Multiplexing Methods for Power Watermarking

Daniel Ziener, Florian Baueregger, and Jürgen Teich
Hardware/Software Co-Design, Department of Computer Science
University of Erlangen-Nuremberg, Germany
email: {*daniel.ziener, teich*}*@cs.fau.de*

***Abstract*—In this paper, we present several enhancements to power watermarking that allow to simultaneously transmit and verify multiple signatures. Power watermarking of netlist IP cores for FPGA architectures is used for detecting IP fraud where the signature (watermark) is transmitted over the power supply pins of the FPGA. Many (watermarked) IP cores can be combined in an FPGA design, which raises the question of how multiple signatures can be detected using the same set of pins. As a solution, we propose multiplexing techniques for power side channel communication, so that all watermarked cores inside the FPGA can be identified to establish a proof of authorship. We analyze different multiplexing methods in order to adapt them to power watermarking and provide experimental results with several cores concurrently transmitting signatures.**

I. Introduction

Due to the ongoing miniaturization of on-chip structures, the complexity of chips expressed in terms of transistors per chip rises rapidly. This allows us to implement very complex designs which require careful engineering and a great effort for implementation, debugging, and verification. The complexity increase of such chips, however, outweights the design productivity. This is further aggravated by the market requiring short product cycles. The only solution is to reuse certain common and specific design parts that have been written for other projects or were purchased from other companies to close the *productivity gap*. The market for these so called *IP cores* (Intellectual Property cores) has risen from year to year and this trend seems to continue in the future.

IP cores are licensed and distributed like software. One problem of the distribution of IP cores, however, is the lack of protection against unlicensed usage. Cores can be copied easily. Some core suppliers encrypt their cores and deliver special development tools which can handle encrypted cores. The disadvantage is that common tools cannot handle encrypted cores and that unlicenced cores could still be processed with cracked software.

Another approach to this problem is to embed a signature in the core, a so called watermark, which can be used as a proof of the original ownership. There are many concepts and approaches for the issue of implementing such a watermark inside a core. However, most of these concepts are not applicable due to the lack of verification capabilities. A good verification strategy makes sure that the signature (watermark) can be recovered using only the final product without the need to obtain extra files or information from the accused company.

One watermarking strategy for FPGA IP cores, called *power watermarking* [1], [2], adds a power signature generator to the core and verifies the watermark using power analysis. This can be done by measuring the core supply voltage of the FPGA. The voltage can be sampled with a digital storage oscilloscope, then analyzed and decoded on a PC to verify the signature. This method is non-destructive and can be applied using only the given product (see Fig. 1).

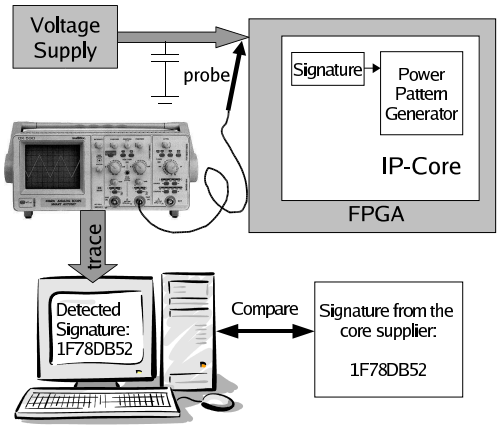

Fig. 1. Watermark verification using power signature analysis: From a signature (watermark), a power pattern inside the core will be generated that can be measured at the voltage supply pins of the FPGA. From the trace, a detection algorithm verifies the existence of the watermark.

Power watermarking adds protection to unencrypted IP cores on the netlist level. This allows for a more flexible application, including the combination with other purchased or self-written cores. However, if the FPGA embeds multiple watermarked cores, the different power watermarking signals superpose each other, which makes decoding harder or even impossible. The solution is to use multiplexing methods, as proposed in this paper, which enables to concurrently send different watermarks.

This paper is organized as follows: In Section II, a short overview of related work for IP watermarking is provided. Section III gives an overview of previously published power watermarking techniques and the different encoding/decoding schemes that enhance the robustness of the decoding. In Section IV, potential multiplexing techniques for the concurrent sending of watermarks from different cores are investigated. Section V presents experimental results and Section VI concludes the paper.

II. Related Work

Hiding a unique signature in user data such as pictures, video, audio, text, program code, or IP cores is called watermarking. Embedding a watermark into multimedia data is

achieved by altering the data slightly on a level where human senses have lower perception sensitivity. For example, one can remove frequencies which cannot be perceived by the human ear by encoding an audio sequence in the MP3 format. It is possible to hide a signature in these frequencies without decreasing the quality of the encoded audio sequence [3].

The watermarking of IP cores is different from multimedia watermarking, because the user data which represents the circuit must not be altered, since functional correctness must be preserved. Watermarking procedures can be categorized into two groups: *additive methods* and *constraint-based methods*.

In additive methods, the signature is added to the functional core, for example, by using unused lookup-tables in an FPGA [4]. The constraint-based methods were originally introduced in [5] and restrict the solution space of an optimization algorithm by setting additional constraints which are used to encode the signature.

The major drawback of these approaches are the limitations of the verification possibilities of the watermarked core embedded into a product. The bitfile of an FPGA can be extracted by wire tapping the communication between the PROM and the FPGA. Unfortunately, only the approaches presented in [4] and [6] have the possibility to detect the watermark from these bitfiles. An overview and evaluation of existing watermarking techniques is presented in [7] and [8].

Furthermore, most watermarking techniques concentrate on bitfiles or layout cores. Therefore, it would be useful to have different strategies for product verifiable watermarks, embedded in IP cores, and delivered on the HDL or netlist level.

The problem of applying watermarking techniques to FPGA designs does not lie in the coding and insertion of a watermark, but rather in the verification with an FPGA embedded in a system. Hence, our methods concentrate on the verification of watermarks. There are five potential sources of information: Bitfile, Ports, Power [9], [2], Electromagnetic (EM) radiation [10], and temperature [11]. The temperature approach for bitfile cores is the only watermarking technique which is commercially available.

III. Overview of the Power Watermarking Technique

There is no way to directly measure the relative power consumption of an FPGA, but only through measuring the relative supply voltage or current, or indirectly by measuring the temperature. We have decided to measure the core voltage as close as possible to the voltage supply pins, so that the smoothing from the plane and block capacitances are minimal and no shunt is required. Most FPGAs have *ball grid array* (BGA) packages and the majority of them have vias to the back of the PCB for the supply voltage pins. So, the voltage can be measured on the rear side of the PCB with an oscilloscope.

In the power watermarking approach described in [12] and [1], the amplitude of the interferences in the core voltage is altered. The basic idea is to add a power pattern generator (e.g., a set of shift registers) and clock it either with the operational clock or an integer division thereof. This power pattern generator is controlled according to the encoding of the signature sequence which should be sent.

The mapping of a signature bit sequence $s = \{0,1\}^n$ onto a sequence of symbols $\{\sigma_0, \sigma_1\}^n$ is called encoding: $\{0,1\}^n \rightarrow \mathcal{Z}^n, n \geq 0$ with the alphabet $\mathcal{Z} = \{\sigma_0, \sigma_1\}$ [2] . Here, each signature bit $\{0,1\}$ is assigned to a symbol. Each symbol σ_i is a triple $(e_i, \delta_i, \omega_i)$, with the *event* $e_i \in \{\gamma, \bar{\gamma}\}$, the *period length* $\delta_i > 0$, and the *number of repetitions* $\omega_i > 0$. The event γ is *power consumption through a shift operation* and the inverse event $\bar{\gamma}$ is *no power consumption*. The period length is given in terms of number of clock cycles. For example, the encoding through 32 shifts with the period length 1 (one shift operation per cycle) if the data bit '1' should be sent, and 32 cycles without a shift operation for the data bit '0' is defined by the alphabet $\mathcal{Z}_R = \{(\gamma, 1, 32), (\bar{\gamma}, 1, 32)\}$.

Different power watermarking encoding schemes were introduced and analyzed in [1] and [2]. This includes the basic method with encoding scheme: $\mathcal{Z}_0 = \{(\gamma, 1, 1), (\bar{\gamma}, 1, 1)\}$, the enhanced robustness encoding: $\mathcal{Z}_R = \{(\gamma, 1, 32), (\bar{\gamma}, 1, 32)\}$, the BPSK approach: $\mathcal{Z}_B = \{(\gamma, 1, \omega_i), (\bar{\gamma}, 1, \omega_i)\}$, and the correlation method with encoding $\mathcal{Z}_C = \{(\gamma, 25, 1), (\bar{\gamma}, 25, 1)\}$. To avoid interference from the operational logic in the measured voltage, the signature is only generated in the reset state of the core.

The power pattern generator consists of several shift registers, causing a recognizable signature- and encoding-dependent power consumption pattern. In some FPGA architectures (e.g., Xilinx Virtex), the lookup tables (LUTs) can also be used as shift registers [13]. A conversion of functional lookup tables into shift registers does not affect the core functionality if the new inputs are set correctly. This allows us to use functional logic for implementing the power pattern generator. The core operates in two modes, the *functional mode* and the *reset mode*. In the functional mode, the shift is disabled and the shift register operates as a normal lookup table. In the reset mode, the content is shifted according to the signature bits and consumes power which can be measured outside of the FPGA. To prevent the loss of the content of the lookup table, the output of the shift register is fed back to the input, such that the content is shifted circularly. When the core changes to functional mode, the content must be shifted to the proper position first in order to become a functional lookup table for the core.

The watermark embedding procedure is easy to use and consists of two steps only. First, the core has to be embedded into a wrapper, which contains the control logic for emitting the signature. This step is done at the HDL level and before synthesis. The second step happens at the netlist level after synthesis. A program converts suitable four input lookup tables (LUT4) into shift registers for the generation of the power pattern generator and attaches the corresponding control signal from the control logic in the wrapper.

The advantages of using the functional logic of the core as a shift register are the reduced resource overhead for watermarking and the robustness of this method against attacks. The most common attacks against watermarking are *removal, ambiguity*, and *copy attacks* [14]. It is hard, if not impossible,

to remove the shift registers without destroying the functional core, because they are embedded in the functional design. Even if an attacker identifies the sending logic, a deactivation is useless if the contents of the power shift registers are initialized shifted so that the core is unable to start if the signature was not transmitted correctly.

In case of *ambiguity attacks*, an attacker analyses the power consumption of the FPGA in order to implement a core whose power pattern disturbs the detection of the watermark. Such a *disturbance core* uses a lot of power to create pseudo-random noise to make the detection of the watermark impossible. By adding up the traces of several subsequent signatures one can eliminate this additional noise, as long as the attacker does not know the exact time difference between two signatures and adapts the disturbance core accordingly. Although a disturbance core might be successful, this core needs area and most notably power which increases the overall costs for the product. The presence of a disturbance core in a product is also suspicious and might lead to further investigation if a copyright infringement claim has arisen. Finally, the attacker may watermark another core with his watermark and claim that all cores belong to him. This can be prevented by adding a hash value to each original core and applying the multiplexing methods presented here. *Copy attacks* where a key from a credible author is copied and used for watermark a work of lower quality can be prevented by asymmetric cryptographic methods.

The advantage of power watermarking is that the signature can easily be read out from a given device. Only the core voltage of the FPGA must be measured and recorded. No bitfile is required which needs to be reverse-engineered. Also, these methods work for encrypted bitfiles where methods extracting the signature from the bitfile fail. Moreover, we are able to sign netlist cores, because our watermarking algorithm does not need any placement information.

IV. Multiplexing Methods

The power watermarking technique as introduced above is applicable to unencrypted netlist cores. Customers of netlist cores and product developers can combine different cores and integrate them into an FPGA design which is embedded into the product. Therefore, it is possible that more than one signatures from watermarked cores are present in the design. The different transmission mechanisms in different cores are not aware of each other and send their signature with the programmed encoding scheme, leading to superpositions and interferences which complicate and possibly even prohibit the decoding of the signatures.

In order to make the decoding of all signatures possible, our approach is to employ *multiplexing* or *multiple access* methods. Multiplexing means to divide the communication channel into multiple logical information channels – one channel for each transmitted signature. Substantially, there are four different categories of multiplexing methods: *Space Division Multiplexing* (SDM) as well as Multiplex in *Time* (TDM), *Frequency* (FDM), and *Code* (CDM).

A. Space Division Multiplexing

In *space division multiplexing*, the transport media for the different information channels are physically independent. For example, different isolated wires or antennas with directional radiation characteristics may be used.

For power watermarking, space division multiplexing could be implemented by measuring the voltage swing on different power pins. However, the power pins of an FPGA are usually interconnected, therefore only small amplitude differences can be measured when a power shift register is located close to that pin. Transmitted signatures measured at power pins which are near the shift register should have a higher amplitude than those at more distant power pins. If many watermarked cores are present, these are usually spread across the FPGA. Therefore, measurements on many power pins might successfully decode all signatures, even if they send simultaneous.

B. Time Division Multiplexing

When doing *time division multiplexing*, the communication channel is divided into several time slots, each forming a virtual channel. Each signal is split into blocks which are sent over the same communication medium in predefined time slots. There are two different categories of time division multiplexing techniques: *synchronous* (STDM) and *asynchronous time division multiplexing* (ATDM).

For STDM methods, each sender is assigned a fixed time slot. Furthermore, the definition of the time slot length and the assignment is done at design time and have to be known to all senders. If the assigned sender has no data to transmit, the corresponding time slot remains unused. In ATDM methods, the time slots are assigned dynamically. A sender only reserves a time slot if data will be sent and therefore the channel utilization is increased, compared to STDM methods. Since there is no fixed assignment of the slots, the demultiplexer has to know the receiver of the data. This is usually done by providing the receiver information over the channel, for example, in the header of the data. Therefore, these methods are also known as *address multiplexing methods*. The time slot length is either fixed or variable, depending on the used technique.

The problem of adapting either one of these TDM approaches for power watermarking is that the senders, i.e. the watermarked cores, have no synchronization possibilities except, maybe, the reset signal. However, the different cores may use different reset or clock signals and, as described above, due to power watermarking the reset state length might be altered further. Therefore, a probabilistic approach for time devision multiplexing is chosen instead, where each signature is sent repeatedly within a fixed time period. During one period, the signature is sent once and after that the power pattern generator inside the watermarked core is inactive until the period ends. The period length ϕ_i consists of the time for sending the signature $t_{sig,i}$ and the waiting time $t_{wait,i}$: $\phi_i = t_{sig,i} + t_{wait,i}$. By choosing different waiting times $t_{wait,i}$ for different cores, the sending times for each of the different signatures drifts away over time and the probability of a successful decoding increases with increasing measurement

time (see Fig. 2). One constraint for the usage of this method is that every watermarked core must have its own unique period length ϕ_i which should be relatively prime to the period lengths of the other cores in order to minimize the number of possible superpositions of different signatures. If S_{max} is the overall number of all existing power watermarked cores, then

$$\text{GCD}(\phi_i, \phi_j) = 1 \qquad \forall i, j \in \{1 \dots S_{max}\}.$$

If all cores begin transmission at the same start time, then the time span t_d until the first collision free decoding for two cores is (with $\phi_i < \phi_j$):

$$t_d = \phi_i \cdot \left\lceil \frac{t_{sig,i}}{\phi_j - \phi_i} \right\rceil,$$

and for three cores (with $\phi_i < \phi_j < \phi_k$):

$$t_d = \text{LCM}\left(\phi_i \cdot \left\lceil \frac{t_{sig,i}}{\phi_j - \phi_i} \right\rceil, \phi_i \cdot \left\lceil \frac{t_{sig,i}}{\phi_k - \phi_i} \right\rceil, \phi_j \cdot \left\lceil \frac{t_{sig,j}}{\phi_k - \phi_j} \right\rceil \right).$$

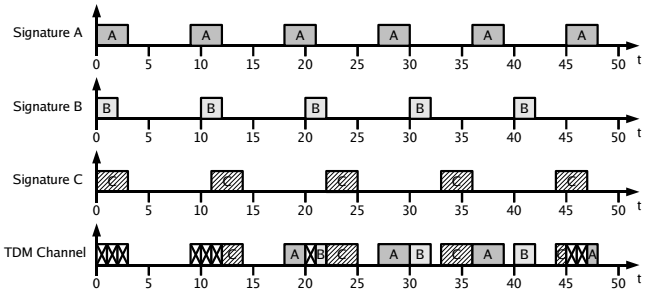

Fig. 2. A schematic example for the TDM method with transmission of three signatures with different lengths. The used periods are very short, leading to collisions marked on the TDM channel row below: $\phi_A = 9, \phi_B = 10$, and $\phi_C = 11$. The first collision free transmission of all signatures occurs at time step $t = 22$.

To estimate the period length for real systems, we must first know the time t_{sig} for sending the signature. For the correlation method ($\sigma_C = (e, 25, 1)$), the usual symbol length at 50 MHz is 500 ns. If we assume a signature length of 32 bit plus 12 bit preamble, the time for sending the signature only is $t_{sig,C} = 22\ \mu s$. Additionally, we must add the time needed to shift the power shift register back to the right position for the functional logic. For the enhanced robustness method ($\sigma_R = (e, 1, 32)$), the time for sending is $t_{sig,R} = \omega \cdot n \cdot f_{clk}^{-1}$ which results in 20.48 μs for sending the signature. Therefore, a value of 40 μs is a safe approximation. One exception is the BPSK method ($\sigma_B = (e, 1, \frac{f_{clk}}{10})$). Its sending time of a 32 bit signature is $t_{sig,B} = 205\ \mu s$.

On the other hand, the transmission of the watermarks is limited by the overall reset time, because all our power watermarking methods send the signature during the reset phase only. If we assume a reset time of $t_{rst} = 100\ ms$ and we would like to send each signature at least 20 times for better decoding, then the maximum repetition period is $\phi_0 = 5\ ms$. Thus, for each additional signature we reduce the maximum period length by the approximated 40 μs for sending the signature: $\phi_1 = 4.96\ ms, \phi_2 = 4.92\ ms, \dots$. Using this scheme, we have enough possible period times for signatures and still keep within a realistic reset time. Figure 3 shows a measurement of the transmission of three different signatures with our TDM method.

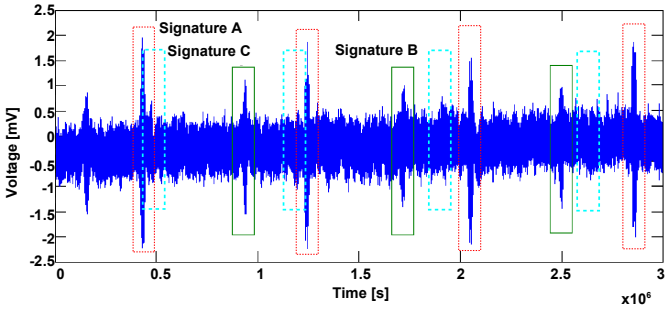

Fig. 3. This measurement shows the sending of three different signatures in TDM on a Spartan-3 FPGA. The minimal period difference between the signatures is approximate 16 μs. *Signature A* is produced by 64 SRL16 shift registers, *Signature B* uses 40 SRL16 shift registers, and *Signature C* uses 25 SRL16 shift registers. The measurement shows both collisions and proper decodeable signature transmissions.

C. Frequency Division Multiplexing

In *frequency division multiplexing* (FDM), the different information channels are assigned to different *carrier frequencies*. This can be done by modulating the information on a certain frequency. Different modulation methods, like *amplitude* (AM), *frequency* (FM), or *phase modulation* (PM) can be used. Furthermore, digital signals are often transmitted with *shift keying modulation*, for example, *Frequency Shift Keying* (FSK) or *Phase Shift Keying* (PSK).

The BPSK method (*Binary PSK*), introduced in [1], uses an on-off keying (OOK) modulation to shift the signature carrier frequency away from the clock frequency. Since most interferences from other cores are on the clock frequency or its divisions, this results in improved decoding. On different carrier frequencies, more than one core can simultaneously send its signature. The information of the signature bits is embedded into the carrier signal S_c in a BPSK-modulation. Each watermarked core has its own carrier signal $S_{c,i}$, which is OOK-modulated onto the clock frequency with different encoding schemes: $\mathcal{Z}_B = \{(\gamma, 1, \omega_i), (\bar{\gamma}, 1, \omega_i)\}$. By choosing different repetition rates ω_i, the different $S_{c,i}$ signals are sent on different frequencies, which enables a congruent sending of all signatures. However, the number of usable frequencies or repetition rates ω_i is limited by the clock frequency and the sending time of the signature. Nevertheless, this is an interesting approach for further research.

D. Code Division Multiplexing

In *code division multiplexing* methods, the transmitted data from different senders is spread up with different unique codes, so called *chips*. A chip is either able to encode one bit or a sequence of bits [15]. The chip encoded signals superpose on the communication channel. By knowing the chip sequences, all different transmitted data sequences can be reconstructed from the measured signal.

The most widely used code division method is based on encoding after *Hadamard and Walsh* [16]. The signals are mapped onto longer code sequences by multiplication with

mutually orthogonal chip sequences. A set of such chip sequences can be derived from a *Hadamard-Walsh matrix* [16]. The binary data sequence must be present in a *Non-Return-To-Zero* (NRTZ) code, which means that the two bits are encoded with '1' and '-1' and can cancel out each other. To reconstruct a signal, the measured superposed signal is multiplied again by the corresponding chip sequence and the total sum is calculated over the chip length. Figure 4 shows a simple example of transmitting two data sequences with a chip length of four. It is important for this method that all encoded data is synchronized to ensure a successful decoding.

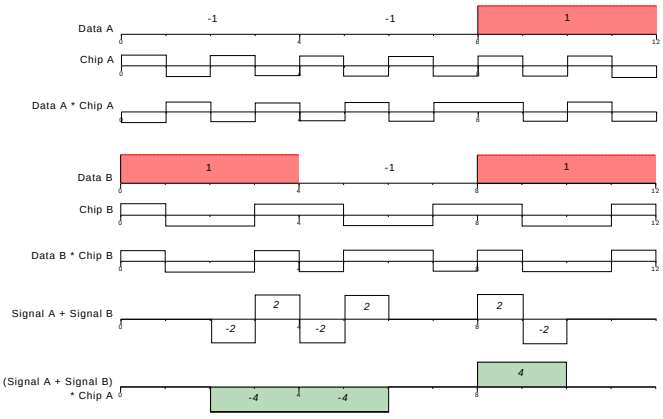

Fig. 4. Example of sending of two signals in CDM. The signals are spread with different chip sequences and superpose each other on the communication channel. By multiplying the recieved signal with the desired chip sequence, the signal can be reconstructed.

Experimental results have shown that this method is very sensitive to interference. Furthermore, because of the lack of NRTZ encoding, the different amplitudes for each core (due to different numbers of shift registers), and the lack of synchronization possibilities for power watermarking, Hadamard Walsh encoding does not seem to be applicable for concurrent sending of different signatures.

A more promising CDM approach uses chips which are adopted from optical transmissions, the so called *Optical Orthogonal Codes* (OOC) [17], [18]. The main characteristic of these codes is their low cross correlation and high auto correlation values, which make them robust against shifting and suitable for asynchronous CDM [19]. The codes consist of long runs of zeros separated by only few ones. Figure 5 shows an example of two OOC chips. By using these codes the superposed signatures can be reconstructed like the Hadamard-Walsh-codes.

The disadvantage is the long chip sequence, which results in much longer sending times for the signatures. Experimental results in Section V show promising results for the usage of CDM with OOC.

V. Experimental Results

In the following experiments, we used the same two FPGA-boards as in [1], the Digilent Spartan-3 Starter Board [20], and a board equipped with a Xilinx Virtex-II XC2V250 FPGA. On the second board many other components such as an ARM micro-controller and interface chips are integrated to

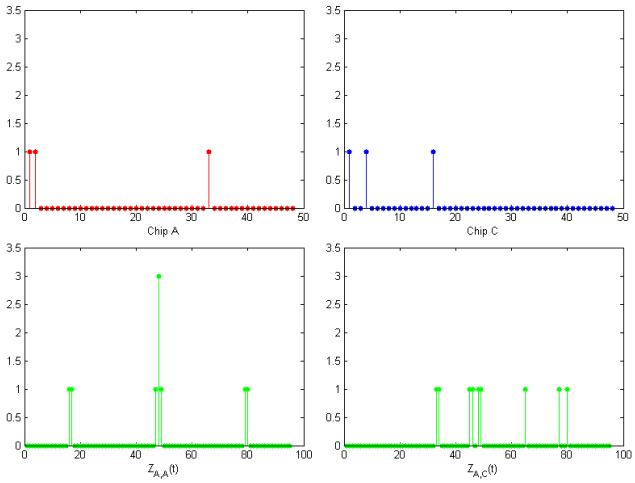

Fig. 5. Two chips A and C from a set of optical orthogonal codes with 3 ones over a chip length of 48. Each chip has an auto correlation value $Z_{A,A} = Z_{C,C} = 3$ and a minimal cross correlation value $Z_{A,C} = 1$.

demonstrate that the algorithm is also working on multi-chip boards. The Spartan-3 board operates at a clock frequency of 50 MHz, the Virtex-II board at 74.25 MHz.

On both boards, the voltage is measured on the back of the printed circuit board, directly on the via which connects the FPGA to the power plane of the printed circuit board. We used a 50 Ω wire with a 50 Ω terminating resistor soldered directly on the vias. We have used a *DC block* element and a 25 MHz high pass filter to filter out the DC component and the interferences of the switching voltage controller. We used a *LeCroy Wavepro 7300 oscilloscope* with 20 Giga Samples per second to measure the voltage. The voltage amplitude of the measured switch peak is very small, so we used a *digital enhanced resolution filter* to improve the dynamics, at the cost of a decreased bandwidth. The signal of the length of 200 μs is recorded on the internal hard disc of the oscilloscope. This trace file is then transferred to a personal computer and analyzed there.

For the concurrent sending of different signatures, we investigate *space division multiplexing* (SDM), *time division multiplexing* (TDM), and *code division multiplexing* (CDM) using *optical orthogonal codes*, introduced in Section IV. To obtain experimental results, we applied the correlation detection method. However, the other approaches might also be adapted for concurrent sending of different signatures. To receive realistic results we used the *des56*, the *3DES*, the *keyboard controller*, and the *i2c* core from *opencores.org* [21] for the experiments.

Our experimental results for SDM on a Spartan-3 have shown that the maximum amplitude difference between signatures at different power pins is less than 0.05 mV or 2% of the amplitude for a core with 512 shift registers. Therefore, using a SDM approach only is probably not applicable.

To evaluate TDM, we choose different signatures S_i and period times ϕ_i for each core and embedded the signatures with the corresponding sending logic into the cores. The decoding results in Table I show that the bit error rate is similar

to the correlation method without multiplexing. It is possible to use other encoding methods to enhance the detection rate, where we achieved a lower bit error rate.

TABLE I
DECODING RESULTS FOR TDM.

Core	Number of used SRLs	Signature s_i	period time ϕ_i	Bit Error Rate in % Dec. 1 pattern	Dec. 4 patterns
Des56	40	0x153CA9F8	460 μs	9.4	3.1
3DES	64	0x128E92C1	500 μs	9.4	3.1
kbd	18	0x928CB241	420 μs	18.3	6.2
i2c	25	0x74DE4FC1	380 μs	21.3	6.2

Finally, the results for optical orthogonal code multiplexing are shown in Table II. Here, we have used 48 bit chip sequences. This means that for transmitting one bit of the signature, it takes 48 times longer than in the original correlation detection method. In a set of chip sequences of length 48, only 8 *optical orthogonal sequences* exist, which means that we are only able to watermark 8 different cores. To get more optical orthogonal sequences, the chip length must be extended by 6 digits for each additional chip. This is difficult if every watermarked core should get a unique chip sequence to ensure correct decoding for all possible combinations of different watermarked cores. Nevertheless, the results show that it is possible to decode simultaneously transmitted signatures with this method. Note that the results are obtained from decoding only one pattern of each signature. By averaging many patterns to lower the noise, the results can be improved.

TABLE II
DECODING RESULTS FROM CDM USING OPTICAL ORTHOGONAL CODES.

Core	Number of used SRLs	Chip Sequence	Bit Error Rate in %
Des56	40	0xC00000004000	15.6
3DES	64	0xA00002000000	12.5
kbd	18	0x900100000000	6.25
i2c	25	0x884000000000	18.75

VI. CONCLUSION

In this paper we have shown how interfering power watermark signatures may be concurrently detected using multiplexing methods. When multiplexing is applied, several watermarked cores in an FPGA can transmit a signature at the same time and the proof of authorship for each single core is still possible. Out of the presented multiplexing methods, the most promising for use on an FPGA are time (TDM) and code (CDM) multiplexing. This was also shown by experiment. The quality of multiplexing by frequency (FDM) in BPSK is left for future research.

When designing a watermarking scheme for a range of cores, the parameters introduced in this paper may serve as a good starting base. Through a clever choice of parameters, every watermarked core in an FPGA or even in an ASIC can be identified concurrently. The proposed encoding schemes are quite robust against interference, so that it should be entirely possible to design a receiver hardware that can do the scanning and decoding without the need for other measurement appliances like a high resolution memory oscilloscope and establish a correct and reliable method for proofs of authorship for IP cores.

REFERENCES

[1] D. Ziener and J. Teich, "Power Signature Watermarking of IP Cores for FPGAs," *Journal of Signal Processing Systems*, vol. 51, no. 1, pp. 123–136, April 2008.

[2] D. Ziener, F. Baueregger, and J. Teich, "Using the Power Side Channel of FPGAs for Communication," in *Proceedings of the 18th Annual International IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM10)*, May 2010.

[3] L. Boney, A. H. Tewfik, and K. N. Hamdy, "Digital Watermarks for Audio Signals," in *International Conference on Multimedia Computing and Systems*, 1996, pp. 473–480.

[4] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "Signature Hiding Techniques for FPGA Intellectual Property Protection," in *proceedings of ICCAD*, 1998, pp. 186–189.

[5] Kahng, Lach, Mangione-Smith, Mantik, Markov, Potkonjak, Tucker, Wang, and Wolfe, "Constraint-Based Watermarking Techniques for Design IP Protection," *IEEETCAD: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, 2001.

[6] D. Ziener, S. Aßmus, and J. Teich, "Identifying FPGA IP-Cores based on Lookup Table Content Analysis," in *Proceedings of 16th International Conference on Field Programmable Logic and Applications*, Madrid, Spain, Aug. 2006, pp. 481–486.

[7] A. Abdel-Hamid, S. Tahar, and E. Aboulhamid, "A survey on IP watermarking techniques," *Design Automation for Embedded Systems*, vol. 9, no. 3, pp. 211–227, 2004.

[8] D. Ziener and J. Teich, "Evaluation of Watermarking methods for FPGA-based IP-cores," University of Erlangen-Nuremberg, Department of CS 12, Hardware-Software-Co-Design, Am Weichselgarten 3, D-91058 Erlangen, Germany, Tech. Rep. 01-2006, Mar. 2006.

[9] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," *Lecture Notes in Computer Science*, vol. 1666, pp. 388–397, 1999.

[10] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi, "The EM Side-Channel(s)," in *CHES '02: 4th International Workshop on Cryptographic Hardware and Embedded Systems*. London, UK: Springer-Verlag, 2003, pp. 29–45.

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

[12] D. Ziener and J. Teich, "FPGA Core Watermarking Based on Power Signature Analysis," in *Proceedings of IEEE International Conference on Field-Programmable Technology (FPT 2006)*, Bangkok, Thailand, Dec. 2006, pp. 205–212.

[13] Xilinx, Inc. Virtex-ii platform fpgas: Complete data sheet. ds031.pdf. [Online]. Available: direct.xilinx.com/bvdocs/publications

[14] M. Schmid, D. Ziener, and J. Teich, "Netlist-Level IP Protection by Watermarking for LUT-Based FPGAs," in *Proceedings of IEEE International Conference on Field-Programmable Technology (FPT 2008)*, Taipei, Taiwan, Dec. 2008, pp. 209–216.

[15] A. J. Viterbi, *CDMA: principles of spread spectrum communication*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 1995.

[16] A. Viterbi *et al.*, "Very low rate convolutional codes for maximum theoretical performance of spread-spectrum multiple-access channels," *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 4, pp. 641–649, 1990.

[17] J. Salehi, B. Res, and N. Morristown, "Code division multiple-access techniques in optical fiber networks. I. Fundamental principles," *IEEE Transactions on Communications*, vol. 37, no. 8, pp. 824–833, 1989.

[18] J. Salehi, C. Brackett, B. Res, and N. Morristown, "Code division multiple-access techniques in optical fiber networks. II. Systems performance analysis," *IEEE Transactions on Communications*, vol. 37, no. 8, pp. 834–842, 1989.

[19] F. Chung, J. Salehi, and V. Wei, "Optical orthogonal codes: Design, analysis and applications," *IEEE Transactions on Information theory*, vol. 35, no. 3, pp. 595–604, 1989.

[20] Digilent, Inc. Spartan-3 board. S3BOARD.cfm. [Online]. Available: www.digilentinc.com/info

[21] Opencores.org, "Opencores," *URL: www.opencores.org*.