SOC IC Trojan Thwarting by Identification of Trojan

Josho Bennet John
P.G Scholar
College of Engineering
Guindy, Chennai

P. Nirmal Kumar
Assistant Professor
College of Engineering
Guindy, Chennai

ABSTRACT
IC Trojan horse is a serious technical problem faced by SOC developers. Most of the SOC developers use third party IP components for their design purpose. So comparing with a Trojan-less counterpart is impossible. Villasenor and Kim proposed bus architecture to detect Trojan activity where, when the circuit realizes the presence of a Trojan, it informs the CPU by an interrupt**.** In this paper, we are improving the bus architecture by proposing one more potential Trojan horse existence condition and also give a method to identify the exact cause that initiated the Trojan initiation. Then, we try to propose improvements to the bus architecture to make the thwarting and detection faster and efficient by using the obtained results of the above circuits.

Keywords
SOC, AMBA,TROJAN CIRCUIT,XILINX ,SPARTAN

1. INTRODUCTION
Now a day, SOCs are becoming complex and more difficult to test as, it has more components and more functions to perform. So, SOC manufacturers depend on third-party IP blocks and outsourced components for economic and technical reasons. Due to the outsourcing, there can be serious threat of presence of IC Trojan horses. Due to use of third-party IP blocks, it is not possible to compare with Trojan free counterpart of SOCs by using power and current comparison techniques which is discussed in [2].The solution was proposed in [1] by Villasenor and Kim. This solution was to have bus architecture to identify any unintended activity. The bus architecture proposed by them was a modification to the AMBA bus architecture. AMBA bus architecture details are in [3]. In the paper, they found out three conditions of Trojan activity which are malicious slave access, malicious bus lock detection and malicious wait detection. Malicious slave access is when the master accessing a slave which it is not allowed to access. Malicious bus lock is the condition of bus lock active for more than allowed time. Malicious wait is the condition of wait active for more than allowed time. In this paper, we propose one more Trojan existence condition which is data consistency Trojan. This is the condition when the master is accessing an allowed slave but is giving a wrong format data to the slave.

The efficient Trojan IC circuits are designed in such a way that it will not be active from power on. If it is active from power on, it can be easily detected by ATPG methods. This is explained in [4]. So, Trojan circuits are designed in such a way that the Trojan gets activated in run-time. The run-time activated Trojan is of two types as discussed in [5].The two types are time activated Trojans and input activated Trojans. Time activated Trojans are Trojans which get activated when a specified time has been lapsed after power on. Context activated Trojans are those Trojan circuits which get activated when a specific input context is received after power on as discussed in [4]. In this paper, we propose a method in which the bus architecture identifies the Trojan detected. It identifies weather the Trojan is time activated or context activated .If it is a time-activated Trojan, the circuit proposed will inform the time period after which the Trojan is activated to the user. If it is a context-activated Trojan, the circuit introduced will inform the context of inputs which causes the Trojan to be activated. Then the context or time data obtained can be used to modify the coming versions of the SOC or it can be used to reconfigure the affected master or slave at runtime by using reconfigurable blocks.

Another proposal of the paper is the concept of critical master. Some masters may be so critical that it cannot be inactive for the time of servicing of the Trojan detected interrupt. If on detecting Trojan circuit in such a master, bus architecture can be modified for switching such master with a backup master to avoid the time lapse. Backup master need not be there for every critical master. A reconfigurable block can serve the purpose for every critical master by feeding the reconfigurable block with the Trojan master id, the context data and time data. The paper is organized in 7 sections. Section 2 will discuss the IC Trojan horse in brief. In Section 3, the two types of Trojans based on initiation in brief. In Section 4, the previous work is discussed. In Section 5, the proposed modifications are discussed in detail. In section 6, Implementation results are given and section 7 is Conclusion.

2. IC TROJAN HORSE
ICs now a day are so complex that most of the components are outsourced. There is a possibility that they add some unintended circuit to the components during the mask making unintended behavior. This unintended behavior can be mere inactivity, work as spy circuit or even can cause serious consequences. For example, An IC Trojan horse can cause

self destruction of a fighter plane having such a Trojan circuit in it. Thus, the issue of IC Trojan horse is so critical in this context. The paper is going to discuss some proposed methods for thwarting this IC Trojan horses issue in system-on-chip. By using a bus architecture, which is a modification of the AMBA bus architecture.AMBA bus architecture is advanced microprocessor bus architecture which is employed in the Cypress PSOC and other modern programmable SOC architectures.

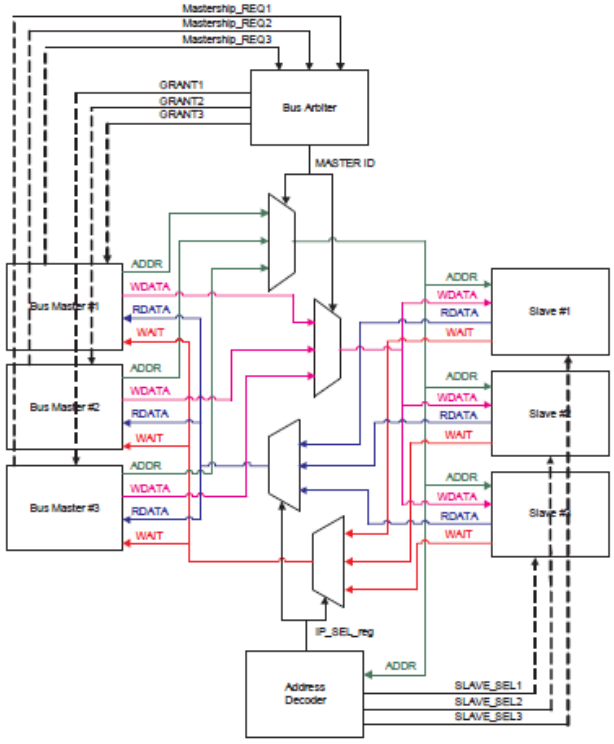

Fig 1: conventional SOC bus architecture

3. TYPES OF TROJAN INITIATION

In Section 4, methods are proposed is to identify the Trojan initiation. There are two ways in which Trojan circuits get initiated in runtime. These are time-based Trojan initiation and Context based Trojan initiation.

3.1 Time-based Trojan Initiation

In time-based Trojan initiation, the Trojan circuit gets initiated after a specified time period after power on. This is implemented by a timer which counts the clock to check the time after power on and when it reach the specified time, the Trojan circuit gets activated. As the trojan circuit activates on a particular time,these type of Trojans are called time-activated trojans

3.2 Context-based Trojan Initiation

In context-based Trojan initiation, the Trojan circuit gets initiated detecting a specified input context after power on. This is implemented by a comparator which compares the input context with a specified context and if equal, activates the Trojan circuit.

4. PREVIOUS WORKS

Trojan resistant bus architecture concept was put forward by Villasenor and Kim. In their paper, they introduced three ways of detecting a IC Trojan horse using a modified AMBA bus architecture. The three methods are unauthorized address access, malicious bus lock detection and malicious wait detection.

4.1 Unauthorized Address Access

This is a method of modifying the address decoder. In the conventional address decoder, based on the address accessed by the master, the slave is selected. In modified address decoder, some masters are not allowed to access some address ranges. If such an access occurs, the master is detected as a Trojan master as the address access is unintended.

4.2 Malicious bus lock detection

This is a modification to the bus arbiter. In modified bus arbiter, If the bus lock signal is active for more than some indented time, the master is detected as a Trojan. More access by the master is suppressed.

4.3 Malicious wait detection

This is a modification to the bus arbiter. In modified bus arbiter, If the wait signal is active for more than some indented time, the slave is detected as a Trojan. More access by the slave is suppressed. Finally a modified interrupt controller will be there which on detecting any of the above conditions will give an interrupt to the CPU informing that a Trojan has been detected. This proposal does not detect a Trojan circuit which modifies the data to an unintended format.

5. PROPOSED MODIFICATIONS

5.1 Data Consistency Checker

By the methods proposed by Villasenor and Kim, we cannot detect a Trojan circuit which changes the data send to an unintended format. We are covering the data Trojan problem with the proposed modification. The data bus is given as input to a data consistency checker block. The data consistency checker has the master id of the sending master and the slave id of the receiving slave as the other inputs. By using the master id and the slave id, according to the logic of the data consistency checker, the block will check whether the data format is as indented for the master and the slave. If the format is not as expected by the data checker, it will give a data consistence Trojan detected output.

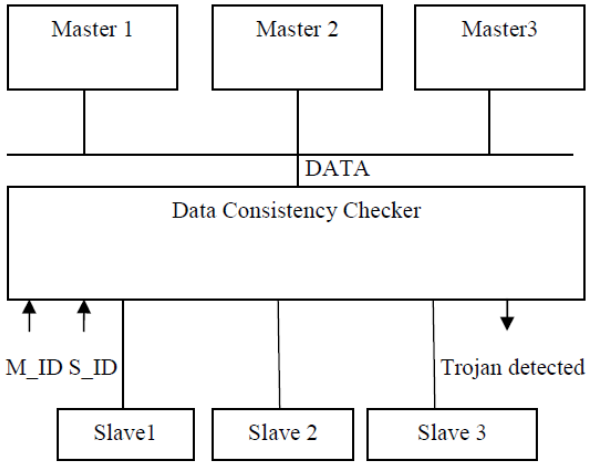

Fig 2: data consistency checker

5.2 Modified Interrupt Controller

Interrupt controller is modified to include the data consistency Trojan detected signal also. One more input is given to the AND gate which generates the Trojan detected interrupt. The added input is the data consistency Trojan detected signal.

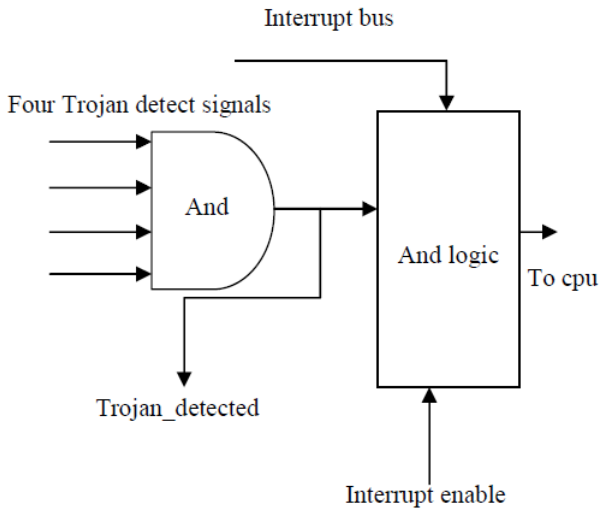

Fig 3: modified interrupt controller

5.3 Time Trojan Identification Logic

The idea underlying time Trojan identification is so simple. If at two times Trojan is detected after a specified time period so, we can state that the Trojan existing in the IC is a time Trojan and the time value after which the Trojan become active is a saved in a file which can be read and used to avoid the Trojan in the next version.

This is implemented by a counter which counts the clock to check the time and when it receives a Trojan, the time value is stored in a register. After some delay the register1 value is saved to register2.We compare the registers R1 and R2 in the next Trojan detection, If the values are equal, we can confirm that it is a time Trojan and value can be read from R2 or R1.

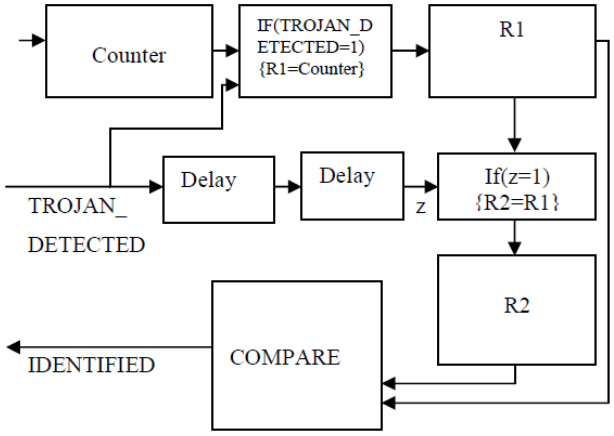

Fig 4: time Trojan identification logic

5.4 Context Trojan Identification Logic

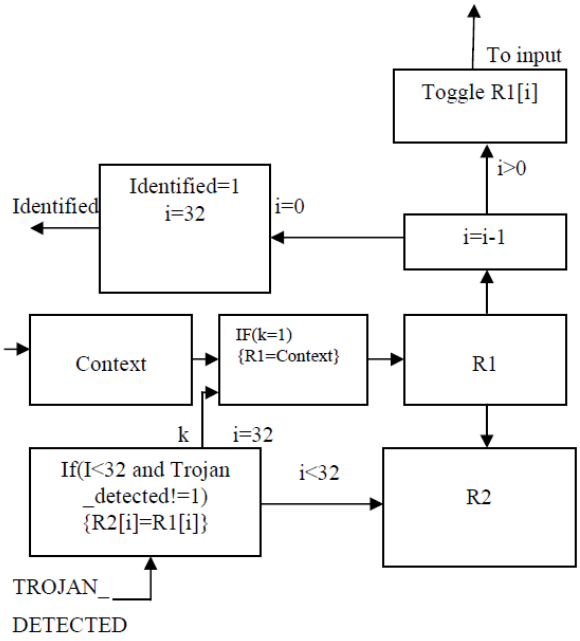

Fig 5: context Trojan identification logic

The idea underlying context Trojan identification is so simple. After receiving the Trojan for first time at runtime, the circuit goes to the test mode. In test mode, circuit will check the inputs that trigger the Trojan by toggling each bit in the received Trojan input and checking whether it will cause a Trojan output. If on toggling, the Trojan is not detected .The input bit is saved in Trojan identified register. The advantage of such a procedure is that, to identify the Trojan cause only double the number of inputs is needed as testing inputs. If such a method is not available, the device must check for Trojan activity for all of its inputs. As, if the Trojan activation circuit depend on only some input bits of the input sequence, there can be more inputs causing the Trojan activation. Here, the obtained sequence is used to reduce the number of test inputs to find out the Trojan dependant bits. This method thus reduces the number of test sequences required to identify the Trojan activation inputs.

Now the partial goal of finding out the cause that initiated the Trojan attack has been identified. We need to propose a method to thwart any IC attack using the obtained identification. For this, we have to design a reconfigurable master whose implementation depends on the identified Trojan horse. This needs much heavier work on this field to have an efficient implementation for this reconfigurable master circuitry. This reconfigured master or slave has to be switched with original master or slave when a Trojan detected bit comes. How this is done is next. This is done by using master switching.

5.5 Critical Master Switching

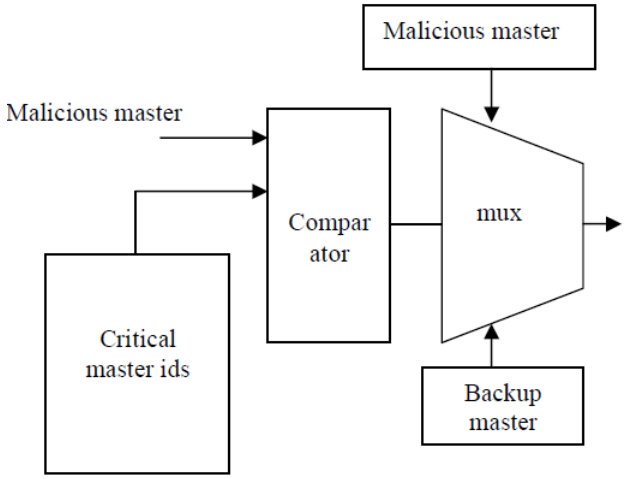

Fig 6: Critical master switching

It would not be efficient to do master switching for every master and slave so it is only done for critical masters. These are those masters which are so critical to the function of the device.

The critical master switching is implemented as in Fig.6.There will be a register which is a part of the bus architecture. The register will be saving the id value if the critical masters. On receiving a Trojan in a critical master, the comparator gives a 1 output. As the comparator output is 1.Output switches from the normal master to the backup master. The backup master can be reconfigurable circuit configured based on the identified Trojan. This reconfigurable master can be implemented using fpga which gives the output directly when the Trojan input is obtained. After getting the Trojan detected signal, the regular master is switched with the reconfigurable master.

6. IMPLEMENTATION AND RESULTS

The time Trojan identification was simulated using Xilinx simulator. Using the Xpower analyzer the power requirement was analyzed for spatan3a device with 8 bit counter register. The power obtained was 0.023W. The devices required for implementation of the logic in FPGA is obtained in Xilinx. The results obtained are as shown below.

Device Utilization Summary		
Logic Utilization	**Used**	**Available**
Number of Slice Latches	3	7,168
Number of 4 input LUTs	2	7,168
Number of occupied Slices	4	3,584
Number of Slices containing only related logic	4	4
Number of Slices containing unrelated logic	0	4
Total Number of 4 input LUTs	2	7,168
Number of bonded IOBs	11	311
IOB Latches	1	
Number of BUFGMUXs	2	24
Average Fanout of Non-Clock Nets	2.00	

Fig 7: Result of time trojan identification logic

From the observation, it is found that the logic can be implemented in a very efficient way with less power and gate requirement. Then the Context Trojan identification was simulated using Xilinx simulator. Using the Xpower analyzer the power requirement was analyzed for spatan3a device. The power obtained was 0.028W.

The devices required for implementation of the logic in FPGA is obtained in Xilinx. The results obtained are as shown below. Context Trojan identification in test mode can be done by an FPGA device with very less power requirement and less device utilization.

Device Utilization Summary		
Logic Utilization	**Used**	**Available**
Number of Slice Latches	16	7,168
Number of 4 input LUTs	5	7,168
Number of occupied Slices	12	3,584
Number of Slices containing only related logic	12	12
Number of Slices containing unrelated logic	0	12
Total Number of 4 input LUTs	5	7,168
Number of bonded IOBs	18	311
IOB Latches	8	
Number of BUFGMUXs	1	24
Average Fanout of Non-Clock Nets	1.57	

Fig 8: Result of context trojan identification logic

Tables of the results obtained after simulating it in Xilinx is given below. The tables are screenshots of the results page itself.

7. CONCLUSION

The above methods proposed will be solving the Trojan problem in an SOC device to some extent in a nice and efficient

way. But there are still serious chances for failure of the methods. So, more research in this field is a necessity. With the improving device technology, more devices and faster speed can be achieved in future implementations. So in future, more problems can be resolved without much reduction in efficiency.

REFERENCES

[1] Villasenor.J.D and Lok-Won-Kim,"A system-on-chip bus architecture for thwarting integrated circuit trojan horses, *IEEE transactions on very large scale integration (VLSI) ciruits*.pages-1921-1926

[2] X. Wang, H. Salmani, M. Tehranipoor, and J. Plusquellic, "Hardware Trojan detection and isolation using current integration and localized current analysis," in *IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems (DFTVS'08)*, 2008, pp. 87 – 95.I.

[3] AMBA specification

[4] Susmit jha and Sumit kumar jha, "Ramdomisation based probabilistic approach to detect trojan horses"*in IEEE high assurance systems Engineering Symposium, 2008.*

[5] Jim Plusquellic, University of New Mexico"Taxonamy of Trojans for IC Trust"

[6] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, 2008, pp. 51 – 57.

[7] R. Rad, J. Plusquellic, and M. Tehranipoor, "Sensitivity analysis to hardware Trojans using power supply transient signals," in *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*,2008, pp. 3 – 7.

[8] M. Banga and M. S. Hsiao, "A novel sustained vector technique for the detection of hardware Trojans," in *22nd International Conference on VLSI Design*, 2009, pp. 327 – 332.

[9] M. Abramovici and P. L. Levin, "Protecting integrated circuits from silicon Trojan horses," January/February 2009, http://www.mil-embedded.com/articles/id/3748.

[10] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou,"Designing and implementing malicious hardware," in *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats(LEET)*.San Francisco, California: USENIX Association, 2008, pp. 1–8.

[11] T. Kean, D. McLaren, and C. Marsh, "Verifying the authenticity of chip designs with the Design Tag system," in *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, 2008, pp. 59 – 64.

[12] R. S. Chakraborty, S. Paul, and S. Bhunia, "On-demand transparency for improving hardware Trojan detectability," in *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, 2008,pp. 48 – 50.

[13] D. Flynn, "Amba: enabling reusable on-chip designs," *Micro, IEEE*, vol. 17, no. 4, pp. 20 – 27, 1997.

[14] F. Lin, H.Wang, and J. Bian, "Hw/sw interface synthesis based on Avalon bus specification for nios-oriented soc design," in *Field-Programmable Technology, 2005. Proceedings. 2005 IEEE International Conference on*, 2005, pp. 305 – 306.

[15] X. Xing, C. Zezong, J. Jing, and K. Hengyu, "Porting from

[16] wishbone bus to avalon bus in soc design," in *Electronic Measurement and Instruments, 2007. ICEMI '07. 8th International Conference on*, 2007, pp. 862 – 865.

[17] S. Murali and G. D. Micheli, "An application-specific design methodology for stbus crossbar generation," in *Design, Automation and Test in Europe, 2005. Proceedings*, 2005, pp. 1176–1181

[18] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *IEEE Symposium on Security and Privacy (SP'07)*, 2007, pp. 296 – 310.

[19] F. Wolff, C. Papachristou, S. Bhunia, and R. Chakraborty, "Towards Trojan-free trusted ICs: problem analysis and detection scheme," in *Proceedings, Design Automation and Test in Europe (DATE'09)*, Munich, Germany, March 10-14, 2008, pp. 1362–1365.

[20] J. Li and J. Lach, "At-speed delay characterization for IC authentication and Trojan horse detection," in *IEEE International Workshop on Hardware-Oriented Security and Trust(HOST'08)*,2008,pp.8–14.