Microprocessors and Microsystems 38 (2014) 899–910

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Customized and automated routing repair toolset towards side-channel analysis resistant dual rail logic

Wei He *, Andres Otero, Eduardo de la Torre, Teresa Riesgo

Technical University of Madrid, Spain

ARTICLE INFO

Article history:
Available online 24 February 2014

Keywords:
Dual-rail Precharge Logic (DPL)
Routing repair
FPGA
Power consumption attacks
Xilinx Design Language (XDL)
Routing conflict

ABSTRACT

Dual-rail Precharge Logic (DPL) has been widely studied as an effective countermeasure category for mitigating Side Channel Attack (SCA) threats, where unwanted physical leakages from running crypto devices are inspected and analyzed to retrieve confidential data. DPL protocol requires compensated behavior between the corresponding rails, which differs from conventional logic principles. Thus it needs unusual design flows with repetitive and tedious workload. In this article, we present a custom execution tool to automatically realize a dual rail logic. This controllable and automated design flow relies on Xilinx FPGA platforms, to obtain dual rails with highly symmetric networks. The tool is able to automate the logic transformation from a raw single rail on Xilinx Design Language (XDL) to the Precharge Absorbed DPL (PA-DPL) format. Users can fully or partially convert the circuit in arbitrary placement schemes, without concerning the routing conflicts. Another significance is that this proposal is potentially to be used to other circuits that require precise routing control. SCA Security verification to an 8-bit AES coprocessor on SASEBO-GII indicates enhanced security grade due to the rigorous routing networks achieved by the repair process. Timing analysis further demonstrates that the net delay differences between complementary nets are minimized.

1. Introduction

Modern crypto-algorithms provide enhanced protections against conventional algorithmic crypto-analyses which require unacceptable computation capability and time cost. Side Channel Attack (SCA) was first introduced in [1], and formalized in [34], which typically observes the power consumption (Power Analysis) or Electromagnetic (EM) leakages from running crypto-devices (microprocessor, ASIC, FPGA, etc.) to restore confidential data or internal behaviors. Typical SCA focuses on specific intermediates in logic path, where a part of the known data and the cipher key are correlated. By making simple or correlation comparison between the recorded side channel leakages and the hypothetic leakages, the crypto keys or other secrets can be disclosed with handy laboratory equipments. Since it passively eavesdrops and analyzes the physical emanations without disturbing the circuit function, it is difficult to be detected and defended by prior ordinary countering mechanisms.

DPL is a rendezvous of SCA-resistant logic styles on fundamental logic cells [35], due to its dynamic and compensated behavior between its True (T)/False (F) rails [2]. In DPL, each logic value is represented by two complementary variables in the T and F rails, and the evaluation/precharge phases are switched alternatively. The value a (T) in T rail and a (F) in F rail compensate with each other (i.e., 'a (T): a (F)' is always in state of '1:0' or '0:1' during the Evaluation Phase). In the precharge phase, a (T) and a (F) are both reset to a fixed state (typically '0'). In this way, variants in power or EM traces are dynamically flattened.

DPL theoretically hides data-dependant variations without releasing exploitable leakage. However, the tremendous number of cells required to populate a modern crypto-circuit makes the implementation challenging in practise. Actually, any minor asymmetry (detectable unbalanced compensation) in circuit might possibly be reflected in eavesdropped traces [3–5,16], and hence cripple the robustness. Normally users have extensive control over ASIC design on synthesis, placement and routing. Comparatively, hardware resources are prefabricated by FPGA vendors, thus designers can only depend on the fixed resource to constitute the logic. Optimization strategies in FPGA synthesizer favor designs while they might significantly alter the expected schemes that need to be reserved in dual rail framework. This is particularly true

* Corresponding author. Tel.: +34 913363191.
E-mail addresses: wei.he@upm.es (W. He), andres.otero@upm.es (A. Otero), eduardo.delatorre@upm.es (E. de la Torre), teresa.riesgo@upm.es (T. Riesgo).

http://dx.doi.org/10.1016/j.micpro.2014.02.005
0141-9331/

in terms of routings because of the deficient flexibility over vendor provided FPGA tools. A number of prior literatures have discussed about DPL realization in FPGA environments [2,17–20]. A masking method is proposed in [6,7,37] that aims to mitigate the dual-rail bias by swapping the dual rails in a random behavior. A later investigation reveals that only masking is not secure enough to overcome the unbalanced routings [26].

Routing in FPGA always exists as a security concern because of its unflexibility. In this article, we intend to remove this barrier by means of the development of a specific purpose toolset which is able to automatically generate a dual rail logic from an unprotected single one. Our method depends upon Java script from a set of APIs, RapidSmith [21,23], to yield highly symmetric networks in Xilinx FPGA platforms. One of the major benefits is that it works on Xilinx Design Language (XDL) and the repairer can exclusively parse the nets. In this way, the repair work ignores logic styles and can hence be partially or be fully applied to other circuits in similar hardware environments.

Xilinx commercial tools employ a proprietary binary format named Netlist Circuit Description (NCD) to internally describe designs either after technology mapping or after place and route steps. The tool proposed in this work takes those netlists generated by the commercial tools as an input, and carries out the necessary transformations both on instance and nets, without impairing logic functions. To have access to the design, instead of NCD, a human-readable equivalent version, Xilinx Description Language (XDL) is used. The XDL design is parsed and modified exploiting a set of Java APIs based on RapidSmith, which has been purposely enhanced with some specific routing functions. Processed XDL can be subsequently transformed into a valid NCD file to create the configuration bitstream.

The rest of the paper is presented as follows. Section 2 introduces the previous work for solving the routing challenges in DPL. Section 3 presents the technique principles and details of the tools developed, especially focusing on the routing repair mechanism. Section 4 specifies the security evaluation with a simplified cryptographic coprocessor generated from our tool. Conclusion and perspective for the future work are drawn in Section 5.

2. Background and related work

Security of DPL logic mainly suffers the defects from its unbalance routing between each T and F net pair. In order to produce a sound DPL logic, compensated behavior between the complementary rails must be achieved precisely. However, physical parasite asymmetry [8,16,24] and limited control over vendor provided router make this task difficult to be implemented in practical applications, particularly when the target device is FPGA where massive optimization always overrides users' routing schemes and therefore incur undesirable route formats after the auto – place and route process.

2.1. Previous routing solutions

Quite a few methods have been proposed to alleviate the bias from unbalance net pairs that exist in FPGA implemented DPL logics. Typically, T/F gates are located side by side to reduce the routing variants. This way merely reduce signal skew because the route algorithm is still uncontrollable and any minor adjustment in logic placement may significantly change the network pattern.

A copy and paste strategy is proposed when placing the combinational logic part of T/F rails of a WDDL circuit separately [2]. It is based on the fact that the FPGA fabric encloses a large number of logic cells deployed in a highly regular array. So, the single rail logic (T rail) can be completely duplicated and deployed in a neighboring fabric which is not occupied by rest design part. This technique is further adopted in Double WDDL [17], light weight DDL [18] and Separate PA-DPL [25]. However, the separate dual-core framework releases a critical weakness since the EM field strength from T and F parts cannot be guaranteed to be balanced if a sophisticated micro-EM antenna is precisely positioned closer to either one of the two cores. In this scenario, side channel EM emanations are not compensated well and may lead to failure in resisting differential power or EM analyses. In [28], a masking solution to counter route unbalance is stated where interconnect pairs are randomly swapped with bit masking. In this way, the routing bias can be ignored in both T and F rails. Whereas, the logic size remains as a concern because the value-fetch mechanism used in MDPL is based on majority function. Combined with the masking logic, each logic unit is pretty sizable. A later investigation reveals that only masking is unsecure to overcome the unbalance routings [3,26,27]. Improved from MDPL, iMDPL [29] aims to solve Early Propagation Effect [3]. Further augmented logic complexity makes it not attractive in use.

Balanced Cell-based Dual-rail Logic (BCDL) introduced in [19] is capable of resisting Early Propagation Effect (EPE) [27] by synchronizing all the N pairs of inputs in a compound N-input gate. But the route unbalance is as well unsolved. According to the analyses in [30], high fan-out and high gate count contribute to a large portion of side channel leakages because of T/F route bias. Similar conclusions are stressed in [31,36]. Accordingly, reducing fan-out of each logic tree by Block RAM (BRAM) to implement security-sensitive combinatorial part does decrease impact from routing bias. Nevertheless, route bias is not totally eliminated since this solution just creates similar T/F nets (compared to the result from nets with high fan-out), but not strictly identical as desired. To the best of our knowledge, Copy and Paste process is the only viable technique published to date being able to yield identical routing pair in FPGA environment.

2.2. DREAMS tool

Dynamic and partial reconfiguration deliver important benefits, ranging from cost reduction due to the use of smaller FPGAs, to power consumption savings [15]. Moreover, dynamically reconfigurable systems deliver an unrivalled capacity to be adapted or upgraded at run-time. This flexibility enables envisaging more autonomous operating conditions.

DREAMS [9] is the result of an academic research work, which tries to remove the limitations found in commercial tools for the design of this kind of systems. Main limitations are (a) the communication overhead and the lack of flexibility introduced by the interfaces among reconfigurable modules; (b) the area overhead resulting from the impossibility of introducing area constraints in the routing phase; and (c) the dependence with the low level device details. With regard to the communication overhead, solution implemented in DREAMS constitutes a trade-off between the first Xilinx approach, based on the so-called bus-macros [10], and the proxy logic, proposed in the last version of the commercial tool PlanAhead [11]. In the case of bus-macros, this solution incurs high delay and area overhead, since it consists on using presynthesized structures, including fixed logic and wire resources, instantiated at both sides of every reconfigurable border. This way, after the reconfiguration process, signal building up the interface use the same wires. Differently, proxy logic reduces this cost by substituting logic resources by wires belonging to the static design, but located in the reconfigurable region. This way, all the signal crossing the module boundaries use the same resources. Main drawback of proxy logic is the impossibility of relocating modules. This reduces the flexibility provided by dynamic and partial reconfiguration technique. Moreover, it requires the simultaneous generation of