Efficient Design of Routing Node to Evaluate the Performance of Network Based Communication Infrastructure for SOC Design

Nilesh A. Mohota[1] and Sanjay L Badjate[2]

[1] *Department of Electronics Engineering, J D College of Engineering, Nagpur, India*
[2]*Department of Electronics Engineering, S. B. Jain Institute of Technology Management & Research*
nileshmohota@gmail.com, s_badjate@rediffmail.com

Abstract

The current trend in technology has lead to the emergence of complex Systems-on-Chip (SoC). Traditionally, shared busses were used for communication between the different components in an SoC in which a communication link is shared between components in a time-division fashion, resulting in a communication latency. To overcome the limitations of common bus based design we have proposed Network-On-Chip based SoC architecture. The aim of this work is to present a modified architecture of the routing node to achieve higher area and power efficiency using changes at the RTL architecture level. FPGA implementation of 4x4 Router has been performed on Xilinx Spartan-3 FPGA XC3S400. ASIC implementation has been done using Design Compiler and IC compiler of SYNOPSYS with 90 nm SAED technology library. It was found that the proposed router has latency of 4 clock cycles, occupies 0.2 sq. mm of silicon area and operates at 500 Mhz frequency.

Keywords: *AHB, ASIC, NoC, NI (network interface), PE (processing element), SoC*

1. Introduction

1.1. Motivation for Network-on-chip based System-on-chip Architecture

The current trend in technology has allowed an ever increasing number of circuits to be placed on a wafer of silicon. This has lead to the emergence of complex Systems-on-Chip (SoC) where entire systems consisting of analog as well as digital components are being implemented on a single chip. Traditionally, shared busses were used for communication between the different components in an SoC [1]. In a shared bus architecture (Figure 1(a)), a common communication link is shared between components is a time-division fashion, resulting in a communication latency that increases with the number of components sharing the bus. An improvement to the shared bus approach is the hierarchical bus (Figure 1(b)). This architecture consists of several shared busses interconnected by bridges to form a hierarchy. SoC components are placed at appropriate levels in the hierarchy according to the performance level they require. This decreases the load on each bus and improves performance. A switched interconnects providing more than one parallel point to point link is a more efficient option offering higher performance [2]. Configurable on chip interconnects such as Wishbone offer a switched interconnects option (Figure 2). Off-chip networks have shown that all of these approaches have limited scope for scalability. The most successful off-chip system interconnect has proved to be the network. Hence, a natural progression of the switched interconnect is to employ a network design based on a number of small switching

components inside an SoC [3]. Packet switching can be used in SoCs with an arbitrarily large number of components (resources). The proposed platform would effectively separate the specification of inter-task communication from the implementation of that communication [4]. ASIC [5] and FPGA [6] implementations of packet switched networks on chip have been demonstrated to be viable solutions for the SoC interconnect problem. Hence, as stated in [7], packet switched NoCs are the clear solution to the problem of complex SoC interconnect design (Figure 3).

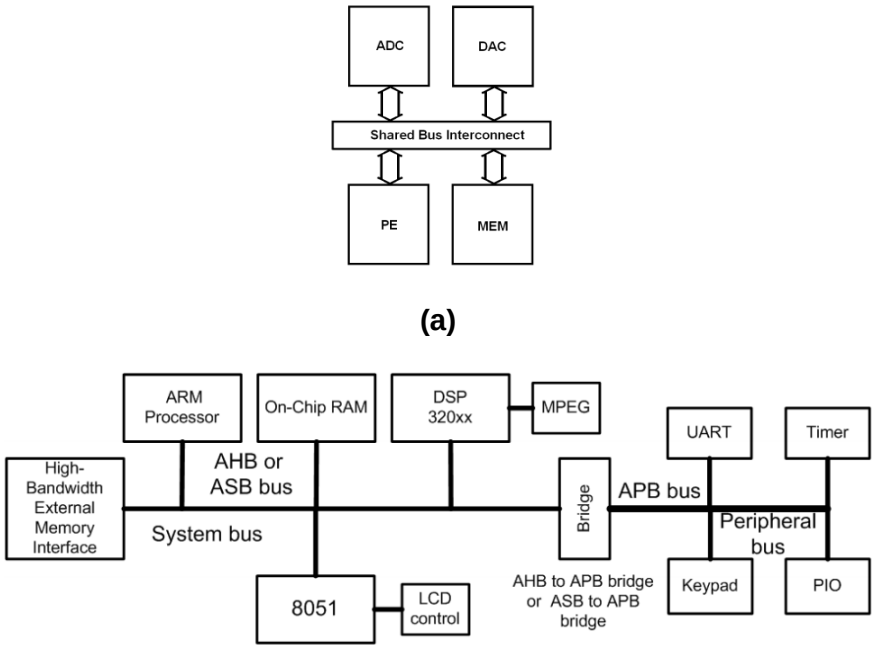


Figure 1. (a) Simple bus-based SoC; (b). Hierarchical bus based SoC

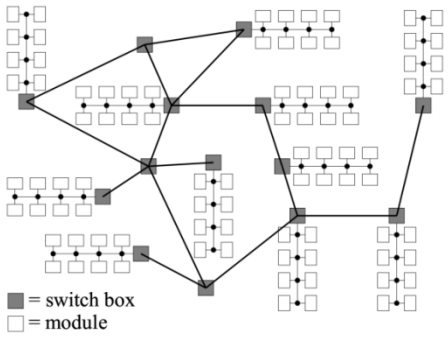


Figure 2. Switched Interconnect based SoC

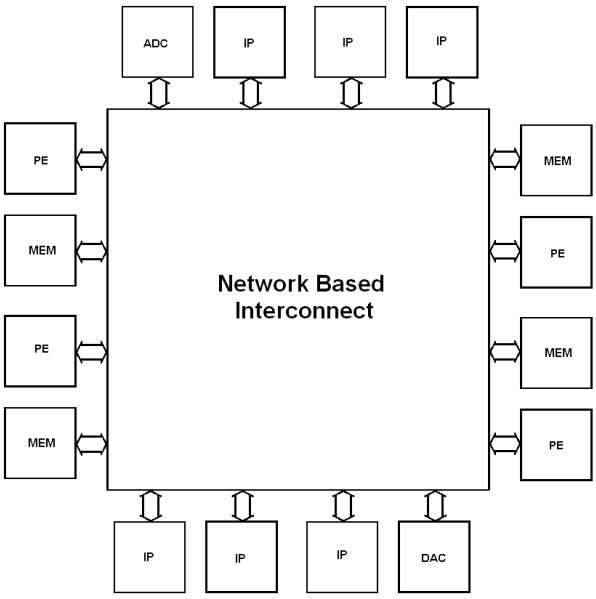


Figure 3. Network based SoC

1.2. Problem Statement

The implementation of networks-on-chip presents certain challenges. Two of the most critical design metrics for networks-on-chip are area requirements and power consumption. Due to the fact that die area per wafer of silicon is limited, the NoC implementation should be carried out using an approach that minimizes area requirement. Also, due to likelihood of most SoCs being implemented in battery powered devices, power consumption of the NoC should also be as low as possible. Usually, reduction in area results in a saving in power requirements due to the fact a smaller area is achieved using fewer components on-chip. Fewer components on-chip will consume less power compared to an architecture requiring more components on-chip.

1.3. Proposed Design and Implementation Task

Given an existing implementation of a routing node for on-chip networks, it is the goal of this work to present a modified implementation of the routing node to minimize the area requirements and as a result lower the power requirement. The routing node consists of four basic components: the input ports, the output ports, the crossbar switch, and the scheduler. The design of the modified routing node is implemented using standard-cell based VLSI flow with provision for custom IP core inclusion. The Synopsys tool chain is used to implement the design from RTL coding to synthesis and place and route. Design verification is carried out using hierarchical functional simulation at each level of the design flow. Also, static timing analysis is used to verify timing closure in the final design layout. A credit debit system is used to control the flow of packets into the input block. Whenever a packet is transferred out of the input block, one memory location becomes available in the packet array. This is indicated by a credit signal that is sent to the input block master, signaling that the input block is ready to receive another packet. When all locations in the packet array are full, there is no credit signal sent to the input block master, preventing arrival of new packets. Packets arrive in two phases of 16 bits each. This is done to prevent simultaneous arrival and departure of packets during the same clock interval. This simplifies the memory design of the

packet array as it allows the packet array memory to be single port. At each clock interval, data will either be only written to the packet array or read from the packet array. The read and write operation to the packet array will not be simultaneous.

2. Literature Survey

2.1. Motivation for Router Design

The router is the most redundant component in the design of Network-On-Chip based SOC architecture. It appears with every processing element along with Network Interface in NoC based architecture. The performance of the Network architecture is based on efficient design of the router. The design of a router also involves determining the flow control techniques, number of VCs, buffer organization, switch design, pipelining strategy while adhering to target clock frequency and power budgets. All these issues require careful design since they have significant impact in terms of performance, power consumption, and area. Accurate performance analysis of on-chip routers under arbitrary input traffic and methodologies for choosing the correct design parameters such as optimal channel width, buffer depth, pipeline depth, and number of VCs for high performance and low power remain open problems [8]. Energy-efficient routers that can interface with a variety of IP cores designed for legacy communication protocols with minimal performance overhead is an important challenge.

2.2. Network -on- Chip

The network-on-chip is a concept to overcome the challenges of intra-system communication in complex future generation systems-on-chip. The NoC concept was first proposed in [9, 10, 11]. The basic idea behind the NoC is to replace all traditional bus-based on-chip interconnects with a packet-switch based network architecture similar to the traditional off-chip network model. However, on-chip networks differ from off-chip networks with respect to certain parameters [12]. Power, area, and latency are the more critical metrics in NoCs compared to off-chip networks. Also, NoC functionality is implemented using simpler hardware compared to off-chip networks where functionality is realized using complex software. Traffic patterns in the case of on-chip networks may be known a priori, which is not the case for off-chip networks. Off-chip networks must adhere to standards, but NoCs can be custom designed to fit a certain application. Link cost is important in off-chip networks, but NoCs have abundant wiring resources available on-chip. Compared to the existing SoC communication model of shared busses, NoCs have a number of advantages, as summarized in Table 1. The main advantage of the NoC over the shared bus is that the NoC is able to scale well as the size of the system increases, whereas the shared bus performance degrades when the size of the system increases, requiring a hierarchical bus approach, which tends to approach a NoC architecture as the hierarchy increases. However, the advantage with bus based systems is that standard architectures are available and easy to implement, whereas with NoC, no standard models are present as it is a concept under research.

Table 1. Shared Bus vs. Network-on-Chip

Shared Bus	Network-on-Chip
Every unit attached to the bus adds parasitic capacitance, degrading electrical performance of the bus and limiting growth of the system.	Only point to point one way wires are used for all system sizes thus local performance does not degrade with scaling up of system.
Bus timing is difficult in a deep submicron process.	Network wires are pipelined because links are point to point.
Bus arbitration becomes a bottleneck because delay grows with the number of masters.	Routing decisions are distributed to each routing node.
The bus arbiter is instance specific.	The same routing node can be instantiated for all network sizes.
Bandwidth is limited and shared by all attached units.	Aggregated bandwidth scales with the size of the network.

2.3. NoC Topologies

Networks-on-Chip have been proposed on various topologies [13, 14, 15, 16]. The point-to-point network is the simplest architecture, but it is only feasible for systems with a small number of blocks. No arbitration or interface design is required. Latency is zero. Bandwidth is maximum. Implementation becomes difficult due to wiring as number of components increase beyond four or five. The bus is suitable for larger systems and is the standard approach for simpler SoCs. Latency is small and bandwidth is high. Hierarchical bus or multibus architecture is common in more complex SoCs with a dozen or more components. The 2D mesh is the most common implementation due to its simple design. The torus implementation offers greater link and router utilization. The folded torus implementation ensures constant link length while maintaining the higher router and link utilization offered by the plain torus implementation. Tree and fat tree implementations have been demonstrated in [9] and [17]. The butterfly topology has been implemented in [18] and used for DVD video decoder applications. A detailed look at various topologies implemented in various research designs is given in [15].

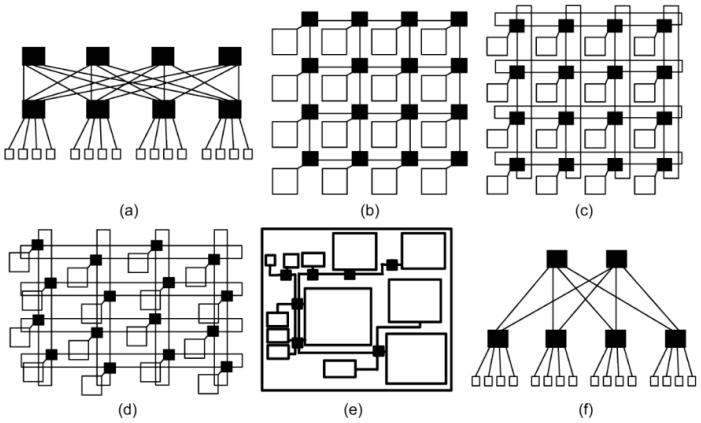


Figure 4. Network on Chip (NoC) Topologies (a) Butterfly (b) 2D Mesh (c) Torus (d) Folded Torus (e) Irregular (f) Fat Tree

2.4. Switching Policy

Packet switching is the more common switching policy used in networks-on-chip [15, 16]. There are three choices regarding how packets are forwarded and stored at routers: store and forward, virtual cut-through, and wormhole switching. Figure 5 illustrates the classification of switching policy. Store and forward method waits for the whole packet to arrive at the router before a routing decision is taken. Store and forward is the easiest policy in terms of implementation complexity. Virtual cut-through performs the routing decision as soon as header information is available. Both methods require router buffering capacity equal to one full packet at minimum.

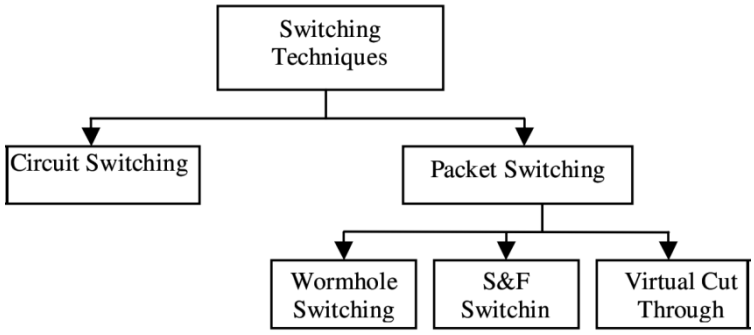
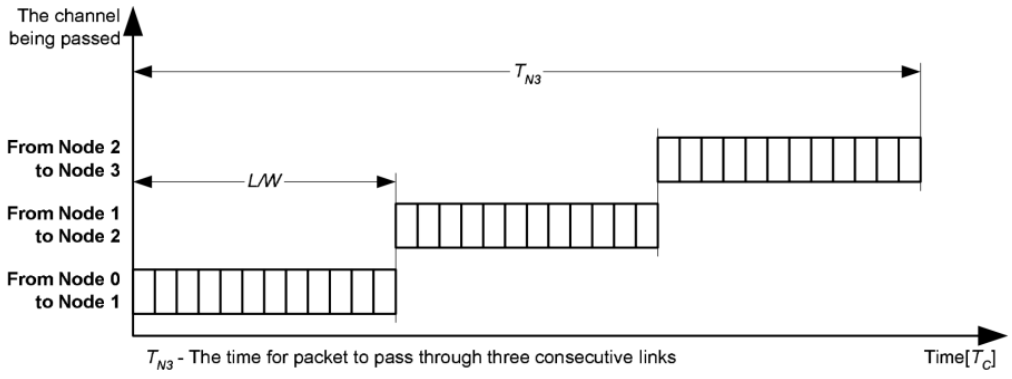
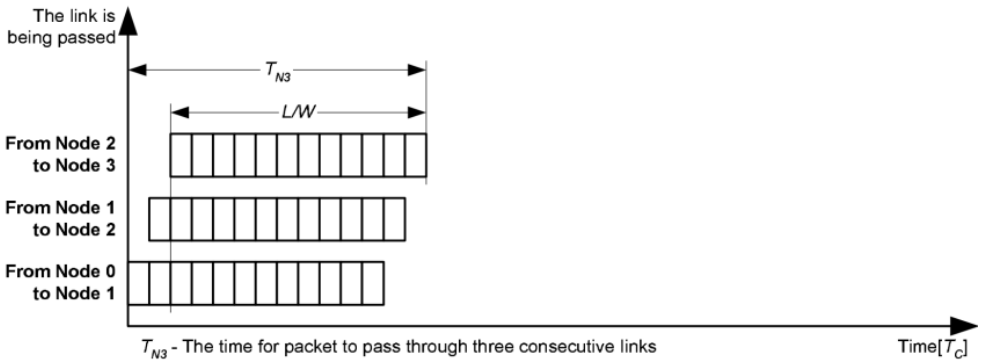


Figure 5. Switching Techniques

Wormhole switching is the most popular switching technique for NoCs. It splits packets in several smaller flow control units (called flits). Routing is done as soon as header information is available, similar to virtual cut-through, but due to the smaller flit size, router buffering capacity required is much smaller. Figure 6 illustrates data transfer using the three switching techniques. Table 2 presents a comparison between the various switching techniques. Figure 7 illustrates the formation of flits and flit composition. Figure 8 illustrates packet delivery in the form of flits by pipelining.



(a)



(b)

Figure 6. Data Transfer based on Switching Policy (a) Store-and-forward (b) Wormhole

Table 2. Switching Policy Comparison

Protocol	Per Router Cost		Stalling
	Latency	Buffering	
Store- and-forward	Packet	Packet	At two nodes and the link between them
Wormhole	Header	Header	At all the nodes and links spanned by the packet
Virtual-cut-through	Header	Packet	At the local node

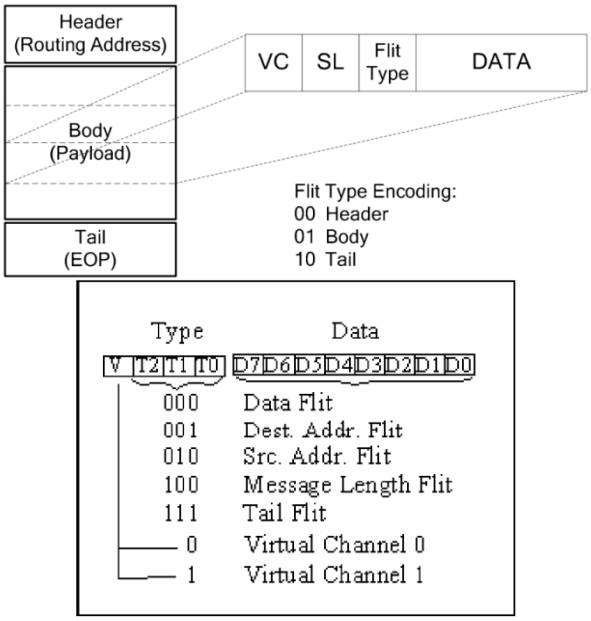


Figure 7. Flit Composition

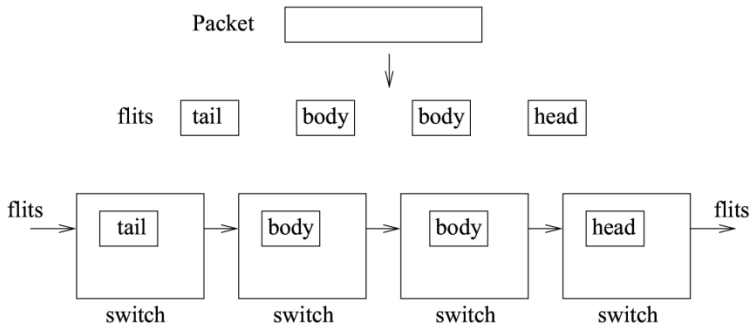


Figure 8. Flits Delivered in a Pipeline

Wormhole switching is the most common switching policy for NoCs [15, 16]. This is due primarily to the smaller buffer size requirements at each router. At the system level, other NoC parameters such as routing algorithms, flow control, and QoS (quality of service) are important design attributes. Figure 9 summarizes the different flow control mechanisms in use for NoCs. These are discussed in more detail in [15] and [16].

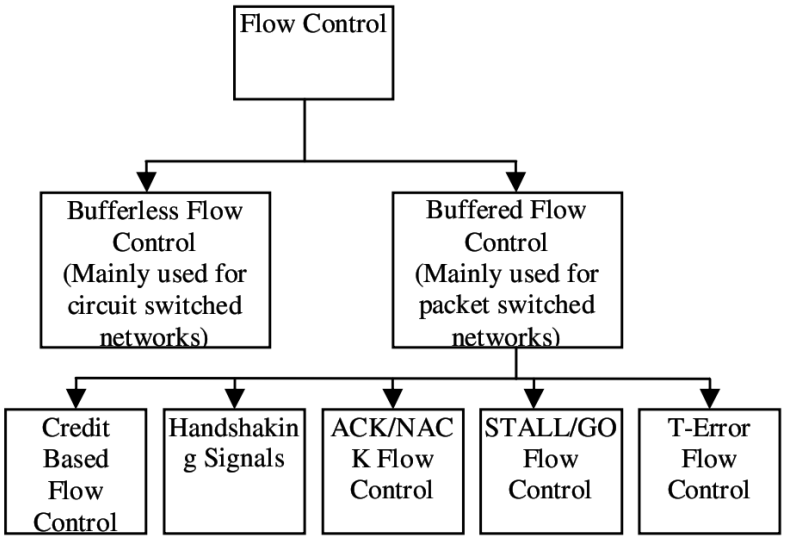


Figure 9. Flow Control Techniques

3. Introduction to Generic Network on Chip Architecture

Traditionally, the design space exploration for systems-on-chip (SoCs) has focused on the computational aspects of the problem at hand. However, as the number of components on a single chip and their performance continue to increase, the design of the communication architecture plays a major role in defining the area, performance, and energy consumption of the overall system. Furthermore, with technology scaling, the global interconnects cause severe on-chip synchronization errors, unpredictable delays, and high power consumption. To mitigate these effects, the network on-chip (NoC) approach emerged recently as a promising alternative to classical bus-based and point-to-point (P2P) communication architectures. Aside from better predictability and lower power consumption, the NoC approach offers greater scalability compared to previous solutions for on-chip communication. As shown in Figure 10, modern SoC architectures consist of heterogeneous IP cores such as CPU or DSP

modules, video processors, embedded memory blocks, etc. Each such processing element (PE) is attached to a local router which connects the core to the neighboring nodes via a NoC. Research challenges involved in the implementation of NoCs are topology synthesis, channel width, buffer sizing, floorplanning, routing, switching, scheduling, and IP mapping. All of these challenges result in area, power, and performance tradeoffs [19]. NoC research is aimed at addressing these design issues while optimizing the implementation for area, power, and performance. Area and power can be estimated from hardware requirements. Performance is generally estimated using analytical models.

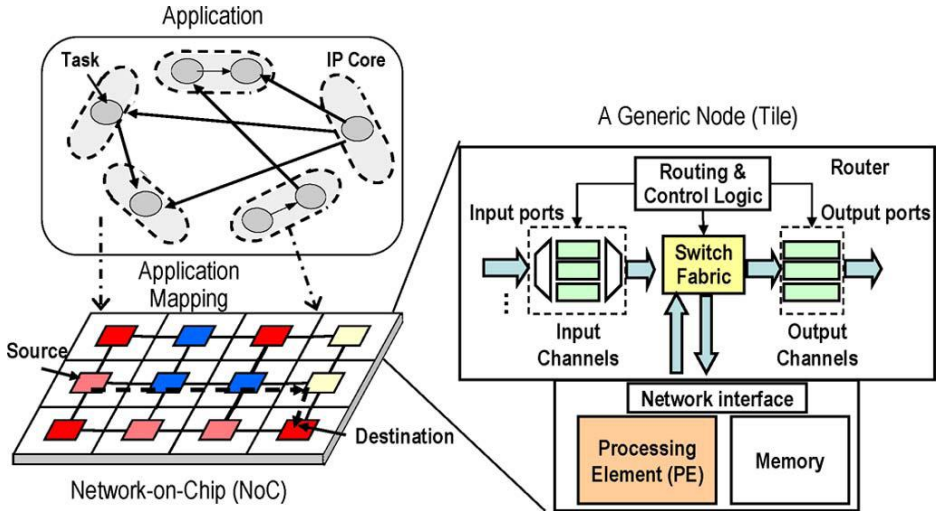


Figure 10. Generic Network-On-Chip Architecture

3.1. Routing Node

The routing node is one of the three basic components of the network-on-chip. The function of the routing node is to transfer data from an incoming port to an outgoing port depending upon the destination address of that particular packet of data. The main components of the routing node are depicted in Figure 11. The input block is responsible for storing incoming packets until they are ready to travel to their respective output block. The crossbar is the shared medium over which packets are routed from input blocks to output blocks. The scheduler arbitrates between requests for access to the shared crossbar switch by packets stored in the input blocks. The routing node configuration as shown in Figure 11 is 4x4. It is based on a 2D mesh NoC topology where each routing node is connected to four other routing nodes.

3.2. Input Blocks

The input blocks store packets in the form of queues, which form a FIFO structure. The simplest implementation is to maintain one FIFO queue at each input block, as shown in Figure 12. This is referred to as input queuing. The disadvantage of input queueing is a phenomenon called “head of line blocking” [10]. Head of line (HoL) blocking is illustrated in Figure 13. Consider the FIFO queue in the first input block. Packet 1 of this queue is blocked due to an arbitration decision allowing FIFO queue at the second block to transmit to the second output block.

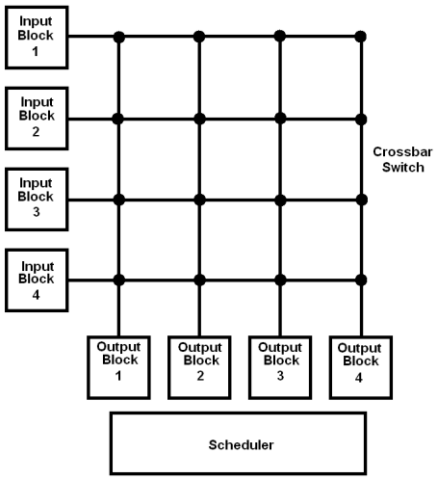


Figure 11. Routing Node

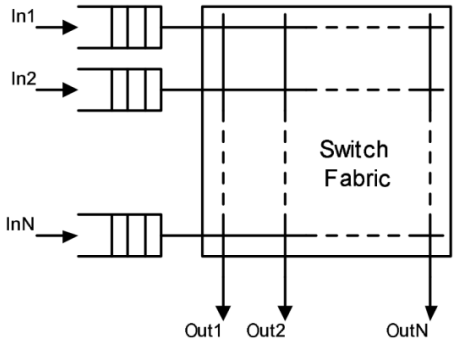


Figure 12. Input Queuing

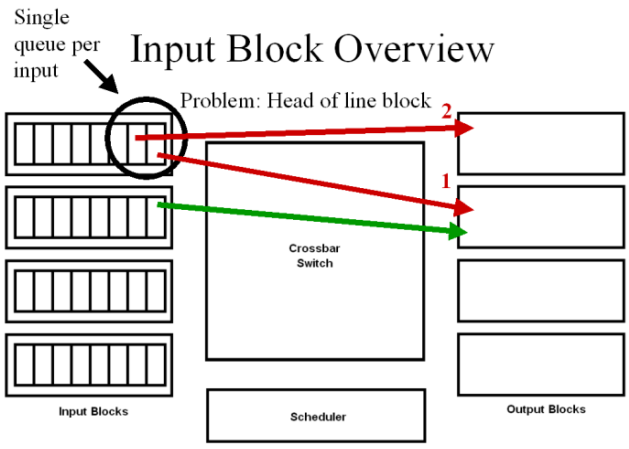


Figure 13. Head of Line Blocking

However, even though the arbitration decision does not block access to packet 2 to travel to the first output block, packet 2 is blocked by packet 1. This is head of line blocking, which is a shortcoming of the queuing strategy and not of the arbitration strategy. Head of line blocking can be eliminated by adopting the concept of virtual output queuing. In this configuration, each input block maintains a separate queue for each output block, instead of maintaining one common queue for all output blocks. This prevents head of line blocking and improves throughput across the crossbar switch. Output queuing is illustrated in figure 14.

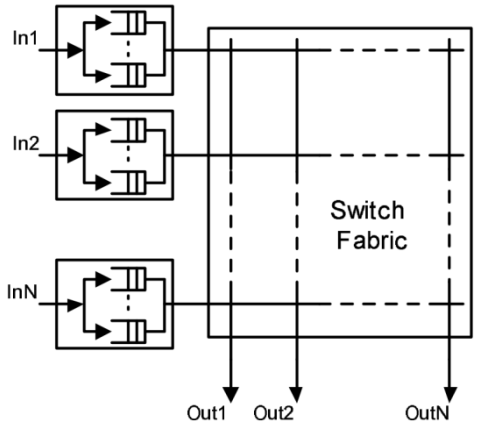


Figure 14. Virtual Output Queuing

4. Routing Node Simulation Results and FPGA Implementation

4.1. Simulation Results of a 4 x 4 Routing Node

The simulation results for the routing node in a 4x4 configuration with 32-bit size packets are shown in Figure 15. Consider the arrival of the packet on the input port named input0_packet[15:0]. The packet data are 16'h02F7 and 16'84C5. From the MSB it is clear that the destination address is output port 0. This data appears on the output port named output0_packet[15:0] as 16'h02F7 and 16'h84C5. Two other packets are transferred across the crossbar during the same interval. They are data packets from input port 2 to output port 3 and from input port 3 to output port 1. A total of six clock cycles pass between the arrival of the last packet at the input and the arrival of the first packet at the output. One clock cycle is required to generate the request vector, four clock cycles are required to arrive at a scheduling decision, and one clock cycle is required to configure the crossbar for data transfer and latch the incoming data to the output port. The FPGA prototype can use to illustrate the impact of application-specific long-range links on the performance and energy consumption of 2-D mesh networks. The adaptive system-on-chip (aSoC) architecture supports compile-time scheduling for on-chip communication and provides software-based dynamic routing. Both architectural aspects and circuit-level techniques can be addressed for practical NoC implementation.

4.2. FPGA Implementation of Routing Node

FPGA implementation of 4x4 Router has been performed on Xilinx Spartan-3 FPGA XC3S400. The top level schematic of the 4x4 Router is given in Figure 16 and The Device Utilization Summary of the 4x4 Router is given in Figure 17. The design of the 4x4 router has been implemented on XC3S400-5pq208. The device utilization summary indicates that 50 IOBs out of 141 IOBs, have been used.

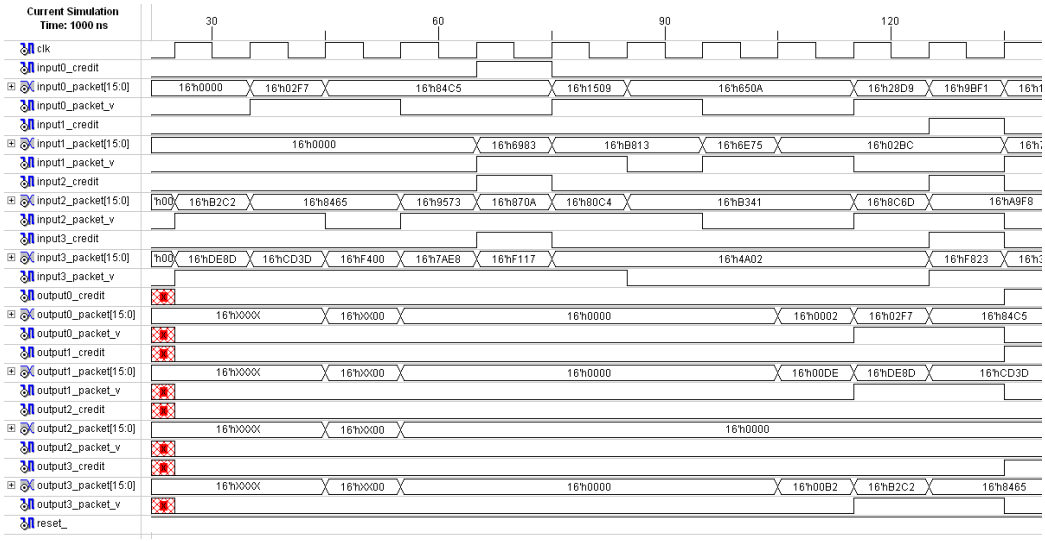


Figure 15. 4x4 Routing Node Simulation Waveforms

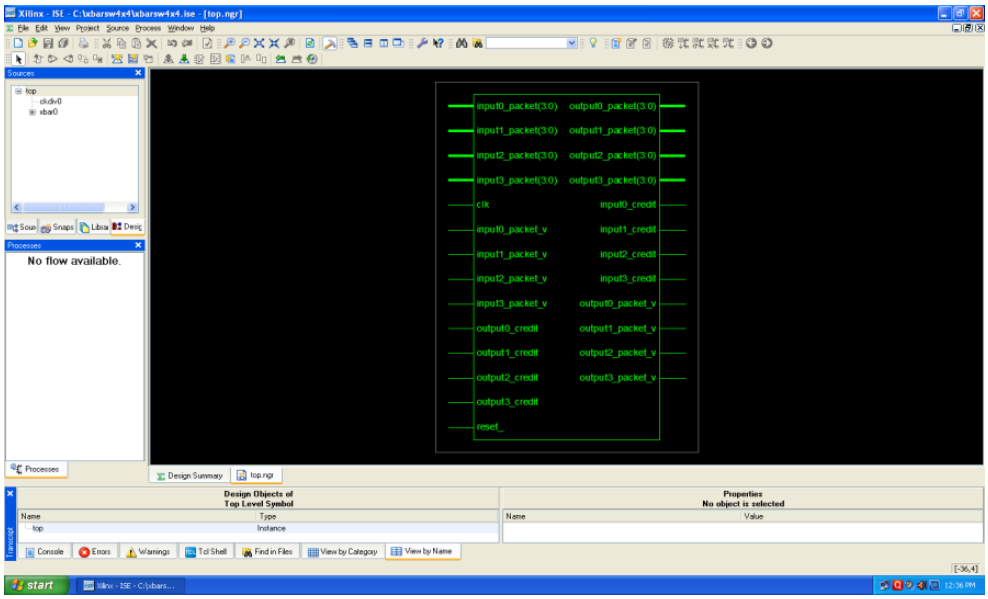


Figure 16. Top Level Schematic of the 4x4 Router

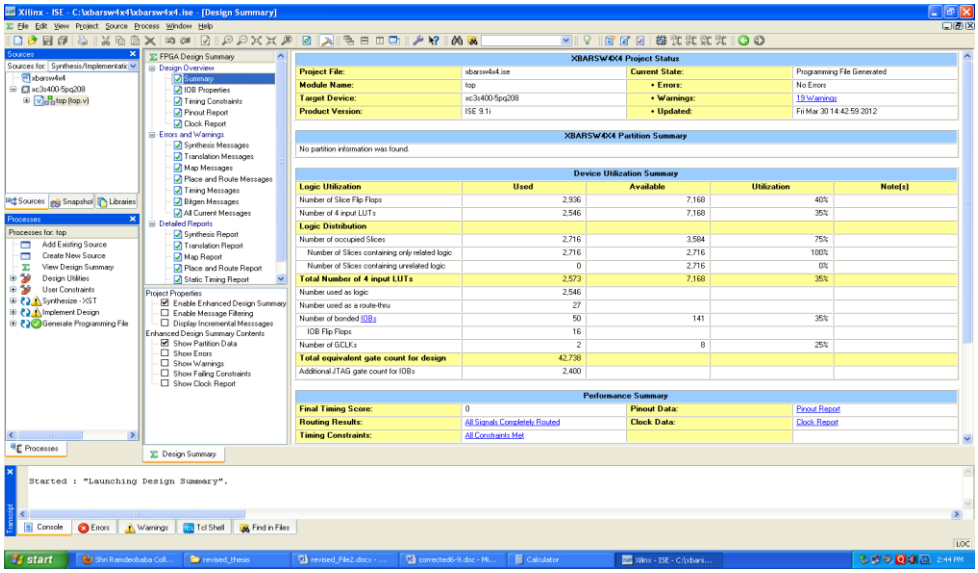
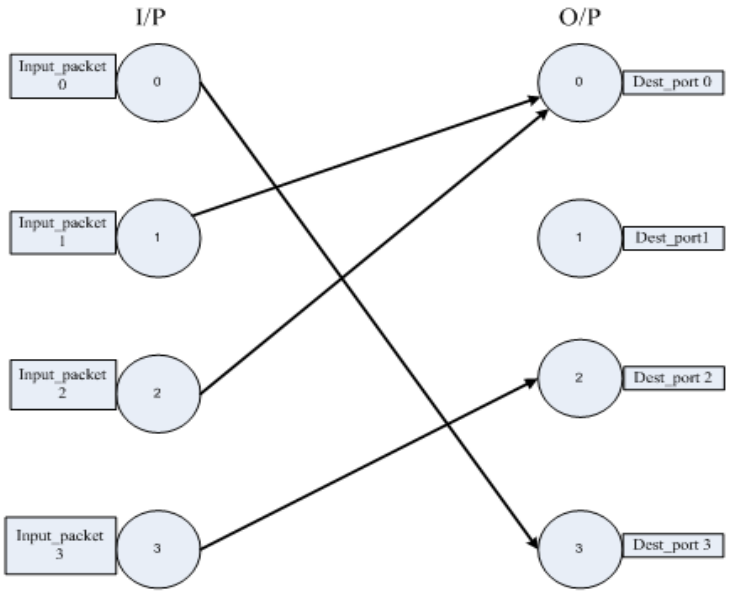


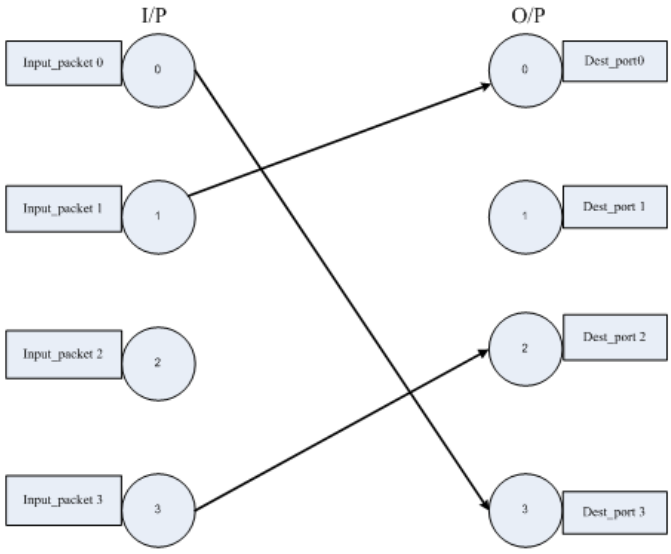
Figure 17. Device Utilization Summary of 4 x 4 Router

4.3. Testing Scheme for FPGA

As shown in Figure 18(a), for input_packet0, destination port is Dest_port3, for input_packet1 & input_packet2, destination port is Dest_port0, for input_packet3 destination port is Dest_port2. After routing, the input packets will reach to their destination port same as that mentioned in Figure 18(b). The above discussed testing scheme has been implemented on FPGA device.



(a)



(b)

Figure 18. Testing Scheme

4.4. The Hardware Setup to Implement the 4x4 Router

The 8-bits version of the input block was used for implementation of the interconnect on a Spartan-3 FPGA as the 32-bit version could not be accommodated on this FPGA.

Figure 19. Implementation on FPGA

The input packets are generated using DIP switches. The input packet size is 8 bits. They are generated in a two-phase manner, with four bits in the first phase and four bits in the next phase. The packet format is as follows

S1 S0 D1 D0 P3 P2 P1 P0

Bits S1S0 specify the packet source from the four possible values (00, 01, 10, 11). Bits D1D0 specify the packet destination in a similar fashion. Bits P3P2P1P0 form the four-bits payload data. In the FPGA implementation of the router, only bits D1D0 are assigned using DIP switches. Depending on the destination address, packet arrival at the corresponding output port is shown by the appropriate LED using the output0_packet_v, output1_packet_v, output2_packet_v, and output3_packet_v signals. With respect to Figure 18(b), destination addresses of two packets are specified as follows – for input_packet3, the destination port is 2; for input_packet0, the destination port is 3; for input_packet1 and input_packet2, the destination port is 0, see testing scheme in Figure 18. Due to successful completion of routing, output ports 3, 2, and 0 show packet arrival. For more clarification, DIP switches are used to set the destination address. LED_R1 indicates the selected destination port. Starting from left to right, first two LEDs show '10' meaning destination port 2 has been selected. LED 3 & 4 show '00' meaning destination port 0 has been selected. LED 5 & 6 show '11' meaning destination port 3 has been selected.LED_R2 indicates the arrival of data packet. When the data packet is received at the destination port, corresponding 'output_packet_v' signal is activated. Here, first four LEDs (starting from left to right) correspond to four 'output_packet_v' signals. In Figure19, LED3, LED2 & LED0 are glowing meaning data packets have been arrived at the corresponding destination port.

Input packet format for source '0'

0 0	1 1	X X X X

Input packet format for source '1'

0 1	0 0	X X X X

Input packet format for source '3'

1 1	1 0	X X X X

4.5. ASIC Implementation of Routing Node

The RTL coding of the design was verified for functionality by functional simulation using Synopsys VCS. The functional simulation approach was hierarchical. Each unit of the design was coded into a separate module. Each module was tested for functionality independently. The top level module was tested using BFM test benches. The RTL code was converted to a gate level net list using the Synopsys synthesis tool Design Compiler. The gate level net list was verified for functionality using the Synopsys static timing analysis tool Primetime. The Synopsys layout tool IC Compiler was used to place and route the gate level net list. STA was performed on the post place and route net list to verify timing closure. The implementation technology was Synopsys EDK 90 nm.

Table 3. Comparative Table

Network Topology	Flit size (bits)	ports	Buffer size (flits)	Tech (nm)	Latency (clock cycles)	Area (sq. mm)	Frequency (Mhz)
Teraflop Mesh	32	4	16	65	5	0.34	4270
ANoC Mesh	32			130		0.25	500
Mango Mesh	32	5	1			0.19	795
Hermes Mesh	32	5	2,8	130	10	0.05 0.11	435
ASoC Mesh	32	4	2	180		0.04 0.08	400
2D Mesh (proposed router)	**32**	**4**	**8**	**90**	**4**	**0.2**	**500**

5. Conclusion and Future Work

The design of the 4x4 router has been implemented on XC3S400-5pq208. The 8-bits version of the input block was used for implementation of the interconnect on a Spartan-3 FPGA as the 32-bit version could not be accommodated on this FPGA. After ASIC implementation, it was found that the proposed routing node works perfectly for 2D Mesh NoC topology with 4 clock cycles latency. It is efficient in terms of area over the some of the contemporary designs of the routing node mentioned in the table. Further optimization is possible in the crossbar switch implementation by implementing a custom crossbar architecture based on pass transistors that is more area efficient compared to the mux based implementation of the present design. The pass transistor implementation would require the analog ASIC flow for generation of the crossbar IP core. Another area of exploration left open is the design and optimization of the network interface (or network adapter, as mentioned in some sources). One possible implementation would be a FIFO based design with additional packet disassembly and assembly logic.

References

[1] M. Mitic and M. Stojcev, "An Overview of On-Chip Busses", Facta Universitatis (Nis) Ser: Elec. Energ., vol. 19, no. 3, **(2006)** December, pp. 405-428.
[2] D. Wiklund and D. Liu, "Switched Interconnect for Systems-on-a-Chip Designs", in Proceedings of the IP 2000 System-on-Chip for a Connected World Conference, **(2000)**, pp. 187-192.
[3] A. Brinkmann, et. al., "On-Chip Interconnects for Next Generation Systems-on-Chip", in Proceedings of the 15th Annual IEEE International ASIC/SOC Conference, **(2002)**, pp. 212-215.
[4] S. Kumar, et. al., "A Network on Chip Architecture and Design Methodology", in Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI'02), 0-7695-1486-3/02, **(2002)**.
[5] T. Henriksson, D. Wiklund and D. Liu, "VLSI Implementation of a Switch for On-Chip Networks", in Proceedings of the International Workshop on Design and Diagnostics of Electronic Circuits and Systems, Poznan, Poland, **(2003)**.
[6] R. Holsmark, A. Johansson and S. Kumar, "On Connecting Cores to Packet Switched On-Chip Networks: A Case Study with Microblaze Processor Cores", in Proceedings of the 7th IEEE Workshop DDECS-04, Slovakia, **(2004)** April.

[7] S. Furber, "Future Trends in SoC Interconnect", in Proceedings of the International Symposium on System-on-Chip, **(2005)**, pp. 183-186.
[8] E. Salminen, et. al., "Survey of NoC Proposals", OCP-IP, **(2008)**.
[9] P. Guerrier and A. Grenier, "A Generic Architecture for On-Chip Packet Switched Interconnects", Design Automation Test in Europe Conference, **(2000)**, pp. 250-256.
[10] W. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks", DAC (Design Automation Conference), **(2001)** June, pp. 684-689.
[11] L. Benini and G. De Micheli, "Networks-on-Chip: A New Paradigm for Systems-on-Chip Design", IEEE Computer, **(2002)** January.
[12] R. Ginosar, "Networks-on-Chip", Technion ESA, (2009) September.
[13] N. Kavaldjiev and G. Smit, "A Survey of Efficient On-Chip Communications for SoCs", Department of EEMCS, University of Twente, Netherlands, **(2003)**.
[14] A. Agarwal, et. al., "Survey of NoC Architectures and Contributions", Journal of Engineering, Computing, and Architecture, vol. 3, Issue 1, **(2009)**.
[15] E. Salminen, et. al., "Survey of NoC Proposals", OCP-IP, **(2008)**.
[16] T. Bjerregaard and S. Mahadevan, "A Survey of Research and Practices of NoC", ACM, **(2006)**.
[17] Andriahantenaina and Greiner, "SPIN: The Scalable Programmable Integrated Network", **(2003)**.
[18] M. Arjomand and H. Sarbazi-Azad, "Performance Evaluation of Butterfly On-chip Network for MPSoCs", 2008 IEEE International SoC Design Conference, **(2008)**, pp. 296-299.
[19] U. Ogras, J. Hu and R. Marculescu, "Key Research Problems in NoC Design: A Holisitic Perspective," in Proceedings of the CODES, ISSS, **(2005)**, pp. 69-74.
[20] C. Wang, et. al., "Area and Power Efficient Innovative NoC Architecture", 18th EuroMicro International Conference, PDP 2010, Pisa, Italy, **(2010)**.

Authors

Nilesh A. Mohota received his B.E. (Electronics Design Technology), completed his M.Tech. (Electronics) and pursuing his PhD from R. T. M. Nagpur University, Nagpur, India. He is an Academician, since last 10 years and presently associated with J D College of Engineering, Nagpur as an Senior Lecturer. He as 5 publications to his credit in International and National Journals. He is a Life Member of ISTE.

Sanjay L. Badjate received his B.E. (Electronics) from R.T.M. Nagpur University,Nagpur and completed M.E.(Digital Tech.) from Devi Ahilya University, Indore in distinction and PhD from Sant Gadgebaba Amravati University, Amravati. He is working as a Vice-Principal at S.B. Jain Institute of Technology, Management and Research, Nagpur. He has 18 publications to his credit in International and National Journals & 5 in International conference.. He is a Life Member of ISTE and Fellow member of IETE. He is guiding 5 students for doctoral degree.