SCARE: Side-Channel Analysis based Reverse Engineering for Post-Silicon Validation

Xinmu Wang, Seetharam Narasimhan, Aswin Krishna, and Swarup Bhunia
Department of Electrical Engineering and Computer Science
Case Western Reserve University, Cleveland, OH 44106, USA
Email: xxw58@case.edu

***Abstract*—Reverse Engineering (RE) has been historically considered as a powerful approach to understand electronic hardware in order to gain competitive intelligence or accomplish piracy. In recent years, it has also been looked at as a way to authenticate hardware intellectual properties in the court of law. In this paper, we propose a beneficial role of RE in post-silicon validation of integrated circuits (IC) with respect to IC functionality, reliability and integrity. Unlike traditional destructive RE approaches, we propose a fast non-destructive side-channel analysis approach that can hierarchically extract structural information from an IC through its transient current signature. Such a top-down side-channel analysis approach is capable of reliably identifying pipeline stages and functional blocks. It is also suitable to distinguish sequential elements from combinational gates. For extraction of random logic structures (e.g. control blocks and finite state machines) we combine side-channel analysis with logic testing based Boolean function extraction. The proposed approach is amenable to automation, scalable, and can be applied as part of post-silicon validation process to verify that each IC implements exclusively the functionality described in the specification and is free from malicious modification or Trojan attacks. Simulation results on a pipelined DLX processor demonstrate the effectiveness of the proposed approach.**

***Index Terms*—Reverse engineering, side-channel analysis, logic testing, self-referencing.**

I. Introduction

Since the Cold War era, reverse engineering (RE) has been considered as a powerful tool to analyze electronic hardware for gaining competitive intelligence or for commercial piracy. Although regarded illegal in common belief, in most countries around the globe, RE is allowed for analysis, evaluation or teaching purposes [1]. In military and many mission-critical applications, RE can provide enabling technology for post-silicon validation of integrity and reliability of complex chips, which are designed and fabricated in untrusted environments [2]. For semiconductor industry, RE has become an attractive (and often, the only) option for claiming hardware Intellectual Property (IP) rights in the court of law. This requirement has led to the formation of a number of industrial entities, e.g. ChipWorks [3], dedicated to reverse engineering and the analysis of microchips and electronic systems.

In recent years, IC trust has emerged as a critical concern in semiconductor industry. Dictated by economic reasons, modern semiconductor design and fabrication flow involves third party IP cores, outsourced design and test services, as well as CAD tools supplied by third-party vendors. Lack of control on the design and fabrication steps greatly increases the vulnerability to malicious design modifications, called *hardware Trojan* attacks [4]. An attacker can mount these Trojan attacks to cause malfunction during field operation or leak secret information from inside a chip. Both side-channel and logic-testing based non-invasive approaches have been proposed earlier in the context of Trojan detection [5] when golden chip instances are available. However, due to untrusted fabrication facility in most cases, golden chips, which are needed to benchmark and detect compromised chips, are hard to achieve and demand reverse-engineering.

Image recognition based structural extraction involving de-packaging and de-layering an IC has been conventionally used as a reverse engineering approach [3]. Such a method is highly expensive, time-consuming, and destroys the chip. Since the chip "validated" in this way cannot function properly anyway, it can no longer be used as the benchmark for detecting other potentially compromised chips. On the other hand, some functional RE approaches have been investigated in recent years, e.g. [6] and [7]. Yet the complexity of logic testing approaches increases dramatically with the circuit size, especially in absence of full-scan testability in the design. More importantly, logic testing based approaches rely on random test vector generation, which can fail to detect extraneous undesired functions reliably if the functions are activated and observed only under rare conditions [4]. This implies that logic testing approaches aim to identify only the Boolean functions while considering the actual structural connectivity information transparent, which itself implies potential ignorance of design-parameter-violation-Trojans.

In this paper, we propose a top-down, hierarchical unified side-channel and logic testing approach that can extract both structural and functional information from a manufactured

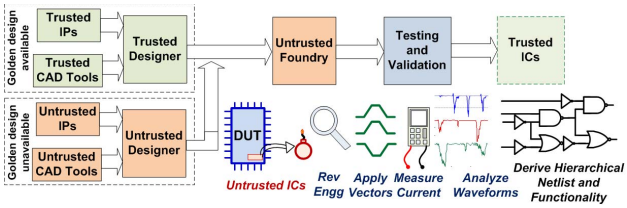


Fig. 1. Untrusted stages of the IC manufacturing flow. Steps of the proposed methodology to perform non-invasive RE and trust validation.

1063-9667/12 $26.00 © 2012 IEEE
DOI 10.1109/VLSID.2012.88

IC. The method assumes the availability of a golden design (not golden chip instance), and can be extended to scenarios without the golden design. Fig. 1 illustrates the proposed top-down approach. This approach is valuable in two contexts:

(1) For validating a golden chip instance as Trojan detection benchmark, it is a significantly more low-cost, time-efficient and reliable choice compared to image recognition and logic testing based reverse-engineering approaches.

(2) When the method is considered directly as a Trojan detection approach, it is applicable to detecting comprehensive types of Trojans with no need of a golden chip by providing circuit structural information with the resolution of a single gate. Also, by using temporal and spatial self-referencing, this approach is invulnerable to significant process noise. Comparatively, conventional logic testing and side-channel based approaches are limited by their effective Trojan ranges, lower resolution, vulnerability to environmental noise, and need of a golden chip. When extended to no-golden-design scenarios, the proposed approach can depend only on the datasheet specifications to detect malicious hardware inserted in any stage of the design and fabrication flow. The hierarchical approach is scalable to large designs.

II. Background

Malicious insertions (or *Hardware Trojans*, Fig. 2), are usually cleverly designed so as to be rarely triggered during normal operation. The reasons for the failure of logic testing based approach to detect hardware Trojans are as follows:
(1) Exhaustive enumeration is impractical for large designs, especially for sequential designs with/without scan-chains, creating chances of omitting rare events which trigger hardware Trojans. Fig. 2(a) shows an example of a combinational rarely-triggered Trojan, which can evade non-exhaustive testing.
(2) Trigger of sequential malicious insertions requires a sequence of unknown rare events, which can hardly be achieved even with exhaustive testing. An example of such sequential Trojan is given in Fig. 2(b), which cannot be triggered during one-time exhaustive enumeration. Moreover, state-elements in such sequential Trojans could use rare switching activity of internal circuit nodes as their clock signal, as illustrated in Fig. 2(c), which again lowers Trojan trigger possibility rendering them almost transparent in logic-based circuit extraction.

On the other hand, side-channel analysis based approaches [5] using transient current (IDDT), quiescent current or path delay fingerprint have been proposed for Trojan detection in untrusted ICs. The main deterrent to such approaches is the large amount of process-induced parameter variations [8] which can mask the effect of malicious circuitry on the measured side-channel parameter. To overcome this drawback, various statistical techniques have been proposed to make process-invariant self-similarities in the design get reflected in the measured side-channel parameter such as transient supply current [9]. While logic values at the primary output reflect only the Boolean function with respect to the present state and primary inputs, the current waveform contains information about relative timing of different paths in the form of glitches, which can reveal significant information about internal structure, such as number of switching gates for particular vector pairs and their connectivity. Similarly, quiescent leakage current [10] contains information about all the gates in an IC, but it is difficult to observe the effect of small Trojans on the total leakage current, hence such methods have decreasing sensitivity for large designs. Therefore we choose an IDDT based side-channel approach.

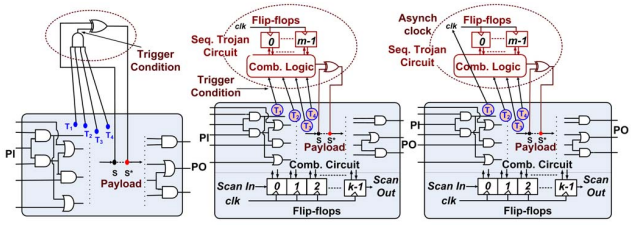


Fig. 2. Examples of different forms of malicious insertions: (a) Combinational Trojan. (b) Sequential Trojan using global clock. (c) Sequential Trojan using internal node activity as clock signal.

III. Methodology

Transient current (IDDT) signature of an IC in response to input transitions contains structural information of an IC including connectivity and dependency among blocks. However, to identify structural blocks of an IC from its current signature, two major challenges have to be addressed: (1) avoid the aliasing effect due to simultaneous switching of multiple blocks; (2) eliminate the effect of process variations and measurement noise. We adopt a novel side-channel analysis approach, referred as self-referencing, which compares an IC with itself - either spatially between two or more regions or temporally between two time instances. The idea of spatial self-referencing can be explained using Fig. 3, which shows that the self-similarity of circuit blocks can be exploited hierarchically to identify constituent logic sub-blocks in structured logic. Similarly, temporal self-similarities in current signature are used to build a transient current *signature library* containing process and technology independent current signatures for each datapath block. The overall flow of the automated reverse engineering approach is illustrated in Fig. 4. Next, we describe key steps in detail with specific examples.

From the golden structural block diagram, functional blocks are defined along with their input/output dependencies. Next, functional vector sets are generated targeting activation of specific blocks [9]. In circuits with pipeline stages, temporal self-referencing can be used to restrict the switching activity to one stage by appropriate choice of vectors. Spatial self-

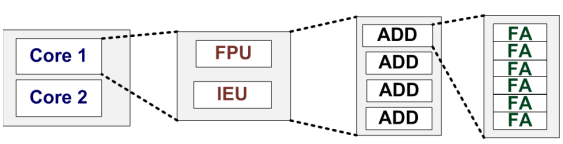


Fig. 3. Spatial self-referencing for identifying hierarchical functional blocks.

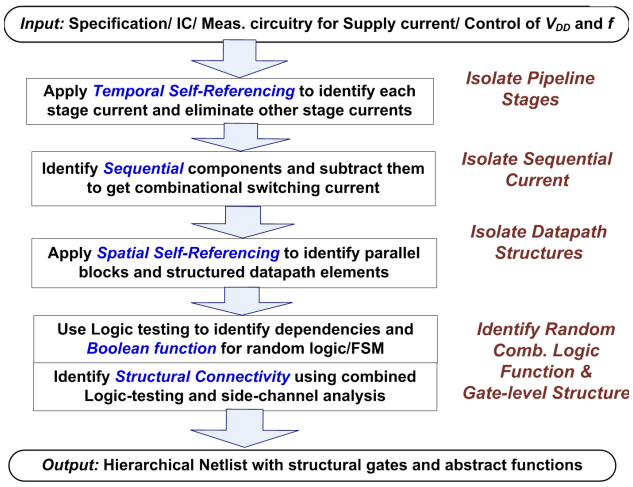


Fig. 4. Main steps of the proposed approach for IC reverse engineering.

referencing can also be used to identify parallel structural blocks and homogeneous array structures such as memory.

The next step is to isolate the sequential and combinational parts of switching current by using the correlations between the switching at the positive and negative edges of the clock. By using a slow clock, all the combinational switching can be confined to the positive half-cycle and the switching in the negative half-cycle corresponds only to the master stages of the flip-flops and clock coupling current. After the sequential current and the switching current during memory access has been subtracted from the stage current, the switching current caused by combinational circuit activity can be isolated out.

Due to their regular structures, standard datapath elements exhibit technology and process independent transient current features in response to specific input test patterns, which can be exploited to identify their specific types and implementation. A *signature library* based on relative features of transient current shapes, e.g. waveform correlation and number of observable ripples, is built after comprehensive characterization of different datapath elements and their standard implementations. One can match the measured signature with macro-elements from the library to confirm the implementation specified in the golden netlist. Signature characterization is performed with the following perspectives:

(1) *Architecture-specific signature information*: One can sensitize different paths in a circuit which relate to some particular functional behaviors, and manifest information of structural features. For example, overall topology (e.g. flattened structure or blocked structure) of an adder can be revealed by transitions involving carry propagation.

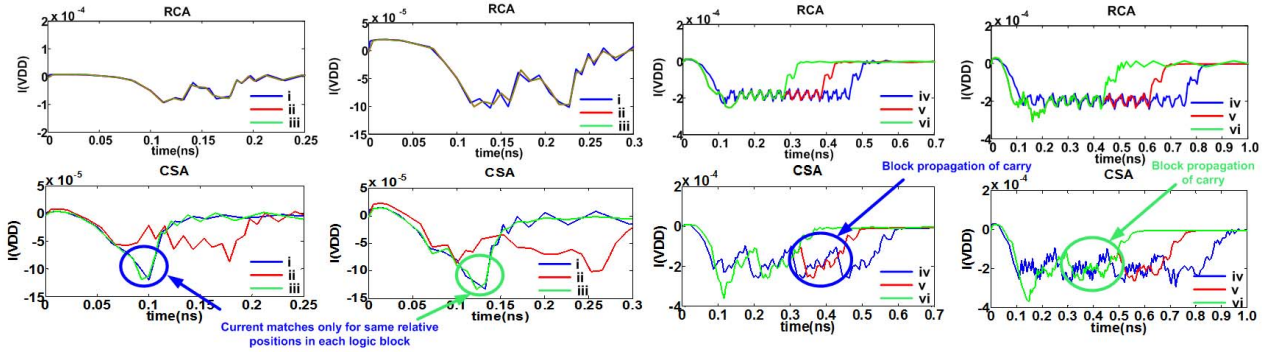
(2) *Temporal self-referencing*: Transient current signatures can be obtained by comparing switching current for different transitions that trigger the same part of the circuit.

(3) *Spatial self-referencing*: Structural symmetry causes similar transient current for different transitions, helping in detection of repeated structures at high level (e.g. parallel structures) and low level (e.g. repeated full-adders in multi-bit adder).

Adder: Fig. 5 provides an instance demonstrating all the above three perspectives. Current waveforms for two test sets containing 3 vector pairs each are obtained.

(1) **Set S1** contains vectors to perform *single bit addition without carry propagation*. In particular, three vector pairs i, ii, and iii are used to perform single-bit addition at *bit0*, *bit1* and *bit8* and the current waveforms for two types of adders are shown in Fig. 5(a) and (b) for two technology nodes. Test vectors used on the Ripple Carry Adder (RCA) give closely matching current waveforms for all three vectors, implying that RCA contains a repeated bit-wise structure. In the case of Carry Save Adder (CSA), the shape of switching current for different operations depends on the relative bit position inside its block (4 bits are grouped as a block). This can be observed in Fig. 5(a), where current waveforms match for addition in the same relative positions inside each block. Besides, from Fig. 5(a) and (b) we can see the invariance in shape in terms of relative features across different technology nodes.

(2) **Set S2** consists of vectors to *activate carry propagation paths of different lengths* to explore self-similarity inside the adder architectures, by propagating the carry from the *carry-in bit*, *bit3* and *bit7*. In the top sub-figure of Fig. 5(c) we



(a) switching current of 2 types of adders for Test Set 1 (45nm)
(b) switching current of 2 types of adders for Test Set 1 (65nm)
(c) switching current of 2 types of adders for Test Set 2 (45nm)
(d) switching current of 2 types of adders for Test Set 2 (65nm)

Fig. 5. Current signatures of RCA and CSA adders for 45nm and 65nm technology nodes used for self-referencing based reverse engineering.

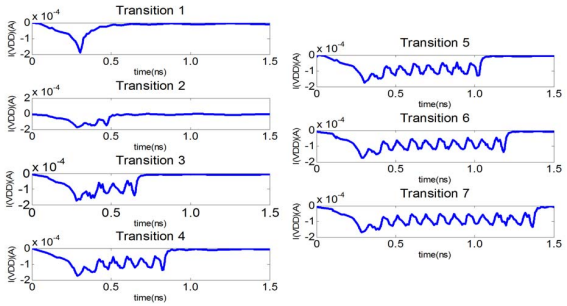


Fig. 6. Self-referencing current signatures of 8-bit Array Multiplier.

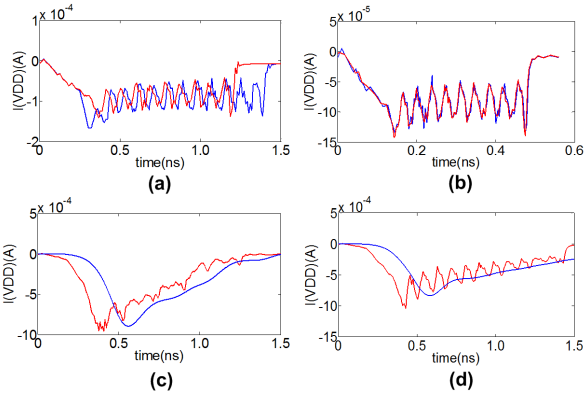


Fig. 7. Self-referencing current signatures of an 8-bit Wallace Tree Multiplier and the corresponding current of Array Multiplier for comparison.

can clearly see the rippling effect in supply current which indicates the carry propagating to the Most Significant Bit (MSB) for RCA. The overlapping current for the 3 vectors confirms the ripple propagation of the most significant 8 bits. For the blocked CSA, if the carry is at the input of a block, the triggered blocks have the same switching activity, forming the block propagation signature(red and blue traces in the center sub-figure of Fig. 5(c)). However, if generated inside a block, the block propagation waveform will only appear when the carry propagates to the next block (the green curve). Similar signatures can be derived at another technology node (65nm), as shown in Fig. 5(d), again confirming the technology and process independent nature of the signatures. Quantitatively, cross-correlation is performed between pairs of shapes to measure the similarity. Then the correlation values in response to different vector pairs are digitized and multiplied together to obtain the overall correlation with respect to one test set. Finally, signatures from all test sets collaboratively define the actual signature of a datapath element implementation.

Multiplier: Current signatures exploring structural self-similarity of multipliers can be obtained in a similar way. For 8-bit Array Multiplier, the following transitions are applied: **T1:** (0x02, 0x00)→(0x02, 0x01); **T2:** (0x04, 0x00)→(0x04, 0x01); **T3:** (0x08, 0x00)→(0x08, 0x01); **T4:** (0x10, 0x00)→(0x10, 0x01); **T5:** (0x20, 0x00)→(0x20, 0x01); **T6:** (0x40, 0x00)→(0x40, 0x01); **T7:** (0x80, 0x00)→(0x80, 0x01). The corresponding switching current is shown in Fig. 6. In each transition, only one partial product is made to be 1 and propagate to one primary output through a series of full adders. Regularly increasing number of ripples in the switching current indicates an array structure.

On the other hand, the structure of a Wallace Tree Multiplier (WTM) is relatively irregular. Test vector pairs T1, T2 and T3 are applied for triggering current signatures. In particular, T1 sensitizes the longest path with no carry propagation (Fig. 7(a) red curve), indicating a shorter path than that of an 8-bit array multiplier (Fig. 7(a) blue curve), thus implying WTM. T1 and T2 sensitize two different paths with exactly the same structure, which is specific to WTM. The identical waveforms form a signature verifying this self-similarity.
T1: (0x20, 0x00)→(0x20, 0x08); **T2:** (0x80, 0x00)→(0x80, 0x01); **T3:** (0x00, 0xff)→(0x80, 0xff). Another feature of WTM is that the switching activities are more focused on the former levels compared to other types of multipliers to reduce the critical path delay, which is explored by T3. We first pre-process the current waveform by filtering out the high frequency components, then use a "normalized slope" of the rising part to represent the signature metric.
Metric 1: The ratio of the peak current value (I_{peak}) over that of the middle time point of the rising part of the switching current (I_{mid}). (WTM (Fig. 7(c)) $>$ 2.3, Array multiplier (Fig. 7(d)) $<$ 2)
Metric 2: The ratio of a normalized switching current amplitude over a normalized switching current duration. The former one is defined as (I_{peak} -I_{end})/I_0, whereas the latter one is T_{tran}/T_0. I_{end} is the current value of the last time point in the post-filtering waveform, T_{tran} is the switching duration of the real switching waveform, while I_0 and T_0 are the peak current and switching duration of a 1-bit full adder, which can be obtained from both multipliers by applying certain test vectors. (WTM $>$ 3.2, Array multiplier $<$ 2.3)

After obtaining datapath element structures, the remaining combinational logic is grouped as random logic with no pre-determined current signature. By applying test vectors to trigger each small group of gates, different gate-level transient current signatures can be obtained and compared with a pre-characterized signature library, e.g. trigger certain paths while setting other inputs to non-controlling values.

Scenarios Without A Golden Design: Unavailability of a golden design makes reverse engineering gate-level implementation of random logic to be a remarkably difficult task. Because there is no golden netlist to verify, test vector generation is not oriented. In this case, we adopt the approach described in the flow chart in Fig. 8. First the logic expression obtained from logic testing [7] is synthesized to a gate-level netlist. Then iterative side-channel based verification is performed based on this *initial guess*, during which the predicted netlist is updated with the confirmation or modification of each predicted gate. For each logic level, test vectors are intelligently generated to *focus the switching activity* on a small number of gates. Considering F=A&B′, the dual manners to implement

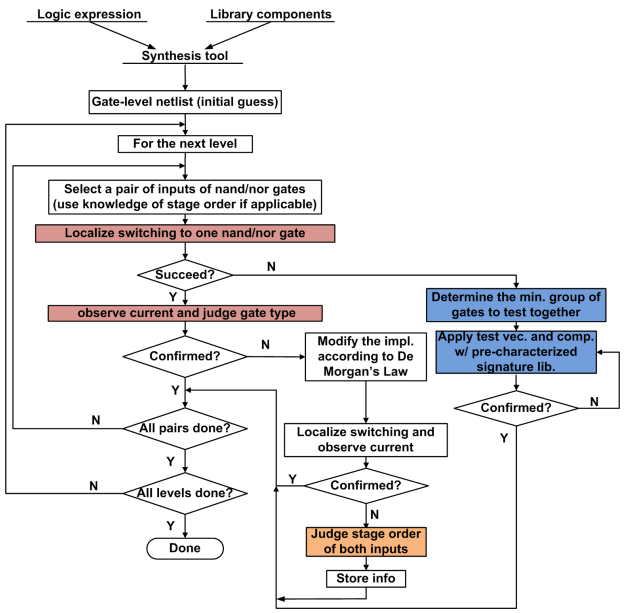


Fig. 8. Steps of random logic structure identification.

this function using a reduced library is shown in Fig. 9(a) and (b). Considering Fig. 9(a) as the initial guess, to verify this gate, input B is set to the controlling value '1' while switching A. If the prediction is correct, switching of a single inverter should appear; while for the case of (b), no switching activity is expected. Repeating the test with A kept constant helps confirm that A and B are direct inputs of the gate. A case where B indirectly limits the switching caused by A for function F is given in Fig. 9(c). Here, both 0→1 and 1→0 at B would cause significant switching even if A is set to its controlling value.

However, in this step two exceptions might be encountered. First, if neither of the dual implementations can be confirmed, it implies mis-prediction of the existence of a gate; hence a different set of nodes have to be tried as the inputs. The other exception occurs when the switching activity cannot be limited to one NAND/NOR gate according to the predicted netlist, which could happen because of *shared input logic cone* that leads to loss of independent controllability of different gates. In this case test vectors are generated targeting multiple gates as a group, followed by a current signature comparison step.

The hierarchical top-down reverse engineering process, as described above, is very amenable to automation. The side-channel analysis steps at different levels of hierarchy can also be fully automated. However, the only step that requires

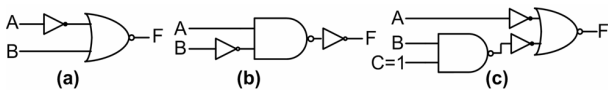


Fig. 9. An example of the verification unit: (a),(b) Dual implementations of function F=A&B′. (c) Here, B indirectly limits the switching caused by A.

Fig. 10. Temporal self-referencing helps to identify the pipeline stage currents of a DLX processor.

manual intervention (and hence can only be partially automated) is the high-level test generation based on functional specifications. This needs to be based on the functional block-diagram for a chip and can vary widely from design-to-design. The final result of the RE process is the complete gate-level implementation, along with hierarchical functional and structural description. Any undesired gate or function is easily identified as a malicious modification or Trojan circuit.

IV. Case Study: DLX Processor

We perform the automated reverse engineering procedures on a 32-bit DLX processor to prove its effectiveness. All simulation results are obtained by performing HSPICE simulations in Predictive Technology Model (PTM) 70nm [11] technology.

1. Partitioning sequential space using Temporal Self-Referencing: By filling the processor pipeline with the same instruction, we can ensure that only one pipeline stage has switching activity in each clock cycle. Special instructions such as NOPs and JUMPs are used to characterize the background switching current of program counters and state transition of the pipeline stage control FFs. Once all the background current information is obtained, it is subtracted from the total current to focus on the individual pipeline stages such as Instruction Decode (ID), Execute (EX), Memory (MEM) and Write Back (WB). As shown in Fig. 10, the current signature for each stage corresponding to an ADD instruction is different from that for NOP, and the current for each stage has a unique signature in terms of peak current, delay and other transient current shape information.

2. Identifying and isolating sequential current component: As shown in Fig. **??**, for structured sequential cir-

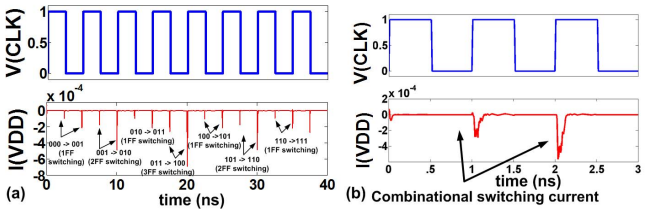


Fig. 11. Extraction of combinational logic current by subtracting sequential current component: (a) 3-bit binary counter shows the FF switching pattern of 1-2-1-3 which can be easily identified from the current at the positive or negative edge of CLK. (b) Extracting combinational current.

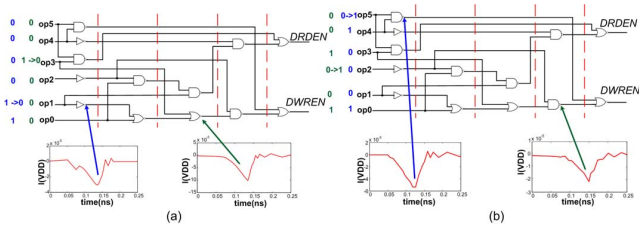


Fig. 12. Transient current signatures corresponding to specific vectors used to identify random logic structure isolated from the MEM stage of the DLX processor, with dependence on (a) a0, a1 and a3; and (b) a2, a4 and a5.

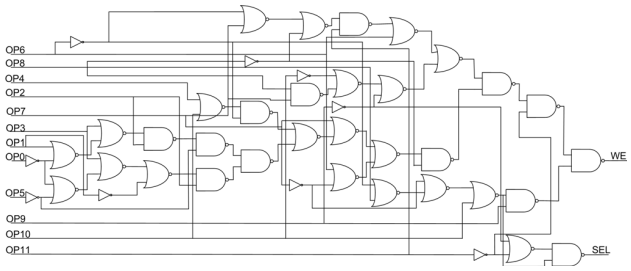


Fig. 13. Random logic structure of WB stage of the DLX processor.

cuits such as shift registers and counters, there are process-independent current signatures which are clearly identifiable and can be detected and eliminated. Similarly, memory access instructions such as LOAD/STORE can be used to find current specific to memory access circuitry. By careful selection of instructions, we can estimate width of memory, structure of address decoders and other peripheral logic, and timing of memory access relative to other operations.

3. Identifying datapath elements by Spatial/Temporal Self-Referencing: By exploring self-similarity of datapath elements using temporal/spatial self-referencing, we reverse engineer the implementation of the structured datapath elements. For example, we identified a CSA and a WTM in EX stage by applying vectors as described in Section III.

4. Identifying random logic and datapath sub-structure by combining side-channel analysis with logic testing: In this step, we successfully reverse engineer random logic in MEM and WB stages of the DLX processor after subtracting out background current due to other stages, the sequential current, and memory current. In MEM stage, two output logic cones structures *DRDEN* and *DWREN* with function are derived, where *DRDEN* is data read enable signal and *DWREN* is data write enable signal for memory access, respectively. The Boolean functions obtained from logic testing approach:
DRDEN = op5 & op4′ & op3 & (op1′| op2′ & op0)
DWREN = op5 & op4 & op3′ & op2′ & (op1′| op0)
Particularly, by switching different input bits of the MEM stage, we first figure that they are functionally dependent on input bits *op5*, *op4*, *op3*, *op2*, *op1*, *op0*. Then the Boolean equations are derived by applying exhaustive test vectors at these six inputs. Based on this, we obtained the predicted netlists using synthesis tool. After applying the verification procedure, the actual circuits are found as illustrated in Fig. 12, in which some transient current waveforms are also shown to demonstrate the netlist verification process. Similarly, in WB stage, we reverse engineer the structure of logic for MUX select signal *SEL* and write-back enable signal *WE*. The schematics for the actual logic are illustrated in Fig. 13.

V. Conclusion

We have presented a novel reverse engineering based IC trust validation process which combined transient current based side-channel analysis with logic testing based function extraction. We have shown that RE can be used for trust validation in two scenarios: 1) when golden design is available; 2) without golden design (i.e. with functional specification only). Although we focus on using RE for trust validation, the process can also be adapted to improve the effectiveness of conventional manufacturing test. The validation steps can be easily automated to minimize the cost and time of trust validation. Since the technique works at multiple levels of hierarchy, it is easily scalable to large designs. The approach can work without scan, although presence of scan can be leveraged to improve the logic function extraction process. The proposed RE based trust validation can be used in conjunction with other existing protection approaches. For example, low-cost hardware Trojan detection approaches using static/transient current signature can be used for fast security screening of manufactured ICs, while the proposed approach can be used to increase the level of trust significantly. Future investigation would focus on developing an automation framework and validation with measurement results from commercial ICs.

Acknowledgment

We acknowledge the support from NSF grant CNS-1054744.

References

[1] D. James. “Reverse engineering delivers product knowledge, aids technology spread”. *Electronic Design*, 2006. [Online]. Available: http://electronicdesign.com/Articles/Index.cfm?AD=1&ArticleID=11966

[2] DARPA, “Integrity and Reliability of Integrated Circuits (IRIS)”, 2010. [Online]. Available: https://www.fbo.gov/index?id=342ac5ed191ae7b8b03357fead590c4e

[3] Chipworks, Inc., “Semiconductor manufacturing - reverse engineering of semiconductor components, parts and process”. [Online]. Available: http://www.chipworks.com

[4] R.S. Chakraborty, F. Wolff, S. Paul, C. Papachristou and S. Bhunia, “MERO: A statistical approach for hardware Trojan detection”, *CHES Workshop*, 2009.

[5] M. Tehranipoor and F. Koushanfar. “A survey of hardware Trojan taxonomy and detection”. *IEEE Design and Test of Computers*, 2010.

[6] M.C. Hansen, H. Yalcin, and J.P. Hayes. “Unveiling the ISCAS-85 Benchmarks: A case study in reverse engineering”. *IEEE Design and Test of Computers*, vol. 16, no. 3, pp. 72-80, 1999.

[7] D.G. Saab, V. Nagabudi, F. Kocan, and J. Abraham. “Extraction based verification method for off the shelf Integrated Circuits”. *ASQED*, 2009.

[8] S. Borkar *et al*, “Parameter variations and impact on circuits and micro-architecture”, *DAC*, 2003.

[9] D. Du, S. Narasimhan, R.S. Chakraborty and S. Bhunia, “Self-referencing: A scalable side-channel approach for hardware Trojan detection”, *CHES*, 2010.

[10] R. Rad, J. Plusquellic and M. Tehranipoor, “A sensitivity analysis of power signal methods for detecting hardware Trojans under real process and environmental conditions”, *IEEE TVLSI*, 2010.

[11] Predictive Technology Model, [Online] http://www.eas.asu.edu/∼ptm/