ISSN: 0976-3104
ISSUE: Science and Technology

ARTICLE

FPGA BASED DESIGN AND ARCHITECTURE OF NETWORK-ON-CHIP ROUTER FOR EFFICIENT DATA PROPAGATION

Hirebasur Krishnappa Krutthika*, Anur Rangappa Aswatha

Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru-560 078, Karnataka, INDIA

ABSTRACT

Network on Chip (NoC) architectures are proposed to address the communication problems and various fabrication issues in block-based design existing in System on Chip (SoC). In this paper, an efficient NoC router architecture is proposed to route the data from one block to other structurally. For proper routing operation, the proposed architecture has bidirectional data transfer capabilities with design of an efficient controller to control and synchronize the overall operations. The FIFO is optimized for efficient handling of the data, which are transferred in different directions. To generate optimized hardware architecture, various optimization techniques are used at the architectural level. As a result, the overall performance of the architecture is improved than the existing architectures which further proved in the comparison table.

KEY WORDS
System on Chip, Network on Chip, FPGA Architecture, Router, XY Routing, FPGA.

Received: 7 Oct 2020
Accepted: 18 Dec 2020
Published: 19 Dec 2020

***Corresponding Author**
Email:
krutthika-ece@dayanandasagar.edu
Tel.: 91- 9886421452

INTRODUCTION

Network on Chip (NoC) architectures are commonly used in designing of high-speed integrator circuit architectures [1] that provides various advantages over commonly used for data transfer technique. Basically, a sophisticated controller is used to control the data flow, but in many real time circumstances, due to various fabrication issues, the problem arises with the controller. Also, many of the sophisticated controllers used dedicated path for data transfer which was decided before the data transmission and this fails to achieve optimized results whenever the traffic flow in that path was high. So there is a need for intelligence for the circuit to decide at real time to decide the exact path of data transfer. This situation can be overcome by using NoC, instead of a sophisticated controller where each module can take own decision of transferring the data depending upon the routing logic. This achieves better speed for the transfer of data and also it can be able to send the data through some other alternative path. The existing NoC hardware architectures have many limitations with effective handling of buffers for data transfer, hardware implementation and synchronization issues. In this paper, we have proposed an efficient VLSI architecture to implement a NoC where we have addressed the buffering issues.

Various types of NoC architectures are proposed for efficient implementation. Among these implementations some of works are as follows: In paper [2] the authors have proposed an architecture in which the Memory Management Unit (MMU) is implanted in Network-on-chip (NoC) and therefore the MMU functionality in the proposed NoC was implemented without any modifications in the processor design. It also supports for multiprocessing required for the dedicated high-speed processing applications. The proposed architecture consumed 16~23% more resources requirements. In paper [3] the authors have proposed Open NoC (Open-Source Light Weight deflection torus-based NoC) with PCIe-based communication controller. This method saved the event time and also has the higher clock performance. This also was adapted in many other high-speed communication technologies to operate at higher speed. The trade-off lies with disadvantage of more LUT are compared to Hoplite with respect to the implementation for AXI4 stream interface. In [4] the authors have proposed the design of NoC RingNet using FPGA. In this method, the architecture has consumed a smaller number of resources in terms of LUT's and Flip-flops. This method supported higher clock frequencies with portability for many applications. The distributed RAM available in FPGA's are used to implement buffers and an efficient virtual cut-through switching technique used to reduce latency. The FPGA based Multi-accelerators and Chip-Multiprocessor (CMPs) connected through the NoC is proposed in [5]. This proposed method is light-weight and scalable in characteristics. The proposed method has the high performance in terms of latency and throughput. The NoC simulator, titled Dual-Clock router micro-architecture (DuC-NoC) is elaborated in paper [6]. Furthermore, the authors have implemented two-layer configurable global interconnection to minimize the time overhead, to make an efficient trade-off between simulation time of the entire simulator and also the resource utilization. The advantage of this approach is that, it can simulate 2048 nodes virtualized Mesh Network. In paper [7] the authors have proposed the CDMA spreading codes which further increased to emphasize the capacity of CDMA Network-on-Chip (NoC) crossbars. The simulation results in the paper showed the sharp increase in the bandwidth utilization with a lesser amount of resource utilization and by reducing the facility consumption by 45. The proposed NoC based network architecture in [8] has modified router architecture which is optimized at architectural level. The integrated run-time solution for security and fault tolerance FPGA based System on Chip through digital signatures supported by x-network is proposed in [9]. By implementing the digital signature technique, the numbers of routers used for designing a network are reduced and also offer more flexibility. The network design has used 10% more LUT's compared to traditional methods for both security and fault tolerant features. In paper [10], the authors have proposed butterfly-based NoC's and ASIP (Application Specific Instruction Set Processor) architecture. The benefits of this method are, it supports modulation schemes from BPSK to 256-QAM for

IIOAB JOURNAL
ISSN: 0976-3104

any mapping style and eight stages single & double binary codes. The disadvantage is that, extra hardware cost is required to enable turbo demodulation.

Contributions: The main contributions of this research paper are,

- Design of an efficient controller to control and synchronize the overall operations in NoC.
- The existing FIFO architecture is modified to get proper status signal for the data buffering and also the FIFO is implemented using lesser hardware requirements.

This research paper is organized as follows, In Section 1, the methodology of the proposed architecture and its internal block diagrams are explained in detail, followed by Section 2, the implementation on FPGA and results are discussed and the lastly comparisons with existing techniques are given in Section 3.

MATERIALS AND METHODS

The proposed enhanced NoC architecture is shown in [Fig. 1]. The modules in the architecture consist of Concatenation module which formats the data as per the requirement, Controller module to control and synchronize all the operations and lastly the modified Memory blocks which stores the intermediate data. The data can be accepted from any direction and can be routed to any other direction using proposed an adaptive routing algorithm. The proposed architecture has four input and output ports in respective directions, which can be categorized as East, West, North and South. To perform proper routing operation, the data format should in the appropriate format, where the source and destination addresses is surrounded within the data. The main function of the concatenation block is to extract only the data. After the data extraction process, the source and destination addresses are analyzed to route the data to the specific port. In some specific cases, the data is stored temporarily by the modified memory unit.

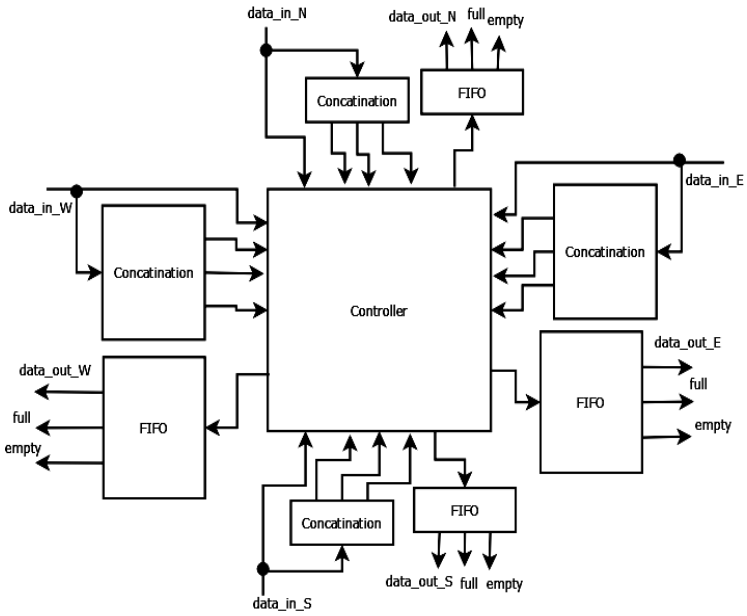

Fig. 1: The Proposed NoC Architecture

Data format

For proper routing purpose, it is essential for any routing algorithm to get the destination address of the data packet. As a result, the data format is used to implement the proposed architecture is shown in [Fig. 2]. In this case, the data is of 8 bits and the total address is of 4 bits where each field (i.e., X and Y) addresses consisting of 2-bits each. As and when the data packet arrives, the addresses are extracted and processed based on the X and Y addresses and the data will be delivered to appropriate port using the X-Y routing algorithm.

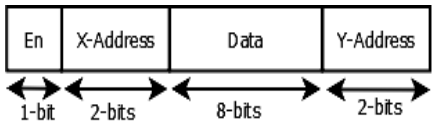

Fig. 2: Data Format

IIOAB JOURNAL
ISSN: 0976-3104

Concatenation

The extraction of destination address is done by concatenation block. The concatenation block is shown in [Fig. 3], which comprises of the Concatenation-1, Concatenation-2 and Concatenation-3 blocks respectively.

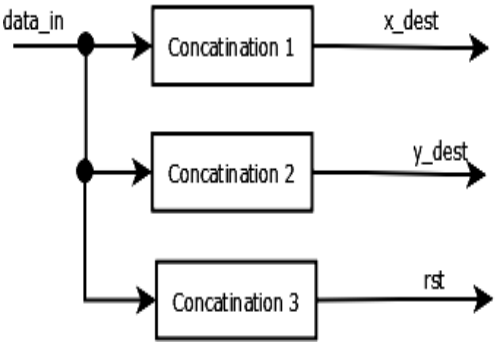

Fig. 3: Concatenation architecture

The Concatenation-1 and Concatenation-2 blocks are used to generate 'X' and 'Y' address of the destination router. The pseudo code is used to design Concatenation-1 is given in Pseudo code 1.

Pseudocode 1:

```
if (rst=0)
{
data_out =0
}
else if (Positive Edge clk)
{
X_dest=data_in (11 downto 10)
 }
end if
```

Similarly, the pseudocode for Concatenation-2 block is given in Pseudocode 2. Finally, the Concatenation-3 block is used to extract the 'rst' field data, which is used by the module as 'req' signal which acts as request signal to the appropriate router for establishing exact routing path. The pseudocode which is used to build this block is given in Pseudocode 3.

Pseudocode 2:

```
if (rst=0)
{
data_out =0
 }
else if(Positive Edge clk)
{
rst_dest=data_in (12)
}
end if
```

Pseudocode 3:

```
if(rst=0)
{
data_out =0

}
else if(Positive Edge clk)
{
Y_dest=data_in (1 downto 0)
 }
end if
```

Controller

The controller basically controls the data path for proper routing of data. It consists of MUX, DEMUX, D Flip Flop and modified X-Y routing algorithm respectively. The controller block is shown in [Fig. 4].

IIOAB JOURNAL
ISSN: 0976-3104

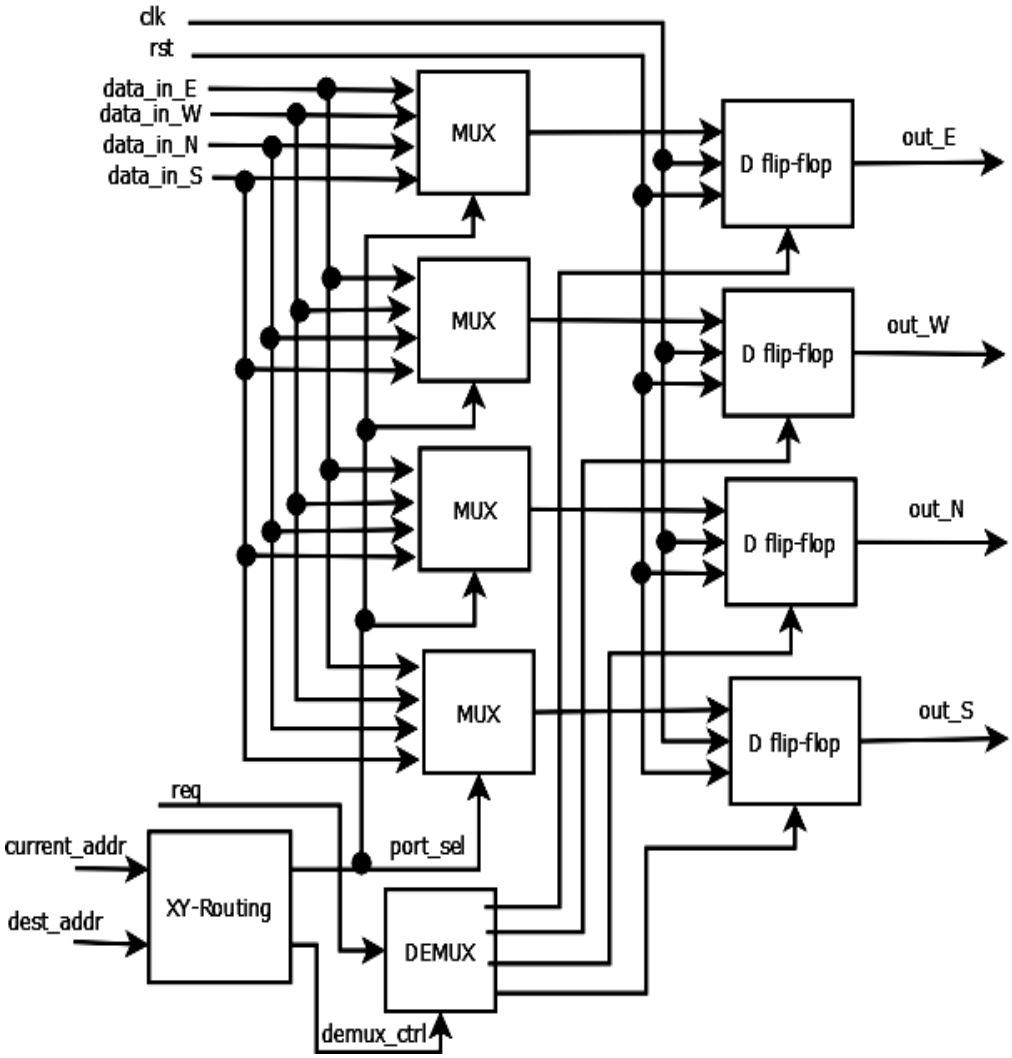

Fig. 4: Hardware architecture of proposed Controller

The controller block decides the proper routing path using modified X-Y routing algorithm. It generates the control signal for controlling the MUX and DEMUX blocks. For this purpose, the modified X-Y routing algorithm generates a "port-sel" and "demux-ctrl" signals where, the "port-en" signal is used to route the proper input data through "MUX" block and the "demux_ctrl" signal is used to route the request signal through "DEMUX" block, which act as reset switch for D-flip flops respectively. The output of the MUX and DEMUX blocks are connected to the respective D-flip flops for establishing proper routing path. The MUX and flip flops are properly synchronized with clock so that, the data mismatch does not occur.

X-Y Routing algorithm

The traditional routing algorithm used is the basic X-Y routing algorithm, [11] presented the data transmission between the cores. The main advantage of this algorithm is, it is to implement in hardware and also gives good trade-off between different parameters. The flowchart of the X-Y routing is given in [Fig. 5].

FIFO

The FIFO's are used to store data briefly. In this case, here it is used to synchronize the data temporarily, when the next router is busy in accepting the data packet. The architecture of modified FIFO is shown in [Fig. 6], which consists of FIFO and the indicator block where it is used to indicate the state of the FIFO (i.e., full or empty).

IIOAB JOURNAL
ISSN: 0976-3104

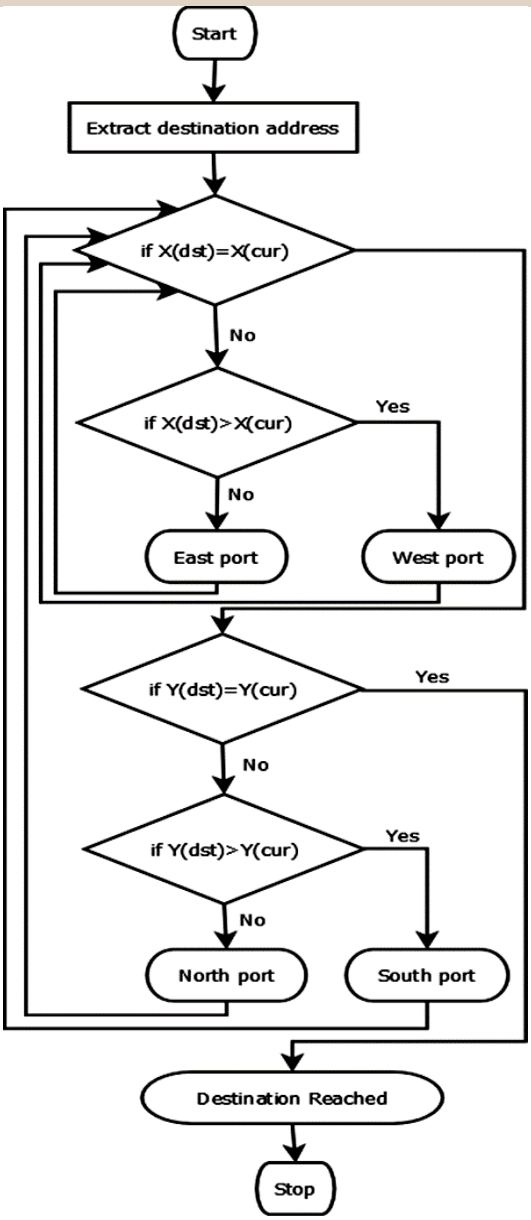

Fig. 5: X-Y Routing Algorithm

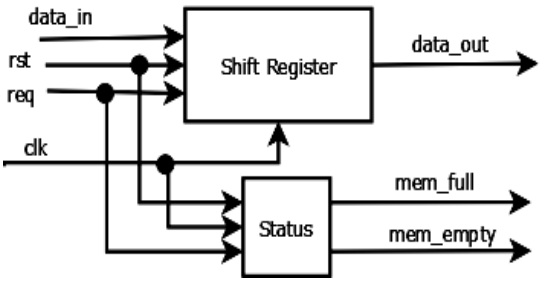

Fig. 6: Hardware architecture of modified FIFO

To optimize the indicator block, the Finite State Machine (FSM) module is implemented. The proposed FSM model implementation is done through Pseudo code 4.

Pseudocode 4:

```
if (rst=0 & (req=1 or req=0))
{
```

IIOAB JOURNAL
ISSN: 0976-3104

```
pointer=0
mem_empty=0
mem_full=0
 }
else
      {
if(positive edge clk)
      {
if(pointer<N)
      {
pointer=pointer+1
mem_empty=1
mem_full=0
      }
else
     {
pointer=pointer
mem_empty=0
mem_full=1
}
end if
}
end if
end if
```

RESULTS AND DISCUSSION

FPGA implementation

The proposed architecture is implemented in Digilent Atlys FPGA [12, 13] board through Xilinx ISE tool [14] and the coding is done using VHDL language [15]. The hardware utilization of the proposed NoC architecture with the sub modules are given in [Table 1].

Table 1: Hardware utilizations of proposed architecture

Parameters	Concatenation	XY-Routing	FIFO	Values
FPGA	Spartan-6 (XC6SLX45-3CSG324)			
Slice Registers	0	0	17	122
Slice LUTs	2	20	23	141
LUT-FF Pairs	1	11	17	103
Maximum Frequency (MHz)	-	-	297.987	373.413

The RTL schematic gives the gate level connection, the illustration of the enhanced design in terms of generic symbols, such as adders, multipliers, counters, AND gates, and OR gates, is generated by the Xilinx tool is shown in [Fig.7].

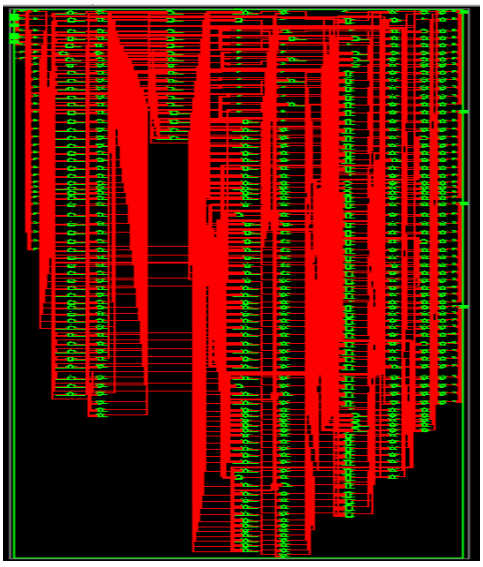

Fig. 7: RTL schematic of the proposed NoC architecture

IIOAB JOURNAL
ISSN: 0976-3104

Similarly, the generated technology schematic is shown in [Fig. 8] such as LUTs (Look Up Tables), Carry logic and I/O buffers and other components are shown.

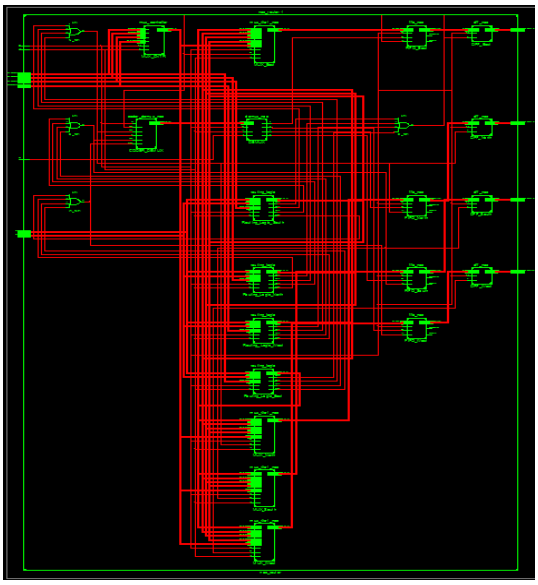

Fig. 8: Technology schematic of the proposed NoC architecture

Simulation

The timing diagram of proposed NoC router is shown in the [Fig. 9] where timing of each tasks are shown which takes one clock cycle.

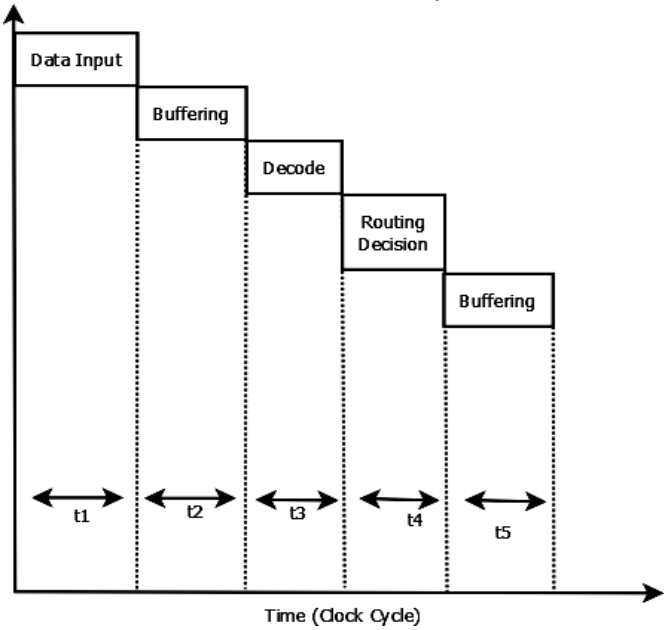

Fig. 9: Timing diagram of the proposed NoC

The simulation results of the proposed NoC router is shown in the [Fig. 10] to [Fig. 13] for different cases given below.

Case 1

The simulation result of NoC router in [Fig. 10] transfers the data at West port through proper destination address.

IIOAB JOURNAL
ISSN: 0976-3104

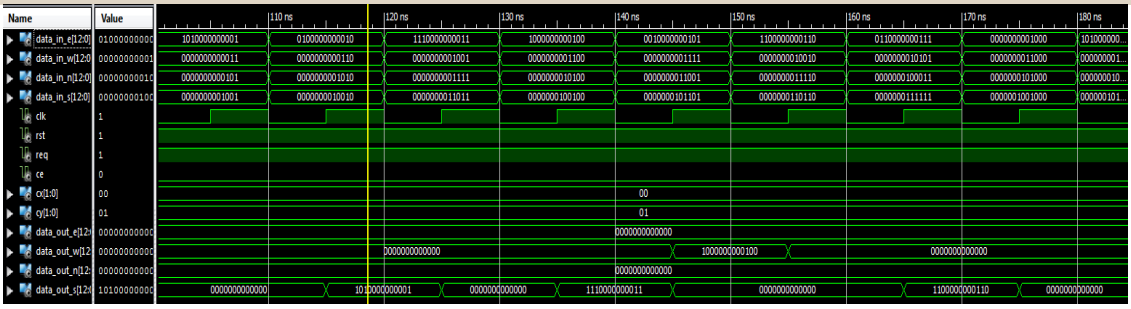

Fig. 10: Simulation result of the proposed NoC transferring data at West Port

Case 2

The simulation result of NoC router in [Fig. 11] transfers the data at East port through proper destination address.

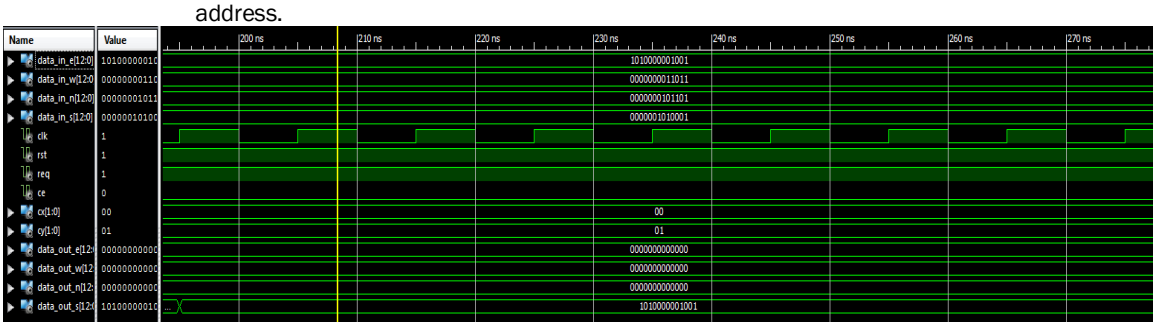

Fig. 11: Simulation result of the proposed NoC transferring data at East Port

Case 3

The simulation result of NoC router in [Fig. 12] transfers the data at North port through proper destination address.

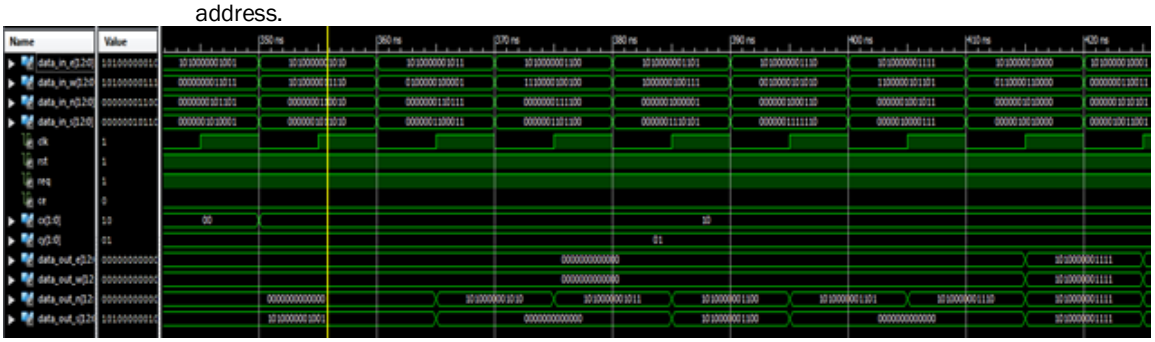

Fig. 12: Simulation result of the proposed NoC transferring data at North Port.

Case 4

The simulation result of NoC router in [Fig. 13] transfers the data at East port through proper destination address.

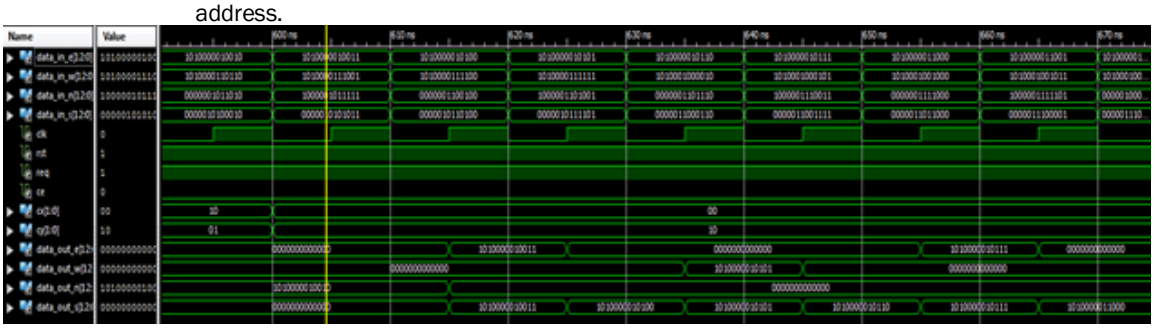

Fig. 13: Simulation result of the proposed NoC transferring data at South Port.

IIOAB JOURNAL
ISSN: 0976-3104

The comparison of the proposed NoC architecture with the existing NoC architectures are compared with respect to the hardware parameters is shown in [Table 2]. In [16] the authors have proposed the FSM based model. Here, with the growth in the number of redundant nodes, the overall hardware utilization in the architecture increases. This intern has increased the Look Up Table (LUT)- Flip Flop (FF) pairs and also the operating frequency. The increase in the LUT-FF pairs occupies more area and power. Similarly, in paper [17] the authors have presented the architecture which uses, large area due to the use of un-optimized FIFO architecture. Here also, when compared to paper [16], the LUT-FF pairs have been reduced minimally. But when compared to our proposed NoC architecture, both these parameters are addressed effectively to achieve optimization with respect to the area and throughput.

Table 2: Comparison table of the existing and proposed NoC architectures

Parameters	Raaed Faleh, et al. [16]	Santrupti M, et al. [17]	Proposed Architecture
FPGA	Spartan-3A	Spartan-6	Spartan-6
Slice Registers	-	224	122
Slice LUTs	-	162	141
LUT-FF Pairs	1861	128	103
Maximum frequency (MHz)	82.734	292.987	373.413

CONCLUSION

The proposed hardware architecture performs the routing of the data efficiently using NoC methodology which performs; various optimization techniques using FSM modelling are realized. The optimized NoC architecture is compared with other existing architectures to validate the result and there is a significant increase with respect to the area and throughput are realized. The future work includes the design and techniques used for 2D NoC architectures can be upgraded to 3D NoC architectures as the layout and algorithms used for data propagation is unique and complex. Our main goal for 3D NoC is to concentrate on the performance parameters of NoC architectures.

CONFLICT OF INTEREST

There is no conflict of interest.

ACKNOWLEDGEMENTS

This work was carried out in the Department of Electronics and Telecommunication Engineering department, Dayananda Sagar College of Engineering Bengaluru.

FINANCIAL DISCLOSURE

None.

REFERENCES

[1] Hyeonguk J, Kyuseung H, et al. [2019] MMNoC: Embedding Memory Management Units into Network-on-Chip for Lightweight Embedded Systems, IEEE Access, 4: 80011 – 80019.

[2] Reddy KS, Vipin K. [2019] OpenNoC: An Open-Source NoC Infrastructure for FPGA-Based Hardware Acceleration. IEEE Embedded Systems Letters, India, 11(4): 123 – 126.

[3] Jakub S, Adam Ł, et al. [2019] Ring Net: A Memory-Oriented Network-On-Chip Designed for FPGA. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 27(6):1284 – 1297.

[4] Lin Z, Sinha S, et al. [2018] Scalable Light-Weight Integration of FPGA Based Accelerators with Chip Multi-Processors, IEEE Transactions on Multi-Scale Computing Systems, 4(2): 152 – 162.

[5] Hadi MK, Kimia Z A, et al. [2018] DuCNoC: A High-Throughput FPGA-Based NoC Simulator Using Dual-Clock Lightweight Router Micro-Architecture, IEEE Transactions on Computers. 67(2): 208 – 221.

[6] Ahmed KE, Mohamed R, et al. [2017] Overloaded CDMA Crossbar for Network-On-Chip, IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 25(6):1842 – 1855.

[7] Abdelfattah SM, Bitar A, et al. [2017] Design and Applications for Embedded Networks-on-Chip on FPGAs, IEEE Transaction on Computers. 66(6): 1008 – 1021.

[8] Vijayalaxmi J, Satish B. [2018] An Efficient FPGA Based NoC Architecture for Data Communication, International Journal of Advanced Computer Research, 8(39):335-341.

[9] Raza AJ, Amer B, et al. [2017] Heterogeneous Multi-ASIP and NoC-Based Architecture for Adaptive Parallel TBICM-ID-SSD, IEEE Transaction on Circuits and Systems. 65(3): 259 – 263.

[10] Yao C, Swathi T, Gurumani, et al. [2016] FCUDA-NoC: A Scalable and Efficient Network-on-Chip Implementation for the CUDA-to-FPGA Flow IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 24(6): 2220 - 2233

[11] Priti S, Narayan P. [2019] Modified X–Y routing for mesh topology based NoC router on field programmable gate array, IET Circuits Devices & Systems, Devices and Systems. 13(3): 391.– 398.

[12] https://reference.digilentinc.com/reference/programmable-logic/atlys/start. (Accessed on 15 May, 2020)

[13] Satish B, Sayantam S, et al. [2018] FPGA Implementation of Optimized Karhunen–Loeve Transform for Image Processing Applications. Journal of Real-Time Image Processing, 17(6):357–370.

[14] https://www.xilinx.com/support/documentation/sw_manuals/xilinx14_1/ise_tutorial_ug695.pdf (Accessed on 18 May, 2020)

[15] Roth CH [1997] Digital Systems Design Using VHDL, PWS Publishing Company, Boston, MA, USA

[16] Raaed F, Raaed H, et al. [2018] Hardware Implementation of NoC based MP-SoC Prototype using FPGA, International Journal of Applied Engineering Research, 13(7): 5443-5451.

[17] Santrupti SM, Sayantam S, et al. [2016] FPGA Implementation of High Speed and Low Area Four Port Network-On-Chip (NoC) Router, IOSR Journal of VLSI and Signal Processing (IOSR-JVSP), 6(6): 52-57.