Exploiting Emerging Reconfigurable Technologies for Secure Devices

Ansh Rupani, Shubham Rai, Akash Kumar
Chair For Processor Design, CfAED, Technische Universität Dresden, Dresden, Germany
<firstname.lastname>@tu-dresden.de

***Abstract*—In the present work, we show how new and emerging reconfigurable technologies provide promising improvement over CMOS in the field of hardware security and encryption. We demonstrate how security features are a natural outcome of the circuits based on Silicon Nanowire reconfigurable transistors. This forms the basis of authentication key based security technique. Using the authentication key based system, we obtained the maximum possible key-length for MCNC benchmark circuits. Further, we formulated security as a tunable aspect for a circuit, by introducing don't care adjustment. A combination of the above two is used to establish security in terms of Shannon's entropy. We show that using the above concepts, Shannon's entropy increases for 99.1% benchmarks out of which maximum entropy is reached for 38.5% of all the benchmarks. We demonstrate these concepts using a case study for a 2-bit Ripple Carry Adder (RCA) based on SiNW RFETs and compare the design with its CMOS counterpart.**

I. Introduction

Hardware security plays an ever important role in today's era, owing to the growing cases of IC counterfeiting, side-channel attacks, IP piracy etc. Various components involved in the manufacturing of a particular IC are spread all throughout the world and this makes the ICs more prone to such attacks at various levels [16] [14]. IC piracy has become a great threat to the industry [18]. According to a report [17], the IC industry, on an average, loses about $4 billion annually due to IP infringement.

Recent developments in emerging technologies have opened up new avenues to bring security from the technology side within the hardware system. Silicon Nanowire (SiNW) based RFETs [7] offer extended functionality which can be tapped to provide inherent security features at the logic level [1]. In [11], Rai et.al. demonstrated logic gates which can exhibit multiple functionalities within the same structure. Such logic gates can hide functionalities and can prove to be a suitable candidate for making polymorphic logic gates [9] for embedding security at the device level.

Various emerging technologies have been explored in works like [4] and [12] to make hardware systems inherently secure. However, the ideas laid in these papers were just at the conceptual level. No design methodology was provided for larger circuits using emerging technologies. In the present work, we present a study on emerging technologies for security purposes.

Contributions: Major contributions of the present work are–

- By using input pattern at the program gate of a silicon nanowire RFET, we devised a sequence of bits to form an authentication key for any circuit. This authentication key is like a password for the hardware circuit. We have used this sequence for a benchmark suite and generated maximum possible keylength for MCNC benchmarks.
- We introduced *Don't Care adjustments* as a tunable knob for security aspect for a circuit using unique properties of reconfigurable transistors.
- Using a 2-bit RCA as a case study, we demonstrate the above concepts and compared the SiNW and CMOS based implementation of the circuit.

We evaluate the security aspect of a circuit based on Shannon's entropy using a combination of authentication keys and don't care adjustment technique. The entropy is increased for 99.1% of the benchmarks and amongst those, in 38.5% of the cases, we have achieved the maximum entropy.

The paper is organized in five sections. Section II describes the background and motivation of using reconfigurable technologies for hardware security. Section III explains various concepts that are analyzed in our study. Section IV summarizes the experimentation and discussion, followed by the Section V, which is a case study of a 2-bit ripple-carry adder. Our concluding remarks come in Section VI.

II. Background and Motivation

In this section, we summarize the previous works and discuss how those works have motivated our present work.

A. Reconfigurable Transistors

Reconfigurable technology is exhibited by transistors made with materials like silicon nanowires [7], carbon nanotubes [8], graphene nanoribbons [6] etc. These transistors are ambipolar i.e. they can show p-type or n-type behaviour within the same device. In the present work, we will focus on SiNW based reconfigurable technology. Silicon Nanowire RFETs have two gate terminals in contrast to the CMOS technology. The **control gate** receives the normal input to the transistor which controls the creation of a carrier channel. The **program gate** (PG) decides the direction of the carrier types. SiNW RFETs even exhibit low leakage power as compared to CMOS, owing to the Schottky metal contacts. Results have shown that SiNW transistors can drastically reduce the transistor count in circuits when compared to CMOS [11].

B. Hardware Security

In [9], the concept of polymorphic gates has been demonstrated for CMOS, where various gates in a netlist are randomly chosen and replaced by polymorphic gates, which are activated for a particular function only when a predefined key is provided. Another form of a polymorphic logic gate has been defined in [19]. In this work, Simek et al. have devised a polymorphic gate which can behave as a NAND gate or a NOR gate depending on the applied voltage potential V_{DD}. However, all these concepts were feasible only with a huge area and cost overhead in CMOS technology. The exciting characteristics of novel SiNW RFETs directly fit into the requirement for hardware security.

In [2], the authors suggested camouflaging layout and polymorphic gates to help obfuscate layout and netlists using emerging reconfigurable nanotechnologies. They suggested keys to differentiate between actual and dummy functionalities using logic gates design mentioned in [5].

While previous works have introduced the concept of keys, the discussion was restricted only till the logic gate level. To leverage the inherent functional polymorphism in emerging reconfigurable transistors, it is imperative to carry out formal study to evaluate these concepts on a wider spectrum of benchmarks and use metrics like Shannon's entropy to establish the efficacy of such concepts. In the present work, we have focused on this aspect and also explored using hardware primitives like *Don't Care* adjustments and evaluate how all these concepts can help in making more secure systems.

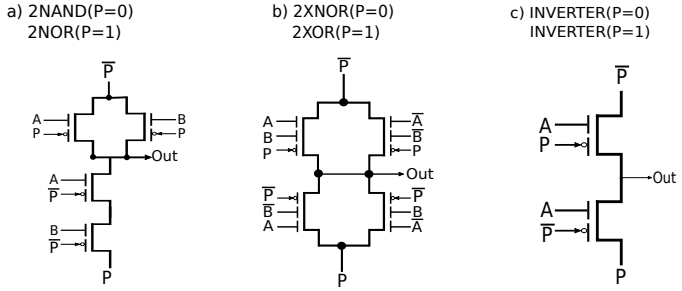


Fig. 1: Reconfigurable Gates [11]

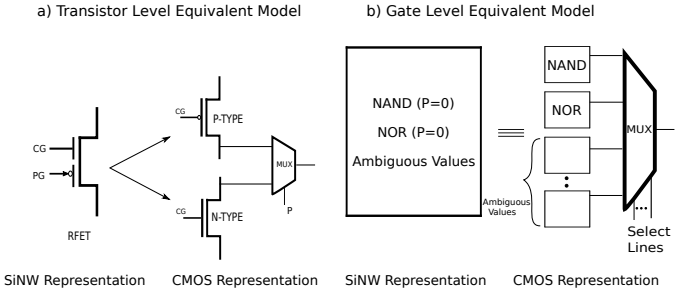


Fig. 2: Equivalent Models

C. Attack Model

We presume that the attacker has access to the layout of a particular ASIC. On obtaining the layout, it is easy to identify particular gates and their functionalities using various reverse engineering methodologies as mentioned in [20]. Having obtained this information, the attacker can easily obtain the netlist to replicate the functionality of an ASIC. We tackle this problem by introducing a key-based authentication technique where a key is required to obtain the functionality of a particular logic gate.

D. Shannon's Entropy

Shannon's entropy is a measure used to analyze the key strength for various logic circuits. Shannon's entropy basically indicates the number of (entropy) bits needed to encode a string of symbols, based on the frequency of occurrence of symbols. The equation for entropy $H(X)$ is given by:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i) \tag{1}$$

where n is the number of distinct symbols in a key and $p(x_i)$ is the probability of occurrence of the i^{th} symbol. The higher the entropy, the more the strength of the key [21]. It also implies that there is a wider range of combinations one has to go through in order to decrypt a key with a higher entropy. Let's consider an example of two versions of a random 10-bit key. *Version-1* has seven 1s and three 0s whereas *version-2* has five 1s and five 0s. The entropy per symbol for *version-1*, as calculated using equation 1, is 0.88 bits and the same for *version-2* is 1 bit. This means that the amount of information concealed in one bit of *version-1* is 0.88 bits and the amount of information stored in one bit of *version-2* is 1 bit. This concept can also be understood by dealing with permutations. *Version-1* can provide 120 different keys ($\frac{10!}{7!\times 3!}$) and *version-2* can provide 252 different keys ($\frac{10!}{5!\times 5!}$). Therefore, *version-2* provides stronger encryption.

III. Harnessing Security Features Through SiNW RFETs

In this section, we focus on various aspects of security which are possible due to tunable properties of emerging reconfigurable technologies.

A. Key For Polymorphic Logic Gates

Efficient combinational logic gates proposed in [11] based on SiNW RFETs are suitable candidates for polymorphic gates. These logic gates are shown in Fig. 1. By changing the value of P in these logic gates, different logical functionality is achieved through a single gate. A logical circuit based on SiNW RFETs would have multiple such logic gates. To make the circuit behave in a desired fashion, a specific set of values for P will be required for each logic gate. A sequence of inputs to P for a series of logic gates can form the basis of a key-based authentication technique for ICs. For finer control on the security of a circuit, we can have separate individual program gate input to each RFET. However, such control will come at an increased routing overhead. To deliver such features in CMOS, one had to either replace a normal logic gate from the netlist with additional circuitry [9] or resort to other means which intentionally increased the overhead.

B. The Concept of Don't Cares

As explained before, the strength of an encoded key is directly proportional to its Shannon's entropy. The entropy would be maximum if the number of 0's and 1's in the key are equal. To achieve this goal, the SiNW RFET based inverters can be exploited. Fig. 1(c) shows an inverter. In SiNW based RFETs, since there is no separation between p-type and n-type behaviour, the pull-up and pull-down part of an inverter can be readily interchanged without affecting the full voltage swing. Hence, both 1 or 0 as inputs to the program gate P of an inverter, does not alter its functionality. Such use of RFET based inverters have been shown in [13]. We can say that the program gate terminal input of an inverter is a **Don't Care** bit i.e. it can be both 1 or 0. This behaviour can be used for our benefit. Due to the presence of a large number of inverters in the circuit, we can distribute the 1's and 0's for inverters based on our security requirement. Additionally, number of don't care bits can be increased for a system by including extra buffers without altering its functionality. Hence *Don't Care* bits act as a tunable knob to alter the security metric of a system by striving to maximize Shannon's entropy.

IV. Experimentation

In this section, we perform various experiments to evaluate the strength of the security provided by SiNW RFET based logic gates. SiNW has a wider range of logical expression per unit area than CMOS based circuit as shown in Fig. 2. This is tapped to increase the security of the circuit. Firstly, we evaluated the above concepts for MCNC benchmark suite [22] and then for a proof of concept, we use a case study of 2-bit RCA to have a fair comparison between SiNW and CMOS based logic circuit.

A. Experimental Setup

We used ABC [3] tool for logic synthesis for our study. We carried out logic synthesis for the MCNC benchmark suite. For the genlib, we used an abridged version of mcnc.genlib containing only the logic gates which are mentioned in [11]. For each logic gate, we designate the key as a value of the input to the program gate. For instance, Fig. 1 suggests that the input key bit for the NAND/NOR combination would be 0 for the NOR gate and 1 for the NAND gate. Similarly, for the XOR/XNOR combination. The Table I shows a subset of test cases from 219 MCNC benchmarks, including all the

Algorithm 1 Don't Care Adjustment

Input: No. of Ones, Zeros, DCs and Key Length
Ensure: New Ones+New Zeros=Key Length
  if $Ones > KeyLength/2$ **then**
    $NewOnes = Ones$
    $NewZeros = Zeros + DCs$
  else if $Zeros > KeyLength/2$ **then**
    $NewZeros = Zeros$
    $NewOnes = Ones + DCs$
  else
    $ExtraOnes = KeyLength/2 - Ones$
    $NewOnes = Ones + ExtraOnes$
    $NewZeros = KeyLength - Ones$
  end if

TABLE I: Entropy Calculation and Number of Trials

File Name	Ones	Zeros	Dont Cares	H(X) (With DC Adj.)	H(X) (Without DC Adj.)	No. of Trials Required
alu4	237	420	145	0.9984	0.9433	2.67×10^{241}
apex1	478	1613	305	0.9116	0.7756	1.85×10^{721}
b9	27	65	38	1.0000	0.8732	1.36×10^{39}
C1355	164	100	8	0.9692	0.9572	7.59×10^{81}
C1908	167	136	36	0.9999	0.9924	1.12×10^{102}
C2670	281	306	141	1.0000	0.9986	1.41×10^{219}
C3540	532	426	216	1.0000	0.9912	2.56×10^{353}
C432	107	103	52	1.0000	0.9997	7.41×10^{78}
C6288	466	1022	35	0.9138	0.8968	2.94×10^{458}
C7552	692	706	297	0.9999	0.9999	1.76×10^{510}
dalu	138	827	135	0.8084	0.5920	1.36×10^{331}
des	979	1135	405	0.9999	0.9961	1.97×10^{758}
exep	230	308	101	1.0000	0.9848	2.28×10^{192}
i7	318	335	197	1.0000	0.9995	7.51×10^{255}
k2	715	782	347	1.0000	0.9986	1.26×10^{555}
rot	594 202	245	147	1.0000	0.9933	6.48×10^{178}
t481	683	659	124	1.0000	0.9998	2.04×10^{441}
x3	927 297	435	195	0.9999	0.9742	1.13×10^{279}

important circuits mentioned in [22]. In the table, we have used keys for all the logic gates in a circuit. This means that a bigger circuit will have more key-length. Users are free to choose any compression mechanism to have a fixed key length.

B. Discussion and Results

The experimental results for our tunable security flow for a subset of 219 MCNC benchmark circuits are shown in Table I. First, we obtained the key length, number of 1s, 0s and Don't Cares for all the circuits. The *H(X) Without DC Adj.* column shows the basic entropy of the circuit without using don't care adjustment. We can observe that the value of Shannon's entropy is less in the case where the number of 1's and 0's are highly skewed as in the case of C6288 benchmark. C6288 has 466 1's and 1022 0's. The entropy calculated is 0.8968. The fifth column enumerates the number of don't care bits due to the inverters present in the mapped logic network. To increase the entropy of the key, we applied the method of Don't Care adjustment. The *H(X) With DC adj.* column lists the entropy after DC adjustment and we can see that in almost all the cases, the entropy has increased. The last column describes the number of trials required for guessing the exact key. With the help of Don't Care adjustment, we were able to increase the entropy in **99.1%** of the circuits and obtain the maximum entropy in **38.5%** of the circuits.

Don't Care Adjustment: *Don't Care (DC) Adjustment* enables a tunable knob for deciding the security metric of a circuit. We then hard-coded some of the *Don't Cares* to be equal to 1 and some to be equal to 0. We noticed that using *Don't Cares* significantly increases the entropy. For many cases (38.5%), we were even successful in obtaining the maximum entropy. In Table I, we can see that entropy has increased for all the benchmarks and has reached closed to 1 in most cases. The modification basically strives to equalize the number of 1s and 0s in a particular key, based on Algorithm 1. Our algorithm takes the number of 1s, 0s and DCs as input and ensures that 1s and 0s sum up to the key length. Further, it checks if any of the 1s or 0s occupy more than half the key length. In such a case, we assign all the DCs to the opposite bit. If none of the 1s or 0s occupy more than half the key length, we attempt to assign the DCs to 1s and 0s in such a way that equalizes the number of 1s and 0s in the final key. In Fig. 3, we show the difference between the probability of occurrence of 1s and 0s with and without don't care adjustment. We can see that the skew between the probabilities has decreased from Fig. 3b to Fig. 3a.

Consider the case of benchmark b9. In total, including the don't care bits, the key-length is 130 for b9. Out of 130 bits, there are 27 ones, 65 zeros and 38 don't care bits. Shannon's entropy calculated without considering don't care bits at this point comes out to be 0.87. Fortunately, we have 38 don't care bits and if we make all of them as ones, we get Shannon's entropy as 1. It is to be noted that assigning don't care bits is totally a runtime choice since either value will enable the inverter.

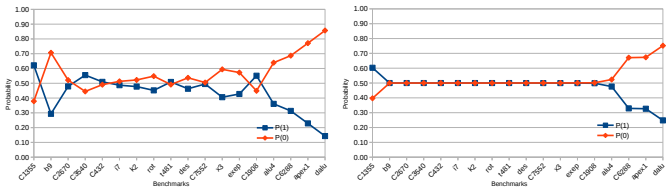
Since don't care bits do not affect the actual functionality of the circuit, we can add extra inverters or buffers in the circuit for increasing the entropy of a circuit at the cost of increased area. Hence, *Don't Care adjustment* is an excellent and cheap method to increase the entropy of a key encryption for a logic circuit.

Compression of Keys: In the process of assigning a particular key, especially for large circuits, it is important that we use some compression algorithm, which compresses the key in a fashion that has the least overhead and maximum encryption. Consider the case of k2 benchmark, which natively has around 1844 key bits. Such key-lengths are impractical to use. A better approach would be to compress the keys and use various compression mechanism. An interesting part is that using a particular compression will reflect on the layout of the circuit as well. This is because some of the logic gates which are sharing keys will have lesser routing overhead. Selecting keys for logic gates based on some order like topological as in from PI to PO, reverse topology order or level order will add extra levels of securities to the design under consideration.

Camouflaging: Camouflaging is an added functionality of reconfigurable logic gates. SiNW RFETs show runtime reconfigurability inherently. As mentioned earlier, from the point of view of the layout of a logic gate which, even after using *X-rays* and delayering techniques, an attacker will find it tough to deduce the functional output of a logic gate. Such concept has been explored previously for CMOS technology using camouflaging layouts as mentioned in [15] but the overhead related to camouflaging in CMOS based gates is very high. The area overhead is around 4 times as compared to a normal NAND gate. A direct relation can be laid between the physical layout of a logic gate and the range of logical functions it can deliver. More the number of functions, more is the strength of camouflaging.

V. Case Study: 2-bit Ripple Carry Adder

For a proof of concept, we demonstrate the above concepts with a simple circuit of a 2-bit Ripple Carry Adder (RCA) in both SiNW and CMOS technology. For the study, we implemented and mapped a 2-bit RCA in the CMOS technology using the ABC Logic Synthesis Tool. Then considering the expressive power of SiNW, we build an equivalent mapping in CMOS by modifying the netlist. To obtain a similar range



(a) Without DC Adjustment (b) With DC Adjustment

Fig. 3: Change in probabilities of 1s and 0s as a result of Don't Care Adjustment

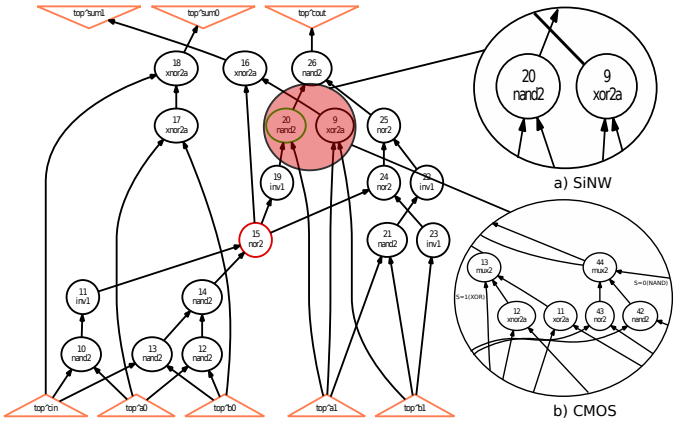


Fig. 4: A Synthesized Netlist for 2-bit RCA for SiNW and CMOS

of functions as exhibited by RFETs for enhanced security, we replaced the nodes with corresponding additional functional gate along with a multiplexer as shown in Fig. 2 in CMOS technology. Further, since changing the program gate input for the inverter does not modify the output of the gate, we insert a multiplexer after the inverter keeping both inputs to the multiplexer to be the same as the output of the inverter. Fig. 4 helps us visualize the difference between the SiNW based 2-bit RCA and the traditional CMOS based 2-bit RCA. The figure represents the synthesized netlist in ABC.

For getting area numbers for SiNW based circuit, we use the standard SiNW library as mentioned in [10]. To focus on the key, we find that the circuit uses 7 NAND gates, 3 NOR gates, 4 inverters, 1 XOR gate and 3 XNOR gates. To designate a basic set of authentication key bits for a mapped netlist, we used the topological order from *Primary Inputs* (PI) to *Primary Outputs* (PO) and use *left to right* order of occurrence of logic gates in the mapped logic circuit. It is to be noted that this order is generic and circuit designers are free to choose whichever order they want to generate their own specific key. Using the above order, we get authentication keys for each level as shown in Table II. This means that we have 10 zeros, 4 ones and 4 DCs for this combination of logic gates.

Don't Care as a Tunable Knob for Security: As explained earlier, inverters can be treated as don't cares in the authentication key. For the case of a 2-bit RCA, even after applying the don't care adjustment techniques as mentioned in sections III-B and IV-B, we do not attain the maximum possible entropy. However, we still increase our entropy from 0.8631 bits per symbol to 0.9911 using this adjustment. To achieve the maximum possible entropy, i.e. 1 bit per symbol, it is necessary to equalize the occurrences of 1s and 0s in the key. This can be done by an inserting equivalent number of inverters in the gate level netlist such that the final output is not affected. For such a demonstration, we insert two inverters between the nodes 14 (NAND) and 15 (NOR) (Fig. 4). This

TABLE II: Authentication keys for each level of RCA mapping

Levels	1	2	3	4	5	6
Keys	000	X0	10X	X1X	0011	000

will provide us with two more don't cares to control without affecting the output. These two DCs can be configured to be 1s, thereby increasing the entropy bits per symbol to 1.

VI. Conclusion

This study demonstrates how emerging reconfigurable technologies can be used for hardware security with increased encryption. We have analyzed the concept of a key based authentication technique for hardware systems and examined its reliability. We have evaluated the strength of encryption provided by the key-based authentication process and how Don't Care adjustment leads to higher Shannon entropy.

References

[1] Y. Bi et al. "Leveraging Emerging Technology for Hardware Security - Case Study on Silicon Nanowire FETs and Graphene SymFETs". In: *Asian Test Symposium*. 2014.

[2] Yu Bi et al. "Emerging Technology-Based Design of Primitives for Hardware Security". In: *J. Emerg. Technol. Comput. Syst.* (2016).

[3] Robert Brayton et al. "ABC: An academic industrial-strength verification tool". In: *Computer Aided Verification*. Springer. 2010.

[4] A. Chen et al. "Using emerging technologies for hardware security beyond PUFs". In: *DATE*. 2016.

[5] Pierre-emmanuel Gaillardon et al. "Nanowire systems: technology and design". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* (2014).

[6] Naoki Harada et al. "A polarity-controllable graphene inverter". In: *Applied Physics Letters* (2010).

[7] Andr Heinzig et al. "Reconfigurable Silicon Nanowire Transistors". In: *Nano Letters* (2012).

[8] Yu-Ming Lin et al. "High-performance Carbon Nanotube Field-effect Transistor with Tunable Polarities". In: *IEEE Trans. Nanotechnol.* (2005).

[9] J. T. McDonald et al. "Functional polymorphism for intellectual property protection". In: *HOST*. 2016.

[10] S. Rai et al. "A physical synthesis flow for early technology evaluation of silicon nanowire based reconfigurable FETs". In: *DATE*. 2018.

[11] S. Rai et al. "Designing Efficient Circuits Based on Runtime-Reconfigurable Field-Effect Transistors". In: *TVLSI* (2018).

[12] S. Rai et al. "Emerging Reconfigurable Nanotechnologies: Can They Support Future Electronics?" In: *ICCAD*. 2018.

[13] S. Rai et al. "Hardware Watermarking Using Polymorphic Inverter Designs Based On Reconfigurable Nanotechnologies". In: *ISVLSI*. 2019.

[14] J. Rajendran et al. "Security analysis of logic obfuscation". In: *DAC*. 2012.

[15] Jeyavijayan Rajendran et al. "Security Analysis of Integrated Circuit Camouflaging". In: *CCS*. ACM, 2013.

[16] J. A. Roy et al. "EPIC: Ending Piracy of Integrated Circuits". In: *DATE*. 2008.

[17] SEMI. *Innovation is at risk as semiconductor equipment and materials industry loses up to $4 billion annually due to IP infringement*. 2012.

[18] SEMI. *Intellectual Property (IP) Challenges and Concerns of the Semiconductor Equipment and Materials Industry*. 2012.

[19] V. Simek et al. "Reconfigurable Platform with Polymorphic Digital Gates and Partial Reconfiguration Feature". In: *European Modelling Symposium*. 2014.

[20] Randy Torrance et al. "The State-of-the-Art in IC Reverse Engineering". In: *CHES*. Springer-Verlag, 2009.

[21] A. Vassilev et al. "The Importance of Entropy to Information Security". In: *Computer* (2014).

[22] Saeyang Yang. *Logic Synthesis and Optimization Benchmarks User Guide Version 3.0*. 1991.