A Simplified Router Architecture for the Modified Fat Tree Network-on-Chip Topology

A. Bouhraoua, O. Diraneyya and M.E. Elrabaa
King Fahd University of Petroleum and Minerals,
P.O. Box 969, 31261 Dhahran, Saudi Arabia
Email: {abouh, orwa, elrabaa}@kfupm.edu.sa

***Abstract*—The architecture of the class of routers to implement the modified Fat Tree topology is shown. The router architecture is buffer-less with a simplified routing function. The routing function is obtained from a model that describes the Fat Tree topology and from where the equations governing the routing circuitry are derived. A parameterized router model is developed and coded in verilog. A modified Fat Tree network generator that uses the router model is also developed. The generator produces verilog files directly used in functional simulation.**

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

I. Introduction

The *Networks-on-Chips* (NoCs) paradigm, in *Systems-on-Chips* (SoCs), has emerged as an alternative to ad-hoc wiring or bus-based global interconnecting networks. Actually, because technology scaling has enabled the integration of a higher number of processing elements, computational cores and memories, the complexity of communication between these cores is also increasing. Added to that, very short time-to-market constraints and higher stress on the design methodologies are two conditions that led to the consideration of NoCs. The idea of turning the on-chip interconnecting network into yet another IP block that is both flexible and scalable is very appealing to the time limitations imposed by the market requirements. The main advantage of this approach is the fact that it constitutes a systematic solution for the common issues of compatibility, bandwidth requirements, and performance. Hence, the general consensus is that the communication requirements, as well as the design flow of billion-transistor SoCs are best accommodated by shared, segmented interconnection networks [3,4].

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[7], more realistic traffic models have been proposed [9], and more adapted hardware synthesis methodologies have been developed.

However, high throughput architectures haven't been addressed enough in the literature. Besides the efforts related to the Nostrum[5] and the Æthreal[6], most of the other efforts were based on a regular mesh topology with a throughput (expressed as a fraction of the wire speed) not higher than 30%[5]. The modified Fat Tree (FT), proposed in[1,2] aims to address the throughput issue. The topology is modified in order to eliminate contention and to achieve a throughput of nearly a 100%[1]. This result does not come without a price which is mainly the high number of wires at the edge of the network. Many of the aspects related to this issue have been discussed in previous publications[1,2]. In this paper, the network construction, the formal determination of the routing function and the router architecture are presented. After this introduction, section II presents the fundamentals and network construction of the FT topology. The routing function is formally defined in section III while the router architecture with the routing function circuitry are presented in section IV. Section V concludes this paper and draws the future directions of this effort.

II. Fat Tree Topology

The architecture considered 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. This class is well known in the literature under the name of Fat Trees (FT) [10]. The FT has been enhanced by removing contention from it as detailed in [2].

A. Network Topology

The network 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. Any router is identified by its position (r, c); r denoting its row index and c denoting its column index.

In general, a router (r,c) is connected to two routers at row $r+1$:

- router $(r+1, c)$
- router $(r+1, c-2^r)$ or router $(r+1, c+2^r)$ based on whether $\lfloor c/2^r \rfloor$ is odd or even, respectively.

Hence, two clients can be reached from any router in row 0. A router at row 1 is connected downwards to two routers at row 0. This means that it can reach $2x2 = 4$ clients. A router at row 2 is connected to two routers at row 1; thus reaching $2x4$

= *8* clients. So, in general, a router at row *r* can reach $2^{(r+1)}$ clients.

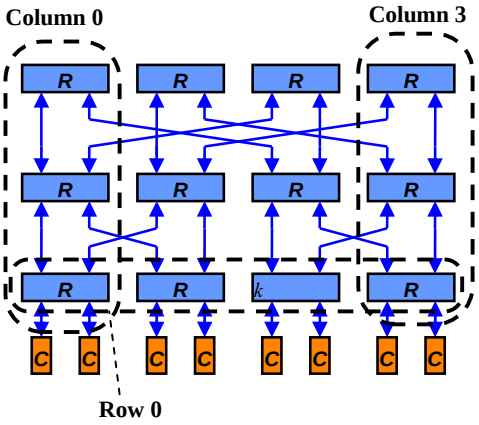
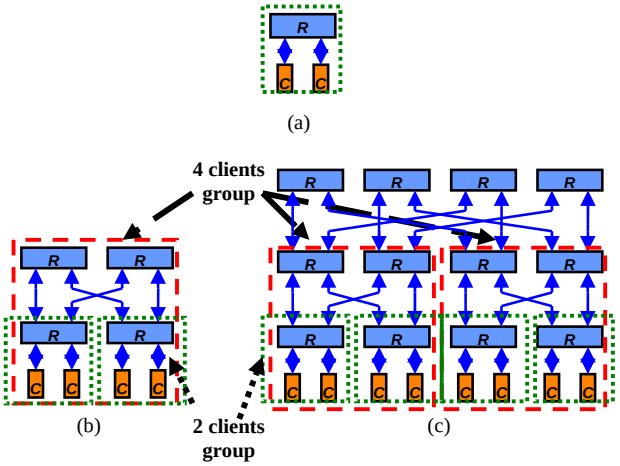


Figure 1: Regular Fat Tree Topology

Figure 1 shows a regular FT of *3 x* (2^{3-1}) routers and 2^3 clients.

B. Network Structure

In a more general view, the FT network is built recursively starting from a single router with two clients attached o it. Figure 3 illustrates the recursive building of the network. It also shows that the routers and clients belong to groups. The notion of group will be useful later when describing the routing scheme.



(a) 2 clients group (order 1);
(b) 4 clients group made of 2 two clients groups (order 2)
(c) 8 clients group made of 2 four clients groups (order 3)

Figure 2: Router Groups

A group of order *r* can be defined as follows (as illustrated in Figure 2):

- A structure of routers organized in *r* rows and 2^{r-1} columns.
- Routers at row *0* are always included in the group. Consequently, the clients, which are attached to routers at row *0*, are also part of the group
- A group of order *r* is made of two adjacent groups of order *r-1*. Recursively, a group of order *r* contains 2^{r-1} routers columns and 2^r clients.
- The number of groups of order *r* in the network is equal to 2^{n-r} since the total number of columns is 2^{n-1}.

Any router of coordinates *(r, c)* is connected to two adjacent groups of order *r*. The same router belongs to the group of order *r+1* that contains the two groups of order *r* it is connected to. Any router at row *r* has its left link connected to the group of order *r* on the left and its right link connected to the group of order *r* on the right.

III. ROUTING

Routing in FT simply follows the routing in binary trees. A packet is routed up until it reaches a router that has a path to its destination. This router is called the routing *summit* for convenience. The FT structure, based on a superposition of binary trees, naturally provides packets with several upward paths. Any upward path will eventually lead to a summit where a downward path to the packet's destination is provided.

A. Client Labeling

Clients are labeled in an increasing order starting from the left to the right, with all the labels (i.e. the addresses) being within the interval $[0, 2^n - 1]$. The relation between the client address and the column coordinate of row-0 routers is given by the following equation:

$$addr = 2c + s \quad (1)$$

Where the selector $s = \{0, 1\}$, based on the client's position. For the clients connected to the left of row-0 routers, $s = 0$, and for those connected to the right, $s = 1$.

B. Packet Structure

Packet boundaries are indicated via the use of two flag signals, called the start-of-packet SOP and the end-of-packet EOP signals, respectively. This allows packets of randomly variable-length to be sent over the NoC. The packet data flow in-between the activation of both signals is commenced by a header field, which contains the destination address of the client to which the packet is to be routed, followed by a variable-length data field, which contains the actual communication data. Note that the routing information is all contained within the packet's header.

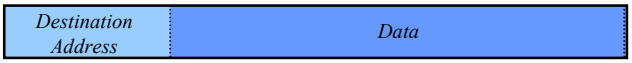


Figure 3 – Packet structure

C. Reach Range

The first step in building the routing scheme is to determine the range of clients that can be reached by a router *(r, c)* on the downward path. From the network scaling section, it has been deduced that any router *(r, c)* is connected to two groups

of order r; a group G_L to its left with an associated address interval I_L and another to its right, G_R with a corresponding address interval I_R. These groups have the following properties:

- The size of I_L is equal to the size of I_R and is 2^r clients.
- G_L and G_R are adjacent because connected to router *(r, c)* thus I_L and I_R are contiguous intervals and $I_L < I_R$.

The two intervals I_L and I_R are hence computed as follows:

$$I_L = [P_L, P_L+2^r -1] \text{ and } I_R = [P_L+2^r, P_L + 2^{r+1} -1] \quad (2)$$

P_L corresponds to the address of leftmost client in G_L. Since P_L is an address, it can be written as (see equation 1):

$$P_L = 2c_L + s_L$$

c_L corresponds to the column coordinate of the leftmost router at row *0* of G_L. Since, P_L corresponds to the address of *leftmost* client in G_L, then $s_L = 0$ because client is on the left. Therefore:

$$P_L = 2c_L$$

Hence, the two intervals I_L and I_R become:

$$I_L = [2c_L , 2c_L + 2^r -1] \quad \text{and} \quad I_R = [2c_L + 2^r, 2c_L + 2^{r+1} -1]$$

c_L is computed from the router column coordinate c as follows:

Any router of coordinates *(r, c)* connected to G_L and G_R has its column coordinate c within the interval $[c_L , c_L + 2^r - 1]$. This is because the router *(r, c)* belongs to a group of order *r+1* composed of the two groups G_L and G_R, which makes its lowest column coordinate the same as for G_L. This also means that the number of router columns in this group is $2^{(r+1) -1} = 2^r$. This actually means that:

$$c = c_L + k;\ 0 < k < 2^r.$$

It clearly shows that the value k can be represented using r bits. Thus c_L is obtained by simply clearing the lowest r bits of the column coordinate c.

D. Finding the "Summit"

A first a packet is routed up until it reaches a summit. It is the first router reached by the packet that provides a path to the packet destination address *(daddr)*. Providing a path to destination means that one of the two intervals I_L and I_R; associated with the summit's left and right lower order groups it is connected to; will contain *daddr*.

Therefore, a router *(r, c*) is a summit if $daddr \in I_L$ or $daddr \in I_R$. If $daddr \in I_L$, the packet is routed to the left and if $daddr \in I_R$, the packet is routed to the right. Before reaching a summit, the packet is always routed up. After traversing the summit, the packet is routed downwards (left or right) until it reaches its destination.

E. Routing Upward

As long as the packet's destination address *daddr* is outside I_L and I_R, the packet is routed up.

Comparing *daddr* with I_L and I_R is apparently needed.

$daddr \in I_L$ or $daddr \in I_R$, is equivalent to:

$$2c_L \le daddr \le 2c_L + 2^{r+1}-1$$

$$0 \le daddr - 2c_L \le 2^{r+1}-1$$

Therefore, when the double inequality is satisfied, the value $daddr - 2c_L$ is represented on a maximum of $r+1$ bits. Because, the lower r bits of c_L (lower $r+1$ bits $2c_L$) are all *0*, to represent the value $daddr - 2c_L$ on *(r+1)* bits requires that the upper *(n- (r+1))* bits of this value to be all 0. Consequently:

$$daddr[n-1:n-r+1] - c_L\,[n-2:n-r] = 0$$

$$daddr[n-1:n-r+1] = c_L\,[n-2:n-r]$$

Finally, the condition upon which a summit is found is that the upper *(n - (r+1))* of *daddr* are equal to the upper *(n-1 – r)* bits of c_L.

F. Routing Downward

A packet reaching a router *(r, c)* from its upper links has already been through a summit and is moving downwards. Similarly, if $daddr \in I_L$, the packet is routed to the left and if $daddr \in I_R$, the packet is routed to the right.

$daddr \in I_L$ if $2c_L \le daddr \le 2c_L + 2^r-1$

$daddr \in I_R$ if $2c_L + 2^r \le daddr \le 2c_L + 2^{r+1}-1$

This means:

$daddr \in I_L$ if $0 \le daddr - 2c_L \le 2^r-1$

$daddr \in I_R$ if $2^r \le daddr - 2c_L \le 2^{r+1}-1$

The value $daddr - 2c_L$ corresponds to the lower r bits of *daddr* which are *bits[r-1:0]*. This is because any value of c_L has its *r-1* lower bits equal to *0* by construction. Thus:

If $0 \le daddr - 2c_L \le 2^r-1$ then $daddr[r-1] = 0$ and if $2^r \le daddr - 2c_L \le 2^{r+1}-1$ then $daddr[r-1] = 1$.

This clearly shows that a single bit of the destination address *daddr* is sufficient to decide the routing direction.

IV. Router Architecture

The router architecture relies on the fact that contention is eliminated along the route to destination, which is achieved, as mentioned in section …, by doubling the links in the downward direction.

A. Overall Architecture

As a consequence of doubling the links in the downward direction, several models of routers will be present in the network. Models differ from one another by the number of links in the downward path. Routers belonging to the same row will be instances of the same router model. A generic router model has been defined using parameterized HDL description.

The architecture of a router model will comprise:

- two input ports and two output ports for the upward path
- *k* input ports and *2k* output ports for the downward path.
- two extra downward outputs folding the upward left input to the right and the upward right input to the left, to address the case when the router is the "summit" for some packets.

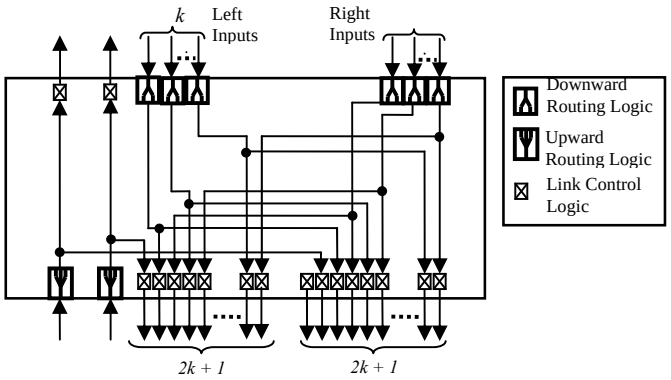


Figure 4 – The adopted router architecture

Figure 4 shows the adopted router architecture with the doubling of the output ports on the downward direction. No internal FIFOs are required since no contention can occur. The internal structures of the different ports making the architecture of the router are shown in Figure 5. The sheer simplicity of the router compensates for the increase in the number of ports.

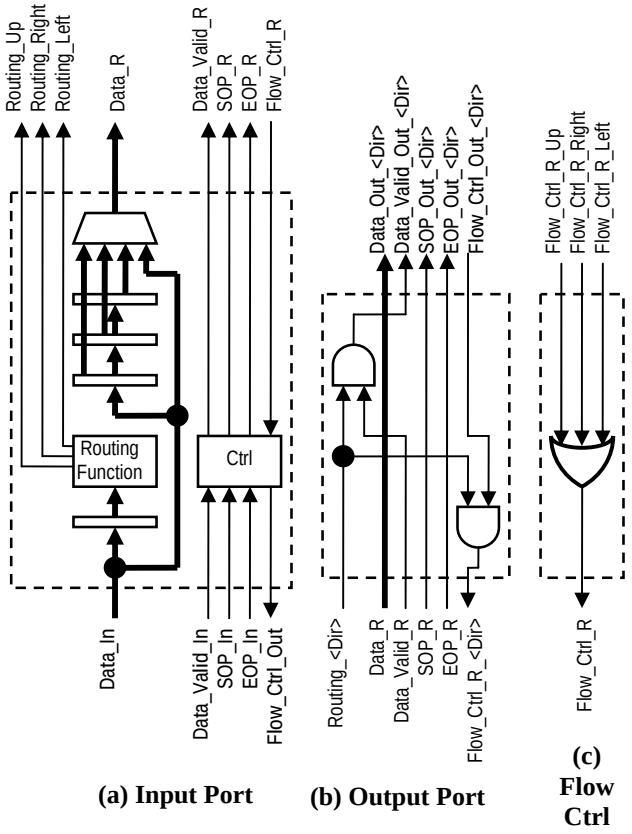


Figure 5: Port Structures

V. CONCLUSIONS

A simplified router architecture has been presented in this paper. Its sheer simplicity comes from the contention elimination which exempted the router from the need of buffering structures to accommodate packets when contention occurs. It also comes from the simple routing function circuitry obtained from a formal determination of the routing function itself. Gate hungry magnitude comparators have been eliminated, compared to the preliminary architecture presented in[1]. This has tremendously reduced the gate count so that a first trial synthesis when writing this paper resulted in a gate count of 3200 nand-equivalent gates for a router that has 16 inputs and 32 outputs. A parameterized router has been created for the purpose of carrying out functional simulations. This model is used by a network generator that produces *Verilog* files containing the network description of a user selectable size. In the future, a more pragmatic approach will be followed in addressing the issue of the number of wires accumulated at the edge of the network (client interface). This approach will aim to provide the user with the necessary tools to trade-off some of the performances (in terms of throughput) versus less wires at the client interfaces.

ACKNOWLEDGMENT

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

REFERENCES

[1] 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.

[2] 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 ,to appear, Dec. 2007.*

[3] L. Benini and G. D. Micheli, "Networks on chips: A new SoC paradigm", *IEEE Computer*, 35(1):70 – 78, January 2002.

[4] W. J. Dally and B. Towles. Route packets, not wires: On chip interconnection networks. In *Proceedings of the 38thDesign Automation Conference*, pages 684–689, June 2001.

[5] E. Nilsson. "Design and Implementation of a hot-potato Switch in a Network on Chip". Master's thesis, Royal Institute of Technology, IMIT/LECS 2002-11, Sweden, June 2002.

[6] 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

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

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

[9] 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