THE ANNALS OF "DUNĂREA DE JOS" UNIVERSITY OF GALATI
FASCICLE III, 2021, VOL. 44, NO. 1, ISSN 2344-4738, ISSN-l 1221-454X
ELECTROTECHNICS, ELECTRONICS, AUTOMATIC CONTROL, INFORMATICS
Article DOI: https://doi.org/10.35219/eeaci.2021.1.04

TRIGGERING DETECTION TECHNIQUE OF THE HARDWARE TROJANS IN THE COMBINATIONAL LOGIC SYSTEMS

Grigore Mihai Timis, Alexandru Valachi

Technical University "Gh.Asachi"Iasi, Faculty of Automatic Control and Computer Engineering
(e-mail: mtimis@tuiasi.ro, avalachi@tuiasi.ro)

Abstract: This paper presents a new triggering detection method of the Hardware Trojans in the combinational logic systems, using static hazard analysis. A malicious entity can introduce a Hardware Trojan (HT) into a design in order to denial of service, destroy or disable the system. Moreover, it could leak the confidential information and the secret keys before altered them. The Hardware Trojan (HT) threats should be analyzed with maximum importance through the entire lifecycle of the integrated circuit (ICs). A hardware protection against the detected harmful logic should also be implemented.

Keywords: Hardware Trojan (HT), Hardware Trojan triggering detection, Hardware Trojan attacks, Static hazard analysis, Combinational digital systems.

1. INTRODUCTION

Hardware Trojans (HT) are modifications of the original circuit, inserted by an unintended entity in order to exploit and to gain access to data or software running on chips, [1].

Globalization in the integrated circuits industry decreases the control of the System on Chip(SoC) design engineers on the fabricated IC chips. Adding of the 3rd party Intellectual Properties (IPs), design tools, outsourcing fabrication, leads to lower costs and meet the time to market targets, [15]. Thus, using a 3rd IP cores known as black boxes, instead of building these blocks from the scratch, they can contain Hardware Trojans (HT) which could generate potentially malfunction of the SoC normal functionality. The type of the malfunctions can be denial of service attack (DoS) or cause privacy leakage. These Trojans must be detected in the pre-silicon phase, otherwise it can infect millions of ICs through a Trojan affected IP core, [9], [12].

Based of the Intellectual Properties (IP) reasons, the IP core vendors do not offer the RTL source code of the IP core. Thus, the digital IP cores are generally offered in the netlist format which consists in the common digital logic gates and memory elements.

Even if the RTL source code of the IP core is available, it could be not so feasible in case of large IP's cores to manually inspect the source code for the Hardware Trojans.

These vulnerabilities have raised concerns regarding the possible threats to the financial infrastructures, military systems, transportation security etc. [10], [18].

A malicious entity can introduce a Hardware Trojan into a design in order to denial of service, destroy, disable the system. Moreover, it could leak the confidential information and the secret keys before altered them, [9], [14].

Trojans can be implemented as hardware changes to ASICs, microprocessors, digital signal processors (DSPs), microcontrollers, different kind of processors. They can be also implemented as firmware modifications – FPGA bitstreams. According to [10], an IC fabrication process contains three major steps: designs, fabrications and manufacturing. Thus, also the fabrication and manufacturing steps might be considered un-trusted since an attacker can substitute Trojan ICs for genuine ones.

In general, the hardware based security techniques modify the hardware in order to prevent possible attacks and to protect IP blocks or secret keys. It's considered that an unintended person-attacker, will alter the design before or during the fabrication. As specific literature relates [10], [12], [13], [16], [17], third Hardware Trojans (HT) circuits are usually activated if a specific or couple of specific conditions are meet: eg. sensing a specific design signal such as temperatures, power or an output value of a specific logic is activated.

Hardware Trojans (HT) detection is still a new research area, but it has accounted a significant attention in the past decade, [2], [5], [6]. Hardware Trojans (HT) can be triggered upon some rare conditions and altering the design functionality, shown in Fig.1. Hardware Trojans (HT) are typical for changing the parametric characteristics of the design, for example, by degrading the performance, changing power characteristics or introducing reliability problems in the chip. This will influence power and delay characteristics of the gates in the affected chip, [10], [15].

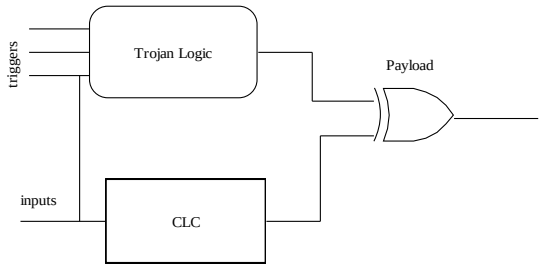

Fig. 1. Combinational HT triggering

By using the chip in an environment with normal temperatures, it will have a time propagation delay value, but this will be changed in a different one.

For example, the triggering condition of a Hardware Trojans (HT) could be the cases when inputs have the following wrong values: $x = \bar{x} = 1$ or $x = \bar{x} = 0$ due of propagations delays through digital logic.

2. INSERTION METHODS OF HARDWARE TROJAN (HT)

According to [1], Hardware Trojan (HT) insertions can be classified into three categories:

- internal trigger;
- external trigger;
- HTH storage;
- HTH driver;

Triggers can be associated with an external/internal event or a predefined value of a bus/signal. After the trigger is activated, the action to be taken can be stored in a sequential circuit or in a memory. The implementations of the trigger are executed by the driver. In [9], considering several attacks, it's designed and implemented the Illinois Malicious Processor with a modified CPU. Thus, a malware firmware was executed using stealthy execution. The attack was evaluated using a FPGA evaluation development board by changing the VHDL code of a Sparc V8 processor that includes a MMU (Memory Management Unit).

The additional timing overhead compared with the original version is approx. 12%, while in the logic is about 1%, [11], [15].

Three following potential attacks were implemented: a privilege escalation attack, a log-in backdoor in shadow mode and a password stealing service which is sending to the attacker. The malicious circuit is inserted into the design using the mechanism for actively IC controlling (e.g. IP core).

3. TRIGGERING DETECTION TECHNIQUE OF THE HARDWARE TROJANS USING THE STATIC HAZARD ANALYSIS METHOD

Based on the delay for the combinational logic block [8], due to the propagation delay, there will be an uncertainty period where $x \cdot \bar{x} = 1$ (noted with 'b'), respectively $x \cdot \bar{x} = 0$ (noted with 'd'), as shown in Fig.2:

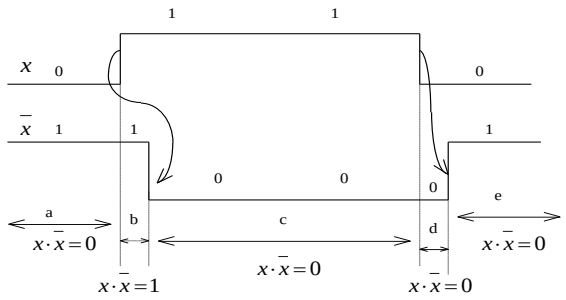

Fig. 2. Propagation delay with uncertainty values

There can be observed that on state "b" we have the following relation: $x \cdot \bar{x} = 1$ and on state "d", $x \cdot \bar{x} = 0$. These cases could be a trigger condition for Hardware Trojan (HT) and it could lead to malfunction and circuit leakage information.

The novelty proposed idea of this research paper is to detect the triggering condition of Hardware Trojan (HT) by using the static hazard analysis.

Let us consider a combinational digital system described by the following prime implicants:

$$f(x_4x_3x_2x_1x_0) = R_1(2,4,8,12,13,15,18,21,26,30) + \\ + R\_(0,5,10,16,23,24,26,29,31) \quad (1)$$

The implementation with logic gates is like in Fig.3:

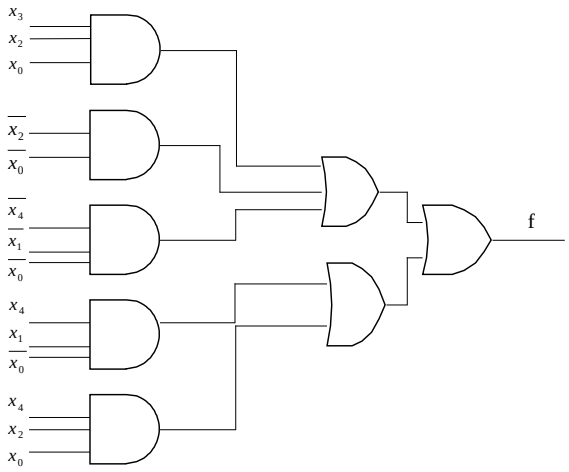

Fig. 3. Logic Design Synthesis

The prime implicants for the output logic function are shown in (2):

$$\begin{aligned}
A &= \overline{x_4}\,\overline{x_1}\,\overline{x_0}(0,4,8,12)\\
B &= \overline{x_4}\,x_2\,\overline{x_1}(4,5,12,13)\\
C &= x_2\,\overline{x_1}\,x_0(5,13,21,29)\\
D &= x_4 x_1\,\overline{x_0}(18,22,26,30)\\
E &= x_3 x_2 x_0(13,15,29,31)\\
F &= x_4 x_2 x_0(21,23,29,31)\\
G &= x_4 x_2 x_1(22,23,30,31)\\
H &= \overline{x_2}\,\overline{x_0}(0,2,8,10,16,18,24,26)
\end{aligned} \tag{2}$$

A minimal solution is like in (3):

$$f(x_4x_3x_2x_1x_0) = E + H + A + D + F = x_3x_2x_0 + \overline{x_2}\,\overline{x_0} + \\ + \overline{x_4}\,\overline{x_1}\,\overline{x_0} + x_4x_1\overline{x_0} + x_4x_2x_0 \tag{3}$$

The static hazard analysis [8], based on the x_0 input, (4):

$$y = f(x_4x_3x_2x_1x_0) = (x_3x_2 + x_4x_2)\cdot x_0 + (\overline{x_2} + \overline{x_4}\,\overline{x_1} + \\ + x_4x_1)\cdot\overline{x_0} \tag{4}$$

According to (1), the static hazard may exist if, (5):

$$\begin{cases} x_3x_2 + x_4x_2 = 1 \\ \overline{x_2} + \overline{x_4}\,\overline{x_1} + x_4x_1 = 1 \end{cases} \Rightarrow \begin{cases} x_2(x_3 + x_4) = 1 \\ \overline{x_2} + \overline{x_4 \oplus x_1} = 1 \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} x_2 = 1, x_3 + x_4 = 1 \\ \overline{x_4 \oplus x_1} = 1, x_4 \oplus x_1 = 0 \end{cases} \tag{5}$$

Based on the (5) relations, the solutions are shown in (6):

$$\begin{aligned}
&x_4 = 0, x_3 = 1, x_2 = 1, x_1 = 0\\
&x_4 = 1, x_3 = 0, x_2 = 1, x_1 = 1\\
&x_4 = 1, x_3 = 1, x_2 = 1, x_1 = 1
\end{aligned} \tag{6}$$

According to the relations (5) and (6), the combinational logic system may present static hazard on the commutation of the x_0 variable, as described in (7):

$$\begin{aligned}
&x_4x_3x_2x_1x_0 = 01100 \Leftrightarrow 01101, (12 \Leftrightarrow 13)\\
&x_4x_3x_2x_1x_0 = 10110 \Leftrightarrow 10111, (22 \Leftrightarrow 23)\\
&x_4x_3x_2x_1x_0 = 11110 \Leftrightarrow 11111, (30 \Leftrightarrow 31)
\end{aligned} \tag{7}$$

In order to eliminate the static hazard and also the possibility to be exploited by the Hardware Trojan (HT), the logic function from (1) should be updated with the following prime terms implicants which are equals to "1" (e.g. B and G), like in (8).

$$\begin{aligned}
y &= E + H + A + D + F + B + G = x_3x_2x_0 + \overline{x_2}\,\overline{x_0} + \\
&+ \overline{x_4}\,\overline{x_1}\,\overline{x_0} + x_4x_1\overline{x_0} + x_4x_2x_0 + \overline{x_4}\,x_2\,\overline{x_1} + x_4x_2x_1 = \\
&= (x_3x_2 + x_4x_2)\cdot x_0 + (\overline{x_2} + \overline{x_4}\,\overline{x_1} + x_4x_1)\cdot\overline{x_0} + \\
&+ \overline{x_4}\,x_2\,\overline{x_1} + x_4x_2x_1 = \\
&= (x_3x_2 + x_4x_2)\cdot x_0 + (\overline{x_2} + \overline{x_4}\,\overline{x_1} + x_4x_1)\cdot \\
&\cdot\overline{x_0} + x_2(\overline{x_4 \oplus x_1})
\end{aligned} \tag{8}$$

Based on equation (8), a possible synthesis of the combinational digital system with detection of the Hardware Trojan triggering condition $x_0 \cdot \overline{x_0} = 1$ or $x_0 + \overline{x_0} = 0$ is presented in Fig.4:

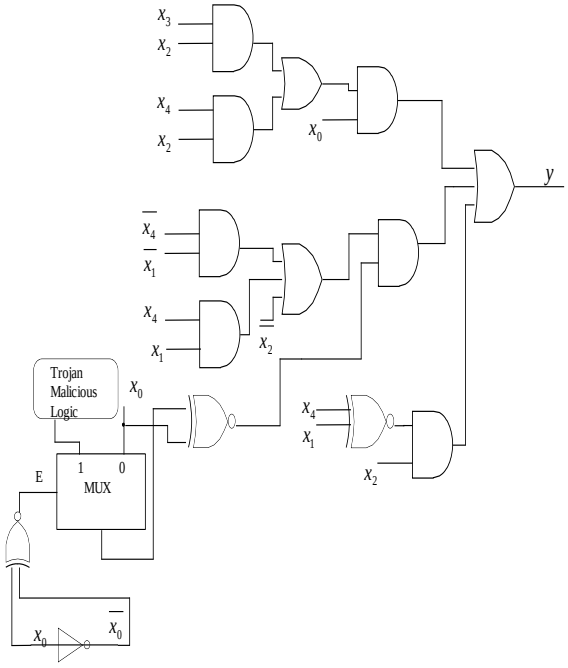

Fig. 4. Hardware Trojan triggering detection synthesis

Based on the proposed method, the equation (8) represents the free static hazard function (regarding the x_0 commutation from 0 to 1). Since the combinational digital system is free of static hazard, the malicious Hardware Trojan logic will not affect the digital system. The logic truth table (Table I) for the cases when the Hardware Trojan could be triggered are presented next, described by logic equation (9):

TABLE I. TROJAN DETECTION CASES

$x_0\,\overline{x_0}$	Trojan triggering condition
00	1
01	0
10	0
11	1

$$T = \overline{x_0 \oplus \overline{x_0}} \tag{9}$$

From (9) it can be observed that Hardware Trojan will be triggered only when the following conditions are met: $x_0 = \overline{x_0} = 1$ or $x_0 = \overline{x_0} = 0$. But since the implementation from Fig.6 is free of static hazard in relation with x_0, the hardware trojan malicious logic will not affect the system.

We demonstrated that although the Hardware Trojan malicious design had the output signal as 1 - Trojan malicious logic was activated due to the propagation delay of $x_0 \to \overline{x_0}$ and $x_0 = \overline{x_0} = 0$ or $x_0 = \overline{x_0} = 1$ and since the logic was static hazard free, HT malicious logic triggering was successfully detected, thus it would have no effect to the combinational digital system.

We have to mention that our new Hardware Trojan detection method can be successfully applied on any combinational digital system.

The proposed algorithm can be used for all inputs (x_1, x_2, x_3, x_4) where static hazard analysis must be applied, thus the final combinational system will successfully detect the triggering of Hardware Trojan malicious logic.

For the study of the static hazard based on x_1 input, the output f equation is, (10):

$$\begin{aligned} y = f(x_4, x_3, x_2, x_1, x_0) = (x_4 x_0 + x_4 \overline{x_0}) x_1 + \\ + (\overline{x_4}\,\overline{x_0} + \overline{x_4} x_2)\overline{x_1} + x_3 x_2 x_0 + \overline{x_2}\,\overline{x_0} + x_4 x_2 x_0 \end{aligned} \tag{10}$$

We make the following assignments, (11):

$$\begin{aligned} A_1 &= x_4(x_2 + \overline{x_0}) \\ B_1 &= \overline{x_4}(x_2 + \overline{x_0}) \\ C_1 &= x_3 x_2 x_0 + \overline{x_2}\,\overline{x_0} + x_4 x_2 x_0 \end{aligned} \tag{11}$$

From (11), it can be observed that because $A_1 \cdot B_1 = 0$ so it's obvious that $C_1 \geq A_1 \cdot B_1$, thus the function won't have static hazard in relation with x_1 input. Further, following the described algorithm, it can be verified that the function *f* doesn't have static hazard in relations with all the remaining inputs x_2, x_3, x_4. Based on this, we can say that the combinational logic synthesis from Fig.4 is completely free of static hazard and thus the Hardware Trojan malfunction logic will not meet the triggering condition and finally will not affect the system.

4. COMPARISON WITH EXISTING HT DETECTION TECHNIQUES

State of the art Hardware Trojan detection techniques are typically based on verification trust with the assumption that a Hardware Trojans malicious logic always remains inactive in the circuit to pass the functional verification. Due to functional verification constraints, the entire circuit cannot be analyzed and activated in optimal time, thus large portions of the design are unused and we can consider them as potential HT malicious logic.

The most used Hardware Trojans detection techniques are:

- **VeriTrust**: described in [3], it detects HTHs by identifying the inputs in the combinational logic that seems to be redundant for the normal functionality of the output under the non-trigger condition. For detection of the redundant inputs, it records the activation history of the inputs in the form of SumOfProducts (SOP) and ProductsOfSums(POS) in order to find the redundant inputs. But, because VeriTrust can see several unactivated SOP, POS logics, it can be potentially reporting false positive cases.

- **FANCI**: described in [4], it applies the boolean function analysis to flag suspicious wires in a design which doesn't have a strong in-out dependency. It's calculated a control value which holds the impact of changing inputs on the outputs. This method is applied on each input of the system. This is a probabilistic method where a threshold is calculated with some heuristic in order to reach a balance between security and the false positive rates.

- **DeTrust**: described in [7], presents a new way to design HTS which cannot be detected by either FANCI or VeriTrust. Thus, the DeTrust algorithm designs a new HTHs, the circuits of which are mixed with the normal design and then distributed over multiple stages such as FANCI/VeriTrust. Also, DeTrust splits the original combinational logic less stealthy into two sequential stages by the inserted flip-flops.

According to [2], due to comparison of computational complexities of different techniques for detection of HTH Trojans, the computational complexity is $\Omega(m2^m)$ because most of them monitor the history of each entry in the truth table, instead of the SOP and POS Boolean functions terms. This leads to an exponential computational complexity logic as the truth table has 2^m entries in total.

Based on the above methods, it has been proven that our new static hazard method analysis algorithm for triggering detection of the Hardware Trojans is much robust, faster and it is using less logic.

5. CONCLUSIONS AND FUTURE WORK

The Hardware Trojan issue has become a more and more sensitive security concern for a design service nowadays, thus their detection becomes a very challenging problem.

Based on the diversity of the Hardware Trojans types, one unique method cannot be reliable for all of them, that is why a 100% detectability seems to be impossible.

Hardware Trojan is designed to avoid detection since they run in stealth mode. There are widen methods regarding the triggering modes which avoid the IC's testing procedures (e.g. different combinations of the primary inputs).

As a first conclusion, the Hardware Trojan logic design can have any complexity. Its task is to inject into digital system bad values, thus exploit the case when $x = \bar{x} = 1$ or $x = \bar{x} = 0$.

Based on the elimination of the static hazard technique, we've proved that the combinational digital system can't be exploited by a Hardware Trojan which couldn't speculate the delay propagations in the digital system cases ($x = \bar{x} = 1$ or $x = \bar{x} = 0$).

Since the digital logic system is static hazard free regarding the input x_0 variable, the Hardware Trojan is detected and it will not affect the functionality of the digital system.

The same idea can be also applied to any combinational design, no mater its complexity.

A future research will be designing a digital prototype in order to simulate different trojans triggering and detection scenarios.

REFERENCES

[1] Mohammad Tehranipoor and Farinaz Koushanfar, "A survey of hardware trojan taxonomy and detection", IEEE Design & Test of Computer, 27:10-25, 2010.

[2] Syed Kamran Haider, Chenglu Jin, Masab Ahmad, Devu Shila, Omer Khan, Marten van Dijk, "Advancing the State-of-the-Art in Hardware Trojans Detection", IEEE Transactions on Dependable and Secure Computing (Volume: 16, Issue: 1, Jan.-Feb. 1 2019).

[3] Zie Zhang; Feng Yuan; Lingxiao Wei; Zelong Sun; Qiang Xu "VeriTrust: Verification for hardware trust" 50th ACM/EDAC/IEEE Design Automation Conference (DAC), 29 May-7 June 2013, IEEE.

[4] A. Waksman, Matthew Suozzo, S. Sethumadhavan, "FANCI: identification of stealthy malicious logic using boolean functional analysis", Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, Published 2013, IEEE.

[5] Y. Xie and A. Srivastava, "Mitigating SAT Attack on Logic Locking", In Conference on Cryptographic Hardware and Embedded Systems (CHES'16), pp. 127–146, 2016.

[6] M. Palanichamy, P.S. Ba, S. Dupuis, "Duplication-based Concurrent Detection of Hardware Trojans in Integrated Circuits", In Workshop on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE'16), 2016.

[7] Jie Zhang, Feng Yuan, Qiang Xu, "DeTrust: Defeating Hardware Trust Verification with Stealthy Implicitly-Triggered Hardware Trojans", CCS '14: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications SecurityNovember 2014 Pages 153–166.

[8] Alexandru Valachi, Bogdan I. Aignătoaiei, Mihai G. Timiș "The comparative study of two analytical methods for detection and elimination of the static hazard in Combinational Logic Circuits", 15th International Conference on System Theory, Control and Computing, 2011, INSPEC Accession Number: 12390388, publisher IEEE.

[9] 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, ser. LEET'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 5:1–5:8.

[10] R. Karri, J. Rajendran, K. Rosenfeld and M. Tehranipoor, "Trustworthy Hardware: Identifying and Classifying Hardware Trojans", In IEEE Computer, pp. 39–46, 2010.

[11] Y. Jin, N. Kupp and Y. Makris, "Experiences in Hardware Trojan Design and Implementation", Proc. IEEE Int'l Workshop Hardware-Oriented Security and Trust (HOST 09), pp. 50-57, 2009.

[12] W. Hu, L. Zhang, A. Ardeshiricham, J. Blackstone, B. Hou, Y. Tai, et al., "Why you should care about don't cares: Exploiting internal don't care conditions for hardware Trojans", pp. 707-713, 2017.

[13] Ayush Jain; Ziqi Zhou; Ujjwal Guin, "Survey of Recent Developments for Hardware Trojan Detection", 2021 IEEE International Symposium on Circuits and Systems (ISCAS), 22-28 May 2021, DOI: 10.1109/ISCAS51556.2021.9401143, IEEE.

[14] I. Exurville, L. Zussa, J.B. Rigaud and B. Robisson, "Resilient Hardware Trojans Detection based on Path Delay Measurement", In International Symposium on Hardware-Oriented Security and Trust (HOST'15), pp. 151–156, 2015.

[15] M Tehranipoor, H. Salmani, X. Zhang, X. Wang, R. Karri, J. Rajendran and K. Rosenfeld, "Trustworthy Hardware: Trojan Detection and Design-for-Trust Challenges", In IEEE Computer, pp. 66–74, 2011.

[16] Ovidiu Ursaru, Cristian Aghion, Mihai Lucanu, Liviu Tigaeru, "Pulse width Modulation Command Systems Used for the Optimization of Three Phase Inverters", Advances in Electrical and Computer Engineering Journal. Suceava, Romania, 2009, pag.22-27.

[17] N. Jacob, D. Merli, J. Heiszl and G. Sigl, "Hardware Trojans: current challenges and approaches", in IET Computers & Digital Techniques, 8(6):264– 273, 2014.

[18] ML Flottes, S Dupuis, PS Ba and abd B Rouzeyre, "On the limitations of logic testing for detecting Hardware Trojans Horses", International Conference on Design & Technology of Integrated Systems in Nanoscale Era IEEE,pp.1-5,2015.