ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse222942020.pdf
https://doi.org/10.30534/ijatcse/2020/222942020

Design and Implementation of Single Node NoC Router using Small Side Buffer in Input Block and iSLIP Scheduler

Priti Shahane[1]
[1]Symbiosis Institute of Technology, Symbiosis International (Deemed University) Pune, India,
pritis@sitpune.edu.in

ABSTRACT

Network on chip (NoC) has been recommended as an emerging alternative for scalability and performance needs of next generation System on chips (SoCs). NoCs are actually forecast to solve the bus based interconnection problem of SoC where large numbers of Intellectual property modules (IPs) are incorporated on a single chip for much better results. NoC provides a solution for communication infrastructure for SoC. The router is a foremost element of NoC which significantly impacts the performance of NoC. The architecture of router consists of an input port with buffers, arbiter, crossbar as well as an output port.The input block of NoC router requires buffers to keep the new data packets. These buffers improve the overall performance of the router in terms of throughput however they consume far more area and power. Bufferless deflection routing will be the solution for improvement in power efficiency, but latency might increase due to unnecessary hopping of data packets. Therefore use of small buffer will help to optimize latency and area in the router design. One more issue of router design is scheduler which allows contention free transfer of data packets among several IP modules or perhaps processors. For starvation free scheduling, Iterative Serial in Line Protocol (iSLIP) scheduler with programmable priority encoder provides best solution. The proposed design of high performance single node router with small side buffer in the input block and iSLIP scheduler has been implemented on FPGA in this paper.

Key words: Network on chip, System on chip, Bus based interconnection, Router, Input buffers, iSLIP scheduler.

1. INTRODUCTION

Today's technology development allows for the incorporation of a large number of memory elements and computing devices on a single chip. System on chip (SoC) emerged as an integrated approach for most applications that require considerable computation and storage requirement. Generally, for such architectures, intercommunication among different cores or perhaps Intellectual properties (IPs) is based on a shared bus. A common feature of bus architecture is that it enables one communication at a time between master as well as slave devices. The standard bus based device has a limitation of scalability and turns into a bottleneck in a complex system. Shared bus for example AMBA bus as well as IBM's core connects becomes overloaded fast when more number of master modules is connected to it. To overcome this problem, Network on chip (NoC) is suggested as a communication subsystem among various IP cores of a SoC. NoC has emerged as an effective and scalable communication centric approach to solving the interconnection problem for such architecture [1]-[3]. The NoC structure consists of 4 major components specifically routers, link and network interface and IP modules. The router is certainly NoC's heart as it manages data packet traverse from source to desired destination port based on routing strategy. It essentially performs two functions according to the switching techniques like protocol conversion and data packet creation. The link provides a channel for data packet transmission among the different network devices [4]-[5].

The router is actually a foremost component in NoC architecture which significantly affects the overall performance of NoC [6]. The architecture of router consists of the input port, scheduler, crossbar, and output port. The performance of router would ultimately depend on the design of input port and scheduler. Nearly all the router designs use buffers to hold flits or data packets until they are transmitted within the network. Buffering lowers drop off and data packet misrouting [7].

The use of buffers increases performance dramatically in terms of much better bandwidth capacity. Yet buffers consume substantial power on the network. Buffer dynamic power will be consumed in the router during read / write operation and even static power will be consumed when buffers are empty. Firstly, buffers consume more area of the chip network and, due to flits allocation and deallocation in the buffers, they also increase design complexity. In the TRIPS prototype chip, 75 per cent of the entire on-chip network area had been occupied by router input buffers. Therefore, there is a bufferless router requirement that prevents buffering in input and output ports [8]. It has been proposed that various bufferless routing algorithms resolve the drawbacks of a buffered router [9]-[15]. The best examples of bufferless routers are the CHIPPER and BLESS [13]-[15]. Yu Cai et al. have shown that bufferless routing saves up to 38 per cent area reduction in mesh or may

be torus topology and 30 per cent power consumption compared to buffer router architecture[15]. Implementation of CHIPPER NoC showed that when design is tested with buffered routing, average network power is reduced by 54 per cent and area by 36.2 per cent for 8X8 mesh topology [13]. The main idea for bufferless routing is simply to eliminate input buffers and use deflection of data packets to address network contention.Bufferless routing provides best solution by reducing area as well as power efficiency for low network traffic, however at high network utilization it gives more deflections which cause unnecessary hopping of data packets. It leads to more power consumption and also reduces performance.

At high network load, Minimally Buffered Deflection Routing (MinBD) overcomes the problem of bufferless deflection routing [12]. This particular router uses small side buffer compared to standard input buffers which is much larger. In case of data packet contention, a MinBD router holds data packets in the side buffer that would usually have been deflected in the network. Thus MinBD router provides solution for energy efficient design with smaller area requirement.

Another aspect of router design is an efficient scheduler design. The scheduler plays a vital role in the NoC router for the scheduling of data packets. Any scheduler (arbiter) has the core function of coordinating competing demands for the same destination. Since the arbiters precisely decide the router's operating speeds, it is vitally important to design fast arbiter. Improper scheduler architecture leads to both data packet congestion and starvation issue. Much work has been done on algorithms for fast scheduling of crossbar switches [16]-[22].
The general approach of this particular paper is to explore a single node FPGA-based NoC router design. The input block of NoC router is recommended with small side buffer to avoid loss of data packet. Furthermore iSLIP scheduler is recommended with programmable priority encoder over round robin arbiter. Section II introduces proposed single node NoC router architecture. Section III details the implementation of each block of NoC router on FPGA. The results are presented in section III which gives optimized area and low latency for single node NoC router. Lastly, section IV concludes summary of the recommended NoC router implementation on FPGA and focus on future work.

2. PROPOSED SINGLE NODE NOC ROUTER DESIGN

Many researchers have been proposed NoC router architecture with input block having buffers in virtual channels and round robin scheduler. The uniqueness of the proposed design is that a router is implemented using small side buffer rather than using bufferes in virtual channels and iSLIP as a fast scheduler. The proposed design of router consists of the following blocks.

- Input block using single side buffer
- iSLIP arbiter
- Cross bar switch

The following figure1 shows the architecture of individual node NoC router. Single node router architecture is having five ports architecture as shown in the following figure 2. Data packets may be transferred in any one of the direction such as North, South, East and West and finally Core port is a destination port of respective node. The design of the input block is suggested in each direction with single memory buffer which only holds deflected data packets in the side buffer memory. Therefore, in case of destination ports contention, the buffer is used to store data packets in a particular direction.

Thus the small side buffer in the router helps to avoid data loss of packets. The next block is iSLIP scheduler which includes programmable priority encoder and it determines the particular direction of data packet transfer. The benefit of iSLIP scheduler is that data packets are directly transferred to the particular port rather than travelling in round robin manner and thus provides fast scheduling.
.

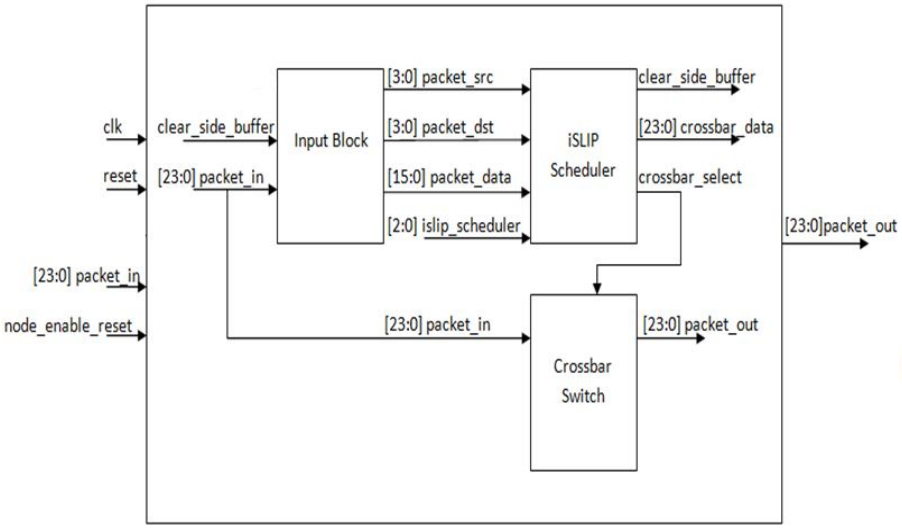

Figure 1: Proposed NoC router block diagram

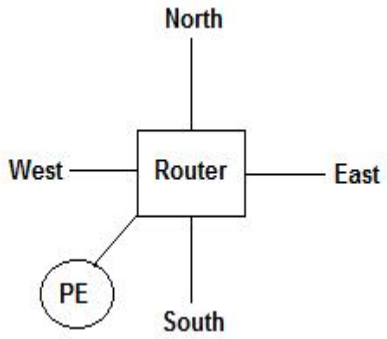

Figure 2: Five port NoC router

3. IMPLEMENTATION OF VARIOUS BLOCKS OF NOC ROUTER

Design as well as implementation of various blocks of the NoC router is discussed in the following sections with results.

3.1 Implementation of input block of router

As discussed in the earlier section, buffers in the input block consumes much more energy as well as chip area while bufferless input block raises latency for extremely utilized network. MinBD deflection router provides the best solution for these problems. In case of contention of destination port, an additional side buffers in each direction is used to hold data packets. Thus the small buffering in the input block will help to reduce deflection in the network. The key principles of working of MinBD router are [12]

- In the network of contention of data packet, it is often easier to buffer data packets and arbitrate them in a later cycle. Small buffering in this case helps to reduce chip size. This also prevents data packet deflections because of unavailability of destination port.
- It could be possible to route data packets to the destination port in the very first attempt. Thus with this situation data packet buffering leads to excessive power and area overhead. Router will then buffer a packet of data if necessary. Thus the side buffer comes in to the picture when data packets would have been deflected due to contention of destination port. These data packets should be temporarily withdrawn from the network, and placed in the side buffer. It is called side buffer injection of data packets. This reduces the deflection rates for the network.
- Eventually, the data packet will be ejected from the network as it arrives at its destination.

Normally it is best to have small side buffer in the router's input port which helps to lower the deflection rates.Within the proposed design, single buffer is used in five directions for each router node. Buffer size is assumed as a single memory location in each direction of the router for this input port, but FIFO buffer memory may be added to side buffer if required.

The data packets have 24 bits out of which data is of 16 bits, source address is of 4 bits and the following 4 bits are of the destination address. The subsequent cases are discussed in this study to explain the functioning of input block with side buffer.

Case I: For availability of the destination port
If there is no data ports contention, otherwise data packet will be transferred to destination port immediately. Data packet does not need to be stored in the side buffer.

Case II: Use of side buffer when destination port is not available
If the destination port is not free then it is possible that data packets will be saved in the side buffer. Reinjection of data packets from side buffer can occur after a few clock cycles.

Case III: If the data packets are filled with the side buffer and the next data packet still needs a side buffer
To prevail this situation, the scheduler design must be efficient. After the few clock cycles, the scheduler will grant the access to re-inject the data packets. In case of congestion, preference is always given to first re-inject the data packets and then store the new data into the side buffer. The following figure 3 shows the flowchart of the working of the input block of the router.

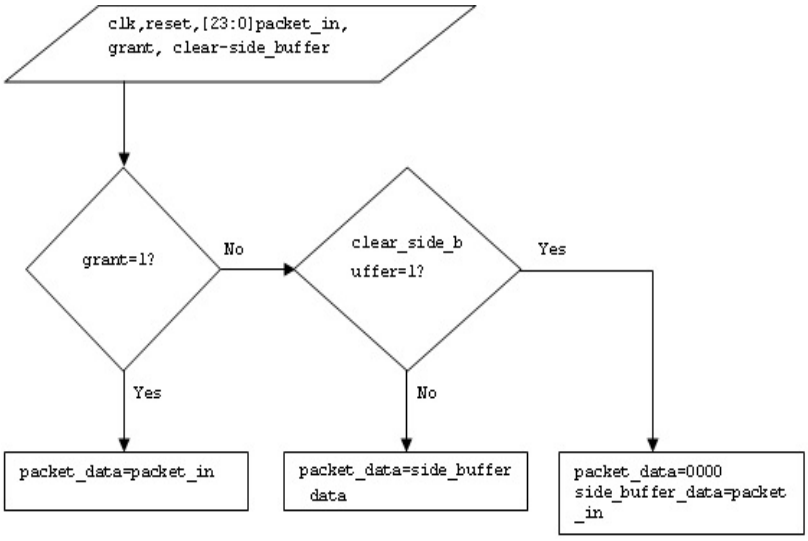

Figure 3: Flow chart of input block

For the implementation of the input block, the scheduling algorithm is not considered at present. Consequently the signals such as packet_in (input data), packet_data (output data) and grant signal are assigned separately to each port. Based on the different test cases, the precise signals in each direction are activated by having a test bench. This design is implemented on a Spartan 3 family XC3S400 device, FG 456 package. The coding is done using Verilog HDL. For simulation of the design, iSIM simulator is used and the synthesis is performed using the XST tool of Xilinx ISE 14.2. Different test cases are reviewed below to explain the operation of the input block.

a) No contention of data packets

Assume packet_in is FFFF0F h which implies a data packet FFFF h with source node 0 and destination node F. In this case as per X-Y routing algorithm for mesh topology, the data packet will be travelling in the East direction. In the event if the output port isn't busy, then the data packet is directly transferred to the destination node i.e. East port in this case. Hence packet_out_e has received data FFFF h. The simulation result of this test case is provided in the following figure 4. The data on the various signals are mentioned below,

packet_in_e = 24'hFFFF0F
packet_src_e = 0
packet_dst_e = F
grant_e = 1
packet_data_e = FFFF h

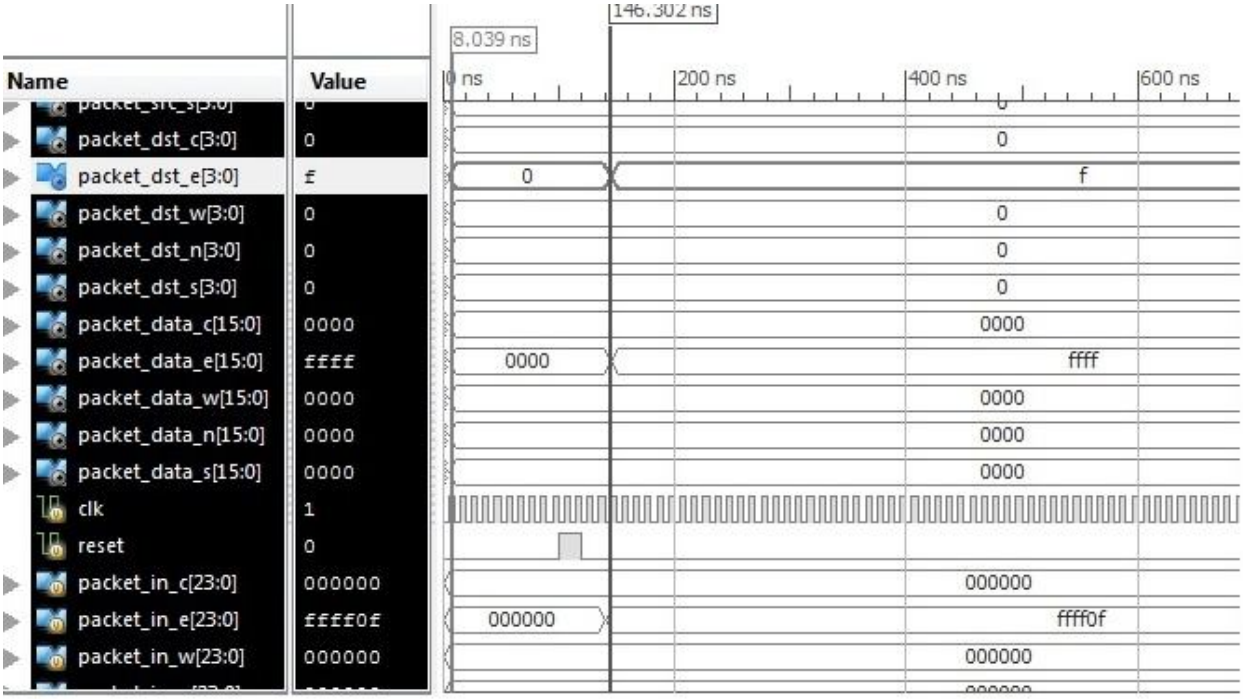

Figure 4: Simulation result of input block with no contention of data packets

b) Use of side buffer when there is no destination port available

If the destination node is occupied, then the data is stored in the side buffer.Consider the case packet_in_n= FF00F0h which indicates the source node is F and destination node is 0. So as per X-Y routing algorithm for mesh topology architecture, the data travels in the North path. In this case if the destination node is not accessible (grant_n=0), the packet will be saved in the respective port side buffer. For this specific case, the data packet is stored in side_buffer_packet_n in the North direction. It saves data packet loss in case of contention. As shown in the following simulation result, the data on various signals are as follows,

packet_in_n = 24'hFF00F0
packet_src_n = F
packet_dst_n = 0
grant_n = 0
clear_side_buffer_n=1
side_buffer_packet_n= FF00F0
packet_data_n=0000

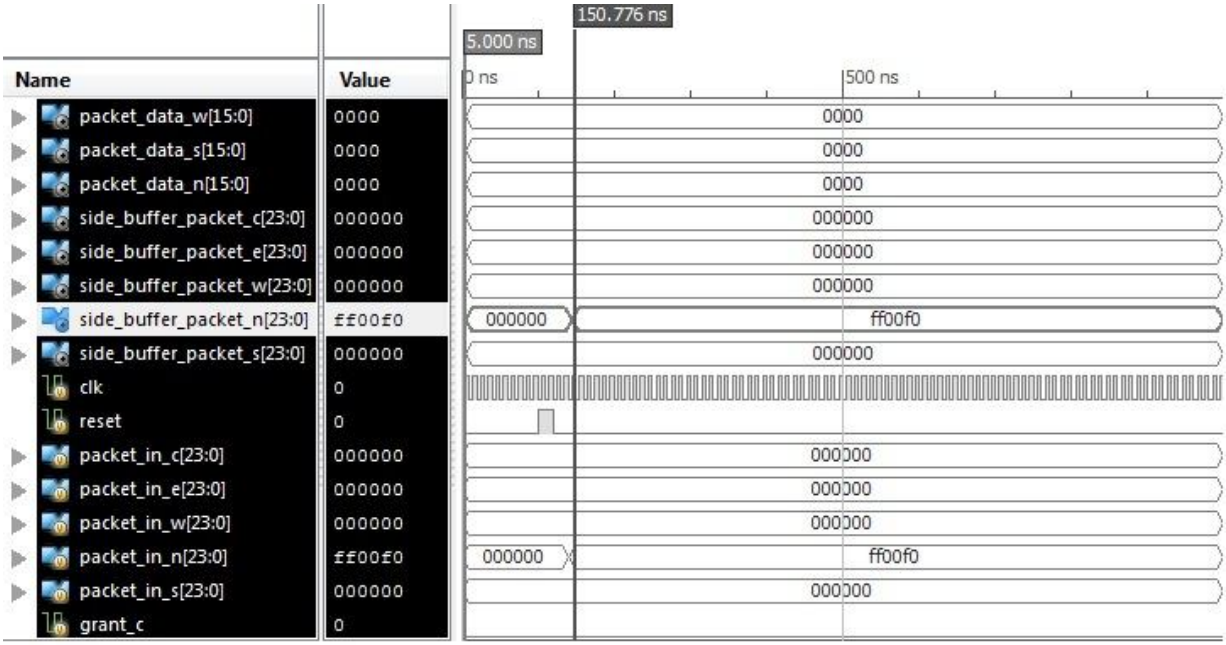

Figure 5: Simulation result of input block with the contention of the data packet

b) Reinjection of data from side buffer

As per the flow chart, if the grant signal for a certain port is not offered for the requested node, then the data packet is kept in the side buffer of the respective node. It helps to prevail over the drop off of data packet. After the output port gets free, the data packet from side buffer has to shift to the respective port. With this algorithm, reinjection of data packet is considered when grant signal as well as clear_side_buffer signal are active low. The simulation outcome of the test case is represented in the following figure 6 and data on the various signals is as follows,

Packet_in_n= FF00F0
side_buffer_packet_n=FF00F0
clear_side_buffer_n=1
packet_data_n=FF00

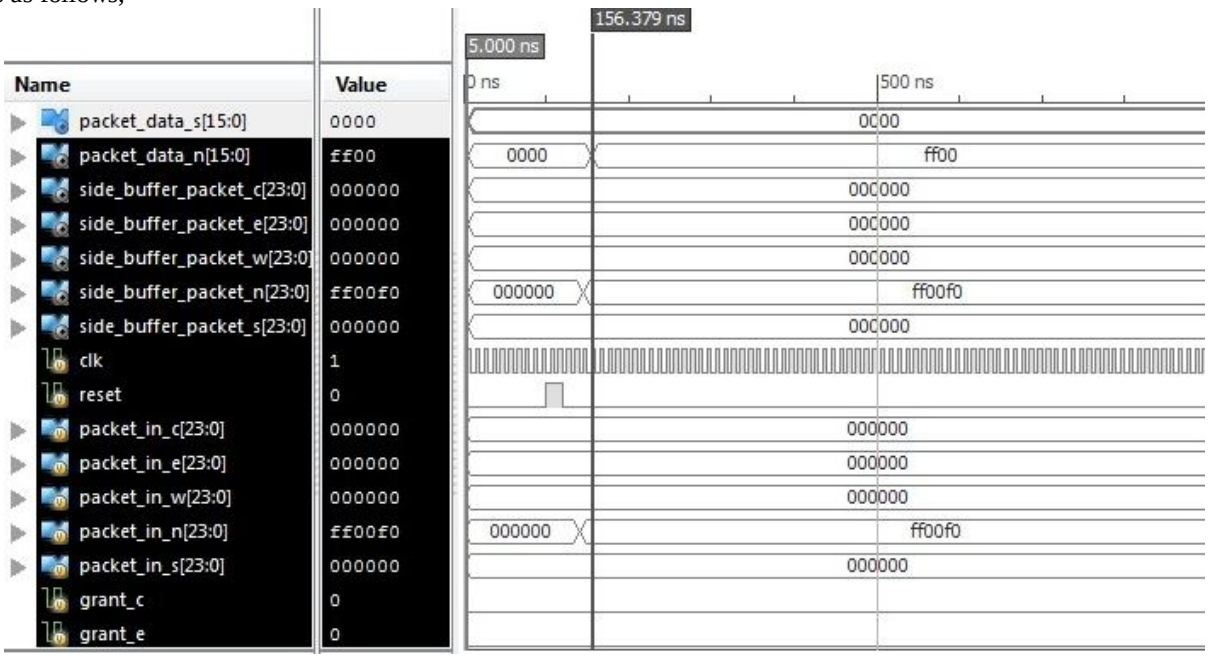

Figure 6: Simulation result of input block with reinjection of data from side buffer

3.2 Implementation of iSLIP scheduler

Scheduler plays a very important function in data packet communication in NoC. Any scheduler (arbiter) has the main feature of scheduling competing requests for the same destination. Since the arbiters directly decide the router's operating speed, the design of fast arbiter is a paramount important. An improper design of scheduler prospects to starvation as well as congestion issues.

Here iSLIP scheduler is implemented on hardware platform such as FPGA. iSLIP scheduler is an improved round robin arbiter version where programmable priority encoder is used to eradicate starvation and achieve maximum throughput.
In an extremely busy system, the main shortcoming of the priority arbiter is that there is no limit to how long a lower priority request may have to wait for it to obtain a grant. A round-robin arbiter, on the other hand, requires every requester to take a turn. To keep record of the next requester a pointer register is maintained. If that requester is active, it will obtain the grant. If not, then pass the pointer to the subsequent requester. Therefore, the optimal time a requester is waiting for is limited by the number of requesters minus 1. Therefore, iSLIP scheduler is the solution for round robin algorithm improvement. Essentially iSLIP algorithm is a round robin algorithm with programmable priority encoder (PPE). In the proposed work iSLIP arbiter is designed without any set priority.

For priority consideration an additional port pointer is used. Source and destination nodes are chosen based on the X-Y routing algorithm and the grant signal is activated in a particular direction for data packet transfer. It thus becomes a programmable priority encoder that overcomes the round robin arbiter's problem of cyclic order priorities [23]-[25].

The flowchart of the iSLIP scheduler is as shown in the following figure 7. Also, the scheduler states for different directions are mentioned in the following table. Based on the source and destination node pair, the iSLIP scheduler state is assigned in a specific direction. The iSLIP scheduler state activates crossbar signal for a destination node. The iSLIP scheduler is implemented using Verilog HDL on Spartan 3 XC3S400 family and device is selected as 4FG456.

Table 1: Different iSLIP scheduler states

Direction	**iSLIP Scheduler state**
East	000
West	001
North	010
South	011
Core	100

The following simulation waveform as shown in figure 8 verifies the implementation result of the iSLIP scheduler. Various ports are having following data,
Packet_data_w= FF11A0
packet_src = A
packet_dst = 0
islip_scheduler=1(west direction)
crossbar_data_w= FF11A0

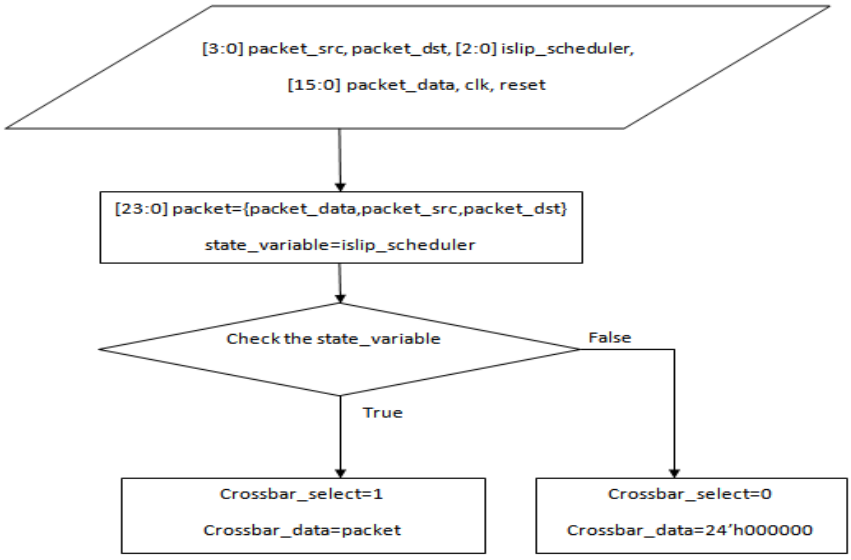

Figure 7: Flow chart of iSLIP scheduler

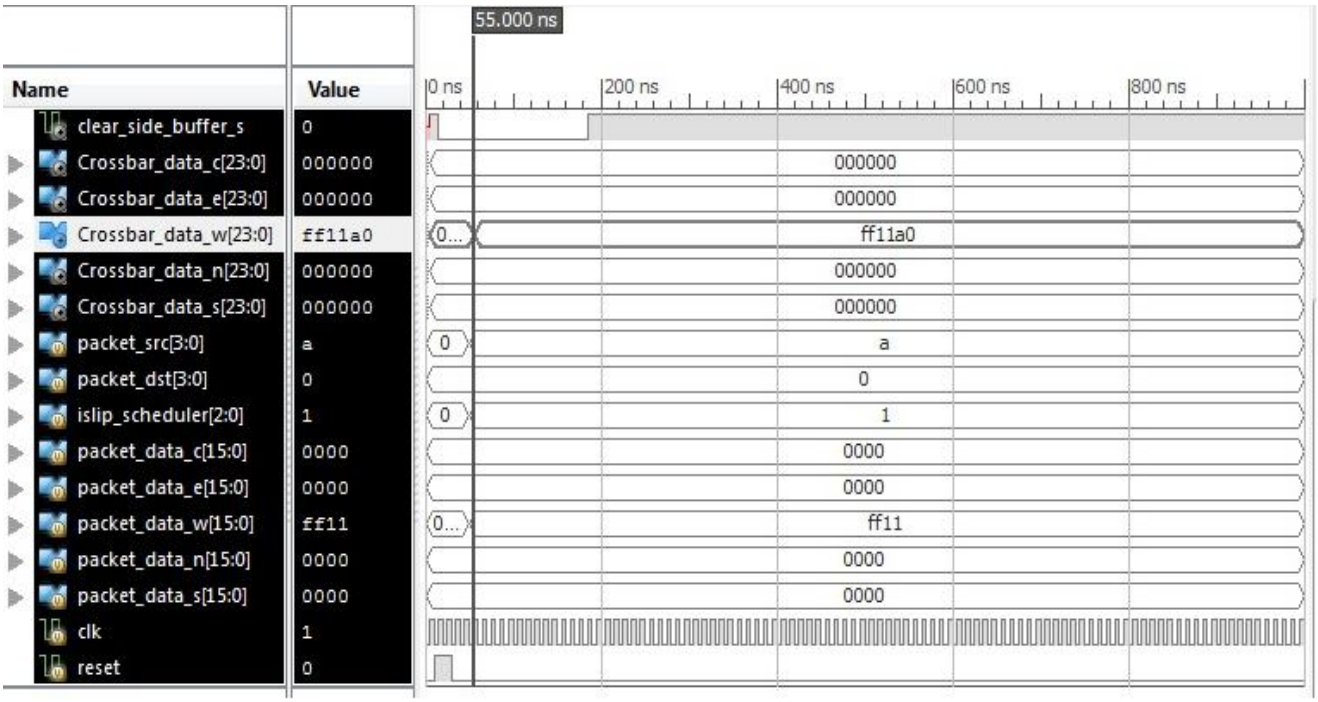

Figure 8: Simulation result of iSLIP scheduler

3.3 Implementation of cross bar switch

The crossbar switch is in charge for linking an input port to its destined output port based on the grant signal issued by the scheduler. In this design, the iSLIP scheduler makes the decision for crossbar switch activation in a specific direction of the destination port. The data is to be transferred to the output node depends on the select lines. These select lines are provided by the arbiter (islip_scheduler), depending on the request signals. Therefore the activation of a specific crossbar signals depend on the islip scheduler. In the following figure 9, the flowchart of the crossbar switch is given.

The following simulation result of figure 10 shows the working of the crossbar switch. The data packet has to transfer from node 3 to node F in the South direction. Therefore the crossbar select signal in the South direction is active and it allows the data packet to be transferred to node F in the packet_out_s port. Various signals are given as follows,

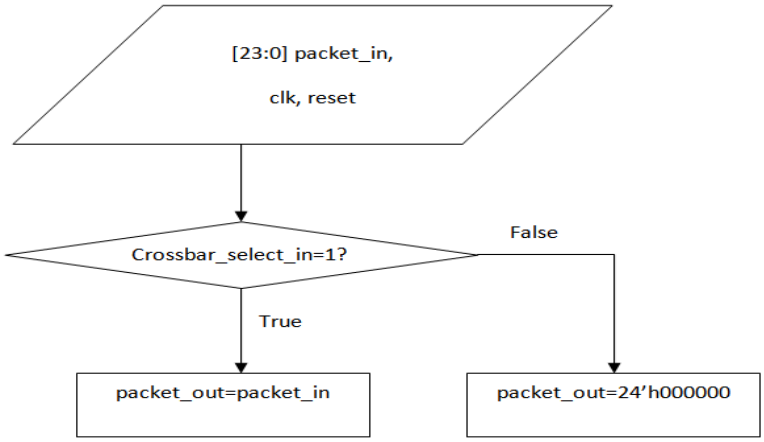

Figure 9: Flow chart of cross bar switch

packet_in_s= F0003F
packet_src = 3
packet_dst = F

crossbar_select_in_s=1(data transfer in South direction)
packet_out_s=F0003F

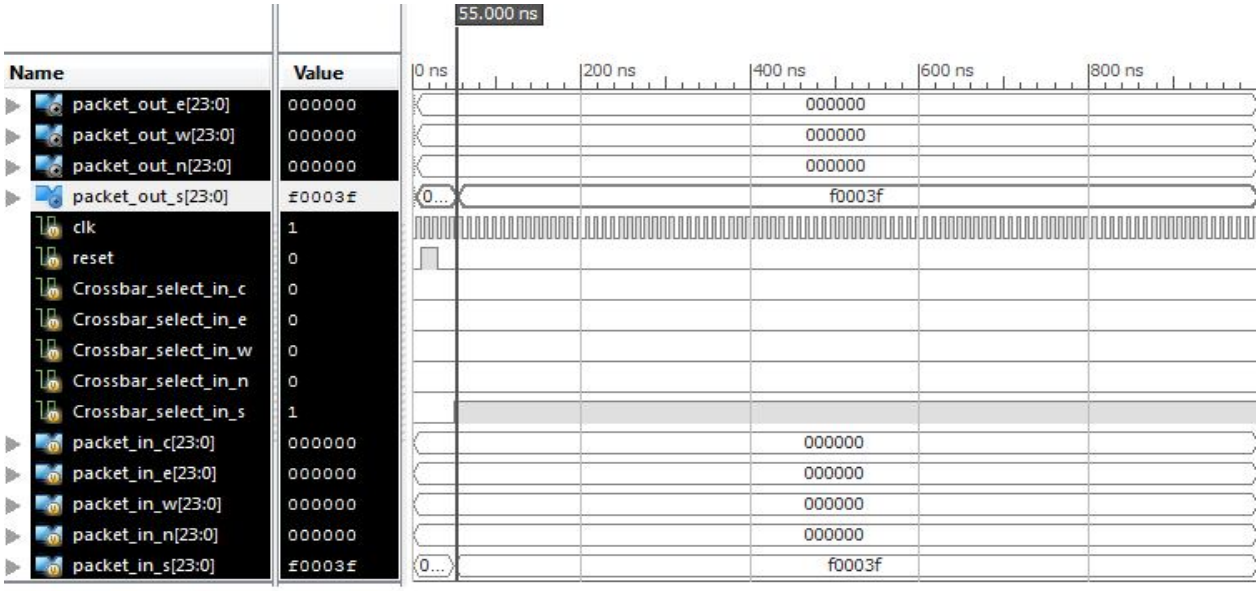

Figure 10: Simulation result of cross bar switch

3.4 Implementation of single node NoC router

Each block of NoC router is analyzed individually and then all these blocks are integrated in the top module to form single node NoC router. The design is modeled in Verilog HDL, using Xilinx ISE 14.2 with family Spartan 3, device XC3S400, package FG 456. Simulation is performed using iSIM simulator and synthesis is achieved using XST tool. A simulation result is given in figure 11 and logic utilization is given in table 2. If packet_in is FFFF73 h, this means the transfer of FFFF h data from source node 7 to destination node 3. Transfer of data packets occurs in north direction. Therefore the iSLIP scheduler state is assigned as a 2(010). Further, it selects the crossbar switch in North direction and the data packet is transferred to node 3.It is observed that the single node NoC router required 6% LUT utilization on Spartan 3 FPGA. For data packet transfer the latency of 2 clock cycle is observed and the design is run at a maximum operating frequency of 118 MHz. The Xpower analyzer provides overall power consumption of 0.570 W.

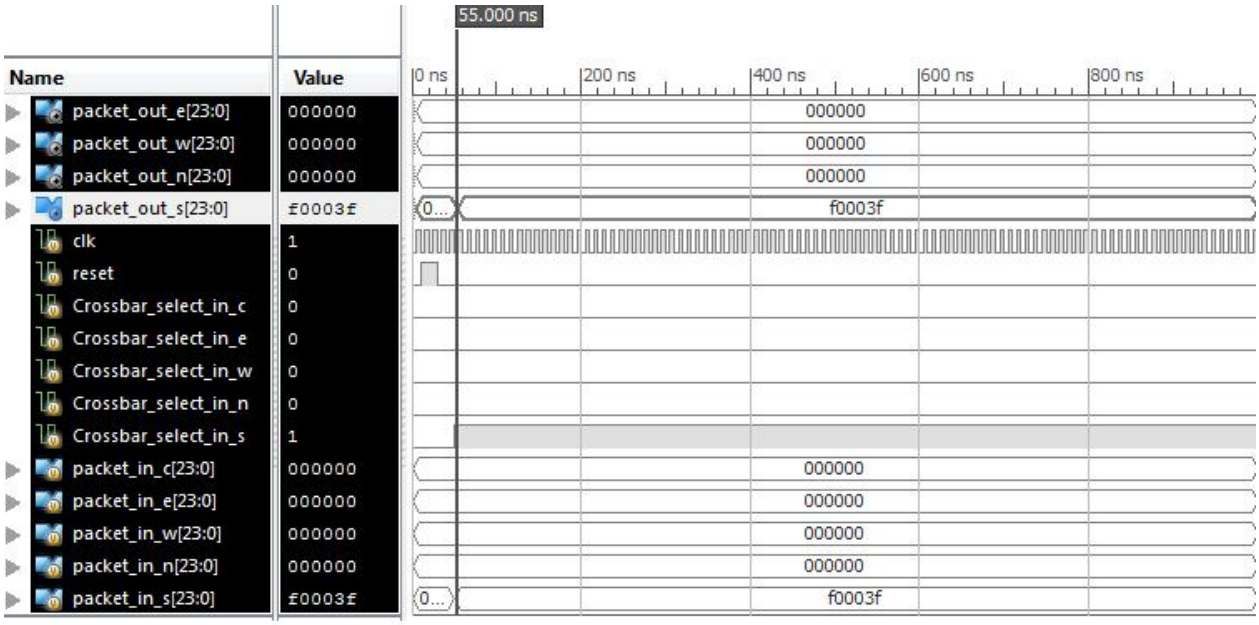

Figure 11: Simulation result of single node router

Table 2: Resource utilization summary of single node router

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	124	7168	1%
Number of LUTs	436	7168	6%
Number of bonded IOBs	245	264	92%

4. CONCLUSION

The proposed design of single node router architecture is examined for the area and latency optimization. The area of router as well as power depends on the amount of buffers utilized in the input port. The scheduling algorithm is useful to control latency optimization. Consequently this paper proposed modified router design with area optimization with a small side buffer in each direction and an iSLIP scheduler as a fast scheduling algorithm.

It's observed that input block with single buffer provides optimized result for the area as against traditional buffered router architecture. In the case of destination port contention, the data packets get preserved in the side buffer and those data packets are re injected after few clock cycles. In addition, it reduces the design complexity of router design due to use of small buffer size. The iSLIP scheduler block is another feature of the router which has a programmable priority scheduler to provide access to just the requested port rather than adhering to a round robin scheduling for assigning grant to the port. It offers the fast scheduling of most of the requests and hence optimizes the latency of the router. The future scope of the design is to develop 4x4 mesh topology based router and compare the results with different NoC architectures.

REFERENCES

1. A. Salaheldin, K. Abdallah, N. Gamal and H. Mostafa. **Review of NoC-based FPGAs architectures,** *5th International Conference on Energy Aware Computing Systems & Applications*, Cairo, 2015, pp. 1-4. https://doi.org/10.1109/ICEAC.2015.7352172
2. D. Atienza, F. Angiolini, S. Murali, A. Pullini, L. Benini, and G. De Micheli, **Network on chip design and synthesis outlook,** Integration the VLSI Journal Vol. 41, no. 3, pp 340-359, May 2008.
3. W.J Dally and B. Towels. **Principles and Practices of interconnection networks**,Morgan Kaufmann 2004.
4. C.Amaresh and Anand Jatti. **Performance analysis of synchronous network on chip(NoC) for high speed and low power multiprocessor on FPGA**, International Journal of Advanced Trends in Computer Science and Engineering, Vol 9, no 2 pp 2241-2252. 2020. https://doi.org/10.30534/ijatcse/2020/203922020
5. K. Shashi, A. Jantsch, J-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani. **A network on chip architecture and design methodology**,*VLSI Proceedings, IEEE Computer Society Annual Symposium on*, pp. 117-124,2002
6. S. Singh, J. V.R. Ravindra and B. Rajendra Naik. **Power and Area optimized FRA-CSLA for high speed NoC applications**. International Journal of Advanced Trends in Computer Science and Engineering, Vol 8, no 3 pp 883-888, 2019. https://doi.org/10.30534/ijatcse/2019/84832019
7. T. Moscibroda and O.Mutlu. **A Case for Bufferless routing in on Chip Networks**, *36th International Symposium on Computer Architecture (ISCA)*, 2009.
8. G. Paul, C. Kim, R. McDonald, Stephen W. Keckler, and D. Burger. **Implementation and evaluation of on-chip network architectures**, *International Conference on in Computer Design(ICCD)*, pp. 477-484, 2006.
9. Lu, Zhonghai, M. Zhong, and A. Jantsch. **Evaluation of on-chip networks using deflection routing**, *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, pp. 296-301, 2006.
10. M.George, Daniel Sanchez, W.J. Dally, and C. Kozyrakis. **Evaluating bufferless flow control for on-chip networks**, *Proceedings of the Fourth ACM/IEEE International Symposium on Networks-on-Chip, IEEE Computer Society*, pp. 9-16, 2010.
11. N. George, C. Fallin, T. Moscibroda, O. Mutlu, and S. Seshan. **On-chip networks from a networking perspective: Congestion and scalability in many-core interconnects**, ACM SIGCOMM computer communication review 42, no. 4,pp. 407-418,2012.
12. C Fallin, G. Nazario, Xiangyao Yu, K. Chang, R.

Ausavarungnirun, O. Mutlu, **MinBD: Minimally Buffered Deflection Routing for Energy Efficient Interconnect** in NOCS 2012, Lyngby, Denmark, May 2012.

13. F. Chris, C. Craik, and O. Mutlu. **CHIPPER: A low-complexity bufferless deflection router**, *IEEE 17th International Symposium on High Performance Computer Architecture (HPCA), 2011*, pp. 144-155, 2011.
14. C. Fallin, G. Nazario, Xiangyao Yu. K. Chang, R. Ausavarungnirun O. Mutlu. **Bufferless and Minimally Buffered Deflection Routing**, Chapter in Routing Algorithm in Network on Chip, Springer 2014. https://doi.org/10.1007/978-1-4614-8274-1_10
15. Y. Cai, K. Mai and O. Mutlu. **Comparative evaluation of FPGA and ASIC implementations of bufferless and buffered routing algorithms for on-chip networks** *16th International Symposium on Quality Electronic Design, Santa Clara, CA*, pp. 475-484, 2015.
16. V. Rantala, T. Lehtonen, J. Plosila, **Network on Chip Routing Algorithms",Journal of Systems Architecture**, TUCS Technical Report No 779, pp. 5-7, August 2006.
17. S. Swapna, A. K. Swain and K. K. Mahapatra, **Design and analysis of five port router for network on chip**, *Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics*, Hyderabad, pp. 51-55,2012.
18. N. McKeown. **The iSLIP scheduling algorithm for input queued switches**, *IEEE/ACM transaction on Networking* , vol. 7,No2, April 1999.
19. Pape J. **Implementation of an on-chip interconnect using the i-SLIP scheduling algorithm,** Intermediate Report – Specification and Timeline," Nov 2006.
20. Gupta P., McKeown, N. **Designing and implementing a fast crossbar scheduler**, Micro, IEEE, vol.19,no.1 pp.20-28, Jan/Feb 1999. https://doi.org/10.1109/40.748793
21 D. Belhajali, Mbarek, I. B., Hasnaoui, S., and Jelassi, K. **A SoC interconnection scheduling: Design and implementation of a scheduler based on credited iSLIP algorithm**, International Journal of Advances in Optoelectronic Materials(AOM) Volume1, Issue 2, May 2013.
22 M. Gospodinov, E.Gospodinova. **Analysis of iSLIP scheduling algorithm for input-queuing switches**, *International Conference on Computer Systems and Technologies – CompSys Tech'2004.*
23 R. Deb, and Rajrajan. **Speed efficient implementation of round robin arbiter design using VERILOG**, *International Journal of Enhanced Research in Science Technology and Engineering*, ISSN: 2319-7463 Vol. 2 Issue 9, pp. 1-9, September-2013.
24 U.Saravanakumar , K. Rajasekar and R. Rangrajan, **Implementation of scheduling algorithms for on chip communications**, International Journal of Computer Applications (0975 – 8887) Volume 62– No.14, January 2013. https://doi.org/10.5120/10151-4986
25 D. Mahobiya, **Designing of efficient iSLIP arbiter using iSLIP scheduling algorithm for NoC**, International Journal of Scientific and Research Publication,volume 3,Issue 12, December 2013