Materials Today: Proceedings xxx (xxxx) xxx

Contents lists available at ScienceDirect

Materials Today: Proceedings

journal homepage: www.elsevier.com/locate/matpr

Design and implementation of asynchronous NOC architecture with buffer-less router

Trupti Patil *, Anuradha Sandi

GNDEC, Bidar 585403, India

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Asynchronous design
Artificial Intelligence
Buffer less Routing
Network On Chip
System On chip

ABSTRACT

In present days, development in chip technology enables us to incorporate heterogonous cores on one chip. Synchronization and communication between such diverse cores is a major issue. Current Synchronous NOC architecture with buffered router consumes a substantial chip area and Power (clock distribution & I/O buffers). Hence in recent times Buffer-less routers based on Deflective routing are projected as a possible solution, but they suffers from issues like sequential port allocation and slow critical path and also increases the Latency. In this paper we propose 2D 4x4 Asynchronous Mesh NOC architecture with a novel router design using XY routing algorithm. In the proposed router design, we have eliminated the conventional input and output buffers and crossbar switch. We integrated priority encoder and FSM based Arbiter which efficiently solves long critical path issues of conventional Buffer less router by dynamically changing the port priority and hence solves the starvation and Live/Dead lock issue and improves the Area Consumption (i.e., reduced by ≈ 43%). We have used parallel transmission which enables us to improve Speed (by ≈73%) and elimination of I/O buffers reduces the power consumption (≈ 0.56w).

1. Introduction

Advancement in chip technology has made it possible to integrate heterogeneous cores from simple memories to complex DSPs on one-chip (SOC) [1]. As the number of processing elements on-chip increases, the intricacy of communication between them also increases. In recent time, NOCs have become the central nervous system of SOCs and projected as a potential solution to solve the communication issues of intricate SOC designs [2]. NOC defines the physical interconnection of the processing element through a network of routers and links. The NOCs are designed with primary objective of network-on-chip (NOC) is to move data around a chip as efficiently as possible with as little impact as possible on design while meeting or exceeding key design metrics (PPA, etc.) and solve the issues of conventional shared bus architecture.

The Router is the main component of NOC, plays an essential role in coordinating data transfer in the network. The number of Router increases proportionally with the number of cores in a network; hence, designing an efficient Router is the main requirement to achieve the required system performance. Router efficiency is mainly defined by NOC architecture, Routing technique, Network topology, Buffer size and Arbiter design. The first parameter is NOC Architecture; it could be Synchronous, Asynchronous or GALS. In Synchronous NOC architecture, the Routers are driven by global clock hence it consumes more power Synchronous designs are area efficient and fast, but implementation of high frequencies will be challenging and they suffer from EMI (Electromagnetic interference) [3]. To solve the Global clock distribution led problems (e.g.; clock skew/clock Jitter) of Synchronous NOC, Researchers have come up with many intermediate solutions like GALS (Globally Asynchronous and Locally Synchronous) [4], in this technique the whole system is divided into smaller synchronous regions which removes the need for global clock. NOC which supports both Synchronous (end to end path) and Asynchronous communication (NOC-IP) is proposed in [5] to achieve reduced energy consumption and improved Latency. An energy efficient Synchronous- Asynchronous Circuit switched NOC [6] is proposed with two sub router (one for Synchronous control and other for Asynchronous Data transfer). In an Asynchronous NOC design, the Routers are self-

* Corresponding author at: Dept of ECE, GNDEC, Bidar-585403. Tel.: +91-9611589131.
E-mail addresses: patiltrupti11@gmail.com (T. Patil), anu29975@gmail.com (A. Sandi).

https://doi.org/10.1016/j.matpr.2021.05.282
2214-7853/

timed and they are liberal to process variations, the whole chip in Asynchronous design can be divided into many isolated clock domains, it reduces the overall design time but the design will be comparatively slow. Asynchronous designs are most preferable choice in real time application where the transmission of small data packets within limited power requirement. Synchronous designs are good choices where large data packets and continuous transmission are involved for example multimedia applications [7]. So far now many Asynchronous NOC designs are presented by researchers such as Bundled data logic which provides high throughput with simple hardware requirements but they are very sensitive timing variations. A 2 Φ click based Mesh NOC with bonded bundled data is proposed [8] to improve the Latency. The Asynchronous designs based on QDI (Quasi delay insensitive) technique where the application performs correctly, independent of delays are used for NOC design. The QDI techniques implemented using many different encoding techniques such as LETS (Level Encoded Transition signaling) [9] and LEDR (Level Encoded Dual Rail encoding) [10], the first one uses four phase encoding, for this throughput is low as the communication step are more, whereas in LEDR is a two phase encoding tecnhique, which reduces the communication steps hence increases the Throughput but they can applied only small applications (e.g.: Links) [5] as handling the two phase information is difficult task. An architectural exploration is presented in [11] shows that Asynchronous NOC Router designs shows better results in terms throughput, Latency and inherent Fault tolerance. A concept of Roundabout NOC RNOC and RNOC-A (for Asynchronous NOC) are proposed [12] where a lane of buffers are shared by multiple IO ports to improve the power and area consumption

The Second parameter is routing protocol, if we select a complex routing technique; it will complicate the router design and increase power consumption and chip area. On the other hand, if we use a simple routing protocol, it will outperform in energy and cost but not efficient in routing traffic across the network.

The third parameter is the Buffer size; buffers stores the data packet, which reduces the dropping off and misrouting of the parcels [13].But, they consume considerable power (dynamic power - read/write operations and static power - empty) and chip area. For example, in TRIPS [14], input buffers occupy 75% of network area). In recent times, Buffer-less routing algorithms (e.g., CHIPPER [15] & BLESS [16]) offers solution to Buffered routing algorithms, and they work on the principle that packets are never stored, but are deflected in the network. Deflection causes unwanted hopping which reduces network throughput and increases power consumption, and present Buffer-less router is based on sequential port allocation, which leads to long critical paths. Buffer-less routers are only suitable for low to medium network load; hence Minimally buffered deflection routing are suitable for higher network load [17]. Randomly prioritized Buffer-less GALS NOC based on 3D lottery based routing is proposed [18].

Buffer-less NOCs present a tradeoff: by eliminating buffers, the peak network throughput is reduced, potentially degrading performance. However, network power is often significantly reduced. For this tradeoff to be effective, the power reduction must outweigh the slowdown's effect on total energy. Minimal performance reduction with buffer-less when NOC is lightly loaded, which constitutes many of the applications they evaluated. Buffer- less NoC design thus represents a compelling design point for many systems with low-to-medium network load, eliminating unnecessary capacity for significant savings.

The fourth factor is Arbiter; it controls the scheduling of the ports and hence solves contention issue. Arbiter monitors the status of all the ports and tracks which port is communicating and which is free. The Arbiter makes sure that no packet waits forever for the output port. In round-robin arbiter [16], an equal time slice is assigned for each requester. Once the request is served, it has maximum wait time (twait_max=no. of requester-1). The RRA iSLIP algorithm [19] is used which based programmable priority encoder, where instead of round-robin fashion, packets are transferred in a specific direction. It will face the starvation issue if the data packets are arriving continuously from a specific direction. Hence designing an efficient arbiter is essential to solving the Live-lock and Deadlock and Starvation issues in NOC.

In this paper, we propose Asynchronous 4x4 Mesh NOC architecture with a Buffer less router using XY routing on FPGA. In the proposed Router design, we have eliminated the conventional crossbar switch and Input/output buffer. Priority encoder and Arbiter in our design are designed to solve the long critical path problem of traditional buffer less techniques. This design gives improved performance in terms of area, speed and power consumption.

2. Proposed NOC router architecture

In this paper, we chose to design a Buffer-less Router with 5 I/O ports; four directional (E, W, N& S) ports and one local port. Local port input and local port output provide the connection with the local processor or controller, whereas directional (north, east, west and south) ports are used to connect to other neighboring nodes in the network. Here number of input port is equal to number of output port which makes router Dead lock free. Fig. 1 shows the internal structure Router. It consists of priority encoder block, Arbiter (scheduler) block, Routing Controller (XY routing) block and LEDR encoder-decoder block; we have eliminated input/output buffers and crossbar switch to improve the area and power consumption. There is no need for virtual channels in our design as it is a buffer-less design with equal number of I/O ports. The packet which is arriving router will be directed to corresponding output port, which makes the router design Fast (No need to store the packet in the router.), Simple (as there is no read/ write operation in router level), Area efficient (No buffers, virtual channels & crossbar switch) and Power efficient (buffers consumes reasonable power static: when empty & Dynamic: while RD/WR operations). The proposed design is implemented on FPGA [14]. Reconfigurable property of FPGAs makes it a better choice in evolving SOCs and AI applications [13].

2.1. Priority encoder

While transferring the packets in the network, we can either choose to have hardware control, by configuring the hardware in such a way that high priority packets are routed through high priority links, or we can go for software control by using credit-based flow control. These choices can be very application sensitive and provide a lot of flexibility in configuring the NOC. In our design we are using software control through Priority Encoder. When many ports requests for same output port, Arbiter (scheduler) is in charge of port allocation; hence, it should be designed to make contention-free and fast scheduling. In our design, instead of hard-wired switch-based design, we are using FSM based arbiter with priority encoder. The priority encoder Fig. 2 here Assigns priority in clockwise direction, highest priority to port connecting to the processor/core, next to North, East, South and West. Instead of conventional Round Robin design, directional priority assignment helps to fast scheduling, however this alone may lead to starvation, if packet is arriving continuously from particular direction. Hence we designed our arbiter block to dynamically handle the priorities of different ports (Fig. 2).

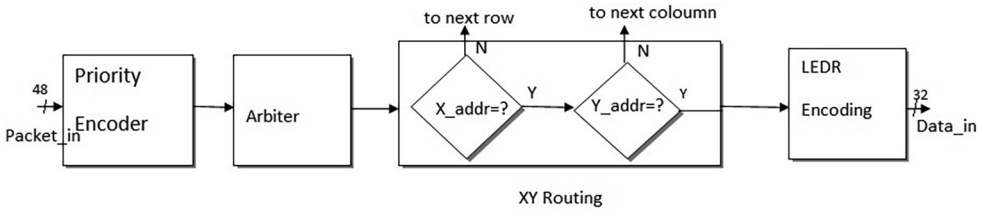
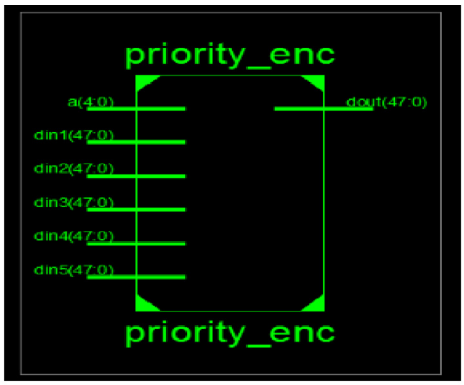
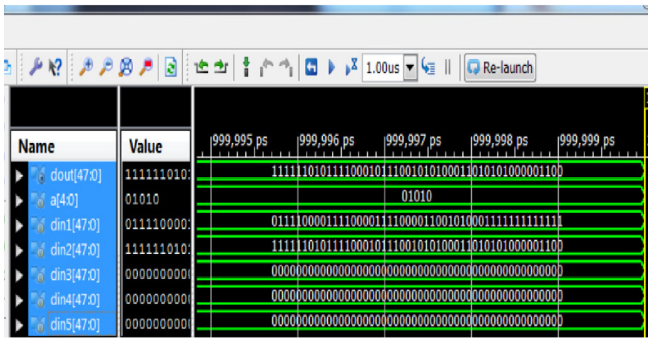


Fig. 1. Internal structure of the router.



(a)



(b)

Fig. 2. (a) Top view of priority encoder (b) Simulation results.

2.2. Arbiter

The Arbiter here barrows the concept from Weighted Round Robin scheduling [21] & iSLIP scheduler. Priority encoder in our design assigns directional priority (as in iSLIP), whereas arbiter dynamically changes the port priority i.e., even though the port is assigned with highest priority by the Priority Encoder, Arbiter masks the priority once the port served as in case of WRRA, unlike WRRA it does not follow the cyclic order. The Priority encoder and the Arbiter in our router are designed in such way neither it requires an extra block to calculate next priority generation block nor credit assignment and maintenance block. If inputs from each port are destined for different ports then the packet will be sent normally to the corresponding output port without any issue. But if more than one input port request to send the data packet to the same destination port then, Arbiter first checks the priorities of port and starts with the port with highest priority (as assigned by the Priority encoder). After each transfer the request in hand will be given least priority in next round of arbitration (whatever may be the port priority). This arbitration is somewhat similar to credit based arbitration but instead of employing a credit assignment and management block a simple Priority assignment block is used which assigns the priority and some internal registers are used keep track of the port served. Our Arbiter keeps track of the ports been served accordingly changes the port priority dynamically, which makes the design starvation and Live lock free without complicating the design (Fig. 3).

2.3. Routing controller

Routing controller in our design employs XY routing algorithm which is simple and suitable for both regular and irregular topologies. It makes the design Live/Dead Lock free. The XY routing algorithm generally follows the shortest path. This block is implemented using two comparator sub blocks first block checks the X address and second block checks the Y address. Once Arbiter selects the packet, Routing controller unit routes the packet to destination node by extracting the address information from packet header, it checks the destination address; the first X address is compared with the current address if it matches the packet will be transferred in the same row till Y address matches. If not then the packet will be sent to the next row without processing Y address which saves the processor time. The packet is sent two next row this process continues till X address matches. Once the X address is matched, the next block will compare the Y address with current address if that address matches it means the packet is destined for that node.

2.4. LEDR

The packet after removing header information is sent to the LEDR encoder decoder block. Our idea is to make the design asynchronous by using two phases non-return to zero LEDR encoding, which gives improved performance compared to a conventional four-phase return to zero encodings, in terms of area, power and throughput as it eliminates the spacer between two consecutive data. So far now we have not implemented this part but our next task would be to incorporate LEDR encoding in Router Figs. 4 and 5.

2.5. Packet structure

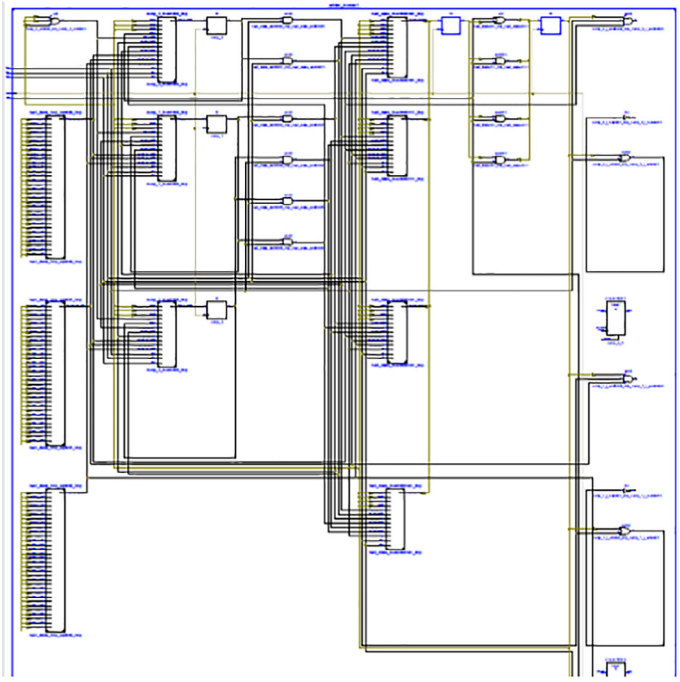
The Network packet in our design is of 48 bit. The Fig. 6 shows the packet structure, First bit is reserved next four bits for address bit. Two-bit holds Row (x-axis) information and next two-bits for column (y-axis). Data packet is of 32 bits of data and remaining are 11 bits are reserved for future use. The NOC should be configurable to add and check parity or ECC where needed, to configure and check for communication timeouts, and even for configuring some of the NoC units to work with a duplicate in lock-step. Also, if needed, the NoC should be able to detect and even isolate misbehaving hardware functions in preparation for a reset/reboot (Fig. 6).

```
   GNT0 : if (req_0 == 1'b1) begin
// updating port priority
                comp_0 = 1'b1;


                        if (comp_0 == 1'b1)
                                Begin
                                comp_0_i = comp_0_i +1;
                                end
                                        if (comp_0_i==1)
                                        begin
                                        comp_0_i = 0;
                                        comp_0 = 1'b0 ;
//Arbitration
                                                if (req_1 == 1'b1) begin
                                                 comp_1 = 1'b1;
                                                next_state = GNT1;
                                                end else if (req_2 == 1'b1 ) begin
                                                comp_2 = 1'b1;
                                                next_state= GNT2;
                                                end else if (req_0 == 1'b1)  begin
                                                comp_0 = 1'b1;
                                                next_state= GNT0;
                                                end else begin
                                                next_state = IDLE;
                                                end
                                        end
                                        else begin
                                        next_state = GNT2;
                                        end
// Priority masking once served

                end else if (req_1 == 1'b1 && comp_0 == 1'b1) begin
                comp_0 = 1'b0;
                comp_1 = 1'b1;
                next_state= GNT1;
                end else if (req_2 == 1'b1 && comp_0 == 1'b1) begin
                comp_0 = 1'b0;
                comp_2 = 1'b1;
                next_state= GNT2;
                end else begin
                next_state = IDLE;
                comp_0 = 1'b0;
                end
```

(a)



(b)

Fig. 3. (a) Arbiter code (b) RTL Schematic.

3. Implementation of NOC architecture

The key idea of efficient chip design is using special-purpose circuitry that allocates computation insensitive tasks across different cores on chip and designing an efficient communication network to transfer the processed information. AI SoC designers frequently implement multiple instances of the same type of hardware in a grid hence they prefer regular topologies such as rings, meshes, or torus, [19]. It helps to ensure predictable data flow, reduces R&D cost, and can help guarantee design scalability over

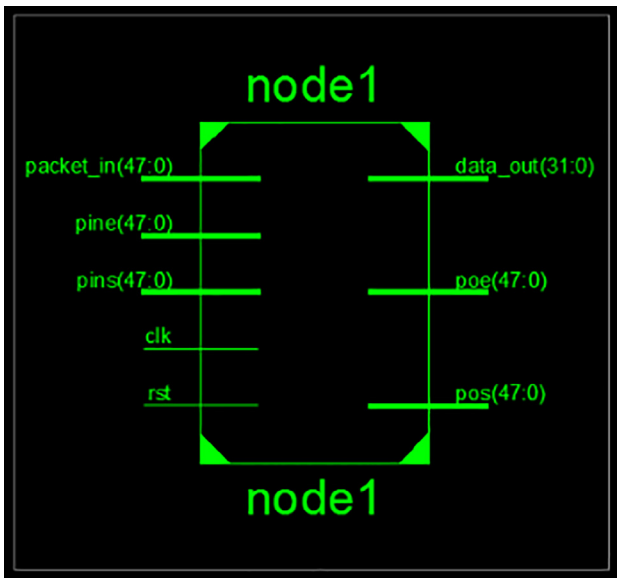


Fig. 4. Top view of Edge node with three I/O port.

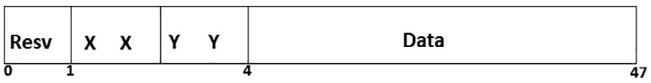


Fig. 6. Packet Structure.

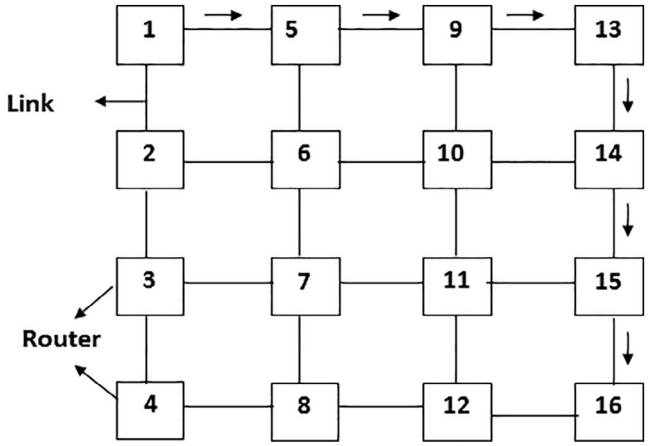


Fig. 7. 2D 4×4 Mesh NOC architecture.

time. Reconfigurable FPGAs is a better choice for evolving chip technology. It has become clear that the NOC has a unique role to play in ensuring that SOC data flow and data integrity. In this paper, we proposed to design a 2D 4x4 Mesh architecture on FPGA. We aim to reduce the area and power burden of the on-chip network. Fig. 7 shows two- dimensional 4x4 Mesh NOC architecture with 16 Asynchronous Buffer-less NOC Routers with five input ports and five output ports. Each Router except the Router at the edge has five-ports; four directions such as East, West, North, South and one local port for connecting to processing core, all the Routers are connected through communication link. Router nodes are arranged (numbered) in column of 4 nodes as shown in Fig. 1. The X–Y routing algorithm [20] is used to establish the communication between the nodes (data packet transfer from the source node to the destination node). In this algorithm, every node sends the data packets first in the X-direction till the column address matches and then sends the data packets in the Y- direction till the destination node is matches. Fig. 1 depicts the case of data transfer between two farthest nodes; node1 to node 16

The proposed Asynchronous NOC architecture with Buffer less Router is modelled in Verilog HDL, using Xilinx ISE 14.7 with Virtex 7 family, on xc7a100t device, using 3csg324 package. Simulated using ISE simulator (iSIM) and synthesis is performed using the Xilinx synthesis tool (XST) tool. The design process, we started with the design of a single router node, here we selected 5 port router, which includes programmable Priority encoder, arbiter controller. The priority encoder's select(sel) signal decides which packet appears at the output port and arbiter is in charge of efficient port allocation. In the second step we have implemented the Routing controller block, which is based on XY routing Algorithm. This block in the router structure enables to pass packets from source to destination node. XY Routing is a simple yet efficient routing technique which makes the design deadlock and livelock free. In third step, we have extended single router node to implement a structured two-dimensional 4x4 mesh Asynchronous

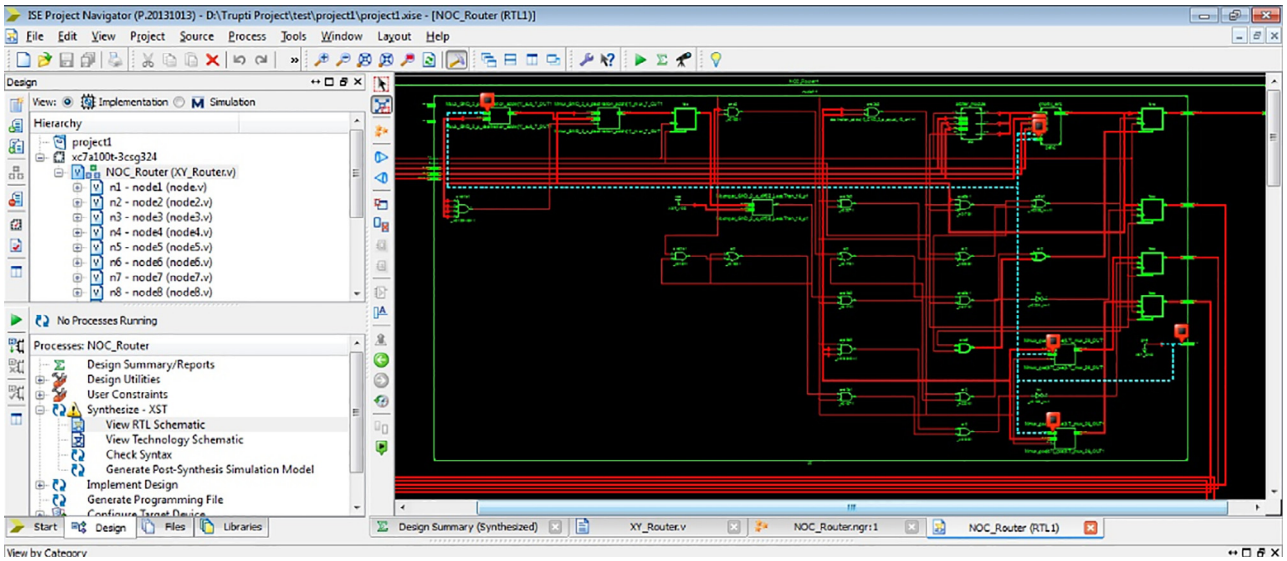
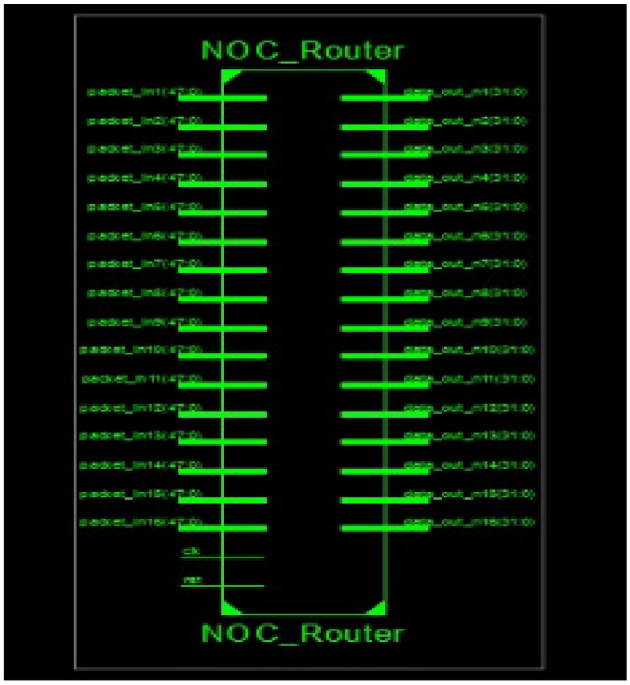
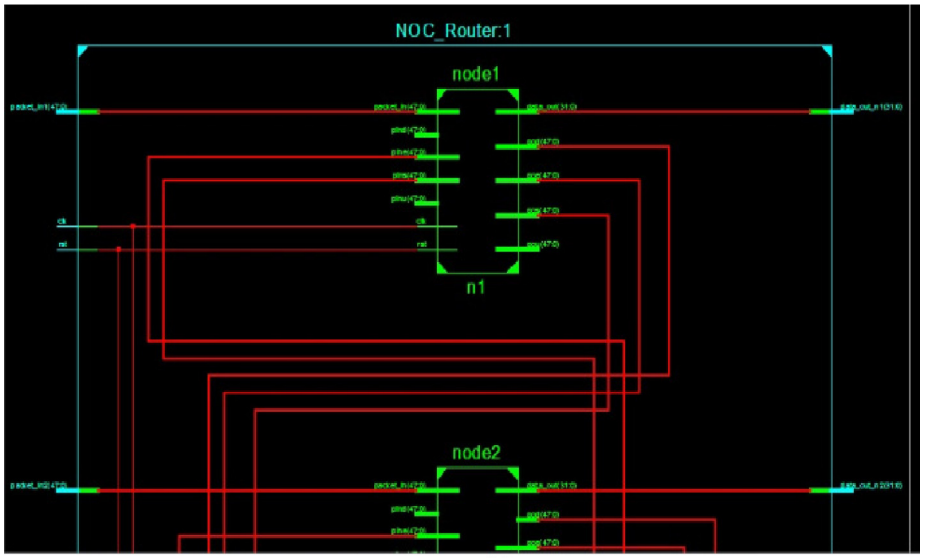


Fig. 5. Single Node (Router) RTL schematic.



(a)



(b)

Fig. 8. (a): Top view of 4×4 Mesh (b): Inter connection between nodes.

NoC architecture. Fig. 8(a) shows the top view of NOC architecture. One pin is assigned for each Router in network. The packet can be injected into the network through any node in NOC. Fig. 8(b) shows the connection between two nodes.

From simulation results we can observe the asynchronous and parallel transmission of data packet. Fig. 9 shows a case of transferring data between two farthest Router nodes, i.e. node 1 to node 16. In our design we are using parallel transmission, we can observe that to transfer the packets between two farthest router nodes (node 1 to node 16), it takes less than 50 ps for a packet to reach to the destination node. Fig. 10 shows simulation results showing data propagation in the internal node before appearing at the output node; this shows how the packet is first transmitted in the X direction and then in the Y direction (1-5-9-13-14-15-16).

From the above figure, we can observe that for transferring data packet from router node 1 to 16 the packet is taking six hops the total time is less than 0.5 ns and latency of 5 clock cycle. Table 1 shows the utilization summary of the proposed design we can see here that NoC architecture is efficient in terms of resource utilization; it is consuming only 2% of registers and 5%of total LUTs available. The power analyzed using Xpower analyzer, it is about 0.56 W. Table 2 shows the comparison of proposed with standard

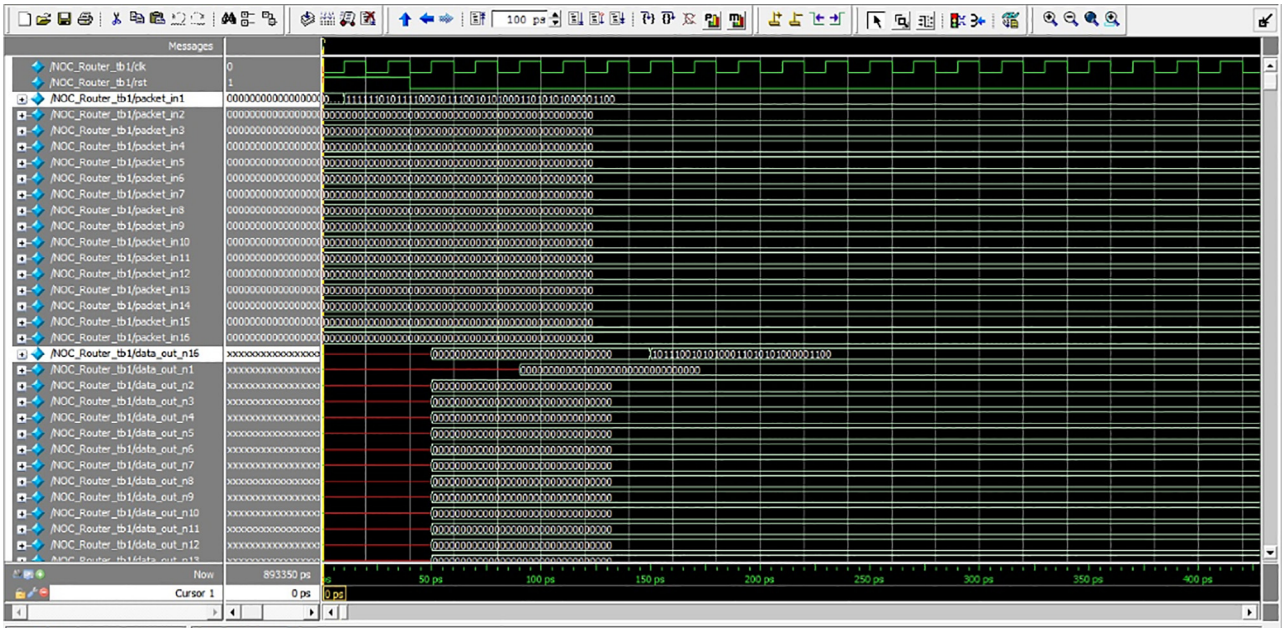


Fig. 9. Simulation result showing input given at node1 and output appearing at node 16 core.

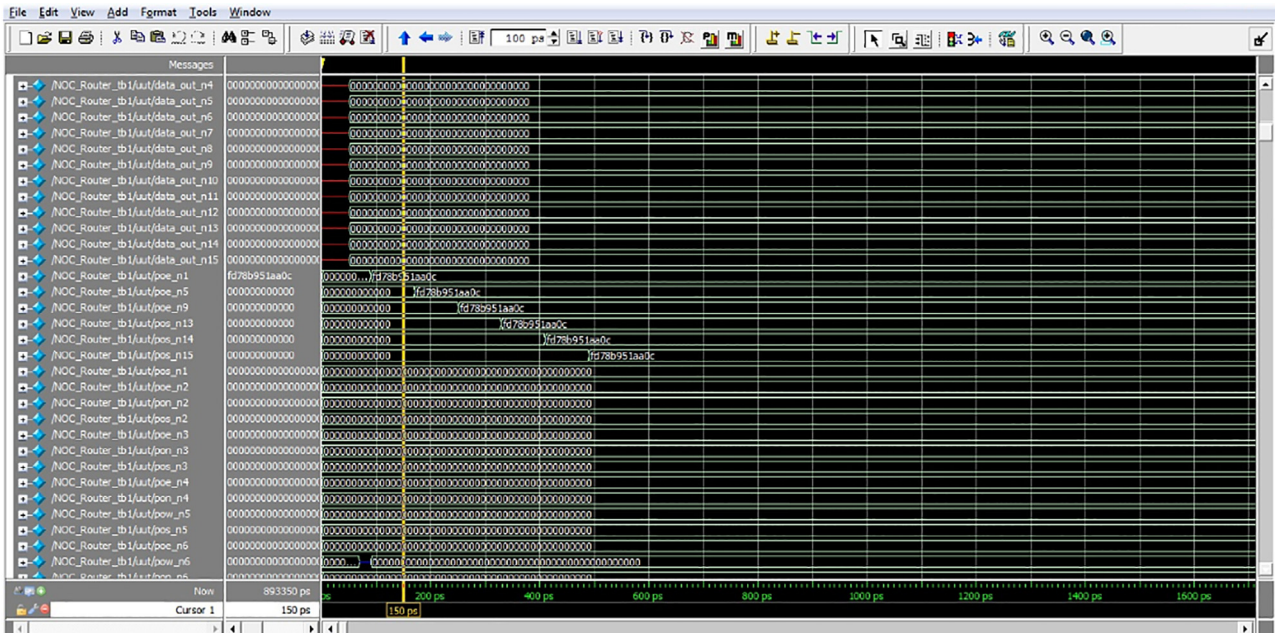


Fig. 10. Simulation results showing data flow in internal nodes before appearing at the output.

Table 1
Device utilization summary 2D 4x4 Mesh Asynchronous NOC Architecture.

	Device utilization Summery		
Logic Utilization	Used	Available	Utilization
Number of Slice of Registers	2978	126,800	2%
Number of slice of LUTs	3544	63,400	5%
Number of fully used LUTs and FF pairs	2895	3627	79%
Number of BUFG/BUFGCTRLs	1	32	3%

designs like conventional Buffered router, Buffer less designs like BLESS and CHIPPER. The proposed design gives improved performance in terms of area and speed.

4. Conclusions and future scope

We have presented two-dimensional 4×4 Asynchronous Mesh NOC Architecture with buffer-less Router. In our proposed router design we have removed I/O Buffer and replaced traditional cross-bar switch with FSM based Arbiter which handles the port Allocation efficiently with improved speed and reduced the hardware cost. The proposed Priority encoder and Arbiter together efficiently solve the starvation& Live/Deadlock issues. The design is capable of handling low-to-medium network load efficiently. Area consumption is reduced by (12–43%) compared to buffered router and present buffer less router. The parallel transmission method employed in our design improves the speed by (73–81%).our design most

Table 2
Comparison of proposed router with standard Buffered and Buffer less Router designs.

	Buffered	BLESS	CHIPPER	Proposed Router	Percentage improvement		
					Buffered	BLESS	CHIPPER
Area (in μm^2)	480,174	311,059	306,165	271,185	43%	12.8%	11.4%
Time (in ns)	1.88	2.68	1.90	0.5	73%	81%	73%

suitable option for high speed communication intensive application like artificial intelligence. Furthermore, we will be employing two-phase Level Encoded Dual Rail (LEDR) encoding at the Router level, which improves the throughput & power consumption. Our buffer-less router design lacks functionality like fault detection and congestion awareness, our next step is to incorporate such properties in the proposed design. Furthermore, design can be easily upgraded from five port router design to seven port Router hence we can widen present NOC to 3D NOC Architecture.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] A. Jantsch, H. Tenhunen, Network on Chips, Kluwer Academic Publishers, Boston, 2003.
[2] A. Karthikeyan, P. Senthil, Kumar, GALS implementatuion of randomly prioritized buffer-less routing Architecture for 3D NOC, Cluster Comput. 21 (1) (2018) 177–187.
[3] A. Aziz, J. Pape, Implementation of an on-chip interconnect using the i- SLIP scheduling algorithm, University of Texas, Austin, 2006.
[4] A. Lines, Asynchronous Interconnect for Synchronous SoC Design, IEEE Micro 24 (1) (2004) 32–41.
[5] P. Bhojwani, R. Mahapatra, Interfacing cores with on-chip packet-switched networks, in: 16th International Conference on VLSI Design, 2003, pp. 382–387.
[6] C. Effiong, G. Sassatelli, A. Gamatie, Scalable and power efficient implementation of an Asynchronous router with Buffer Sharing, Euromicro Conf. Dig. Syst. Des. (2017) 171–178.
[7] Don Monroe, Chips for artificial intelligence, Commun. ACM A 61 (4) (2018) 15–17, https://doi.org/10.1145/3185523.
[8] Dee lin, Kurt shuler, Optimizing Enterprise- Class SSD Host Controller design with Arteris FlexNoC Network-on –chip Interconnect IP, arteries.com, 2016.
[9] B. Dhia, I. Mbarek, S. Hasnaoui, A SoC interconnection scheduling: design and implementation of a scheduler based on credited iSLIP algorithm, Int et al. (2008) 234–239.
[10] R. Deb, Rajrajan, Speed efficient implementation of round robin arbiter design using VERILOG, Int. J. Enhanc. Res. Sci. Technol. Eng. 2 (9) (2013) 1–9.
[11] C. Erika, A. Morais, M. Soares, NoC basics, in: G. Charles (Ed.), Reliability, availability and serviceability of networks-on-chip, Springer, New York, Dordrecht, Heidelberg, London, 2012, pp. 11–24. Adv. Optoelectron. Mater. 1 (2) (2013) 18–24.
[12] C. Fallin, C. Craik, O. Mutlu, CHIPPER: a low-complexity bufferless deflection router, in: IEEE 17th Int. Symp. Proc. Int. Conf. High-Performance Computer Architecture (HPCA), San Antonio, TX, USA, February 2011, pp. 144–155.
[13] C. Fallin, G. Nazario, Y. Xiangyao, et al., MinBD: minimally buffer deflection routing for energy efficient interconnect, in: Proc. Int. Conf. NOC S, Copenhagen, Denmark, May 2012.
[14] G. Batra, Z. Jacobson, S. Madhav, A. Queirolo, N. Santhanam, Artificial-intelligence hardware: new opportunities for semiconductor companies, 2018.
[15] G.E. Moore, Cramming more components onto integrated circuits, Electronics, 1965.
[16] G. Lacey, G.W. Taylor, S. Areibi, Deep learning on FPGAs: past, present, and future, arXiv preprint arXiv:1602.04283, 2016.
[17] I.I. Weber Santa Maria, F. Gehm Moraes PUCRS, L.L. de Oliveira Santa Maria, B. A. Carara, Exploring asynchronous end-to-end communication through a synchronous NoC, IEEE Transactions On Very Large Scale Integration (VLSI) Systems 24 (1) (2016).
[18] J. Nurmi, Processor design: System-on-chip computing for ASICs and FPGAs, Springer Science & Business Media, 2007.
[19] J. Hu, R. Marculescu, Application-specific buffer space allocation for Networks-on-Chip router design, in: Proc. ICCAD, 2004.
[20] K. Shuler, Re-Architecting SoCs for the AI Era, http://www.arteries.com.
[21] W.-G. Ho, K.-S. Chong, K.Z.L. Ne, B.-H. Gwee, J.S. Chang, Asynchronous-logic QDI quad-rail sense-amplifier half-buffer approach for NoC router design, IEEE Trans. Very Large Scale Integr. Syst. 26 (1) (2018).

Further Reading

[1] L. Benini, G.D. Micheli, Networks on chips: a new SoC paradigm, Computer 35 (1) (2002) 70–78.
[2] M. Ima, T. Van Chu, K. Kise, Tomohiro Yoneda I, The Synchronous vs. Asynchronous NoC Routers: An Apple-to-Apple Comparison between Synchronous and Transition Signaling Asynchronous Designs.
[3] N. Onizawa, A. Matsumoto, T. Funazaki, T. Hanyu, High-throughput compact delay-insensitive asynchronous NoC Router, 58 (8) (2011) 1933–1943.
[4] P. McGee, M. Agyekum, M. Mohamed, S. Nowick, A level- encoded transition signaling protocol for high-throughput asynchronous global communication, in: Proc. IEEE 14th Int'lSymp. Asynchronous Circuits and Systems, April 2008, pp. 116–127.
[5] P. Russell, J. Döge, C. Hoppe, T.B. Preußer, P. Reichel, P. Schneider, Implementation of an Asynchronous Bundled-Data Router for GALS NoC in the Context of VSoC, https://ieeexplore.ieee.org/xpl/conhome/7932334/proceeding.
[6] P.B. McGee, M.Y. Agyekum, M.A. Mohamed, S.M. Nowick, A level encoded Transition signalling protocol for high throughput Asynchronous global communication, in: 14th IEEE International Symposium on Asynchronous Circuits and Systems, 2009.
[7] P. Shahane, N. Pisharoty, Modified X-Y routing algorithm for mesh topology based NoC Router on FPGA, IET Circuits Devices & Systems, January 2019.
[8] R. Marculescu, J. Hu, U.Y. Ogres, Key research problems in NoC design: a holistic perspective, in: Third IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'05), 2005.
[9] Rantala V, T. Lehtonen, J. Plosila, Network on chip routing algorithms, Turku Centre for Computer Science, University of Turku, Turku, Finland, 2006, pp. 5–7.
[10] T. Moscibroda, O. Mutlu, A Case for Bufferless Routing in On-Chip Networks, ACM, 978-1-60558-526-0/09/06, ISCA'09, June 20–24, 2009.