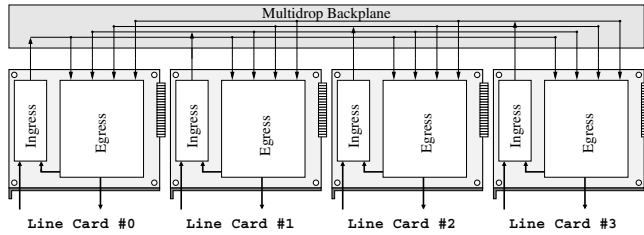


Figure 1. A 4x4 GMDS general architecture

Multidrop feature implies that every line card has a dedicated asynchronous uplink connected in point-to-multipoint to all the line cards in the switch. Consequently, inherent multicast/broadcast transmission and variable length packets are supported.

In Fig 1, four line cards named as #0,#1,#2 and #3 are interconnected through this high speed multidrop backplane. Each line card has an input port (*Ingress*) and an output port (*Egress*). In the GMDS, the *Ingress* is transparent to packets, which are inserted in the multidrop backplane. Each *Egress* is listening to all *Ingress* connected to the backplane, decoding the destination address. Only packets with destination addressed related to each particular *Egress* are accepted and enqueued. The main features of this switch architecture are:

1. Non Switch Fabric architecture, increasing system reliability.
2. Real Output Queuing structure, avoiding HOL blocking problems of systems based on input queues.
3. Variable length packet support. This design has a dedicated link for each transmitter, so, inherently, variable

length switching without speed-up requirement is supported.

4. Segmentation & reassembly of packets are not needed, improving overall efficiency, since functions like padding are not required.
5. Broadcast/Multicast natively supported.
6. Packets classification in multiple output queues based on traffic classes and sources according to Differentiated Services philosophy.
7. Enhanced QoS support, since scheduling can be performed over output enqueued traffic.
8. Accurate end-to-end flow control based on individual threshold configuration in each queue.

## 3 Detailed Description

The architecture of the GMDS is composed by two main elements: the *multidrop backplane* and the *line cards*. The line cards are responsible of frame reading/processing/enqueuing. The multidrop backplane, which replaces the switch fabric, distributes the packets among the line cards.

### 3.1 Multidrop Backplane

A gigabit multidrop backplane is proposed to be based on broadband power splitters that accept an input signal and deliver multiple output signals with specific phase and amplitude characteristics, while maintaining a good impedance match at all ports. These features make maximum data bit-rate of a multidrop serial link to be limited principally by the PCB fabrication material, which theoretically allows achieving *10Gbps* using FR4 substrate.

In [10], a new lossless asymmetrical broadband power splitter configuration with matching trace impedance was introduced as the basis for a gigabit multidrop serial link. The basic configuration of this power splitter consists of an internal bifurcated transmission line in which the characteristic impedances of each bifurcation,  $Z_1$  and  $Z_2$ , must enforce  $Z_0 = Z_1 // Z_2$ , with  $Z_0$  being the characteristic impedance of the source transmission line. Since the impedance formed by the two bifurcated transmission lines in parallel perfectly matches the characteristic impedance  $Z_0$ , then power transfer will be maximal at each receiver end. Furthermore, if termination resistors are sized according to the characteristic line impedance at each receiver end, then output voltage is the same as the source voltage applied at the transmitter end.

### 3.2 Line Cards

The detailed architecture of each line card in a  $4 \times 4$  GMDS is shown in Fig 2. It is composed by the follow-

ing modules: IM (Ingress Manager), FF (Frame Filter), QM (Queue Manager), MX (Multiplexer), SM (Status Manager), SC (Scheduler) and Memory System.

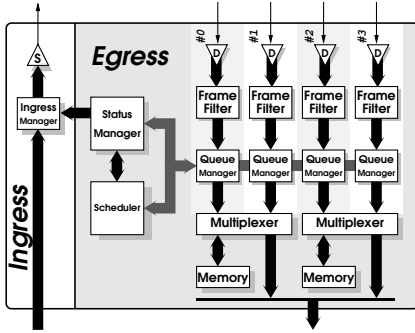


Figure 2. Line card architecture

### 3.2.1 Ingress Manager

The *Ingress Manager* module adds some overhead information to the incoming packets, composing a *multiframe*. This *multiframe* is serialized and transmitted through the multidrop backplane in a dedicated uplink. The overhead inserted by the *Ingress Manager* includes status & flow control information, provided by the *Status Manager* module, as well as alignment & error control information. The incoming packets are not enqueued nor delayed beyond the time required to compose the multiframe and to serialize the information. The rate on the backplane is slightly greater than the incoming frame rate to accommodate the overhead information.

### 3.2.2 Frame Filter

Each line card has a *Frame Filter* (FF) module dedicated to each multidrop downlink on the backplane. Each *Frame Filter* deserializes the *multiframe*, extracts each packet, and selects packets which destination address matches the line card id. To perform this operation, a pattern matching algorithm with a set of programmable patterns are included in the *Frame Filter* module. Also a direct port assignment in the frame header can be done, in order to support multicast. Selected packets are transferred to the *Queue Manager* module in order to be enqueued in the *Memory System*. Also the status & flow control information contained in the multiframe is decoded and relevant information to the line card is extracted.

The *Frame Filter* module also performs clock rate adaptation between remote *Ingress Manager* and local *Egress* clock, since to simplify clock distribution, each line card can operate based on their own local clocks.

### 3.2.3 Queue Manager

The *Queue Manager* module assigns a slot in the *Memory System* to each incoming packet. Packets are enqueued separately by source downlink and packet class. In the current implementation, the *Queue Manager* module uses fixed size slots of the maximum packet size. The *Queue Manager* module also reports to the *Status Manager* when a new packet is received, together with its class and source downlink. In addition, the *Queue Manager* module selects a slot to be dequeued when a specific source/class is requested to be read by the *Scheduler* module.

### 3.2.4 Multiplexer

In the GMDS, the memory bandwidth is splitted among the active *Frame Filters/Queue Managers* in the line card. However, the data rate and the memory rate will not necessarily match. The *Multiplexer* is a module that couples data flows from one/several *Queue Managers* to the *Memory System*. In the presented demonstrator, low-cost 8ns cycle-time ZBT memories are used. Since memory bandwidth is 4 Gbps per device, each two *Queue Managers* are assigned to a single memory. The internal clock of the *Multiplexer* runs at 3× the nominal frequency of the *Queue Manager* module, so in one *Queue Manager* cycle the multiplexing system is able to write from both assigned sources on each memory, and has an extra cycle reserved for a read operation. When the number of ports is increased, a memory device has to be added per each pair of ports, together with a *Multiplexer* module.

### 3.2.5 Status Manager

The *Status Manager* module collects information about packets incoming/outgoing of the *Egress*, keeping an accurate frame count on any queue. It reports about the empty/fillness levels of each single queue to the *Scheduler*. The SM keeps some threshold levels per class to trigger flow control mechanisms. Since packets are queued per source/class, an end-to-end flow control can be supported.

### 3.2.6 Scheduler

Packet scheduling specifies the queue service discipline in GMDS. Since packets may depart from the same outgoing interface, packet scheduling also enforces a set of rules in sharing the output link bandwidth. Consequently, the *Scheduler* module defines the performance of the packet switch and provides the QoS features, being therefore a key module on the system. In this paper, a *Deficit Round Robin* packet scheduler is presented in order to evaluate basic functionality of GMDS.

## 4 Implementation

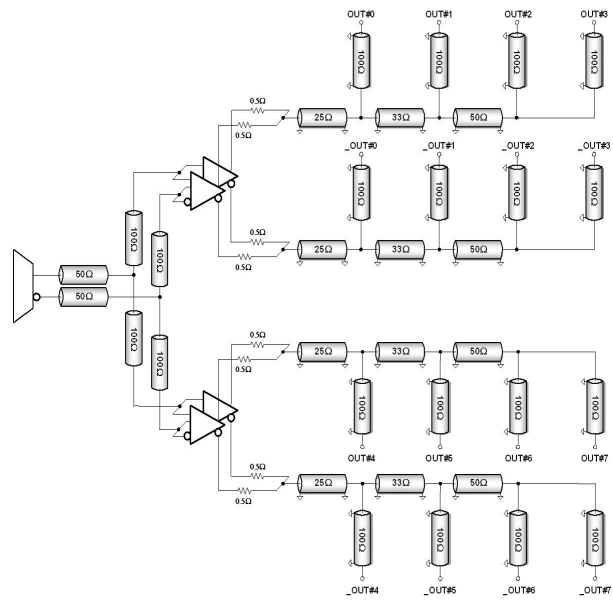
In order to demonstrate the feasibility of the proposed switch, a demonstrator has been implemented. This demonstrator is capable to operate in configurations up to  $8 \times 8$ . The line rate has been selected to be  $1\text{Gbps}$ , which is high enough to face signal integrity concerns, and ease to develop a low cost FPGA based demonstrator.

### 4.1 Multidrop Backplane

The fabricated gigabit multidrop serial backplane consists of eight line cards interconnected through eight differential 1-to-8 multidrop serial links, with receivers from each line card receiving the same set of dedicated multidrop links. In this backplane, proper selection of the source transmission line characteristic impedance  $Z_0$  is a key issue on the design of each 1-to-8 multidrop serial link.

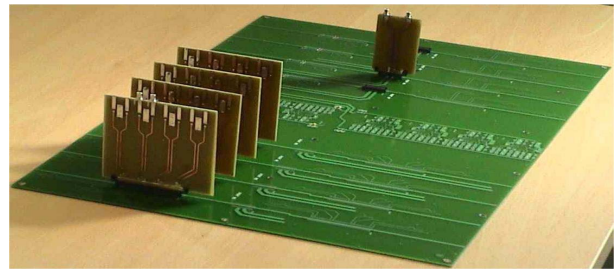
For the implementation of each differential 1-to-8 multidrop serial link,  $25\ \Omega$  was adopted as the source transmission line characteristic impedance  $Z_0$ . In addition, to minimize highest impedance value required on the  $8 \times 8$  gigabit multidrop backplane construction, each differential 1-to-8 multidrop serial link was implemented through two parallel arranged differential 1-to-4 asymmetrical power splitters with matching trace impedance, as shown in Fig 3. With this implementation, the transmitter should be able to drive a load of  $25\ \Omega$ , and the end termination resistors at each receiver should match  $100\ \Omega$ . These values represent a compromise between trace width requirements and highest controlled impedance achievable in standard FR4 substrate. In order to effectively drive a load of  $25\ \Omega$ , a pair of parallel arranged Maxim MAX9400 buffers with open inputs and open emitter outputs are used at the transmitter end, while an additional 1-to-2 power splitter with short connections between buffer pairs provide best impedance match and minimize capacitance at the transmitter output.

To build the designed  $8 \times 8$  gigabit multidrop serial backplane, a 4-layer PCB with a signal:plane:plane:signal stackup was used. This configuration produces a PCB that has trace layers directly on top of a ground plane. Board layout is based on differential transmission lines implemented with microstrip traces on board surface layers. The impedance of each differential transmission line is determined by the configuration of the pair of conductors and the relationship between their signal polarities, in addition to the trace geometry, distance to the nearest ground plane and the dielectric constant of the material between the trace and the ground plane. Differential pair traces are closely together to ensure that noise is coupled as common-mode and  $45^\circ$  bevels are used to avoid impedance discontinuities. In addition, differential pairs are widely spaced to prevent crosstalk.



**Figure 3. Differential 1-to-8 multidrop serial link implementation**

In Fig 4, a photo of the fabricated  $8 \times 8$  gigabit multidrop serial backplane with a transmitter and four receiver test line cards, is shown. Final dimensions for the prototype are  $34\text{cm} \times 52\text{cm}$ . Experimental results demonstrate a satisfactory operation at  $3\text{Gbps}$  data rate, which implies almost one order of magnitude above existing commercial multidrop backplane supporting less number of line cards.



**Figure 4. Fabricated  $8 \times 8$  multidrop gigabit serial backplane prototype**

### 4.2 Line cards

The main challenge in the line card design is to find a partition that fits in a monolithic implementation. The partition proposed is based on a two devices chipset: a *Queue Engine* and a *Scheduler*. The *Queue Engine* proposed, which is shown in Fig 5, includes all the logic related to

a line card in a configuration  $1 \times 4$ , but the *Scheduler* module. In the current demonstrator, the *Memory System* was implemented outside the *Queue Engine*, in order to evaluate the impact on memory size in the performance of the GMDS. However, depending on the target technology and queue model implemented, this memory can be integrated within the *Queue Engine* to improve integration level.

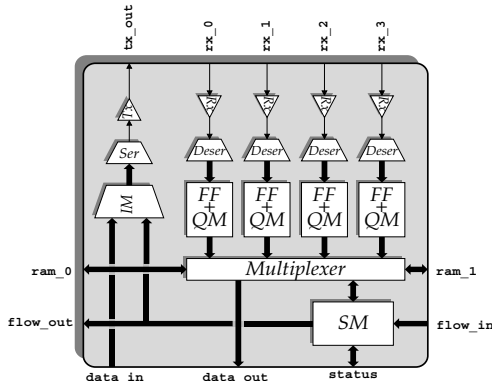


Figure 5. Queue Engine Unit

#### 4.2.1 Queue Engine

Each *Queue Engine* contains a complete *Ingress*, which can be disabled if not used. Output data of *SM* and *Multiplexer* modules are connected via specific buses. With this architecture, the number of ports, and so the scalability, is not limited by the ability of the *Scheduler* to manage information from all the *SM* modules. In Fig 6 a sample configuration of a line card for a  $8 \times 1$  GMDS is presented. In this configuration only left *Queue Engine* device has its *Ingress* enabled. Status & flow control messages are transmitted in a daisy chain approach.

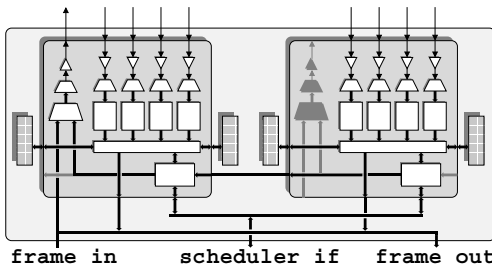


Figure 6. Configuration for a  $8 \times 1$  line card

The *Scheduler* is connected to the *SM* in the *Queue Engine* through a status bus, (marked as *scheduler\_if*). When a particular source/class is selected to be dequeued, the associated packet is transmitted through a data bus (marked as *frame\_out*). Only the selected queue write on the data bus at any time.

Module	Slices	FF	BRAM
IG	830	1140	3
FF	1408	1494	8
QM	790	797	0
MX	542	841	0
SM	4882	4482	0

Table 1. Synthesis summary on a Virtex-II

#### 4.2.2 Scheduler

The *Scheduler* design should take advantage of all the specific features of the GMDS. Instead of including a *Scheduler* within the *Queue Engine*, it was implemented outside the Egress system, on external device. This *Scheduler* device may be configured depending on the features desired for different traffic configurations. The results presented on this paper are based on a *Deficit Round Robin*.

#### 4.2.3 Synthesis Results

The GMDS demonstrator was described in *Verilog HDL* and implemented on a FPGA device. For the *Queue Engine*, it has been selected Xilinx's Virtex-II device. The synthesis results are shown in Table 1. According to these results, the *Queue Engine* will fit on a 1.5-million gates device (such as XC2V1500), and a *Queue Engine* capable of managing up to 16 input ports on a XC2V6000. These synthesis describes the *Queue Engine* as a medium complexity device, that can be implemented in a very-low-cost medium-complexity ASIC with gate count below 1 million. Serializers & Deserializers were not integrated, but specific external devices were used.

In the case of usual chip set solution (queue engine and switch fabric), the switch fabric chip is replaced by a passive multidrop backplane with only two Maxim MAX9400 chips per input. This solution provides an enormous reliability and a remarkable hardware reduction.

The overall local frequency on the line card is kept below  $40MHz$ , while the memory access is performed with a clock cycle of  $8ns$ . For a  $2.5Gbps$ , the local frequency should be increased to  $100MHz$ , which is not a challenge for current technologies. If external RAM is used, standard low cost devices, such as RDRAM should be able to absorb easily the required throughput. For  $10Gbps$  systems, local frequency will be kept at  $100MHz$  and 128 bit-wide systems will be used instead of 32 bits as current implementation.

## 5 Performance

The performance of GMDS was compared among Output Queue architecture with FIFO policy and fixed length

ATM packets, Input Queue architecture with four iteration iSLIP policy and also fixed length ATM packets, and CICQ architecture with RR/RR algorithms and fixed length packets (1500 bytes) [11]. The Y axis represents the Mean Queuing Delay in packets which includes the switch matrix delay and the queuing delay. This parameter does not include transmission time on the output link. The X axis represents the traffic load measured at the traffic generators.

The simulation conditions of the proposed switch include a Bernoulli traffic model generator with variable length packets from 1 byte to 256 bytes support. Packets are uniformly distributed among all the destination ports. In the Fig 7, the delay represented is the sum of the delay from the *Ingress*, multidrop backplane, *Egress*, RAM and *Scheduler* modules. It is important to remark that the Mean Queuing Delay from *Ingress* module, multidrop backplane, and *Frame Filter* module is almost constant with a slight variation due to the multiframe composition and insertion process in the multidrop backplane. All these results are obtained from Verilog HDL simulations.

It should be noticed the difficulty of comparing the proposed architecture with previous architectures. Firstly, there are not available real output queue switches with similar performance, that is, variable length packet switching without bandwidth efficiency losses due to not using segmentation & reassembly mechanisms and not based on switch fabric. Secondly, the results showed in Fig. 7 are based on clock cycle delay from RTL Verilog HDL simulation. The three previous references are based on high level model, therefore, their results are more optimistic than our results, however, they are less realistic. In a high level model of GMDS, the complete system performance is similar to CICQ reference.

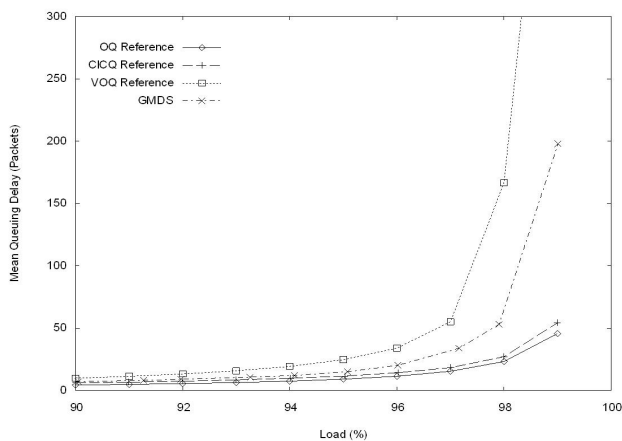


Figure 7. GMDS delay performance

## 6 Conclusions

In this paper a real output queuing packet switch is presented. A demonstrator for a 1 Gbps based on a  $8 \times 8$  prototype multidrop backplane and commercial FPGA has been build. The main features of this implementation are the switch matrix replacement by the backplane increasing system reliability, variable length packet switching supported avoiding bandwidth efficient loss, multiple output queuing structure based on traffic classes and sources for supporting QoS (Quality of Service) and a minimum speedup. The results, shown in the previous section, demonstrate a similar behaviour to an ideal CICQ switch.

## References

- [1] Newman, P. ATM technology for corporate networks. *IEEE Communications Magazine*, 30(4):90–101, April 1992.
- [2] Yeh, Y.-S., Hluchyj, M.G., and Acampora, A. S. The knockout switch: a simple, modular architecture for high-performance packet switching. *IEEE Journal on Selected Areas in Communications*, 5(8):1274–1283, October 1987.
- [3] Karol, M.J., Hluchyj, M.G., and Morgan, S.P. Input versus output queueing in a space division switch. *IEEE Transactions on Communications*, COM-35(12):1347–1356, December 1987.
- [4] Ali, M.M., and Nguyen, H.T. A neural network implementation of an input access scheme in a high-speed packet switch. *GLOBECOM'89*, 2:1192–1196, November 1989.
- [5] Chuang, S.-T., Goel, A., McKeown, N., and Prabhakar, B. Matching output queuing input/output queued switch. *IEEE Journal on Selected Areas in Communications*, 17(6):1030–1039, June 1999.
- [6] Krishna, P., Patel, N., Charny, A., and Simcoe, R. On the speedup required for work-conserving crossbar switches. *IEEE Journal on Selected Areas in Communications*, 17(6):1057–1066, June 1999.
- [7] Magill, R.B., Rohrs, C.E., and Stevenson, R.L. Output-queued switch emulation by fabrics with limited memory. *IEEE Journal on Selected Areas in Communications*, 21(4):606–615, May 2003.
- [8] Danilewicz, G., Glabowski, M., Kabacinski, W., and Kleban, J. Packet switch architecture with multiple output queueing. *GLOBECOM'04*, 2:1192–1196, November-December 2004.
- [9] Lee, H.-I., and Seo, S.-W. Matching output queueing with a multiple input/output-queued switch. *IEEE/ACM Transactions on Networking*, 14(1):121–132, February 2006.
- [10] Esper-Chain, R., Tobajas, F., Tubio, O., Arteaga, R., de Armas, V. and Sarmiento, R. A gigabit multidrop serial link for high-speed digital systems based on asymmetrical power splitters. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 52(1):5–9, January 2005.
- [11] Yoshigoe, K., and Christensen, K. An evolution to Crossbar Switches with Virtual Output Queuing and Buffered Cross Points. *IEEE Network*, 17(5):48–56, September-October 2003.