PARTIAL-ENCRYPTION TECHNIQUE FOR INTELLECTUAL PROPERTY PROTECTION OF FPGA-BASED PRODUCTS

Kun-Wah Yip, *Member, IEEE*, and Tung-Sang Ng, *Senior Member, IEEE*
Department of Electrical and Electronic Engineering
The University of Hong Kong
Pokfulam Road, Hong Kong
E-mail: {kwyip, tsng}@eee.hku.hk

Abstract — **The configuration-data sequence of a field-programmable gate array (FPGA) is an intellectual property (IP) of the original designer. This paper proposes a partial-encryption (PE) technique for IP protection of configuration-data sequences by means of increasing the reverse-engineering cost. The PE technique encrypts a few selected data of the sequence. These data are selected in a judicious way such that, when a rival competitor copies the partially encrypted sequence into a cloned product, the cloned product performs the expected task to a certain degree of correctness but not absolutely error-free. Debugging is required. It is shown that, without an initial knowledge that a reverse-engineering countermeasure is employed, the PE technique outperforms the full-encryption technique in terms of the reverse-engineering cost. This paper describes implementation details of the proposed PE technique. Issues regarding system designs that embed hidden imperfections are also discussed.**

Index Terms — **Partial encryption, Intellectual property protection, Reverse engineering, Reverse-engineering countermeasure, Field-programmable gate arrays, SRAM-based FPGA, Obfuscation.**

I. INTRODUCTION

Static-random-access-memory-based (SRAM-based) field-programmable gate arrays (FPGAs) have become increasingly popular as building blocks of electronic systems because of advantages such as easy design modification, rapid prototyping, economical cost for low-volume production, lower startup cost which leads to lower financial risk in comparison to fully-customized application-specific integrated circuits (ASICs), availability of sophisticated design and debugging tools, and the ability of in-circuit reprogrammability. Tutorial overviews and architectural details of FPGAs can be found in a number of books and papers, such as [1]-[6]. Applications of FPGAs in the areas of consumer electronics include, for example, television circuits [7]-[11], communication and video-processing devices [12]-[18], and software-defined radios [19], [20].

To enable the FPGA to perform the task specified by the user, the user is required to design the configuration data according to the task. The configuration-data sequence is therefore an intellectual property (IP) owned by this user. This user is hereafter referred to as the original designer. The IP of the original designer can be protected against unauthorized use by means of patent and copyright practices. For advantages that are elaborated in Section II, original designers may want to achieve a higher degree of IP protection by preventing rival competitors from successfully reverse engineering their FPGA-based products. A direct approach is to encrypt the entire configuration-data sequence by an encryption algorithm. This approach is effective to prevent reverse engineering. However, we note that the reverse-engineering process is most often terminated at an early stage once an encrypted sequence is identified, so that the cost paid by rival competitors is not likely to be significant. Based on the law-enforcement philosophy that heavy penalty deters crime, one may think that boosting up the reverse-engineering cost is a good countermeasure against reverse engineering. This idea has been implemented by means of obfuscation techniques to hinder reverse engineering of integrated-circuit designs [21], patches containing security information for software maintenance [22], and Java codes [23], but has not yet been widely practiced in protecting configuration-data sequences. Using this idea, we propose a new reverse-engineering countermeasure for FPGA-based systems based on partial encryption (PE) of configuration-data sequences.

In Section II, we provide a literature review on existing approaches for IP protection of configuration-data sequences. Advantages of using reverse-engineering countermeasures to protect these IPs are detailed, and the

This work was supported by the Hong Kong Research Grants Council and by the University Research Committee of The University of Hong Kong, Hong Kong. Part of the materials of this paper have been applied for a patent.

Contributed Paper

Original manuscript received December 15, 1999

novelty of the PE technique is established. The PE technique is elaborated in Section III. Advantages of the PE technique beyond those of the full-encryption technique are discussed. It is also shown that successful applications of the PE technique require FPGA-system designs having hidden imperfections that increase the debugging complexity. Section IV provides implementation details of the PE technique. In Section V, we discuss issues relating to appropriate FPGA-system designs. Examples are also given for illustration. Finally, conclusions are given in Section VI.

II. Existing Methods for IP Protection of Configuration-Data Sequences

A SRAM-based FPGA is a one-chip programmable device comprising a number of input/output (I/O) peripheral cells, an array of user-configurable logic blocks, a network of interconnect resources (wire segments, crossbar switches, etc.), and an on-chip static random access memory (SRAM) [1]. Information necessary to configure logic blocks and to control crossbar switches that define the interconnect topology is contained in the configuration-data sequence, which is stored in the on-chip SRAM. Since SRAM is volatile, configuration data are lost upon removal of the FPGA power. It is necessary to download configuration data into the on-chip SRAM. In most FPGA-based systems, configuration data are located outside the FPGA and stored in nonvolatile memory such as EPROM (erasable programmable read-only memory). It is also possible that configuration data are loaded into target FPGAs through wireless communication links, such as in software-defined radios [24].

Suppose that a FPGA-based electronic product is designed and manufactured by an original designer. A rival competitor wishes to clone the original designer's product via reverse engineering in order to gain some business advantages such as lower design cost and faster to-market time. Interested readers may refer to [25] for general approaches in carrying out reverse engineering. In case configuration data are placed outside the FPGA, the rival competitor can easily retrieve the configuration data by reading the content of the nonvolatile storage device that hold these data, or by tapping the communication link that connects the storage device and the FPGA. IPs of the original designer can easily be copied to and exploited by the rival competitor without the need to obtain the original designer's approval.

In general, IP rights can be protected within the legal framework by patents and copyrights [26]. If the rival competitor's product is based on patented or copyrighted IPs of the original designer, and if the original designer can demonstrate that it is the case, the original designer can take appropriate legal actions against the rival competitor. Techniques that assist the original designer to protect his/her patented or copyrighted IPs are related to authorship identification of configuration-data sequences through the use of watermarking techniques [27]-[31]. Although patents and copyrights are effective means for IP protection, in many circumstances original designers may even want to prevent rival competitors from successfully reverse engineering their products at the very beginning. Such strategy offers the following advantages over IP protection solely based on legal means:

- protection of original designers' non-financial loss, in that cloned products appeared in the market may damage company reputation and customer relations;
- avoidance of possible failure to prove that rival competitors' cloned products are based on protected IPs of original designers;
- avoidance of high cost incurred in analyzing cloned products, gathering evidence, and taking legal proceedings against rival competitors; and
- protection of original designers' IPs during the time between patent filings and publications.

In particular, advantages of this strategy for IP protection of consumer products include:

- prevention of early appearance of cloned products so that original designers can maintain product uniqueness for as long as possible; and
- delaying the to-market time of cloned products so that their appeal to customers may be lost due to short life cycles of electronic products.

Previously published techniques that relate to security issues of FPGAs against reverse engineering include the following methods. The first method, detailed by Oldfield and Dorf in [1], comprises the steps of

(1) downloading the configuration data from an external source into the FPGA,
(2) removing the source from the circuit board on which the FPGA resides,
(3) turning off the read-back mode of the FPGA, so that the content in the on-chip SRAM cannot be retrieved from outside the FPGA, and
(4) always keeping the FPGA power on.

This method removes the need for an on-board configuration-data storage device and instructs the FPGA not to disclose the content of the on-chip SRAM. Security of the IP is therefore ensured. However, this method has the disadvantages that the circuit power has to be maintained throughout the operation life of the product and that reprogrammability of the FPGA has to be sacrificed. The second method, disclosed by Austin [32], is based on encryption of the entire configuration-data sequence by scrambling it with a pseudo-noise sequence generated by a feedback shift register with a given key. A special FPGA able to decrypt the sequence is used. The original configuration-data sequence is recovered inside the FPGA and then transferred to the on-chip SRAM for proper operation of the FPGA. A rival competitor cannot decode the encrypted sequence unless the key is obtained from the original designer, or, in rare case, correctly guessed. Note that the method of [32] can be easily extended to a general technique in which the entire sequence is encrypted by a sophisticated encryption algorithm (rather than scrambling).

However, to the authors' best knowledge, none of the published literature except the present work has reported an IP-protection technique that is based on encrypting part of the configuration-data sequence with an objective to increase the reverse-engineering cost.

III. The Partial-Encryption Technique

Unlike encrypting the entire configuration-data sequence, the PE technique encrypts only a few selected data of the sequence. These data are selected in a judicious way such that, when a rival competitor copies the partially encrypted sequence to a cloned product, the cloned product can perform the expected task to a certain degree of correctness but not absolutely error-free. Without prior approval to use the concerned IP from the original designer and additional information provided thereof, imperfections that exist in the unchecked cloned product prevents it from functioning error-free. The original designer has the full discretion to decide where to put these imperfections into the unchecked cloned product. Preferably these imperfections are placed where they are difficult to be located by the rival competitor. The unchecked cloned product, which bears hidden imperfections deliberately introduced, requires thorough debugging. Preferably this debugging process performed by the rival competitor requires considerable effort[1], thereby increasing the reverse-engineering cost and delaying the to-market time. It is possible that the debugging process is incomplete and unsuccessful, a malfunctioned cloned product may risk the rival competitor for (possibly serious) financial loss and a damage to company fame. Notice that successful applications of the PE technique require good FPGA-system designs that embed hidden imperfections with an intent to maximize the debugging complexity. Relevant system-design methodologies are the subject of discussion in Section V. Also note that a special FPGA is required by the original designer to decrypt the configuration-data sequence in order to remove/deactivate imperfections embedded in the design. This special FPGA is discussed in Section IV. A normal FPGA, which does not have the capability to remove the imperfections, is used by the rival competitor.

Acknowledged use of the PE technique, in which rival competitors are explicitly informed (or warned) that a reverse-engineering countermeasure is being used, makes rival competitors assess the need for a higher reverse-engineering cost and the risk of trading a possibly malfunctioned product. Maximizing this cost and risk deters rival competitors from reverse engineering original designers' products. IP rights of original designers are thus protected. *Uninformed* use of the PE technique, where rival competitors do not have a prior knowledge that a reverse-engineering countermeasure is being employed, is likely to encourage rival competitors to continue the reverse-engineering process because the first prototypes of cloned products seem almost functional. Until they realize at a later stage that the cloned products do not function as desired and that the complexity to be involved in debugging is expected to be exceedingly huge, rival competitors may have put considerable resources in performing reverse engineering. Protection of original designers' IPs is achieved by forcing a failure in reverse engineering.

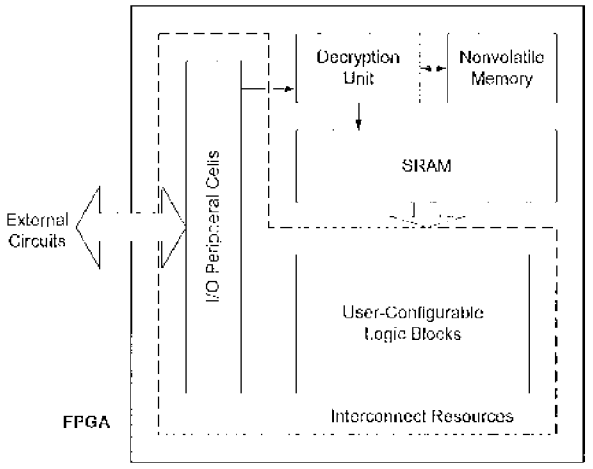

Fig. 1. A FPGA model that supports implementation of the PE technique.

The PE technique and the full-encryption technique (the technique that encrypts the entire configuration-data sequence) are compared as follows. In the acknowledged use of reverse-engineering countermeasure, the full-encryption technique prevents rival competitors from ever cloning the products while the PE technique deters them to do so. The full-encryption technique requires a good encryption algorithm, which is readily available. The PE technique, on the other hand, requires a good FPGA-system design bearing imperfections that are intended to maximize the debugging complexity experienced by rival competitors. Such design is not trivial. The PE technique therefore does not have an advantage over the full-encryption technique in the acknowledged use of reverse-engineering countermeasure. In the case of uninformed use, the PE technique does offer an important advantage in that the PE technique can make rival competitors incur a higher cost than they do when entire configuration-data sequences are encrypted. It is because in the latter case rival competitors can immediately identify that the sequences they obtain are encrypted ones[2] and hence abandon the reverse-engineering process at an early stage, whereas the use of the PE technique can stimulate rival competitors to put more efforts and resources to continue this process. A greater cost or a

[1] This is usually the case because debugging has a complexity similar to that of the fault-detection problem and it is known that fault detection is in general a NP-complete problem [34], [35].

[2] Direct transfer of a fully-encrypted sequence into a normal FPGA cannot make the cloned product functional, and the resultant system behavior is most likely chaotic. Chaotic behavior leaves a signature to the rival competitor that the sequence is likely to be (intentionally) corrupted in some way.

Position / Address	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Row 1	1	1	1	0	1	0	0	1	1	0	0	0	1	0	1	1	1	0	0	1
Row 2	1	1	1	0	1	0	1	1	1	0	0	0	1	1	0	0	1	0	0	1
Row 3	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	0	0	0	0

Keys: Row 1: Original configuration-data sequence
Row 2: Encrypted configuration-data sequence
Row 3: Secret sequence stored in the nonvolatile memory
Positions 7 and 14–16: Bits to be encrypted

Fig. 2. An example to illustrate design of the encrypted sequence and secret sequence for an original configuration-data sequence.

longer time to do reverse engineering that eventually yields nothing offers a greater protection of investment towards original designers in consumer-product design, manufacture and marketing.

IV. Required Hardware and Implementation Details

The PE technique encrypts part of the configuration-data sequence, and the encrypted sequence is placed outside the FPGA. When loaded into the FPGA, the encrypted sequence is decrypted by modifying some of its content based on the decryption information stored inside the FPGA. Currently available FPGAs do not support the PE technique due to lacking a decryption unit and a nonvolatile memory that stores the decryption information. A special FPGA similar to the one described in [32] is required. A FPGA model that incorporates an on-chip nonvolatile memory and a decryption unit together with standard components of a SRAM-based FPGA is shown in Fig. 1. For readers' interest, we mention that such a FPGA can be fabricated by current CMOS technology without considerable increase in the manufacturing cost due to inclusion of nonvolatile memory [33].

Prior to inserting this FPGA into a target electronic system, the on-chip nonvolatile memory is loaded with a secret sequence on which decryption of the encrypted sequence is based. Since this memory unit is nonvolatile, the secret sequence stored therein is not lost upon removal of the circuit power. The FPGA is then inserted into the target electronic system. When the power is up, or when the reset or reconfiguration mode is asserted, the encrypted sequence stored outside is loaded into the FPGA via I/O peripheral cells. The encrypted sequence is decrypted by the on-chip decryption unit based on the secret sequence. The decrypted sequence is subsequently loaded into the SRAM module for configuring the logic blocks and interconnect resources. The FPGA can then execute the desired operation.

A method to construct the encrypted sequence and the secret sequence is illustrated with the aid of an example given in Fig. 2. This method is not the only one method but is the simplest one, so it is considered here for illustrating the construction of these sequences. The original configuration-data sequence, shown in Row 1, is a string of binary-valued data each of which taking on a logic value

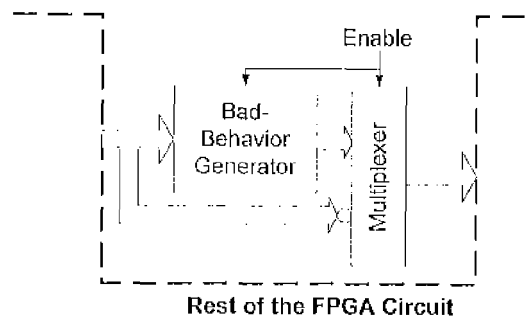

Fig. 3. The functional model of a FPGA-based system for IP protection using the PE technique.

from either 1 or 0. It is desired to encrypt data at positions 7 and 14-16 by toggling logic values of these data. The encrypted configuration-data sequence is shown in Row 2. To recover the original sequence in the FPGA, one constructs the secret sequence by assigning logic values 1's to data at positions 7 and 14-16, and 0's otherwise. The secret sequence is shown in Row 3. It is apparent that the original configuration-data sequence can be recovered by EXCLUSIVE-ORing the encrypted sequence with the secret sequence.

V. Relevant System-Design Methodologies

The original designer designs a FPGA-based system according to a given task with an additional objective to introduce hidden imperfections (or faults) that are difficult to be detected so that the debugging complexity experienced by the rival competitor is increased. Since it is demonstrated in Section III that the PE technique is most useful when the rival competitor does not have an initial knowledge that a reverse-engineering countermeasure is employed, we consider this situation. In the original designer's system and the cloned system, the task is performed, respectively, by FPGA(s) described in Section IV and normal FPGA(s). It is advantageous for the original designer to consider the following principles in designing FPGA systems.

1) The first prototype of a cloned system should appear to be seemingly functional (though not really is). That is, the behavior of the cloned system should not be made chaotic. It prevents the rival competitor from early discovery that the configuration-data sequence is

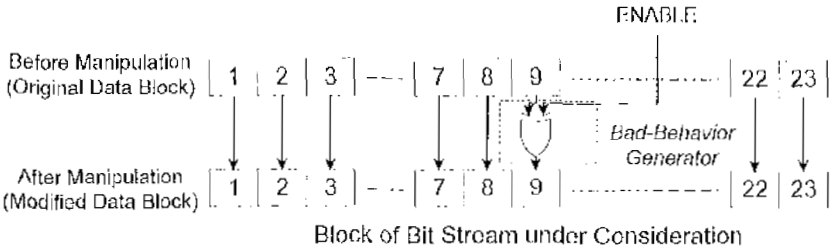

Fig. 4. A method to introduce errors into the incoming bit stream.

encrypted. Consequently, the rival competitor is encouraged not to abort the reverse-engineering process at an early stage, so that the resultant reverse-engineering cost is increased.

2) Imperfections that are intentionally introduced should be difficult to be detected but should have significant adverse consequences to the cloned system. It is preferred that these imperfections exhibit some forms of random behavior. That is, errors or undesirable effects pop up randomly in the cloned system. Random appearance of errors or undesirable effects also makes the rival competitor difficult to debug the cloned system. Thus, the debugging complexity is enhanced.

To enable original designers to design their FPGA-based systems that embed hidden imperfections, we propose a functional model depicted in Fig. 3 for modeling the original designer's system and the rival competitor's clone. The bad-behavior generator in the model is responsible to generate errors and undesirable effects. In the cloned system, the signal ENABLE is set to 1 so that the bad-behavior generator is activated. Errors or undesirable effects are introduced to the rest of the system. For the original designer's system, this signal is set to 0. The bad-behavior generator is deactivated and signals of the rest of the system bypass this generator. The original designer can encrypt the configuration-data sequence similar to the example given in Section IV, wherein data at the 7th position in Fig. 2 corresponds to the signal ENABLE.

System-design methodologies that enable effective use of the PE technique require ingenious placement of hidden imperfections into the designs. These methodologies are developed depending on the tasks to be performed, and are different for different situations. In the forthcoming discussion, we provide two examples for demonstrating applications of the PE technique. These examples are intended to provide readers an appreciation on how relevant system-design methodologies can be developed.

A. *Example 1: Reducing the error-protecting capabilities of error-correcting schemes*

Error-correcting techniques are normally used in communication systems to improve the system performance. An error-correcting code can be characterized by the

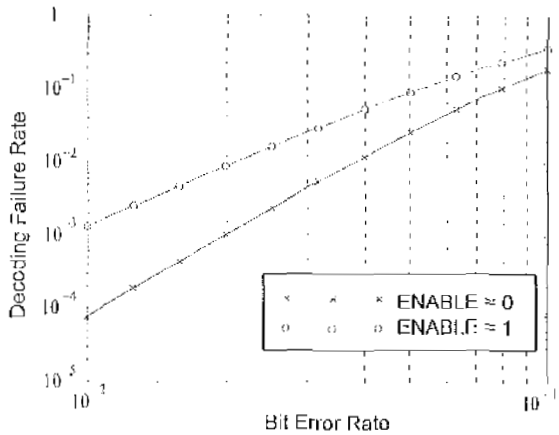

Fig. 5. Decoding failure rate for the Golay-code decoder (P_f) versus bit error probability of the incoming bit stream (P_b). Note: Original designer's product (ENABLE = 0); Cloned product (ENABLE = 1).

number of corrupted bits that can be corrected. If one deliberately introduces errors into the incoming bit stream, the resultant error density is increased so that the error-protecting capability of the error-correcting scheme is correspondingly reduced. It follows that the system performance is degraded, a desirable effect for the cloned system. It is preferred to reduce the error-protecting capability to a point where the overall performance of the cloned system falls below the product specification, causing the end user dissatisfaction in the performance.

We illustrate how errors can be added into the incoming bit stream by an example shown in Fig. 4. The coding scheme under consideration is the Golay (23,12) code [36] capable of correcting up to three errors for a block of 23 bits. In Fig. 4, an original data block of length 23 is processed by a bad-behavior generator, which is a simple EXCLUSIVE-OR gate, and the modified data block is sent to a Golay-code decoder. An error is intended to be introduced to the 9th bit. In the original designer's system, ENABLE is always set to 0 so that the original data block is directly sent to the decoder without modification. In the rival competitor's cloned system, ENABLE is set to 1. It follows that the resultant 9th bit is always different from the original one and an error is most probably introduced.

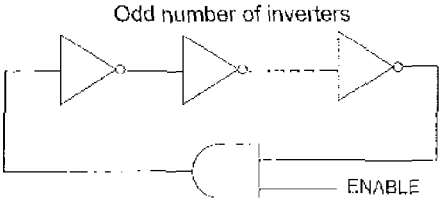

Fig. 6. Oscillator that can be built into a FPGA.

The reduction of the error-correcting capability is demonstrated by computing the decoding failure rate, which is the probability that decoding of a modified data block fails, that is, if the modified data block contains more than three errors. Fig. 5 plots P_f, the decoding failure rate, versus P_b, the bit error probability for each of the incoming 23 bits in the original block. Closed-form formulas[3] of P_f are easily derived [36]. Suppose that the decoder output is sent to a voice synthesizer and that a P_f of 10^{-3} or below gives a satisfactory voice quality. In the original designer's product (ENABLE = 0), it is apparent from Fig. 5 that the bit error probability of the incoming bit stream, P_b, to satisfy a requirement of $P_f = 10^{-3}$ is $P_b = 2 \times 10^{-2}$. If the cloned product (ENABLE = 1) also operates at this P_b value, the corresponding P_f becomes 9×10^{-3}, a nine-fold increase in the decoding failure rate when compared to the original designer's product. The resultant voice quality is less satisfactory and may become unacceptable.

This example satisfies the above-mentioned system-design principles 1) and 2) in the following aspects.

1) The first prototype of the cloned system appears to be functional. In fact, the reduction of error-protecting capability does not make the cloned system malfunction at once, but a degradation of system performance is introduced.
2) The observability of the system-performance degradation depends on the bit error probability of the incoming bit stream, and hence also depends on the signal-to-noise ratio (SNR). At high SNR, the performance degradation is less noticeable, but at low SNR, the degradation becomes noticeable and more significant. It follows that the undesirable effect (a degradation in the system performance) pops up selectively.
3) Depending on individual's reverse-engineering culture, a rival competitor at an early stage may test the first prototype of the cloned system by using a high SNR in order to establish its functionality. When the rival competitor notices that there is a performance degradation observed at a lower SNR, the belief that the cloned system is functional drives the rival competitor to initiate the debugging process in order to recover a loss in the performance. Hence, the cost spent on reverse engineering is increased.
4) Depending on individual's reverse-engineering culture, the rival competitor may debug the cloned system by focusing on the analog part and the analog-digital interface because faults are more likely to be found in these two parts than in the digital part. Placing hidden imperfections in the digital part intentionally confuses the rival competitor.
5) Noticeable adverse effect is introduced. It is intended to reduce the error performance of the cloned system to be below a certain level so that the resultant performance becomes unacceptable. It follows that the first prototype, although functional, is inferior to the original designer's system and hence difficult to have a market value.

B. *Example 2: Increased power consumption*

Fig. 6 shows a schematic diagram of an oscillator that can be implemented in a FPGA without the need for external components, such as capacitors. This oscillator has been used in [37] to experimentally characterize the relationship between the oscillation frequency and the temperature of a commercially available FPGA. Oscillation occurs only when ENABLE = 1. On-chip oscillation can be utilized to introduce undesirable effects to the cloned product by noting that, since the oscillation frequency can be made very high, a high current is drawn so that the power consumption is increased. However, in a consumer product that is realized by FPGA(s), low power consumption is very desirable. Turning on an unwanted oscillator inside a cloned product effectively makes it inferior to an original designer's product. As a demonstration, we experimentally studied this oscillator by using a commercially available FPGA[4] with ENABLE explicitly set to 0 or 1. Table 1 lists the values of oscillation frequency and drained current for the two cases of ENABLE settings. It is apparent that a sizeable amount of extra current is required when oscillation occurs.

We make the following remarks.

1) The functionality of the cloned system is not affected. Only a nuisance, viz., a higher power consumption, is introduced.
2) Depending on individual's reverse-engineering culture, the rival competitor may put more emphasis on the functionality of the cloned product. It follows that debugging for reducing the power consumption may be given a lower priority and could be performed at a later stage, though it is possible that high power consumption may have been observed very early. If the rival competitor eventually finds that it is not possible to reduce the power consumption, and if low power consumption is a necessary requirement of the product, the rival competitor may have to abandon the

[3] For ENABLE = 0, $P_f = 1 - (1 - P_b)^{23} - 23(1 - P_b)^{22}P_b - 253(1 - P_b)^{21}P_b^2 - 1771(1 - P_b)^{20}P_b^3$. For ENABLE = 1, $P_f = 1 - (1 - P_b)^{23} - 23(1 - P_b)^{22}P_b - 253(1 - P_b)^{21}P_b^2 - 231(1 - P_b)^{20}P_b^3 - 1540(1 - P_b)^{19}P_b^4$.

[4] Altera FLEX 10K100GC503-4 with a supply voltage 5V.

TABLE 1. OSCILLATION FREQUENCY AND DRAINED CURRENT FOR THE OSCILLATOR SHOWN IN FIG. 6.

Number of inverters in the oscillator	ENABLE	Oscillation frequency	Drained current of the FPGA
3	0	no oscillation	28.3 mA
	1	131 MHz	50.8 mA
5	0	no oscillation	28.3 mA
	1	61 MHz	44.4 mA
7	0	no oscillation	28.3 mA
	1	29.7 MHz	40.6 mA
9	0	no oscillation	28.3 mA
	1	19.7 MHz	38.3 mA

cloned product even though a lot of resources may have been involved in the reverse engineering.

3) Depending on individual's reverse-engineering culture, it is possible that the rival competitor gives precedence to examine the analog part rather than the digital part in debugging for power-consumption reduction. Putting this imperfection in a FPGA is an obfuscation tactic intending to hinder reverse engineering.

VI. CONCLUSIONS

We have proposed the PE technique that enhances the IP protection of FPGA-based systems when used with system designs that embed hidden imperfections for maximizing the debugging complexity. Effects of the use of the PE technique are summarized as follows:

- increasing the effort spent on reverse engineering so as to increase the reverse-engineering cost;
- delaying the to-market time of cloned products; and
- increasing the risk faced by rival competitors in trading malfunctioned products.

It has been shown that the PE technique outperforms the full-encryption technique as a reverse-engineering countermeasure when rival competitors do not have an initial knowledge that a reverse-engineering countermeasure is employed. Implementation details of the PE technique have been given. In particular, it has been shown that a FPGA that includes a nonvolatile memory and a decryption unit as well as components of a normal SRAM-based FPGA is required to use the PE technique. We have discussed issues relating to designing FPGA systems that embed hidden imperfections for the PE technique. Examples have been given to illustrate the design principles.

REFERENCES

[1] J. Oldfield and R. Dorf, *Field Programmable Gate Arrays*, Wiley, New York, 1995.

[2] R. J. Francis, "A tutorial on logic synthesis for lookup-table based FPGAs," *IEEE/ACM International Conference on Computer-Aided Design (ICCAD-92)*, pp. 40-47, 8-12 Nov. 1992.

[3] J. Rose, A. El Gamal and A. Sangiovanni-Vincentelli, "Architecture of field-programmable gate arrays," *Proceedings of IEEE*, vol. 81, pp. 1013-1029, Jul. 1993.

[4] J. Rose, "FPGA and CPGA architectures: a tutorial," *IEEE Design and Test of Computers*, vol. 13, Issue 2, pp. 42-57, Summer 1996.

[5] P. Chow, S. O. Seo, J. Rose, K. Chung, G. Páez-Monzón and I. Rahardja, "The design of an SRAM-based field-programmable gate array part I: architecture," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, pp. 191-197, Jun. 1999.

[6] , "The design of a SRAM-based field-programmable gate array - - part II: circuit design and layout," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, pp. 321-330, Sep. 1999.

[7] Q. Li, W. Li, X. Yuan, P. Xiao, J. Hu and L. Yang, "A TV graphics demonstration system design using FPGA," *Proceedings of 2nd International Conference on ASIC*, pp. 228-230, Oct. 21-24, 1996.

[8] S.-H. Kim, H.-S. Park, S.-Y. Ro, Y.-T. Lim, G.-C. Park and K.-D. Kim, "An optimum FIR filter implementation for color signal decimation in SD-DVCR system using FPGA," *Proceedings of 1997 IEEE International Conference on Consumer Electronics (ICCE'97)*, pp. 234-235, Jun. 11-13, 1997.

[9] K. Athikulwongse and E. Leelarasmee, "An FPGA based teletext inserter chip," *Proceedings of 1998 IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS 1998)*, pp. 751-754, Nov. 24-27, 1998.

[10] M. Hasan, M. S. Imam, R. Sharma and S. A. Abbasi, "A novel design and FPGA based implementation of a complex synchronization signal generator for television," *IEEE Transactions on Consumer Electronics*, vol. 43, pp. 1143-1152, Nov. 1997.

[11] H. Yang, Y. Zhong and L. Yang, "An FPGA prototype of a forward error correction (FEC) decoder for ATSC digital TV," *IEEE Transactions on Consumer Electronics*, vol. 45, pp. 387-395, May 1999.

[12] O. Meneer, M. Morf and M. J. Flynn, "Hardware software tri-design of encryption for mobile communication units," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3045-3048, 12-15 May 1998.

[13] K. Wong, M. Wark and E. Dawson, "A single-chip FPGA implementation of the data encryption standard (DES) algorithm," *Proceedings of IEEE GLOBECOM'98*, pp. 827-832, 8-12 Nov. 1998.

[14] G. L. Do and K. Feher, "Efficient filter design for IS-95 CDMA systems," *IEEE Transactions on Consumer Electronics*, vol. 42, pp. 1011-1020, Nov. 1996.

[15] S. K. S. Chan and V. C. M. Leung, "An FPGA receiver for CPSK spread spectrum signaling," *IEEE Transactions on Consumer Electronics*, vol. 45, pp. 181-191, Feb. 1999.

[16] O. Fatemi, F. Idris and S. Panchanathan, "FPGA implementation of the LRU algorithm for video compression," *IEEE Transactions on Consumer Electronics*, vol. 40, pp. 337-344, Aug. 1994.

[17] N. W. Bergmann and Y. Y. Chung, "Video compression with custom computers," *IEEE Transactions on Consumer Electronics*, vol. 43, pp. 925-933, Aug. 1997.
[18] B. Bosi, G. Bois and Y. Savaria, "Reconfigurable pipelined 2-D convolvers for fast digital signal processing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, pp. 299-308, Sep. 1999.
[19] M. Cummings, "FPGA in the software radio," *IEEE Communications Magazine*, pp. 108-112, Feb. 1999.
[20] G. C. Ahlquist, M. Rice and B. Nelson, "Error control coding in software radios: an FPGA approach," *IEEE Personal Communications*, vol. 6, pp. 35-39, Aug. 1999.
[21] F. S. Ozdemir, R. L. Seliger and G. B. Rosenberg, "Method and apparatus for securing integrated circuits from unauthorized copying and use," U. S. Pat. No. 4,766,516, Aug. 23, 1988.
[22] M. A. Bashar, G. Krishnan, M. G. Kuhn, E. H. Spafford and S. S. Wagstaff, Jr., "Low-threat security patches and tools," *1997 International Conference on Software Maintenance*, pp. 306-313, Oct. 1-3, 1997.
[23] C. Collberg, C. Thomborson and D. Low, "Breaking abstractions and unstructuring data structures," *1998 International Conference on Computer Languages*, pp. 28-38, May 14-16, 1998.
[24] W. H. W. Tuttlebee, "Software-defined radio: facets of a developing technology," *IEEE Personal Communications*, vol. 6, pp. 38-44, Apr. 1999.
[25] M. G. Reboff, Jr., "On reverse engineering," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, pp. 244-252, Mar./Apr. 1985.
[26] R. H. Stern, "Protecting industrial property rights," *IEEE Micro*, vol. 13, pp. 2-3, Oct. 1993.
[27] J. Lach, W. H. Mangione-Smith and M. Potkonjak, "FPGA fingerprinting techniques for protecting intellectual property," *1998 IEEE Custom Integrated Circuits Conference*, pp. 299-302, May 11-14, 1998.
[28] ———, "Signature hiding techniques for FPGA intellectual property protection," *1998 IEEE/ACM International Conference on Computer-Aided Design*, pp. 186-189, 8-12 Nov. 1998.
[29] ———, "Fingerprinting digital circuits on programmable hardware," *Information Hiding 1998, Lecture Notes in Computer Science*, vol. 1525, pp. 16-31, 1998.
[30] ———, "Robust FPGA intellectual property protection through multiple small watermarks," *Proceedings of 36th Design Automation Conference*, pp. 831-836, Jun. 21-25, 1999.
[31] I. Hong and M. Potkonjak, "Techniques for intellectual property protection of DSP designs," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASS'98)*, pp. 3133-3136, May 12-15, 1998.
[32] K. Austin, "Data security arrangements for semiconductor programmable devices," U. S. Pat. No. 5,388,157, Feb. 7, 1995.
[33] L. Chang, C. Kuo, C. Hu, A. Kalnitsky, A. Bergemont and P. Francis, "Non-volatile memory device with true CMOS compatibility," *IEE Electronics Letters*, vol. 35, pp. 1443-1445, Aug. 1999.
[34] O. H. Ibarra and S. K. Sahni, "Polynomially complete fault detection problems," *IEEE Transactions on Computers*, vol. 24, pp. 241-248, Mar. 1975.
[35] H. Fujiwara and S. Toida, "The complexity of fault detection problems for combinational logic circuits," *IEEE Transactions on Computers*, vol. 31, pp. 555-560, Jun. 1982.
[36] J. G. Proakis, *Digital Communications*, 3rd ed., McGraw-Hill: New York, 1995.
[37] S. Lopez-Buedo, J. Garrido and E. Boemo, "Thermal testing on programmable logic devices," *Proceedings of 1998 IEEE International Symposium on Circuits and Systems*, pp. II240-II243, 31 May - 3 Jun. 1998.

Kun-Wah Yip (M'96) received the B.Eng. (Hons) and Ph.D. degrees in electrical engineering from The University of Bradford, UK, in 1991 and The University of Hong Kong in 1995, respectively.

From 1995 to 1998, he was a research associate and then a postdoctoral fellow at The University of Hong Kong. Currently, he is a research assistant professor at the same university. His research interest is on spread-spectrum communications, multicarrier signal transmission, efficient simulation techniques, and recently on methods for combating against reverse engineering.

Tung-Sang Ng (S'74-M'78-SM'90) received the B.Sc.(Eng.) degree from the University of Hong Kong in 1972, and the M.Eng.Sc. and Ph.D. degrees from the University of Newcastle, Australia, in 1974 and 1977, respectively, all in electrical engineering.

He worked for BHP Steel International and The University of Wollongong, Australia, after graduation for 14 years and returned to Hong Kong in 1991, taking up the position of Professor and Chair of Electronic Engineering. His current research interests include mobile communication systems, spread spectrum techniques, CDMA and digital signal processing. He was the General Chair of ISCAS'97 and is currently VP-Region 10 of IEEE CAS Society. He is also an Executive Committee Member and a Board Member of the IEE Informatics Divisional Board. He has published over 170 international journal and conference papers. He is currently a Regional Editor of the International Journal – Engineering Applications of Artificial Intelligence (Pergamon Press).

He was awarded the Honorary Doctor of Engineering Degree by the University of Newcastle, Australia, in August 1997 for his services to higher education generally and to engineering education specifically.

Professor Ng is a Fellow of the IEE, HKIE, IEAust and a Senior Member of IEEE.