IP Core Protection using Voltage-Controlled Side-Channel Receivers

Peter Samarin, Kerstin Lemke-Rust,
Bonn-Rhein-Sieg University of Applied Sciences
Sankt Augustin, Germany
Email: {peter.samarin,kerstin.lemke-rust}@h-brs.de

Christof Paar
Ruhr University Bochum
Bochum, Germany
Email: christof.paar@rub.de

***Abstract*—This paper presents a new method for protecting netlist-based Intellectual Property (IP) cores in FPGAs by actively using voltage-controlled side-channel receivers. The receivers are realized by modulating the supply voltage of the chip, while at the same time detecting these changes from within the chip using a ring oscillator. The levels of the supply voltage can be determined by constantly monitoring the frequency of the ring oscillator.**

To prove authorship of an IP core, the verifier authenticates himself to the core over the voltage side-channel and sends commands that limit the core's functionality. By monitoring the regular outputs of the overall system, it is possible to detect illegitimately used cores after repeatedly turning them on and off. The working principle of our method is demonstrated by a case study, in which we protect several IP cores and place them on a Spartan 3 FPGA, and show the steps necessary for successful proof of ownership verification.

I. Introduction

Intellectual property (IP) theft exists in many forms. Unauthorized usage, reproduction and distribution of IP, possibly under different labels, reduces revenue and can damage reputation of the original developers. IP theft can be counteracted by embedding watermarks and computing fingerprints that can help prove the authorship of the IP and enable tracking the distribution of a product.

Since the beginning of the wide-spread use of FPGAs, several IP protection methods have been developed. The techniques can embed watermark into IP cores at several design levels, and make it possible to identify unauthorized IP use. For example, constraint-based watermarks [1] embed a signature as a set of additional constraints into a design. If a suspicious design satisfies the watermarking constraints, it is considered as a proof of authorship. Watermarking methods based on finite-state machines (FSM) [2], [3] place a signature into existing and unused transitions of a FSM and recover the signature by traversing through the FSM using specific inputs. Other approaches use the scan chain to embed and read out a watermark [4].

The authors would like to thank Jacek Samotyja for comments on the watermarking schemes, Timur Saitov, Georg T. Becker, and the EMSEC group at the Ruhr-Universität Bochum for valuable comments and discussions.

This work has been supported in parts by the German Federal Ministry of Education and Research (BMBF) through the project DePlagEmSoft, FKZ 03FH015I3.

Depending on the level at which the adversary has access to the IP, he can either use the IP core "as-is", as a stand-alone design, or incorporate it as part of a larger design with several other IP cores. Many watermarking schemes work only on full designs, and have the problem that they cannot be verified in the field. In addition, some PROM-based FPGAs provide the ability to encrypt the bitstream, increasing the effort needed to prove ownership of the IP because the key has to be broken prior to the verification step, e.g., by using a side-channel attack [5].

To solve this problem, several new watermarking schemes have been developed that make use of side-channels. Side-channels have the advantage that they can leak data without the need of an extra pin, even from IP cores in a large design. For example, in [6], the authors present a commercial system for IP protection of individual cores using the temperature side-channel. Each IP core sends a unique signature that can be read out using a temperature sensor and correlated to a set of signatures from a database. The power side-channel was used in [7] to transmit a signature in a way that can be decoded after sampling the power trace using a digital oscilloscope. An improvement of this approach was developed by [8], whose contribution was to hide the watermarking signal below the noise level, so that an adversary does not learn anything from recording the power trace, unless the watermark is made public. In an attempt to reduce the size of the leakage circuit that is used for transmitting the watermark over the power side-channel, [9] presented a method that utilizes clock gating. A watermark is sent over the power side-channel by turning on and off the clock of the IP core. The proof of ownership is achieved by correlating the power traces with the known watermark.

In contrast to the state of the art approaches that use *output-only* side-channels that constantly broadcast the authorship mark, we propose a method based on *input-only* side-channel that can be used to control each individual IP core. By allowing commands to be sent to the IP core, we can turn it off, or limit its functionality and in this way prove the authorship. In this paper we focus on protecting netlist-based IP cores.

Our approach is related to previous work by [10], who investigate temperature-modulated side-channel receivers.

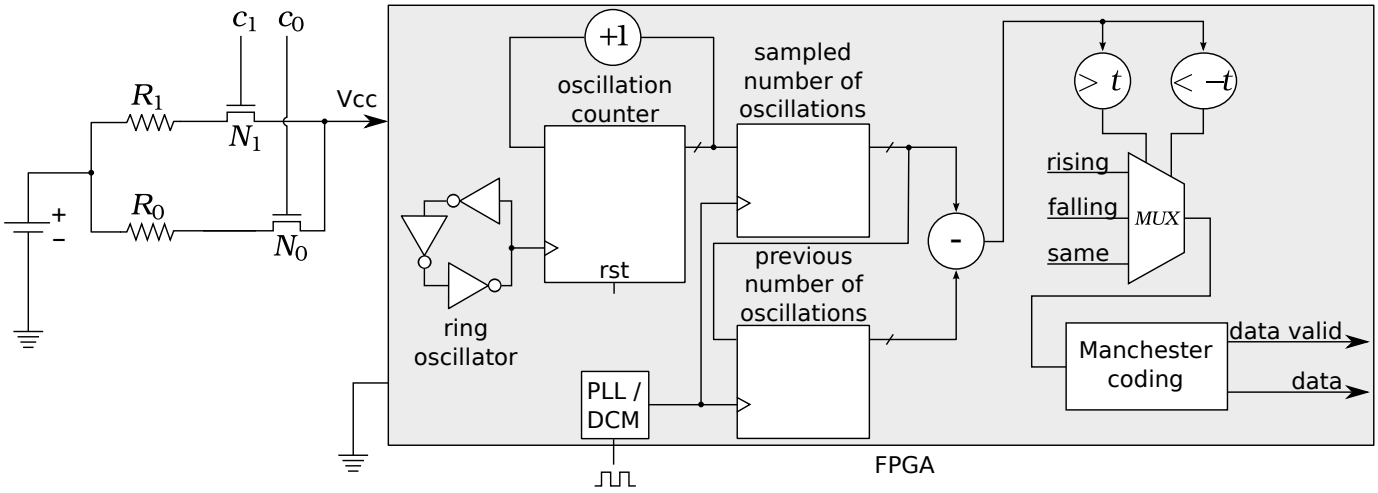

Fig. 1. The voltage side-channel for data transmission to an FPGA. The analog circuit on the left is used to control the level of the supply voltage of the FPGA by setting and unsetting the wires c_0 and c_1. The digital circuit in the FPGA on the right determines the voltage level indirectly by sampling the frequency of the ring oscillator with a fixed clock. Two consequtively sampled frequencies are subtracted and compared to a fixed threshold t, in order to detect rising and falling edges. In the next step, Manchester coding is used to convert this information to one bit of data at a time.

However, unlike slow temperature modulation with data rates of few bits per second, we use voltage modulation that allows much faster transmission speeds. Though voltage modulation is also mentioned in [10], it is dismissed for two reasons: the authors claim that it is difficult to build a voltage-modulating system with high enough accuracy; the second claim is that in order to perform the voltage-modulation, the voltage regulator of the FPGA board has to be modified or replaced.

In this paper we show that we do not have to be accurate in setting the voltage of the FPGA board, as long as we can produce two distinct voltage levels that are sufficiently distinct from each other. In addition, the results of our experiments using a Spartan 3 board show that it is not necessary to replace or modify the voltage regulating circuit, and that it is sufficient to replace the power supply of the FPGA board by a small breadboard circuit and a voltage source.

II. Voltage-controlled side-channel receivers

In this work, we establish a side-channel for data transmission to an FPGA that is based on voltage modulation. The input to this side-channel is realized by setting the supply voltage of the FPGA to two distinct levels that can be distinguished inside the chip and converted to data.

A. Supply voltage control

The motivation for our method is the variable switching time of CMOS transistors that depends on the manufacturing process, supply voltage, temperature, and electromagnetic field. The manufacturing process is static and determines the base level switching speed, whereas the last three factors can influence the switching speed dynamically at run time. Out of the three effects, the impact of voltage is the strongest, and also the easiest to implement.

The supply voltage of the chip is controlled by the analog circuit shown on the left of Fig. 1. The voltage can be either controlled directly, by cutting the wires to the power supply pins of the chip and reconnecting them to our circuit, or indirectly by replacing the power supply of the board.

TABLE I
Voltage levels and their control

c_1	c_0	Vcc
0	0	V_{reset}
0	1	V_0
1	0	unused
1	1	V_1

The circuit is operated by setting and unsetting the wires c_0 and c_1. Four voltage levels of Vcc can be set, however, only three are used, as shown in Table I: $V_{reset} = 0V$ is used to reset the chip; the voltage levels V_0 and V_1 are used for data transmission. V_0 and V_1 are chosen as far as possible from each other, while making sure that the chip can still be operated. The stronger the difference, the easier can the voltage levels be distinguished on the chip. The wires c_0 and c_1 set either by hand using push-buttons, or automatically by using a microcontroller, or even an FPGA to have better control over the activation timings. To transmit some data, each voltage level is applied for a fixed amount of time.

B. Voltage monitoring

To detect changes in supply voltage without having access to an analog-to-digital converter, we use inverter-based ring oscillators (ROs) in combination with a fixed reference clock source. A RO consists of an odd number of inverters connected in series. The output of the last inverter is fed back as input to the first inverter. This makes the inverters oscillate continuously at a frequency that depends upon the

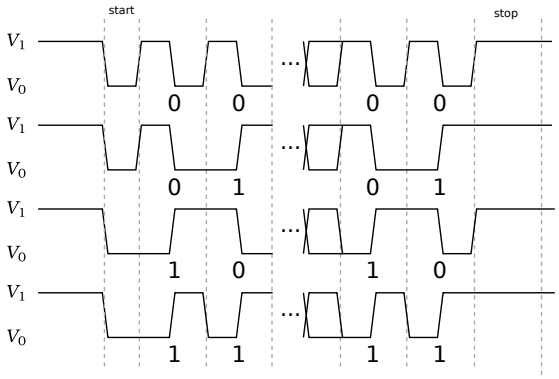

Fig. 2. Custom Manchester coding with a falling-edge start sequence

total number of inverters in ring oscillator, and the signal propagation delay from one inverter to the next.

The right hand side of Fig. 1 illustrates how the changes in the supply voltage are detected on an FPGA. The free-running ring oscillator is repeatedly sampled for a fixed amount of time using a fixed reference clock that is unaffected by the voltage modulation. Two consecutive samples are subtracted from each other and compared to a threshold, in order to find out whether the frequency of the RO stayed the same, is rising, or falling.

To convert the three cases into data, we use a modified version of Manchester coding. Manchester coding converts a sequence of rising and falling edges into a sequence of bits. The original Manchester coding cannot be applied in our case because we do not have the "high-impedance" level, because there is an ambiguity about the start of the transmission. For this reason, our version requires a falling edge in the beginning of each transmission. Fig. 2 shows examples of how different bits can be sent using our version of Manchester coding.

The threshold can be estimated based on some parameters of the IP core. The number of oscillations of a ring oscillator measured by a fixed clock can be computed as follows:

$$\frac{c \cdot T}{r \cdot d}, \tag{1}$$

where c is the number of clock cycles of the fixed clock with period T that the protection circuit waits before sampling the number of oscillations of the RO, r is the number of inverters in the ring oscillator, and d is the time that it takes for a signal to propagate from the input of one inverter to the input of the next. Thus, the numerator represents the sampling time, and the denominator represents the time it takes for the ring oscillator to perform a single oscillation. We purposefully omit including voltage levels into the formula because there is no easy way to convert it into propagation delays, since an IP core might be used on a wide range of FPGAs made using different manufacturing processes. By increasing the sampling time, increasing the difference between the voltage levels, and choosing a smaller threshold, we can nevertheless cover a wide range of FPGA technologies and clock speeds.

Using Eq. (1), we can compute the upper bound of the threshold:

$$t \leq \left| \frac{c \cdot T}{r \cdot d_0} - \frac{c \cdot T}{r \cdot d_1} \right| = \left| \frac{c \cdot T}{r} \cdot \left(\frac{1}{d_0} - \frac{1}{d_1} \right) \right| \tag{2}$$

For example, if the period of the fixed clock is $T = 5ns$, the number of clock cycles is $c = 128$, a ring oscillator with three inverters $r = 3$, and the two voltage levels can set the propagation delays of each inverter to $d_0 = 1.2ns$ and $d_1 = 1ns$, then the threshold is $t \leq |213.33 - 177.77| \approx 35.56$ oscillations.

An inverter is realized by occupying a lookup table of the FPGA. Due to the algorithms used in placement and routing, the position of the lookup tables will be different after each compilation run. Thus, the oscillation frequency of resulting ROs will be different as well. If the frequency is too low, the subsequent threshold might not be large enough to distinguish between two consecutively sampled oscillations. To have more reliable results, the relative position of the inverters has to be fixed by using either attributes or constraints. In addition, the synthesis tools of FPGA vendors will usually detect redundant inverters and optimize the circuit by reducing the chain to a single inverter. This optimization step can be prevented by using signal attributes at the level of hardware description language, or constraints in the netlist.

III. IP CORE PROTECTION

This section describes how voltage-controlled side-channel receivers can be used to protect any arbitrary IP core and to prove the authorship in the field.

A. Protection method

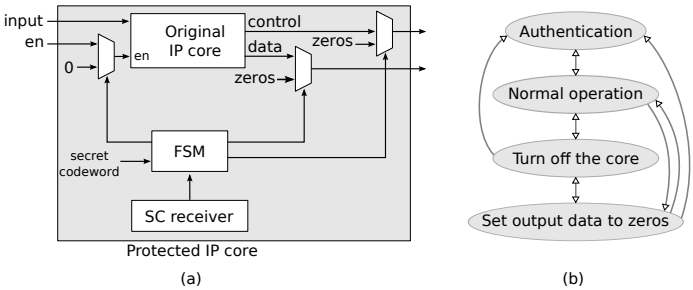

Fig. 3. IP core protection using SC receivers. (a) A high-level view of a protected IP core (b) States of the finite state machine that controls the proof of authorship verification process

To protect an IP core, it is packaged in a wrapper together with a side-channel (SC) receiver and a finite state machine (FSM) that can control the operation of the core. Fig. 3 shows a block diagram of how these component interact with each other. The FSM interprets the data received over the side-channel and intervenes with inputs and outputs of the original IP core.

Initially, the FSM waits for a secret codeword in the authentication state. In this state, the IP core functions normally, and the data from SC receiver is continuously compared to the codeword. Depending on the desired security level, the size of the codeword can be chosen accordingly. In case that a correct codeword is received, the FSM starts accepting commands. In this state, the behavior of the IP core still remains unchanged. Based on the received command, the FSM might replace inputs and outputs of the IP core by some predefined values.

Four commands are possible. The *first* command turns off the core by switching off its enable signal en. In cases when the core does not have any enable signal, all outputs of the core can be set to zeros instead. The *second* command sets the output data of the core to zeros, while keeping original control signals intact. The surrounding IP cores will not notice any changes in the functionality of the core, but its data will be corrupted. The *third* command returns the core back to its normal operation. The *last* command deselects the core, allowing the verifier to communicate to another core. Depending on the command, the state machine chooses the appropriate input and output signals and routes them to the original IP core. More sophisticated commands can also be implemented by integrating the FSM deeper into the core and let it select internal states of the core, and set the values of chosen signals. However, this will require a larger effort to protect each IP core.

B. Proof of authorship verification

To prove the authorship of an IP core that is hidden between other cores of a system, the verifier should have physical access to the suspicious chip, and should be able to control the voltage of the FPGA either directly or indirectly. In the next step the verifier authenticates himself to the core with codewords from a database. Though the verifier might have a suspicion about the nature of the illegitimately used IP core, in the worst case, he has to check all codewords from the database.

Since no direct feedback is available about the success of the authentication, the verifier tries out several commands right after sending a codeword. The verifier will be able to notice that a core has stopped working by monitoring the behavior of the overall system. In the final step, the verifier repeats the authentication with the correct codeword and demonstrates that he is able to turn the core off and on again, and even corrupt the outputs of the core. In this case, he will have a high confidence that the IP core is used without a license.

IV. Possible attacks and countermeasures

There are several ways in how an attacker might try to successfully assimilate a protected IP core. The attacker might have access to the IP at different design levels. We rule out the soft IP available in the form of hardware description languages (HDLs) because it is easy to find and remove the protection layer. Therefore, we concentrate only on the IP cores available in the netlist form, in which case the attacker might incorporate the core as part of a larger design, with a mixture of legitimate and stolen IP cores; or as a bitstream, in which case the attacker will load it on an identical FPGA.

There are several ways to attack our protection method: clock, bitstream, and physical manipulation of components on the FPGA board. Here we address possible attacks and countermeasures to stop them.

A. Attacks using the clock

The attacker is free to choose the inputs to the stolen IP core including the clock. Modern FPGAs have several components to manage the clock, such as PLLs, clock buffers, clock gating, etc. Thus the attacker can replace the clock source by turning it off completely, switching between several clock sources that run at different frequencies, and use a ring oscillator as a clock source.

Stopping the clock — The attacker can stop the clock of the IP core for some time. Stopping the clock stops the counting of the fixed clock, but the ring oscillator continues running. At the time when the RO frequency is sampled, the ring oscillator will have ran much longer than in the previous time it was sampled. This might be detected as a rising or a falling edge even when the voltage channel was not touched. This problem can be addressed by making the sampling interval higher, so that waiting for several clock cycles of the fixed clock will not make any difference. To successfully perform this attack, the adversary has to turn the clock off for a long time, which will severely limit the performance of the IP core.

Use two different clock sources — The unauthorized user of the IP core, after learning that the core is protected, can infer two clock sources using a PLL and switch between them from time to time. Since the fixed clock source is used as a reference to count the number of oscillations for presumably fixed amount of time, this attack might corrupt the data sent to the side-channel receiver. However, it might also corrupt the performance of the IP core, because switching between two clock sources might violate the timing constraints of the IP core. Another problem with this attack is that it might not be able to meet the timing requirements in case when the second frequency is much higher. Additionally, the design needs to be synchronized with the other cores, which will require additional design effort.

Use a ring oscillator as a fixed clock — Instead of fixed reference clock derived from a quartz crystal, the attacker can provide the IP core with a clock based on a ring oscillator. Our input side-channel will not work because measuring the number of oscillations of one RO using another RO always result in the same number, since both ROs are affected by the voltage changes in the same way. However, this attack has the disadvantage that the resulting clock will have a large jitter, which might violate some timing constraints, so that the IP core might not function

properly. Also, the IP core will be sensitive to temperature, EM field, and fluctuations in the supply voltage and might stop working randomly.

B. Bitstream and netlist manipulation

The attacker might know that an IP core is protected using our method and try to remove the protection by manipulating netlist of the IP core or the bitstream of the final design. If the attacker knows the algorithm, he has no way of knowing whether or not the protection was removed, because the protection circuit becomes active only after successful authentication.

C. Physical attacks

Additive analog noise — The attacker can add some components to the FPGA board that add noise to the supply voltage. However, the verifier can observe the supply voltage of the chip and detect any anomaly. The difference between voltage levels for rising and falling edges can be increased, so that the noise has less influence. In addition, the effect of the noise can be reduced by increasing the sampling time, or by taking several measurements in a row. For example, instead of a measuring a frequency once and deciding about the received bit value, the protection circuit can take several measurements, sum them up, and used that value to make a decision.

Adding capacitors to the FPGA board — Additional capacitors between the voltage regulating components and the FPGA will act as a low-pass filter and reduce the frequency of operation of the voltage side-channel. The slow change in the chip voltage can be detected by measuring the supply voltage of the chip directly or indirectly by using EM probe. The problem can be addressed by increasing the sampling time of the ring oscillator. The board can also be inspected visually, and the extra capacitors physically removed. As a last-resort countermeasure, this problem can be addressed by cutting the wires of the supply voltage of the chip and feeding it from external voltage source.

Temperature modulation — An attacker can try to influence the temperature of the chip by using a temperature element. This will change the oscillation frequency of the ROs and might result in errors during data transmission. However, the verifier can measure as well as control the temperature of the chip during the verification.

Laser beam attacks — This attack requires the knowledge of the implementation details in order to be successful, as the ring oscillator and the protection circuit are relatively small compared to the rest of the application. In addition, it cannot be used in large volumes, so that only a very small number of chips can be sold. Also, the attacker cannot be sure that the protection was removed without knowing the secret codeword.

D. Ghost attacks

An attacker can always implement his own watermark and claim that it is used to protect a particular IP core. However, it will be difficult to explain why an IP core responds to the commands of the verifier.

V. Experimental results

We use Digilent's Spartan 3 starter board for our proof-of-concept implementation. The board has a Spartan 3 FPGA with 1920 slices, uses 4-to-1 LUTs, and runs at 50 MHz. We protect four IP cores using our method. The overall system is shown in Fig. 4.

The FPGA is used to perform two tasks independently. One task is to receive some data from a PC over a serial interface, to encrypt it using a 128-bit AES core, and to send the encrypted data back to the PC. The second task is to generate some pseudo-random data and to visualize it on a display using a 4-bit VGA controller. Each IP core is wrapped in a protection layer together with a voltage-regulated side-channel receiver and a FSM, as described before. Furthermore, each core has a unique 80-bit codeword.

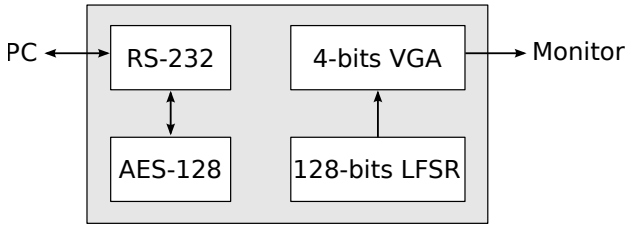

Fig. 4. A case study with four protected IP cores

The voltage to the Spartan 3 board is supplied by a direct current voltage source and a breadboard circuit that can generate three voltage levels: $V_{reset} = 0V$, $V_0 = 2.8V$, and $V_1 = 3.2V$. Despite the fact that both voltage levels are lower than 5V that are normally supplied by the original power supply, the FPGA is able to operate properly, except for the time when it needs to be configured by a new bitstream, in which case it requires the regular voltage level.

The breadboard circuit is controlled by a second FPGA board that can send data to the Spartan 3 using Manchester coding. The board has a Cyclone 2 and also runs at 50 MHz. We tested several transmission speeds and were able to achieve a data rate of approximately 24 kbit/s. To achieve this data rate, a half-bit is sent out every 1024 clock cycles.

Due to the capacitors on the board, a voltage level takes some time to propagate, so that at higher speeds, it is not possible to reach distinct voltage levels before switching to the next one. This effect is demonstrated in Fig. 5. Higher speeds should be possible if we can control the voltage of the FPGA directly, over its Vcc pins.

Proof of authorship — To prove the authorship of each IP core, the verificating FPGA repeatedly tries to send some data to the core by probing the codewords from a database one after another, and follows them up with some commands. The commands are always issued in the same order: turn the core off, change the data output to all-zeros, return the core to normal operation and unselect the

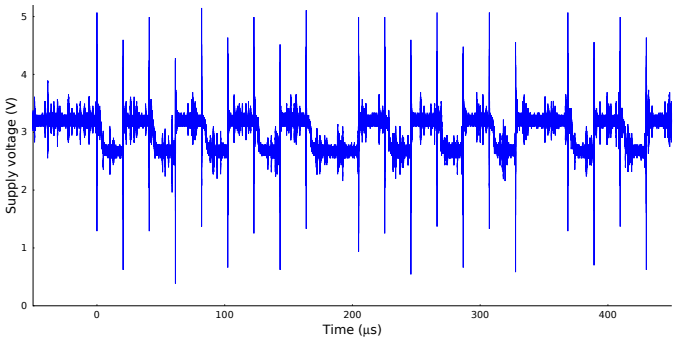

Fig. 5. Modulated supply voltage of the Spartan 3 evaluation board. The sequence "0000111100" is being transmitted.

core. At the same time, the board is operated normally—the PC constantly sends data to the board using the RS-232 interface, waits for the encryption to finish and receives the encrypted data, while the monitor is observed by a human operator. In general, every input and output of the system should be recorded, so that later on unexpected behavior can be detected and analyzed automatically. In our case, the synchronization and the data signals of the VGA core should be recorded by a digital oscilloscope, and saved in a file together with the codewords, the data sent to the RS-232 core, and the corresponding timestamp.

After turning off the RS-232 and the AES core, no communication is possible with the board, however, after setting the data of the cores to zeros through the next command, only zeros will be received on the PC. By knowing at what time the "all-zeros" command has been issued, and the currently active codeword, the verifier can find out which IP core is present in the system with a high confidence.

The other two cores—the LFSR, and the VGA core—can also be detected reliably. By turning off the LFSR core, the monitor switches from a pseudo-random pattern to a unicolor pattern. By setting the output bit of the LFSR core to zero, the monitor turns black. Same with the VGA core—it shows a "no signal" message when the core is turned off, and is black when the FSM switches into the "all-zeros" state.

Resource usage — The overhead of the protection circuit mostly depends on the size of the codeword. Table II shows the resource usage.

TABLE II
RESOURCE USAGE BASED ON THE SIZE OF CODEWORD

Codeword size (bits)	Number of slices
32	49
64	70
80	81
128	111

VI. CONCLUSIONS AND FUTURE WORK

We presented a new method for protecting netlist-based IP cores by using an input-only side-channel that is realized by modulating the voltage of an FPGA board. To prove authorship, the verifier has first to authenticate himself to the protected core with a secret codeword. After successful authentication, the verifier can reliably change the behavior of the protected core by sending it commands over the side-channel.

Apart from detection of stolen IP cores, the input-only side-channel can have other uses. For example, the side-channel receiver together with the state machine can be integrated into a chip in order to prevent counterfeiting. A chip can be proven to be a counterfeit if it does not answer to the commands after receiving correct codeword. In the same manner, our approach can be used to trigger a hardware trojan and even choose which data it should leak. Another use case is debugging—by using commands, an IP core can be set to a specific state, or a signal can to a specific value.

In the future, one of our goals is to establish two-way communication based on side-channels. This will allow realization of more interesting watermarking schemes, such as the challenge-response scheme. Another research direction is a tighter integration of the IP core and the protection circuit so that it becomes impossible to remove the protection circuit without damaging the IP core.

REFERENCES

[1] A. Kahng, J. Lach, W. Mangione-Smith, S. Mantik, I. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Constraint-based watermarking techniques for design IP protection," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 20, pp. 1236–1252, Oct 2001.

[2] I. Torunoglu and E. Charbon, "Watermarking-based copyright protection of sequential functions," *Solid-State Circuits, IEEE Journal of*, vol. 35, pp. 434–440, March 2000.

[3] A. Oliveira, "Techniques for the creation of digital watermarks in sequential circuit designs," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 20, pp. 1101–1117, Sep 2001.

[4] A. Cui and C.-H. Chang, "An improved publicly detectable watermarking scheme based on scan chain ordering," in *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, pp. 29–32, May 2009.

[5] A. Moradi, M. Kasper, and C. Paar, "Black-box side-channel attacks highlight the importance of countermeasures," in *Topics in Cryptology – CT-RSA 2012* (O. Dunkelman, ed.), vol. 7178 of *LNCS*, pp. 1–18, Springer Berlin Heidelberg, 2012.

[6] T. Kean, D. Mclaren, and C. Marsh, "Verifying the authenticity of chip designs with the designtag system," in *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*, pp. 59–64, June 2008.

[7] D. Ziener, F. Baueregger, and J. Teich, "Using the power side channel of FPGAs for communication," in *Field-Programmable Custom Computing Machines (FCCM), 2010 18th IEEE Annual International Symposium on*, pp. 237–244, May 2010.

[8] G. T. Becker, W. Burleson, and C. Paar, "Side-channel watermarks for embedded software," in *9th IEEE NEWCAS Conference (NEWCAS 2011)*, 2011.

[9] J. Kufel, P. Wilson, S. Hill, B. Al-Hashimi, P. Whatmough, and J. Myers, "Clock-modulation based watermark for protection of embedded processors," in *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pp. 1–6, March 2014.

[10] J. Sun, R. Bittner, and K. Eguro, "FPGA side-channel receivers," in *Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, FPGA '11, (New York, NY, USA), pp. 267–276, ACM, 2011.