# Implementation and Evaluation of Large Interconnection Routers for Future Many-Core Networks on Chip

Amir H. M. Zaytoun, Hossam A. H. Fahmy and Khaled M. F. Elsayed
Electronics and Communications Engineering
Cairo University
12613 Egypt
eng_amirhasan@ yahoo.com, hfahmy@alumni.stanford.edu, khaled@ieee.org

*Abstract*— **As the number of processing elements in the future Networks on Chip (NoC) increases from multi-cores to many-cores, the role of the interconnection communications becomes more critical. The number of cores on a System on Chip (SoC) will reach thousands in the near future as predicted by the International Technology Roadmap for Semiconductors (ITRS). Currently, NoC interconnections are mostly implemented with m×n 2-D mesh topology connecting small size routers. This will represent the bottleneck to the communication latency for the increasing number of cores where the average number of hops the data have to pass will increase. In this paper, we propose an alternative NoC interconnecting scheme by using large routers interconnecting large number of cores in star topology. This interconnection scheme can be scaled up by using hierarchical-star or fat-tree topologies. We present the implementation and performance evaluation of three large router architectures and compare their efficiency to the small 5×5 router used in the mesh topology. We develop a simulating environment that resembles the real NoC conditions to test the routers throughput and average latency on different buffer sizes and under different traffic loads. We also synthesize them to estimate the area and power consumption. Then, routers efficiencies are calculated with respect to the area and power consumption.**

***Keywords- System-on-chip (SoC); Networks-on-chip (NoC); Crossbar; Batcher-Banyan.***

## I. INTRODUCTION

The increased demand on electronic devices led to the need of compact products with larger and more complex integrated circuits. This challenged integrated circuit designers to shrink the size of transistors on silicon to make room for implementing hundreds of functions on a single chip. As the present transistor geometry keeps shrinking, the relation between gate delay and wiring delay became different [1]. Gate delay decreased due to the decrease of parasitic capacitance of transistors because of the scaled down dimensions. On the other hand, the decrease of wire cross sectional area increased its resistance which decreased the data integrity over long wires and that required inserting amplifiers, hence increased its delay. So, interconnection delay became dominant over gate delay. For a design of "2cm × 2cm" chip, a bus connecting system modules comprised of a long diagonal global wire will cause a delay of 100 ps, which is a full period of a 10 GHz clock [2]. So, the interconnection between modules would become a bottleneck in chip design as clock speeds cannot be higher.

The increase of the wire resistance and delay prevented the ability to increase the operating frequency because of the increased power consumption and heat dissipation. The solution was increasing the number of processing elements operating on lower frequency on one chip. Oracle SPARC T4 is an example for the change in the design of general purpose processors where it is comprised of 8 cores at 2.58 GHz and 3.0 GHz [3]. Ambric Am2045 is another example for the change of the Digital Signal Processors (DSP) which is 336 cores of 32-bit RISC-DSP processors and 336 of 2-kB memories, which run at up to 350 MHz [4]. NVIDIA Tesla C2075 is an example of a Graphic Processor contains 448 cores on one chip [5].

Combining more than one function unit on one chip is called SoC (System on Chip). SoC is simply a chip which performs the functions of a complete system because it is a collection of general purpose processors, DSP, Graphics processors, memories, I/O, mixed signal modules, application specific hardware, Intellectual Properties (IP) cores, peripherals, etc. SoC generally shows better performance than conventional designs characterized by higher processing capability, smaller size, low power, and lower cost.

Typically, shared bus is the conventional connection scheme for on chip communication. In shared bus, all modules are connected to one global bus. The long bus requires inserting repeaters and amplifiers to compensate the degradation in signal integrity (correctness of data transmission), that increased the power consumption and delay. There is also a centralized arbitration unit which organizes the communication between the modules. As the number of connected modules increases, the complexity of the arbitration unit increases and hence transmission latency increases. Shared bus is able to serve a limited number of modules because of long global wires and arbitration problems. On the other hand, it showed a prolonged time to market (TTM) in case of scaling up the design by adding more modules to the pre-designed system where designers had to redesign and test the whole bus arbitration, connections and communication from scratch.

Significant research efforts were directed towards developing new interconnection schemes to solve the

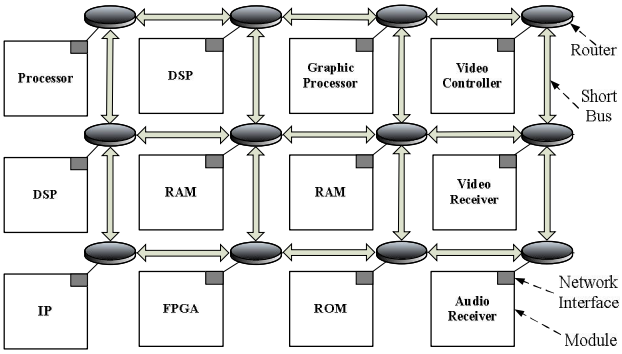


Figure 1. Twelve tiles mesh NoC.

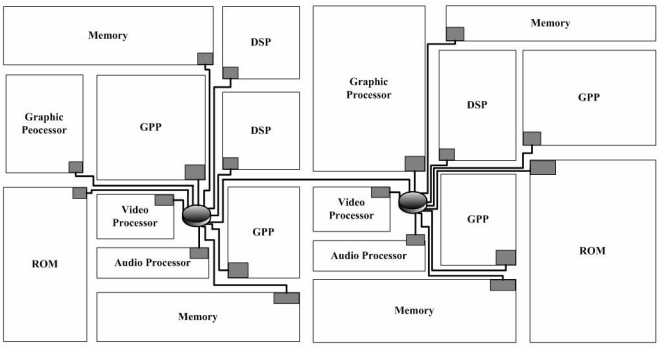


Figure 2. Part of a hierarchical-star topology NoC.

aforementioned problems of shared bus. Proposals were made to connect the modules of SoC in a network with an architecture resembling that of packet-based computer networks and called it Networks-on-Chip (NoC). In NoC, modules are connected by routers and small buses between routers in a network. The presence of routers eliminated the need of complex centralized arbitration unit where data is put in a form of packets which include its destination address and any useful information for routing and ordering. Fig. 1 shows an example of twelve tiles in 3×4 2-D mesh NoC. Each tile contains one module. Each module is connected to its near router through a network interface. Each router has 5×5 input/output ports, four ports connecting other neighbor routers in the mesh and the fifth port connects its dedicated tile module.

Modules can be formed in any network topology. Choosing the suitable topology depends on the nature of the modules. Network topologies determine the number of hops and the wire length involved in each data transmission, both critically influencing the energy cost per transmission [6]. For homogeneous or similar size module system, mesh and similar topologies are suitable. Mesh topology suffers from the accumulated delay of message delivery for far modules especially when the number of modules increases where the average number of hops the message has to cross to reach its destination increases. Each hop shares an added delay and power consumption to the communication. Otherwise, systems with heterogeneous modules which are the dominant feature of future SoC designs do not fit in uniform shaped topologies like mesh. So, other topologies like star will suit this nature. Hierarchical-star and fat-tree topologies are the best candidate for scaling up the star topology. Using hierarchical-star and fat-tree topologies with larger routers reduces the number of hops, hence reduces the latency and power consumption. Fig. 2 shows a part of hierarchical-star topology employing large routers.

In NoC, scalability and reusability are more feasible than in shared buses. Scalability is the ability to increase the number of modules easily by just inserting a router and a network interface and connect to the existing network without redesign from scratch. Reusability means design one and use many times even in the same project.

Routers are the basic backbone of the NoC interconnection communication. NoC routers must be simple in construction and fast in operation to meet the on-chip interconnection requirement of low latency and high throughput. Designers also must target the power and area constraints. Buffers are the most area [7] and power consuming elements in NoC routers. They consume 64% of the total router leakage power [8]. The power consumption of the internal wires dominates the overall power consumption in case of the large routers (beyond 32×32) [9].

In this work, we propose large router architectures targeting the increasing number of many-core NoC employed in star, hierarchical-star and fat-tree network topologies. The main concern of this work is to evaluate various router architectures upon their throughput, area, and power consumption. The rest of this paper is organized as follows. Section II sheds light on previous related designs. Section III provides overview of the basic building blocks in our design such as switching cores, and arbitration units. Section IV discusses the proposed router architectures. In section V, we discuss the simulation and results. Section VI provides the conclusion and future work.

## II. PREVIOUS WORK

Most related previous work was focused on developing the mesh or torus network architectures employing small crossbar routers with simple x-y routing algorithm. A simple 4×4 tiles NoC architecture was proposed in [7], where each tile contains a module and router in a mesh topology, the network presents a simple reliable datagram interface to each tile, the packets consist of 256 bit data field.

The power consumption of four types of switch fabric architectures was estimated in [9] by tracing the dynamic power consumption with bit-level accuracy, which is the power consumed by a bit traveling from input to output ports inside the router. The influence of technology, load capacity, and number of ports on power consumption is also studied.

Passas et al. studied the implementation of a crossbar switching fabric connecting 128 tiles [10]. Area and power of the crossbar switching fabric and the proposed 128 tiles are deeply studied till the placing and routing using 90nm CMOS standard-cells. The crossbar is 32 bit width designed to deliver a 32 bit packet in one system clock (no serialization). Area cost of the 128×128 switching fabric (control not included) was 6% of the total tile area.

## III. BASIC BUILDING BLOCKS

### A. Switching Core

The switching core is a network of connection switches that can be closed to route the input data from the input to its destination in the output. In this work, we are concerned with the Crossbar and Batcher-Banyan switching cores.

*1) Crossbar Switch Architecture*

In N×N Crossbar switch, the N inputs can be connected to the N output via $N^2$ switching nodes called crosspoints. The crosspoint is simply an ON/OFF switch that can be opened or closed by an external control. The Crossbar switch is interconnection contention free because every input/output connection has its dedicated path. The Crossbar is the most relevant for small switches. It is overly expensive for sizes above 32 or 64 [9] whereas the number of crosspoints grows exponentially ($N^2$).

*2) Banyan and Batcher-Banyan Switch architectures*

Banyan switch is one of the multi-stage families of switching fabric architectures [11]. The basic component of the Banyan switch is the switching node. The switching node is a complete self-routing 2×2 switch where it is able to route the input packet from any input port to any output port without any intervention from other control unit. Any larger switch is comprised of a collection of these small 2×2 nodes connected together in a multi stage interconnection network. The multi stage network is comprised of $\log_2 N$ stages, where N is the number of input/output ports and each stage contains N/2 nodes.

Due to the fact this switch architecture utilizes the least switch elements among all other switching core architectures, some sections of each input-output path inside the switching fabric may be shared between other paths. Hence, it suffers from the problem of internal blocking where an internal resource will be shared by two packets that causes a collision. The internal blocking can be avoided if the following three conditions are met:

1. There are no two input packets with the same destination address.
2. Input packets must be arranged in ascending or descending order according to their destination address.
3. There must not be any idle input port between any two active input ports (no gaps between arranged packets).

The first condition can be fulfilled by using an appropriate arbitration unit but the second and third conditions can be fulfilled by using a sorting network that is able to arrange the input packets in ascending or descending order.

K. E. Batcher proposed a sorting network for the Banyan switch fabric [12]. The function of the sorting network is to concentrate the entering packets to the upper or lower ports of the Banyan interconnection network and arrange them in descending or ascending order according to their destination address. Batcher sorting network consists of a collection of sorting nodes connected in stages with different sizes. The sorting node is comprised of a self-routing 2×2 sorting element. Batcher sorting network consists of $\log_2 N$ stages, where N is the number of input/output ports.

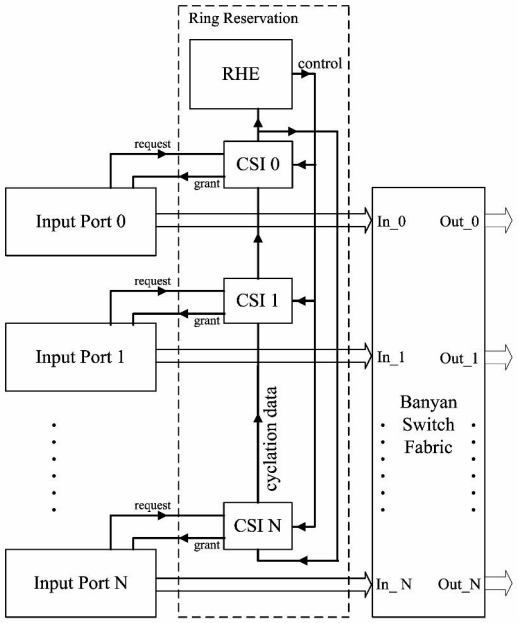


Figure 3. The Ring Reservation unit with respect to the banyan Switch.

### B. Arbitration Units

Arbitration unit is usually used to resolve the destination contention between input packets. Choosing an arbitration unit depends on the buffering strategy used in the input unit. The following are two examples of arbitration units used in this work.

*1) Ring Reservation*

Bingham *et al.* presented an arbitration unit for the First-In-First-Out (FIFO) input buffered Batcher-Banyan switches called Ring-Reservation arbitration unit [13]. The Ring-Reservation keeps track of repeated requests (requests with repeated destination address) during scanning each request from the input units, and then issues the grants to a group of none repeated requests. The Ring-Reservation consists of one main control unit called Ring Head End (RHE) and many small controllers called Cell Switch Interfaces (CSI) where each CSI is dedicated to one input unit. Every CSI compares the request address from its dedicated input unit with all available addresses of each output port in circulation manner and reserves it if the matched destination address has not been reserved for another input before in the arbitration cycle. Fig. 3 shows the Ring Reservation unit with respect to the input units and the Banyan switch.

*2) The Diagonal Propagation Arbiter*

Hurt et al. presented the Diagonal Propagation Arbiter (DPA) in [14]. DPA is an arbiter unit for Virtual-Output-Queuing (VOQ) input buffering strategy that provides a simple and fast arbitration algorithm that can be easily implemented in hardware. The most interesting advantage of this algorithm is that it can be implemented using combinational logic (conventional standard cells without the need of flip-flops).

DPA is based on a two-dimensional ripple carry arbiter architecture proposed in [15]. For VOQ input buffered router, there are N ports and there are N virtual output

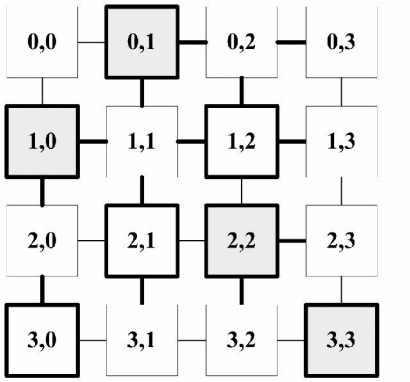

Figure 4. 4×4 two-dimensional ripple carry arbiter.

queues for each port. Each queue issues a separate request and waits a separate grant. So, the total number of requests is $N^2$ and the total number of grants is $N^2$. The two-dimensional ripple carry arbiter consists of $N^2$ arbitration cells; each cell is responsible of receiving request and issuing grant from its corresponding virtual output queue. Fig. 4 shows an arrangement of 4×4 two-dimensional ripple carry arbiter. Each square represent an arbitration cell. The two numbers separated by a comma inside each cell (*i,j*) corresponding to the associated input port (same row) and the virtual queue (same column) respectively.

Each cell receives a request $R(i,j)$ and issues a grant $G(i,j)$ to its corresponding virtual queue $j$ in the input port $i$. At every arbitration cycle, at most only one virtual queue in each input port must be granted and at most only one packet must be granted to one output port. So, if cell $(i,j)$ issued a grant, it prevents any lower priority cell in column $j$ (cells below row $i$) and any lower priority cell in row $i$ (cells right of column $j$) from issuing a grant. The arbitration process begins at cell $(0,0)$ which is the highest priority cell and moves diagonally from the top-left to the bottom-right corner of the arbiter. In Fig. 4, bolded squares indicates that the cell is requested ($R(i,j)=1$), and shaded cells indicate that the cells issued a grant ($G(i,j) = 1$).

The drawback of this architecture is that the higher priority cell always takes the top left cell then any cells in the lower diagonals take lower priority. To ensure fairness, any priority rotation must be added. So, Hurt et al [14] proposed two amendments using round-robin scheme to rotate the priorities.

The first architecture is called the RPA (Rectilinear Propagation Arbiter) and consists of replicating the basic two-dimensional ripple carry arbiter design in both the horizontal and the vertical directions. Two priority vectors are used to activate cells inside a window sliding in both horizontal and vertical directions. The arbitration cell must be modified by gating the request entering to the cell by both the priority vectors.

The second architecture is the DPA. It consists of putting independent cells in one diagonal and replicating the basic cells only in the vertical directions and using only one priority vector to activate cells inside a window sliding in the vertical direction. The arbitration cell must be modified by gating the request entering to the cell by the priority vector. DPA shows better delay due to using lower number of cells than the RPA and hence lower area on silicon. Fig. 5 shows an example of a DPA for 4×4 VOQ buffered router, the bolded lined region is the sliding window.

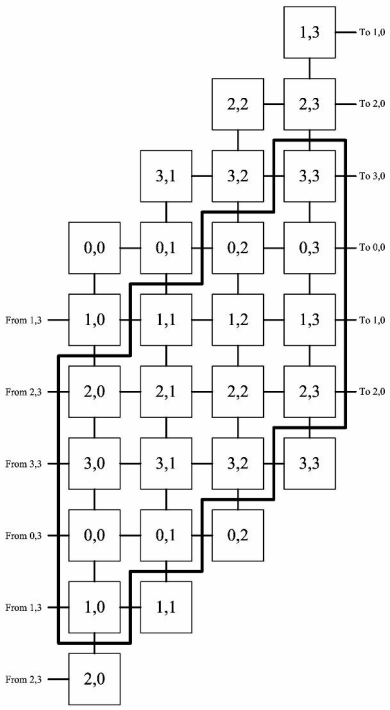

Figure 5. DPA for 4×4 VOQ.

## IV. The Proposed Architecture

In this paper, we propose interconnection network topologies and router architectures suitable for large scale NoC implementation representing a departure from the conventional m×n mesh and small routers. In future many-core NoC, the number of modules is expected to be large and could reach above one thousand in the near future [16]. A mesh topology will be a bottleneck because of the power consumption and accumulated delay of the increased hops between routers in the mesh network. So, the mesh topology will lead to an inefficient design and should be replaced by other topologies that can be segmented and divided according to traffic shape. So, star, hierarchical-star, and fat-tree topologies may be good candidates for future designs. In this work, we implemented three 128×128 input/output router architectures and evaluated them compared to the distributed 5×5 Crossbar routers used in 128 tiles mesh.

Fig. 6 depicts the main components of the routers. Our designs consist of only three main components: the switching core, the input unit, and the arbitration unit where the output unit is not employed because we implement FIFO

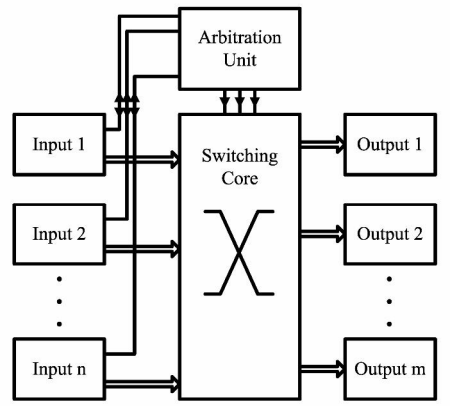

Figure 6. The main router components.

and VOQ buffering strategies which are input buffers. Each component is discussed in the following subsections.

### A. Switching Cores

We implemented two large switching cores. The first is 128×128 Crossbar. The Crossbar is consists of 128 horizontal buses represent the 128 inputs. On each bus, 128 crosspoits are connected and each cross point is dedicated to one output. The crosspoint is constructed by a 2-input AND gate; the first AND gate input is the data input, and the second input is the control signal. An OR gate at each output used to collect all outputs of the crosspoints dedicated to the same output unit.

The second switching core is a 128×128 Batcher-Banyan. The Banyan switching network consists of seven stages, each stage contains 64 nodes and each node is a 2×2 self routing switch. Each node includes two multiplexers and a controller for the self routing property, the controller controls the two multiplexers according to a certain bit of the destination address in the header of the input packets according the stage where the node belongs.

The Batcher sorting network consists of seven stages; each stage contains a different combination of the self-routing 2×2 sorting nodes in progression manner. Each sorting node includes two multiplexers and a controller for the self routing property, the controller controls the two multiplexers according to the destination address in the header of the input packet.

### B. Input Units

The input unit is the front stage of the router. There is an input unit for each input port. It handles address extracting, and storing the incoming packet.

We implemented two different buffering strategies; FIFO input queuing and VOQ. FIFO Input queuing buffering strategy is the direct and simplest way of implementation where incoming packets are stored in one queue. There are two main controllers inside each FIFO input unit, Write Controller which is responsible of storing the incoming packets in the buffer and Read Controller which is responsible of restoring packets from the buffer.

The drawback of this strategy is that if the head packet had to wait because its destination port is busy receiving from another input unit, other packets in the queue have to wait until the head packet could be served even if its destination port is idle. This problem is named Head of Line (HOL) blocking and limits the throughput to 58.6% [17].

VOQ buffering strategy divides the buffering area to a number of queues equal to the number of output ports. Each queue holds packets with same destination address. The input unit with VOQ is able of issuing a separate request from each virtual queue and can read from the only one granted queue. The buffering area can be divided among the virtual queues in the design phase or in the run phase. If the buffering area is divided in the design phase, the virtual queues are assigned fixed sizes and that is called Fixed Allocation VOQ. The traffic pattern dictates each queue share from the whole buffer. If the traffic is uniform or unknown, the buffer must be divided equally among the virtual queues. Otherwise, virtual queues can be assigned different sizes. The buffering area can be divided dynamically in the run phase as needed by incoming packets. The virtual queues are dynamically assigned different sizes according to the input traffic shape and in this case it will be named Dynamic Allocation VOQ and we implemented it in this work.

In this work, we implemented Dynamic allocated VOQ buffering strategy. There are many management algorithms to control dynamically allocated VOQ like linked list, self compacting, and circular buffer [18]. Linked list buffer management is employed in this work. The linked list buffer management scheme keeps a list (queue descriptor) for each virtual output queue and another list for additional queue called the free space queue which holds the rest and unoccupied memory blocks. Each queue list (descriptor) contains a flag that shows whether the block is empty or not, and two pointers; one points to the head of the queue and the other points to the tail of the queue. For each block, a pointer is needed to point to the next block that belongs to the same queue. Managing the input unit employing Dynamic Allocation VOQ using Linked List algorithm required three controllers; Writing Controller, Reading Controller, and Linked List Update Controller.

Virtual output queuing overcomes the HOL problem in the FIFO input queuing where each queue can issue a separate request and any queue can be served independently. The use of dynamic buffer allocation provides better utilization of the buffer space, compared to the Static Allocated VOQ [18]. Dynamic allocation decreases the number of dropped packets by approximately 30% on average, also provides higher reliability. In addition, dynamic allocation of buffers reduces packet loss which is equivalent to improving efficiency [19]. Reducing packet loss in queuing systems as the result of utilizing an effective buffer management system is an important issue in the design of high performance switches.

### C. Arbitration Units

We applied two different arbitration units to solve the destination contention in the switching cores. The choice of the arbitration unit is tied to the input buffering strategy and the type of switching core that will be employed.

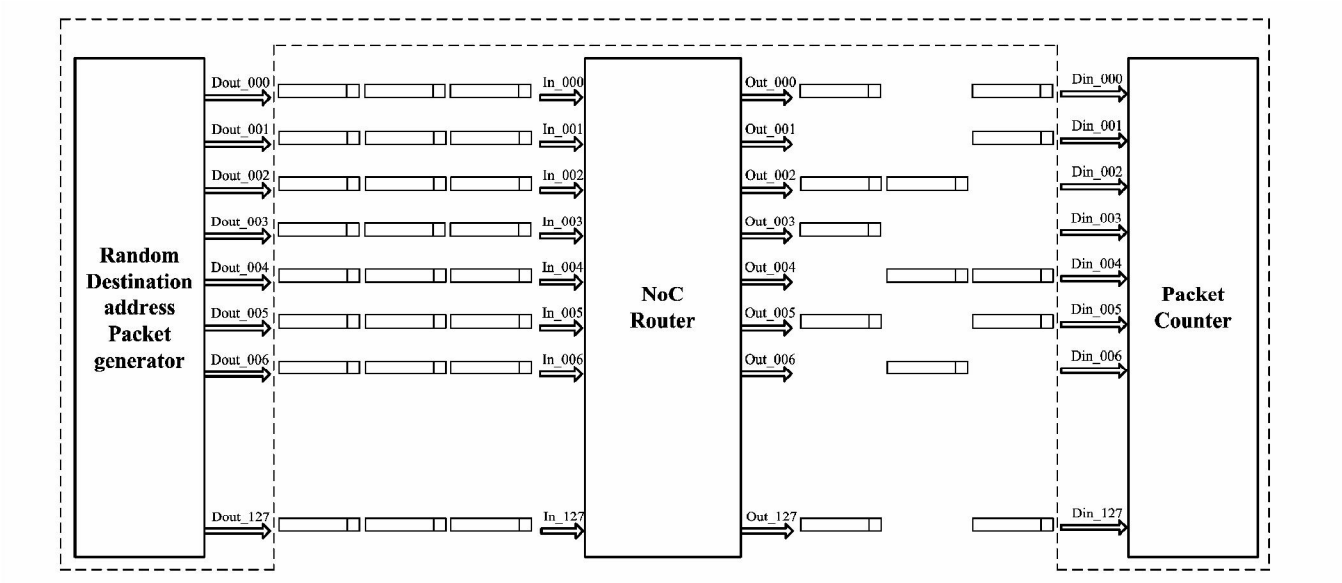


Figure 7. The simulating environment includes the router.

If Batcher-Banyan with FIFO input buffer strategy is employed, the simplest arbitration is the Ring-Reservation discussed before. If VOQ is employed, a fast and intelligent arbitration mechanism is required. There are many theoretical arbitration algorithms for virtual output queuing like Maximum Size Matching, Maximum Weight Matching, Oldest Cell First (OCF), Longest Port First (LPF), Parallel Iterative Matching (PIM), Round Robin Matching (RRM), iSLIP, etc. Most of them are hard to implement in hardware or even impossible [14] [20]. We implemented the DPA. It consists of about 32k arbitration cells and a 255 bit priority vector rotates each arbitration cycle.

### *D. Implementation and Evaluation Details*

We implemented three different 128×128 input/output router architectures. The first is FIFO input with Batcher-Banyan, the second is VOQ input with Batcher-Banyan and the last is VOQ with Crossbar. The designs are based on synchronous transmission where each input is synchronized and detected using separate one bit signal indicates the start of packet when pulsed high. The input port data width is eight bits wide which represent one phit (phit is the data transmitted in one clock cycle). The design is based on fixed packet size. The packet can contain any number of phits, so it can be transmitted in one clock cycle as one wide phit or divided on multiple cycles as small phit to reduce the number of wires and hence area. All the router components are developed in VHDL. Each component is tested on Mentor Graphics ModelSim 6.5. The overall switch is then tested as a unit. We developed a simulating environment resembling normal operations to test performance aspects, throughput and average packet delay under various traffic loads. The simulating environment is consisting of two components. The first component is the load generator and it is attached to the input ports of the router. The second component is the data calculator and it is attached to the output ports of the router. The load generator is responsible for generating the input traffic load to the router. It generates packets with random destination addresses using a large pseudo random generator which generates random destination addresses with uniform distribution. It also assigns a four bytes time stamp to every packet upon its output from the generator to the router using a time counter that counts the router operating clock. The load generator can generate fixed traffic loads which is considered more severe test that burst loads if adjusted to high load rate. The fixed traffic load can be adjusted by varying the width of the gap between the generated packets on each port. It can also control the time of exposing the router to the specified load by changing the amount of packets entering the router with the specified traffic load during simulation. The data calculator consists of a packet counter to count the number of received packets at the output to calculate throughput. It also compares the time stamp of every packet with the entrance time of the time counter to calculate its delay, and then it calculates the average delay and standard delay deviation. Fig. 7 shows the router with respect to the simulating environment.

We synthesized the routers using Synopsys Design Compiler to get the area and dynamic power, and then we evaluate the routers by calculating the efficiency of the routers. We are defining two router efficiencies, the first is the area efficiency which is throughput normalized to one area unit and the other is the power efficiency which is the throughput normalized to one dynamic power unit.

## V. PERFORMANCE EVALUATION RESULTS

In FIFO input buffering strategy, throughput is limited due to (HOL). We tested the throughput with varying the traffic load for 32, 64, and 128 blocks input buffer for time duration of 1000 packets input to each port. As expected, larger buffers enhance the performance but it is limited by

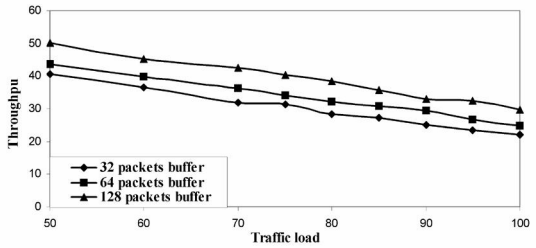


Figure 8. Throughput of the FIFO input buffer strategy with different loads.

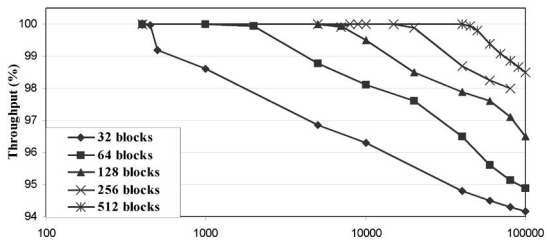


Figure 9. Throughput of the VOQ buffering strategy under 100% traffic load with varying the time of exposed load.

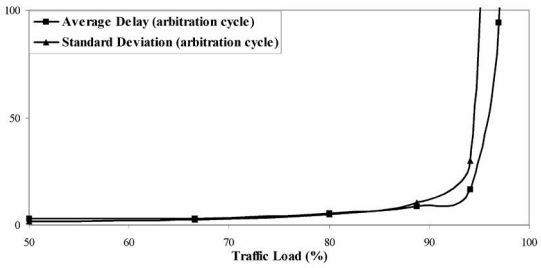


Figure 10. The average latency of the VOQ router with the exposed load.

the maximum throughput in input buffering system due to HOL blocking in line with the results of [17]. Fig. 8 shows the FIFO throughput with traffic load.

VOQ buffering strategy gives optimum performance (100% throughput) when exposed to 97% continuous traffic load for 100,000 packets on each port. For 100% traffic load, optimum performance can be obtained by using infinite buffer size. But for finite buffer sizes, throughput will start to decrease (packets will start to be dropped) depending on the amount of buffers due to the lack of buffering space. Fig. 9 shows the throughput of the router with the exposure time duration under 100% traffic load.

The latency of the router comes from the contention between the arriving packets. As load increases, packet average latency increases. At high traffic loads, latency increases exponentially. Fig. 10 shows the measured average latency and its standard deviation in arbitration cycles with different traffic loads.

We synthesized our designs using Synopsys Design Compiler. As theoretically, we found that Crossbar occupies 147% larger area because it uses more crosspoints than the Batcher-Banyan. However, Batcher-Banyan consumes 120% more dynamic power consumption because each Batcher-Banyan node contains sequential control logic for the self routing property which consumes more power. Fig. 11 shows both area and dynamic power consumption of synthesized gates of the switching cores.

For area costs of the router architectures (excluding the buffer memory), FIFO input with Batcher-Banyan shows the least area because the simple logic of all of the FIFO input and the Ring-reservation arbitration unit. Both the VOQ's occupy about five times larger area than the FIFO. Whereas the sum of 128 small 5×5 VOQ-Crossbar has about three times larger area than the FIFO. The VOQ-Crossbar occupies only 1.04 times the VOQ-Batcher-Banyan area because the arbiter dominates the area over the switching cores where the area of the DPM arbiter increases exponentially with the size of the router. Fig. 12 shows the area costs of the router architectures excluding the buffer memory.

FIFO input consumes very low power compared with the VOQ's because of its simple logic. Whereas the sum of 128 small 5×5 VOQ-Crossbar consumes only about three times the FIFO power. The VOQ-Crossbar showed different result than expected from the previous results, it consumes only 1.03 more power than the VOQ-Batcher-Banyan, and that is because the Crossbar requires a huge number of control signals (wires) from the arbitration unit to turning on and off the crosspoints, and that causes more power consumption in the internal wires. Whereas the Batcher-Banyan has a self-routing property and does not need any control from outside. The power consumption of the large VOQ's is about ten times that of 128 small 5×5 VOQ-Crossbar because all of the VOQ logic, DPA and the Crossbar exponential growth in complexity with the increase of the number of ports. Fig. 13 shows the dynamic power consumption of the router architectures.

The FIFO input router shows the highest efficiency despite its low throughput because its very low area and power consumption. The efficiency of 128 small 5×5 VOQ-Crossbar is in the second stage after the FIFO. Both the 128×128 VOQ gives lower area efficiency and a lot lower power efficiency. VOQ-Batcher-Banyan gives slightly more area and power efficiencies than the VOQ-Crossbar. Fig. 14 shows area efficiency and Fig. 15 shows power efficiency of the full routers.

## VI. CONCLUSION

In this paper, we discussed our proposal to the future many core NoC architectures employed by star, hierarchical-star, and fat-tree network topologies and large size routers. We implemented three 128×128 router designs, the first is FIFO input with Batcher-Banyan switching core, the second is VOQ input with Batcher-Banyan and the last is VOQ input with Crossbar switching core. We evaluated the three 128×128 router designs according throughput, area and power. We developed a network simulating environment for testing throughput and average delay under various loads and number of input packets. We evaluated their efficiency compared to 128 5×5 VOQ-Crossbar routers employed in conventional 2-D mesh.

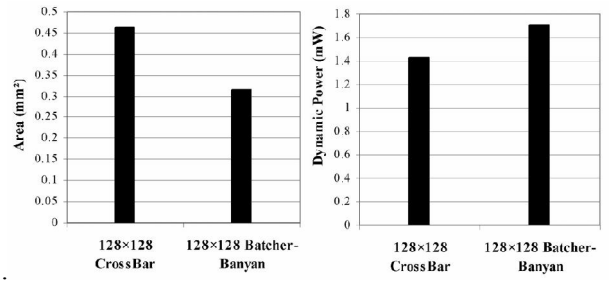


Figure 11. Area and Power of Batcher-Banyan vs. Crossbar switching cores.

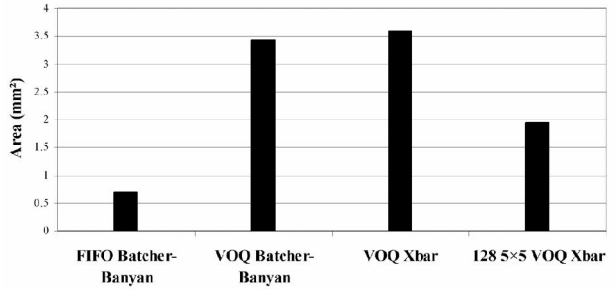


Figure 12. Area costs of the router architectures excluding the buffer memory.

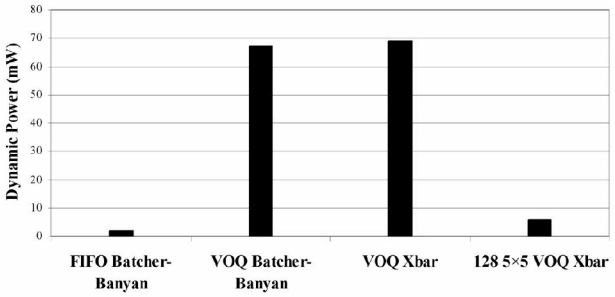


Figure 13. Dynamic power consumption of the full router architectures excluding buffer memory.

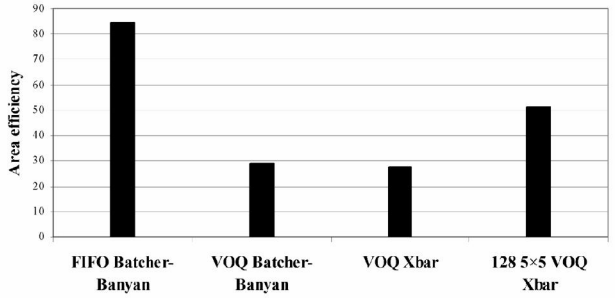


Figure 14. Area efficiency of the full routers.

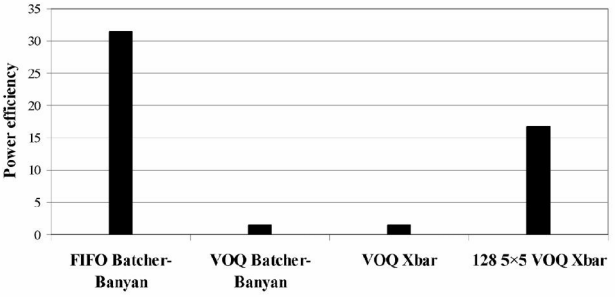


Figure 15. Power efficiency of the full routers.

The results showed that the FIFO with Batcher-Banyan is the highest efficiency, then the 128 5×5 VOQ-Crossbar. Both the VOQ with Batcher-Banyan and Crossbar showed the lowest efficiency.

## REFERENCES


[1] C. Bamji, M. Berkens, A. B. Kahng. "Automated layout and migration in ultra-deep submicron VLSI", June 25, 1999.

[2] L. Benini, G. DeMicheli, "Networks on Chips: a New SoC paradigm", computer 35 (1) (2002) 70–78.

[3] http://www.oracle.com/us/products/servers-storage/servers/sparc-enterprise/t-series/sparc-t4-processor-ds-497205.pdf

[4] http://www.ambric.com/products_mppa.php

[5] http://www.nvidia.com/docs/IO/43395/NV_DS_Tesla_C2050_C2070_jul10_lores.pdf

[6] H. Wang, L. Peh, and S. Malik, "A Technology-Aware and Energy-Oriented Topology Exploration for On-Chip Networks", in Proc. DATE, 2005, pp.1238-1243.

[7] W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks", Proc. 38'th DAC, 2001.

[8] C. Xuning and L. S. Peh, "Leakage power modeling and optimization in interconnection networks", in Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED), pp. 90–95, 2003.

[9] T. T. Ye, L. Benini, and G. De Micheli. "Analysis of power consumption on switch fabrics in network routers", In Proceedings. 39th Design Automation Conference, 2002.

[10] G. Passas, M. Katevenis, and D. Pnevmatikatos, "A 128 × 128 × 24Gb/s crossbar interconnecting 128 tiles in a single hop and occupying 6% of their area", 4th ACM/IEEE International Symposium on Networks on Chip, NOCS 2010. (pp. 87-95). IEEE Computer Society.

[11] L. R. Goke and G. J. Lipovski, "Banyan networks for partitioning multiprocessor systems", Proc. 1st Annu. Int. Symp. Comput. Architecture, pp. 21-28, Dec. 1973.

[12] K. E. Batcher, "Sorting networks and their application", Proc. Spring JointComput. Conf., AFIPS, pp.307-314, 1968.

[13] B. Bingham and H. Bussey, "Reservation-based contention resolution mechanism for Batcher-banyan packet switches", Electron. Lett., vol. 24, no. 13, pp. 772 773, Jun. 1988.

[14] J. Hurt, A. May, X. Zhu, and B. Lin, "Design and implementation of high-speed symmetric crossbar schedulers", Proc. IEEE International Conference on Communications (ICC'99), Vancouver, Canada, June 1999, pp. 253-258.

[15] Y. Tamir, H. C. Chi, "Symmetric crossbar arbiters for VLSI communication switches", IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 1, pp. 13-27, 1993.

[16] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang, and A.N. Choudhary, "Firefly: illuminating future network-on-chip with nanophotonics", in Proc. ISCA, 2009, pp.429-440.

[17] M. J. Karol, M. Hluchyj, and S. Morgan, "Input vs. output queuing on a space-division packet switch", in Proc. GLOBECOM 1986, pp. 659–665.

[18] N. Ni, M. Pirvu, and L. Bhuyan, "Circular buffered switch design with wormhole routing and virtual channels," International Conference on Computer Design, pp. 466-673, 1998.

[19] M. Fayyazi, D.R. Kaeli, and Z. Navabi, "Dynamic Input Buffer Allocation (DIBA) for Fault Tolerant Ethernet Packet Switching", in Proc. PDPTA, 2003, pp.819-823.

[20] N. McKeown, V. Anamtharam, and J. Warland, "Achieving 100% throughput in an input-queued switch", Proc. INFOCOM'96, San Francisco, March 1996, pp. 296-302.