Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

IMPLEMENTATION AND ANALYSIS OF CONGESTION PREVENTION AND FAULT TOLERANCE IN NETWORK ON CHIP

KRUTTHIKA H. K
Assistant Professor, Electronics and Communication Engineering,
Dayananda Sagar College of Engineering, Bangalore-560078, Karnataka, India.
Email: krutthika-ece@dayanandasagar.edu

Dr. A.R. ASWATHA
Professor and Head, Electronics and Telecommunication Engineering,
Dayananda Sagar College of Engineering, Bangalore-560078, Karnataka, India.

Abstract: As semiconductor technology has evolved, the convergence of a large series of processing cores on a single silicon chip known as a System on Chip (SoC) [1] has expanded substantially. With the increase in the applications for modern technology, the amount of computational power that can be handled on a single chip has greatly expanded. The Network on Chip (NoC) has successfully replaced the traditional bus-based mode of transportation in order to satisfy the SoC connectivity standards. NoC is an on-chip communication strategy aimed at reconciling inter-core and system messages in tandem. The NoCs are also scalable since, it overcomes network congestion, area utilization and operates at higher frequencies. Congestion is also a critical part of any wireless networks against packet loss and latency. This motivated us to design and implement a FPGA based router architecture for Network on Chip. The goal of this research paper is to efficiently propagate data packets to their intended destination and thereby, the network parameters such as area and transmission rate are considered to improve the performance of the network.

Keywords: FPGA Implementation; Network-On-Chip Architecture; Round Robin Scheduling, Virtual Channel Allocator.

1 Introduction

Network-on-Chip (NoC) has evolved as an effective communication mechanism for Chip Multiprocessors. As the number of intensive applications increases, so would the communication between cores. This causes network congestion and degradation in the performance of the network. Congestion management is a critical issue in any network. An efficient Network-on-Chip (NoC) architecture with dynamic routing, congestion control and fault tolerance are proposed in this research paper. The coding is done in the standard System Verilog language and implemented on the Digilent Zybo Z7-10 FPGA board. The Round robin scheduling block determines, which packet should be routed next based on the priority of the packets. The dynamic memory block is presented in this architecture that stacks all data packets arriving from neighboring routers and accepts or discards additional data requests based on the status of the memory block. Following that, the packets will be forwarded to the correct destination using the X-Y routing algorithm and also based on the acknowledgment reception method proposed in this research work. This research paper is organized as follows.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

Section 2 discusses the proposed architecture and its related blocks in detail followed by section 3, fault tolerance techniques adopted for the two-dimensional architecture is explained. In the final inference, the FPGA implementation and its results are discussed in section 4.

2 Proposed architecture

The proposed Network-on-Chip (NoC) router architecture is implemented using a bottom-up technique [2] where, each module being designed in the System Verilog [3] language. The communication between different NoC modules through up-stream and down-stream ports are shown in Figure 1 where, the East, West, North, South and local ports are attached to the router.

For efficient packet routing between the routers, many handshaking signals are exchanged between the up-stream and down-stream ports. The data packets arrive in any direction and the router channels these data packets efficiently. The signals is_valid is used to check the status of FIFO/Buffer, which indicates the availability of the next router and if available an acknowledgement signal will be received. The is_on_off signal is used as a start signal and is_allocatable allocates the location in FIFO/Buffer where, the data has to be stored.

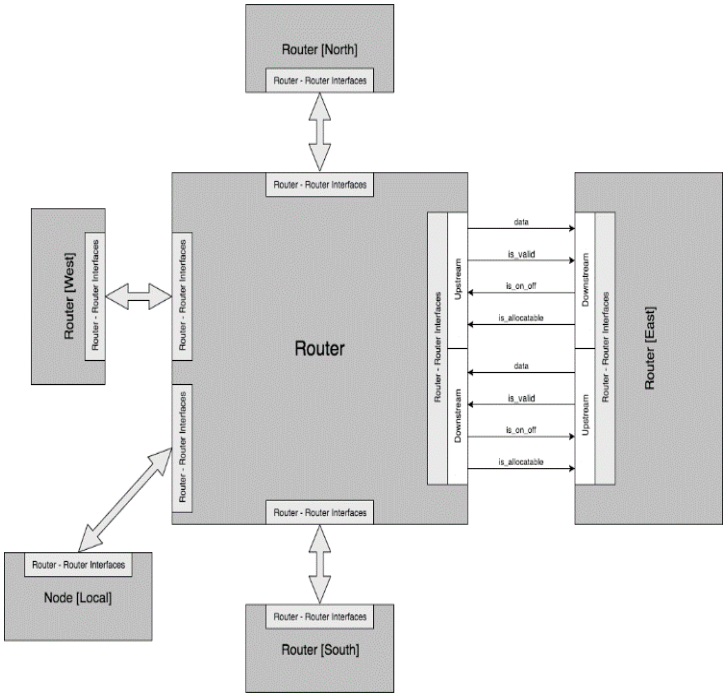

Figure 1: Proposed internal architecture of a single NoC

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

The fault-tolerant NoC architecture is a novel technique introduced in this research that allows the data packets to be routed around the faulty or defective routers. The objective of this method is to allow, routers to communicate without causing any communication problems.

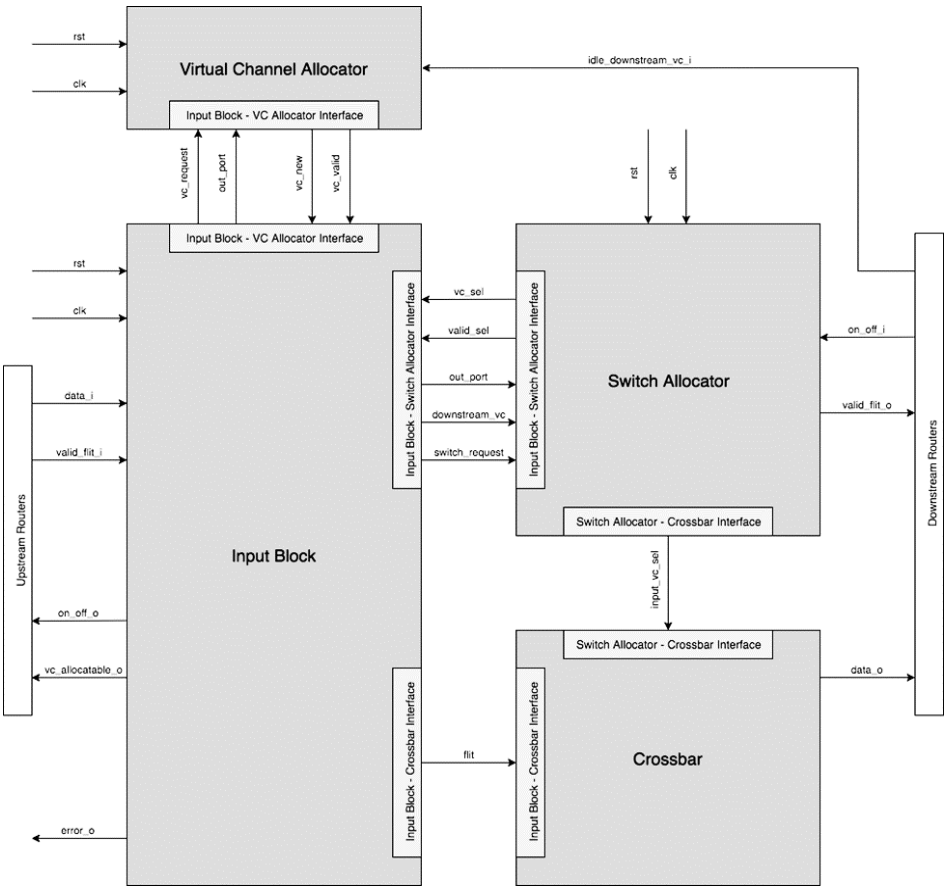

Figure 2: Internal Architecture of a Single NoC Router

The Figure 2 shows, the projected internal architecture of a single NoC router which comprises of Input block, Virtual Chanel Allocator (VCA), Switch allocator (SA) and Crossbar module. The data arrives at any of these input ports and stored in the input block's small memory before, being mapped into the virtual memory through the Virtual Chanel Allocator.

The Switch Allocator and Crossbar module now transmits, the data packet to the next router based on the priority and availability of the channels. If the virtual channel memory of the next router is free, then only the data packet will be accepted;

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

otherwise, it will be denied. If the next router's VC channels are busy then, memory status full will be set high. The X-Y routing algorithm is employed to generate a redundant path, ensuring that the packets reach its destination on time. The ability to identify the memory status of immediate neighbors allows for quick and precise congestion identification, which intern leads to a congestion prevention mechanism.
The Internal architecture of a single NoC Router blocks in the Figure 2 are explained in the following sections.

A. Input block

The Input block's major function is to receive data and other requests from the neighboring router. It is referred to as upstream signals in the Figure 2. This block

comprises tiny memory module and a Controller. The virtual memory module is attached to this tiny memory module. This block serves as a temporary storage location for intermediate data. An additional signal is used to indicate the memory status signal and the acknowledgement signal. The controller design is built using counter related logic [4].

B. Virtual Channel Allocator:

The Virtual Channel Allocator (VCA) module produces a request matrix as an input to the input-first allocator block, the matrix is produced based on the availability of virtual channels at the destination downstream input port. The VCA module produces the identifiers for the virtual channels at downstream routers for the input buffers. The internally computed grant matrix is activated only if, the contention [5] is resolved in the network.
The basic operation of the circular buffer is shown in the Figure 3, which generates virtual channels. The data enters into the VCA's circular buffer and the packets are enqued in the circular fashion. The 'head' and 'tail' will be incremented and decremented accordingly. Finally, based on availability of the channels in the next block, the data will be dequed from the circular buffer for further processing.
The overall architecture is simpler due to the absence of the sophisticated address generator circuitry.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

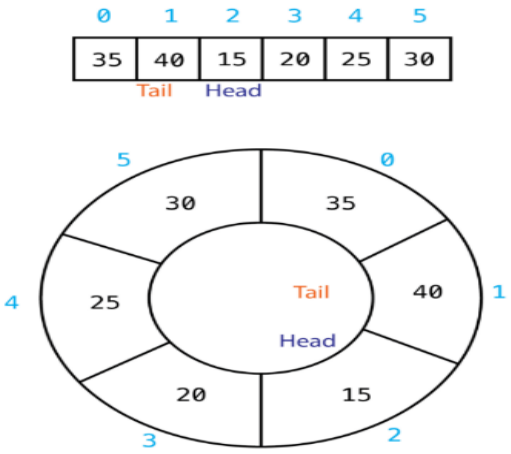

Figure 3: Circular Buffer architecture

C. Switch Allocator block

By encapsulating a separate input-first allocator and thereby delivering control signals to the crossbar module the input buffers and the switch allocator module eliminates the conflict between input buffers for access to the crossbar switch. Access to the shared crossbar switch depends on, the previous grant signals for the resources (with a Round-Robin policy [6]) and also depends upon the availability in the downstream virtual channel.
The proposed router in the Figure 4, consists of the distributive Round Robin Arbiter (RRA) architecture with five requests and five grant signals. Only mesh-based NoC router architectures are acceptable for the Round Robin Arbiter (RRA). When the number of requests is minimal, the network's performance is high. When the RRA is

overloaded with requests, the latency and area occupancy increase. This leads to the performance degradation in the network. In order to overcome these limitations, an enhanced RRA arbiter module has been proposed which mainly has two components the counter and the multiplexer.
The counter acts as select line to the Multiplexer (nx1) in a circular manner and the enable/disable signal of the counter is activated by the grant signal of multiplexer hence, the counter is dependent on the multiplexer. The multiplexer's purpose is to transport data from the input to the output ports, which is dependent upon on counter's selection lines. Because of its distributive nature, the arbiter functions well when the number of inputs is high and occupancy is low.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

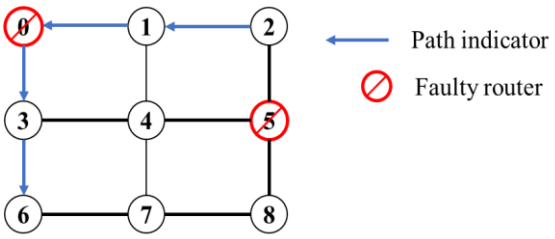

Figure 4: Proposed Round Robin Arbiter (RRA) architecture

D. Crossbar

Packets can be sent from each input port to the router's output port. The packets from the input port are propagated to the switch allocator's auxiliary module through the output terminal [7].

3 Fault Tolerance Technique

The fault tolerance technique presented in this work is distinguished by the use of additional data paths that allows for routing around defective routers. The goal of this strategy is to allow communication between the non-functional routers in an effective manner. An acknowledgement from the receiver is essential to understand whether, the packet has reached the destination successfully or not. This further indicates that, the router is free and ready to accept a greater number of packets. In case, any malfunction like Stuck at zero (SA-0) and Stuck at one (SA-1) in the receiving router, an acknowledgement is not sent to the transmitting router. This proposed technique avoids the extra circuitry required for checking the status of the router.

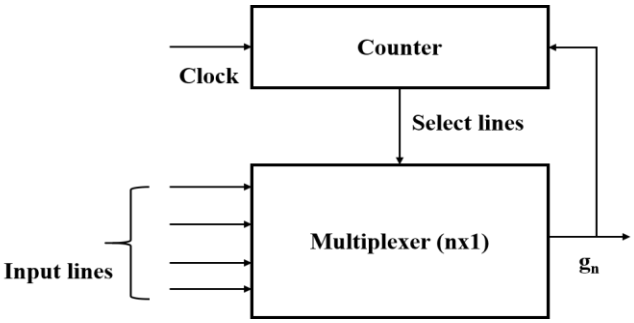

Figure 5: Example: Here the routers 0 and 5 are faulty, packets are routed from router 2 to router 6.

The proposed technique in the Figure 5 uses a novel concept to mitigate loss of packets due to faulty routers present in the network. The routers 0 and 5 are the faulty

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

routers in this scenario, even with two failed routers, the packets are delivered to its intended destination successfully. It is evident that the proposed architecture supports the faulty routers during the packet's transmission and also when various flits of the packet that are in transmission are stored in the routers buffer. In this instance, each router must include an error control mechanism to allow the packet to be re-transmitted.

The X-Y routing algorithm is modified to appropriately fit into the proposed architecture. The algorithm leads the flits along the axis X at first and when they reach the same column as the router's destination, the transition to the axis Y is accomplished. Suppose, that router incharge is transmitting data from the X to the Y axis and the algorithm's differential is apparent if the route fails. The suggested acknowledgement-based routing algorithm reconfigures data routes to avoid the faulty router and the flit is also sent to the router in the adjacent column but then again, on the same path. Later, this router takes responsibility of directing the packets/flits to the Y axis.

The X-Y forwarding method computes the route during congestion and fault tolerance. The source node addresses in Pseudocode-1 are sx and sy, and the destination node addresses are dx and dy, respectively. The input and output ports are defined by the 'inPort' and 'OutPort'. If the acknowledgement is 1, then the current node addresses cx and cy are checked with dx and dy. If the current and destination addresses are same, the information is delivered to the local port from which, it is forwarded to the next router.

If the target node's x-axis address is greater than the existing node's x-axis address, the message is routed to the East channel; otherwise, the information is delivered to the West port. When the packet has completed its route along the X-axis, the Y-axis addresses are verified; if the destination node's Y-axis address is larger than that of the existing node's Y-axis, the message is passed towards the North port; otherwise, it is transmitted to the South port. These cases are considered when the Ack value is 1.

The case is different if, the acknowledgement is 0, then the current node addresses cx and cy are checked with dx and dy. If the current and destination addresses are same, the information is delivered to the local port, from which it is forwarded to the next router. If the target node's x-axis address is greater than the existing node's x-axis address, the message is routed to the West channel; otherwise, the information is delivered to the East port. When the packet has completed its route along the X-axis, the Y-axis addresses are verified; if the destination node's Y-axis address is larger than that of the existing node's Y-axis, the message is passed towards the South port; otherwise, it is transmitted to the North port. The Pseudocode 1 is given below,

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

Pseudocode 1:

```
Source node: (sx,sy); Destination node(dx,dy);
Existing router: (cx,cy); Input port: inPort;
Output port: outPort;
if Ack==1;
if (cx == dx) & (cy == dy) then
Exit after delivering the message to the local port.
else
if dx > cx then
outPort  EAST
otherwise
if dx < cx then
outPort  WEST
else
if dx == cx then
if dy > cy then
outPort  NORTH
else
if dy < cy then
outPort  SOUTH
else if Ack=0
if (cx == dx) & (cy == dy) then
Exit after delivering the message to the local port.
else
if dx > cx then
outPort  WEST
else
if dx < cx then
outPort  EAST
else
if dx == cx then
if dy > cy then
outPort SOUTH
else
if dy < cy then
outPort NORTH
```

4. Results and Discussion

The proposed NoC architecture's performance is assessed using simulation and synthesis results, as well as comparisons to certain existing methodologies. The simulation results are obtained using the Xilinx Vivado tool, while the synthesis results

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

are obtained using Vivado 2018.3 [8].

4.1 Timing analysis: -

The Figure 6 depicts, the timing analysis of the suggested architecture in which, each operation is executed differently and requires different clock cycles. Each task requires one clock cycle for each intermediate operation, resulting in a total of six clock cycles for data transmission between routers

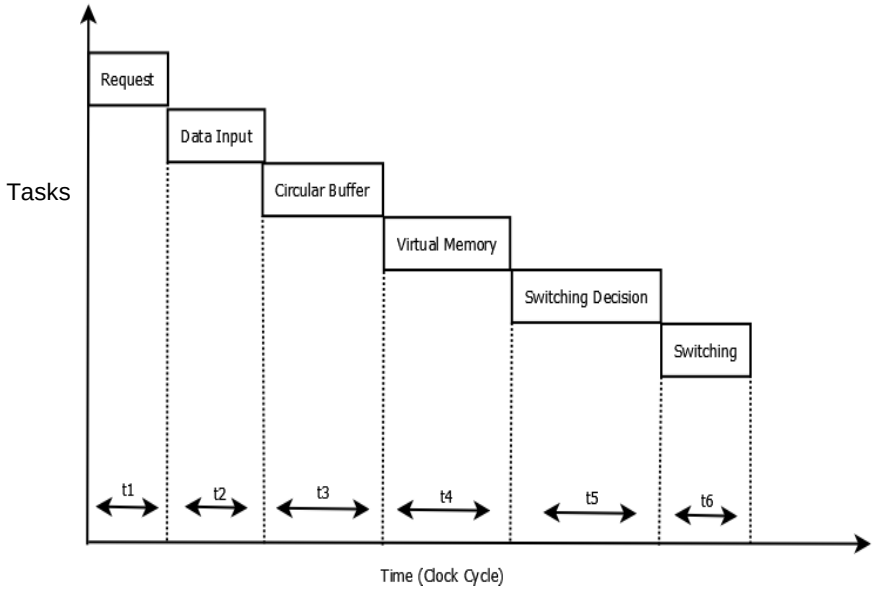

Figure 6: Timing analysis of the proposed NoC

4.2 Simulation results

The simulation results of the proposed NoC router architecture are shown from Figure 7 to Figure 12 for different states are given below.

A. Data transfer to the East Port:

The simulation waveform of the proposed architecture is shown in the Figure 7 where, the current router address is Cx=0 and Cy=0 and destination address are Cx=0 and Cy=1 respectively.

The information is sent at 500ns and it is sent out through the East output port at 500ns successfully. The other variables are clock-‘clk’, reset-‘rst’, chip enable-‘ce’, request signals to transmit through ports i.e., e_req, w_req, n_req, s_req and l_req respectively. The output ports are data_out_e for East, data_out_w for West,

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

data_out_n for North, data_out_s for South and data_out_l for Local.

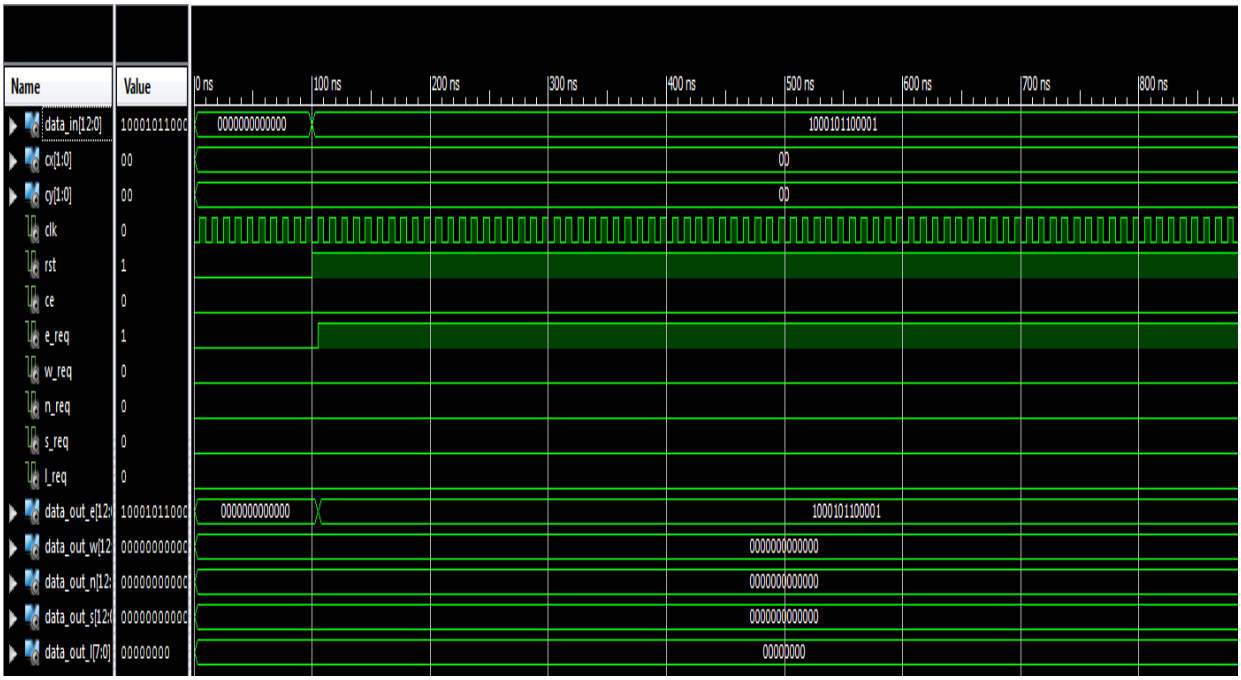

Figure 7: Simulation waveform to data transfer to the East Port

B. Data transfer to the West Port:

The simulation waveform of the proposed architecture is shown in the Figure 8 where, the current router address is Cx=0 and Cy=2 and destination address are Cx=0 and Cy=1. Now, depending upon the routing algorithm, the data present in the virtual memory is routed to the West port after multiple iterations.
The information is entering through the data_in [12:0] port which is of totally 13bits according to the format. The information is sent at 500ns and it is sent out through the West output port at 500ns successfully.

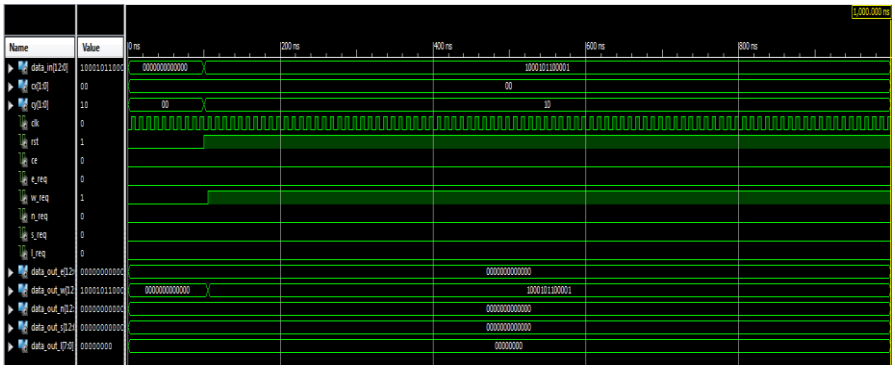

Figure 8: Simulation waveform to data transfer to the West Port

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

C. Data transfer to the North Port:

The simulation waveform of the proposed architecture is shown in the Figure 9 where, the current router address is Cx=1 and Cy=1 and destination address are Cx=0 and Cy=1. Now, depending upon the routing algorithm and after repetitive iterations, the data present in the virtual memory is routed to the North port. The data is received via the data in [12:0] port. The data is transferred at 500ns and effectively sent out via the East output port.

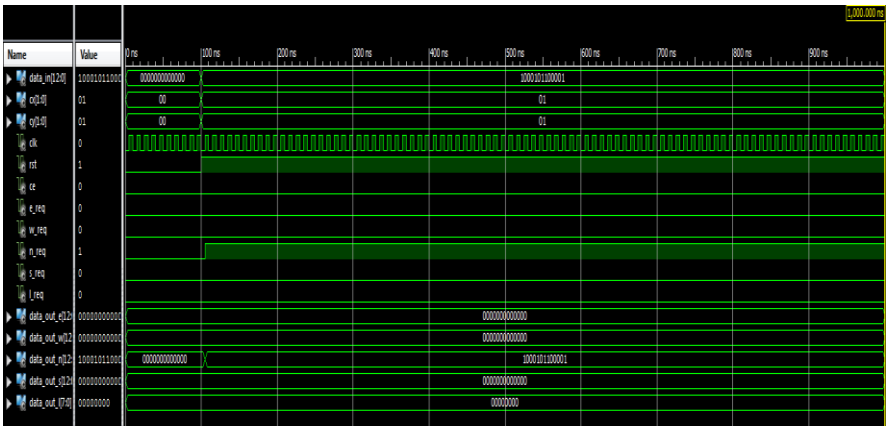

Figure 9: Simulation waveform to data transfer to the North Port

D. Data transfer to the South Port:

The proposed architecture's simulation waveform is presented in Figure 10 where, the current router address is Cx=0 and Cy=0 and the destination address is Cx=1 and Cy=0. The data in the virtual memory is now transmitted to the south port after many iterations.

The data is received through the data in [12:0] port, which has a total of 13 bits in length. The data is transferred at 500ns and successfully sent out over the East output port at 500ns.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

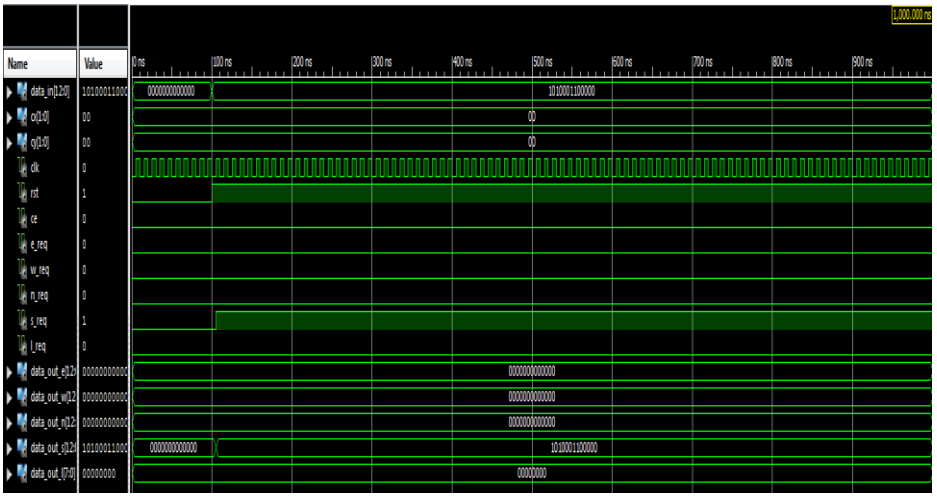

Figure 10: Simulation waveform to data transfer to the South Port

E. Data transfer to the Local Port:

The simulation waveform of the proposed architecture is shown in the Figure 11 where, the current router address is Cx=0 and Cy=0 and destination address are Cx=0 and Cy=0. The Local port is utilized when, the data belongs to the same router. Now, depending upon the routing algorithm, the data is routed to the local port itself. The data '01100000' is sent to the local port.

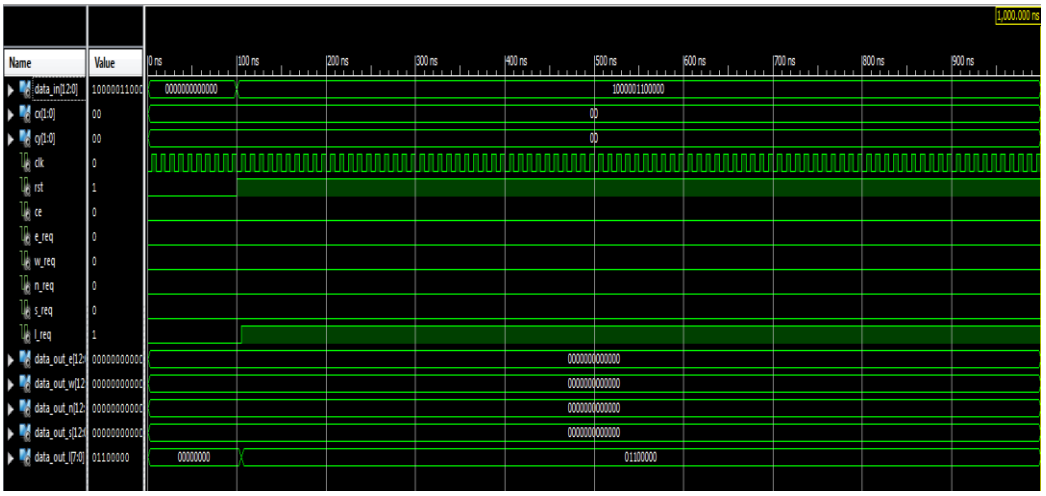

Figure 11: Simulation waveform to data transfer to the Local Port

4.3 Verification

The congestion scenarios cannot be shown in the simulation waveforms due to various constraints. As a result, the congestion conditions are detected using a generic script and is shown in the Figure 12.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

The whole router architecture is verified using the test-bench technique, which is written using the System Verilog language. Random source and destination address packets are generated using the random operator [9] in the System Verilog tool to improve test coverage.

The output from the router block is compared with the actual output and depending on this comparison, the test results are generated. The snapshot of some random test result is shown in the Figure12 where, different packet IDs, source and destination address are used for higher degree of verifications. From the Figure 12, it can be seen that different data packets are fed into the network in different network environments (i.e., normal and congested).

The text file results are generated after the simulation. The congestion avoidance is used when, the packet encounters the congestion in the NoC network. Different ports are detected to determine their availability in order to efficiently propagate the data packets. The packets are sent to the next router if the ports are available.

In Figure 12 consider, in the first scenario the port number 20 is examined; if the port 20 is unavailable, the next ports will be sensed simultaneously. Later, the port number 25 is identified and it is available at this time. Then, the data is immediately pushed into this port. If the ports are not free or the network is congested then an alternate route is selected and the data will be processed next.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

```
#    }
# }
WARNING: Simulation object /tb_all_modules/flit_queue was not traceable in the design for the following reason
Vivado Simulator does not support tracing of System Verilog Dynamic Type object.
# run 1000ns
Packet size:          4
Time                  15,port_num: 1 total time:          0, Route: Normal, Status: PASSED          0
Time                  15,port_num: 1 total time:          0, Route: Alternate, Status: PASSED          0
push                  15
Time                  20,port_num: 1 total time:          1, to read          1, timer          0
Time                  25,port_num: 1 total time:          1, Route: Normal, Status: PASSED          1
Time                  25,port_num: 1 total time:          1, Route: Alternate, Status: PASSED          1
push                  25
Time                  30,port_num: 1 total time:          2, to read          2, timer          0
Time                  35,port_num: 1 total time:          2, Route: Normal, Status: PASSED          2
Time                  35,port_num: 1 total time:          2, Route: Alternate, Status: PASSED          2
push                  35
Time                  40,port_num: 1 total time:          3, to read          3, timer          0
Time                  45,port_num: 1 total time:          3, Route: Normal, Status: PASSED          3
Time                  45,port_num: 1 total time:          3, Route: Alternate, Status: PASSED          3
push                  45
Time                  50,port_num: 1 total time:          4, to read          4, timer          0
Time                  55,port_num: 1 total time:          4, Route: Normal, Status: PASSED          4
Time                  55,port_num: 1 total time:          4, Route: Alternate, Status: PASSED          4
Time                  60,port_num: 1 total time:          5, to read          4, timer          0
Time                  65,port_num: 1 total time:          5, Route: Normal, Status: PASSED          4
Time                  65,port_num: 1 total time:          5, Route: Alternate, Status: PASSED          4
Time                  70,port_num: 1 total time:          6, to read          4, timer          0
Time                  75,port_num: 1 total time:          6, Route: Normal, Status: PASSED          4
Time                  75,port_num: 1 total time:          6, Route: Alternate, Status: PASSED          4
Time                  80,port_num: 1 total time:          7, to read          4, timer          0
Time                  85,port_num: 1 total time:          7, Route: Normal, Status: PASSED          4
Time                  85,port_num: 1 total time:          7, Route: Alternate, Status: PASSED          4
Time                  90,port_num: 1 total time:          8, to read          4, timer          0
Time                  95,port_num: 1 total time:          8, Route: Normal, Status: PASSED          4
```

Figure 12: Text file generated for the congestion control

4.3 Synthesis results

The synthesis result shows, the entire schematic implementation and its utilization in terms of Slices, LUT's and delay on the targeted FPGA Board. In our case, Xilinx Zybo Z7-10 board is used to validate the results.

The RTL diagram shows, the gate level connections for the entire module. As shown in Fig.13, the Xilinx tool generated the enhanced design using generic symbols such as adders, multipliers, counters, AND gates and OR gates.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

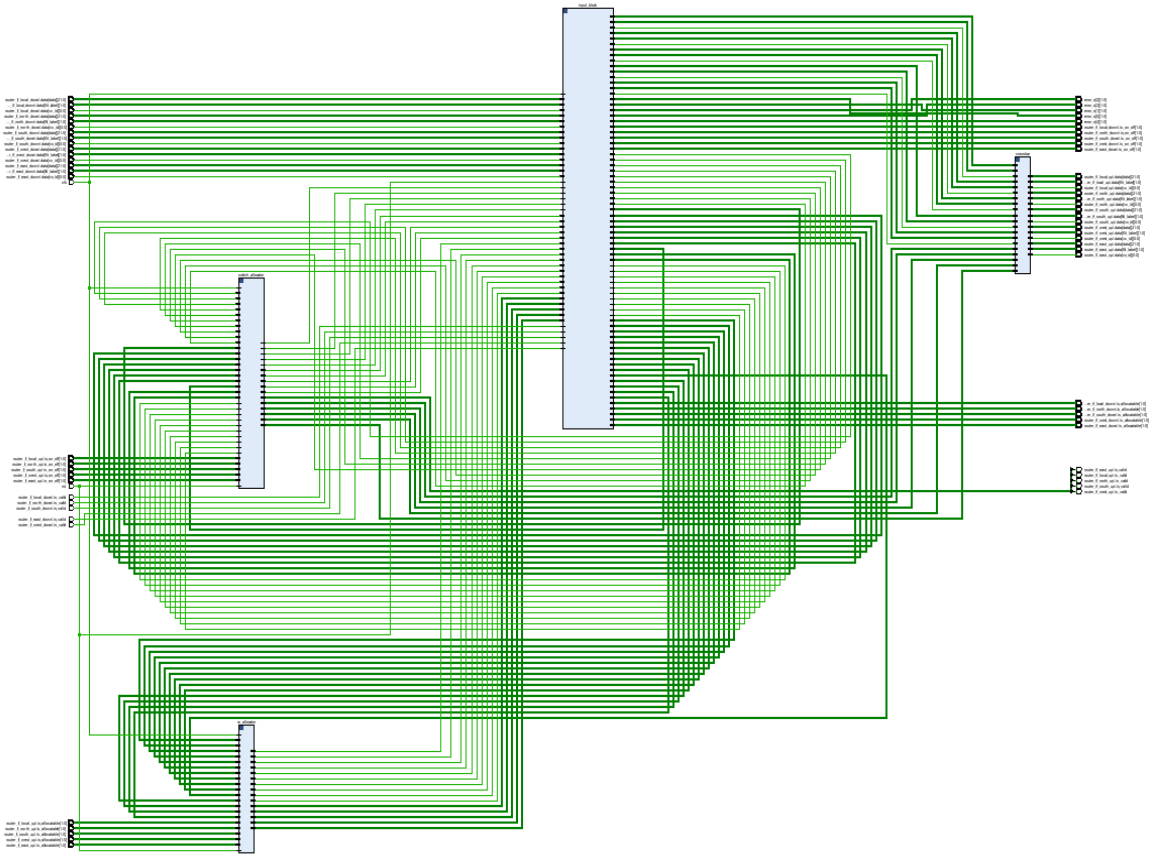

Figure 12: RTL Schematic of the Proposed NoC Router

4.4 FPGA Implementation

The proposed architecture is coded in the system verilog language and implemented on a Xilinx Zybo Z7-10 FPGA board. The suggested architecture's hardware utilizations are shown in Table 1 and the Vivado 2018.3 tool is utilized for synthesis.

Table 1: Hardware utilization of the proposed router

Parameters	**Hardware Utilizations**	
	Router	**Network**
Slice LUT	2874	11093
Flip-flop	2195	8826
BUFG	1	1

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

4.5 Comparison with the existing techniques

With the approaches described by Jayaprakash. M et al., [10], the suggested architecture is compared with existing architecture in terms of hardware utilization in Table 2. It is clearly visible that the number of Flipflops and Slice

LUTs are reduced based on the optimization techniques used in the architecture.

Table 2: Hardware utilization of the proposed router with the existing router

Parameters	Jayaprakash.M et al. [10]	Proposed
FPGA	Zynq-7	Zynq-7
Slice LUT	4287	2874
Flip-flop	4252	2195

5. Conclusions

This research paper proposes a smart NoC architecture with collision prevention and fault tolerance techniques. The suggested architecture is implemented on a Zybo Z7-10 FPGA board and coded in System Verilog. Using an efficient X-Y routing algorithm, the architecture prevents congestion and redirects the data packets. The hardware utilization of a single router is compared to that of existing router architectures, the comparison table reveals that the suggested architecture consumes fewer hardware resources thus, lowering network area occupancy effectively.

Acknowledgements

I sincerely thank my research supervisor Dr.A.R. Aswatha for his continuous guidance and encouragement in carrying out this research work.

References

[1] P. P. Pande, C. Grecu, “Performance evaluation and design tradeoffs for network-on-chip interconnect architectures,” Computers, IEEE Trans. on, vol. 54, no. 8, pp. 1025–1040, 2005.

[2] Wayne Wolf, “FPGA Based System Design”, Pearson Education, 2004.

[3] Stuart Sutherland, Simon Davidmann and Peter Flake, “A Guide to Using System Verilog for Hardware Design and Modeling”, Springer, 2nd edition 2006.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

[4] S.Sarkar and Satish S, “Efficient FPGA architecture of optimized HAAR wavelet transform for image and video processing applications”, Journal of Multidimensional Systems and Signal Processing, vol.32, pp.821-844, 2021.

[5] Masoud Oveis-Gharan and Gul N. Khan, “Dynamic virtual channel and index-based arbitration based Network on Chip router architecture”, IEEE International Conference on High Performance Computing & Simulation (HPCS), pp.1-5, 2016.

[6] Ansuman Mishra, “Design of a Round Robin Bus Arbiter using System Verilog”, International Research Journal of Engineering and Technology, pp. 1492-1494, 2020.

[7] Hirebasur Krishnappa Krutthika and Anur Rangappa Aswatha, “FPGA Based Design and Architecture of Network-On-Chip Router for Efficient Data Propagation”, IIOAB Journal, pp. 17-25, 2020.

[8] Vivado User Manual [Online] Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_1/ug910-vivado-getting-started.pdf.

[9] Stuart Sutherland, Simon David mann and Peter Flake, “System Verilog for Design Second Edition”, Springer US, 2nd Edition, 2006.

[10] Jayaprakash M, Manikandan S, Pradeep Kumar S, Sam Jasper P and Prakash C, “Design of Optimized Network on-Chip for Reliable Communication”, Global Research and Development Journal for Engineering, pp. 162-166, 2018.

[11] Raaed Faleh Hassan and Riyam Layth Khaleel, “Hardware Implementation of NoC based MPSoC Prototype using FPGA”, International Journal of Applied Engineering Research, vol. 13, no. 7, pp. 5443-5451, 2018.

[12] Adesh Kumar, Gaurav Verma, Mukul Kumar Gupta, Mohammad Salauddin, B. Khaleelu Rehman and Deepak Kumar, “3D Multilayer Mesh NoC Communication and FPGA Synthesis”, International Journal of Wireless Personal Communications, Springer, vol. 106, pp. 1855-1873, 2018.

[13] Changqing Xu, Yi Liu, Yintang Yang, “SRNoC: An Ultra-Fast Configurable FPGA-Based NoC Simulator Using Switch–Router Architecture”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no.10, 2020.

[14] Subodha Charles, Yangdi Lyu, Prabhat Mishra, “Real-Time Detection and Localization of Distributed DoS Attacks in NoC-Based SoCs”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no.12, 2020.

[15] Kasem Khalil, Omar Eldash, Ashok Kumar, Magdy Bayoumi, “Self-Healing Router Approach for High-Performance Network-on-Chip”, IEEE Open Journal of Circuits and Systems, vol. 2, 2021.

[16] S. Y. Jiang, Y. Liu et al., “Study of fault-tolerant routing algorithm of NoC based on 2D-Mesh topology”, 2013 IEEE International Conference on Applied Superconductivity and Electromagnetic Devices (ASEMD), pp. 189–193, 2013.

[17] V. A. Palaniveloo, J. A. Ambrose and A. Sowmya, “Improving GA-Based NoC

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

Mapping Algorithms Using a Formal Model", 2014 IEEE Computer Society Annual Symposium on VLSI, pp. 344-349, 2014.

[18] Dominique Borrione, Amr Helmy, Laurence Pierre and Julien Schmaltz, "A Generic Model for Formally Verifying NoC Communication Architectures: A Case Study", First International Symposium on Networks-on-Chip (NOCS'07), pp. 127–136, 2007.

[19] Dhiman Ghosh, Prasun Ghosal, Saraju P. Mohanty, "A Highly Parameterizable Simulator for Performance Analysis of NoC Architectures", 2014 International Conference on Information Technology, pp. 311–315, 10.1109/ICIT.2014.66, Dec.2014.

[20] N. Genko, D. Atienza and G. De Micheli, "NoC Emulation on FPGA: HW/SW Synergy for NoC Features Exploration," Proceedings of the International Conference on Parallel Computing (ParCo 2005), pp. 753-760, 2005.

[21] S. Foroutan, Y. Thonnart et al., "Analytical computation of packet latency in a 2D-mesh NoC" 2009 Joint IEEE North-East Workshop on Circuits and Systems and TAISA Conference, pp. 1-4, Jun 2009.

[22] Zhiliang Qian, Da-Cheng Juan, et al., "SVR-NoC: A performance analysis tool for Network-on-Chips using learning-based support vector regression model 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp.354-357, March 2013.

[23] Hemayet Hossain, Mostak Ahmed et al., "Gpnocsim - A General Purpose Simulator for Network-On-Chip", 2007 International Conference on Information and Communication Technology, pp. 254-257, 2007.

[24] Mike Brugge, Mohammed A. S. Khalid, "A Parameterizable NoC Router for FPGAs", Journal of Computers, vol.9, no.3, March 2014.

[25] Priti S, Narayan P "Modified X–Y routing for mesh topology based NoC router on field programmable gate array", IET Circuits Devices & Systems, Devices and Systems, vol.13, no.3, pp.391-398, Jan 2019.

[26] Krutthika H.K and A R Aswatha, "Design of Efficient FSM Based 3D Network on Chip Architecture", International Journal of Engineering Trends and Technology, 68(10), pp. 67-73, 2020.

[27] Dung Hoang Duong1,2(B), Albrecht Petzoldt1, and Tsuyoshi Takagi1,2 "Reducing the Key Size of the SRP Encryption Scheme", Springer International Publishing Switzerland 2016 .J.K. Liu and R. Steinfeld (Eds.): ACISP 2016, Part II, LNCS 9723, pp. 427–434, 2016.DOI: 10.1007/978-3-319-40367-0 27.

[28] Alexander W. Dent. "Choosing key sizes for cryptography", 2010 Elsevier Ltd. All rights reserved. doi:10.1016/j.istr.2010.10.006.

[29] Arjen K. Lenstra, Eric R. Verheul. . "Selecting Cryptographic Key Sizes", J. Cryptology (2001) 14: 255–293, DOI: 10.1007=s00145-001-0009-4.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:11:2021
DOI/10.5281/zenodo.5746712

[30] Stephen M. Matyas. “Key Processing with Control Vectors 1”, Journal of Cryptology, J. Cryptology (1991) 3:113-136.