Algorithm for Hardware Trojan Avoidance in Network-on-Chip

Naveed Khan Baloch, Ayaz Hussain, Ayesha Haq, and M.Iram Baig

Abstract – **Network on Chip (NoC) is the promising solution to the existing scalability issues in System on Chip (SoC). However, it is exposed to security threats like extraction of secret information from IP cores, availability of network or information on time which is called Hardware Trojan. In this paper, we propose an efficient hardware trojan detection and avoidance technique. Trojans can be inserted at various locations in the network i.e. links and internal modules of the router. These trojans affect the performance of the chip. We have selected trojans that are inserted in the internal modules and results in increased latency and permanent deadlock situations. The proposed Trojan detection and avoidance algorithm named as Bypassing Trojan Affected Router (BTER) is capable of avoiding a trojan effected router in a 2D mesh NoC architecture by modifying the routing algorithm. We use four traffic patterns uniform, shuffle, transpose and tornado for performance evaluation. Results show that proposed technique not only provide better reliability but also decreases latency at least 2 times in case of the uniform traffic pattern, 1.5 times in case of shuffle pattern and transpose pattern 1.2 times in case tornado as compared to the state of the art techniques.**

***Index Terms* – Network on Chip, Hardware Trojan, Trojan Detection, Trojan Avoidance**.

I. INTRODUCTION

System On Chip (SOC) is a solution to previous old technique. It is an integrated circuit (IC) which consists of computer component and electronic system, e.g., digital system, analog system, etc. All its benefits there is a problem of scalability as we know with the passage of time growing technology demands less increase performance with less cost and area. Now Network on Chip (NoC) is the solution to this problem. The fundamental properties of the NoC are as follow:

Communication and computation are separates from each other. As the system size grows, additional links can add in Topology. Customization (link width, buffer sizes, even topology).

Still, we face the problem of security, performance and availability problems in NoC. Those are due to the third party intellectual property. When an IC is manufactured, its all components are not designed and fabricated from scratch by one party. IC manufacturing process involves many parties for different components, which cause threats. The third party may introduce malicious module which may affect actual functionality of that module. It may be readily detected or may be hard to detect even from reverse engineering. This malicious inclusion is called Hardware Trojan.

Some conventional techniques to detect Hardware Trojan are Functional testing, built-in self-test (BIST), design for testability (DFT), Side-channel analysis, etc. In function testing, the input ports of a chip stimulate and observe the output continuously to identify manufacturing faults. If the result (logic values) of the output miss-match the genuine pattern, then a fault or a Trojan could be found.
BIST and DTF both techniques add circuit (which is design according to test logic) to the chip to verify either IC perform its function correctly or not. Usually, checksums and scanning technique are used to design internal test logic for the circuit. Both methods are original design for error detection, but we can use these to detect Hardwar Trojan.
Different signals like magnetic and electric fields are emitted by the electrically active device. By analyzing the electrical activity, information about the state and the data processed by the device can get. The sensitive advance method is developed to measure these side-effects. Measurement of these analog signals helps to avoid measurement errors or other inaccuracies also help to detect Trojan.

The remaining paper is divided into following sections: in Section II, Related work is summarized. In Section III, an overview of Baseline Router Architecture is explained. Section IV consists of the overview of the method which is proposed now. Section V presents the latency and reliability evaluation of NoC according to proposed technique. Conclusions and future work are discussed in Section VI.

II. RELATED WORK

Network-on-chip (NoC) topologies [1] [2], routing algorithm, switching mechanism and challenges in NoC and their solutions are described. Reliability issues of NoC are also discussed. A significant goal of NoC is to get more excellent performance and design productivity to control and fulfill the requirements of current age [1].

There is a vast range of research, e.g., topology, switching, communication layers, routing, etc. in the field of NoC. There are a variety of open issues in NoC domain on which researchers are working and can work in future for further enhancements. The various proposed topological structures in both 2D NoC and 3DNoC research domain are discussed in [3]. Trojan classification, models of Trojan operations and an audit of the best in class Trojan counteractive action and identification methods are discussed [4]. Additionally, talk about the significant difficulties connected with future research hand the security concern needs to address them.

Complete taxonomy of hardware Trojans [5] [6] considering five natural qualities. This taxonomy is used to study the HT and detection of missing classes of HT. This

Naveed khan Baloch, Ayaz Hussain, Ayesha Haq and M.Iram Baig are with the Department of Computer Engineering, University of Engineering and Technology Taxila, Pakistan. Email: naveed.khan@uettaxila.edu.pk, academia4ayaz@gmail.com, ayshahaq@yahoo.com, iram.baig@uettaxila.edu.pk. Manuscript received on Feb 07, 2017 revised on July 20, 2017 and accepted on Nov 30, 2017.

paper creator sorted out the Embedded Systems Challenge (ESC) to incorporate Trojans and analyzed them to accept the classification. Results show that 50 percent Trojans are activated by the user and 75 percent of them are placed in the I/O units [5].

Due to increase in computing demand, SoC cannot fulfill the requirements of the current age. For this technology shrink and NoC develop. In Noc there are many security and reliability challenges. Many techniques are proposed to resolve these issues. Logic testing is useful in process noise and is useful for detecting small Trojan. It is not worthwhile to generate test vectors and to detect large Trojans [7].

Another comparison results in [6] tell that side-Channel Analysis is efficient for massive Trojan and easy to produce test vectors. It does not work for process noise and to detect ultra-small Trojan. Taxonomy of hardware Trojans and a survey of existing techniques to detect the Trojan are presented in [8]. Using a shadow register, explain the path delay measurement architecture which can be used for IC verification and Trojan exposure.

In paper [8] an Authenticated Encryption (AE)-based security framework for NoC based systems is presented. Secure communication among IP cores is possible due to the presence of the security framework in Network Interface The paper [13] talks about how a strategy for entirely measuring the combinational deferral of a self-assertively expansive quantity of register-to-register ways inner to the useful part of the IC can be utilized to give the necessary validation and configuration modification (counting HTH im-plantation) discovery. This minimal effort delay estimation method does not influence the fundamental IC usefulness and can be performed at-pace at both test-time and run-time.

A smart enemy can embed a modest difficult to-distinguish Trojan into a substantial configuration, which can without much of a stretch avoid ordinary post-silicon test and approval prompting disastrous result [14]. A Trojan can attack various phases of IC life cycle. It can have many targets of operational failure in field or release mystery data from inside a chip e.g. it can release secret information or the key in a cryptographic IC. In this paper, Trojan issue is categories and propose an assurance approach for Trojan assaults, joining pre-deployment outline/acceptance procedures with post-sending internet checking.

Hardware Trojan attack as a vindictive change of a configuration has risen as a noteworthy security danger. Side channel examination has been explored as another option to regular logic testing to recognize the nearness of Hardware by traffic limiting counter. Simulation and implementation are done on NoCsim emulator using Verilog/VHDL modules. The outcomes demonstrated average area overhead and did not influence the system execution separated from some underlying inertness.

The design flow of an IC design is the stage where Trojan horses can be injected easily. They turn into practically undetectable by remaining inactive them self for a long time. Therefore, dynamic verification paradigm based on countermeasures was proposed. Performing computation is difficult; the critical insight that checks the results of a computation makes it easy. Resourceful attackers can be slow by this technique but not completely stop these attackers. Reverse engineering is useful but the excessive cost and time-consuming process to detect the Trojan in an IC. In paper [10] reverse engineering steps and types are explained in detail.

The paper [11] dissects time to (i) create a move in useful Trojans and (ii) entirely initiate them. A productive sham flip-flop inclusion procedure is introduced to build Trojan action. Depending upon verification time and circuit topology, a move likelihood edge is chosen so that embedded dummy flip-flop would reasonably affect zone overhead.

(NI) of every IP core. Communication between secure and non-secure cores can be done using the temporary key and permanent keys are used by the secure cores for communication. Unauthorized memory access avoids by using right access table [9]. Bandwidth denial is controlled The reproduction results on s38417 benchmark circuit exhibit that, with an unnecessary zone cost, the proposed method can extensively increase Trojan action and minimize Trojan activation time. A new method based on behavior is proposed to classify Trojans into two categories: explicit payload Trojan and implicit payload Trojan. This will give an inexpensive but effective and efficient hardware Trojan detection. This classification technique makes it easy to construct Trojan models It also make it achievable to lower the cost of testing. Path delays of supposed chips are collected to create a series of fingerprints, each one shows one feature of the characteristics of an authentic design. By comparing the delay parameter of the chip with its fingerprints, chip validation is confirmed. From the delay point of view, path delays comparison of path delays makes small Trojan circuits important. The experiment's result of this method shows that the detection rate on explicit payload Trojans is 100 percent. Further development of this method will help to detect implicit payload Trojans [12].

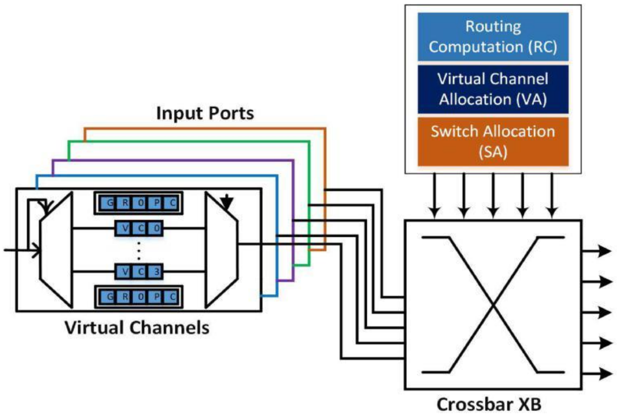

Fig. 1 Baseline Router Architecture

Trojans. However, these strategies experience the ill effects of diminished affectability toward little Trojans, mainly as a result of the broad procedure varieties present in current nanometer advancements. In this work, a noninvasive; different parameter side-channel examination based Trojan recognition methodology is introduced. It utilizes the inborn relationship between element present and greatest working recurrence of a circuit to include the impact of a Trojan circuit from procedure clamor [15].

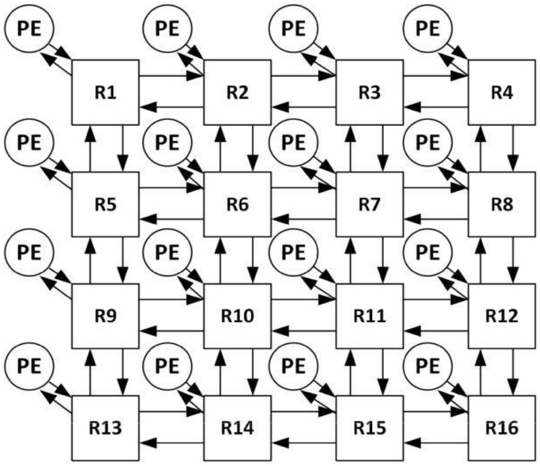

Fig. 2 4x4 Network connected in Mesh Topology

The same issue as in the past is a location by displaying another procedure TeSR, a Temporal Self-Referencing approach that looks at the present mark of a chip at two unique time windows to take out the impact of procedure clamor, consequently giving great discovery affectability to Trojans of fluctuating size. Moreover, not at all like existing methodologies, it does not require brilliant chip examples as a kind of perspective. Reproduction results for three elaborate plans and three delegate consecutive Trojan circuits exhibit the viability of the methodology under expansive between and intra-pass on procedure varieties [16]. Other paper [17] talks about the identification of HTs by their symmetry breaking inside coordinated circuits (ICs), s measured by the way postpone. Ordinarily, way postpone, or side channel techniques depend on correlations with a brilliant or trusted, example. In any case, brilliant guidelines are influenced by entombing and intra-bite the dust varieties which restrict the trust in such correlations. Symmetry is an approach to identify changes to an IC with expanded certainty by affirming sub-circuit textures inside as it was initially outlined. The distinction in deferrals from an offered way to an arrangement of symmetric ways is the same unless an embedded HT breaks the symmetry. The symmetry can generally exist in ICs or can be misleadingly included. A technique is portrayed to discover, and measure way postpones against symmetric ways, and additionally the focal points and burdens of this strategy.

In Run Time Latency Auditor for NoC (RLAN), a monitoring packet Proximal Analogous Packet (PAP) is injected into the network. Original packet and PAP has the same priority and hope count. Attacker inserts Trojan into the packet that is traveling towards victim node. The victim node A injects packets face latency. On the other hand, consider that the packet X is sent from node B to node A. Here a PAP is created and injects it to one of the proximal nodes P around A. PAP identifies at P when it reaches and re-injected towards A. At A, the arrival time of PAP is stored and then it compares with an arrival time of X. In this way delay in packets is detected [18].

In work mentioned above in [18], RLAN can detect only the path where Trojan exists. It cannot detect the exact location and cannot avoid the Trojan. To remove this deficiency, we extend this work which is explained in later section.

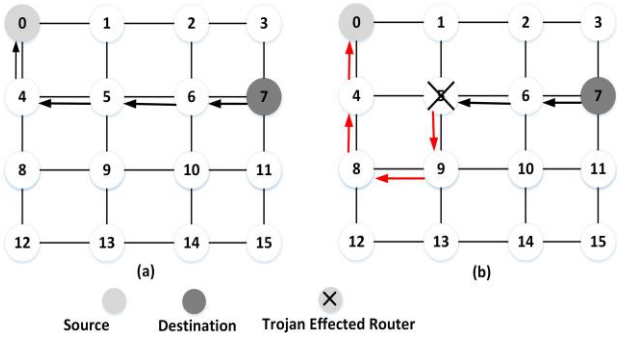

Fig. 3 Problem in NoC Case 1

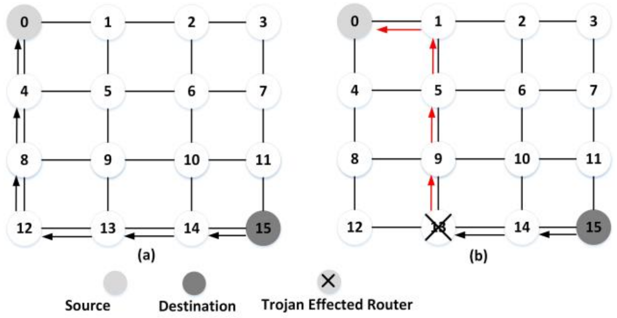

Fig. 4 Problem in NoC Case 2

III. BASELINE ROUTER ARCHITECTURE

Fig. 1 represent the overview of the baseline router architecture. The baseline router in Network on chip (NoC) consists of Routing computation (RC), Virtual allocator (VA), Switch allocator (SA) and Crossbar (XB) unit. The routing computation unit plays a crucial role in the reliable communication of the network on chip. It is the responsibility of the RC unit to extract the destination information from the packet. The architecture of the RC unit depends upon the routing protocol employed in the network. The RC unit determines the output port through which packet moves towards next hop. The virtual allocator (VA) and switch allocator (SA) for providing access to the crossbar. It is their responsibility to manage fair arbitration in the network. The VA is used to manage fair arbitration among all input port. It gives turns to each of the input port virtual channels uniformly. The SA is used for managing conflicts among the virtual channels to access the crossbar.

It is the responsibility of SA to manage the fair arbitration among the virtual channels to transmit a flit through the crossbar. The Crossbar (XB) is used to create a connection between input and output port of the router. Each of the virtual channels transmits a flit to the neighboring router with the help of XB. Fig. 2 shows the interconnection of 4x4 Network in a 2D mesh topology. Each router relates to the Processing element (PE) with the help of network interface (NI).

If a hardware Trojan exits in the router, which is activated upon a specific event. This hardware Trojan may change the destination information of the packet to violate the roles implements by the routing protocol. If the network is following minimal routing protocol then maybe because of hardware Trojan packet may be rerouted to the non-minimal path, which may result in discarding the packet and incurs an extra latency which is not tolerated able in NoC architecture.

IV. PROPOSED TECHNIQUE BTER

A. *Realizing BTER*

To realize a BTER, we modified the working of the normal router micro-architecture. The control and data path of the trojan are fed to various router pipeline stages to allow convert manipulation of the router functionality.

B. *Trojan Activation*

The trojan activation depends upon the specific condition. Trojan activation can be classified into two broad categories. Activation can be achieved through a various method such as software hardware coalition, time-based, event-based, monitoring traffic patterns in the network [19] [20].

C. *Overview of the Proposed Idea*

The decreasing size of transistor facilitates the designer to fabricate hundreds of IP cores on a single chip. These IP blocks can be high-bandwidth I/O, processors, memory elements or DSP cores. The technology scaling has made the Network on chip (NoC) vulnerable to faults. Another performance degradation parameter is Hardware Trojan. Hardware Trojan is malicious medication in the IC. It is intentionally inserted at any location of IC and any stage of IC fabrication. It aims to degrade the performance of IC or leakage of secret information.

In network on chip, Data transfer takes place in the form of packets. Each packet contains multiple flits. These flits broadly classified into three categories. Head flit contains destination information; the payload contains actual data and tail flit which is used to deallocate the resources allocated by Header part. As we say Trojan can be inserted at any place of the IC, so Trojan can alter the data present in the packet field. It depends on the aim of Trojan that which field or data it has to be altered.

The proposed work assumes that Trojan aim is to re-route the packet by changing its intermediate destination. For example, in 4*4 Mesh when a packet wants to move from source router 7 to destination router 0, as shown in Fig. 3(a), the minimal path adopted is 7, 6, 5, 4, 0. A Trojan effect this minimal path routing and miss route the packet from router 5 to router 9 which is not in the minimal path and take more time to reach the packet at the destination, as shown in Fig. 3(b). The complete traversal path of the router is defined using dimension order routing. The packet contains the intermediate address of all the router through which packet passed during its traversal towards the destination node. When the packet reaches the intermediate router, if the ID of the router is not in defined it means the previous node misroutes the packet. In this way, we came to know about the faulty router. A broadcast message is sent to all the router which contains faulty router ID. Next time at the step of defining a path in the packet, the router bypass faulty router using deflection routing.

Now consider the case when source and destination are at the corner of the network. In a 4*4 Mesh when a packet wants to move from source router 15 to destination router 0, as shown in Fig. 4(a), the path adopted is 15, 14, 13, 12, 8, 4, 0. A Trojan effect this path and miss route the packet from router 13 to router 9 which is not in its path as shown in Fig. 4(b).

D. *Path Calculation, Updating, and Comparison.*

When a packet comes at source router, here path calculation is done by the dimension order routing algorithm, which is a minimal path. It is called route computation. The whole path is written in the packet along with the source and destination information. After that, the packet proceeds towards its destination. The packet follows the calculated path to reach its destination. During the journey of packet towards its destination, the ID of each router from which the packet is passed, update the packet field name as update path. In this way, we detect the Trojan.

Now we have two paths one is calculated at the source which is an original path. The packet follows the original path if no Trojan exists in the router. The packet contains the second path which is computed during the packet traversal towards its destination named as an updated path. When the packet reaches the destination router, both paths are compared. If the updated path is equal to the calculated path, is mean Trojan is not active. In case Trojan is active during the router path traversal, then it misroutes the packet from its designated path.

E. *Trojan Detection*

The Trojan is detected with the help of comparing both paths of the packets to reach the destination. One path is computed at the transmission of the packet which is stored in the calculated path field. The second path is updated during the traversal of the packet through the network. As the packet passed through the intermediate nodes, the ID of intermediate nodes stored in the updated path by Routers field. When the packet reached the destination, both paths are compared to determines the router ID through which packet deviated from its original path.

For example, in 4x4 mesh, we have router 2 as an intermediate node for a packet coming from router 0 currently on router 1. Router 2 sends this packet to router 3 according to its original path computed at the source, but in case of Trojan, router2 send the packet towards the router 6 instead of sending this packet towards the router 3 according to its original path definition. We see change occur after router 2. It indicates that router is affected by Trojan or say Trojan is at router 2.

F. *Trojan Avoidance*

When the information about affected router reaches to all routers, every router then stores the ID of the router and next time when route computation is performed router check that is that router is in the calculated path or not. If yes, then it changes the routing form XY to YX and calculates the path accordingly. Fig. 7 show the working of the proposed algorithm.

Source	Destination	Packet ID	Data

Fig. 5 Basic Packet Format in NoC

Source	Destination	Calculated Path	Updated path by Routers	Data

Fig. 6 Packet Format Used in Proposed Algorithm

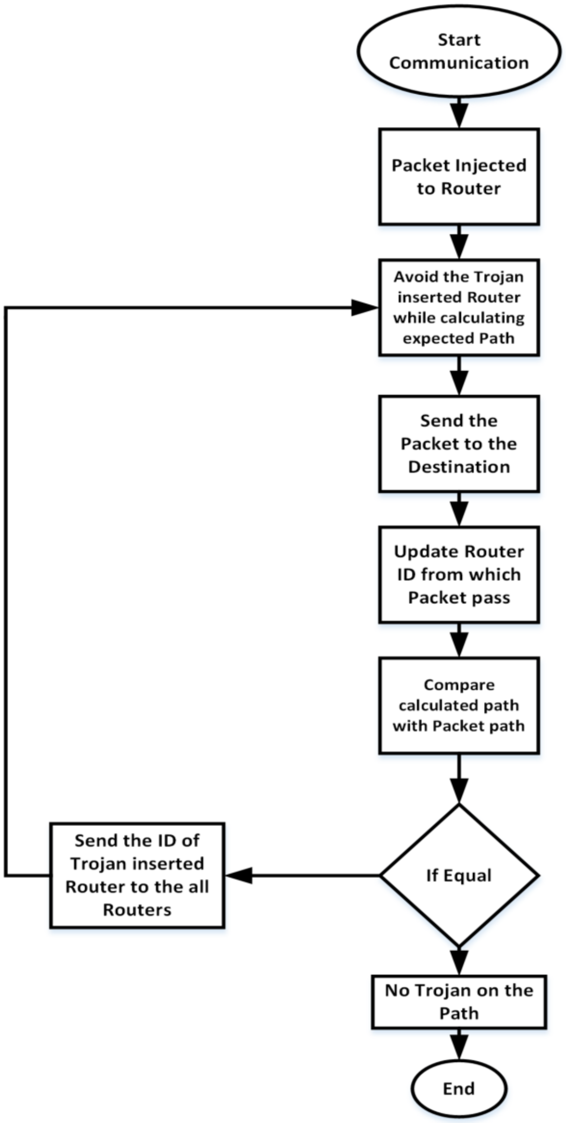
Start Communication
Packet Injected to Router
Avoid the Trojan inserted Router while calculating expected Path
Send the Packet to the Destination
Update Router ID from which Packet pass
Compare calculated path with Packet path
If Equal
Send the ID of Trojan inserted Router to the all Routers
No Trojan on the Path
End

Fig. 7 Proposed BTER Algorithm

G. *Proposed BTER Packet Format*

The packet format is necessary to get proper bandwidth in the network for communication, Network on a chip has packet format as shown in Fig. 5. It has four fields source, destination, packet ID, and data. Source field contains information about source router from where the packet is sent. Destination filed has information about the destination of the packet where the packet must reach. Packet ID represents the identity of the packet.

The proposed technique used the following packet format as shown in Fig. 6. The packet has 5 fields source, destination, calculated (updated path by routers from which packet pass) and a data field.

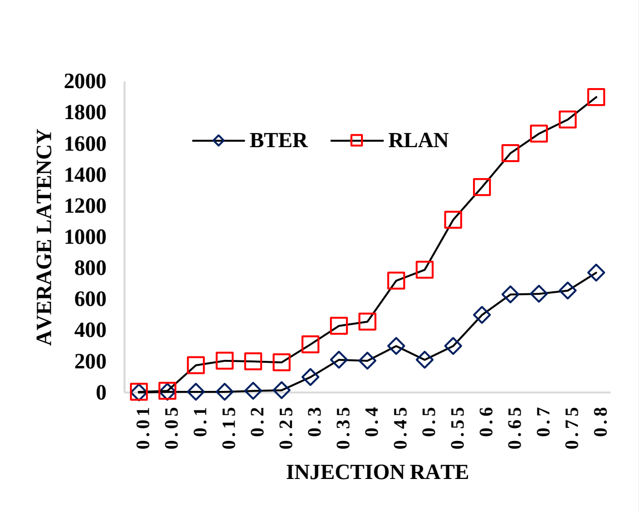

Fig. 8 x-axis Latency, y-axis Injection rate, Comparison between BTER and RLAN, Traffic Pattern Uniform

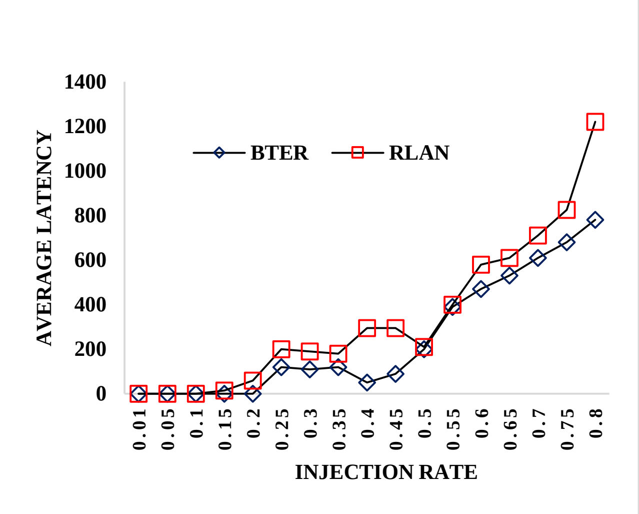

Fig. 9 x-axis Latency, y-axis Injection rate, Comparison between BTER and RLAN, Traffic Pattern Shuffle

V. EXPERIMENTAL RESULTS

For evaluation of BTER algorithm, we utilized a cycle accurate simulator Bookism. A 4x4 mesh network is established. The configuration of the network is placed with each router having VC set to 8, buffers size remains 8 and wait for tail credit value is kept yes. In this section, we analyzed our proposed network on different synthetic traffic patterns. We also analyze the two performance parameters latency and reliability during simulation. The Proposed techniques are evaluated on different inject rates. It is observed that as the as the injection rate increases, the latency also increase. This algorithm gives the best result in case of the uniform traffic pattern and is more reliable in case if four routers are affected by Trojan.

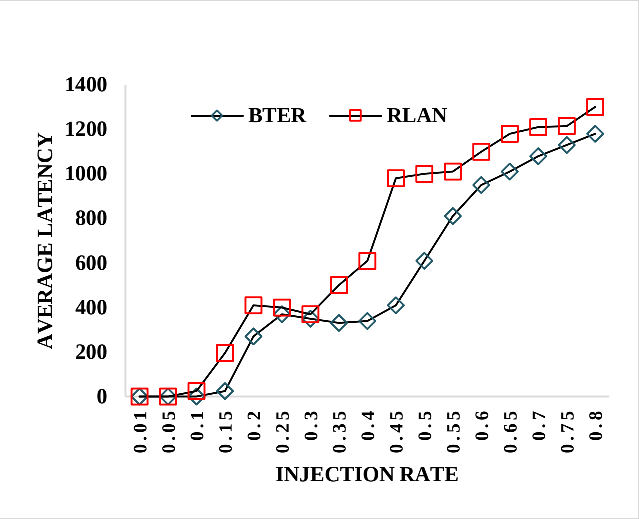

Fig. 10 x-axis Latency, y-axis Injection rate, Comparison between BTER and RLAN, Traffic Pattern Transpose

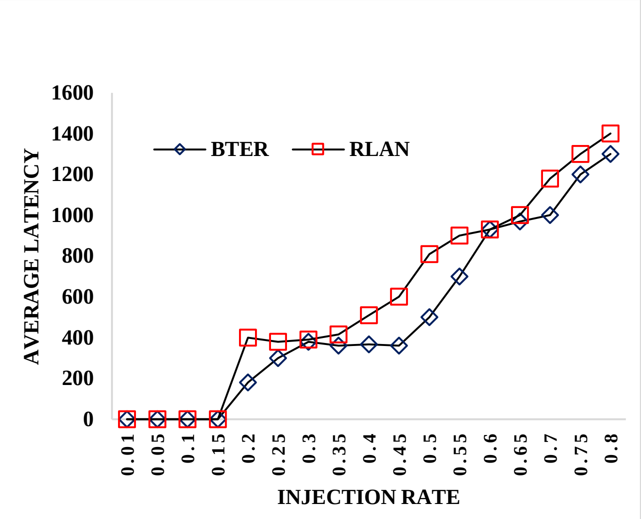

Fig. 11 x-axis Latency, y-axis Injection rate, Comparison between BTER and RLAN, Traffic Pattern Tornado

A. *Latency*

The time need for a packet to cross the network is called latency of the network. This time is measured when the head of the packet arrives at the input port to the time when the packet tail leaves the output port. We separate latency T into two components shown in (1).

$$T = T_h + \frac{L}{b} \qquad (1)$$

The head latency, Th, is the time needs by the head of the message to pass through the network, and the serialization latency is the time required for the tail to catch up that is the time for a packet of length L to cross a channel with bandwidth b, shown in (2). Link latency and throughput depends not only on topology but also depends on flow control, routing protocol and design of the router.

$$T_s = \frac{L}{b} \qquad (2)$$

B. *Impact of Uniform Traffic Pattern on Latency*

Fig.8 shows the latency of BTER and RLAN on different injection rate. We observe that with an increase in injection rate results in increase latency of both the algorithms. Comparisons of both techniques show that this increase is less in case of proposed algorithm BTER. The analysis shows that proposed algorithm has an average latency of 350.164284and previous algorithm has an average of 606.3121. It means proposed algorithm have latency 2-time lees then RLAN.

C. *Impact of Shuffle Traffic Pattern on Latency*

Fig. 9 shows the latency of BTER and RLAN on different injection rate and traffic pattern of shuffle. We observe that with an increase in injection rate latency of both the algorithm also increase. Comparison of both techniques shows that this increase is less in case of proposed algorithm BTER. The analysis shows that that proposed algorithm has an average latency of 241.489141 and the previous algorithm has an average of 340.251118. It means proposed algorithm have latency 1.5-time lees then RLAN.

D. *Impact of Transpose Traffic Pattern on Latency*

Fig. 10 shows the latency of BTER and RLAN concerning injection rate and traffic pattern Transpose. We observe that proposed algorithm has an average latency of 487.989425 and the previous algorithm have an average of 643.589269. It means proposed algorithm have latency 1.5-time lees then RLAN.

E. *Impact of Tornado Traffic Pattern on Latency*

Fig. 11 shows the latency of BTER and RLAN on different injection rate and traffic pattern tornado is selected for analysis purpose. We observe that with an increase in injection rate latency of both the algorithm increases. It is observed that this increase is less in case of proposed algorithm BTER. We observe that proposed algorithm has an average latency of 394.8279 and the previous algorithm have an average of 506.400353. It means proposed algorithm have latency 1.2-time lees then RLAN.

F. *Reliability*

Reliability is dependent on the perspective of the user. Reliability can be measured by counting how many packets reach the destination. We can say less the error more the system is reliable.

Fig. 12 shows the reliability calculated in case of traffic pattern uniform. Observation shows that BTER improved reliability maximum 14 percent and minimum 3 percent in case of one and 4 routers affected respectively.

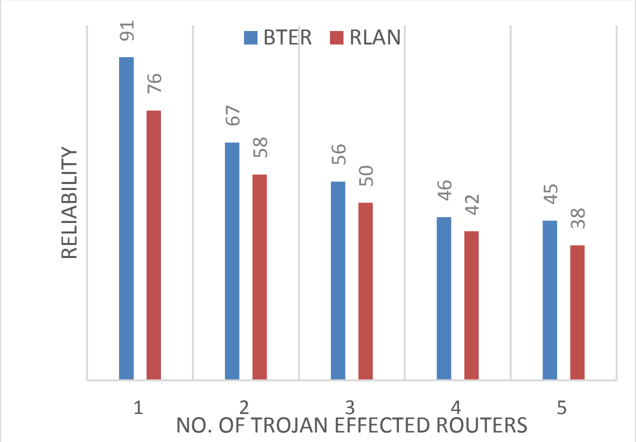

Fig. 12 x-axis Number of Trojan Effected Routers, y-axis Reliability, Comparison between BTER and RLAN [20]

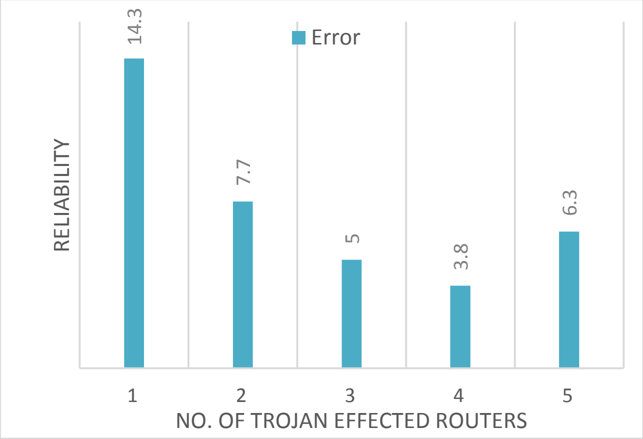

Fig. 13 Difference of Reliability of BTER and RLAN

Fig. 13 shows the reliability error of BTER and RLAN. We can see the error of 13 percent when one router is affected, 9 percent error when two routers are affected, 7 percent error when three routers are affected, 3 percent error when four routers are affected, and 7 percent error in case of four routers are affected. So according to the definition of less the error more the system is reliable. So, if there are a maximum of four routers are affected we have less error of 3 percent.

The analysis shows that proposed technique BTER gives the best result in case of the uniform traffic pattern.

As injection rate grows latency of both techniques also grows, but it is observed that BTER gives 2 times less latency then RLAN [18]. The Table I shows the comparison of Proposed Technique with [18].

TABLE I. COMPARISON OF BTER WITH RLAN

Sr#	Injection Rate	Latency of BTER	Latency of RLAN
1	0.01 to 0.25	16.946 to 22.2754	18.9672 to 165.8954
2	0.3 to 0.55	110.345 to 320.3188	222.9934 to 824.6653
3	0.6 to 0.8	532.9579 to 737.553	824.6653 to 1134.4873

VI. CONCLUSION

This research contributes to render high performance, reliable and fault-tolerant approach for 2D mesh network-on-chip. By utilizing the proposed BTER technique, routers affected by Trojan can be avoided successfully with optimal performance and reliability of the system. Previous algorithm RLAN [18] just detect the path where Trojan exists, but BTER algorithm detects the Trojan affected path as well as Trojan affected router. We analyzed proposed algorithm is more reliable and suffer less from latency than existing algorithm RLAN. The reliability and performance of the proposed technique could be increased more. One suggestion is to check this algorithm for the case when Trojan affected Router is present in both XY and YX routing algorithms. Moreover, as our algorithm works on 2D NoC, one can extend this algorithm for 3D NoC for more improvement in Trojan avoidance field.

REFERENCES

[1] N. A. P. N. S. Ravanaraja, “Survey exploration of network-on-chip architecture.”

[2] S. S. Bhople and M. Gaikwad, “A comparative study of different topologies for network-on-chip architecture,” International Journal of Computer Applications, pp. 1–3, 2013.

[3] N. Choudhary, “Network-on-chip: a new soc communication infrastructure paradigm,” International Journal of Soft Computing and Engineering, vol. 1, no. 6, pp. 332–335, 2012.

[4] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, “Hardware Trojan: Threats and emerging solutions,” in High-Level Design Validation and Test Workshop, 2009. HLDVT 2009. IEEE International. IEEE, 2009, pp. 166–171.

[5] J. Rajendran, E. Gavas, J. Jimenez, V. Padman, and R. Karri, “Towards a comprehensive and systematic classification of hardware trojans,” in Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on. IEEE, 2010, pp. 1871–1874.

[6] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, “Trustworthy hardware: Identifying and classifying hardware trojans,” Computer, vol. 43, no. 10, pp. 39–46, 2010.

[7] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, “Hardware trojan attacks: threat analysis and countermeasures,” Proceedings of the IEEE, vol. 102, no. 8, pp. 1229–1247, 2014.

[8] M. Tehranipoor and F. Koushanfar, “A survey of hardware trojan taxonomy and detection,” IEEE design & test of computers, vol. 27, no. 1, 2010.

[9] A. Agarwal, C. Iskander, and R. Shankar, “Survey of network on chip (noc) architectures & contributions,” Journal of engineering, Computing and Architecture, vol. 3, no. 1, pp. 21–27, 2009.

[10] R. Torrance and D. James, “Reverse engineering in the semiconductor industry,” in Custom Integrated Circuits Conference, 2007. CICC’07. IEEE. IEEE, 2007, pp. 429–436.

[11] H. Salmani, M. Tehranipoor, and J. Plusquellic, “New design strategy for improving hardware trojan detection and reducing trojan activation time,” in Hardware-Oriented Security and Trust, 2009. HOST’09. IEEE International Workshop on. IEEE, 2009, pp. 66–73.

[12] Y. Jin and Y. Makris, “Hardware trojan detection using path delay fingerprint,” in Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on. IEEE, 2008, pp. 51–57.

[13] J. Li and J. Lach, “At-speed delay characterization for ic authentication and trojan horse detection,” in Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on. IEEE, 2008, pp. 8–14.

[14] S. Narasimhan, D. Du, R. S. Chakraborty, S. Paul, F. G. Wolff, C. A. Papachristou, K. Roy, and S. Bhunia, “Hardware trojan detection by multiple-parameter side-channel analysis,” IEEE Transactions on computers, vol. 62, no. 11, pp. 2183–2195, 2013.

[15] F. Koushanfar and A. Mirhoseini, “A unified framework for multimodal submodular integrated circuits trojan detection,” IEEE Transactions on Information Forensics and Security, vol. 6, no. 1, pp. 162–174, 2011.

[16] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, “Tesr: A robust temporal self-referencing approach for hardware trojan detection,” in Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on. IEEE, 2011, pp. 71–74.

[17] N. Yoshimizu, “Hardware trojan detection by symmetry breaking in path delays,” in Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on. IEEE, 2014, pp. 107–111.

[18] R. JS, D. M. Ancajas, K. Chakraborty, and S. Roy, “Runtime detection of a bandwidth denial attack from a rogue network-on-chip,” in Proceed-ings of the 9th International Symposium on Networks-on-Chip. ACM, 2015, p. 8.

[19] D. M. Ancajas, K. Chakraborty, and S. Roy, “Fort-nocs: Mitigating the threat of a compromised noc,” in Proceedings of the 51st Annual Design Automation Conference. ACM, 2014, pp. 1–6.

[20] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, “Designing and implementing malicious hardware.” LEET, vol. 8, pp. 1–8, 2008.