Run Time Mitigation of Performance Degradation Hardware Trojan Attacks in Network on Chip

Manoj Kumar JYV, Ayas Kanta Swain, Sudeendra Kumar, Sauvagya Ranjan Sahoo, KamalaKanta Mahapatra
Department of Electronics and Communication Engineering
National Institute of Technology, Rourkela, Odisha, India
{manojkumarjyv, swain.ayas, kumar.sudeendra, sauvagya.nitrkl, kmaha2}@gmail.com

***Abstract*—Globalization of semiconductor design and manufacturing has led to several hardware security issues. The problem of Hardware Trojans (HT) is one such security issue discussed widely in industry and academia. Adversary design engineer can insert the HT to leak confidential data, cause a denial of service attack or any other intention specific to the design. HT in cryptographic modules and processors are widely discussed. HT in Multi-Processor System on Chips (MPSoC) are also catastrophic, as most of the military applications use MPSoCs. Network on Chips (NoC) are standard communication infrastructure in modern day MPSoC. In this paper, we present a novel hardware Trojan which is capable of inducing performance degradation and denial of service attacks in a NoC. The presence of the Hardware Trojan in a NoC can compromise the crucial details of packets communicated through NoC. The proposed Trojan is triggered by a particular complex bit pattern from input messages and tries to mislead the packets away from the destined addresses. A mitigation method based on bit shuffling mechanism inside the router with a key directly extracted from input message is proposed to limit the adverse effects of the Trojan. The performance of a 4x4 NoC is evaluated under uniform traffic with the proposed Trojan and mitigation method. Simulation results show that the proposed mitigation scheme is useful in limiting the malicious effect of hardware Trojan.**

Keywords- Hardware Trojan, Router Architecture, Performance Evaluation.

I. INTRODUCTION

Multiprocessor system on chips (MPSoC) is ubiquitous to meet the computing demands of current day semiconductor applications. Network on chips (NoC) are pervasive in most of the MPSoCs, required to manage the computing and data transfers between the processing elements in the MPSoC efficiently [1]. MPSoCs are developed and integrated with the help of several third-party vendors. Further, globalization of semiconductor design and development has led to several security issues. Hardware Trojans (HT) are one of the serious threats, which is widely discussed and researched in both academia and industry [2]. HTs are widely discussed in connection with cryptographic cores to leak the secret data and keys. The denial of service

(DoS) type attacks using HTs are targeted on processing elements. In DoS attacks, the intention of the adversary is to create revenue and reputation loss to the organization. HT attacks on NoC/MPSoC are largely DoS attacks. The modern day MPSoC architectures are heterogeneous designs and include hundreds of different types of cores. The adversary may try to include the malicious code in IP core or in the communication network. Insertion of HT in the network inside the chip will cause disruptions in the functioning of the system which may lead to DoS type attacks and high amounts of latency. On-chip network is a very attractive design module for an adversary to perform DoS type attack by inserting the HT. With the advent of HT in NoC, security of on-chip networks has taken a new dimension. The conventional security and reliability aspects of NoC discuss unintentional bit corruption, packet loss, misrouting, and packet duplication etc. Malicious hardware/HT purposefully injects the DoS attack, leak information, bypass encryption/authentication and corrupt data. The general classification of HTs in NoC is based on functional correctness, path diversity, isolation, on-chip fault tolerance and quality of service. The existing HT models are based on generating surge in traffic injection and flood network resources to perform DoS attacks.

The NoC security is an active research area from last one decade. A secure router design for NoC is described in [3][4]. This paper proposes an HT and mitigation method with the following features:

- Generally, HT trigger in NoC is based on timing and conditions. In the proposed HT, logic-based triggering is used. When a specific stream of bits from input data is used for transmission, the HT is triggered. This technique requires the Trojan to bypass the functional and code coverage during verification process like any other malicious modifications do.
- Bit shuffling based method is used to mitigate runtime hardware Trojan.
- A complete Hardware synthesizable Trojan design's equivalent is used in the NoC simulation.

Section II describes the related work. Section III presents the model of hardware Trojan. Section IV gives an illustration of proposed HT mitigation method. The effects of proposed HT on NoC and its performance evaluation is listed in section V. Section VI presents the conclusion of the paper.

II. RELATED WORK

Hardware Trojans (HT): Hardware Trojans are malicious inclusions by a dishonest engineer. The intentions of an adversary will vary from design to design based on its functionality. The widely discussed and generally known intentions of the adversary are to promote side channel

2159-3477/18/$31.00 ©2018 IEEE
DOI 10.1109/ISVLSI.2018.00139

attacks to leak confidential information and denial of service (DoS) attacks. Intentions of an adversary to insert HT in NoC are similar to HT described in literature like leaking secret data and DoS attacks. HT in NoC can be categorized based on the location of the HT, effect of HT and trigger mechanism. Types of HT in NoC are shown in Table I.

Table.1 Classification of HTs

Type of HT	**Description**
Location of HT	HT in NoC can be inserted in communication links or in router architecture.
Effect of HT	Leaking secret information, Denial of Service, packet loss and performance degradation.
Trigger of HT	Software Trigger: - MPSoC are designed using NoC. Adversary can use software to trigger HT. Hardware Trigger: - When a specific and rare scenario like specific input or a specific condition is met, HT will get triggered.

Authors of [3] propose an aggressive HT which is placed in router architecture to leak the confidential data. The HT [3] will forward the packets with sensitive data to the rogue thread operating on a different core. This HT needs both hardware and software support. HT trigger is in hardware, and data collection is supported by software thread running on a different core. The authors of [4] propose HT which alters the bits during router to router data transmission causing DoS type attack. In [5], DoS attack HT suppresses the allocation requests and de-prioritizes the arbiters. This HT is inserted into arbiter architecture. TASP (Targeted Activated Sequential Payload) HT proposed in [6] injects the faults in the links between the routers when a specific target is identified in the data stream. An HT attack on routing tables is discussed in [3]. Boraten et al proposed a fault injection side channel attack using a covert HT and its mitigation techniques for such attacks. Further injected faults will create deadlocks and failure in the functionality of chip (DoS type attack) [6]. An illegal packet request attack (IPRA) HTs proposed in [7] are triggered conditionally inside the routers. One or a number of HTs are placed at buffer sites and are triggered in idle mode. Frey et al have proposed Physical Unclonable Function (PUF) based mitigation scheme to counter hardware Trojan [8]. In this paper, we primarily focus on attacks which cause PD inducing DoS attacks. It is easy to design PD and DoS HTs than data leaking HT.

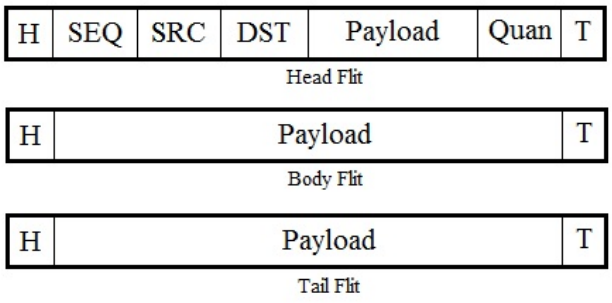
III. HARDWARE TROJAN MODEL

Network on |Chip comprises of a router, a network interface, and several IP cores. Router is the basic building block that transfers messages over networks via packets. A router consists of an arbiter, a routing computational module, a crossbar, and five Input and output ports. Each input port consists of a buffer that is used to store information. The buffers are usually implemented with FIFO [9] [11].

Flit format and Trojan's effects

Data is transferred between the nodes of NoC in terms of packets. The packet is further divided into flits. The parts of packet are structured as head flit, tail flit and required number of body flits based on the packet size. Generally, a valid packet contains one head flit and one tail flit and number of body flits. The head flit is comprised of destination address, quantity of flits in the packet, type of packet (valid head bit), packet identity number and also source address. The tail flit contains a valid tail bit and body flits are a major source of data. The format of flit is shown in Fig.1.

Directly targeting the data and corrupting it is very difficult because the probability of getting caught in the verification/code coverage is very high so in the proposed work we deal with Trojans that targets only critical fields of the flits and its main target is performance reduction.



H : Head bit (1 for Head Flit) , T : Tail bit (1 for Tail Flit)
SEQ : Packet's sequence number, Quan : Flit quantity of packets
SRC : Source tile's **x**-y positions, DST : Destination tile's **x**-y positions

Fig.1 Flit Format

Attack scenario

The sequence of attack activities are as follows

- The flit, after entering the router through one of its input ports gets stored in the Buffer queue.
- When the designed trigger condition is met on the input flit, the payload mechanism changes the particular field and then stored in the buffer.
- The modified flit then passes to the crossbar switch and with the help of route computation module, it gets forwarded to another node. It starts affecting different components of router based on the field effected.

In our work, we consider following HTs: - Flit Quantity Trojan (QT), Address Trojan (AT), and Head Hardware Trojan (HHT) and Tail Hardware Trojan (THT). They target the critical fields of flit as explained below.

Quan Trojan

It targets the Flit quantity indication field. When the packet finally arrives, the destination node finds more or less flits with respect to flit quantity in head flit. The result is loss of flits by abandoning of packets, and wastage of resources due to waiting caused at destination node thus results in more latency and hence a performance decline. It's equivalent to creation of dummy flits or duplication of flits.

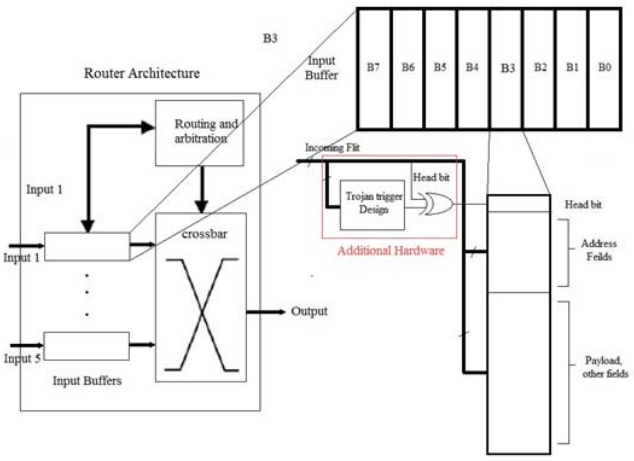


Fig. 2 Trojan design and attack scenario

Address Trojan

The address Trojan targets the destination address field (and source address if source routed algorithm is used for routing). After Trojan's attack on address field, when the packet reaches the crossbar switch, based on routing algorithm's output, the packet will be misrouted from destined address to another node. A malicious receiver-inserted node can receive the packets leading to data leakage. This will also block the destination node to receive packets sent to it. This is a major type of denial of service attack for the receiving node considered in this work.

Head Hardware Trojan

This Trojan targets the head bit. If the head bit of head flit is attacked, route computation module will not access the address field. Hence the whole packet will be dropped and retransmission is requested. This continuously performed action leads to resource starvation and NoC's performance is severely affected and degraded.

Tail Hardware Trojan

The tail Trojan changes the tail indication bit and after a successful passage through crossbar and output buffer, the receiving node keeps on waiting for a tail flit and this causes packet mixing and packet loss.

The Head Hardware Trojan operation is given in Fig. 2. In general, output of Trojans trigger module is a logic 0 and so after entering the router's input port, the flit fields are stored in input buffer without any modification. But when the Trojan activation condition is present in the input data, the Trojan's trigger is activated and give a logic 1 and the Head bit gets reversed and then pushed into the buffer. A similar mechanism is designed for other types of Trojans as well.

All the above-mentioned effects may vary a little depending on how router is architected. The performance degradation and denial of service effects are measured by increased latency, reduced throughput, decline in packet count, and misrouted packets (determined by packet distribution).

IV. PROPOSED HT MITIGATION USING BIT SHUFFLING

The proposed method shuffles the critical bit fields of the flits among themselves and others, obscuring the data fields and making them less sensitive to the Trojan. So even if the Trojan targets the packets, the attacks are applied on randomly shifted data and hence mostly inappropriate. As an additional feature of protection, a 1-bit Hamming code, an error correcting code (ECC) to foil the single bit targeted attacks is also adopted since type indication bit of a flit typically is of one-bit length only. The proposed work focusses on the Trojans that present in routers alone and we assume that the network interfaces and links are not a good place to insert Trojans and a bit difficult in implementation. In the router configuration, a Trojan can be present at any place like within the buffer, after buffer, or even just before or after the crossbar switch. The modified router design with the additional modules is as shown in the Fig. 3

Bit shuffling method

When the flit enters the Router, first it passes through the Encoder. The shuffle encoder of this method is shown in Fig. 4. The shuffler is typically a set of selector modules i.e. multiplexers that select particular bits into targeted output lines. This shuffle encoder shuffles up the informative bit fields given in Fig.1 other than the data among themselves. So, all the bit fields are displaced to other places.

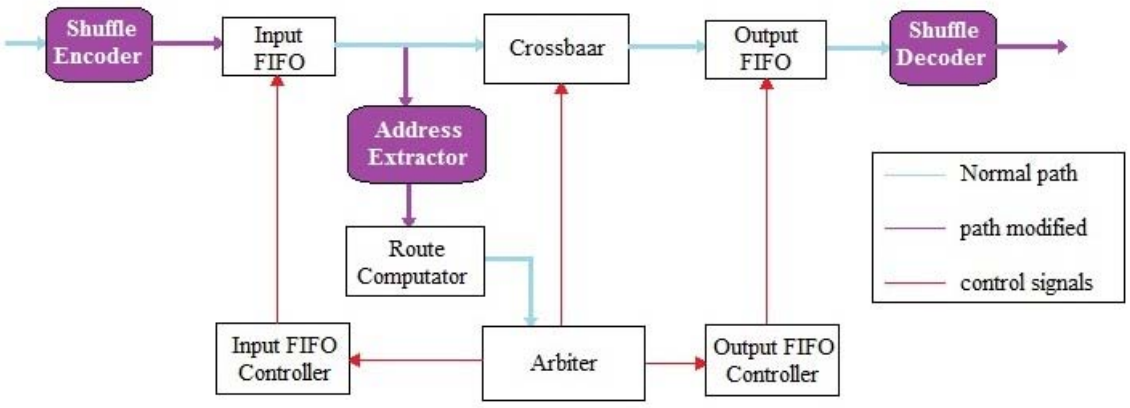


Fig. 3 Bit Shuffling Technique

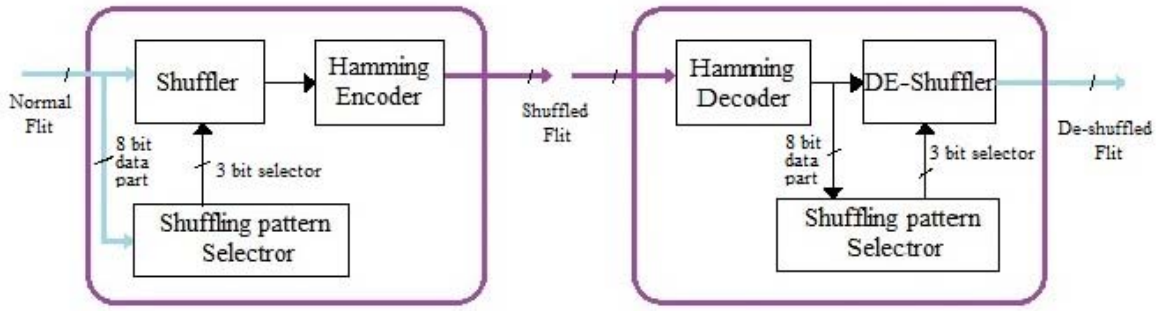


Fig. 4 Shuffle Encoder and Decoder

In order to have runtime mitigation of the Trojan, the selection of pre-planned shuffling patterns is done in cycle wise timely manner with selector generating different selections to the shuffling multiplexer's input as shown in Fig.4. In this method, shuffling pattern selector takes inputs from the least significant data lines of payload and derives a small set of selection lines. After the critical fields are shuffled, the flit passes through the input FIFO and then to crossbar switch.

At the same time the address extractor partially, de-shuffles the flit fields and extracts destination address for route computation and the information is passed to arbiter. Now the arbiter arbitrates the packet that came into the crossbar and the packet is sent to selected output FIFO. From the output FIFO, the packet (i.e., Head flit followed by body and tail flits) enters the shuffle decoder which is exactly reciprocal in design to the shuffle encoder. Here the 1-bit errors are corrected and the original pattern is restored by de-permutation of the bit fields. After this module, the packet will be outputted and sent to another router or to local node. The shuffler used is of size 3 bits to select one of 8 shuffling patterns to shuffle 14 critical field bits in total.

To balance the burden of fuzziness of the shuffled bit fields with additional hardware, we suggest to use x-bit shuffler (x-bit shuffler selects one out of 2 power x shuffling patterns) for 5x to 6x number of targeted bit fields.

The bit shuffler is designed in such a way that the highest significant fields like address or packet length are more likely to be shifted to and arranged randomly in less significant fields like source address (once the communication is established with a first most packet among two nodes in the network, the next packet's source address hardly matters in most of the existing models), packet's global number (if available) etc. Thus, we have a number of sets of shuffled bit patterns and one need to be selected.

To carry any meaningful attack even to deplete the sources, the Trojan has to target particular filed in a particular way. So, the Trojan will not be able to access the required ones as it will be unaware of the shuffling patterns and thus fails to carry any meaningful attack. For example, if the Trojan wants to leak the information to the local node, it changes the destination field to local address. But as the bits are permutated at the router's input itself, other bits are randomly placed in the destination field, the destination address is not modified to local address and hence packets are not sent to local node and thus the Trojans attack for DoS are foiled.

Hamming code [19, 14] Design

The single bit error correcting code (ECC) just after the shuffling and so erroneous bits up to one bit are recovered just before De-shuffling.

The Block code we used in this work for single bit error correction is [19, 14] Hamming code. The bit fields considered for the 4 x 4 network are as follows:

- 1 - Head bit
- 4 - Source address bits
- 4 - Destination address bits
- 4 – Flit quantity bits
- 1 – Tail bit

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

We have inserted Trojans after the buffer module. Even though the Trojans can be triggered in lot of methods, for simulation purpose, we considered triggering with a specific combination of input data lines. Practically the Trojan can be present anywhere throughout the design of router so the defense mechanism will cover all the modules in the router as the design modifications are placed at the ends of the existing design. We also assume that presence of the Trojan in router design means it is present in all the routers i.e. all the nodes because the same router design will be used at all the nodes. The experimental results are subjected to the designed Trojan's configuration and may vary depending on the activation mechanisms.

Simulation setup

The simulation is done with a 4 x 4 Mesh topology with payload length of 32 bits, flit quantity of 5 per packet, and buffer register is a FIFO having a depth of 8-bits. We used the configuration of Wormhole pipeline with identical inter-arrival time distribution to have uniformity throughout all network. We considered different flit injection rates (FIR) ranging from 0.1 to 0.7 in steps of 0.1 flits per cycle per node. We have run the simulation for 0.1 million cycles considering the first 20k cycle period as warm up time to let

the network settle down with steady state traffic. NoCTweak simulator is used to evaluate the performance of NoC under uniform traffic [10].

Throughput

The rate of information delivered through the network is known as throughput i.e. the number of packets/flits that is generated/accepted/passed by a node and so it gives the network's efficiency.

Fig. 5 shows that when the Trojan is activated throughput is reduced by 63% in case of HHT and 71% in case of THT. Our method recovers completely as the length of the type fields are very small. As shown in the Fig. 6 the Trojan effect is more when it attacks flit quantity than destination address field. The recovery is 67% and 45% in QT and AT respectively.

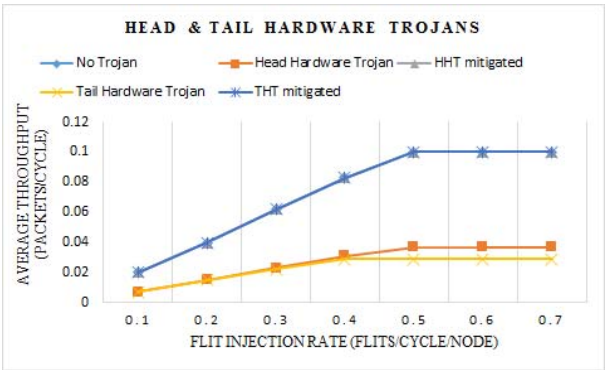


Fig. 5 Average throughput for Head and Tail Trojans

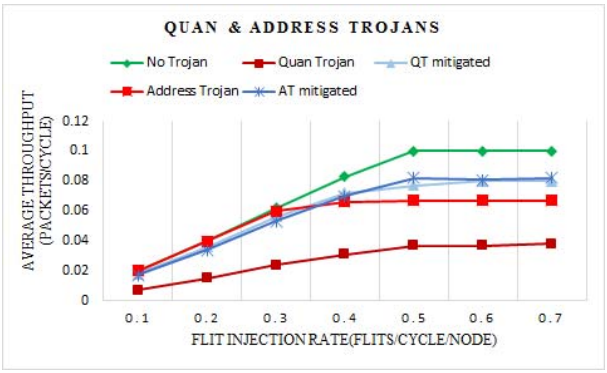


Fig. 6 Average throughput for Quan and Address Trojans

Average Latency

The average latency indicates how much time delay a packet is taking to reach the destination from its origin on an average. Fig. 7 indicates that there is no much change in average latency in HT scenario but latency is drastically increased in Tail Trojan case. This is because of successful delivery of head and tail flits which is not in Head Trojan's case. When the destination address is targeted latency is increased to 1.5 times and this method makes that only 10% extra to the normal case. Even though there is no much change in the case of QT, this method is causing a little more latency in order to mitigate the Trojans effects as presented in Fig. 8.

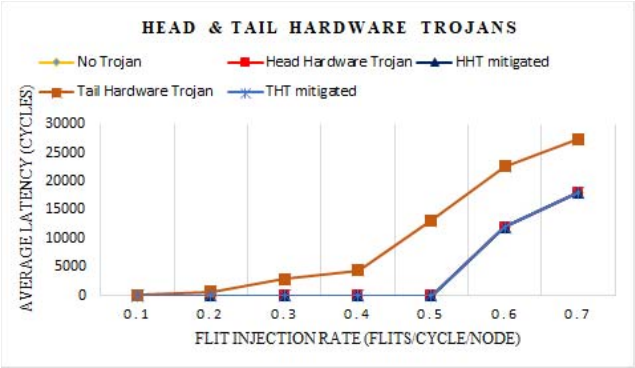


Fig. 7 Average Latency for Head and Tail Trojans

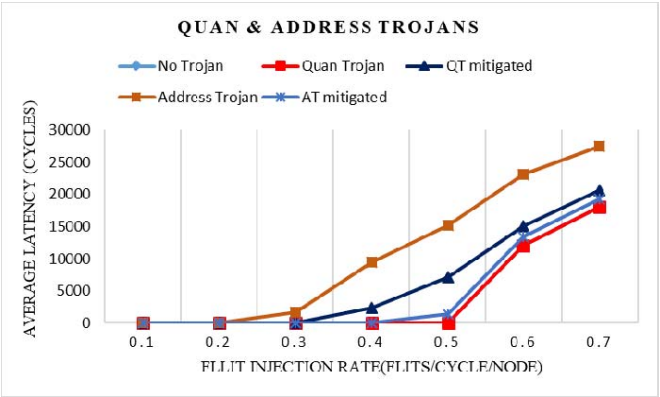


Fig. 8 Average Latency for Quan and Address Trojans

Total number of packets received

When the Trojan modifies the destination address 40% of packets are in effect and only a 40% of the packets are received in all cases of the attacks. This method is providing more packets up to 80% in all cases. These results are shown in Fig. 9 and Fig. 10.

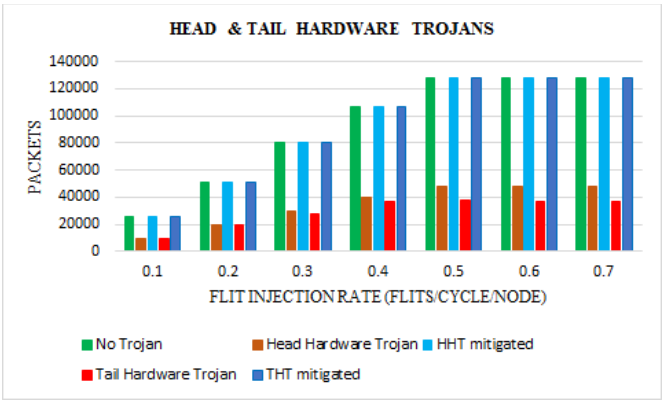


Fig. 9 Total number of received packets for Head and Tail Trojans

Node wise packet reception

The effects caused by tail Trojan and Head Trojan are illustrated in the Fig.11 and Fig. 12. The number of packets received by each node is drastically reduced in these two cases and the proposed method is bringing back the lost packets.

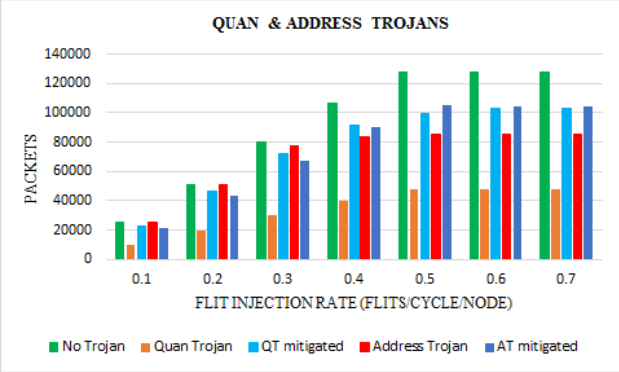


Fig. 10 Total received packets for Quan and Address Trojans

Fig. 13 demonstrates the Address Trojan which causes the diversion of the packets from the targeted node to another node. To create a DoS attack scenario, we designed the address Trojan to block the right-side routers and diverted the packets to other routers on the left side. One can see the huge no. of diverted packets in the Fig. 13. The method is able to mitigate the diversion and is successful with more than 90% of packets reaching their destinations. Fig. 14 shows how the bit shuffling method is able to recover a lot of lost packets at different nodes.

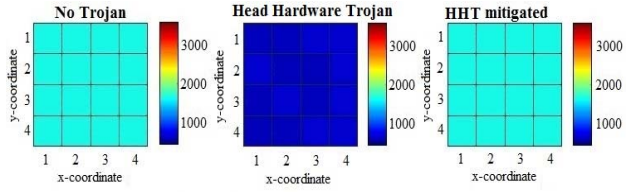


Fig. 11 Received packets at each node for Head Hardware Trojan

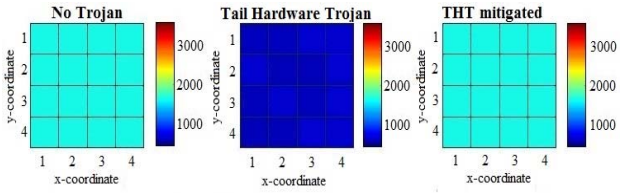


Fig. 12 Received packets at each node for Tail Hardware Trojan

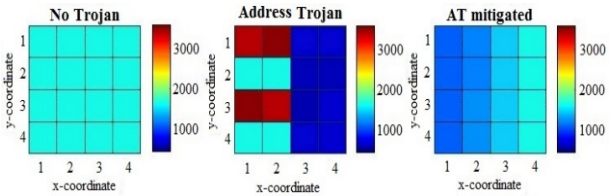


Fig. 13 Received packets at each node for Quan and Address Trojans

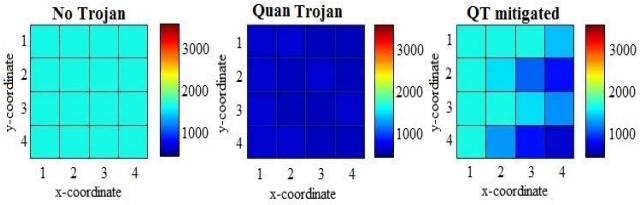


Fig. 14 Received packets at each node for Quan and Address Trojans

VI. Conclusion

In this paper, we proposed four different varieties of hardware Trojans, which degrade the performance and cause DoS of the network on chip. The proposed HTs have a complex triggering mechanism in comparison with earlier techniques. We also propose bit shuffling based HT defense scheme which is an efficient technique against proposed four types of hardware Trojans. The proposed HTs and mitigation scheme are verified on 4x4 NoC. The simulation results show that the proposed method is efficient in thwarting the Trojans attempts. The additional blocks that are used for mitigation are taking 21.2 percent of area overhead when compared with NoC router and this is negligible when compared with the local core/processor which contains huge functionality.

References

[1] W. Dally and B. Towles, “Route packets, not wires: onchip interconnection networks,” Proc. Design Automation Conference, Jun.2001, pp. 684–689.

[2] S. Adee, “The hunt for the kill switch,” IEEE Spectrum, Vol. 45, no. 5, May 2008, pp. 34–39.

[3] D. M. Ancajas et al., “Fort-nocs: Mitigating the threat of a compromised noc,” in DAC, Jun 2014, pp. 1–6.

[4] A. K. Biswas et al., “Router attack toward noc-enabled mpsoc and monitoring countermeasures against such threat,” CSSP, vol. 34, no. 10, Oct 2015, pp. 3241–3290.

[5] R. JS, D. M. Ancajas, K. Chakraborty, and S. Roy, “Run time detection of a bandwidth denial attack from a rogue networkon-chip,” in Proceedings of the 9th International Symposiumon Networks-on-Chip, ser. NOCS ’15, 2015, pp. 8:1–8:8.

[6] T Boraten, A. K. Kodi, “Mitigation of denial of service attack with hardware Trojans in NoC architectures”. In IEEE Parallel and Distributed Processing Symposium, 2016, pp. 1091-1100.

[7] N. Prasad, R. Karmakar, S. Chattopadhyay, I. Chakrabarti, “Runtime mitigation of illegal packet request attacks in Networks-on-Chip”. In IEEE Circuits and Systems (ISCAS), 2017, pp. 1-4.

[8] J. Frey, Q. Yu, “A hardened network-on-chip design using runtime hardware Trojan mitigation methods”, Integration, the VLSI Journal, Vol. 56, June 2016, pp.15-31.

[9] A. K Swain, K. R. Babu, S. N Satpathy, and K. K. Mahapatra, “FPGA prototyping and parameterised based resource evaluation of Network on Chip architecture. In Distributed Computing”, IEEE Conference on VLSI, Electrical Circuits and Robotics (DISCOVER),2015, pp. 227-231.

[10] A. Tran, "NoCTweak: A Highly parameterizable Simulator for Early Exploration of Performance and Energy of Network-on-Chip," VCL Lab, ECE Department, UC Davis, Davis, CA, USA, 2012.

[11] N. E. Jarger, L. Peh, “ On-chip Networks”, Synthesis Lectures on Computer Architecture, Vol. 4(1), 2009, pp. 1-141.