Implementation of an Effective Router Architecture for NoC on FPGA

Sivakamasundari.P
Asst.Prof
Dept.of ECE,

R.Sudha.
Senior.Lecturer
Dept.of Information Tecnology

S.Sasikala
3.Sr.Lect/Dept.of ECE,
Arasu Engineering College
Kumbakonam

ABSTRACT

System on Chip (SOC) designs offer integrated solutions to existing design tribulations in areas which necessitate outsized computation and restriction in certain area. But the performance of these has been sluggish due to the restriction of the common bus architecture espoused by these systems and thereby low processing speeds.

This has been the main drawback for scalability in terms of computation and enhancement in its performance. With the advancement in semi conductor devices and fabrication technology, it is possible to pack more logic in smaller area of silicon. But the implementation of these mega functional modules using common bus architecture, parallel bus architecture, pipelining are becoming ineffective and posing a bottleneck in terms of performance and throughput in this billion transistor era. As a solution for this problem, Network on chip is being adopted in this paper as the core bus architecture across different spectrum of SOCs.

This work presents a simple design using FPGA based system. Hence, it is a very flexible network design that will accommodate to various needs. This router is implemented for four topologies and compared for its speed area and power consumption.

KEYWORDS

on-chip topology; Packetization; Router Architecture.

1. INTRODUCTION

1.1. Need for the System

NOC is a scalable and configurable network that can be adapted to the needs of different workloads while maintaining generality and performance. System on Chip designs cannot respond to the needs and takes too much of time & mapping of applications. Also there is lack of efficient usage of the bus systems in the design. New system would have to rely on reuse of components, architectures, applications and implementations.

NOC design on the other hand is expected to provide good solutions for flexible products that should be reconfigurable and programmable for applications with a heterogeneous task mix for applications with stringent time to market requirements.

There is generally a tradeoff between generality and performance. Performance is essential because generality provides reusability of hardware; performance is achieved by using application specific structures. It brings an increase in efficiency in resource usage and decrease in processing time. The idea of implanting network solutions on silicon is becoming a promising solution for high performance, scalable architecture. Using a network to replace global wiring has advantages of structure, performance, and modularity. Thus, Network on Chip is the solution for the numerous technological, economical and productivity problems[1].

NoC providing a scalable solution for multiprocessors- on-chip (MPoC). In embedded reconfigurable systems, NoCs provide a flexible communication infrastructure, in which links interconnecting processor/DSP cores, memories, and other intellectual property (IP) components can be reconfigured for a certain embedded computing application. NoCs combine performance with design modularity, allowing the integration of many design elements on single chip die [2].

While using the unicast routing the congestion of data can be avoided through the effective algorithms, so that the deadlocks avoided.

2. ARCHITECTURE

Here four architectures had taken, a 2-D mesh, a 2-D Taurus with mxn modules with processing core, a ring with eight processing core and a star with eight processing cores. The data to be processed is dispatched in packets throughout the network. Each of the modules is administered by a controlling unit; the router provides the routing logic. It decides the path which has to be taken by the data to reach the modules. This data bus is utilized simultaneously by all the modules to receive the data and to hurl the result of the processed data. This brings in an element of parallelism into the working of the system. Thus, it results in better performance in terms of speed and enhances the processing time.

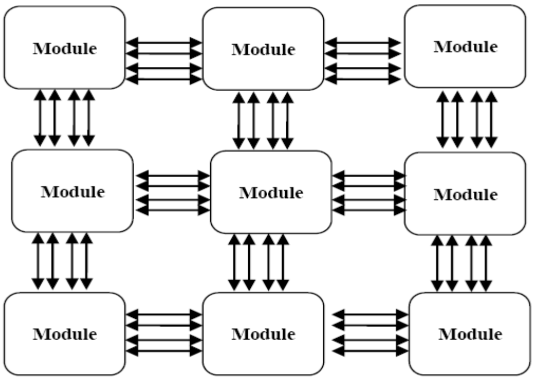


Fig 1: Mesh Topology

Routing in a 2D mesh is easy resulting in potentially small switches, high capacity, short clock cycle & overall scalability. Efficient routing of messages within the network is essential in order to fully exploit the power of the computing resources and achieve good performance for applications running on them. A good routing algorithm should not only provide low latency for messages. This network consists of modules at each node.

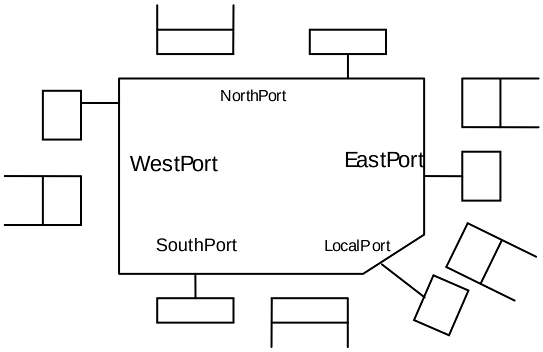


Fig2:Conventional Router for Mesh&Torus

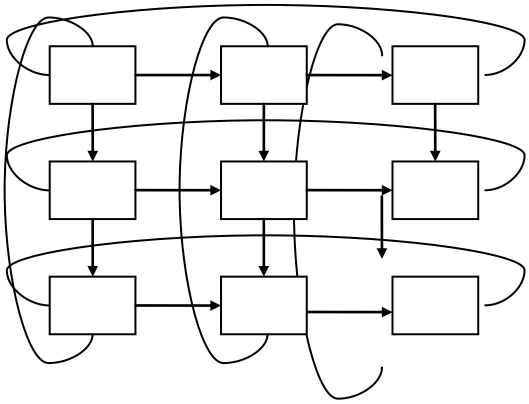


Fig 3: Torus Topology

The inter process communication among different modules takes place by transfer of packets instead of polling or arbitration as in bus architecture. The operations of Network on Chip are assisted by three routing strategies namely store-and-forward, virtual cut-through, wormhole routing which specifies the method on how the router at intermediate nodes processes the packets and forwards to the next node towards the destination.

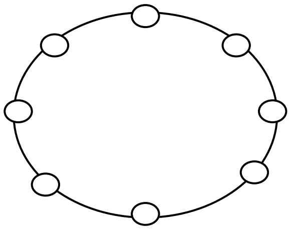


Fig 4: Ring Topology

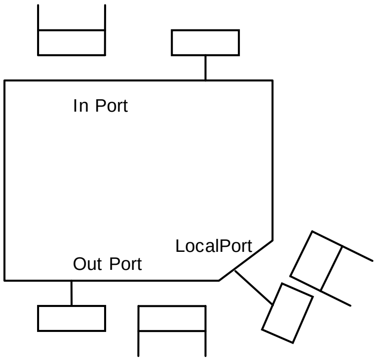


Fig5:Conventional Router for Star&Ring

Fig 2,5 &7 presents the conventional architecture for NoC, which are consist of a FIFO for get request and a synchronous buffer to each port. Because for the usage of the register size of the register will increases with the an increase in requests which causes for higher power consumption and area.

In order to reduce the power consumption and area a new architecture is used here which proposed newly and the idea got from [1] and is shown in fig.5. Each module is connected to four neighboring modules via four different ports. The addressing starts from the top left corner specified as (0,0) and goes towards bottom right as (n-1, n-1) for an N x N module frame work.

Fig: 7 depicts the structure of interfaces inside a module. Each module consists of four interfaces viz. top, down, left and right. These four interfaces simultaneously transmit or receiving a request. The schematic diagram consists of a module with 4 interfaces. These play a vital role in moving the packets to their respective destinations.

A. Routing Algorithm

Typical routing algorithms for Network on Chip systems are designed for a specific network topology and are independent from the application which will be mapped on the Network on Chip.

MS	LS
Dest/Sour.addres	Data

Fig 6: Data Packet

2.1. Packet Format and Address Encoding Scheme

Packetizing helps to cope up with complex SOC communications in which circuit switching is used. In this design, the data to be processed is initially split into packets. The computation is carried out by exchanging packets across the network to different modules. Fig. 6 presents the packet format used in our NoC. The packet consists of a header flit followed by payload flits. Two additional 3-bit heads identify the type and ID (Identity) number of a flit. The flit types can be header, data body, and the end of data body (the last flit).

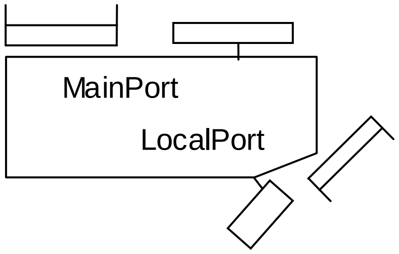
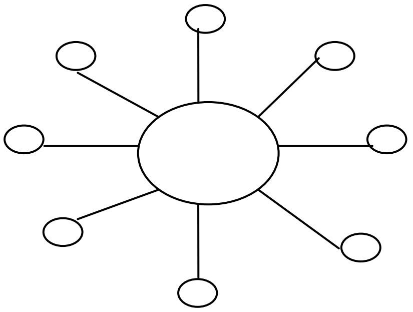


Fig7:Conventional Router for Star Node



In the wormholepacketization model, a message is associated as single packet Hence, for one destination node the message will have only one packet header. Therefore, a packet denotes the same thing as a message in this paper. The packet header will make once a routing direction on each router node. Afterward, payload flits will follow the routing directions made by the header. Therefore, an out-of-order problem can be avoided, even when adaptive routing algorithm is used to route the packet.

The packet starts with the destination address followed by the source address and data. The destination address and the source address of the module is attached with the packets and sent for processing. The size of the source and destination address depends on the number of modules in the system. Modules communicate with each other using address data packets routed for their destination by the interfaces. Thus, the data is checked for their destination address and processed. In this system, XY routing method is followed. The data to be sent is packetized and forwarded in the X-axis direction. If the destination address is not met, the same data is sent in the Y-axis direction.

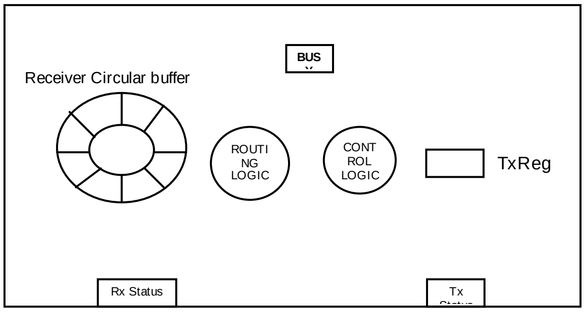


Fig 9: Proposed Router Architecture

This design of packet based routing for transferring data helps in dynamically managing severally communications and processing in parallel.

Fig: 9 explains the internal structure of an interface. Each interface comprises of - two circular queue buffers with a control logic-each for sending and receiving data, known as the temporary send buffer and temporary receive buffer respectively; two temporary registers- for storing the data and transmit them as soon as they reach the interface; two status registers for keeping a check on the status of the temporary registers, a busy bit to showcase the status of the buffer as a whole and a routing logic enables smooth flow of data avoiding congestion between the modules. The control logic is where the busy bit and the status bit of the interface are checked.

2.3. Working

The circular buffers enable in storing a range of source and destination addresses, required by the routing algorithm, for navigating the data packets across the mesh. The control logic controls the flow of data packets into the circular buffers. The involvement of circular buffers facilitates free flow of packets as the oldest address is always replaced by the latest arriving address. In this way, the oldest packet is processed at the earliest.

The temporary registers aid in storing the latest address from the circular buffer before the packet is processed by the routing logic. The status checks for the availability of the temporary register. The routing logic then routes the packets to the respective modules. The busy bit ensures the status of the interface and when set, notifies that the interface is busy.

The pseudo-code given below summarizes the operation of an interface:

```
router_operation ();
        {
        decode_packet ();
        check_address ();
                If (add == dadd)
                { store_packet ();
                }
                Else forward_packet ();
        }
```

In this context, decode_packet, check_address,store_packet, forward_packet are various smaller functions which carry out the specified tasks like decoding a packet for its destination address, checking the destination address in the packet with that of the module address, storing the packet if the destination and the module address harmonize if not forwarding the packet to the next interface. „sadd‟ and „dadd‟ are the variables that account to the address of the module and the destination address that the packet carries. The packets are forwarded to the following interfaces when these addresses are not equal.

As the data are stored in to the receive buffer, the status receive bit is incremented and is decremented whenever the data is deleted for forwarding it to next interface. The same logic holds with status bits of the send buffer also. These bits are also used to find the buffer status.

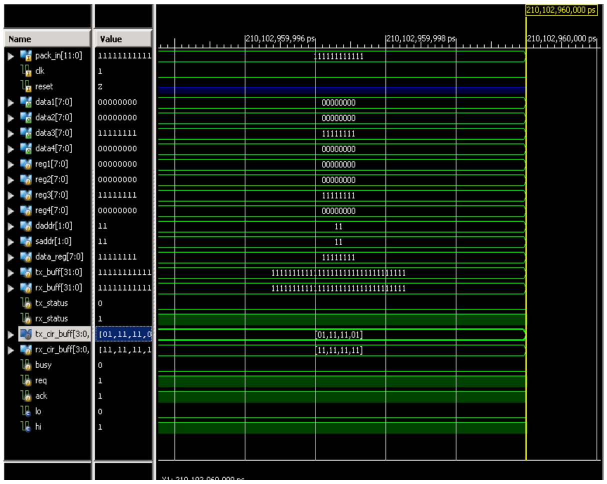
3. SYNTHESIS

The Synthesis of the proposed design has been done using Xilinx and the following is the RTL schematic of NOC design. The figure(fig:9) above is a schematic of a module generated using Modelsim.

Each of these modules is seen containing four interfaces with four input and output ports. This portrays the basic design of the architecture which states that each module is connected with four other modules via four different ports with an interface in each.

4. CONCLUSION

A flexible network design gives the opportunity to easily search the design space for the optimum solution as well as to quickly develop the communication infrastructure. The project work would be used as a plug-n-play module for creation of high performance systems. When compared to the common bus architecture network on chip based systems is proved to be much effective in terms of speed, performance and through-put.



Name	Value
pack_in[11:0]	11111111111
clk	1
reset	Z
data1[7:0]	00000000
data2[7:0]	00000000
data3[7:0]	11111111
data4[7:0]	00000000
reg1[7:0]	00000000
reg2[7:0]	00000000
reg3[7:0]	11111111
reg4[7:0]	00000000
daddr[1:0]	11
saddr[1:0]	11
data_reg[7:0]	11111111
tx_buff[31:0]	11111111111
rx_buff[31:0]	11111111111
tx_status	0
rx_status	1
tx_cir_buff[3:0,	[01,11,11,0
rx_cir_buff[3:0,	[11,11,11,1
busy	0
req	1
ack	1
lo	0
hi	1

Fig. 10 Simulation Result

5. REFERENCES

[1] Shubha B C , Srikanta P "FPGA Implementation of NoC Framework using HDL ", Proceedings of the 2010 IEEE Students' Technology Symposium 3-4 April2010, IIT Karagpur.

[2] William J. Dally , Brian Towles, "Route Packets Not Wires :on chip interconnection network", DAC 2001, June 2001.

[3] J. Nurmi: Network-on-Chip: A New Paradigm for System-on-Chip Design. Proceedings 2005 International Symposium on System-on-Chip, 15– 17 November 2005.

[4] Dally, William J., and Poulton, JohnW.,"DigitalSystemsEngineering,Cambridge University Press, 1998.

[5] Seitz, Charles, "Let's Route Packets Instead of Wires." Advanced Research in VLSI: Proceedings of the Sixth MIT Conference, 1990, pp. 133-138.