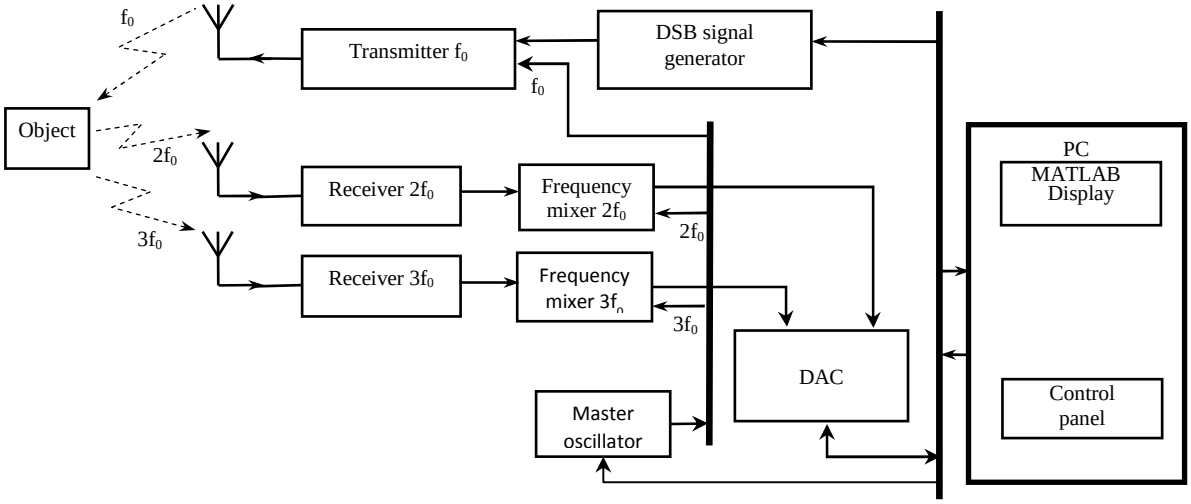


Fig. 4. It is block diagram of the proposed nonlinear junction radar using DSB-signal

Interesting technical challenges for future research could be the development of decision-making algorithms for the identification of various objects on the basis of information extracted from the spectrum of re-signals, as well as adjust the emission power of the nonlinear radar.

REFERENCES

1. Чертков, В.М. Использование фазоманипулированного сигнала в задачах нелинейной радиолокации. / В.М. Чертков, С.В. Мальцев // Вестник Полоцкого государственного университета. – Фундаментальные науки. – 2010. – № 3.
2. Щербаков, Г.Н. Применение нелинейной радиолокации для дистанционного обнаружения малоразмерных объектов / Г.Н. Щербаков // Специальная техника. – 1999. –№ 6. – С. 34–39.
3. Рувинова, Э. Нелинейные радиолокаторы: противодействие радиоэлектронным средствам шпионажа / Э. Рувинова // Электроника: Наука, Технология, Бизнес. – 2000. – №4. – С. 28–32.
4. Пат. № 6057765 США, МКИ G08B 13/14. Non-Linear Junction Detector / Jones T. (США); Research Electronics Int. №09/167844; Заявл. 7.10.1998; опубл. 2.05.2000. НКИ 340/572.2
5. Пат. № 6897777 B2 США, МПИ G08B 13/14. Non-Linear Junction Detector / Holmes S. (США); Audiotel Int. Ltd. №10/467507; Заявл. 11.02.2002; опубл. 24.05.2004. НКИ 340/572.2
6. Вернигоров, Н.С. Исследование многочастотного зондирования в нелинейной радиолокации для увеличения дальности обнаружения нелинейного объекта и определения его координат / Н.С. Вернигоров // Информост. Радиоэлектроника и телекоммуникации. – 2006. – №2(44). – С. 24–28.
7. Вернигоров, Н.С. К вопросу о применении многочастотного сигнала в нелинейной радиолокации / Н.С. Вернигоров, А.Р. Борисов, В.Б. Харин // Радиотехника и электроника. – 1998. – Т. 42. – № 1. – С. 63–69.
8. Пат. № 6765527 B2 США, МПИ G01S 13/04. System and Method of Radar Detection of Non-Linear Interfaces / Jablonski D. (США); The Johns Hopkins University. №10/048769; Заявл. 31.01.2002; опубл. 20.07.2004. НКИ 342/193

UDC 519.681.3/004.41

THE BASIC PRIMITIVES OF DIGITAL CIRCUIT OBFUSCATION

VLADIMIR SERGEICHIK, ALEXANDER IVANIUK
Belarusian State University of Informatics and Radioelectronics, Belarus

Features of obfuscation as applied to specifications in VHDL language are considered. Brief survey of obfuscation types is given and their drawbacks are investigated. Circuit obfuscation methods are considered. Basic primitives of circuit obfuscation are proposed and ways of their usage are explored.

Hardware piracy assisted by modern equipment turned into a threatening problem during past decade. Today the financial loss from illegal manufacturing and usage of digital devices is estimated in more than 1 billion dollars a day [1]. Piracy is not the only danger. Other threats are evolved and improved, new threats are

developed. Successful attacks on hardware implementations of secure cryptographic algorithms (side-channel attacks) take place more often. Examples of such side channels are leakages of power during computations [2], electromagnetic emanation [2], timing imperfections [2], even acoustic emanations from power supply chains are already exploited [3]. Side channels allow deducing secret key values. Another threat is hardware Trojan injection. Hardware Trojan is malicious modification in circuit. Trojans can negatively affect functionality of circuit, leak secret information, switch off and even destroy circuit. Trojans pose the greatest danger to critical applications: power stations, medical equipment and military systems. High difficulty in detecting of such malicious alterations makes situation even worse. In this context the development of hardware protection methods turns into the most urgent and important objective. Obfuscation is one of the possible ways. Obfuscation is intended to make structure and function of circuit difficult to perceive and comprehend in order to prevent reverse engineering or substantially increase its time and cost [4]. Also obfuscation is often used to hide author's watermarks and user's fingerprints [5].

There are two types of obfuscation methods in HDL's case: *lexical* and *functional (circuit)* [6]. *Lexical* obfuscation affects only source code level. And here is rooted its main disadvantage: synthesis results (circuits) before and after obfuscation are identical [6], i.e. circuit remains unchanged. This observation reveals the simplest method of attack which is logical (RTL) synthesis [6]. Many different approaches of lexical obfuscation have been developed for general purpose programming languages. Comprehensive taxonomy is presented here [7]. These approaches are suitable for protection of HDL sources as well.

Formally lexical obfuscation can be described as shown below:

$$V; V^* = obf(V); O(V) < O(V^*); Sch = DD(V) = DD(V^*),$$

here V – HDL source, V^* – lexically obfuscated source, *obf* – obfuscation, O – complexity, *DD* – synthesis procedure, *Sch* – circuit.

Considering lexical obfuscation for HDL description we should note that synthesis can help to create lexical transformations. *Collapsing signals introduction* inserts combinations of interweaved signals into design. These combinations are distributed across source code; also they are parts of expression, which computes constant. Synthesizer always detects and removes such combinations, so they don't affect resultant circuit. But in order to remove them manually reverse engineer forced to locate all of them, put together and compute full expression. Using previous notation this transformation and several others (identifiers scrambling, comments removing, layout destroying) can be described as follows:

$$V^* = V \pm V_r; Sch = DD(V); Sch^* = DD(V^*) = DD(V \pm V_r) = DD(V) \pm DD(V_r) = Sch \pm \varnothing = Sch,$$

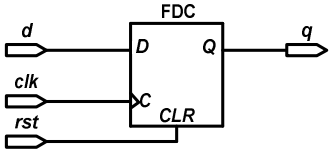
here V_r – HDL description of nonsynthesizable (empty) circuit.

Lexical obfuscation example is presented in fig. 1. Following obfuscation transformations are used in (*a*): identifiers scrambling, comments removing, collapsing signals introduction, layout destroying. Elements (*c*) and (*d*) clearly illustrate main drawback of lexical obfuscation.

```
a)process (o0o1o0ioo,oi1oo0i0o,oi1oo0io0) begin o0o1o0ioo<=(oi1oo0i0o) OR oi1oo0io0; end process ; process (oo0oo0ioo,oi1oooi0o,oi1oo0loo)
  begin oo0oo0ioo<=(oi1oooi0o) AND oi1oo0loo; end process ; process ( oi1oo0io0,oi1oooi0o,oi1oo0loo) begin oi1oo0io0<=(oi1oooi0o) AND  not
    oi1oo0loo; end process ;process (o00100l10,o00100llo,oi1oo0i0o, oi1oooi0o,oi1oo0loo) begin if (oo0100ll0 = '1') then oooi00ll0 <='0'
  ; elsif (rising_edge (o00100llo)) then oooi00ll0 <= o00100l10; end if;o00100ll0<=oooi00ll0;oi1oo0i0o<=(not oi1oooi0o)AND not oi1oo0loo;
  end process; process (o00l00ioo,o01oo0ioo,oo0oo0ioo) begin o00l00ioo<=(o01oo0ioo) OR oo0oo0ioo; end process; process (o0010o0ioo
  ,o0o1o0ioo,o00l00ioo) begin o001o0ioo<=(o0o1o0ioo)OR o00l00ioo; end process ; process (o01oo0ioo,oi1oooi0o, oi1oo0loo)
  begin o01oo0ioo<=( not oi1oooi0o) AND oi1oo0loo; end process ;
```

```
b)process(d, clk)
  begin
    if (rst = '1') then
      qx <= '0';
    elsif
  (rising_edge(clk))then
      qx <= d;
    end if;
    q <= qx;
  end process;
```

c)



d)

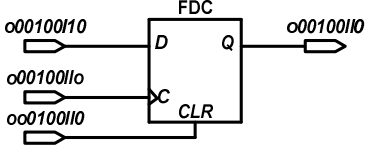


Fig. 1. Example of lexical obfuscation: (*a*) obfuscated snippet; (*b*) target code before obfuscation; (*c*), (*d*) synthesis results before and after obfuscation respectively

The essence of *circuit* obfuscation is creating more sophisticated and unintelligible circuit which functionality (externally observed behaviour) is equivalent to the original one [6]:

$$V; V^* = obf(V); Sch = DD(V); Sch^* = DD(V^*); Sch \neq Sch^*; func(Sch) \equiv func(Sch^*);$$
$$O(V^*) > O(V); O(Sch^*) > O(Sch);$$

here *func* – functionality of circuit.

There are several approaches to circuit obfuscation. "*Stuttering ciruits*" method is based on transformation of circuit in a way that when the wrong key is passed the circuit performance degrades significantly but it is still functioning [8]. Another method is insertion of "*time bombs*" by designer. "*Time bomb*" is logic which disables device functioning after expiring of evaluation period (several number of power-ons) until the correct key is entered [9]. The next method is based on FSM state space expanding. Device begins its work at random state. In order to reach the start a state user has to pass the correct key. If the key is wrong then ciruit goes into "*black-hole-state*" where it loops until reset [10].

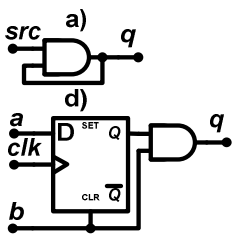
Constant generators insertion is one of the basic methods of circuit obfuscation. Constant generator is a form of *opaque predicate*, which value is known at obfuscation time but has to be deduced by adversary during analysis [7]. The method implies substitution of "0" and "1" pins by circuits (primitives), which generate appropriate logical values permanently:

$$V_{\{0,1\}}; DD(V_{\{0,1\}}) = Sch_{0,1} \notin \{V_{DD}, GND\}; func\{Sch_{0,1}\} \equiv func\{V_{DD}, GND\},$$

here $V_{\{0,1\}}$ – HDL-description of the primitive; V_{DD}, *GND* – sources of logical "1" and "0" respectively.

Complexity of opaque predicate will be determined by analysis difficulty of the primitive.

Fig. 2 illustrates examples of proposed constant generators.



b)
```
x <= x and src;
q <= x;
```

e)
```
process(a, clk, b)
begin
  if (b = '1') then
    val <= '0';
    elsif (rising_edge(clk))
    then val <= a; end if;
end process;
q <= val and b;
```

c)
```
o0ooilool <=o0ooilool
and oo0oilooi;oo0oilool
<= o0ooilool ;
```

f)
```
process (oi1l0li1o,oi1l0l11o,oi1l0l1l0
) begin if (oi1l0l1l0= '1' )then o00l0l110
<='0'; elsif (rising_edge(oi1l0l11o)
) then o00l0l110 <= oi1l0li1o;end if; end
process;oo1l0l110 <= o00l0l110 and oi1l0l1l0;
```

Fig. 2. Generators of constants (*a*), (*d*); original HDL-sources (*b*), (*e*); lexically obfuscated sources (*c*), (*f*)

It is critical to make generator circuit unrecognized and not minimized by synthesis tool. Different circuits which were made using only combinational logic weren't sufficient. Sequential circuits and sequential-combinational circuits are more promising and effective from hardware resources utilization point of view. Considering further tricks, we should mention usage of signals with well-defined semantics (such as system reset, clock) on the inputs of generators.

Generators from fig. 2 were synthesized using XST I.24 Release 8.1i. Synthesizer wasn't able to recognize constants and minimize such circuits. Circuit (*a*) has two flaws: synthesizer throws warning about combination loop, which itself gives good clue for adversary; if input *src* has "1" during power-on then until it will be changed it's possible for output to return "1". Countermeasure against second flaw is to connect generator's input to system reset. Circuit (*b*) is free from such flaws, but it utilizes more hardware resources and can produce glitches.

Basic primitives pave the way for many other circuit obfuscation transformations. With help of basic primitives we can build compound primitives for more complex functions. For example, having "0"-generating basic primitive we can implement logical *and* using 2-input multiplexor. It is illustrated in fig. 3(*a*). We can build logical *or* in similar way, it is shown in fig. 3(b).

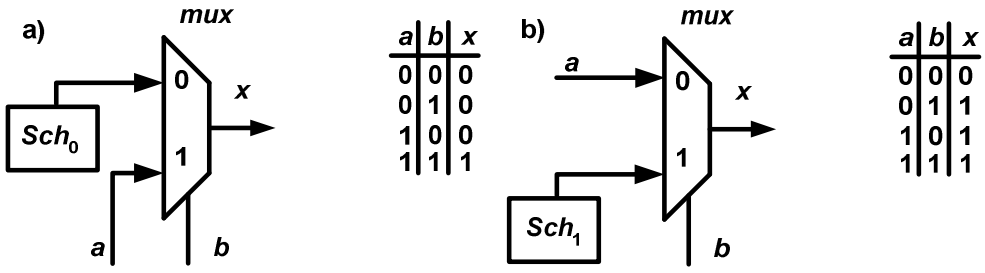


Fig. 3. Obfuscated and gate and its truth table (a); obfuscated or gate and its truth table (*b*)

Synthesizer doesn't recognize these compound primitives, so reverse engineer is forced to completely analyze such circuits.

Main advantage of circuit obfuscation is complicating of comprehension not only of HDL description but also of synthesis result. Mentioned examples illustrate problems of circuit obfuscation which are increase in hardware utilization and regress in performance in some cases.

Primitive's insertion process takes place on HDL level. In order to increase complexity it's possible to create primitives which are depended from key. If key is wrong then they generate random signals, breaking specification and correctness of device's functioning.

Software metrics are usually used for complexity measurement of source codes. Complexity metrics indicate quality of source code. Also these metrics are very useful to measure efficiency of obfuscation transformations against human-reader. The main difference is that in case of obfuscation the goal is not to minimize but to maximize their values [7]. Such metrics have their problems in case of circuit obfuscation. First, they don't indicate hardware overheads and time penalty which is incurred by circuit obfuscation. Second, they don't take into account complexity of the resultant circuit. The solution to the first problem was proposed in [11], where the method for obfuscation transformation quality estimation is described. This method considers area and time overheads and allows flexible metrics tuning. The second problem is still open. The possible solution may be to borrow logical circuits complexity measurement approaches such as the total number of gates or the number of gates on the critical path [12]. The obvious flaw of such approaches is their inapplicability for sequential circuits; also they don't take into account the number of interconnections between blocks or connection lines.

The lexical obfuscation of HDL-sources alone is not sufficient for the protection of circuits because the result of such transformations doesn't survive synthesis. The solution to the problem is circuit obfuscation, which changes not only HDL-sources but also circuit. Different approaches exist to circuit obfuscation. Their main drawback is additional area and time overheads which they impose on design. *Constant generators insertion* allows hiding of constants which are valuable clues for reverse engineer. Necessity to explore interdependencies between primitive's inputs and source circuit's signals further increases time and analysis complexity of obfuscated circuit. It makes sense to use different types of obfuscation complementary in order to increase protection on all levels of abstraction. For example, HDL obfuscation makes source code unintelligible. Circuit obfuscation makes circuit functionality and structure unclear. Processor's commands obfuscation makes understanding of processor actions difficult. Data obfuscation makes data transfer protocol between digital device and memory unclear. It can be accomplished, for example by introducing function which maps source address to destination. It's obvious that every mentioned obfuscation type has its own application therefore their composition allows creating of more secure systems.

REFERENCES

1. Hardware Security Mechanisms for Authentication and Trust [Electronic Resource] / ECE Department, Rice University. – Houston, 2011. – Mode of Access: http://www.glsvlsi.org/archive/glsvlsi11/Koushanfar_MeteringGLS-VLSI.pdf. – Date of Access: 9.01.2014.

2. Majzoobi, M. Introduction to hardware security and trust / M. Majzoobi, F. Koushanfar, M. Potkonjak. – New York: Springer, 2011. – 427 p.

3. Genkin, D. RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis / D. Genkin, A. Shamir, E. Tromer // Cryptology ePrint Archive: Report 2013/857 [Electronic Resource]. – 2013. – Mode of Access: https://eprint.iacr.org/2013/857.pdf. – Date of access: 10.01.2014.

4. Chakraborty, R.S., Hardware Security Through Design Obfuscation: Dis., Ph.D / R.S. Chakraborty. – Clevelend, 2010. – 183p.

5. Chakraborty, R.S., HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection / R.S. Chakraborty, S. Bhunia // Computer-Aided Design of Integrated Circuits and Systems. – 2009. – Vol. 28, № 10. – P. 1493–1502.
6. Ivaniuk, A.A. Design of Embedded Digital Systems and Devices / A.A. Ivaniuk. – Minsk: Bestprint, 2012. – 337 p. (in Russian).
7. A Taxonomy of Obfuscating Transformations: Technical Report / The University of Auckland: Department of Computer Science; C. Collberg, C. Thomborson, D. Low. – Auckland, 1997. – 36 p.
8. Li, L. Structural transformation for best-possible obfuscation of sequential circuits / L. Li, H. Zhou // Hardware Oriented Security and Trust (HOST). IEEE, 2013. – Austin, Texas, USA, 2013. – P. 55 – 60.
9. Chakraborty, R.S. Hardware IP Protection during Evaluation Using Embedded Sequential Trojan / R.S. Chakraborty, S. Narasimhan, S. Bhunia // Design & Test, IEEE, 2011.
10. Desai, A. Anti-Counterfeit and Anti-Tamper Implementation using Hardware Obfuscation: Thesis: Master of Science / A. Desai. – Blacksburg, 2013. – 73p.
11. Brzozowski, M. Obfuscation quality in hardware designs / M. Brzozowski, V.N. Yarmolik // Proceedings Naukowe Politechniki Bialostockiej. Informatyka, 2009. – № 4. – P. 19 –29.
12. Circuit_complexity [Electronic Resource] / Wikipedia.org. – Mode of Access: http://www.en.wikipedia.org/wiki/Circuit_complexity. Date of Access: 8.01.2014.

UDC004.896:613.62

APPLICATION OF FUZZY LOGIC IN PROBLEMS OF RISK ASSESSMENT

ALENA HALYNSKAYA, OKSANA GOLUBEVA
Polotsk State University, Novopolotsk, Belarus

The paperdiscusses the use of fuzzy logic in the problems of risk assessment in economics and IT. Based on fuzzy logic method of occupational risk assessment considers.

The basics of fuzzy logic were formula tedby famous American mathematician Lotfi A. Zadeh attheend of 1960s. The paper "Fuzzy Sets"waspublishedin1965 inthe "Information and Control". It laid the foundations for modeling human intellectual activity and became the reference point for the development of new mathematical theory [1]. Zadeh gave the name for the new field of science – "fuzzy logic" (fuzzy – vague, uncertain). However, this theory was not put to use until the mid-1970s, when Ibrahim Mamdanidesigned of fuzzy controller for a steam engine [1]. Since then, fuzzy logic is widely use in control problems. Especially widespread fuzzy logic typical for Japan, where the world's leading companies are exploring and using fuzzy logic to design more commonsense instruments, devices and systems management.

Fuzzy logic manipulates such vague concepts as «cold», «close», «fast», etc., inherent in human thinking. The notion of fuzziness refers to classes in which there are different scales of toiletries, intermediate between full membership and belonging to this class of objects [2]. In other words, a fuzzy set is a class of objects in which there is no sharp boundary between those objects that are in this class, and those that it does not include.

On the basis of fuzzy inference were obtained solving a large number of problems of analysis and control of power systems [3], and process plants: chemical reactors, electric motors, welding processes, installations for water purification, cooling units, fans and air conditioners, heaters, rechargeable units, communication systems [4], transport.

Based on its instruments fuzzy logic has received its application in expert systems, including control problems and risk assessment. In the early 1990s a health management system applied for large firms in Japan. The fuzzy system diagnoses the health of patients and draw up personalized plans to help them prevent disease and stay fit.

The use of the theory of fuzzy sets expandsto problems of management and risk assessment, decision support. Widespread fuzzy set theory gets in the economic sphere. For example, in [5] provides a comparison of methods and models for risk analysis of bankruptcy. According to the results of the comparative analysis the most highly accurate prediction of bankruptcy of enterprises Mamdani (90%) and Tsukamoto (88%)models showed, followed by fuzzy multiple Nedosekin's method (80%) and finally, the worst performance prediction accuracy has a classic method Altman'sdiscriminant analysis (73%). In a paper [6] also conducted a comprehensive evaluation of the risk of bankruptcy corporations based on fuzzy descriptions.

Paper [7] devoted to the application of fuzzy set theory to analysis of investments in the securities market. The questions assess the risk of bankruptcy of the issuer, the project risk of direct investment, the risk of investments in stocks, bonds, options, and combinations thereof. The paper presents a technique for assessing the investment attractiveness of the shares. Suggested by the author an independent theory of risk assessment using fuzzy sets formed the basis of a number of software products developed by Russian companies. In [8] the use of fuzzy logic in assessing investment risks.