VLSI Support for VOIP: By Using Systolic and Non Systolic Technique Over High Speed Packet Switched ATM Network

Jyothi Noubade *[1], Dr.Rajendra B Konda [2], Prof.Baswaraj [3]
[#] *Research Scholar Department of E&CE, Mewar University, Chittorgarh*
Professor Department of E&CE,Smt,V G Womense College,Gulbarga
Asst. Professor,Department of MCA,CMR Institute of Technology,Bangalore-37
[1]*jyothinoubade12@gmail.com*
[2.] *rbkonda@yahoo.com*
[3]*basawarajnb@gmail.com*

***Abstract*—** this paper presents a methodology for the systematic design of order-preserving multi-input multi-output (MIMO) buffers for large scale or ultra fast packet switches. It focuses on buffer design for fixed-size packets, such as ATM cells, but the design strategy can be extended to variable-length packets. These buffers are capable of inputting and/or outputting multiple packets while maintaining their FIFO (first-in first-out) order, and can be used in ultra high-speed network interfaces and similar applications. Our approach employs a systolic routing network and bank of parallel FIFO buffers to yield a load-balanced MIMO FIFO realization with increased bandwidth [8].

We describe a method for monitoring Voice over IP (VoIP) applications based upon a reduction of the ITU-T's E-Model to transport level, measurable quantities. In the process, 1) we identify the relevant transport level quantities, 2) we discuss the tradeoffs between placing the monitors within the VoIP gateways versus -path monitor requires the definition of a reference de-jitter buffer implementation to estimate voice quality based upon observed transport measurements. Finally, we suggest that more studies are required, which evaluate the placement of the monitors within the transport path, and 3) we identify several areas where further work and consensus within the industry are required. We discover that the relevant transport level quantities are the delay, network packet loss and the decoder's de-jitter buffer packet loss. We find that an inequality of various VoIP codec's in the presence of representative packet loss patterns

Keywords:-ATM, TCP, FIFO, Parallel, MIMO

I. INTRODUCTION

The congestion control is an Internet workhorse. Up to 90% of the traffic is managed by the TCP protocol, used for web browsing, file- sharing, email transmission, and innumerable other applications. The remaining traffic serves applications such as voice over IP, online gaming and Domain Name System (DNS) service. To a first approximation, Internet traffic can be divided into two types: TCP traffic and real-time traffic. The TCP traffic uses window-based congestion control, which adapts each flow's average sending rate to the flow's fair share of available resources. The collection of TCP flows tries to use all available capacity, but it also responds to network congestion, and will reduce each flow's sending rate if the network load increases, or if the capacity in the network is reduced.

The MIMO buffer adaptively manages the available buffer space for statistically multiplexed input traffic. Our systolic routing network provides significant advantages over previously proposed banyan butterfly networks since the systolic network eliminates the need for parallel-prefix adders that compute packet ranks before concentrating them on the output ports [9]. Also, in our designs, load-balancing is done using arithmetic-free circuits.

Using this methodology we derive scalable parallel FIFO buffer structures that can be designed to match the rate of ultra high-speed links using current memory technology that uses moderate clock rates. Our design approach is truly so as to balance the load on all FIFO modules while scalable and has real advantages from a VLSI perspective. As it was described in Chapter 2, if a high-speed buffer is constructed from multiple FIFO buffers then a balanced distribution network (BDN) stage is needed to pack and shift the incoming cells preserving the arrival order of the cells at the same time. Our design approach is based on three-part VLSI architecture as shown in Fig. 3.1. The first part is a systolic two-dimensional array (folded crossbar) of simple switches that performs load-balancing and ensures proper cell ordering.

The second part is a bank of single input single-output FIFO buffers that operate in parallel. The last part is a rotating multiplexer (RMUX) which adapts the buffer output rate to the output link rate. Note that the folded crossbar is used for load balancing in a single MIMO buffer and not for general routing of packets. An example of how MIMO buffers can be used in a multistage switch is shown in Fig. 3.1. Note that a MIMO buffer spreads its cells equally among its output links thus achieving load balancing on the links as well. Each bundle of links connected to the output of a MIMO buffer can be viewed as a "super link" with aggregate bandwidth equal to the total bandwidth of its individual links. The

proper operation of multiple stages of MIMO buffers is possible because of their order preserving capability.

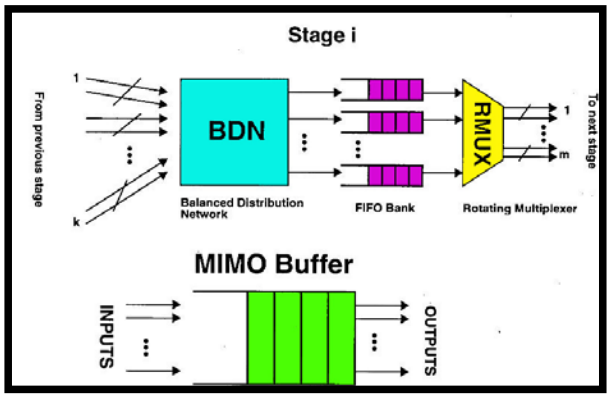

Fig. 1: MIMO buffer in a multi-stage switch and its symbol

II. METHODOLOGY & SYSTEM DESIGN

THE BALANCED DISTRIBUTION NETWORK (BDN)

The BDN is the primary routing engine in the MIMO buffer, performing both packs and shift operations. Additionally, the BDN maintains the state of the last shift operation, which indicates the position of the last FIFO buffer to which a cell has been forwarded. Maintaining this state is essential for the load balancing operation. The 8 x 8 BDN shown in Fig. 2, is comprised of a folded crossbar simple switching elements (SWTs), a horizontal controller (HOC), and a vertical controller (VEC). For FIFO operation, the HOC and VEC are constructed from simple networks of delay elements for the purpose of synchronizing the arrivals of cells and control signals at the appropriate switching elements. In Fig. 3.3, SWTs are represented by squares while delay chains are represented by circles labelled by the amount of delay, e.g. a circle labelled 5D represents a delay chain of length 5, where a unit of delay corresponds to one clock period.

The BDN operates in a synchronous time-slotted manner, with each time-slot equivalent to one cell (fixed-size packet) time over a w-bit link. Cell arrivals are always aligned with the beginning of time-slot for all BDN inputs.

Within a time-slot, a BDN receives either a valid or an invalid (or empty) cell. We assume that a special bit in the cell header is used to indicate whether the cell is valid or not. Such empty cells will be removed from the cell stream during the packing operation in the BDN.

Note that before entering the BDN, the cells are passed through the HOC which routes incoming cells through proper delay stages before they enter the BDN. Fig. 3 explains the timing alignments of the cells at the input of the HOC for the 4 x 4 BDN example of Fig. 3, also at the input of the folded crossbar (i.e. the output of the HOC), and at the output of the crossbar before being written into the FIFO buffers. Note the difference between a clock periods, which defines one time-unit versus a cell period which defines one time-slot. Basically, the incoming cells (whether valid or empty) are always aligned at the HOC inputs. The HOC then "staggers" the cells before entering the crossbar. This is a necessary step because cells in the crossbar can turn-around and rendezvous with one another at a SWT as will be explained later. However, before leaving the crossbar, the cells are aligned back again.

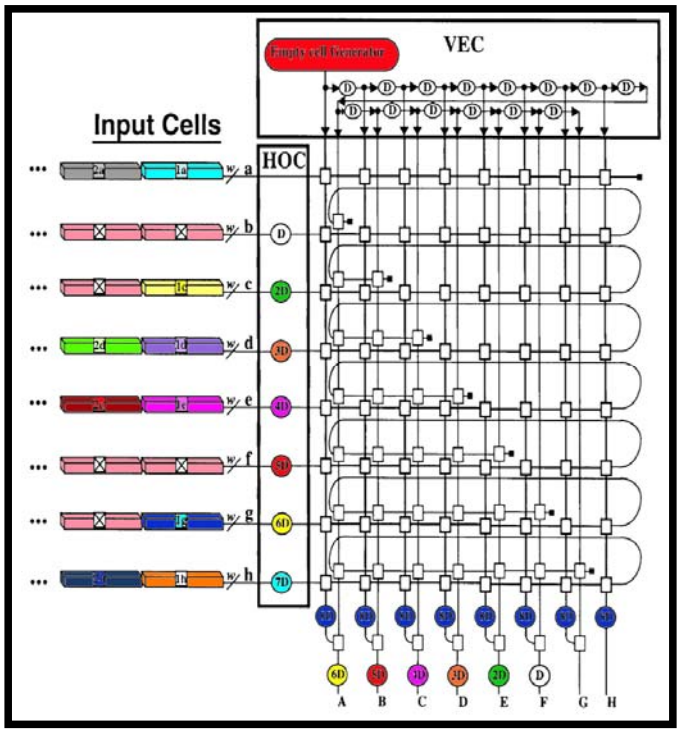

Fig. 2: Internal Structure of an 8x8 BDN

As mentioned above, the folded crossbar network consists of simple 2-input 2-output switching elements. Cells enter a SWT from its north port (i.e. port DN) and/or from its west port (i.e. port Dw)'. Based on its local state and the information available at its input ports, a SWT will steer the incoming cells to the proper output ports; a SWT has a set of locally controllable transmission circuits which enable its four 110 ports to be connected internally in various configurations.

As shown in Fig. 3, each SWT can be in one of two modes simultaneous cell representing its local state: a cross (or unswitched) mode and a toggle (or bar) mode. Normally, the SWT is in the cross mode, i.e. cells from its north (west) port are routed to its south (east) port. Fig. 3 also shows the four possible configurations of a SWT based on the type of arrivals on its two inputs. Note that empty cells are always routed eastwards where eventually they will be discarded (the small solid black squares in Fig. 3 and other figures in this chapter indicate a cell-discard port).

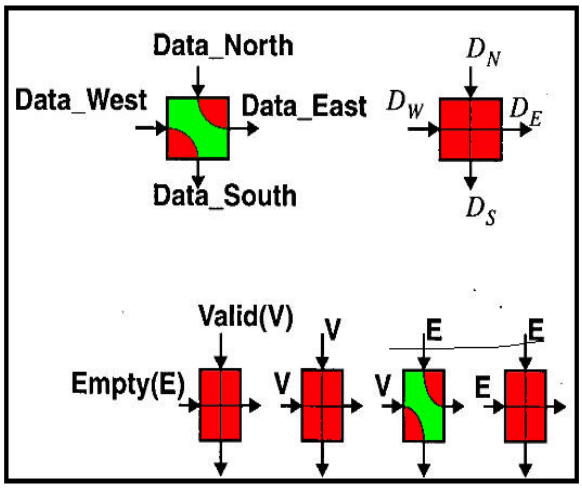

Fig. 3: Two states of the switch element

III. PARALLEL FIFO BUFFERS

We start this section by considering previously proposed techniques for designing parallel multi- input multi-output (MIMO) buffers that preserve the FIFO ordering of cells. It is worth mentioning that such buffers have been also called "shared" FIFO buffers in previous literature and has been used before in several switch designs, for example, see [10]-[20].

As shown in Fig.4, if a high-speed buffer is constructed from multiple FIFO buffers, then a dynamic balanced distribution network (BDN) stage is needed to pack and shift the incoming cells so as to balance the load on all FIFO modules while preserving the arrival order of the cells at the same time. The buffer shown in the figure has 5 input ports and 5 output ports. The five ports are labelled a, b, c, d, and e. The labelled squares in the figure represent cells arriving in three different time slots (1, 2 and 3), and the label of each cell indicates its arrival time-slot and arrival port. For example, the square label (2c) represents a cell arriving on port c in time-slot 2.

Typically load-balancing is done using a multistage network of adders to do packet-ranking, and routing is done in a second pass through another multistage network as was done in the Knockout switch [10], and Multinet switch [11], [12].
Note that the BDN packs incoming cells in adjacent output locations and shifts them appropriately to achieve a balanced loading of the FIFO buffers. Note also the order of cell arrivals is preserved. Fig. 4(b) shows the dynamic configuration of the BDN in each time-slot. In short, the BDN distributes cells arriving in the same time-slot among the FIFO buffers in a cyclic (or round robin) fashion so that the difference between the contents of any two buffers does not exceed one cell. Note, that load balancing eliminates the need for combinational concentrators and restricts the causes of cell loss to buffer overflow only.

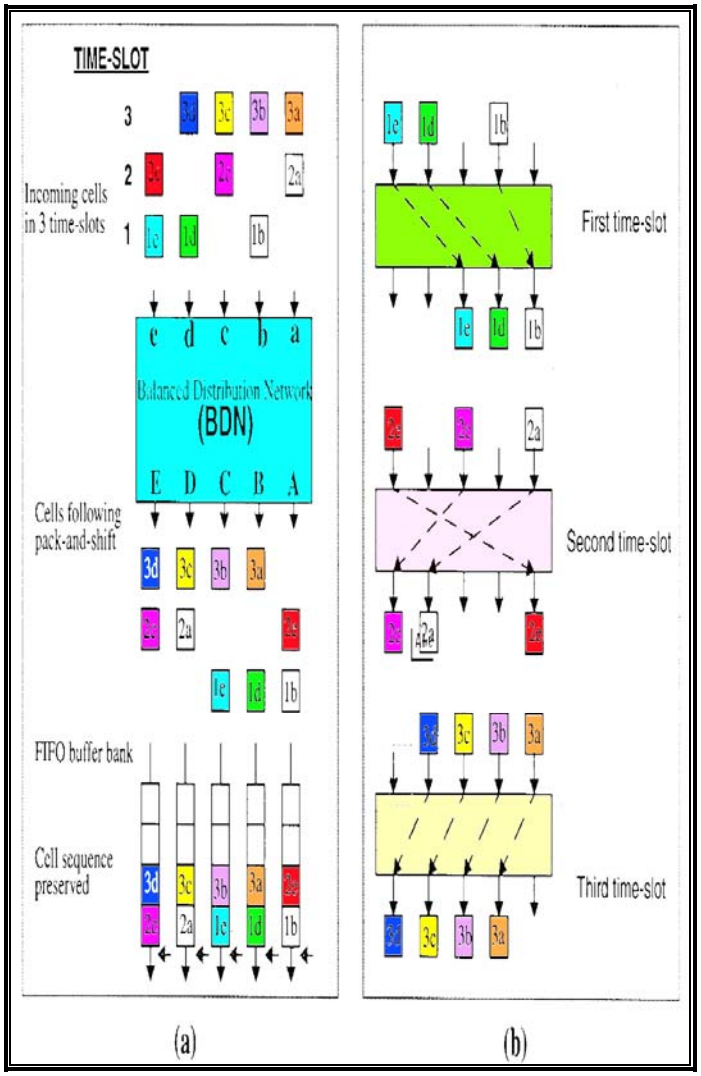

Fig. 4: (a) Operation of a shared-buffer that preserves the cell arrival sequence (b) Illustration of pack-and-shift approach in three time-slots

Several schemes have been proposed to achieve the BDN function using multistage interconnection networks (cf. [25], and the references therein). To perform the pack-and-shift function, cells are first packed using either a concentrator or a routing network then shifted cyclically to distribute the packed cells evenly among the FIFO buffers at the output. Both the pack and shift functions can be implemented by the well-known butterfly, (which is a banyan type) network [25].
The butterfly network nodes must have an arithmetic addition capability to perform parallel prefix addition for ranking the cells arriving at the same time in a given order. Ranking

the cells, can also be done by a low-latency adder network, like the one described in [25].

Fig. 5 shows how cells can be packed and shifted (after they have been ranked) on a butterfly network for a 5 x 5 (5 - input/5 -output or 5110) buffer architecture. Referring to the example in Fig. 5(a), the three cells labelled 1b, 1d, and 1e are first assigned the ranks 0, 1, and 2, respectively, then packed by a butterfly network. These ranks will be used as local routing headers to route these cells to output ports 0, 1, and 2 using a greedy self-routing strategy. At the output of the ranking network of Fig.5 (a), the cells appear labelled by their computed ranks.

Fig. 5(b) illustrates a situation where the cells are ranked first, then packed and shifted cyclically by 3 positions. The shift-offset function modifies the ranks of the cells, so that they can be packed and shifted by the same butterfly network as the one used in Fig. 5(a). The routing procedure utilizes the monotonic routing property of the butterfly to pack-and-shift input cells with no path contention.

It is clear that the above realization of a BDN adds a high degree of complexity and latency to the buffer design. This approach has been employed in the Multinet switch [11], [12]. The internal buffers in the Multinet switch implement a partial buffer sharing (PS) discipline that maintains cells in the correct order [11]. Each PS buffer in the switch is constructed using a reverse banyan network (similar to the butterfly network illustrated in Fig. 5, of appropriate size, that feeds cells to a bank of FIFO buffers. The reverse banyan network is equipped with a Fetch-and-Add capability that contributes to balancing the load in the bank of FIFO buffers constituting the PS buffer space.

Despite claims of simplicity, the Multinet switch is actually complex and cells can experience significant delays, mainly because of the structure of the PS buffers. The Fetch-and-Add reverse banyan network employed in each PS buffer performs arithmetic as well as routing operations. Three passes of computing and routing through the reverse banyan are required before cells are finally fed to the bank of FIFO buffers of PS buffer [12].

In current deep submicron CMOS technologies, the delay in signal propagation over longer wires becomes a significant design limitation, particularly for shared buffer architecture with large number of 110 ports. This limitation is particularly applicable to high wire organizations such as the butterfly network (or reverse banyan network) typically used in interconnection networks [25].

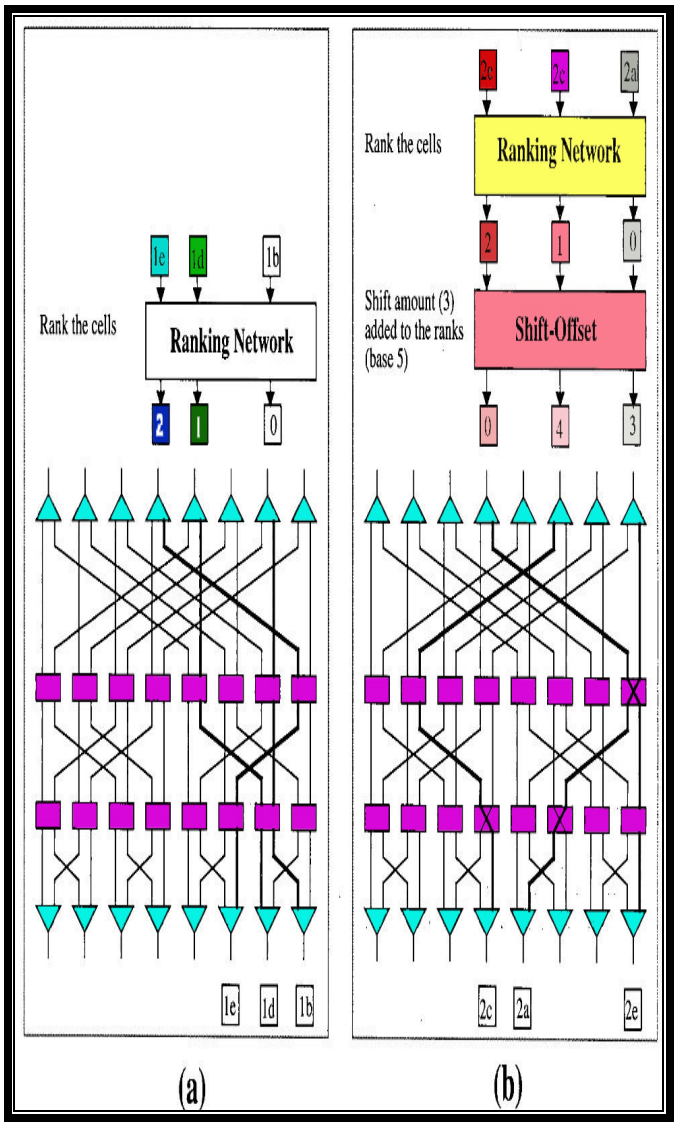

Fig. 5: (a) Packing ranked cells on a butterfly network (5 I/O - first time-slot) (b) Packing and shifting ranked cells on a butterfly network (5 I/O - second time-slot)

A somewhat different approach has been employed by the Knockout switch [10] and many of its realizations (such as the grow able packet switch [13], [14], and the MOB AS switch [16]) to achieve load balancing. The knockout approach employs a lossy deep multi-stage concentration network followed by a shifter network, as shown in Fig. 2.3. The concentration is performed b y N-to-L combinational tournament circuit, in which packets losing contention are knocked-out [10]. The parameter L (for the N-to-L concentrator) is selected in the Knockout switch to achieve a particular cell loss that can be reduced by increasing L. The original Knockout concentrators have depth proportional to N, and their complexity as well as delay can be substantial for large N.

The PINIUM switch [18] is basically another Knockout switch with the concentrators modified to perform priority routing of cells. Each such concentrator is constructed from modified bitonic sorters and mergers [18], [25]. An N-to-L concentrator is constructed iteratively from L-input bitonic sorters and 2L-to-L bionic mergers. As in the Knockout switch, the value parameter L determines the cell loss ratio in the memory less routing and concentrating part of the switch.

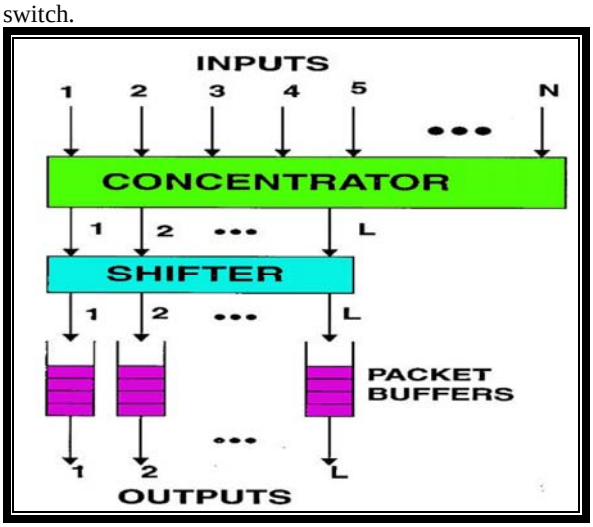

Fig.6: Structure of a shared buffer in the Knockout switch

The main weakness of the switch is the use of sorters in the priority concentrator blocks. The sorters can incur significant routing delays in large-scale versions of the switch. The delay caused by the use of sorting in the priority concentrators becomes unacceptable at high link speeds. It is also not clear whether the use of priority concentration can lead to unfairness.

In the Helical switch [19], which is a self-routing multistage banyan network with output as well as internal buffers, each internal buffer consists of a non-blocking concentrator feeding a bank of FIFO buffers. Each non-blocking concentrator is constructed using a running adder and a two-dilated reverse banyan network of appropriate size. Correct cell sequence, is maintained by the concentrators.

A concentrator routes the incoming cells to the next available concentrator output in round-robin order. At the concentrator output dummy cells are generated and inserted into the proper FIFO to maintain the integrity of cell ordering. The dummy cells generated by one stage of concentrators are not routed by the next stage of concentrators, rather they are used to create idle cycles that keep later cells from advancing past earlier cells. The concentrators employed in each stage of the switch can cause significant cell delays. This is caused by the insertion of dummy cells in internal FIFOs as well as the delays caused by the running adder. In other words, the delays will grow, as the number of stages increase, in large switches and the insertion of dummy cells in internal buffers will aggravate this problem. Additional hidden delays are caused by the running-adder operations in the reverse banyan networks employed in every stage.

One of the main contributions of this research is a very efficient implementation of the pack-and- shift function using a dynamically reconfigurable systolic interconnection array, which is managed by two simple distributed controllers. The systolic architecture essentially realizes a pipelined switch array with embedded load-balancing and order-preserving capabilities. To our knowledge, the work on parallel FIFO implementation has been done mainly under the concept of shared output buffer design in high speed packet switches as reported in [10]-[20]. Our work improves on these techniques by proposing an arithmetic-free, pipelined, load-balancing network that forwards cells to a parallel MIMO FIFO bank [8]. Our design approach is based on three-stage VLSI architecture as shown in Fig.6 [9]. The first stage (BDN) is a systolic array of simple switches that performs load balancing and ensures proper cell ordering. The second stage is a bank of single-input single-output FIFO buffers operating in parallel. We enhance the flexibility of the design by adding a rotating multiplexer (RMUX) at the output, to adapt the buffer rate to that of the output link.

In the system design of the MIMO buffer the following objectives have been pursued :) Maximum buffer space, 2) Minimal routing delay, and 3) Correct order of packet arrival [26]. The detailed structure of the above realization is reported. Here, it is worth mentioning that because of the uniform structure of the proposed MIMO buffer, it has the advantages of simple synchronization and expansion due to the modular structure of the systolic switch array. It is a truly scalable architecture, and amenable to simple VLSI implementation.

As shown in Fig.6, if a rotating multiplexer (RMUX) is applied in the last stage of the parallel buffer, the buffer can also be employed as an ATM access point or as a buffer concentrator [26]. In addition to their use in high speed interfaces, these buffers can be utilized in a number of switch architectures [27].

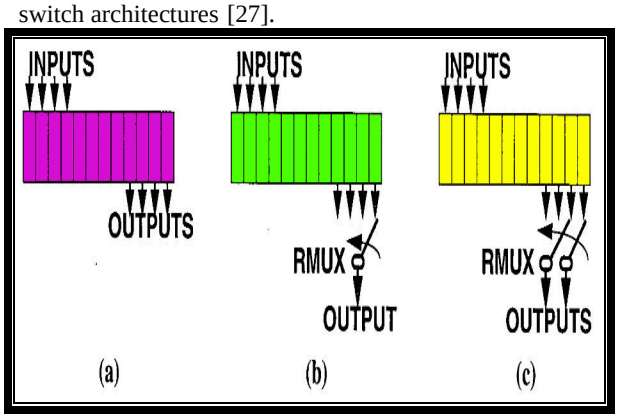

Fig. 7: (a) Parallel buffer, , and (c) Buffer concentrator(b) ATM access point

IV.CONCLUSION

The explosion in data traffic volumes in recent years and the consequent fibre expansion are forcing system designers to rapidly increase packet network capacity. A critical issue in designing of these packet networks is the realization of high speed data buffers. The main thrust of this thesis is the development of two scalable queuing structures in VLSI for very high speed packet-switched networks. The main motivation for this work is the emerging need for scalable architectures for high speed interfaces and packet scheduling systems. However, all architectures and circuits presented in the research can be employed in other applications.

Using this methodology, we derived scalable parallel MIMO FIFO buffer structures that can be designed to match the rate of ultra high-speed links using current memory technology that uses moderate clock rates. Using a 0.5-Jlm CMOS technology, a small prototype of the MIMO buffer was designed which attains a rate of 10.6 Gb/s which is more than adequate to support a Sonet OC-192 link [9]. The combined use of pipelined architecture and dynamic CMOS circuits resulted in significant reduction in design complexity and substantial performance gains in speed and silicon area.

Although the idea of using parallel arrangement of FIFO buffers to realize a faster shared FIFO buffer has been around for many years, our systolic routing network provides significant advantages over previously proposed banyan butterfly networks, since the systolic network eliminates the need for parallel-prefix adders that compute packet ranks before concentrating them on the output ports.

V. **FUTURE DIRECTIONS**

Recently, there has been much interest in using more complex, active queue management algorithms [73] such as multiclass RED (random early detection) [65]-[74] and WFQ (weighted fair queuing) [44]-[67]. To use the proposed PPQ for more sophisticated scheduling algorithms of these queue managements, future improvements are necessary to provide some new features to the architectures. Moreover, the PPQ architectures presented and consist of several identical pipeline stages, where each pipeline stage of the PPQ includes mainly some registers and a parallel sorter as its own processing element. One of the immediate improvements is to apply more advanced processing elements instead of parallel sorters for other promising applications [62]. For example, the processing elements can accumulate certain values while sorting packet headers.

It has been reported that although the global clocking schemes simplifies hardware design, the self-timed approach has definite advantages [63], [64]. Systolic architectures are typically designed under the strict synchronous model of computation. However, for our PPQ architecture it is possible to redesign the sorters as asynchronous processing elements, whereby each computes its output values only when all its inputs are available resulting in a self-timed design [62]. The asynchronous design approach eliminates the need for a global clock and circumvents the problems arising from clock skew. Nevertheless, our systolic PPQ designs are heavily pipelined and retimed; critical paths are properly balanced and little room is left to obtain an asynchronous benefit. Moreover, an asynchronous benefit of this kind must be balanced against a possible overhead in completion signalling and asynchronous control [64]. A self-timed design also has the potential for reduced power consumption relative to its synchronous counterpart.

A synchronous circuit is either *quiescent* (i.e., the clock is turned off) or active entirely (i.e., clock on). An asynchronous circuit, by contrast, consumes energy only when is active. In other words, self-timed circuits can potentially achieve low power consumption because unused circuit parts can automatically turn into stand-by mode. However, it is not obvious to what extent this advantage is fundamentally asynchronous. Synchronous techniques such as *clock gating* may achieve similar benefits, although they have their limitations.

In general, the potential of asynchronous design for low-power depends on the application [64]. Additional savings in power can be accomplished by using self-timed circuits with a mechanism that adaptively adjusts the supply voltage to the smallest possible while maintaining the performance requirements [65]. An important consideration that may limit the applicability of this technique is that the supply voltage must vary at a slow rate relative to the internal operational speed of the circuit otherwise it may interfere with the operation of the circuit. In summary, the issue of clocking in our designs and possible self-timed versions of the PPQ architectures need further investigation [62].

VI.REFRENCES

[1] S. E. Butner, and R. Chivukula, "On the Limits of Electronic ATM Switching," IEEE Network, vol. 10, no. 6, pp. 26-31, Nov.lDec. 1996.

[2] S. E. Butner, and D. A. Skirmont, "Architecture and Design of a 40 Gigabit per second ATM Switch," in Proc. Int'l. Conf. Compo Design, Austin, TX, pp. 352-357, Oct. 1995.

[3] Richard Crisp, "Direct Rambus Technology: The New Main Memory Standard," IEEE Micro, vol. 17, no. 6, pp. 18-28, Nov.lDec. 1997.

[4] A. Demers, S. Keshav, and S. Shenker, "Design and Analysis of a Fair Queuing Algorithm," ACM Sigcomm '89, Austin, Sep. 1989.

[5] A. K. Parekh, "Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks," Technical Report LIDS-TH-2089, MIT, Cambridge, MA 02139, Feb.1992.

[6]D. Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network:

Architecture and Mechanism", ACM Sigcomm '92, pp. 14-26, Aug. 1992.
[7] S. Floyd, and V. Jacobson, "Link-sharing and Resource Management Models for Packet Networks," *IEEE/ACM* Trans. on Networking, vol. 3 no. 4, pp. 365-386, Aug. 1995.
[8] M. Kazemi-Nia, and H. Alnuweiri, "VLSI Design of Parallel FIFO Buffers for Gigabit Packet Networks," accepted for publication in the IEEE Trans. on VLSI Systems.
[9] M. Kazemi-Nia, and H. Alnuweiri, "Parallel Buffer Design for High Speed Interfaces," in proc. CCBR '99, pp. 102-113, Ottawa, Nov. 1999.
[10] Y. S. Yeh, M. G. Hluchyj, and A. S. Acampora, "The knockout switch: A simple architecture for high-performance packet switching," IEEE J. on Select. Areas in Com- mun., vol. 5, no. 8, pp. 1274-1283, Oct. 1987.
[11] H. S. Kim, "Design and Performance of Multinet Switch: A Multistage ATM Switch Architecture with Partially Shared Buffers," *IEEEI* ACM Trans. on Networking, vol. 2, no.6, pp. 571-580, Dec. 1994.
[12] H. S. Kim, "Multinet Switch: Multistage ATM Switch Architecture with Partially Shared Buffers," Infocom '93, pp. 4c.3.1 - 4c.3.8, 1993.
[13] K. Eng, M. J. Karol, and Y. Yeh, "A Growable Packet (ATM) switch Architecture Design Principles and Applications," IEEE Trans. on Communications, vol. 40, no. 2, pp. 423-430, Feb. 1992.
[14] K. Y. Eng, M. J. Karol, and Y. Yeh, "A High-Performance Prototype 2.5 *Gb/s* ATM Switch For Broadband Applications," Globecom' 92, pp. 111-117, 1992.
[15] K. Y. Eng, M. J. Karol, and Y. Yeh, "Concentrator-based growable packet switch," U. S.Patent 5-256-958, Oct. 26, 1993.
[16] H.J Chao, and B. Choe, "Design and Analysis of a Large-Scale Multicast Output Buffered ATM Switch," IEEE/ACM Trans. on Networking, vol. 3, no. 2, pp. 126-138, April 1995.
[17] H. J. Chao, "A Recursive Modular Terabit/Second ATM Switch," IEEE J. on Selec. Areas in Commun., vol. 9, no, 8, pp. 1161-1172, Oct. 1991.
[18] K. L. E. Law, and A. Leon-Garcia, "A Large calable ATM Multicast Switch," IEEE J. On Selec. Areas in Commun., vol. 15, no. 5, pp. 844-854 June 1997.
[19]I. Widjaja, and A. Leon-Garcia, "The Helical Switch: A Multipath ATM Switch Which Preserves Cell Sequence," IEEE Trans. on Communications, vol. 42, no. 8, pp. 2618-2629, Aug. 1994.
[20] A. Pattavina, "Nonblocking Architectures for ATM Switching," IEEE communicationMagazine, vol. 31, no. 2, pp. 38-48, Feb. 1993.
[21] M. Kazemi-Nia and H. Alnuweiri, "A Parallel Priority Queue (PPQ) with Rate Adaptation for High Speed Networks," in proc. CCBR '99, pp. 90-101, Ottawa, Nov.1999.
[22]M. Kazemi-Nia, and H. Alnuweiri, "A Systolic Parallel Priority Queue (PPQ) with Output Rate-Control for High Speed Networks," submitted for publication in the IEEE Trans. on VLSI Systems, June 1999.
[23] H. J. Chao, and N. Uzun, "A VLSI Sequencer Chip for ATM Traffic Shaper and Queue Manager," IEEE Journal of Solid-State Circuits, vol. 27, no, 11, pp. 1634-1643, Nov.1992.
[24]S.-W. Moon, J. Rexford, and K. Shin, "Scalable hardware priority queue architectures for high-speed packet switches," IEEE Real-Time Tech. and Applic. Symp., pp. 203-212, June 1997.
[25]F. T.Leighton,Introductions to Parallel Algorithms and Architectures: Arrays - Trees - Hybercubes, San Mateo, CA: Morgan Kaufmann, 1992.
[26]M.Kazemi-Nia,and H.Alnuweiri, "Balanced Multiport Buffer Design in Silicon," IEEE ATM '96 Workshop, pp. HW.I-6, San Francisco, CA, Aug. 1996.
[27] M. Kazemi-Nia, and H. Alnuweiri, "VLSI Implementation of Highly-Optimized Scalable ATM Switches," 18th Symp. on Communications, pp. 101-104, Kingston, Canada, June 1996.
[28] H.T.Kung,"Why systolic architectures,"Computer, vol. 15, no. 1, pp. 37-46, Jan. 1982.
[29]C.E. Leiserson, Area-Efficient VLSI Computation, Ph.D. Dissertation, Dept. Computer Science, Carnegie-Mellon University, 1981; published in a book form, Cambridge, Massachusetts: MIT Press, 1983.
[30]C. E. Leiserson, "Systolic Priority Queues," Caltech. Conf. on VLSI, pp. 200-214, Jan.1979.
[31] M. Hashemi and A. Leon-Garcia, "A General Purpose Cell Sequencer/Scheduler for ATM Switches", IEEE Infocom '97, Kobe, Japan, April 1997.[32] H.J Chao, "A Novel Architecture for Queue Management in the ATM Networks," IEEE J. on Selec. Areas in Commun., vol. 9, no. 7, pp. 1110-1118, Sep. 1991.
[33] H.J Chao, and N. Uzun, "A VLSI Sequencer Chip for ATM Traffic Shaper and Queue Manager," in Proc. Globecom '92, pp. 1276-1281, 1992.
[34] H. J. Chao, and N. Uzun, "A VLSI Sequencer Chip for ATM Traffic Shaper and Queue Manager," IEEE Journal of Solid-State Circuits, vol. 27, no, 11, pp. 1634-1643, Nov.1992.
[35] H.J Chao, and N. Uzun, "An ATM Queue Manager Handling Multiple Delay and Loss Priorities," IEEE/*ACM* Trans. on Networking, vol. 3, no. 6, pp. 652-659, Dec. 1995.
[36] G.c. Heinze, R. palmer, I. dresser, and N. Leister, "A three chip set for ATM switching," in IEEE Custom Integr. Circ. Conf., pp. 14.3.1-14.3.4, May 1992.
[37] D. Picker, and R.D. Fellman, "A VLSI Priority Packet Queue with Inheritance and Overwrite," IEEE Trans. on VLSI systems, vol. 3, no. 2, pp. 245-253, June 1995.
[38] N. Weste, and K. Eshraghian, Principles of CMOS VLSI Design - a Systems Perspective, Addison-Wesley, 2 edition, 1993.
[39]J.A Pretorius, AS. Shubat, and C.A Salama, "Charge redistribution and noise margins in domino CMOS logic,"

IEEE Trans. Circ. Syst., vol. 33, no. 8, pp.786-793, Aug. 1986.
[40] M. Shoji, "FET Scaling in Domino CMOS Gates," IEEE J. of Solid-State Circuits, vol. sc- 20, no. 5, pp. 1067-1071, Oct. 1985.
[41] J. D. Ullman, Computational Aspects of VLSI, Rockville, MD: Computer science press,1984.
[42]H.M. Alnuweiri, and R. Beraldi, "A General Class of Highly-Optimized Scalable ATM Switches," Globecom '95, Singapore, Nov. 1995.
[43] . W. Marcus, "A CMOS Batcher and Banyan Chip Set for B-ISDN Packet Switching," IEEE J. of Solid-State Circuits, vol. 25, no. 6, pp. 1426-1432, Dec. 1990.
[44]M. D. Marco, "Distributed Routing Protocols for ATM Extended Banyan Networks," IEEE j. on Selec. Areas in Commun., vol. 15, no. 5, June 1997.
[45]P. C. Wong, and M. S. Yeung, "Design and Analysis of a Novel Fast Packet Switct Pipeline Banyan, *IEEE/ACM* Trans. on Networking, vol. 3, no. 1, pp. 63-69, Feb. 1995.
[46] F. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks," Proc. of the IEEE, vol. 78, no. 1, pp. 133-167, Jan. 1990.
[47]J. S. Turner, "Design of a Broadcast Packet Switching Network," IEEE Trans. Commun., vol. 36, no. 6, pp. 734-743, June 1988.
[48]M. Alimuddin, H. M. Alnuweiri, and R. W. Donaldson, "The Fat-Banyan ATM Switch," IEEE Infocom '95, pp. 659-666, April 1995.
[49] E. T. Bushnell, and J. S. Meditch, "Dilated Multistage Interconnection Networks for Fast Packet Switching,"IEEE Infocom '91, pp. 1264-1273, 1991.
[50] C. P. Kruskal, and M. Snir, " The performance of Multistage Interconnection Networks for Multiprocessors," IEEE Trans. on Computers, vol. c-32, no. 12, pp. 1091-1098, Dec.1983.
[51] T. Szymanski, and S. Shaikh, "Markov Chain Analysis of Packet-Switched Banyans with Arbitrary Switch Sizes, Queue Sizes, Link Multiplicities, and Speedups," IEEE Infocom '89,pp.960-971,1989.
[52] Y. Mun, and H. Y. Youn, "'Performance Analysis of Finite Buffered Multistage Interconnection Networks," IEEE Trans. on Computers, vol. 43, no. 2, pp. 153-162, Feb. 1994.
[53] H. Yoon, K. Y. Lee, and M. T. Liu, "Performance Analysis of Multibuffered Packet-Switching Networks in Multiprocessor Systems," IEEE Trans. on Computers, vol.39, no. 3, pp. 319-327, March 1990.
[54] C. M. Chu, H. Tayar, and H. M. Alnuweiri, "Enhanced Packet Switching on a Dilated Banyan Switch with Back Pressure," in Proc. ISCA '99, pp. 130-134, Mexico, 1999.
[55] IEEE Standard VHDL Language Reference Manual, std 1076-1993, New York: IEEE,1993.
[56] C. D. Thompson, "Area-time complexity for VLSI," Proc. ACM Symp. Theory Computation, May 1979.
[57]D. E. Knuth, The Art of Computer Programming. Vol. III: Sorting and searching, Reading, MA: Addison-Wesley, 1973.
[58]H. M. Alnuweiri, "A New Class of Optimal Bounded Degree VLSI Sorting Networks", IEEE Trans. on Computers, vol. 42, no. 6, pp. 746-752, June 1993.
[59]T. H. Cormen, C. E. Leiserson, and R. L. Rivest, Introductions to Algorithms, Cambridge, MA: The MIT Press, 1990.
[60] C. D. Thompson, "The VLSI complexity of sorting," IEEE Trans. on Computers, vol. C-32,no. 12,pp. 1171-1184, Dec. 1983.
[61]D. E. Knuth, The Art of Computer Programming. Vol. In: Sorting and searching, Reading, MA: Addison-Wesley, 1973.
[62] M. Kazemi-Nia, and H. Alnuweiri, lOASelf-timed Parallel Priority Queue (PPQ) with Output Rate-Control," in preparation.
[63] S. Hauck, "Asynchronous Design Methodologies: An Overview," Proc. of the IEEE, vol.83,no. 1,pp.69-93,Jan. 1995.
[64] C. H. Van Berkel, M. B. Josephs, and S. Nowick, "Applications of Asynchronous Circuits," Proc. of IEEE, vol. 87, no. 2, pp. 223-233, Feb. 1999.
[65] Nielsen, C. Niessen, J. Sparso, and K.V. Berkel, " Low-power operation using self- timed circuits and adaptive scaling of the supply voltage," IEEE Trans. on VLSI Systems, vol. 2, no. 4, pp. 391-397, Dec. 1994.
[66] Craig Partridge, Gigabit networking, Addison-Wesley Pub., 1991.
[67] T. Koinuma, and N. Miyaho, "ATM in B-ISDN Communication Systems and VLSI Realization." IEEE J. of Solid-State Circuits, vol.30, no. 4, pp. 341-347, April 1995.
[68]M. Hlucchyj, and M. Karol, "Queueing in high-performance packet switching," IEEE J. on Select. Areas in Commun. vol. 6, no. 9, pp. 1587-1597, Dec. 1988.
[69] H. Kuwahara et al., "A shared buffer memory switch for an ATM exchange," in Proc. ICC '89, Boston, MA, June 1989, pp. 118-122.
[70] T. Kozaki, N. Endo, Y. Sakurai, O. Matsubara, M. Mizukami, and K. Asano, "32 x 32 Shared Buffer Type ATM Switch VLSI's for B-ISDN's," IEEE J. on Se1ec. Areas in Commun., vol. 9, no. 8, pp. 1239-1247, Oct. 1991.
[71] M. Katevenis, P. Vatsolaki, and A. Efthymiou, "Piplelined Memory Shared Buffer for VLSI Switches," Proc. of Sigcomm '95, Cambridge, MA, pp. 39- 48, 1995.
[72] Notani et al., "An 8x8 ATM Switch LSI with Shared Multi-buffer Architecture," Proc. Of Sym. on VLSI Circuits Digest of Technical Papers, pp. 74-75, 1992.
[73] C. Zukowski, and T.B. Pei, "VLSI implementation of ATM buffer management," Proc. Of IEEE Int. Conference on Communications, pp. 716-720, 1991.
[74] M. Katevenis, P. Vatsolaki, and A. Efthymiou, "Pip1elined Memory Shared Buffer for VLSI Switches," Proc. of Sigcomm '95, Cambridge, MA, pp. 39- 48, 1995.