ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

AHB BUS PROTOCOL BASED SOC ARCHITECTURE

[1]Ms.JANANIPRIYA.G, [2]Ms. POORANI.P M.E

[1]UG Scholar, Department of ECE , [2]Assistant Professor, Department of ECE
[1,2] Arulmigu Meenakshi Amman College of Engineering, Vadamavandal.
[1]jananipriyaganesh@gmail.com, [2]poorani_1oct@yahoo.in

***ABSTRACT-* Communications systems are increasingly reliant on system on chip (SOC). As the complexity and size of SOCs continues to grow, the risk of hardware based “Trojan" attacks also increases. Trojan attacks in integrated circuit (IC) may alter system function during the design or manufacturing process. Normally, more effort has been given to software security and little attention has been given to hardware security even less attention to how to detect and respond to run time Trojan attacks. We propose Trojan-resistant AMBA bus architecture suitable across a wide range of System on Chip (SOC) systems that can minimize performance degradation and maximize seamless system operation despite the function replacement. This approach is highly feasible in that it is not required to specially manage system software and other normal system hardware functions for the replacement.**

***Index Terms*-Hardware Trojan horses, dynamic function replacement, system-on-chip, advanced microcontroller bus architecture bus arbitration.**

I.INTRODUCTION

Design outsourcing has become increasingly common over the past 15 years for ICs generally and in particular for SOCs. The incorporation of third party IP designs represents an important potential point of vulnerability. Outsourced designs are typically provided using register transfer level (RTL) descriptions or hard macro cell designs with the result that there is no trusted golden model to use for comparison. In addition simulation models delivered along with third party IPs can themselves be untrustworthy and could be designed to block the modeling of the impact of an activation trigger. In such an environment, a trustworthy system-level model may be difficult or impossible to obtain. Most fundamentally, a true Trojan would involve an attack designed to remain hidden and inactive until triggered either internally or externally and would be extremely difficult to detect during verification. In this regard, outsourced designs need protection against Trojan attacks.

The terms hardware Trojan horse and hardware Trojan are used to describe malicious hardware within an IC or hardware system that is designed to escape detection during verification and to then launch an attack post-deployment. Despite the enormous amounts of effort that has been devoted to software security, relatively little attention has been directed to hardware security in general and even less to the issue of how to detect and respond to run-time Trojan attacks. The run-time handling of Trojan attacks can be partitioned into the two overall tasks of 1) detection and 2) response. The first task involves identifying the presence of a Trojan attack and its source. In some cases, such as when an attack shuts down the system functionality, the presence of an attack may be relatively easy to determine. Other attacks such as those involving a Trojan that works in the background to exfiltrate data to an off-chip location may be more difficult to identify. The second task and in particular on measures allowing the IC to continue operation despite the presence of a Trojan corrupting one of the functional blocks within the chip. More specifically, consider the use of embedded reconfigurable logic or external reconfigurable logic devices to replace system functionalities that are compromised by a Trojan. While not all potential attacks can be resolved in this manner, there is a significant subset of attacks that can indeed be significantly mitigated.

II.PREVIOUS WORK

The previous work relevant to the current paper lies in two broad areas: In the first category, there have been a number of publications that address methods to detect maliciously altered hardware *pre-deployment.* In [6], the authors have developed a scalable hardware Trojan detection and diagnosis method. The approach uses circuit segmentation and gate-level characterization to detect and diagnose hardware Trojan horses even in large circuits. Tehranipoor and Koushanfar presented a classification of hardware Trojans and a survey and analysis of published techniques for Trojan detection in [7]. They also described recently proposed techniques used in designing for hardware trust. In [8] and [9], authors described a set of anti-Trojan design methods and countermeasures that can increase IC security by making it possible to identify and quarantine a functional block found to contain a Trojan. Embedded reconfigurable logic has been used to implement function designs which may be updated in post deployment, unexpectedly added to systems after IC fabrication or temporarily used to perform IC testing before deployment. Conventional SOC bus structure is used in [11]. Christo Ananth et al. [2] proposed a system, this paper presents an effective field programmable gate array (FPGA)-based hardware implementation of a parallel key searching system for the brute-force attack on RC4 encryption. The design employs several novel key scheduling techniques to minimize the total number of cycles for each key search and uses on-chip memories of the FPGA to maximize the number of key searching units per chip. Based on the design, a total of 176 RC4 key searching units can be implemented in a single Xilinx XC2VP20-5 FPGA chip. Operating at a 47-MHz clock rate, the design can achieve a key searching speed of 1.07 x 107 keys per second. Breaking a 40-bit RC4 encryption only requires around 28.5 h.

The Proposed architectural features of System-on-Chip can minimize performance degradation and enables seamless system operation.

III.DFR SEQUENCE AND SEAMLESS SYSTEM OPERATION

The on-the-fly replacement of a function creates some critical obstacles to seamless system operation. During the configuration, a bus master may attempt to access the function being configured because other system functions are not aware of the replacement activity. This obstacle is addressed by taking advantage of delayed response capabilities that are supported in most bus protocols (e.g., the HREADY signal in AHB bus specification). When a slave cannot serve a request immediately, the slave postpones the activation of the acknowledge signal and the accessing master needs to wait for the slave's response until the acknowledge signal is activated by the slave. Similarly, the DFR controller can delay any access of bus masters going to the replacing function during the replacement processes. This method allows other system functions to locally operate even during the configuration. The drawback of this method is that the delayed response locks bus operation and results in temporary halt of bus operation.

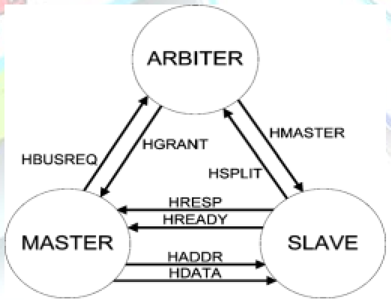


Fig 1 AHB Based Connection among Bus Arbiter, Master and Slave for Bus Split Operation

An proposed architectural method uses the bus split which is used to address the stopped bus transaction. This method allows the reconfigurable logic module to nullify the access on it so that other bus masters can utilize the system bus during the replacement. Fig.1 shows an AHB-based connection

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

among a bus arbiter, master, and slave modules for the bus split operation.

An example of a bus split occurrence is as follows. First, a bus master asserts the HBUSREQ signal to request bus master ship. Based on the arbitration process of the arbiter, the HGRANT signal is activated to properly grant the request for the bus master ship. Then the bus master sends address, data and control signals to a slave through the bus. When the slave module cannot serve the master's access sufficiently, the slave module can assert SPLIT in the HRESP signal and activate a bit representing the current master module in the HSPLIT signal. After the split request from the slave, the master nullifies its access to the slave and waits for the next bus master ship grant. The arbiter deactivates the HGRANT connected to the master and performs the arbitration process excluding the master which accessed the salve. Note that the master is not considered in the arbitration process of the arbiter until the slave deactivates the HSPLIT. When the slave module is ready to serve the master's access, it deactivates the HSPLIT and the arbiter now includes the pending master into its arbitration process. When the arbiter grants the bus master ship to the pending master, the master can resume its task by accessing the slave.

Likewise the function replacement can utilize the bus split to properly delay system accesses on the replacing function and allow other masters to use the system bus during the replacement time. The method is based on a modified SOC architecture including an embedded reconfigurable logic or an auxiliary FPGA used for function regeneration, bus architectures for isolating malicious hardware module and system performance maintenance, as well as hardware modules to manage reconfiguration and reliable interface signaling. This approach provides not only a way of regenerating compromised on-chip function, but does so in a manner that enables seamless system operation during replacement and minimizes performance degradation.In sum, the combination of delaying the response of the replacement function, splitting bus transaction and updating register setting values in the DFR controller provides four advantages in the seamless system operation.

1) In spite of the unexpected function replacement, the method maintains system operation context.
2) It limits the region of operation halt to the function of a module which is being replaced and allows other functions to continue their operations even during the unexpected replacement activity.
3) It eliminates the potential for false responses from the slave during the configuration.
4) It minimizes system performance degradation potentially caused by the replacement.

IV.SOC BUS STRUCTURE AND OPERATION

System-on-a-chip (SOC) technology is the packaging of all the necessary electronic circuits and parts for a system on a single integrated circuit (IC), generally known as a microchip. A SOC consists of multiple heterogeneous functional blocks. To enable data flow among these blocks, a SOC bus is used to interconnect one or more processing cores to each other and to the surrounding interface logic. This approach makes it possible to create functional blocks that are specialized for and therefore highly efficient at computation for specific tasks. Traditional SOCs are designed under the assumption that both the functional blocks themselves and the bus management logic are free from intentionally-inserted malicious circuitry. However, as SOC complexities increases, the number of vulnerabilities also increases as well. In this environment, it is no longer possible to assume that a chip is always free of corrupted circuitry. Instead it is more practical to design chips with the understanding that one or more blocks may prove to be corrupted and that on-the-fly identification and replacement of such blocks can be critical for the operation of the systems in which such SOCs reside. Thus in the present work, we uses SOC based AMBA bus to enable such replacement in the background of common bus architectures.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

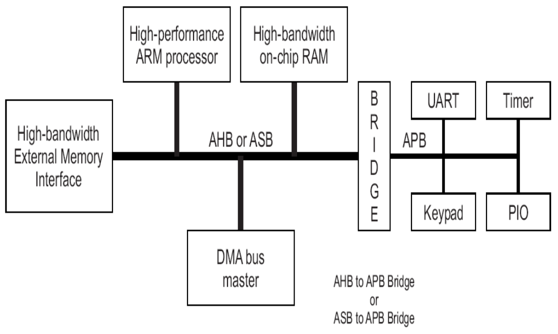

High-performance ARM processor
High-bandwidth on-chip RAM
High-bandwidth External Memory Interface
AHB or ASB
B R I D G E
UART
Timer
APB
Keypad
PIO
DMA bus master
AHB to APB Bridge or ASB to APB Bridge

Fig 2 AMBA Hierarchical Bus Architecture

A.AMBA AHB BUS

Buses are shared communication media used by devices to “talk to” each other both on-chip and off-chip. The communication actions which take place can carry both data and control structures. On chip communication standard called Advanced Microcontroller Bus Architecture (AMBA). The basic AMBA bus architecture is shown in fig1. AMBA specification defines three distinct bus architectures. Advanced high-performance bus (AHB) and advanced-system bus (ASB) are both for high clock frequency module. But ASB is hardly used in peripheral module in popular SOC system recently. The last bus architecture is advanced peripheral bus (APB), which is mainly used for low-power peripheral modules.

AHB is a bus protocol introduced in Advanced Microcontroller Bus Architecture version 2 published by ARM Ltd company. Its main features are burst transfer, split transactions, multiple masters, wider address bus, larger data paths and pipelined operation. AHB bus is a multi-master with arbitration, putting the address on the bus, followed by the data. It also supports wait-state insertion and has a data-valid signal (HREADY). This bus differs in that it has separate read (HRDATA) and write (HWDATA) buses. These bus connections are multiplexed rather than making use of a tristate multiple connections.

All bus operations are initiated by bus masters, which also can serve as a slave. The master-generated address is decoded by a central address decoder that provides a select signal to the addressed bus slave unit. The bus master can "lock" the bus, reserving it with the central arbiter for a series of locked transfers. The slave unit has the option to terminate a transaction as an error, signals the master to retry, or split the transaction for later completion. Split transactions enable the slave to defer the operation until it's able to accomplish it thereby releasing the bus for other accesses. The slave signals a split transaction and saves the master number (HMASTER). When ready to complete the transaction, the slave signals the arbiter with the master number. When the arbiter grants bus access to the master, it restarts the transaction. No master can have more than 1 pending split transaction. It is a synchronous bus that supports bursts and pipelining of accesses to improve throughput. The AHB system bus and APB peripheral bus are linked through a 'bridge' that acts as the master to the peripheral bus slave devices.

AHB supports multiple masters (either through a central arbiter, or through slave level arbiters in the case of a multi-layer AHB-Lite system). The arbiter has the task of determining which master gets to do an access.

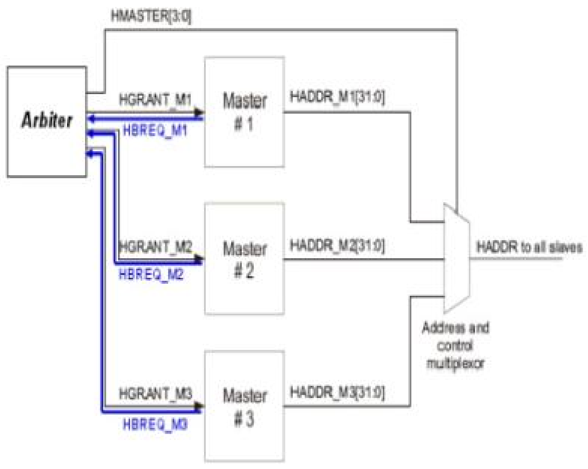


Fig 3 Bus Arbiter with three masters

B.AMBA BUS ARBITRATION

The simple bus arbiter with three master are shown in fig 3. The AHB Arbiter is used in AHB multi-

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

master systems to arbitrate the access to the AHB bus. The AHB Arbiter is basically a “traffic controller” which allows the AHB bus to be shared between multiple bus masters such as processors, DMA controllers, and peripheral core master interfaces. The AHB Arbiter uses a round robin priority scheme with Master0 having the default priority. This priority scheme assures that each master equally has its turn at acquiring and completing an AHB bus transaction. Each inactive master is locked out (HLOCK) while the active master has access to the bus to prevent contention. The AHB Arbiter steers all the AHB HWDATA, HADDR, HTRANS, HWRITE,

HSIZE and HBURST signaling from each master to the AHB system bus.

Every transfer has an address/control phase and a separate data phase. They're both pipelined (able to start the next transfer's arbitration and address phase while finishing the current transfer).The address transfer is always followed by the data phase. A slave (memory or peripheral device which accepts a read or write request from a master) can prolong the transfer (add wait states) using the HREADY signal. Separate unidirectional buses for read (HRDATA) and write (HWDATA) are used. AHB supports bursts which can either be of undefined-length or fixed length (4, 8 or 16 beats). Bursts may be performed to a fixed address (e.g. for FIFO access), increment addresses (in steps of a single increment equal to the size of the access) or wrap (where a critical word within a cache line is accessed first). The address from a master is decoded by a central address decoder that provides a select signal to one of the slaves.

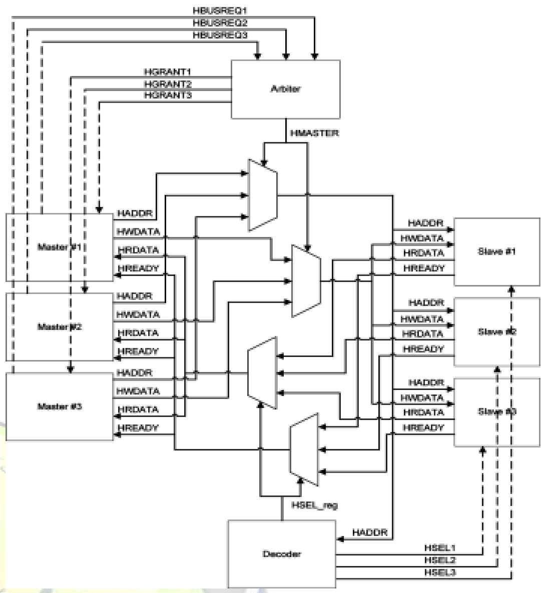


Fig 4 AMBA SOC Bus Interconnections, Showing Master and Slave Devices, an Arbiter, an Address Decoder and Various Multiplexers

Slaves may respond to accesses by the master by signaling OK, or by reporting an error. In the full AHB system (but not AHB-Lite), slaves may also give a retry response, or the less commonly used split response. Split transactions let the slave to delay completion of the access until ready but to free the bus for other accesses by a different master. The slave records the number of the master and signals the arbiter when the split transfer can complete. When the arbiter re-grants the bus to that master, it restarts the transaction. A master can have only

one pending split transaction.

V.EXPERIMENTAL RESULTS

Using Modelsim software, verilog code for master, slave and arbiter are synthesized and simulated using altera quartus tool. The experimental result for reading and writing the data from master to slave has been shown. Figure 5 and figure 6 shows the simulation result of master and slave devices. Initially the global signals (clock and reset signals) are enabled. Based on burst signal and transfer signal from granted master, 32 bit data’s are written from master to selected slave. After the master device has started the transfer, the slave

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

device then determines how the transfer should progress. The slave device receives HSEL signal which comes from decoder to recognize it is chosen or not. Whenever a slave device is chosen, it must provide the response which indicates the status of the transfer like complete or error. When HREADY which comes from the selected slave is equal to HIGH, it indicates that the transfer has finished on the bus. Otherwise, it means the transfer should be extended or the ERROR, RETRY and SPLIT may happen. HRESP signal which comes from the selected slave to granted master, it indicates that the transfer may be OKAY, ERROR, RETRY or SPLIT. Similarly, data's from slave transferred to master based on the control signals.

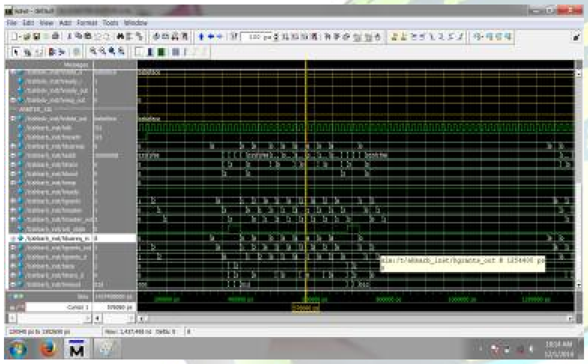


Fig 5 Master Output

Figure 7 shows the simulation result of arbiter. The arbiter determines which master devices has its address and control signals sending to all of slave devices. A central decoder is required to control the read data and response signal from multiplexer which selects appropriate signal from the slave involved in the transfer. Based on HBUSREQx signal (from master to arbiter), arbiter will determine requested (granted) master.

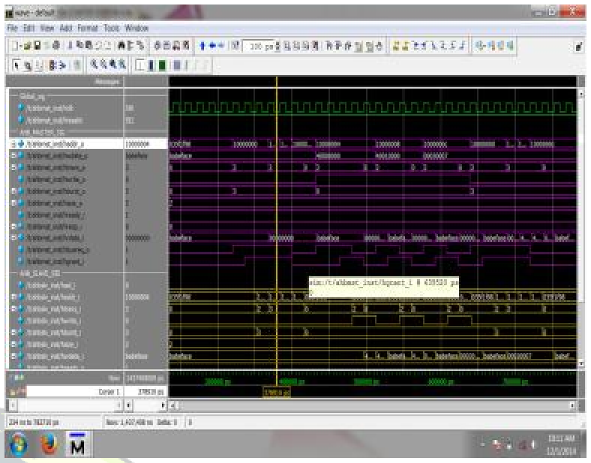


Fig 6 Slave Output

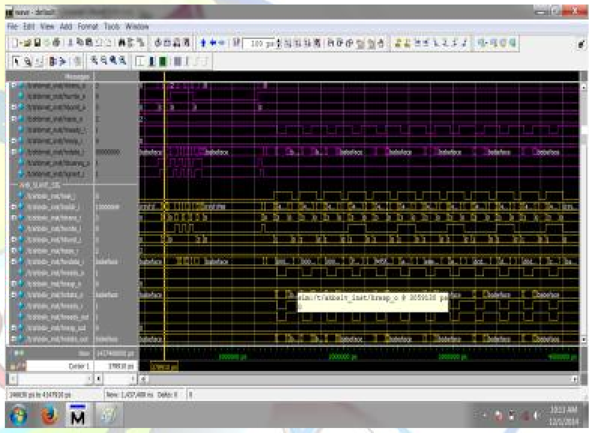
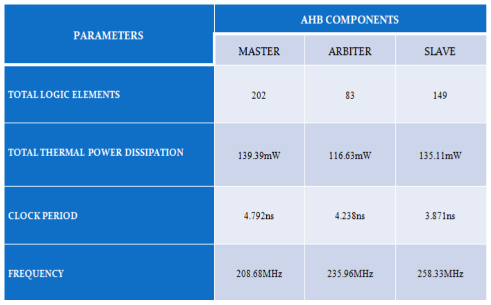


Fig 7 Arbiter Output

Using Altera Quartus tool, we can analysis the area, power and frequency for master, slave and arbiter shown in fig 8.



PARAMETERS	AHB COMPONENTS		
	MASTER	ARBITER	SLAVE
TOTAL LOGIC ELEMENTS	202	83	149
TOTAL THERMAL POWER DISSIPATION	139.39mW	116.63mW	135.11mW
CLOCK PERIOD	4.792ns	4.238ns	3.871ns
FREQUENCY	208.68MHz	235.96MHz	258.33MHz

Fig 8 Analysis of Some Parameters

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

VI.CONCLUSION

We proposed SOC architecture with AHB protocol that is resistant to hardware based Trojan attacks. We implement a 32 bit data transfer in SOC with high speed and highly reliable. SOC architecture enables seamless system operation during replacement and minimizes performance degradation. AHB protocol designed with reduced area, power and time, so the cost to protect the system against Trojan was found to be low.

REFERENCES

[1] R. Saleh, S.Wilton, S.Mirabbasi,A.Hu, M. Greenstreet, G. Lemieux,P. Pande, C. Grecu, and A. Ivanov, "System-on-chip: Reuse and integration,"in*Proc. IEEE*, Jun. 2010, vol. 94, no. 6, pp. 1050–1069.

[2] Christo Ananth, Muthamil Jothi.M, M.Priya, V.Manjula, "Parallel RC4 Key Searching System Based on FPGA", International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE), Volume 2, Special Issue 13, March 2016, pp: 5-12

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

[4] J. Aarestad, D. Acharyya, R. M. Rad, and J. Plusquellic, "Detecting Trojans though leakage current analysis using multiple supply padIDDQ," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 4, pp. 893–904,Dec. 2010.

[5] L. Jianwen and J. Chuen, "A System-on-Chip dynamically reconfigurable FPGA platform for matrix inversion," in *Proc. IEEE Int. Sym.Integr. Circuits*, 2007, pp. 465–468.[27] D. Flynn, "AMBA: enabling reusable on-chip designs," *IEEE Micro*,vol. 17, no. 4, pp. 20–27, 1997.

[6] H. Salmani,M. Tehranipoor, and J. Plusquellic, "A Novel technique for improving hardware Trojan detection and reducing Trojan activation time," *IEEE Trans. Very Large Scale Integration (VLSI) Syst.*, vol. 20,no. 1, pp. 112–125, Jan. 2012.

[7] M. Tehranipoor and F. Koushanfar, "A survey of hardware Trojan taxonomy and detection," *IEEE Design Test Comput.*, vol. 27, no. 1, pp.10–25, 2009.

[8] L. Kim and J. Villasenor, "A system-on-chip bus architecture for thwarting integrated circuit Trojan horses," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 10, pp. 1921–1926, Oct. 2011.

[9] L. Kim, J. Villasenor, and C. Koc, "A Trojan-resistant system-on-chip bus architecture," in *Proc. IEEE MILCOM. Conf.*, Boston, MA, Oct.2009, pp. 1–6.
[10] "ARM, AMBA specification (rev 2.0)," 1999 [Online]. Available: http://www.arm.com/products/solutions/AMBA_Spec.html

[11]Lok Won Kim, and John D.Villasenor, "Dynamic Function Replacement for System-on-Chip Security in the Presence of Hardware- Based attacks" *in IEEE transactions on reliability*,vol.63,no.2,june2014,
pp. 661-675.