Indian J.Sci.Res. 17(2): 375 - 379, 2018 ISSN: 0976-2876 (Print)
ISSN: 2250-0138(Online)

A NOVEL APPROACH FOR FPGA CHIP IDENTIFICATION GENERATOR USING CONFIGURABLE RING OSCILLATORS

[1]K.Venkateswarlu, [2]S.Kirpalkour[3] , CH.Anil Kumar
[1,2,3]Electronics and Communication Engineering, Mahaveer Institute of Science and Technology, Bandlaguda, Hyderabad, T.S

Abstract-Physically unclonable functions (PUF) are commonly used in applications such as hardware security and intellectual property protection. Various PUF implementation techniques have been proposed to translate chip-specific variations into a unique binary string. It is difficult to maintain repeatability of chip ID generation, especially over a wide range of operating conditions. To address this problem, we propose utilizing configurable ring oscillators and an orthogonal re-initialization scheme to improve repeatability. An implementation on a Xilinx Spartan-3e field-programmable gate array was tested on nine different chips. Experimental results show that the bit flip rate is reduced from 1.5% to approximately 0 at a fixed supply voltage and room temperature. Over a 20^0C–80^0C temperature range and 25% variation in supply voltage, the bit flip rate is reduced from 1.56% to $3.125X10^{-7}$.

Keywords-Field-programmable gate array (FPGA), physically unclonable functions, ring oscillator.

I. Introduction

CHIP identification, in which unique binary strings are associated with integrated circuits of the same design, has a wide range of applications including digital intellectual property protection, integrated circuit counterfeit detection/prevention, and public-key cryptography. Field-programmable gate arrays

(FPGAs) are a mainstream hardware implementation platform, and need to be equipped with chip identification capabilities. Today's commercial FPGAs already contain such features.

For example, in Xilinx Virtex devices, a bitstream can be encrypted using a secret key.When the bitstream is downloaded, a hardware decryption core decrypts the bit stream. The bit stream only operates correctly if the device was programmed with the same key. This key is stored in RAM and it is not possible to read back the value [1]. Unfortunately, the chip identifier (ID) used for bitstream decoding is not available for other applications since this value cannot be read.

Xilinx also provides "Device DNA" in Spartan-3A series

FPGAs to protect designs from cloning, unauthorized overbuilding and reverse engineering. This feature is a unique factory set FPGA ID hardwired into the device which can be used to implement designs which only operate with a particular ID.

Instead of stored identification information, a physically unclonable function (PUF) utilizes physical variation to distinguish one chip from another. Using this concept, chip IDs can be obtained from mismatch in the delay, voltage or current values of an array of circuit structures of identical design. The random variation can be extracted, averaged and thresholded to produce a binary output. This technique can be applied to any FPGA, in contrast to "Device DNA" which is only implemented on certain FPGAs.

Chip IDs generated in this way should be *unique* and repeatable. Uniqueness is required to avoid ID collisions between devices, while repeatability is necessary to ensure that a given device returns the same value every time. We use the term*unstable* to describe a chip ID with low repeatability. Ring oscillators (ROs) are often used to generate PUF IDs. One common method is to use a cell consisting of two or more ROs. Due to transistor delay variations, a random output for cell, can be obtained from the difference in period of ROs with the same layout but different spatial locations. A binary output can then assigned depending on the sign of . We show experimentally that is normally distributed with an expected value, of 0 [2]. When is large, this scheme consistently gives the same output. Unfortunately, when it is small, the repeatability is compromised, particularly in the presence of temperature and supply voltage fluctuation. By using configurable ring oscillators and a run-time re-initialization scheme, the near-threshold residue values are eliminated.

This results in a change in the distribution of s from normal to a desirable bimodal one. After thresholding, the resulting IDs have very good statistical properties over a wide range of temperature and voltage and therefore, the reliability of chip ID generation is significantly improved.

The contributions of this work are summarized as follows.

A cell which uses a number of ring oscillators with slightly different, configurable delay paths. They are arranged in a spatially overlapped fashion, saving

[1]Corresponding Author

significant logic resources while maintaining good statistics for ID generation.

A power-up initialization and dynamic re-initialization process which selects and stores paths with the largest Re-initialization serves to improve repeatability in the presence of varying temperature and voltage.

Apost processing technique which generates a bimodal distribution for. This reduces the probability of its value being near the threshold and greatly improves the repeatability of the chip ID. The rest of this paper is organized as follows. Section II describes previous work on chip ID generation and physically unclonable functions (PUF). An analysis of the chip ID generation process is given in Section III. Then, we detail our proposed techniques in Section IV. Experimental results are presented in Section V. Finally, conclusions are drawn in Section VI.

II. Background

PUFs have drawn considerable attention from the hardware security research community since they were proposed in 2001 [3], [4]. Various PUF implementations on both application-specific integrated circuits (ASICs) and FPGAs have been reported. A summary of relevant works are given in the following subsections.

A. PUF on ASICs

Lofstrom *et al.* [5] used an array of addressable NMOS transistors loaded with a common resistive load. Drain current mismatch caused the voltage across the load to be different for different transistors in the array. By addressing the transistors in the array sequentially, a sequence of voltages was generated and successive values converted to a binary sequence via an auto-zeroing comparator to form an ID. A 112-bit ID circuit was shown to have a drift of less than 4% over a wide supply voltage and temperature range. Su *et al.* [6]. reported on an improved circuit which used cross-coupled logic gates to simultaneously generate, amplify and digitize transistor mismatch. This circuit was able to produce a 128-bit, 96% stable ID using only 1.6 pJ/bit. Helinski *et al.* proposed another PUF design based on measured equivalent resistance variations in the power distribution system of an IC [7].

B. PUF on FPGAs

FPGA-based PUF implementations can be categorized into the following types: memory-based, logic-based, arbiter-based, and ring oscillator (RO)-based,

1) Memory-Based PUF*:* Guajardo *et al.* utilized the initialization state of static RAM cells in an FPGA and showed that they had suitable statistical properties for producing an ID [8], [9]. His experiments showed that 4% of the startup bits from the same RAM changed over time. Over a 20 C to 80 C temperature range, bit strings had a maximal fractional Hamming distance of 12% compared to a reference at 20 C. Holcomb *et al.* [10] proposed a Fingerprint Extraction and Random Numbers in SRAM (FERNS) extraction system that harvests static identity and randomness from existing volatile CMOS memory without requiring any dedicated circuitry.

2) Logic-Based PUF: Patel et al.count variation-dependent glitches on the output of a combinational multiplier to generate unique identification [11]. They found that 6 out of 64 bits are changed over a range of temperature. Anderson used an FPGA's carry chain to implement a PUF [12]. On average, 3.6% of bits are changed in high temperature.

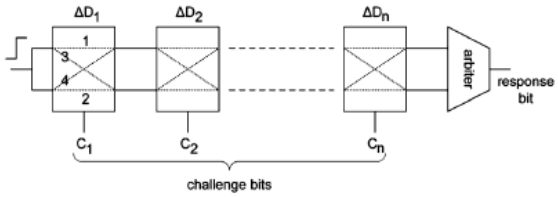

Figure 1:Arbiter Based PUF

3) Arbiter-Based PUF*:* Fig. 1 shows an arbiter PUF, comprising two parallel n-stage multiplexer chains feeding a flipflop. A transition is input to the arbiter which travels through a series of 2-input/2-output switches. Each switch is configured to be either a cross or a straight connection based on its selection bit. The arbiter compares the arrival times of its two inputs and generates a response bit. The path segments are designed to have the same nominal delays but their actual delays differ due to process variation. The difference between the top and bottom path delays on the segment is denoted by in Fig. 1. The PUF challenges are the selector bits of the switches. The output of the arbiter is a function of the challenge bits and different for different chips. Suh and Devadas [13] generated binary outputs from a difference

in path pair delays. This technique achieved a 0.7% unstable bit rate at room temperature and fixed supply voltage. It remained less than 9% when temperature was increased by 100C and voltage varied by 33%. The fractional Hamming distance achieved was 23% of the total bit width, whereas an ideal value is 50%. Majzoobi *et al.* [14] proposed an improved arbiter-based PUF which utilized multiple delay lines for each response bit, transformations and combinations of the challenge bits and combination of the outputs from multiple delay lines. This scheme achieved lower predictability and higher resilience against circuit faults, reverse engineering and other security attacks.

4) RO-Based PUF*:* A RO-based PUF uses differences in period between similar ROs. The RO is typically encapsulated in a hard macro with fixed layout, and

arranged in different spatial locations on the FPGA. Since the logic cells and routing are identical, the same nominal value of loop delay is achieved. Suh and Devadas [13] compared Arbiter and RO based PUFs and found the latter achieved better performance. Ring oscillators with vastly different periods were used to improve the robustness of the generated ID. In particular, a 1-out-of- masking scheme with so to generate bits, ROs are required. For each of pairs, the pair with maximum distance was chosen, and a bit vector of these selections is saved so that the same pairs can be used to regenerate the output. Experimental results show the intra-chip variation was 0.48% for temperatures from 20 C to 120 C and voltages from 1.2 to 1.08 V. Maiti and Schaumont [15] proposed a configurable ring oscillator to achieve a higher reliability in an RO-based PUF. Compared to 1-out-of-8 scheme used in [13], this approach was more efficient in terms of hardware cost and ROs are required to generate bits.

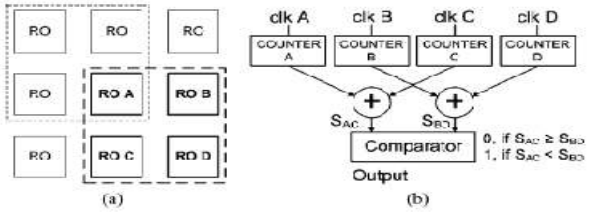

Figure 2:1 Bit ID Generation(a)block diagram of a cell(b)1 bit ID Configuration

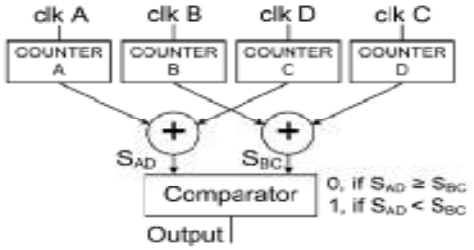

Figure 3: Distribution over all chips and spatial positions

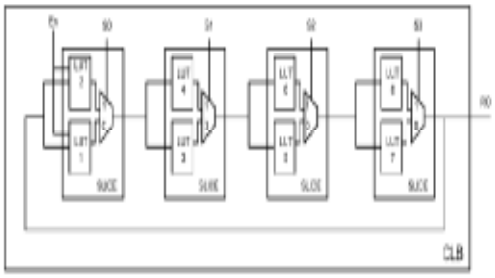

Figure 4: circuit for configurable RO

III Implementation

A. Architecture

Fig. 4 illustrates the architecture of our chip ID generator design. It includes a 9×9 RO array providing 8×8 cells. This can generate 64 separate bits $(i = 0, \ldots, 63)$.

The address generator together with the two decoders select a single RO to operate over a given time interval. A 4-bit global RO configuration signal, detailed in the next subsection, is also sent to each RO. At any given time only one RO can be activated and hence the configuration only affects the operating RO. Two levels of multiplexors are used to route the output of the selected RO to the counter. Handshaking signals connect the timer to the the ARM processor and the residue is calculated in software according to (2).

To facilitate different experiments with the ID generator, postprocessing is implemented on an external ARM processor in software. The postprocessing could also be included in an on-FPGA processor or finite-state machine.

B. Configurable RO

The circuit implementation of the configurable RO is shown in Fig. 5. In this work, a Xilinx Spartan-3e was used to demonstrate the technique. The design could be easily ported to different FPGA families. A four-stage RO is used where three of the stages are non-inverting and the final one is inverting. Each occupies two Xilinx logic elements (LEs) within a slice and a multiplexer is used to choose the signal path. The entire RO occupies a single Xilinx configurable logic block (CLB). By selecting different values of $S_0 - S_3$, 16 different configurations can be chosen. Logic and interconnect delay mismatch in the paths of the different configurations change the frequency of the RO.

Fig. 6 shows the variation in counter values, N_{RO}, for the four ROs of a cell as a function of the configuration value. As one would expect, there is systematic variation as the configuration is changed. Table I summarizes the wiring delay of each path of the four stages. It can be seen that, particularly for stage-1 and stage-3, there are significant differences which account for the correlations. As an example, from Table I, comparing configuration "0000" to "0010", the stage-1 delay is increased, reducing the RO frequency and counter value.

Due to these expected systematic variations, generating R_i using different configurations leads to correlated outputs. Instead, we use the same configuration for all four ROs in a cell, and choose the one with the largest $|R_i|$. This technique employs

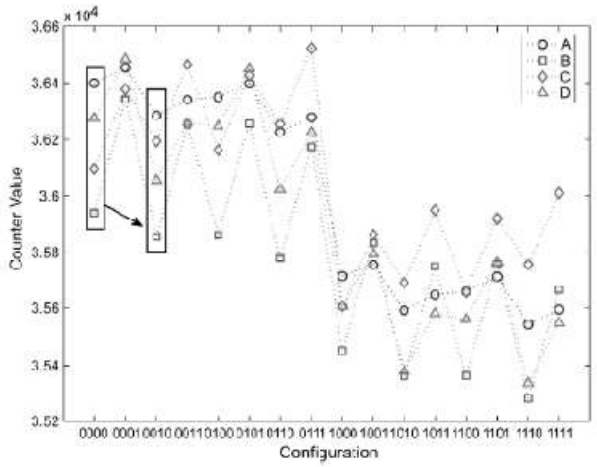

Fig. 6. Counter values of RO components within a particular configurable cell.

TABLE I
WIRE DELAYS FOR DIFFERENT PATHS WITHIN A CONFIGURABLE RO EXTRACTED FROM CAD TOOLS

s	stg 0	stg 1	stg 2	stg 3
0	0.042ns	0.175ns	0.042ns	0.091ns
1	0.045ns	0.216ns	0.042ns	0.131ns

one configurable RO to achieve a similar result to choosing from 16 normal ROs as done in Suh and Devadas's work [13], resulting in a reduction in area.

C. Configuration Initialization

1) Power-Up Initialization: Our proposed method requires a set of configurations to generate stable IDs so a scheme is required to initialize them upon power-up. One straightforward approach is to determine configurations when the FPGA is powered up the first time, and store them in non-volatile memory or on an authorized server. Such an approach would also need to carefully consider the possibility of information leakage and susceptibility to modelling attacks [20].

When the chip is subsequently powered up, configurations are transferred to the chip for ID generation. Unfortunately, for this scenario, a communication channel is required, even though the configuration does not reveal relative speeds of the ROs, it leaks information. For instance, if the same configuration is used for overlapping cells, an adversary may be able to infer a dependency between the ROs involved. Moreover, if an adversary can

IV. Results

A. Summary of Hardware Resource Consumption

We implemented the system on a custom board with a Xilinx Spartan-3e FPGA (*xc3s250e-4pq208*) and a NXP LPC2131 ARM processor. Xilinx ISE Design Suite 12.1 and Vision v3.62c are respectively used for FPGA design and ARM C compilation. Nine identical boards are tested, each of them assembled with identical devices. As shown in Table II, the resource consumption, even on a small FPGA, is minimal.

TABLE II
LOGIC UTILIZATION

Resource	Consumption	Total	Percentage
Number of Slice Flip Flops	65	4,896	1%
Number of 4 input LUTs	772	4,896	15%
Number of BUFGMUXs	2	24	8%
Number of DCMs	1	4	25%

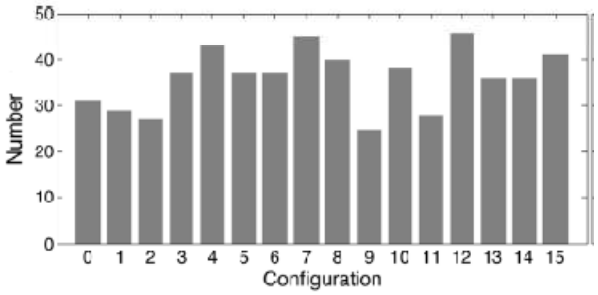

Fig. 13. Histogram of configurations selected over all cells and all chips.

TABLE III
ONE/ZERO RATIO

Chip No.	1/0 ratio	Chip No.	1/0 ratio
1	0.88	6	1.06
2	1.06	7	0.94
3	1.00	8	1.13
4	1.20	9	1.13
5	1.13		

TABLE IV
HAMMING DISTANCE MATRIX

	1	2	3	4	5	6	7	8	9
1	0	31	28	29	30	35	33	32	28
2	31	0	27	28	29	28	22	29	23
3	28	27	0	35	32	35	27	34	26
4	29	28	35	0	33	24	34	35	31
5	30	29	32	33	0	25	37	30	32
6	35	28	35	24	25	0	28	33	31
7	33	22	27	34	37	28	0	25	29
8	32	29	34	35	30	33	25	0	34
9	28	23	26	31	32	31	29	34	0

B. Statistical Analysis

1) Cell Configurations*:* Fig. 13 shows the distribution of selected configurations over all cells and all tested chips. Although the distribution is not uniform, strong biases towards some particular configurations were not evident.

2) One/Zero Ratio*:* The measured one-to-zero ratios of the 9 chips are listed in Table III. On average, the one/zero ratio is 1.06, confirming an equal likelihood for each value.

3) Hamming Distance*:* The Hamming distances between all pairs of chip IDs are summarized in Table IV. The average value is 30, which is 47% of the bit width. This is very close to the ideal of 50% for independent IDs. Taken together, the Hamming distance analysis and one/zero ratio demonstrate the generation scheme has very good statistical properties. It shows that the configuration selection scheme

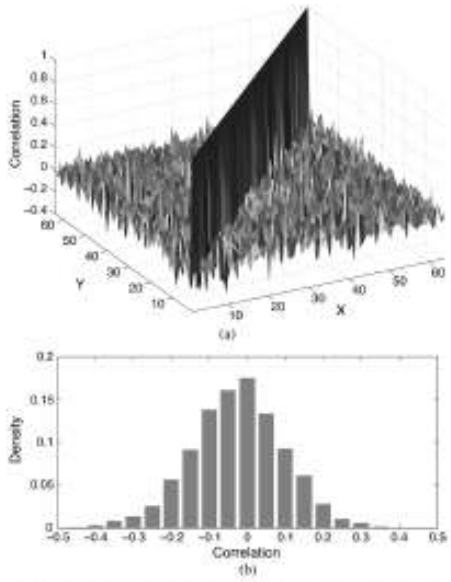

Fig. 14. R_i correlation analysis. (a) R_i correlation matrix. (b) Histogram of correlation between different R_i s.

and overlapped cell composition, do not come at the cost of reduced randomness. In fact, the statistical properties are improved compared with our previous work [2].

4) Correlation Analysis: Fig. 14(a) illustrates correlation between different R_is. As each R_i is fully correlated with itself, the diagonal values are equal to 1. A histogram of the distribution is shown in Fig. 14(b). On average, the correlation between different R_is is -4×10^{-3} and 90% of the correlations are in the range -0.2 to 0.2 with a maximum absolute value of 0.44. From this analysis we conclude that there was no evident correlation between bits in the ID generation process.

5) Stability Analysis: Fig. 15 shows the $E(R_i)$ distribution over all chips and all cells. Compared to Fig. 3, the configuration scheme modifies the distribution from Gaussian to a bimodal one and the occurrence of residues whose absolute values are close to zero are eliminated.

A bit flip occurs when a cell generates an R_i with sign opposite of the mean value in Fig. 16. We define the "*bit flip rate*" P_{bf} as the number of occurrences of bit flips N_{bf} divided by the total number of bits generated N_{all}

$$P_{bf} = \frac{N_{bf}}{N_{all}}. \tag{4}$$

For ease of analysis and expression, we modify all of the residues R_i to be positive

$$\tilde{R}_i = \begin{cases} -R_i, & \text{if } E(R_i) < 0 \\ R_i, & \text{if } E(R_i) > 0. \end{cases} \tag{5}$$

V. Conclusion

We have demonstrated that a chip ID generation method with configurable RO, power-up initialization and adaptive re-initialization can considerably improve its repeatability. Results show that a very stable ID generation can be achieved over a wide range of operating conditions.

Since our design was completely implemented using standard digital circuits, it can also be implemented in an ASIC. As future work, the authors would like to: develop more parallel generation schemes to speed up chip ID generation; introduce countermeasures to machine-learning and side-channel attacks; and study the security implications of storing configuration information on an off-chip server.

References

[1] A. Telikepalli, "Is your FPGA design secure?," *Xilinx XCELL*, pp. 25–32, Fall, 2003.

[2] H. Yu, P. Leong, H. Hinkelmann, L. Moller, M. Glesner, and P. Zipf, "Towards a unique FPGA-based identification circuit using process variations," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 2009, pp. 397–402.

[3] R. Pappu, "Physical one-way functions" Ph.D. dissertation, Program in Media Arts Sci., Sch. Arch. Planning, Massachusetts Inst. Technol., Cambridge, 2001. [Online]. Available: http://pubs.media.mit.edu/pubs/papers/01.03.pappu phd. owf.pdf

[4] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science* vol. 297, no. 5589, pp.2026–2030, 2002. [Online]. Available: http://www.sciencemag.org/content/297/5589/2026. abst act

[5] K. Lofstrom, W. R. Daasch, and D. Taylor, "IC identification circuit using device mismatch," in *Proc. Int. Solid-State Circuits Conf.(ISSCC)*, 2000, pp. 372–373.

[6] Y. Su, J. Holleman, and B. Otis, "A digital 1.6 pJ/bit chip identification circuit using process variations," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 69–77, Jan. 2008.

[7] R. Helinski, D. Acharyya, and J. Plusquellic, "A physical unclonable function defined using power distribution system equivalent resistance variations," in *Proc. 46th Annu. Design Autom. Conf. (DAC)*, 2009, pp. 676–681.

[8] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in *Proc. 9th Int. Workshop Cryptograph.Hardw. Embed. Syst. (CHES)* , 2007, pp. 63–80.

[9] J. Guajardo, S.Kumar,G.-J. Schrijen, and P. Tuyls, "Physical unclonable functions and public-key crypto for FPGA IP protection," in *Proc.Int. Conf. Field Program. Logic Appl. (FPL)*, 2007, pp. 189–195.