Secure FPGA Circuits Using Controlled Placement and Routing

Pengyuan Yu, Patrick Schaumont
ECE Department, Virginia Tech, Blacksburg, VA 24060
Email: peyu1983@vt.edu, schaum@vt.edu

ABSTRACT

In current Field-Programmable-Logic Architecture (FPGA) design flows, it is very hard to control the routing of sub-modules. It is thus very hard to make an identical copy of an existing circuit within the same FPGA fabric. We have solved this problem in a way that still enables us to modify the logic function of the copied sub-module. Our technique has important applications in the design of side-channel resistant implementations in FPGA. Starting from an existing single-ended design, we are able to create a complementary circuit. The resulting overall circuit strongly reduces the power-consumption-dependent information leaks. We show that the direct mapping of a secure ASIC circuit-style in an FPGA does not preserve the same level of security, unless our symmetrical routing technique is employed. We demonstrate our approach on an FPGA prototype of a cryptographic design, and show through power-measurements followed by side-channel power analysis that secure logic implemented with our approach is resistant whereas non-routing-aware directly mapped circuits can be successfully attacked.

Categories and Subject Descriptors

B.6.1 [Design Style]

General Terms

Measurement, Design, Experimentation, Security

1. INTRODUCTION

Secure circuits are used in many portable information devices such as smart-cards, smart-phones, and RFID. These circuits need to resist attacks on the integrity, the confidentiality or the authenticity of embedded electronic information. We can distinguish three broad categories of attacks on embedded electronics [1].

- Logical attacks exploit bugs in the embedded software, or weaknesses in the interfaces [2].
- Side-channel attacks exploit physical phenomena of the electronics, such as power-consumption and execution time [3].
- Physical attacks exploit the physical implementation itself and rely on probe stations, chemical solvents, and microscopes to gain inside knowledge of a chips' operation [4].

Physical attacks are the most systematic kind, but they are very expensive to implement. Side-channel attacks are a more economical alternative, and they are more systematic than logical attacks. Recent research efforts have provided many feasible scenarios for these side-channel attacks. Fortunately, circuit-level defenses against side-channel attacks do exist. Using constant-time, constant-power design techniques, side-channel information leakage can be hidden from the external observer. Virtually all secure circuit techniques proposed so far are specifically designed with an ASIC implementation in mind [6][7][8].

However, also FPGAs are an excellent platform for secure circuit design. Reconfigurable fabrics have an excellent resistance against physical attacks since the underlying platform is regular and does not reveal information on the actual design content. In an SRAM-based FPGA, a design exists only as long as the device is configured and powered. In this paper we will show that it is also possible to develop efficient side-channel resistant circuits in an FPGA.

It is unfortunately not possible to port secure circuit styles from ASIC directly onto FPGA. We will demonstrate this by breaking an FPGA prototype of a well-known secure logic style, based on building complementary-switching circuits.

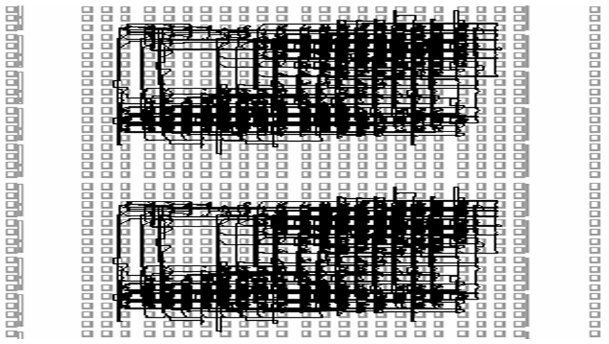

Figure 1: Route Preserving Design Duplication

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CODES+ISSS '07, September 30–October 3, 2007, Salzburg, Austria.
Copyright 2007 ACM 978-1-59593-824-4/07/0009…$5.00.

To remedy this effect, we propose a place-and-route (P&R) technique to complement the selected logic style. Our P&R technique is able to create an identical duplicate of a previously placed-and-routed module, such that all routing information is preserved as shown in Figure 1. This way, we can create two modules which will have identical capacitive loading, and consequently identical power profiles. The symmetric P&R technique is then used to create precisely-matched complementary modules with symmetrical routing. We are able to demonstrate that our symmetrical routing technique is necessary in order to achieve security against power analysis attacks on a FPGA.

Our paper is organized as follows. Section 2 presents the basics of a pre-charged differential logic style. In section 3, we provide an overview of our approach to implement secure logic on FPGA. Then, in section 4, we demonstrate how to implement secure logic on an FPGA and introduce our routing technique. Section 5 presents security evaluations of the circuits implemented with and without our routing approach. We conclude the paper in section 6.

2. BACKGROUND: SIDE-CHANNEL RESISTANT LOGIC

In our research, we are specifically concerned with power-based side-channel attacks. There are two major classes of secure circuits, both of which attempt to remove the correlation between the overall circuit power consumption and the secret data values at selected circuit nodes. *Masking* uses random bits to make the resulting circuit power consumption random [7]. *Differential logic* on the other hand attempts to make the resulting power consumption constant [5][6]. The most prominent of these technologies is Wave Dynamic Differential Logic (WDDL, [6]). It is characterized by the following properties.

- Consistent Switching Activities: In order to keep power consumption constant, dual-rail differential logic is used. It guarantees a single switching event per clock cycle, independent of input data: when the direct logic switches high (consuming current from power supply), the complementary logic will switch low (discharging the capacitive load).
- Pre-charge Wave Generation: In the absence of data changes, a transition is provided by means of a pre-charge circuit. When the clock is high, both the direct and complementary logic enter a pre-charge phase, where every net is switched to logic 0. When the clock becomes low, the circuit enters the evaluation phase, where actual computation is done. WDDL implements the pre-charge circuit at the register outputs and system inputs [6], and lets a 0-wave ripple through the entire circuit.
- Negative-logic Removal: Inverters disrupt the pre-charge wave propagation because wave front is inverted. Therefore, WDDL uses only positive logic and implements inverters by cross-coupling nets from the direct logic with the complementary logic.
- Routing Procedure: Any routing asymmetry between direct and complementary logic results in unbalanced net loading, and in a residual power variation between direct and complementary transitions. Current techniques to control routing keep the direct and complementary gates close to each other, so that resulting nets are as symmetrical as possible.

All four aspects of WDDL have been addressed for ASIC designs, in which layout features (circuit elements and routing) can be fully controlled. The same thing can not be said for FPGA. We therefore adapted an existing FPGA design flow to address each of the four aspects. We will first provide an overview of the flow, and next present circuit- and routing-details.

3. FPGA APPROACH TO SECURE LOGIC

We have chosen to start from WDDL logic to implement side-channel-resistant logic into an FPGA. Current state-of-the-art results have shown that the key challenge for this secure logic style is to maintain the symmetry between the direct and complementary sides of the circuit.

Maintaining this symmetry in an FPGA requires precise control over the placement and routing of the differential circuits. We used a logic-modify-and-relocate approach, as illustrated in the design flow of Figure 2. The starting point is a single-ended high level design, from which we will prepare a fully-routed and pre-charged module for FPGA by going through a set of transformations. This WDDL module is then processed with symmetrical routing technique to produce two complementary modules, A and A_C. The two modules both implement the same WDDL circuit but have interchanged signal pairs. For example, a dual-rail signal pair (1,0) in A is represented as (0,1) in A_C. As a result, the overall loading on each signal pair is constant regardless of the signal transitions. We call the final logic circuit Double WDDL (DWDDL).

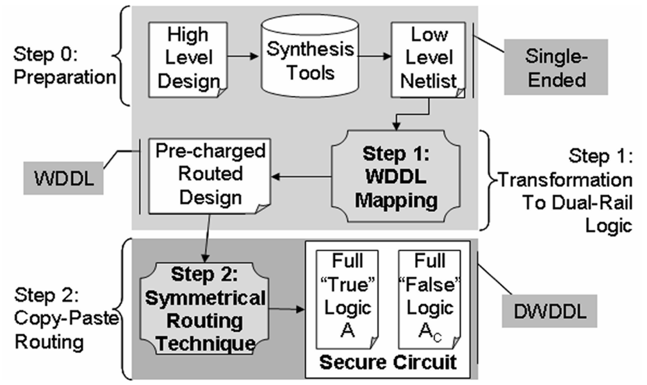

Figure 2: Secure Logic Implementation Flow

4. MAPPING WDDL INTO FPGA

In this section, we will discuss the transformations used to map ASIC WDDL logic style onto FPGA. First, we will create a pre-charged LUT representation of a normal/direct WDDL module. Next, we will create a complementary module out of the direct LUT netlist, while at the same time, copy all the routing information to the new complementary circuit. The next two subsections will introduce the two transformations.

4.1 WDDL for FPGA

We performed our experiments on a Xilinx Spartan3E FPGA. Most Xilinx FPGA have the following properties: for logic implementation, each Configurable Logic Block (CLB) contains 4 slices. Each slice contains two 4-input Look-Up-Tables (LUT), 2 dedicated multiplexers, 2 memory elements and some miscellaneous logic; each CLB is connected through the on-chip routing network through a switchbox. While we will perform the discussion on circuit design for this technology, we note that FPGA from other manufacturers provide a similar grouping of LUTs, multiplexers and registers.

4.1.1 Complementary LUT

To map a design onto FPGA while following WDDL logic style, we start from a gate-level netlist, which can be obtained from a synthesis tool such as Synopsys Design Compiler. The gate library used for synthesis does not have to be limited to all positive logic since this netlist will be converted into its' WDDL equivalent.

We then substitute each gate with 2 LUTs: one that implements the direct functionality of the gate; the other one implement the complementary function. If the gate contains negative logic, the output wires of these LUTs are cross-coupled. As illustrated in Figure 3, complementary LUT can be derived from direct LUT by inverting every bit and reversing the ordering of significance of the inverted bits [6].

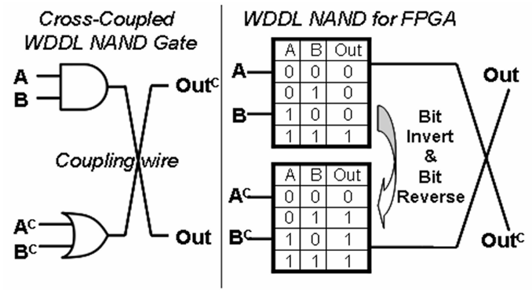

Figure 3: WDDL NAND gate on FPGA

4.1.2 Pre-charge circuits

The purpose of the pre-charge circuit is to force the output of a slice to zero during the pre-charge phase. In a Xilinx slice, each LUT is followed by a dedicated multiplexer and a memory element that can be configured as a register or a latch. We considered the use of multiplexer and memory element for pre-charging, as illustrated in Figure 4.

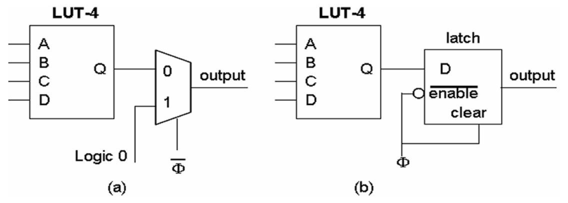

Figure 4: Pre-charge circuit (a) using a clock-controlled multiplexer and (b) using a latch.

Each has advantages and disadvantages:

1. By implementing pre-charge with a clock-controlled multiplexer, we keep the memory element in each slice free for use as a register. However, due to the structure of the multiplexer in the slice, we need to supply and distribute an inverted clock in addition to the regular clock used by registers.
2. We can also use the memory-element as an asynchronously-cleared transparent latch with inverted enable input. This way we only need to supply a single clock signal to both the registers and the pre-charge latches. The pre-charge functionality is implemented by connecting the clock to both the inverting enable input and the clear input of the latch. The disadvantage of this approach is that design-specific register storage now will require a separate slice.

We opted for the second approach because duplication of the clock signal into a direct and complementary form would significantly increase the power consumption and area of the circuit, and it would complicate the routing.

The WDDL circuit obtained after previous transformation steps satisfies all the requirements defined in section 2 except for the routing. Indeed, by default, FPGA routing tools do not provide control over the routing, so it is not possible to guarantee identical loading of the direct and complementary parts of a dual-rail gate. As will be demonstrated, the resulting asymmetry in realistic circuits is large enough to enable power-analysis attacks.

To overcome this problem, we will next develop an automated placement and routing control methodology.

4.2 SYMMETRICAL ROUTING: DWDDL

We have developed a design flow based on existing FPGA synthesis tools that achieves precise control of placement as well as routing in modular designs. While this technique can be useful in many different situations, we applied it in this paper to the placement and routing of the secure logic style developed in previous section.

Modular design is a technique that is frequently used for hierarchical hardware design. It allows reuse of lower-level hardware blocks, and per-module synthesis of complex designs. This procedure is described by current FPGA design flows as the creation of a "hard macro". Present tools only preserve the relative placement of CLB logic, but they do not preserve the interconnections. As a result, the relocation of a hard macro to another place in the FPGA fabric will lose guaranteed timing properties. In addition, the process of making a hard macro for a large functional module requires manual interaction of the designer in an FPGA editor to manually remove a design from physical input/output pin constraints. This process soon becomes laborious and impractical as the design size increases. Therefore we focused on implementing an automatic hard-macro procedure that preserves internal routing information during relocation.

4.2.1 Secure Routing Design Flow

Figure 5 presents the symmetrical routing design methodology for FPGA. All tools used in this flow are standard FPGA synthesis tools. We used Xilinx FPGA synthesis suite (Xilinx ISE Foundation). The design flow requires 3 design iterations, including synthesis, place and route. In between the iterations, we added two more processing steps. These steps, including LUT content modification, are implemented using a scripting language. The scripts perform conversions on an ASCII representation of the FPGA netlist (obtained from the NCD-format using the XDL tool from ISE).

The first iteration only synthesizes the initial single-ended module to be relocated or duplicated, and produces a LUT-level netlist. The module is synthesized as a "closed" design and has all its input/output ports disconnected from I/O pins.

Before the second iteration, the module netlist is preprocessed by adding identifier tags to component and net names. This is useful for quickly identifying which components and nets are part of the initial single-ended module. The module is area-constrained within the FPGA fabric. This can be done automatically based on the structure of the target FPGA platform or manually using floor-planner software. The second iteration then produces a fully closed but routed module based on the area constraints.

During post-processing, the module can be modified in the following two ways:

- **Relocation**: We can modify the module by changing the location of every component and routing resource used. The module can move freely within the boundaries of the FPGA fabric as long as the target location provides sufficient CLB and routing resources. For example, relocated modules typically cannot overlap fabric-level features such as BlockRAM.
- **Logic Modification**: We can also create a second module based on the design information in the first one. Internal components, net names and LUT contents can be modified while the routing information is preserved. We use this to create a complementary module, which can next be relocated.

Post-processing also extracts design information that will be back-annotated into the original constraints file. With the post-processing step, an arbitrary number of copies can be produced as long as the overall design still fits on the FPGA fabric. The layout result of a direct and complementary module looks as shown in Figure 1.

In the third iteration, we need to complete the routing of the closed modules to input/output pins. Each module is instantiated in a top-level file, which acts as a wrapper to external logic blocks. The top design then goes through a regular synthesis flow using the complete constraints generated during post processing. The design is placed and routed using the combined design guide file after post-processing.

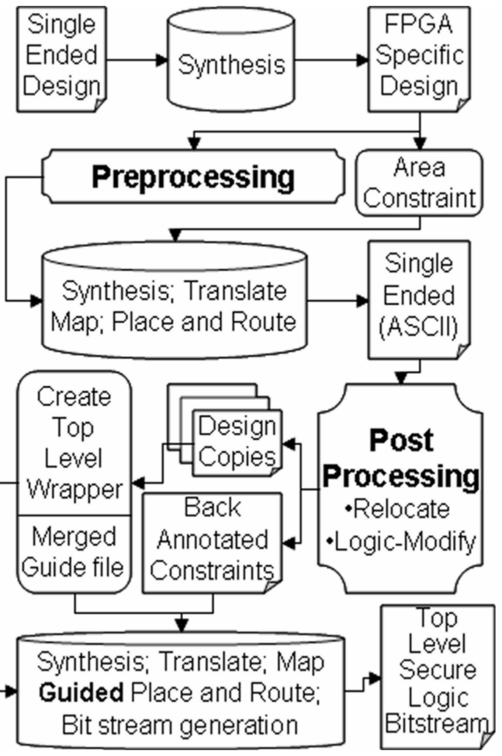

Figure 5: Symmetrical Routing Design Flow

Using the logic-modify-and-relocate approach, we can create a complementary module for an asymmetric WDDL module, as was explained in Section 3. The complementary module A_C is obtained as follows from the module A: During the logic-modify, we can switch the contents of direct and complementary LUT of the WDDL circuit. Next, module A_C is relocated so that it does not overlap with module A.

5. RESULTS AND POWER ANALYSIS

To validate our approach, we implemented a sample design on a Spartan3E FPGA. The design in Figure 6 includes a cryptographic S-Box which is driven with test patterns out of an 8-bit Linear Feedback Shift Register (LFSR). The S-Box table was taken from the advanced encryption standard (AES). The output of the S-box is combined with a secret key-byte to produce the output samples. This architecture is a simplified version of a structure found in many block ciphers, including AES. The circuit is clocked by a digital clock manager (DCM) on the FPGA fabric.

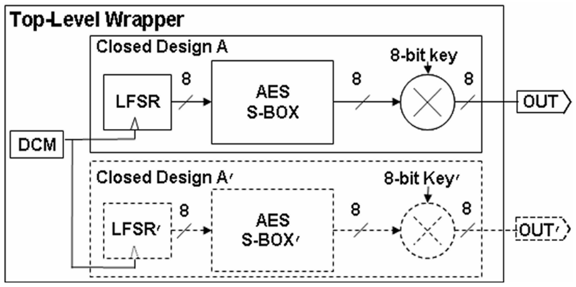

Figure 6: SBOX Test Circuit Setup and Implementation

5.1 Measurement Methodology

We used an Agilent DSO3062A digital storage oscilloscope which has maximum sampling rate of 1Gsample/sec and 60MHz bandwidth. To obtain enough sample points per cycle, we lowered our circuit speed to 5MHz (the circuit operates reliably up to 50MHz). Due to the 8-bit LFSR in Figure 6, we obtain a complete power trace in only 256 clock cycles.

We measured only the power consumption from the FPGA fabric and bypassed the PCB power control IC. The power source was an external linear-mode power supply for minimal noise interference. We then used an inductive 1mV/mA current probe to convert current consumption variation into voltage variation to be detected on the oscilloscope. We also removed all the decoupling capacitors on the internal fabric power input of the FPGA to maximize the success of detecting power consumption variations. The current probe has a band-pass frequency response that will reject any DC signal, but will pass variable AC signals from 1.2 kHz to 200 MHz.

5.2 Power Measurement and DPA-Attack

We prepared 3 different test cases for comparison: a single-ended design, a non-routing-aware WDDL design, and a symmetrically routed DWDDL design. All designs use a fixed 8-bit key with value of 8'hAE, decimal 174.

Figure 7 shows the current variation for the first 100 cycles in a 255-cycle period. A 10-cycle snapshot, corresponding to the shaded area, is shown in the inset. The single-ended (SE) variation is over 10 times larger than that of WDDL and DWDDL and is scaled down to fit inside of the inset. Visually, the variations of DWDDL look smaller than those of WDDL. Both of them are much smaller than the variations of SE.

To test the effectiveness of our DWDDL secure circuit, we mounted a differential-power-analysis (DPA) attack for all three cases. In a DPA attack, we estimate the circuit power consumption using a power model that combines the observed output values (OUT) with an estimate for the key value (KEY_{GUESS}). We then correlate the estimated power, for each key guess, with the measured power. The highest correlation over all key guesses will return the correct key. We used power traces of 256 samples.

For the single-ended design, our power model is the Hamming weight (Hw) of the input value of the Sbox. This value depends on the key guess and the output, since

$$Hw(IN) = Hw(SBOX^{-1}(OUT \otimes KEY_{GUESS}))$$

where *Hw( )* represents the Hamming Weight function. The resulting correlation of the SE design over all key guesses 0…255 is shown in Figure 8.

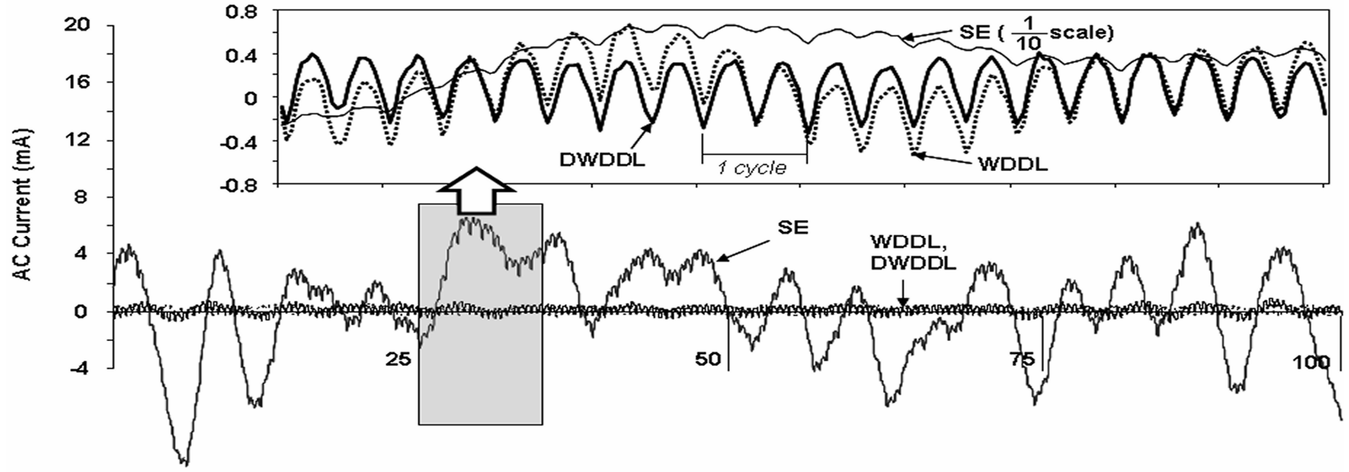

Figure 7 Current Variation Measurement of All Test Cases; enlarged 10-cycle section (inset)

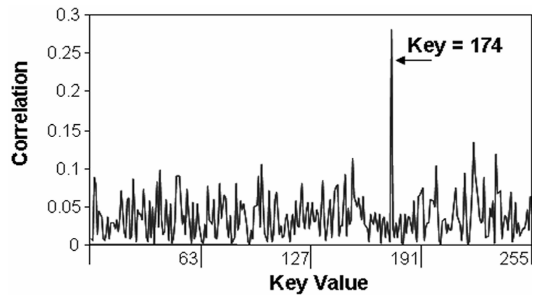

Figure 8 DPA-Attack on SE case

The DPA-attack on the single-ended case results in a very sharp correlation peak at key guess 174. This means that the single-ended design can be easily broken.

WDDL and DWDDL are both dual-rail circuits. Side-channel leaks on these circuits are caused by imbalanced routing. We therefore mounted a DPA attack on WDDL and DWDDL that exploits those imbalances. Our power model in this case is the Hamming Weight of a *single bit* on the input of an SBOX.

$$Hw(bit(IN,i))\Big|_{i:0..7} =$$
$$Hw(bit(SBOX^{-1}(OUT \otimes KEY_{guess}),i))\Big|_{i:0..7}$$

Indeed, the single-bit power model allows each bit of the byte-wide circuit to have a different routing imbalance and a different leakage. We then obtain a correlation plot for each of the 8 bits, find the maximum in each plot and use majority voting among all bits to arrive at the final key. Table 1 shows the key guesses for maximal correlation for each bit. By majority vote on bit 3, 4, 6 and 7 for WDDL, we are able to obtain the correct key value for WDDL. In contrast, we cannot find a correct key for DWDDL. All bits lead to a different key in the correlation process, all of them equally likely. This means that WDDL does not withstand a DPA attack, while DWDDL does. Note that a single bit from the DWDDL circuit still identifies the correct key, and we suspect that this is caused by a parasitic second-order effect.

Table 1 DPA Attack on Individual Input Bits

Bit	0	1	2	3	4	5	6	7
WDDL	19	152	68	174	174	99	174	174
DWDDL	194	174	64	27	190	238	10	113

5.3 Implementation Summary

All secure-logic implementations incurred considerable area and delay overhead, as demonstrated in Table 2. As shown in the table, both WDDL and DWDDL are considerably bigger and slower than the most compact implementation of a single-ended SBOX+LFSR. DWDDL is essentially twice the size of WDDL as expected, but in return, secure-circuit built using our DWDDL technique withstands DPA attacks.

6. CONCLUSION AND FUTURE WORK

As seen from actual measurement and actual DPA attacks, our approach to implementing secure logic for FPGA can achieve a better result than the current, non-routing aware WDDL approach. Perfect security does not exist. Our results quantify the tradeoff that can be made between circuit area increase and side-channel resistance. Eventually, the decisions in that tradeoff will still have to be made by the designer. From the experiments done by us and others, we conclude that a high price is paid for increased security in terms of resources. This makes the problem more challenging and interesting. In the end, circuit-level techniques and system-level techniques (such as partitioning for security) will have to be combined to achieve acceptable security. We plan to improve our technique in order to reduce the area/performance overhead while at the same time, further increase the level of security against DPA-attacks.

Table 2 LFSR+SBOX Implementation Results

	Slice Cost	Delay (ns)	Security
Single-Ended	70	3.99	NO
WDDL	409	19.54	NO
DWDDL	**818**	**20.88**	**YES**

7. REFERENCES

[1] M. Witteman, "Advances in Smartcard Security," Information Security Bulletin, July 2002, p. 11-22.

[2] M. Bond, R. Anderson, "API-level attacks on embedded systems," IEEE Computer, 34(10):67-75, Oct, 2001.

[3] P. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis," in M. Wiener, editor, Advances in Cryptology: Proceedings of CRYPTO '99, LNCS, 1666:388-397, Springer-Verlag, 1999.

[4] S. P. Skorobogatov, "Semi-invasive attacks: a new approach to hardware security analysis," University of Cambridge, Technical Report UCAM-CL-TR-630, April 2005.

[5] D. Sokolov, J. Murphy, A. Bystrov, A. Yakovlev, "Design and analysis of dual-rail circuits for security applications" IEEE Trans. on Computers, 54(4): p 449-460, , April, 2005

[6] K. Tiri, I. Verbauwhede, "A digital design flow for secure integrated circuits," IEEE Trans. on Computer-Aided Design, 25(7):1197 - 1208, July 2006.

[7] D. Suzuki, M. Saeki, T. Ichikawa, "Random Switching Logic: A New Countermeasure against DPA and Second-order DPA at the Logic Level," IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences 2007 E90-A(1):160-168

[8] T. Popp, S. Mangard, "Masked Dual-Rail Pre-charge Logic: DPA Resistance without the Routing Constraints," Proc. of CHES 2005, LNCS 3659, p. 172-186, August 2005.