**International Journal of Advanced Research Methodology in Engineering & Technology,**
***Volume 1, Issue1, December 2016***

# A Guide For Building Simulators For Butterfly And Fattree Architectures

**Arjun R.C[1], Megha Mohan[2] , Anchu Jossy[3], Akshai George[4] and Dr. Sanju .V[5]**
1Student, Muthoot Institute of Technology and Science, Varikoli, Puthencruz- 682 308, India
2 Student, Muthoot Institute of Technology and Science, Varikoli, Puthencruz- 682 308, India
3Student, Muthoot Institute of Technology and Science, Varikoli, Puthencruz- 682 308, India
4Student, Muthoot Institute of Technology and Science, Varikoli, Puthencruz- 682 308, India
5Asso. Professor, Muthoot Institute of Technology And Science, Varikoli, Puthencruz- 682 308, India

**ABSTRACT**

To meet the growing computation-intensive applications and the needs of low-power, high-performance systems, the number of computing resources in single-chip has enormously increased, because current VLSI technology can support such an extensive integration of transistors. By adding many computing resources such as CPU, DSP, specific IPs, etc. to build a system in System-on-Chip, its interconnection between each other becomes another challenging issue. As an improvement on SoC, a new paradigm has been introduced called NoC. NoC is an intercommunication based network system, which is implemented on an integrated circuit, typically between IP cores in a SoC. With the development of IC technology, SoC fails to meet the increasing requirement of network communication since it is based on traditional bus architecture. With the transplantation of network technology from computer systems and the replacement of traditional bus structure with network structure, which solves the communication bottleneck issue of SoC and is rather promising. This paper is about the design and development of a simulator for NoC architectures. More specifically butterfly and fattree architectures.

**Key Words:** NoC; Soc; Butterfly,Fattree

## 1. INTRODUCTION

Now a days, the requirement of network communication is increasing day by day because, the number of modules on a chip multiplies and as a result VLSI technology declined to a greater extend. Some developments were made on VLSI technology so as to cope up the requirement and a new technology was built named SoC where they aimed at bringing entire system to a single chip. But later, researches on VLSI gave a new idea of bringing a number of transistors to a single silicon chip and SoC fails to meet the growing complexities. As a result a new paradigm was introduced named NoC (network on Chip), where it is for networks rather than bringing entire system to a chip. NoC technology is often called “a front-end solution to a back-end problem.” As semiconductor transistor dimensions shrink and increasing amounts of IP block functions are added to a chip, the physical infrastructure that carries data on the chip and guarantees quality of service begins to crumble. Many of today's systems-on-chip are too complex to utilize a traditional hierarchal bus or crossbar interconnect approach. NOC architectures are based on packet-switched networks.

## 2. NODE ORGANIZATION AND WORKING

This section discusses the internals and working of a node in any of the topology. The basic building block of the topology is a node which is composed of interfaces, through which it interacts with the adjacency node.

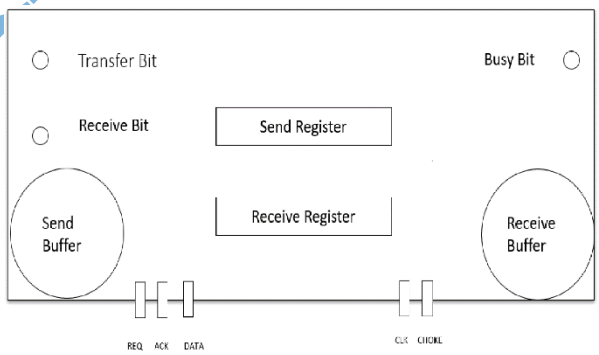



Fig1:interface design

In the above fig, it shows the basic internal structure of node where each node in a butterfly topology consists of the following.

• Receive register: This is to receive the packet after acknowledgement is send.
• Send register: this is used to transmit the packet once it receives the acknowledgement.
• Routing logic: This contains the routing algorithm
Which routes the packet from source to destination through the shortest route.
• Control logic: All the internal working is controlled by
it by generating appropriate control signal.

• Busy bit: This status bit is set whenever the interface is sending or receiving data.
• Transfer bit: It refers that the interface is transmitting packet.
• Receive bit: This bit is set when the interface is given

Acknowledgement and is receiving packet.

## 3. PROPOSED WORK

### 3.1. Butterfly Architecture

Following figure depicts a typical butterfly node.

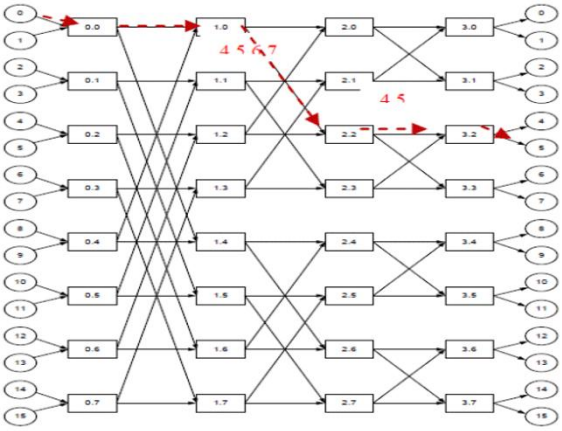


Fig 2: Butterfly Architecture

A butterfly topology is a multistage network on chip topology, where clients are placed in both end of the network and switches are placed in middle. It contains double the number of clients using same number of switches and reduces chip area consumption. In butterfly topology routing is unidirectional, so it uses deterministic routing. The disadvantage of butterfly topology is that it does not have path diversity and uses long wirings

### 3.2 Fat tree Architecture

Following figure depicts the fattree topology:

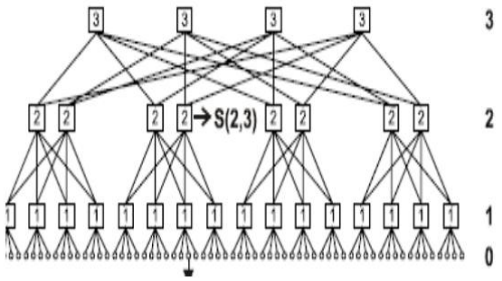



Fig 3:Fattree topology

A tree based topology is proposed by Charles E. Leiserson in known as fat tree. Fat tree is highly scalable and the most efficient network topology in terms of scalability and performance. Also it is efficient in terms of hardware use. As shown in Figure 3 clients are in the leaf node. The main disadvantage of fat tree is that it uses large number of switches for few clients and these switches consume most of the chip area. Again bandwidth requirement increase when going up closer to the root from leaf.

## 4. NODE DESIGN

The following figures depicts the node design of both butterfly and fattree topologies.

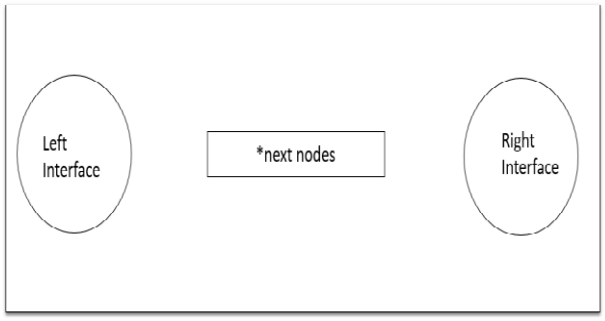



Fig 4: butterfly node

In butterfly topology, a single node contain 2 interfaces namely left interface and right interface where in each interface consists of pins and registers.In this topology each node stores the details of the adjacent nodes. So that routing can be done according to the information stored in the current node.

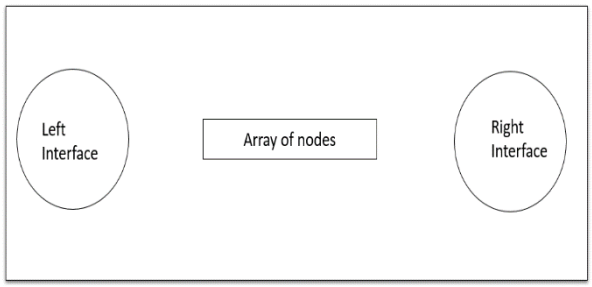



Fig 5: fattree node

Node design of fattree topology is similar to that of butterfly. But the only difference is that it has an array of nodes embedded into each node. This array stores information about the child nodes under alarticular node so that by continuous traversal we can figure out the required nodes.

The design of packet is given below.

```
struct packet
{
int source address;
int dest address;
```

int data;
int start timer;
int end timer;
}

## 5. COMMUNICATION

The intercommunication between nodes takes place via certain protocols. The pins that govern these are given in the interface.

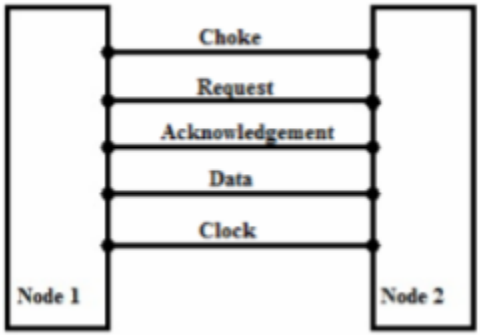



Fig 5: communication between nodes

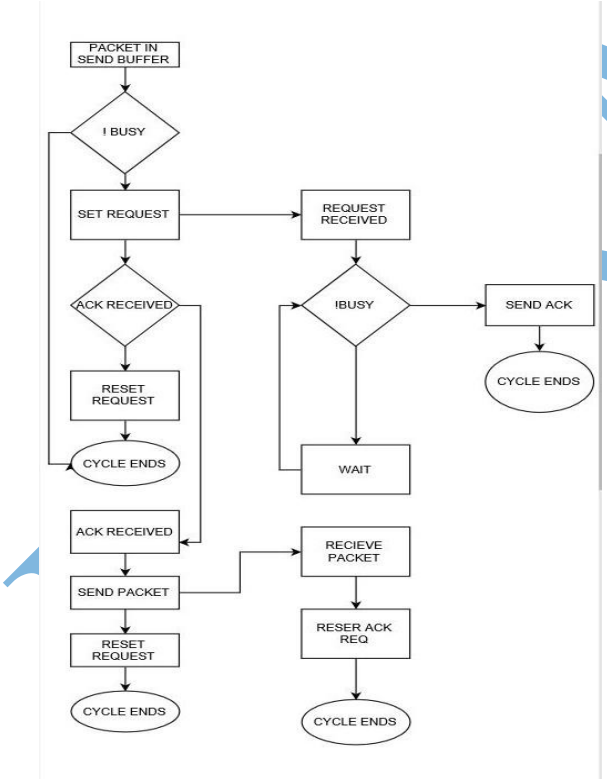
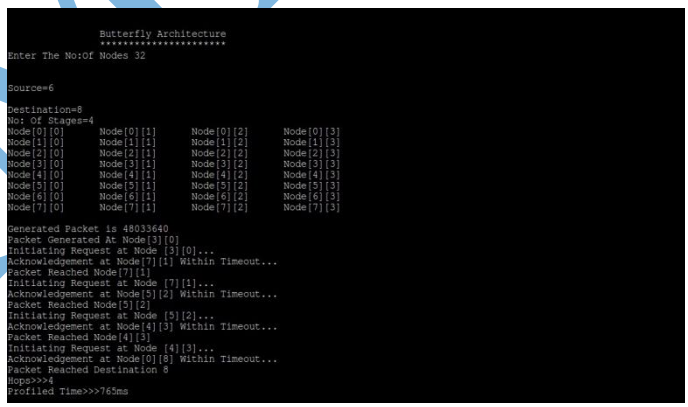



Fig 7: flowchart for communication

The flowchart describes the communication between 2 nodes.At first the node generates a packet and places it in appropriate send buffer. If node1 needs to send data to node2, it sends a request signal to node2. If the request is not acknowledged by any of the interfaces of node2 within timeout the pins are reset. The node2 acknowledges the request of node1 if there is a space in the receive buffer. Then resetting of the request line at the interface and setting of the receive bit and the busy bit takes place. After receiving the acknowledgement , node1 resets its request and places the packet in its send register. The transmission takes place serially through the data pin . Interval clocks are generated simultaneously. At node2 the packet is received serially in the receive register through the data pin .

To integrate fairness in transmission of packets alternately between interfaces,choke bits are used. While receiving data, if the send buffer contains packets then the choke bit is set. The interface checks the choke bit and if it is set it allows the adjacent interface to set the request providing fairness. When full packet is received, the status bits are reset at both the interfaces. Application logic take cares of the packet if it reaches destination else the routing algorithm is performed on the packet to determine the next interface along its destination. Now the cycle continues till the packet reaches the destination and finally the packet is taken out of the system.

## 6. RESULTS



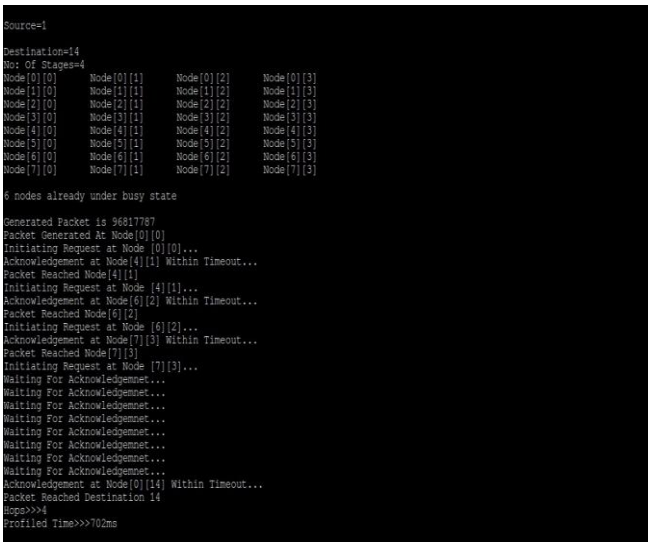




Fig 6: in butterfly topology (busy state)

## 6. CONCLUSION

This paper discusses the challenges of building a new NoC simulator. It also discusses the design and implementation of interfaces and nodes.Efficient designs has to be maintained for nodes and interfaces. Here in this paper we have presented such a design. The interface design can be used for any of the NoC architectures where node designs are specific to concerned architecture ie fattree and butterfly architecture. The communication protocol can be slightly modified so that this can be made to work on multiple core machines and thus can move from single thread of execution to threads running parallel. In short this paper is a guide for those who are building NoC simulators.

## 6. ACKNOWLEDGMENT

I would like to thank Dr. Sanju V for his support and guidance. I would also like to extend our gratitude to Mr.Mahalingam PR. I would also like to thank the Management and Principal of our college for their support.

## REFERENCES

[1] John Kim, James Balfour, and William J. Dally Computer Systems Laboratory Stanford University, Stanford, CA 94305 jjk12, jbalfour,dally@cva.stanford.edu.:Flattened Butterfly Topology for On-Chip Networks

[2] Israel Cidon Dept. of Electrical Engineering Technion, Haifa 32000, Israel cidon@ee.technion.ac.il. Idit Keidar Dept. of Electrical Engineering Technion, Haifa 32000, Israel idish@ee.technion.ac.il.: Zooming in on Network-on-Chip Architectures

[3] William J. Dally and Brian Towles Computer Systems Laboratory Stanford University Stanford, CA 94305 billd,btowles@cva.stanford.edu.:Route Packets, Not Wires: On Chip Interconnection Networks

[4]Sanju V, Koushika C, Sharmili .R, Chiplunkar, N. and Khalid, M. (2014): Design and implementation of a network on chip-based simulator: a performance study, Int. J. Computational Science and Engineering, Vol. 9, Nos. 1/2, pp.95105.