Formal Analysis of Macro Synchronous Micro Asychronous Pipeline for Hardware Trojan Detection

F. K. Lodhi [1], S. R. Hasan [2], O. Hasan [1] and F. Awwad [3]

[1] Sch. of Elect. Engg. and Comp. Sc., National University of Sciences and Technology (NUST), Islamabad, Pakistan

[2] Department of Electrical and Computer Engineering, Tennessee Technological University, Cookeville, TN, USA

[3] College of Engineering, United Arab Emirates University, Al-Ain, UAE

{faiq.khalid, osman.hasan}@seecs.nust.edu.pk, shasan@tntech.edu, f_awwad@uaeu.ac.ae

***Abstract* —Globalization trends in integrated circuit (IC) design using deep submicron (DSM) technologies are leading to increased vulnerability of IC against malicious intrusions. These malicious intrusions are referred to hardware Trojans. One way to address this threat is to utilize unique electrical signatures of ICs, and any deviation from this signature helps in detecting the potential attack paths. Recently we proposed hybrid macro synchronous micro asynchronous (MSMA) pipeline technique while utilizing, non-conventional, asynchronous circuits to generate timing signature. However, traditionally generating these timing signatures with environmental uncertainties require extensive simulations. It is known to the engineering community that computer simulations have its limitations due to the associated heavy computational requirements. In this paper, as a more accurate alternative, we propose a framework to detect the vulnerable paths in the MSMA pipeline for hardware Trojan detection using formal verification methods. In particular, the paper presents a formal model of the MSMA pipeline and its verification results for both functional and timing properties.**

Keywords— Hardware Trojan; Model Checking; nuXmv; MSMA; Asynchronous.

I. INTRODUCTION

With the globalization of integrated-circuit chip design-process, the chances of malicious hardware design intrusion, known as hardware Trojan, have grown tremendously [1][2][3]. Hardware Trojans can lead to many unwanted activities, including leaking confidential information, changes in the timing characteristics of the circuits, malfunctioning, denial of service, counterfeiting and the list goes on [1][4][5]. Researchers have developed various techniques to detect the hardware trojans. Some of the prominent works include micro-architecture modification to improve triggering of the potential Trojan payload during test in [6][7] and the usage of inherent error detection of quasi delay insensitive (QDI) architectures to detect malicious intrusions.

Recently, Lodhi et. al. proposed a macro synchronous micro asynchronous (MSMA) pipeline based hardware Trojan technique, which is based on a hybrid quasi delay insensitive (QDI) pipeline architecture[8][9]. It uses the intrinsic error prevention property to generate the timing signatures with environmental uncertainties. However, similar to other timing signatures this technique also obtains the timing signatures through the extensive simulations of a golden circuit. These simulations can be costly, time consuming and cannot encompass all the possible input conditions. Formal methods have been successfully used for overcoming these limitations for the detection of malicious intrusions [10]. Zhang et. al. [11] have proposed an approach that uses multistage assertion based verification, code coverage analysis, redundant circuit removal, and equivalence analysis along with the sequential automatic test pattern generations. Similarly, researchers have proposed many frameworks for the formalization and verification of the security property of IP cores to identify the malicious intrusions[12]. Rathmair et. al. [13] have proposed a counter-example based methodology to detect the potential attack paths in the system. However, to the best of our knowledge so far no work is reported, which deals with intentional malicious enhancement of hardware design instead of merely identifying the vulnerability of hardware design. The main idea in this work is to intrude the given model with malicious behavior and try to identify the most vulnerable path of the system, by analyzing the counter examples. This method can identify the potential attack path in the given systems and thus can be used to enhance the performance of hardware Trojan detection techniques.

In this paper, we propose a generic a methodology to detect the potential attack paths in MSMA pipelines based on the framework presented in [13]. We have identified a set of functional and timing properties that can be affected by malicious behavior. These properties are verified in a symbolic model checker (nuXmv) for the behavioral model of the MSMA, to extract the counter examples, which provide very useful information for deducing potential attack paths.

II. PRELIMINARIES

In this section, we give a brief introduction to the nuXmv model checker and MSMA pipeline architecture to facilitate the understanding of the rest of the paper.

A. Macro Synchronous Micro Asynchronous (MSMA)

This section describes a MSMA technique based on the NCL pipeline [9][14]. A traditional NCL pipeline architecture is primarily composed of three main blocks: NCL based registers, combinational logic block and completion detection scheme. This pipeline architecture is a hybrid form of the synchronous and asynchronous pipelines. In this design, between every synchronous pipeline stage, there is a NCL based pipeline to replace the combinational logic block. The implementation of the MSMA pipeline, as shown in Figure 1, can be divided into two main blocks: Asynchronous and Synchronous. The asynchronous block includes the NCL pipeline, which consists of three sub-blocks: Dual rail Registers, Combinational Block and Completion Detection.

978-1-4673-6576-5/15/$31.00 ©2015 IEEE.

Since NCL is based on dual rail encoding therefore all these blocks are implemented using dual rail encoding. The synchronous block consists of synchronous registers at both ends of the MSMA pipeline [9].

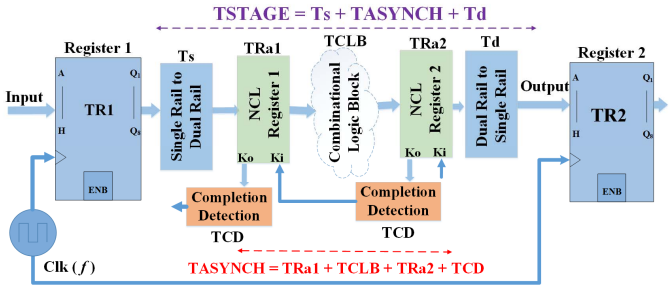

Figure 1. Macro Synchronous Micro Asynchronous Pipeline, where TR1, TR2, TRa1, TRa2, TCLB , TCD, Ts, Td represent delay in Synchronous registers 1 & 2, Asynchronous (NCL) registers 1 & 2, Combinational Logic Block, Completion Detection, Single to dual rail and Dual to single rail [9]

1) Timing Analysis of MSMA

The propagation delays of register1, MUX, combinational block, register 2 and completion detection blocks are assumed as T_{R1}, T_{MUX} , T_{CLB}, T_{R2} and T_{CD}, respectively, as shown in Fig. 2. Equation (1) provides the latency of the NCL handshake signals (T_{ASYNCH}), which is the duration from the availability of data at register 1 to the assertion of the acknowledgement signal through the completion detection unit.

$$T_{ASYNCH} = T_{R1} + T_{MUX} + T_{CLB} + T_{R2} + T_{CD} \quad (1)$$

NCL pipeline consists of two states of data communication: data and null states. These two states have different latencies, and we call them as data latency (T_{DATA}) and null latency (T_{NULL}), respectively. The total latency of the pipeline can be written as:

$$T_{TOTAL} = T_{DATA} + T_{NULL} \quad (2)$$

2) Hardware Trojan Detection in MSMA pipeline

The combinational unit of MSMA is an asynchronous circuit, i.e., based on event driven handshake signals. We took the advantage of this inherent characteristic of NCL and measured one complete handshake cycle to obtain a unique timing signature. Figure 2 shows the implementation of test vector insertion to generate timing signature. The test vectors TA and TB are 2 bit dual rail signals. The two LSBs of TA [0:3] are used to select whether the circuit is in test mode or in normal operational mode.

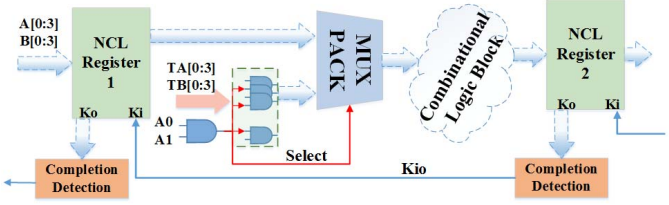

Figure 2. Test vector insertion to obtain timing signature in MSMA [8][9]

B. *nuXmv Model checker*

The nuXmv symbolic model checker [15] extends the capabilities of the NuSMV by complementing its verification techniques using SAT algorithms for finite state systems. For infinite state systems, it introduces new data types of **`Integers`** and **`Reals`** and also provides the support of Satisfiability Modulo Theories (SMT), using MathSAT, for verification [16].

The system that needs to be modeled is expressed in the nuXmv language, which supports the modular programming approach where the overall system is divided into several modules that interact with one another in the **`MAIN`** module. The properties to be verified can be specified in nuXmv using the Linear Temporal Logic (LTL) and Computation Tree Logic (CTL). The LTL specifications are written in nuXmv with the help of logical operations like, AND (**`&`**), OR (**`|`**), Exclusive OR (**`xor`**), Exclusive NOR (**`xnor`**), implication (**`->`**) and equality (**`<->`**), and temporal operators, like Globally (**`G`**), Finally (**`F`**), Next (**`X`**) and Until (**`U`**) notations. Similarly, the CTL specifications can be written by combining logical operations with quantified temporal operators, such as Exists-Globally (**`EG`**), Exists-Next-State (**`EX`**) and For-all-Finally (**`AF`**). In case a property turns out to be false, a counterexample in the execution trace of the FSM is provided.

We have chosen nuXMV for our work, because of its distinguishing ability to handle real numbers and implicit handling of state counters. Thus the continuous values of delays and uncertainties can be modeled properly, which cannot be done in other model checking tools in such a straightforward manner.

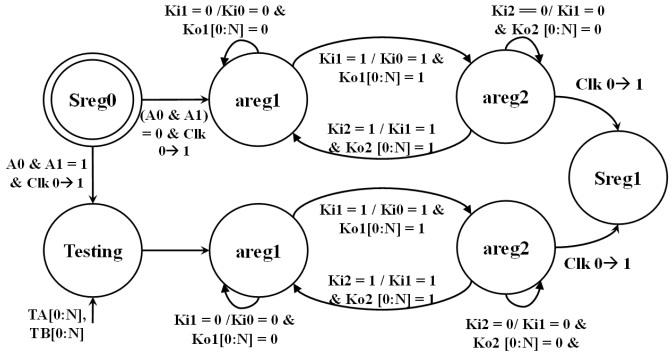

Figure 3. FSM of Hardware Trojan Detetion in MSMA

III. NUXMV MODELING OF MSMA

The state machine of the considered MSMA [9] is shown in Figure 3. This state machine is divided into normal and testing modes. The upper half represents the normal operation and it shows that the system will be in normal mode if it does not receive both A0 and A1 as high. Similarly, the system moves to the testing mode when both A0 and A1 are high and in this mode the system extracts the delay samples for the specific test input vectors TA[0:N] and TB[0:N]. These delay samples are used to detect the malicious intrusions. In both modes, asynchronous blocks operate normally with different input conditions as shown in Figure 2. This state machine is implemented in nuXMV to model the behavior of MSMA pipeline with HT detection. In our nuXMV modeling, shown in Figure 23 the states **`sreg`**, **`areg`**, **`sr2dr`**, **`dr2sr`** and **`add`** represents the synchronous register, asynchronous register, single rail to dual rail converter, dual rail to single rail

converter and NCL adder, respectively. In normal operation mode, this state machine follows the NCL pipeline protocol such that when a receiving register (**areg2**) receives any DATA request (Ki = 1) it receives the data and generates the multiple acknowledgement signals (Ko1 = 1). Then, based on the acknowledgement signals (Kos) receiver (**areg2**), it sets the Ki of the sender (areg1) to high. The nuXMV implementation can be found from [17].

IV. Verification of MSMA Pipeline

We used the version 1.0.1-win64 of the nuXmv model checker along with the Windows 10 Professional OS running on a i3 processor, 2.93GHz(4 CPUs), with 8 GB memory for our experiments. In the experimental setup of the MSMA all the synchronous and asynchronous registers and NCL adders are of 2 bits. For simplicity we have merged the delay of completion detection block with its respective asynchronous register and rise and fall times with its respective synchronous registers.

A. Formal Verification

We have done the functional verification of the HT detection MSMA setup using the nuXmv's bounded model checking (BMC) support for real numbers. The malicious intrusions can affect the timing and functional characteristics of the systems. Since this pipeline architecture is based on a NCL pipeline therefore, it is highly probable that some bits of the system will go into the undefined state for multi-rail logic due to malicious intrusion. Keeping this into consideration, in this paper, we envision that an intruder may leverage either an undefined behavior of MSMA or it will target the non-idealities in its communication. Therefore, we formulated the following properties to validate our concept.

1) Erroneous (undefined) Behavior of MSMA

This pipeline architecture is based on the NULL Conventional Logic (NCL) based asynchronous pipeline, which is defined for multi-rail logic. Multi-rail logic is undefined when more than one signal is high and thus this pipeline architecture also becomes undefined. For dual rail, this property can be written as:

```
G!(Data0 = 1)&(Data1 = 1))
```

Where `Data0` and `Data1` are the two rails of the same data bit. This property states that in dual rails, `rail1` and `rail2`, cannot be 1 simultaneously, at any state.

2) Data Communication of MSMA

In the communication protocol of MSMA, two stages interact through request and acknowledgement signals. In Figure 1, the request and acknowledgement is based on the Ki and Ko signals. This pipeline works in two different states: DATA state and NULL state. DATA state represents the DATA0 (0, 1) and DATA1 (1, 0) while NULL represents the NULL (0, 0). Thus, this behavior can be represented as the following properties:

Reset Property: After each successful data communication, data rails must reset before receiving the next data.

```
G((((Data0 = 1) & (Data1 = 0)) || ((Data0 = 0) &
(Data1 = 1))) U (K0 = 0) -> X(((Data0 = 0) & (Data1
= 0))))
```

Data Receiving Property: Whenever the receiver is ready then output goes from NULL to DATA

```
G((Data0 = 0 & Data1 = 0 ) U (Ki = 1)-> X(((Data0 =
1) & (Data1 = 0))|((Data0 = 0)&(Data1 = 1))))
```

Where `Data0`, `Data1` and `Ko` represent the two rails of the same data bit and acknowledgement signals, respectively. The reset property states that system will remain in the DATA state until it receives the NULL request and the opposite is stated in the data communication property.

3) Deadlock

A deadlock state in a system leads to an undesired cyclic behavior. In case of MSMA, a deadlock may occur if the system is in the requesting (waiting for response) state forever.

```
G(Ki = 1 -> F(Ki = 0))
```

OR

```
G(Ki = 0 -> F(Ki = 1))
```

Where `Ki`, i.e., the request signal. Thus, this property states that whenever the system requests for DATA (Ki = 1) then it receives it finally and asserts the NULL (Ki = 0).

B. Timing Verification

Equations (1) and (2) shows the minimum delay constraints of the synchronous, asynchronous and MSMA pipelines. We have extracted the following properties based on minimum delay constraints:

Asynchronous Request Time: From these constraints it can be depicted that the time between any two consecutive requests must be greater than the T_{ASYNCH}.

```
G(Tasynch >= areg0.delay + add0.delay + areg2.delay)
```

MSMA Request Time: The time between any two consecutive requests of asynchronous pipeline must be greater than $\{T - T_{RISE} - T_{FALL}\}$ and less than $\{T - T_{RISE} - T_{FALL}\}$.

```
G(TMSMA >= sreg0.delay + sr2dr.delay +
areg0.delay + add0.delay + areg2.delay+
dr2sr.delay + sreg1.delay)
```

C. Discussion

We have checked the properties for the experimental setup shown in Figure 3, when there was no intrusions then the MSMA model satisfies all the functional and timing properties. However, when we introduced the malicious behavior (anticipated hardware trojan) in the MSMA model then nuXmv gives the counter examples for both the functional and timing properties. We have developed multiple scenarios to intrude the MSMA in order to identify the potential attack path, which can be divided into functional and timing characteristics. All the scenarios with the resultant behavior of the properties are given in Table 1. Results in this table represent that for a particular intrusion scenario which functional and timing properties of the system hold. As an example, last row of Table 1 shows that if any of the synchronous registers of MSMA is intruded then that intrusion can only affect the timing characteristics of the system.

We have used multiple functional intrusions by intruding one block of MSMA at a time and the other scenario is used

when all the blocks are intruded. Figure 4 shows that when the synchronous registers are intruded then it does not affect any functional properties. Since the single to dual rail and dual to single rail blocks are working properly, therefore the intermediate stages do not receive any undefined multi rail inputs. Similarly, in case of dual rail to single rail, all functional properties remain unaffected because all the functional properties are defined on undefined behavior of multi-rail signals. However, intrusions in the asynchronous registers and single rail to dual rail converters provides the counter examples, and the depicted behavioral path of intrusion in asynchronous registers. Figure 4 also shows that when the asynchronous registers are intruded then in some cases the system remains in the waiting sate of asynchronous communication. We have also introduced timing intrusions in MSMA and shown that whenever any component of MSMA is intruded in such a way that it effects it timing characteristics, it always defers the timing properties. Since the timing characteristics are based on the delay constraints therefore, it is highly probable that this malicious behavior is due to environmental changes or process variations. Therefore, the timing properties are modified to compensate this behavior.

```
G(Tasynch - uncertainty = areg0.delay + add0.delay +
                areg2.delay)
```

```
G(TMSMA - uncertainty = sreg0.delay + sr2dr.delay
  + areg0.delay + add0.delay + areg2.delay +
          dr2sr.delay + sreg1.delay )
```

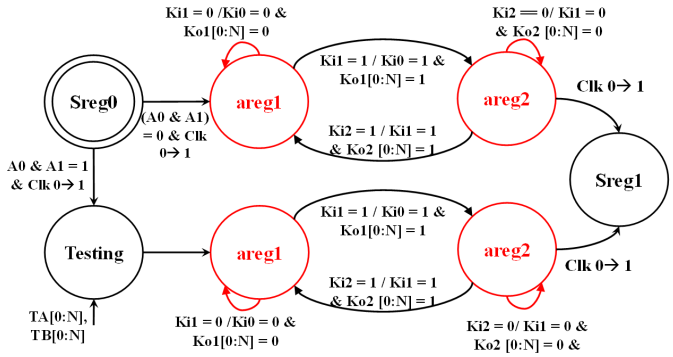

Figure 4. Potential Attack Path when Asynchronous Registers are intruded

TABLE 1: FUNCTIONAL AND TIMING ANALYSIS

Properties		Intruded Components			
		sreg(0,1,2)	*areg(0-5)*	*sr2dr(0-3)*	*dr2sr(0-1)*
Functional	Erroneous	✔	✘	✘	✔
	Reset	✔	✘	✘	✔
	Data	✔	✘	✘	✔
Timing	Asynch. time	✘	✘	✘	✘
	MSMA Time	✘	✘	✘	✘

V. CONCLUSION

We proposed a framework to detect hardware trojan in the presence of malicious intrusion. Unlike traditional implementation of formal verification to detect hardware trojan, this paper demonstrates modeling of the anticipated hardware trojan in the MSMA pipeline. The formal nature of the proposed analysis guarantees the soundness and completeness of the analysis results. In future, we intend to explore these potential attack paths to generalize the methodology for different sets of anticipated hardware trojan.

REFERENCES

[1] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Des. Test Comput.*, vol. 27, no. 1, pp. 10–25, 2010.

[2] G. Di Natale, S. Dupuis, and B. Rouzeyre, "Is Side-Channel Analysis really reliable for detecting Hardware Trojans?," *Conf. Des. Circuits Integr. Syst.*, pp. 238–242, 2012.

[3] X. Zhang, K. Xiao, M. Tehranipoor, J. Rajendran, and R. Karri, "A study on the effectiveness of Trojan detection techniques using a red team blue team approach," *Proc. IEEE VLSI Test Symp.*, pp. 8–10, 2013.

[4] F. Koushanfar and A. Mirhoseini, "A Unified Framework for Multimodal Submodular," *VLSI test Symp.*, vol. 6, no. 1, pp. 162–174, 2011.

[5] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer (Long. Beach. Calif).*, vol. 43, no. 10, pp. 39–46, 2010.

[6] M. Banga and M. S. Hsiao, "A region based approach for the identification of hardware Trojans," *2008 IEEE Int. Work. Hardware-Oriented Secur. Trust*, pp. 40–47, 2008.

[7] M. Banga and M. S. Hsiao, "A novel sustained vector technique for the detection of hardware trojans," *Proc. 22nd Int. Conf. VLSI Des. - Held Jointly with 7th Int. Conf. Embed. Syst.*, pp. 327–332, 2009.

[8] F. K. Lodhi, S. R. Hasan, O. Hasan, and F. Awwad, "Hardware Trojan Detection in Soft Error Tolerant Macro Synchronous Micro Asynchronous (MSMA) Pipeline," in *International Midwest Symposium on Circuits and Systems (MWSCAS'14)*, 2014.

[9] F. K. Lodhi, S. R. Hasan, O. Hasan, and F. Awwad, "Low Power Soft Error Tolerant Macro Synchronous Micro Asynchronous (MSMA)Pipeline," *IEEE Comput. Soc. Annu. Symp. VLSI*, pp. 601–606, 2014.

[10] X. Guo, R. G. Dutta, Y. Jin, F. Farahmandi, and P. Mishra, "Pre-Silicon Security Verification and Validation□: A Formal Perspective," *Proc. 52Nd Annu. Des. Autom. Conf.*, pp. 145:1–145:6, 2015.

[11] X. Zhang and M. Tehranipoor, "Case study: Detecting hardware Trojans in third-party digital IP cores," *2011 IEEE Int. Symp. Hardware-Oriented Secur. Trust. HOST 2011*, pp. 67–70, 2011.

[12] Y. Jin, "Design-for-Security vs. Design-for-Testability: A Case Study on DFT Chain in Cryptographic Circuits," *VLSI (ISVLSI), 2014 IEEE Computer Society Annual Symposium on*. pp. 19–24, 2014.

[13] M. Rathmair and F. Schupfer, "Hardware Trojan detection by specifying malicious circuit properties," *Electronics Information and Emergency Communication (ICEIEC), 2013 IEEE 4th International Conference on*. pp. 317–320, 2013.

[14] F. K. Lodhi, O. Hasan, S. R. Hasan, and F. Awwad, "Modified null convention logic pipeline to detect soft errors in both null and data phases," in *Midwest Symposium on Circuits and Systems*, 2012.

[15] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuXmv Symbolic Model Checker," in *Computer Aided Verification SE - 22*, vol. 8559, A. Biere and R. Bloem, Eds. Springer International Publishing, 2014, pp. 334–342.

[16] Fondazione Bruno Kessler and DISI-University of Trento., "The MathSAT 5 SMT Solver," 2015. [Online]. Available: http://mathsat.fbk.eu/. [Accessed: 24-Aug-2015].

[17] F. K. Lodhi, S. R. Hasan, O. Hasan and F. Awwad, "Formal Analysis of Macro Synchronous Micro Asychronous Pipeline for Hardware Trojan Detection," 2015. [Online]. Available: http://save.seecs.nust.edu.pk/projects/FA-MSMA/.