A Novel Transistor Level Implementation of a High Speed Packet Switch

Shanmugam Arumugam[1], Shanthi Govindaswamy[2], and Praveen Kumar Boya[2]
[1]Bannari Amman Institute of Technology and Science, India
[2]PSG College of Technology, India

Abstract: *High speed packet switches are inevitable for ultra high data rate networking systems. The throughput of these switches has to be ideally 100% for effective utilization of the network. While Output Queued (OQ) switches have the optimal delay-throughput performance for all traffic distributions, they require N-times speed up in the fabric that limits the scalability of this architecture. An Input Queued (IQ) switch is desirable for high speed switching, since the internal operation speed is only slightly higher than the speed of the input lines. However, the input queued switch has the critical drawback that the overall throughput is limited to 58.6% due to the Head-of-Line (HOL) blocking phenomenon. In this paper, we present a novel transistor level implementation of an IQ packet switch, using TSPICE. Our circuit has a regular structure and low transistor count. Our simulation results indicate that the circuit may be used to implement switches working well beyond 1 GHz.*

Keywords: *High speed packet switches, packet switch schedulers, IQ architecture.*

Received December 4, 2004; accepted August 8, 2005

1. Introduction

Network traffic is increasing at a tremendous speed. To cater for the growing demands on networks, it is essential that switches operate at high speeds, provide maximum throughput and introduce minimum queuing delay. Among the Input-Queued (IQ), Output-Queued (OQ) and Combined Input-Output Queued (CIOQ) switch architectures, we consider the IQ architecture as shown in Figure 1 on account of the low memory speed requirements. To over come the limitations of this scheme, such as reduced throughput due to HOL blocking several strategies have been introduced in the literature such as windowing technique, input smoothing, expanding the switch internal bandwidth either by enlarging the switch dimension or replicating the switch fabric, dropping the blocked HOL cells in the FIFO queue etc., for ultra high speed switches, it has been proved that the Virtual Output Queued (VOQ) architecture is highly efficient. However, this scheme requires an efficient cell scheduling algorithm, which should resolve output contention, provide high throughput and satisfy Quality of Service (QoS) requirements and be capable of implementation as a low cost Integrated Chip. Existing scheduling algorithms [3, 4, 5, 8, 9, 10] are complicated and costly to implement in hardware. A survey of scheduling algorithms is given in [11]. In this paper, we present a cell scheduler that utilizes nearly 100% of the available link bandwidth and requires significantly less hardware to implement. We propose a transistor level circuit implementation that features high level of modularity, has a very low interconnect and transistor count and achieves near optimal scheduling within linear processing time (N clock cycles). The simulation results for random, poisson and bursty arrivals are shown, for different switch sizes, when the switches are loaded to near maximum cell arrival rate, for different burst lengths, in [6]. In this paper, the performance has been verified through simulation, and hardware design has been verified using TSPICE in 0.25μm technology.

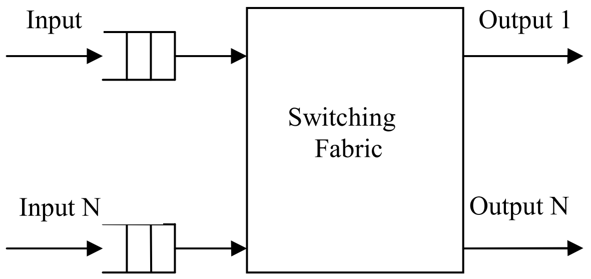

Figure 1. Generic model of the input-queued switch.

The existing algorithms for scheduling in IQ switches, like Parallel Iterative Matching (PIM), Input Round Robin Matching (IRMM) and IRMM-SLIP [1, 2, 7] are difficult to implement in hardware, since for an N-input N-output switch we require 2N round robin arbiters and the interconnect complexity is proportional to N2. Hence these architectures are not scalable. We present here a transistor level implementation for the packet scheduler which is modular and needs a linear processing circuit. The proposed circuit has mixed digital and analog architecture, wherein the weight matrix is calculated by analog means.

2. Matrix Unit Cell Scheduling Algorithm

Matrix Unit Cell Scheduling (MUCS) resolves output contention by computing a traffic matrix P. Initially, each row of P indicates the status of the number of queued cells at each input port of the switch through an N-element traffic vector. Each element xij of P is a positive integer, which indicates the weight associated with the queue at the input port 'i' dedicated for the output port 'j'. When $x_{ij} > 0$, its value indicates that the cells which are having the highest weight among all the buffered cells are to be switched from input port i to output port j. when $x_{ij} = 0$, it indicates no cell is ready to be switched from input port i to output port j.

The MUCS algorithm selects an optimal set of entries as winning cells from matrix P according to the weight w_{ij} of x_{ij}. It calculates the heaviest entries under the constraint that no more than one cell is transmitted from an input port and no more than one cell is delivered to an output port simultaneously. In our implementation of MUCS, the selection process can be performed fully in parallel by the hardware.

Let A be the size of the switch and B the size of reduced matrix P' of P. At each iteration, the algorithm is described as follows:

1. Initially, A = B and P' = P.
2. For each iteration, the entry weight w_{ij} is assigned to every entry of P'. The value of w_{ij} is determined by:

$$w_{ij} = \begin{cases} \dfrac{x_{ij}}{wr_i} + \dfrac{x_{ij}}{wc_{ij}}, & if \;\; x_{ij} \neq 0 \\ 0, & otherwise \end{cases}$$

where

$$wr_i = \sum_{j=1}^{A} [\text{number of } x_{ij} > 0]$$

$$wc_j = \sum_{i=1}^{B} [\text{number of } x_{ij} > 0]$$

3. The entry with the heaviest weight is selected as the winner. Multiple entries can be selected as winners simultaneously if they all have same weight and do not reside in the same row or column. Once the winners have been selected, a new reduced matrix, P', is formed as all elements of current P', excluding the rows and columns with winning cells. The algorithm terminates when all entries have been treated.
4. If a tie occurs, winners are selected randomly.

3. MUCS Implementation

We consider an N x N packet switch which uses a suitable cell scheduling algorithm to resolve contention among inputs for a particular output. For an N x N switch, N2 process units are needed, each of which corresponds to an element of the traffic matrix. The block diagram of a single cross point of the switch is given in Figure 2. The complete switch core may be built by using replicas of the above implementation. As shown in Figure 2, a single cross point is built using 2 current dividers, 3 current controlled switches and a latch comparator in addition to a row source and column source of current. In the original algorithm, the value of x_{ij} is represented by the value of the corresponding capacitor. When fixed capacitors used, the traffic matrix P becomes a binary matrix after being loaded into the MUCS circuit and the solution is optimized for the maximum throughput. When variable capacitors are employed, different values of xij can be included in the scheduling. They use a heuristic strategy to decide the switch configuration during each time slot (switch slot). In our modified version, a previously determined QoS weight is associated with each VOQ and is incorporated as the width of the corresponding transistors, in each cross-point.

The original implementation of a single cross-point of the N x N high speed packet switch is as shown in Figure 2. In our version, the current dividers are implemented by using current mirror circuits as shown in Figure 3. The transistors M1 and M2 form a PMOS-current mirror, while the transistors M4 and M5 form the second PMOS current mirror. The divided currents are proportional to the weight vector; this is achieved by suitably selecting the size of the current mirror transistors. Current flowing through M2 transistor of the current mirror circuit can be changed and hence this itself acts as the current divider circuit. Depending on the Row current source and the column current source, the current mirror can be designed and the current flowing through the individual N x N switches can be controlled. Currents coming from the row source and the column source can be controlled by using the M3 and M6 switches for a proper row and column selection respectively. M7 is used as a control switch to disconnect the capacitor, C from the rest of the circuit after charging to the proper value. The transistor M8 is use do reset the capacitor. The output information (either won or lost) is available across the capacitor. By using two inverters in series at the output of capacitor, the output voltage level can be increased and the output of first inverter may be used to control the transistor M7 which is used to switch off the circuit and disconnect the capacitor from the rest of the circuit. The current sources we have considered are DC current sources. A 2.5V power supply and a 10 pF capacitor have been used for the simulation.

4. Results

Figure 4 shows the variation of current flowing through the capacitor, for various values of Width (W) of transistor M2. The capacitor, (C), in Figure 3 charges at different intervals of time depending on the current flowing through it, which leads to disabling other cells in the N x N switch. The capacitor reaches appropriate threshold voltages at different time intervals. Figure 5 helps to determine the time taken to reach a particular threshold voltage, for different values of charging currents. By properly designing the inverter with different threshold voltages, other cells can be disabled. Using this technique the circuit detects the cell that reaches the maximum voltage at the earliest.

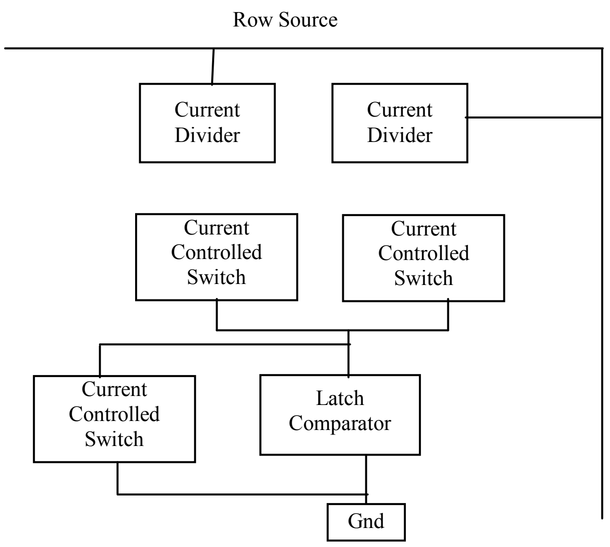
Row Source
Current Divider
Current Divider
Current Controlled Switch
Current Controlled Switch
Current Controlled Switch
Latch Comparator
Gnd
Column Source

Figure 2. Block diagram of a single cross point of (N x N) high speed packet switch (original version).

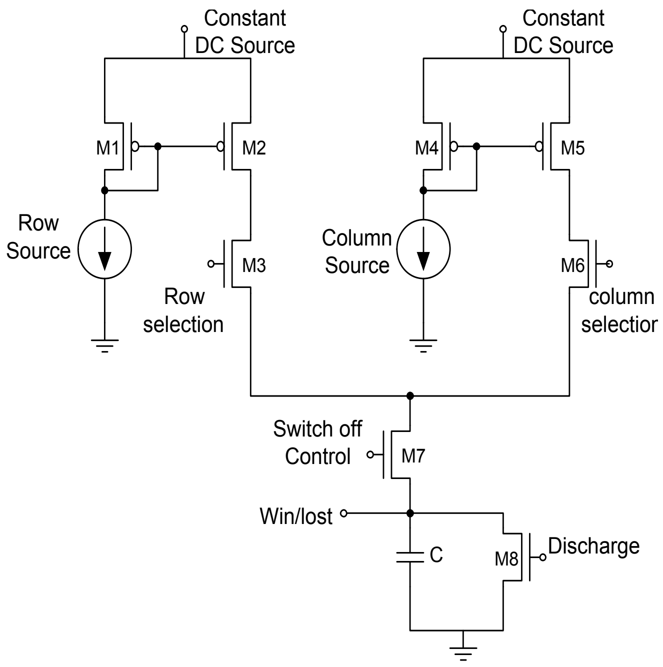

Figure 3. Implementation of a single cross point of N x N high speed packet switch (modified version).

From Table 1, it is clear that the current is varying vastly with respect to the width of transistor M2. Hence, by properly choosing the width of M2, the current flowing through the different cells, (which is the row current), can be controlled. Similarly, the column current can be controlled by varying the width of transistor M5. The current variation is large enough to differentiate the voltage levels at the capacitor which indicates the win or lost condition.

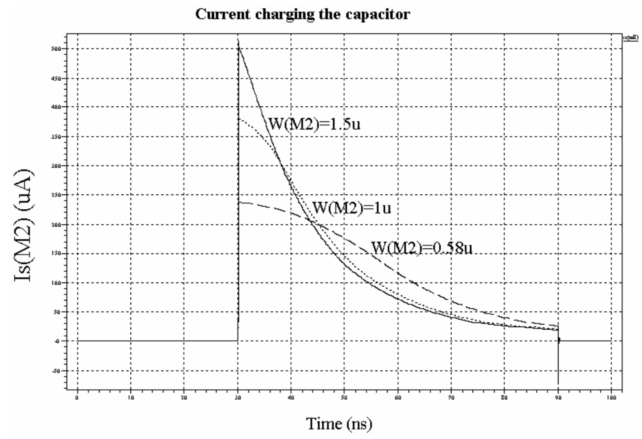

Figure 4. Output current waveforms for various widths (W) of the transistor M2.

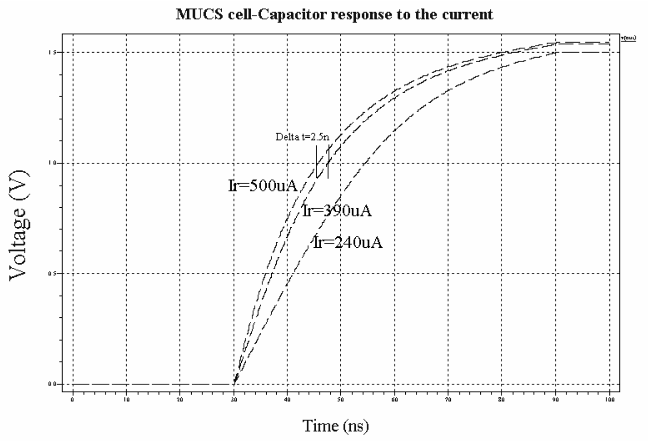

Figure 5. Voltage response of the capacitor for various values of charging current in a single cross point.

Table 1. Variation of source current with width (W) of M2 transistor.

W Value of M2 Transistor	.58μ	1μ	1.5 μ
Current at Source of M2	240μA	390μA	500μA

For selecting the winning cell, a proper cell threshold voltage is fixed for all cells. The cell which reaches the threshold voltage first is treated as winner in the complete matrix of N x N cells. Let the threshold voltage be 1V for the above case. From Figure 5 it is clear that, for different currents, the capacitor takes different time intervals to reach the threshold voltage of 1V. For a current of 500μA, the capacitor reaches the threshold voltage at 45.4ns, for 390μA it is 47.9ns and for 240μA it is 54.5ns. The time difference (delta) between two currents in the least case is 2.5ns which is large enough to disable other cells from the N x N cell matrix by giving the inverted win output to the M7 input of all the other cells in the same row and the same column. Using this technique the cell that reaches the maximum voltage first can be detected.

Each processing unit requires 12 transistors and 1 capacitor only (after including 2 inverters at the win/lost output of Figure 3) which is far less when compared with the circuit implemented in [6]. So, the area occupied by the N x N cell matrix is also less. For an 8 x 8 cell, our method requires only 768 transistors and 64 capacitors, compared to 1856 transistors and 128 capacitors in [6]. This implementation dissipates lesser power and occupies less area compared to the other implementations. TSPICE was used to verify the design. Simulation results indicate that with 0.25μm CMOS technology, the MUCS circuit can operate at clock frequency of 1 GHz. Using larger current sources, reducing the size of capacitor, or reducing the transistor feature size can make the MUCS circuit operate even faster.

5. Conclusion

The MUCS algorithm provides a highly-effective mechanism for scheduling input-buffered packet switches. It provides near-optimal throughput and can be implemented with simple hardware. The performance of the algorithm has been verified through simulation and its hardware design has been verified with TSPICE. Our future work involves reducing the size of the capacitor and transistors used and using larger current sources, which will lead to faster operation.

References

[1] Anderson T., Owicki S., Sexe J., and Thacker C., "High-Speed Switch Scheduling for Local-Area Networks," *ACM Transactions on Computer Systems*, vol. 9, no. 2, pp. 101-124, May 1991.

[2] Duan H., "Design and Development of Cell Queuing Processing and Scheduling Modules for the iPOINT Input Buffered ATM Testbed," *PhD Dissertation*, University of Illinois at Urbana, Champaign, 1997.

[3] McKeown N. and Anderson T. E., "A Quantitative Comparison of Scheduling Algorithms for Input-Queued Switches," *Computer Networks and ISDN Systems*, vol. 30, no. 24, pp. 2309-2326, December 1998.

[4] McKeown N., "iSLIP: A Scheduling Algorithm for Input-Queued Switches," *IEEE Transactions on Networking*, vol. 7, no. 2, pp. 188-201, April 1999.

[5] McKeown N., "Scheduling Algorithms for Input-Queued Cell Switches," *PhD Thesis*, University of California, Berkeley, 1995.

[6] McKeown N., Anantharam V., and Walrand J., "Achieving 100% Throughput in an Input-Queued Switch," *in Proceedings of INFOCOM Conference*, San Francisco, vol. 21, pp. 296-302, March 1996.

[7] McKeown N., Izzard M., Mekkittikul A., Ellesick B., and Horowitz B., "The Tiny Tera: A Packet Switch Core," *IEEE Micro*, vol. 17, no.1, pp. 27-40, February 1997

[8] Mekkittikul A. and McKeown N. "A Starvation-Free Algorithm for Achieving 100% Throughput in an Input-Queued Switch," *in Proceedings of ICCCN'96 Conference*, pp. 226-231, October 1996.

[9] Mekkittikul A. and McKeown N. "Scheduling VOQ Switches Under Non-Uniform Traffic," *CSL Technical Report*, CSL-TR-97-747, Stanford University, 1997.

[10] Mekkittikul A., "Scheduling Non-Uniform Traffic in High Speed Packet Switches and Routers," *PhD Thesis*, Stanford University, 1998.

[11] Shanmugam A. and Shanthi G., "A Survey of Fast Packet Switching Techniques," *IETE Technical Review*, vol. 22, no. 2, pp. 129-138, March-April 2005.

Shanmugam Arumugam obtained his PhD degree from Bharathiar University, Coimbatore. He was with PSG College of Technology, Coimbatore from 1979-2004. During his career, he has undertaken several consultancy and sponsored research activities. Currently, he is a principal in Bannari Amman Institute of Technology and Science, India. His major sponsored research activities include modeling, characterization and synthesis of ASICs for fuzzy based traffic polling in broadband networks using ATM mode, and modernization of fiber-optic laboratory both from AICTE, New Delhi, India. He has published more than 75 papers in national and international conferences, including 13 journal publications.

Shanthi Govindaswamy is a lecturer and researcher in Electronics and Communication Department, PSG College of Technology, India. She completed her Bachelor of Engineering degree in electronics and communication from PSG College of Technology in 1988, and Master of Engineering in Applied Electronics from Government College of Technology, Coimbatore, in 1992. She has 13 years of teaching experience and has been with PSG College of Technology since 2001. Currently, she is pursuing her research in the area high-speed packet switches. Her research interests include congestion control in broadband networks and schedulers for high-speed packet switches.

Praveen Kumar Boya received his graduate degree from College of Engineering, Osmania University, India. Currently, he is pursuing post-graduate degree at PSG College of Technology, India. His areas of interest include analog and digital design and mixed VLSI design.