A New Client Interface Architecture for the Modified Fat Tree (MFT) Network-on-Chip (NoC) Topology

Abdelhafid Bouhraoua and Muhammad E. S. Elrabaa
Computer Engineering Department
King Fahd University of Petroleum and Minerals
PO Box 969, 31261 Dhahran, Saudi Arabia
{abouh,elrabaa}@kfupm.edu.sa; orwa@diraneyya.com

***Abstract*— A new client interface for the Modified Fat Tree (MFT) Network-on-Chip (NoC) is presented. It is directly inspired from the findings related to lane utilization and maximum FIFO sizes found in simulations of the MFT. A new smart arbitration circuit that efficiently realizes a round-robin scheduler between the receiving lanes has been developed. Simulation results show a clear viability and efficiency of the proposed architecture. The limited number of the active receiving links has been verified by simulations. Simulations also show that the central FIFO size need not be very large.**

***Keywords* — Networks-On-Chip, Systems-on-Chip, ASICs, Interconnection Networks, Fat Tree, Routing**

I. INTRODUCTION

There has been a significant amount of effort made in the area of NoCs, and the focus has mostly been on proposing new topologies, and routing strategies. However, recently the trend has shifted towards engineering solutions and providing design tools that are more adapted to reality. For example, power analysis of NoC circuitry has intensively been studied [1, 2], more realistic traffic models have been proposed [3], and more adapted hardware synthesis methodologies have been developed.

However, high throughput architectures haven't been addressed enough in the literature although the need for it started to become visible [4]. Most of the efforts were based on a regular mesh topology with throughputs (expressed as a fraction of the wire speed) not exceeding 30% [5]. In [6, 7] a NoC topology based on a modified Fat Tree (MFT) was proposed to address the throughput issue. The conventional Fat Tree topology was modified by adding enough links such that contention was completely eliminated thus achieving a throughput of nearly 100% [6] while eliminating any buffering requirement in the routers. Also, simplicity of the routing function, typical of Trees, meant that the router architecture is greatly simplified. These results did not come without a price, mainly the high number of wires at the edge of the network in this case. Also buffering was pushed to the edge of the network at the client interfaces. Many of these issues have been discussed in [6, 7].

In order to overcome these limitations a new client interface architecture is proposed. This interface aims at reducing the number of parallel FIFOs into a single centralized FIFO. The modification of the client interface opens the door for a more practical implementation of the MFT NoC. This paper presents the new client interface architecture and its different circuitry. It also shows through simulation that the new architecture draws on the practical results to reduce the amount of required hardware resources. MFT NoCs are first briefly reviewed in the next section. The newly proposed client interface is then presented in section 3. Simulations results are presented in section 4 followed by conclusions in section 5.

II. MODIFIED FAT TREE NOCS

MFT is a new class of NoCs based on a sub-class of Multi-Stage Interconnection Networks topology (MIN). More particularly, a class of bidirectional folded MINs; chosen for its properties of enabling adaptive routing. This class is well known in the literature under the name of Fat Trees (FT) [8]. The FT has been enhanced by removing contention from it as detailed in [7]. Below is a brief description of the FT and MFT network topologies.

A. FT Network Topology

A FT network, Figure 1, is organized as a matrix of routers with n rows; labeled from 0 to $n-1$; and $2^{(n-1)}$ columns; labeled from 0 to $2^{(n-1)}-1$. Each router of row 0 has 2 clients attached to it (bottom side). The total number of clients of a network of n rows is 2^n clients. The routers of other rows are connected only to other routers. So, in general, a router at row r can reach $2^{(r+1)}$ clients.

B. MFT Topology

Contention is removed in MFT by increasing the number of output ports in the downward direction of each router [6, 7], Figure 2. At each router, the downward output ports (links) are double the number of upper links. Then the input ports of the adjacent router (at the lower level), to which it is connected are also doubled. This is needed to be able to connect all the output ports of the upper stage router. This

will continue till the client is reached where it will have $2^n - 1$ input links. Each of these I/P links will feature a FIFO buffer as called for by the original MFT architecture [6, 7].

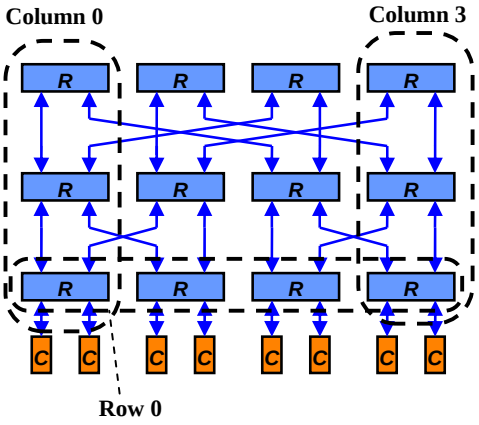

Figure 1: Regular Fat Tree Topology (8 clients)

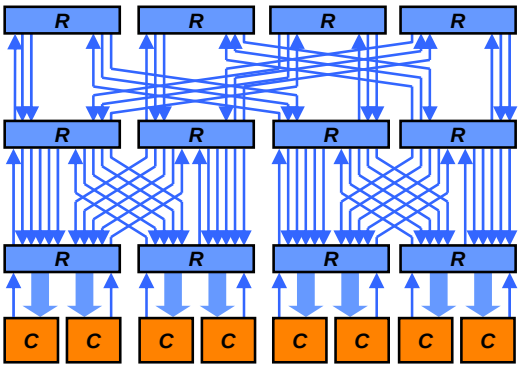

Figure 2 – Modified FT Topology

III. Client Interface

As was explained in section II, the original MFT architecture requires $2^n - 1$ input FIFOs at the client interface. The sizes of these FIFOs is set by the NoC designer depending on many factors such as the communication patterns among clients, emptying (data consumption) rate by a client, application requirements (latency), …etc. [7]. Figure 3 below shows the structure of the client interface in the original MFT.

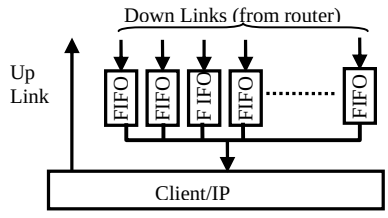

Figure 3: Client interface of the original MFT

It is evident that although these FIFOs may be of a small size, their structure represents the largest part of the cost in terms of area for the MFT network since the routers are bufferless and have a very low gate count. Also, extensive simulations with different traffic generators showed that only a small fraction of FIFO lanes are active per client simultaneously [6, 7].

In order to reduce the wasted FIFOs space represented by the original MFT's client interface, a newly designed interface is proposed, Figure 4. It is made of two parts; an upper part consisting of several bus-widener structures that will be named *parallelizers* from this point forward and a lower part that is simply a single centralized FIFO memory to which all the outputs of the different parallelizers are connected through a single many-to-one multiplexer.

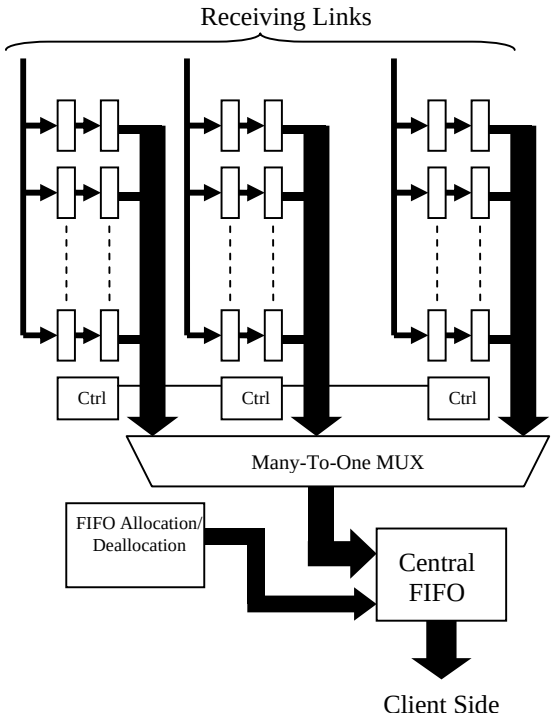

Figure 4: Block Diagram of the New Client Interface.

Each one of the parallelizers is made of two layers. The first layer is a collection of registers connected in parallel to the incoming data bus from one of the receiving ports. Packet data is received into one of these registers one word at a time. When this layer is full, an entire line made by concatenating all the registers of the first layer is transferred to a second set of registers (the second layer in the parallelizer) in a single clock cycle. The ratio between the width of the parallel bus and the width of a single word is called *the parallelization factor*.

Packets portions from different sources are received on different parallelizers simultaneously and independently. When the first portion of a packet is received and transferred to the second layer of the parallelizer, a flag is set to request transfer of a new packet to the FIFO. The control logic responsible for these transfers will first attempt to reserve space in the FIFO corresponding to one

packet. The condition here for this architecture to produce efficient results is the adoption of a fixed packet size. This condition simplifies the space allocation in the FIFO and alleviates the control logic from any space or fragmentation management due to variable size allocation and disposal. In the case the FIFO is full and no space could be reserved, the request is rejected and the backpressure mechanism is triggered on that requesting port.

The control logic continuously transfers the received packet words to the FIFO. Every time a packet portion enters the second layer of registers in one of the parallelizers a flag is set to indicate the presence of data. Those parallelizers which are currently receiving packets are said to be active. Only active parallelizers are continuously polled to check the presence of data. The polling follows a round-robin policy. A single clock cycle is used to process the currently selected parallelizer.

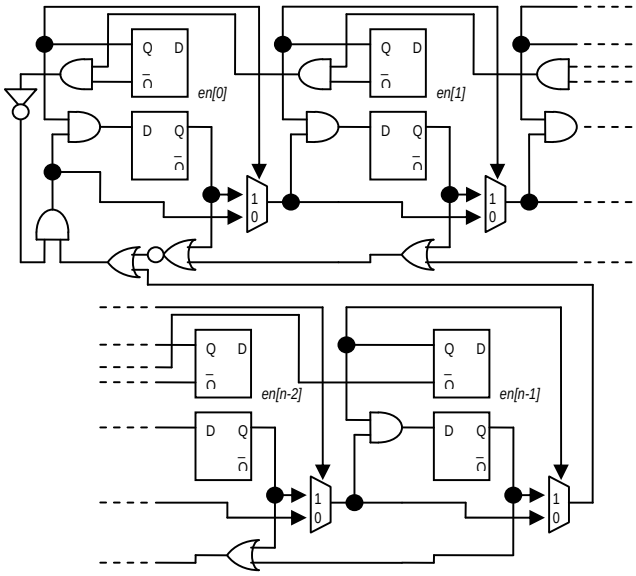

Figure 5: Intelligent Request Propagation Circuitry.

Polling the active parallelizers only supposes some mechanism to "skip" all the non-active parallelizers between two active ones. In order to avoid wasting clock cycles crossing those non-active parallelizers, a special request propagation circuit has been designed. Figure 5 shows this circuit's schematic. The upper set of flip-flops correspond to the status flag indicating whether a parallelizer is active or not while the lower one is used to indicate which parallelizer is selected to transfer its data during a given clock cycle. The multiplexers are used to instantly skip the non-active parallelizers.

As a result of this fast polling scheme packets arriving simultaneously on different parallelizers may be received in different order. Packet order from the same source is still guaranteed though because the network is bufferless.

IV. Simulation Results

Simulations were carried out using a cycle-accurate C-based custom simulator (developed in-house) that supports uniform and non-uniform destination address distribution as well as bursty and non-bursty traffic models. The packet size was fixed at 64 words. Only bursty traffic with non-uniform address generation was used. The varying parameters were: the network size, central FIFO size and the parallelizer factor value.

Five injection rates corresponding to 0.5, 0.6, 0.7, 0.8 and 0.9 words(flits)/cycle/client were simulated. The first result confirming the viability of the solution was the throughput that matched the input rate in most of the cases and with a maximum difference lower than 1% in few cases.

In all the figures that follow, the latency is expressed in clock cycles.

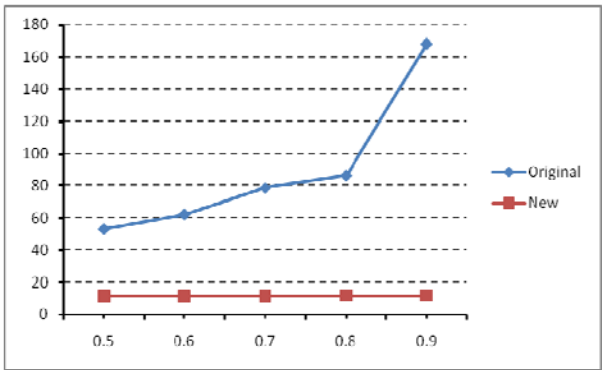

Figure 6: Latency Comparison with the original client interface.

Latency values were reduced dramatically because of the output rate of the FIFO. Dual-port memories are the natural choice for implementing FIFOs. Both data buses are generally the same size on both ports of the dual-port memory. Therefore, the FIFO data bus has the same size as the paralellizers bus. A wide output bus translates in fewer clock cycles to read or write an entire packet. More packets are moved per unit of time which means that packets spend less time in the FIFO waiting to be sent out leading to smaller latencies as shown in Figure 6. It is important to note that the latency figures across the network did not change and is expected to be small as the entire network is bufferless.

Figure 7 shows a subset of the obtained simulation results. The 32 clients network results are shown (left to right) for parallelization factor values of 8, 16 and 32 for different central FIFO sizes (from 8 packets to 32 packets). The latency figures are very low compared to those obtained with the previous architecture of the client interface (Figure 6). The latency range corresponding to a wider parallelizer is lower than the range corresponding to a narrower one. The other results (not shown here for lack of space) are similarly lower for wider parallelizers. These findings confirm the efficiency of the proposed solution.

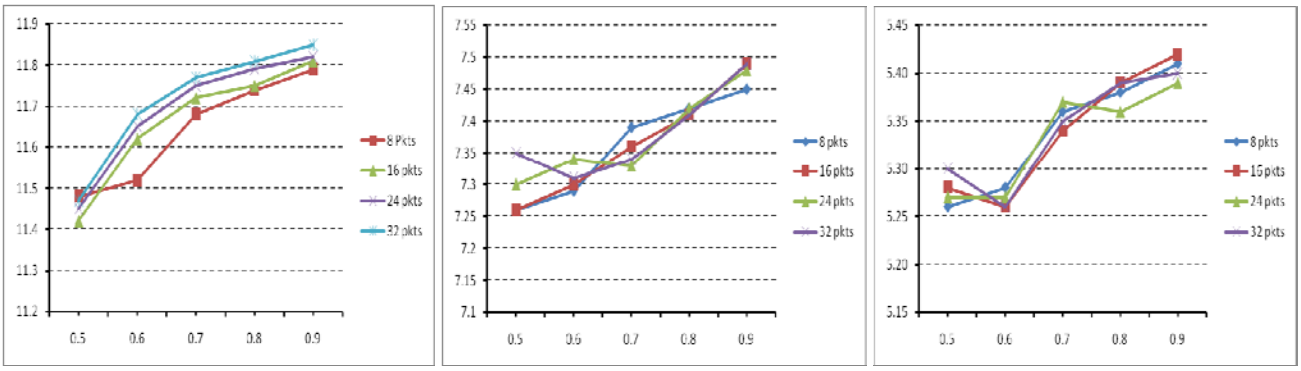

Figure 7 - Simulation Results: latency (clock cycles) versus injection rate (flits/clock cycle/client) for three parallelization factors (8, 16 and 32 from left to right) for several central FIFO sizes (8, 16, 24 and 32 packets).

The central FIFO size has little impact on the results which favors size reduction as a FIFO with a size as low as 8 packets can produce acceptable results.

A tentative synthesis of the new structure yielded about 35K gates per client for a parallelization factor of 8 for a network of 64 clients. This represents approximately an area of 0.185 mm^2 for the 0.13 µ technology. Added to that the dual-port SRAM area of 0.009, 0.018, 0.028 and 0.038 mm^2 for a FIFO size that accommodates respectively 8, 16, 24 and 32 packets. This represents a significant improvement compared to the 2.1 mm^2 occupied by the client interface in the previous architecture and which uses 63 SRAM FIFOs of 2K-Bytes each.

V. Conclusions

A new architecture of the client interface of the MFT NoC has been proposed. This new architecture considerably reduces the hardware resources necessary to implement the receiving client interface. Detailed block diagrams and of this architecture have been shown and described. Its operations and step by step behavior have been described as well. A new arbitration circuit that intelligently "skips" disabled request lines to realize an efficient round-robin where no clock cycles are wasted is presented. Simulations have given clear evidence on the viability of a single centralized FIFO that is simultaneously filled by several, yet limited number, of receiving links. The limited number of the active receiving links has been verified by simulations. The simulation results have shown a considerable reduction of latency compared with the previous solution. They have also shown the little impact of the FIFO size on the latency which implies that a larger size FIFO is not necessary. The client interface synthesis yielded smaller area than in the previous architecture of the client interface by one order of magnitude.

Acknowledgments

This work was supported by King Fahd University of Petroleum and Minerals (KFUPM) through grant # IN070367.

References

[1] E. Nilsson and J. Öberg, "Reducing power and latency in 2-D mesh NoCs using globally pseudochronous locally synchronous clocking", CODES+ISSS 2004.

[2] E. Nilsson and J. Öberg, "Trading off power versus latency using GPLS clocking in 2D-mesh NoCs", ISSCS 2005.

[3] V. Soteriou, H. Wang and L. Peh, "A Statistical Traffic Model for On-Chip Interconnection Networks", in *Proceedings of the 14th IEEE Intl. Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '06)*, Sept. 2006, pp 104-116.

[4] Freitas H.C., Navaux P. "A High Throughput Multi-Cluster NoC Architecture", *11th IEEE International Conference on Computational Science and Engineering*, July 16-18, 2008 Sao Paulo, Brazil, pages 56-63

[5] K. Goossens, J. Dielissen, A. Radulescu, "Æthereal network on chip: concepts, architectures, and implementations", IEEE Design and Test of Computers, Volume 22, Issue 5, Sept.-Oct. 2005 Page(s)414 – 421.

[6] A. Bouhraoua and Mohammed E.S. El-Rabaa, "A High-Throughput Network-on-Chip Architecture for Systems-on-Chip Interconnect," *Proceedings of the International Symposium on System-on-Chip* (SOC06), 14-16 November 2006, Tampere, Finland.

[7] A. Bouhraoua and Mohammed E.S. El-Rabaa, "An Efficient Network-on-Chip Architecture Based on the Fat Tree (FT) Topology", *Special Issue on Microelectronics, Arabian Journal of Science and Engineering,, Dec. 2007*, pp 13-26.

[8] C. Leiserson, "Fat-Trees: Universal Networks forHardware-Efficient Supercomputing", *IEEE Transactions on Computers*, Vol. C-34, no. 10, pp. 892-901, October 1985.