# Self-Healing Router Architecture for Fault-Tolerant Communication in Network-on-Chip Systems

D. Satheesh[1], M. Suneel[2], Dr I. V. Prakash[3]
[1]Department of Electronics and Communication Engineering
[1]Sree Dattha Institute of Engineering and Science, Sheriguda, Hyderabad, Telangana

**ABSTRACT**
Communication reliability is crucial for Network-on-Chip (NoC) systems that support many cores or Processing Elements (PEs). As connectors, routers are essential, and a malfunctioning router can seriously impair NoC functionality, resulting in misunderstandings and system failure. In order to solve router and port buffer issues on its own, this study presents a novel self-healing technique. To fix malfunctioning routers and related port buffers, the system uses self-healing, which is the capacity to bounce back from hardware issues without outside assistance. The suggested approach uses a three-bit routing mechanism in data packets and makes use of nearby routers for computation. Every router has a self-healing block, which guarantees quick fault recovery. Additionally, by using active buffers to store packets from malfunctioning counterparts, the technique expands its coverage to include defective buffers. By improving NoC systems' resilience and dependability, this method lessens the effect of errors on communication and system performance as a whole.

**Keywords:** Network-on-Chip (NoC), Fault-Tolerant Routing, Self-Healing Routers, Autonomous Error Recovery, Port Buffer Fault Mitigation, Three-Bit Routing Mechanism

## 1. INTRODUCTION

The FPGA industry is rapidly expanding to meet the high-performance demands of real-time applications. However, as CMOS channel length decreases, the size and yield loss increase. Moore's Law suggests doubling the number of devices in a system every eighteen months, necessitating a multifaceted approach to VLSI technology application design. To address higher operating frequencies, increased transistor density, and time-to-market pressure, chip multiprocessor (CMP) and multiprocessor system on chip (MPSoC) architectures are introduced. Traditional on-chip communication bus structures and the integration of complex heterogeneous functional devices on a single chip are insufficient for today's semiconductor industry. Network-on-chip (NoC) emerges as a popular solution for intercommunication challenges among processing elements (PEs) in systems-on-chip (SoC). While the idea of using NoC for FPGA intercommunication is not new, existing methods like crossbars exhibit poor scalability, making the development of new NoC architectures essential for FPGAs. Designing an effective NoC architecture remains a challenge for researchers aiming to achieve high performance without compromising speed and throughput.
A NoC architecture comprises multiple wire segments and routers facilitating data transfer among PEs. In the tile-based architecture, NoC is structured like a city block, with wires and routers forming street grids, and PEs separated by wires. The Network Interface (NI) is a crucial design constraint, transforming data packets into fixed-length flow control digits (flits), typically divided into header, body, and tail flits. In city-block-based tile NoC, routers consist of five bi-directional input and output ports: east, south, west, north, and a local port for the associated PE. Efficient intercommunication between ports involves physical interconnected wires. The router serves as the heart of NoC, controlling and transferring data based on various methodologies. A 5x5 crossbar switch is presented to describe

the router's function, moving data from selected input to output ports using control logic. An arbiter selects the appropriate input port based on priority. A typical NoC router includes five input and output ports, crossbar switch, and arbiter modules for PE intercommunication. Source Intellectual Property (IP) initiates data packet transfer through NI, and the router facilitates transfer to the destination IP's neighbor router. The routing algorithm integrated into each router determines the path from source to destination. A typical NoC uses an X-Y deterministic routing algorithm based on X and Y coordinates. Topology, particularly mesh topology, is a major constraint for routing algorithms, and the source IP measures the distance to the destination in terms of the number of routers, locking the path until the data packet reaches its destination. The destination address and information about intermediate routers are added to the data packet before the transfer begins.

## 2. LITERATURE REVIEW

In quest of increased performance, the semiconductor industry has been rapidly switching from a single microprocessor to multiple core architectures. This is enabled by a rapid reduction in the dimensions of the integrated circuit, which enables the placing of several components on a single chip. NoC is used to provide effective and efficient communication for large systems with multiple cores. NoC is used for routing between multiple cores as NoC is a flexible, scalable, and efficient interconnection technique [1], [2]. NoC can be used in many applications, such as processing components of an aircraft [3], [4] and processors in computers [5] – [8]. Apart from providing efficient communication, NoCs are expected to be power efficient [9], [10] and ideally free of fault and failures. NoC, however, is a complex system that consists of billions of transistors and enables enormous amounts of communication, which makes it vulnerable to faults. Therefore, fault recovery is critical for increasing system reliability in such a system. The network faults can be divided into router fault, link fault, and core fault. A router fault may make the network fail at providing the needed performance. The traditional method for a fault-tolerant router uses spare or redundant routers, which increases the area overhead significantly. Sometimes, spare or redundant channels are allowed [11]. Another traditional method uses a fault-tolerant routing algorithm [12], [13]. The drawbacks of these methods are latency overhead and high area overhead that make the cost too high for mission-critical systems such as aircraft or biomedical systems. A self-healing mechanism is used to recover faults [14], [15]. Self-healing can fix faults through healing or repairing without external intervention. Multiple processing cores can be integrated on a chip, and a faulty core can be repaired by isolating it and using a spare core instead. Such a system can be functional but with limited performance. In an NoC architecture, routers manage communications between cores [16]– [19]. In the case of a router failure, the performance degrades due to the disconnection of cores [20]. Therefore, one of the main challenges in NoC is to heal and recover faulty routers. In this paper, we focus on using self-healing to recover faulty routers at a minimal cost. The role of self-healing becomes even more critical for fault recovery in places where there may be no option for external maintenance, such as in an aircraft during flight. Self-healing performs fault repairs and improves reliability, which refers to the ability of the system to perform its function correctly and within a specific period. NoC consists of multiple routers, and each router is connected to PE, as shown in Fig. 1. The main component of NoC for routing is the router, and the main focus of this paper is on proposing a self-healing router in NoC. The architecture of the baseline router consists of multiple components, as shown in Fig. 2. The router has five input/output ports, Virtual Channel (VC) buffers, Virtual-channel Allocation (VA), Routing Computation (RC), and Switch Allocator (SA), and Crossbar Switch (CS). The operation of each one is described as follows. The VC is the base unit of each port buffer.

It is used to maximize the stored data in each port buffer. The VA component is used to make a decision for which packet request access can be the selected one. The RC block has the responsibility

for routing and directing data packets towards the appropriate output channel and port. The SA component moves between VCs requesting access to the crossbar, and it gives permission to the winning packet. The central crossbar switch is a switch that makes a connection between input and output ports. It selects which input port is forwarded to which output port. Hardware faults in NoC are divided into two classes: transient and permanent faults [21]–[23]. The transient fault is a fault that comes from external disturbance, and it may stay for a short period. The permanent fault is irretrievable physical damage in the system, and it is a continuous fault and stable with time. In this paper, we focus on permanent faults, and the reasons are described as follows. Time-Dependent Dielectric Breakdown (TDDB) is one of the sources, and it indicates insulating film breakdown due to continuous stresses to a gate oxide-film causes [24]. Negatively Biased Temperature Instability (NBTI) is another source that causes threshold voltage degradation due to a stressed transistor with negatively biased gate voltage [25]. Electromigration (EM) is also another source that occurred because of the excessive stress of current density. It causes a sudden delay increase, short, or open fault [26]. Stress Migration (SM) occurs due to excessive structural stress, and it also causes short, open, and delay faults [27]. Furthermore, Hot Carrier Injection is a source of the fault, and it causes an increase in the threshold voltage under the stress of source-drain voltage [28]. The focus of this paper to repair a faulty router.

## 3. PROPOSED METHOD

The proposed self-healing method is applied for a faulty router. A faulty router isolates its local PE from the rest of PEs in the network, and the proposed method provides a router recovery to keep the connection of the isolated PE with the others. The self-healing technique considers the faulty components of the router's components and ports' buffers. The proposed method of architecture is shown on each port in addition to the spare buffer, as shown in Figure 1.

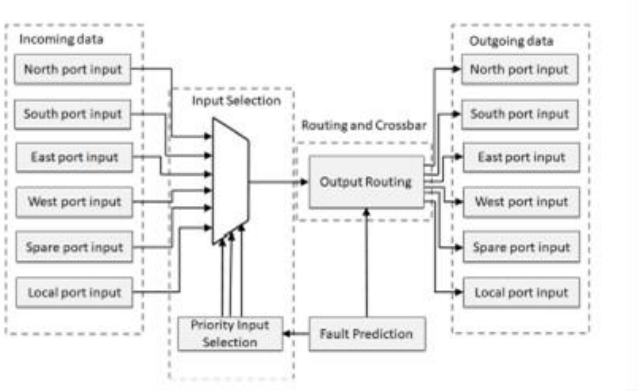


**Figure1: Proposed self-healing method for router**

In the case of a faulty router, the fault detection technique sends a notification to all neighbors of the faulty one. All neighbors consider this notification to update the routing bits. In NoC, each packet is 64 bits and composed of seven fields. The first two fields save the destination X-coordinate and Y-coordinate, respectively. The third and fourth fields have source X-coordinate and Y-coordinate, respectively. The fifth field has the packet sequence number. The sixth field includes the time of transmission of the packet. The last field has the payload data, which models the information inside the packet. The proposed method is based on adding three bits in the packet for routing in the case of a faulty router. These routing bits determine which port in the faulty router receives the packet from the active neighbors.

The value of these bits determines the target port, "000", "0001" "010", "011", "100", "101", are used for Local, East, West, North, South, Spare ports, respectively. When the faulty router receives the packet through ports, a recovery technique is used for routing the packet in the right direction. It has a multiplexer with inputs from the six ports, and input selection signals are used for switching between

these ports. The input selection signals are three bits that decide which port forwards its packet to the output. These values are “000”, “0001” “010”, “011”, “100”, “101” for Local, East, West, North, South, Spare ports, respectively.

The operation is explained as follows. Each clock cycle the input selection value changes between the ports in sequence from “000” to “101”, and then repeats it back. The output packet from the multiplexer is checked by the routing block to decide which output port receives this packet. It checks the last three bits as shown for each port in addition to the spare buffer. The routing bits are sent to the demultiplexer, and it forwards the packet to the appropriate port according to the routing bits. The output port sends the packet to the neighbor router, and it uses XY routing and store-and-forward switching techniques. The algorithm of the proposed method is shown in and Algorithm 1 and Algorithm 2. For more clarification, it is assumed that the router9 is faulty.

The fault detection block sends a notification to the neighbor routers router5, router8, router10, router13. It is also assumed router10 needs to send its packet to router9. Router10 checks the coordinate of the packet with the coordinate of the faulty router. If the coordinate is the same as the faulty router, the router10 updates the routing bits to be “000”. If the X-coordinate of the packet is lower than that of the faulty router, the routing bits are updated to be “010”. This means the packet will forward to West port. If the Xcoordinate of the packet is the same as the faulty router, it means the packet should be sent in this column. Therefore, it checks the Y-coordinate to know which port should be used (North or South). If the Y-coordinate of the packet is higher than the faulty router, the routing bits are updated to “011” for the North port. The routing bits are updated to “100” for the South port if the Y-coordinate of the packet is lower than the faulty router. As demultiplexer is a significant component, a spare one used to avoid a situation of finding the connected demultiplexer is faulty.

For First-In-First-Out (FIFO) buffer consideration, any router has five buffers for five ports: Local, East, West, North, and South. If there is any fault in any port buffer, it leads to losing the received packet to this port. The traditional method is to use a spare buffer for each port to repair the faulty one. Therefore, five buffers are added to the router, and the area overhead of the buffer is 100% relative to the total number of buffers. The proposed self-healing method solves this challenge using a minimum area cost compared to the previous work. The proposed method uses the available buffers in addition to one spare buffer to repair any faulty buffer in the router. The proposed self-healing methods check which buffer has free slots, and then the control block sends the packet of faulty buffer to the available buffer.

The details of the proposed method are described as follows. The method includes a FIFO controller block that has a Fault Signal (FS) input from each port. These are five signals that are coming from (FS_E), West port (FS_W), North port (FS_N), South port (FS_S), and spare (FS_Spare), as shown in Fig. 6. These signals come from the fault detection block. These signals are also used to indicate the availability of the router in terms of buffers storage. If the result indicates the router is full, a signal is sent to the neighbor routers to not send data to this router until receiving a signal initiated it has free slots. If the fault detection block detects a fault in any port, it raises the corresponding signal to the value of one. If there is no fault, the FS value is zero. Therefore, the FIFO controller receives a signal which indicates fault status and its location.

The FIFO controller sends a request to the active ports to ask about the most available one. The FS is input to the FIFO controller, and its value is zero if there is no fault. Therefore, the FIFO controller receives this signal to indicate fault status and its location. The FIFO controller sends a request to the active ports to find the most available one. These ports send back with a grant (gnt) signal for indication of the port availability to store packets. These grant signals are for each port: East (gnt_E), West (gnt_W), North (gnt_N) South (gnt_S), Local (gnt_L), and Spare (gntSpare). Each port’s buffer has five input packets that come from each port in addition to the local buffer, as shown in Fig. 2.

According to the available port, the controller block allows the buffer to pass the packet and save it. Then the buffer updates the number of available slots, and it sends the results back to the controller block. For example, assume the east buffer is faulty. The fault detection block sets the faulty signal to one FS_E ='1'.The FIFO controller receives this signal, and it works to recover it by another buffer. The controller sends a request for the rest buffers (W, N, S, Spare) to check which one is available. If more than one buffers are available, the controller selects the one which has higher available slots than others based on the feedback counter from each buffer. We assume buffer North is available, it means the grant signal is one gnt_N='1'. The FIFO controller sends a signal to the north buffer to receive the packet of the east port (Pkt_E). Then the buffer updates the number of available slots using a Free Slots Counter (FSC). This counter monitors the rest free slots in each buffer: FSC_E, FSC_W, FSC_N, FSC_S, FSC_S, and FSC_Spare for East, West, North, South, and Spare, respectively. The buffer sends the results back to the controller block. The structure of NoC router composed with advanced FIFO buffer and Bi-NoC with self-reconfigurable channels. Initially flow control mechanism discussed and then followed by structure of advanced router.

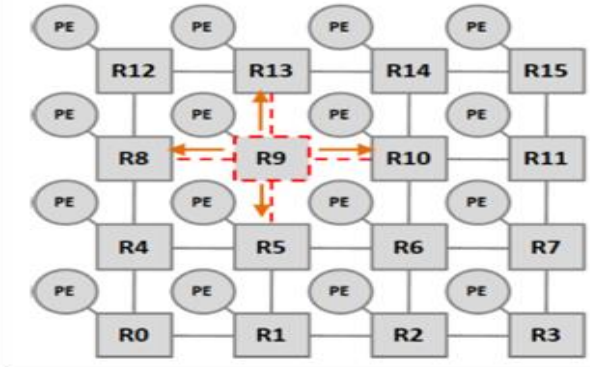


**Figure 2: Block diagram of a faulty router in NoC**

This project composed with buffered strategy for data flow control mechanism because of buffer decouples channel allocation thereby improving the efficiency of flow control whereas in buffered-less strategy, the data packet misrouted or dropped if two channels are allocated a single packet. The buffered flow can be classified into two types that are packet flow and flite flow and again packet and flite flow control divided into two types. The store and forward and virtual cut through are belongs to packet flow whereas Wormhole and virtual channel belongs to flite flow control mechanism. Such as Bi-NoC router consists of three stages of data control that are Routing Computation (RC), Switch Allocator (SA) and Switch Traversal (ST). RC module generates and sends the channel request to SA to transfer data on each buffer. The SA acknowledges same based on availability in FIFO buffer. SA allocates the channel and transfers data to ST stage whenever vacant space available in buffer of neighbour router. In ST stage, the data flits are transferred from input to output port through crossbar switch.

To avoid head of line (HoL) error, virtual channel-based data switching is proposed that composes buffer and protects the data flites. In virtual flow control, virtual channel is assigned to data flite whenever entire data flites are not reached neighbour router. When head flite reaches to queue of buffer through virtual channel therefore it moves to RC stage which decodes and routes request towards associated direction. The FIFO of conventional router located at center and traffic increased huge because of next flite must wiat until current flite transferred to neighbor router. To address this issue, the proposed work distributes the FIFO structure into different stages and also physical channel is divided number of virtual channels. Hence next flite waiting period is reduced and data transfer speed is increased.

The direction of request transfers to the virtual channel allocation (VA) stage to select associated virtual channel of neighbour router. Because number of requests to access same virtual channel, there may be

contention will occur among data flites. The flites which are accessed virtual channel should be present in the current router as previous router is blocked due to contention. Once data flite crosses the VA stage, it assigns to SA stage that presents physical channel into neighbour router. By multiplexing virtual channels to buffer, free data packets never block other data packets which are ready transferred to the destination trough physical channel. Fig.5.1 describes the data transfer with help of virtual channels when physical channel is busy or blocked in Bi-NoC. This structure enables multiple virtual channels with physical channel of associated buffer thereby increasing throughput and avoiding deadlock error. The flow of virtual channel in router from input port to output port is as shown Fig.5.1. The incoming flit which has high priority arrives to the neighbour router accessed appropriate virtual channel initially; thereafter entire data packet will be processed. The incoming first flit of data packet is head flit which arrives to top of virtual channel queue of the buffer thereby entering into RC stage. It decodes in RC stage and creates respective direction of request towards destination router. The direction request of flit transfers to VA stage to obtain selected virtual channels towards destination router

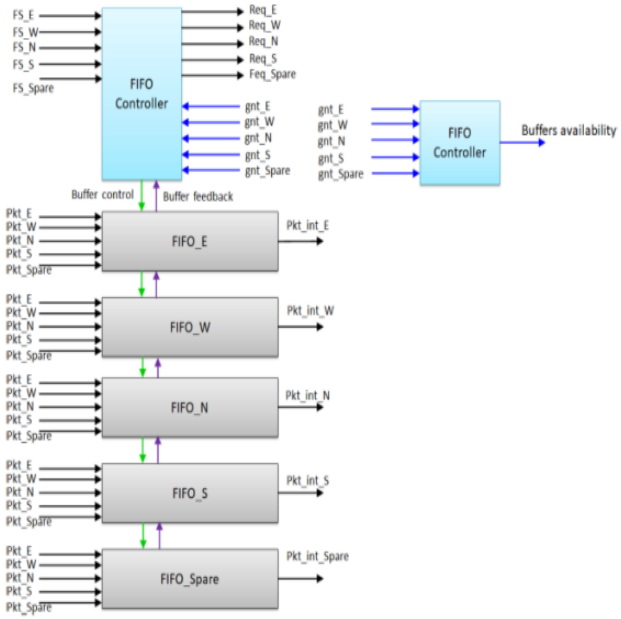


**Figure 3: Proposed self-healing method for faulty buffers**.

## 4. RESULTS & DISCUSSION

Simulation results are pivotal in VLSI design for diverse reasons. Firstly, they serve as a vital tool for verifying that the designed VLSI circuit or system adheres to the specified logic and functionality, ensuring it meets the desired specifications. Secondly, simulation enables an in-depth evaluation of performance metrics such as speed, power consumption, and area utilization, aiding designers in optimizing the design to meet performance requirements. Additionally, simulation identifies critical timing issues, such as setup and hold time violations, clock skew, and signal propagation delays, facilitating refinement of the design's timing aspects. Power efficiency, a paramount concern in modern VLSI design, is assessed through simulation results, allowing designers to analyze power consumption under various operating conditions and pinpoint areas for optimization.

Simulation also assists in assessing noise, crosstalk, and signal integrity issues, enabling designers to enhance the robustness of the design. Moreover, simulation aids in evaluating fault tolerance, reliability, and the circuit's ability to withstand and recover from different types of faults. As a debugging tool, simulation allows designers to observe circuit behavior, identify discrepancies, and

iteratively refine the design. Additionally, simulation results contribute to estimating resource utilization and associated costs, aiding in informed decision-making about trade-offs between performance, area, and power consumption. In essence, simulation is integral to the VLSI design process, providing a comprehensive understanding of the circuit's behavior and performance before physical implementation.

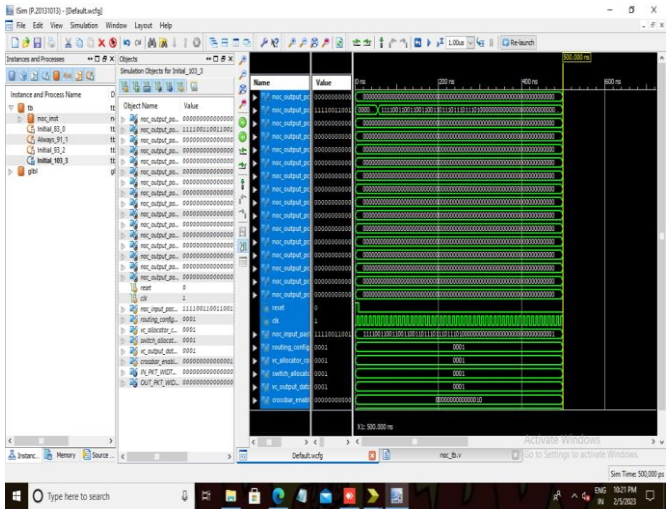

**Figure 4: Simulation Results of the proposed NOC**

Figure 4 shows simulation results of the proposed NOC. The sources make circuits to generate signals that are expected in regular and realistic operation. A fault generator is used to allow each component to receive signals with fault effects. Some components are also can be isolated to present breakdown status. The fault generator also includes a reliability measurement unit to calculate it according to the generated failure rate and the number of routers. The experiments are carried out using a Uniform (UNI) pattern, and in this pattern, all nodes receive the same traffic distribution

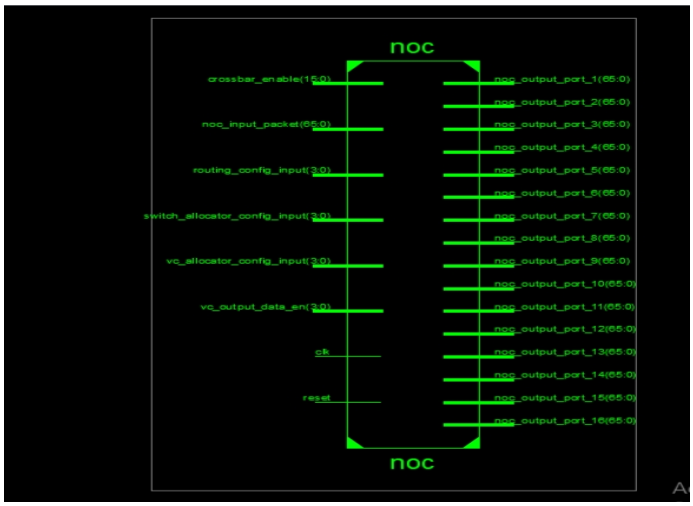


**Figure 5: Block diagram of proposed NOC**

Figure 5 shows the block diagram of NOC which consist of clk, rst, crossbar enable, noc input packet routing_configration, switch allocator and generates output NOC output.

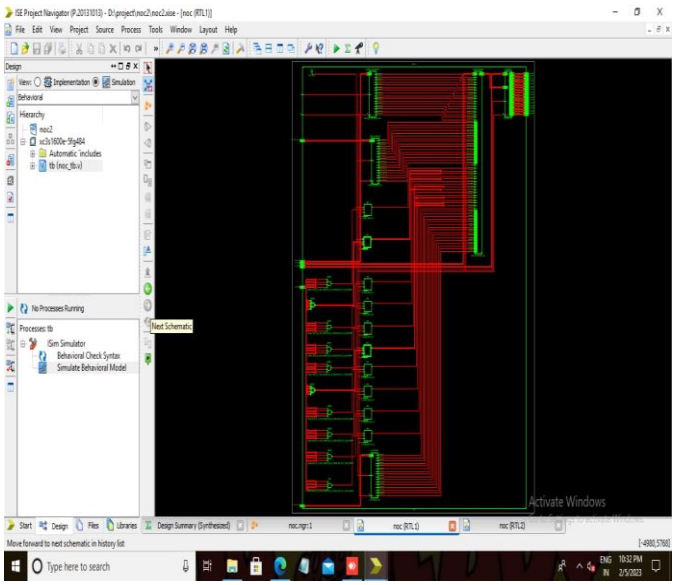

**Figure 6: RTL Schematic of the proposed NOC**

Figure 6 shows the RTL Schematic of proposed system consist of processing unit, Switch allactor, Cross bar circuit.

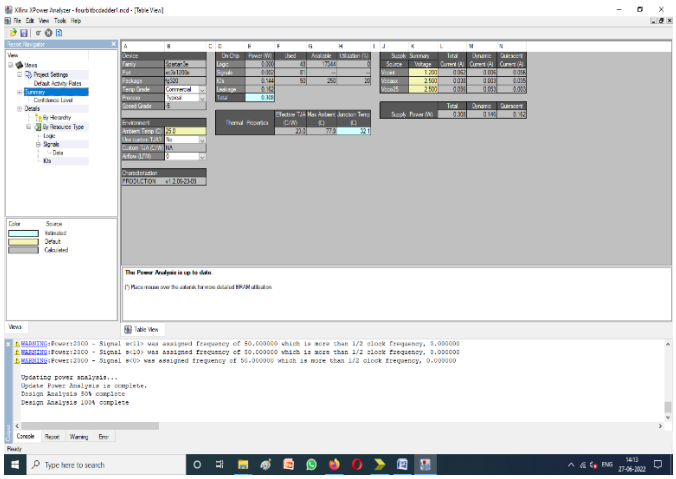

**Figure 7: Estimation of power utilization in the implementation of NOC**

Figure 7 shows the power report of proposed system it generates power is 0.308 mw

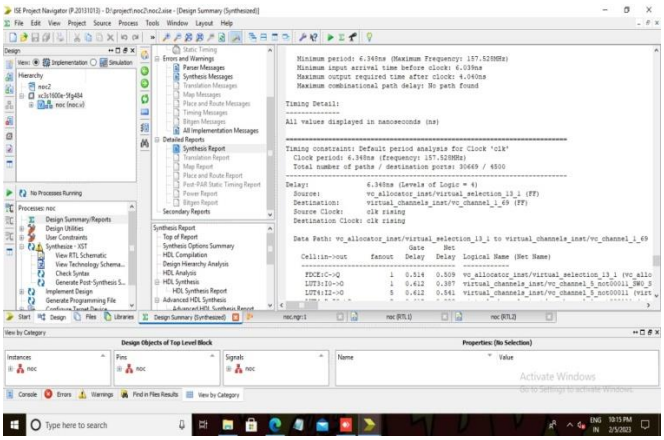

**Figure 8: Estimation of Delay in the implementation of NOC**

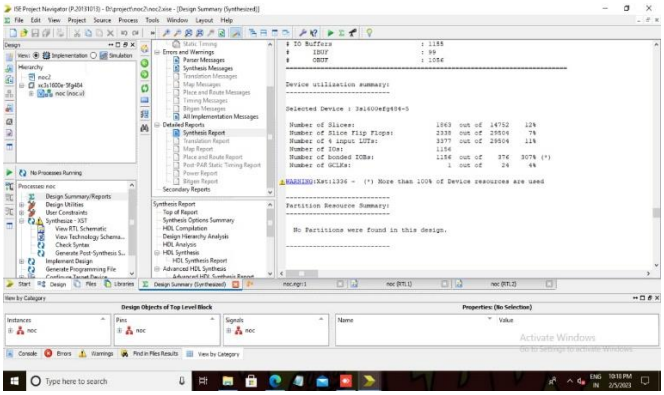

**Figure 9: Estimation of Area in the implementation of NOC**

Figure 8 presents the delay report of proposed system it generates delay is 6.348ns. Figure 8 presents the area report of proposed system consisting of 2338 LUTs

## 5. CONCLUSIONS

This work presents a Reversible Logic Gate Cryptography Design using LFSR key with watermarking. The reversible gates like Feynman, Fredkin, Toffoli and SCL gates are used in this new cryptography system design. Since a cryptography system demands not only high security but low power consumption this work is one of the best among existing systems. This input pixel values are read using Xilinx ISE. The RLGCD architecture consisting of LFSR, encryption block and decryption block is implemented in Xilinx software. This architecture is suitable for both gray scale images and color images. The watermarking using LSB technique is performed to improve the security of the data. The Xilinx performance result for Spartan3E XC3S500E device gives a far better performance as compared to other existing systems.The reversible logic gates are the fundamental requirement in the emerging field of quantum computation. Thus, each work using the reversible logic gates will help to move forward in the field of quantum logics. Since RLGCD is successfully implemented using Verilog code it can be effectively deployed on ASIC in future.

## REFERENCES

[1] N. L. Venkataraman, R. Kumar, and P. M. Shakeel, “Ant lion optimized bufferless routing in the design of low power application specific network on chip,” Circuits Syst. Signal Process., vol. 39, no. 2, pp. 961–976, 2020.

[2] M. S. Sayed, A. Shalaby, M. El-Sayed, and V. Goulart, “Flexible router architecture for network-on-chip,” Comput. Math. Appl., vol. 64, no. 5, pp. 1301–1310, 2012.

[3] S. Majumder, J. F. D. Nielsen, A. La Cour-Harbo, H. Schiøler, and T. Bak, “A real-time on-chip network architecture for mixed criticality aerospace systems,” Aeronaut. J., vol. 123, no. 1269, pp. 1788–1806, 2019.

[4] J. Malburg, K. Janson, J. Raik, and F. Dannemann, “Fault-aware performance assessment approach for embedded networks,” in Proc. IEEE 22nd Int. Symp. Design Diagn. Electron. Circuits Syst. (DDECS), 2019, pp. 1–4.

[5] A. Graillat, C. Maiza, M. Moy, P. Raymond, and B. D. de Dinechin, “Response time analysis of dataflow applications on a many-core processor with shared-memory and network-on-chip,” in Proc. 27th Int. Conf. Real-Time Netw. Syst., 2019, pp. 61–69.

[6] J. Luo, H. Zhou, Y. Zhang, N. Li, and Y. Wang, "An efficient and reliable retransmission mechanism for on-chip network of many-core processor," in Proc. CCF Nat. Conf. Comput. Eng. Technol., 2019, pp. 122–135.
[7] V. Suthar, D. D. Gaitonde, A. Gupta, and J. Singh, "Placement, routing, and deadlock removal for network-on-chip using integer linear programming," U.S. Patent 10 565 346, Feb. 18, 2020.
[8] J. Harttung, E. Franz, S. Moriam, and P. Walther, "Lightweight authenticated encryption for network-on-chip communications," in Proc. Great Lakes Symp. VLSI, 2019, pp. 33–38.
[9] E. Wachter, L. L. Caimi, V. Fochi, D. Munhoz, and F. G. Moraes, "BrNoC: A broadcast NoC for control messages in many-core systems," Microelectron. J., vol. 68, pp. 69–77, Oct. 2017.
[10] K. Khalil, O. Eldash, A. Kumar, and M. Bayoumi, "N2 OC: Neuralnetwork-on-chip architecture," in Proc. 32nd IEEE Int. Syst.-Chip Conf. (SOCC), 2019, pp. 272–277.
[11] B. Bhowmik, J. K. Deka, S. Biswas, and B. B. Bhattacharya, "Performance-aware test scheduling for diagnosing coexistent channel faults in topology-agnostic networks-on-chip," ACM Trans. Design Autom. Electron. Syst., vol. 24, no. 2, pp. 1–29, 2019.
[12] S. Priya, S. Agarwal, and H. K. Kapoor, "Fault tolerance in network on chip using bypass path establishing packets," in Proc. 31st Int. Conf. VLSI Design 17th Int. Conf. Embedded Syst. (VLSID), 2018, pp. 457–458.
[13] J. Khichar, S. Choudhary, and R. Mahar, "Fault tolerant dynamic XYYX routing algorithm for network on-chip architecture," in Proc. Int. Conf. Intell. Comput. Control (I2C2), 2017, pp. 1–6.
[14] K. Khalil, O. Eldash, A. Kumar, and M. Bayoumi, "Intelligent faultprediction assisted self-healing for embryonic hardware," IEEE Trans. Biomed. Circuits Syst., vol. 14, no. 4, pp. 852–866, Aug. 2020.
[15] K. Khalil, O. Eldash, and M. Bayoumi, "A cost-effective self-healing approach for reliable hardware systems," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), 2018, pp. 1–5.
[16] D. Xiang and Q. Pan, "Low-power and high-performance adaptive routing in on-chip networks," CCF Trans. High Perform. Comput., vol. 1, no. 2, pp. 92–110, 2019.
[17] P. Bogdan, T. Dumitra¸s, and R. Marculescu, "Stochastic communication: A new paradigm for fault-tolerant networks-on-chip," VLSI Design, vol. 2007, p. 17, Apr. 2007, doi: 10.1155/2007/95348.
[18] M. S. Sayed, A. Shalaby, M. E.-S. Ragab, and V. Goulart, "Congestion mitigation using flexible router architecture for network-on-chip," in Proc. Japan-Egypt Conf. Electron. Commun. Comput., 2012, pp. 182–187.
[19] F. Angiolini, D. Atienza, S. Murali, L. Benini, and G. De Micheli, "Reliability support for on-chip memories using networks-on-chip," in Proc. Int. Conf. Comput. Design, 2006, pp. 389–396.
[20] K. Khalil, O. Eldash, and M. Bayoumi, "Self-healing router architecture for reliable network-on-chips," in Proc. 24th IEEE Int. Conf. Electron. Circuits Syst. (ICECS), 2017, pp. 330–333.
[21] M. Noda, S. Kajihara, Y. Sato, K. Miyase, X. Wen, and Y. Miura, "On estimation of NBTI-induced delay degradation," in Proc. 15th IEEE Eur. Test Symp. (ETS), 2010, pp. 107–111.
[22] K. Khalil, O. Eldash, A. Kumar, and M. Bayoumi, "Machine learningbased approach for hardware faults prediction," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 67, no. 11, pp. 3880–3892, Nov. 2020.
[23] T. W. Chen, K. Kim, Y. M. Kim, and S. Mitra, "Gate-oxide early life failure prediction," in Proc. 26th IEEE VLSI Test Symp., 2008, pp. 111–118.
[24] T. Nigam, A. Kerber, and P. Peumans, "Accurate model for timedependent dielectric breakdown of high-k metal gate stacks," in Proc. IEEE Int. Rel. Phys. Symp., 2009, pp. 523–530.

[25] M. Ershov et al., "Dynamic recovery of negative bias temperature instability in p-type metal–oxide–semiconductor field-effect transistors," Appl. Phys. Lett., vol. 83, no. 8, pp. 1647–1649, 2003.

[26] G. Gielen et al., "Emerging yield and reliability challenges in nanometer CMOS technologies," in Proc. Conf. Design Autom. Test Europe, 2008, pp. 1322–1327.

[27] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The impact of technology scaling on lifetime reliability," in Proc. DSN, 2004, p. 177.

[28] P.-H. Chen et al., "An 80 mV startup dual-mode boost converter by charge-pumped pulse generator and threshold voltage tuned oscillator with hot carrier injection," IEEE J. Solid-State Circuits, vol. 47, no. 11, pp. 2554–2562, Nov. 2012.

[29] S. R. Vangal et al., "An 80-tile sub-100-W TeraFLOPS processor in 65-nm CMOS," IEEE J. Solid-State Circuits, vol. 43, no. 1, pp. 29–41, Jan. 2008.