Science, Engineering & Education, 7, (1), 2022, 13-20

VHDL Models of the Basic Dual-Rail Asynchronous RSFQ Gates

Valery Todorov

University of Chemical Technology and Metallurgy
Department of Metallurgical Technologies, Electrical Engineering and Electronics
Sofia, Bulgaria

Received 19 October 2022, Accepted 28 October 2022

ABSTRACT

This paper presents a basic approach for applying Very High-Speed Integrated Circuits Hardware Description Language (VHDL) descriptions of dual-rail rapid single flux quantum circuits (RSFQ). Because RSFQ is naturally pulsed, dual-rail data encoding encourages event driven data processing at very high speeds. A method for suitable simulation models of the basic asynchronous RSFQ gates is described. The realization of VHDL models is performed in relation with the subsequent implementation for simulation and parameter optimization of complex logic cells.

Keywords: VHDL models, RSFQ gates, simulations of digital superconductive circuits.

INTRODUCTION

The RSFQ circuits are a new generation of superconducting logic circuits in which as switching elements used overdamped Josephson junctions (JJ) with non-hysteresis I-V curve. They use ultra-fast magnetic quantum pulses (SFQ) as information carriers, from which discrete voltage pulses are obtained, and the storage of information is carried out in the form of current flowing in inductive superconducting loops containing JJ and coils [1].

The information in the RSFQ circuits is naturally digital, which in combination with dual-rail coding predisposes to the use of an asynchronous approach in the implementation of various types of circuits and gates.

Asynchronous circuits do not have a timing signal. In them, each element reacts and changes its state when the input signal changes and transmits a signal to the next element after its transition. Control over the implementation of the communication is done through locally generated signals of *request* and *acknowledgement* organized into the so-called communication protocols [2]. RSFQ circuits are essentially self-clocking, which makes the asynchronous circuit design much more pragmatic [3]. There the dual-rail logic uses two signal lines where a pulse on the one represents a logical '*1*' and a pulse on the other represents a logical '*0*'. This data coding is completely asynchronous, and the timing information is contained in the appearance

**Correspondence to: Valery Todorov, University of Chemical Technology and Metallurgy, 8 Kliment Ohridski blvd., 1756 Sofia, Bulgaria, E-mail: todorov@uctm.edu.*

of signals themselves [4]. Combinations and Boolean logic values in two-rail coding are given in Table 1.

The RSFQ logic circuits can perform logical operations and functions at an extremely high speed - a tunnel JJ has switching times of about 1 - 2 ps. This enables synchronous RSFQ circuits to achieve frequencies in the order of 770 GHz for single gates [5], and more than 100 GHz in circuits with a higher degree of integration [6]. The frequency limit of this new technique is of the order of 1 THz and depends mainly on the level of the manufacturing process.

Combining the advantages of asynchronous circuits based on RSFQ technologies, dual-rail (DR) asynchronous RSFQ logic elements have been developed. The scheme of a unique element allows through it the realization of the following basic logical gates - DRAND, DRNAND, DROR, DRNOR [7]. By combining these elements with DRXOR and DRNOT, a library of elements is formed that may be utilized to design a wide range of logic operations and circuits [8].

VHDL MODELING

VHDL (Very High-Speed Integrated Circuits Hardware Description Language) is a programming language for designing digital circuits with very large-scale integrated (VLSI) at the logic level for design entry, documentation, and verification. The advantages of the VHDL description when designing digital electronic circuits are the following:

1) allows a description of the structure of the scheme, a look at the individual sub-elements and modules, shows the connections between the individual modules;

2) allows the use of already known functions and commands from other programming languages;

3) after the file-description is prepared, it in turn can be simulated on a program-simulator to check whether the scheme works correctly. This saves the cost of prototyping a component.

Table 1. Truth table in dual-rail coding.

Coding		Boolean Logical value
Line 1 “truth”	Line 0 “lie”	
0	0	Null
0	1	0 (Data 0)
1	0	1 (Data 1)
1	1	Invalid

A block diagram of the functional description of a logic circuit in VHDL is shown in Fig. 1. It includes several files and modules. The *description file* of the logic circuit in VHDL contains two modules: *the entity part* and the *architecture part* [9]. The entity module describes only the interface of a logic circuit as well as the type of signals through which it communicates with other circuits and modules. The architecture section describes the behavior of the circuit through a set of sequential and/or parallel operators. To simulate the circuit, in addition to the file description, a *test file* (*test-bench file*) file is required. Similarly, it describes the input signals that are needed and fed to the circuit. The test file is automatically attached to the component ports during the simulation, as shown in Fig. 1.

The timing characteristics of the individual RSFQ gates must be known when describing VHDL models. JSIM simulations are used to determine them, which take into account the times between the different input combinations [10]. The schematic symbol of one of the main logical DR RSFQ element - NAND (DRNAND) and the truth table are given in Fig. 2 and Table 2.

The timing diagram of the DRNAND’s input-output signals made with JSIM is shown in Fig. 3. They take into account the exact moments of time of occurrence of the individual signals as well as the time delays between them, which are used in the description of the VHDL model.

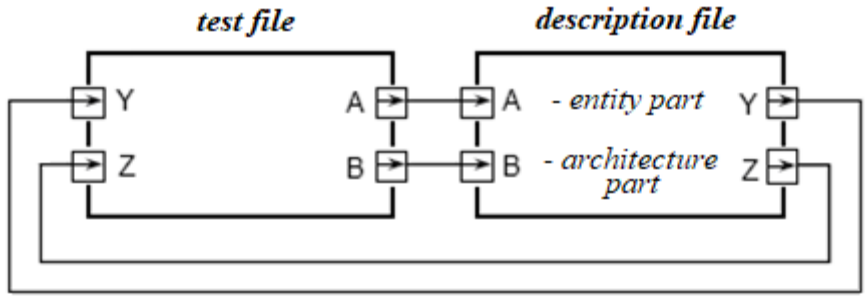

Fig. 1. Block diagram of a functional description of a logic circuit in VHDL.

Fig. 2. The schematic symbol of DRNAND.

Table 2. Truth table and signal combinations of DRNAND.

X	1	1	0	0
$\overline{X}$	0	0	1	1
Y	1	0	1	0
$\overline{Y}$	0	1	0	1
XY	0	1	1	1
$\overline{XY}$	1	0	0	0
	1	2	3	4

VHDL DESCRIPTION OF ASYNCHRONOUS DUAL-RAIL RSFQ NAND (DRNAND)

Complete descriptions of the main files of the VHDL model for the DRNAND are made as shown below.

1. Description file of DRNAND

```
-- The standard library for calling the package with the basic definitions
    library IEEE;
    use IEEE.std_logic_1164.all;
-- Entity file
    entity DRNAND is
      port (x, y, x_0, y_0 : IN std_logic;
                xy, xy_0 : OUT std_logic);
    end DRNAND;
-- Architecture part
    architecture behave OF DRNAND is
-- Declarative part of the architecture
    TYPE pulse_count IS RANGE 0 TO 2;
    constant d1 : TIME := 60 ps;
    constant d2 : TIME := 80 ps;
    constant dp : TIME := 5 ps;
    begin
    process
-- Declarative part of the process
    variable t1,t2,di     : TIME := 0 ps;
    variable k            :  pulse_count := 0;
    variable A,B,C,D  :  BIT :='0';
-- Declarative part of the architecture
    begin
        IF ( k=0 OR k=1) THEN xy <= ,0';
        xy_0 <= ,0';
        end IF;
        IF ((x'EVENT) and (x'LAST_
VALUE='0') and (x='1'))
            THEN k := k+1; A := ,1';
              IF (t1=0 ps) THEN t1:=NOW;
              ELSE t2:=NOW; end IF;
        end IF;
        IF ((y'EVENT) and (y'LAST_
VALUE='0') and (y='1'))
            THEN k := k+1; B := ,1';
```

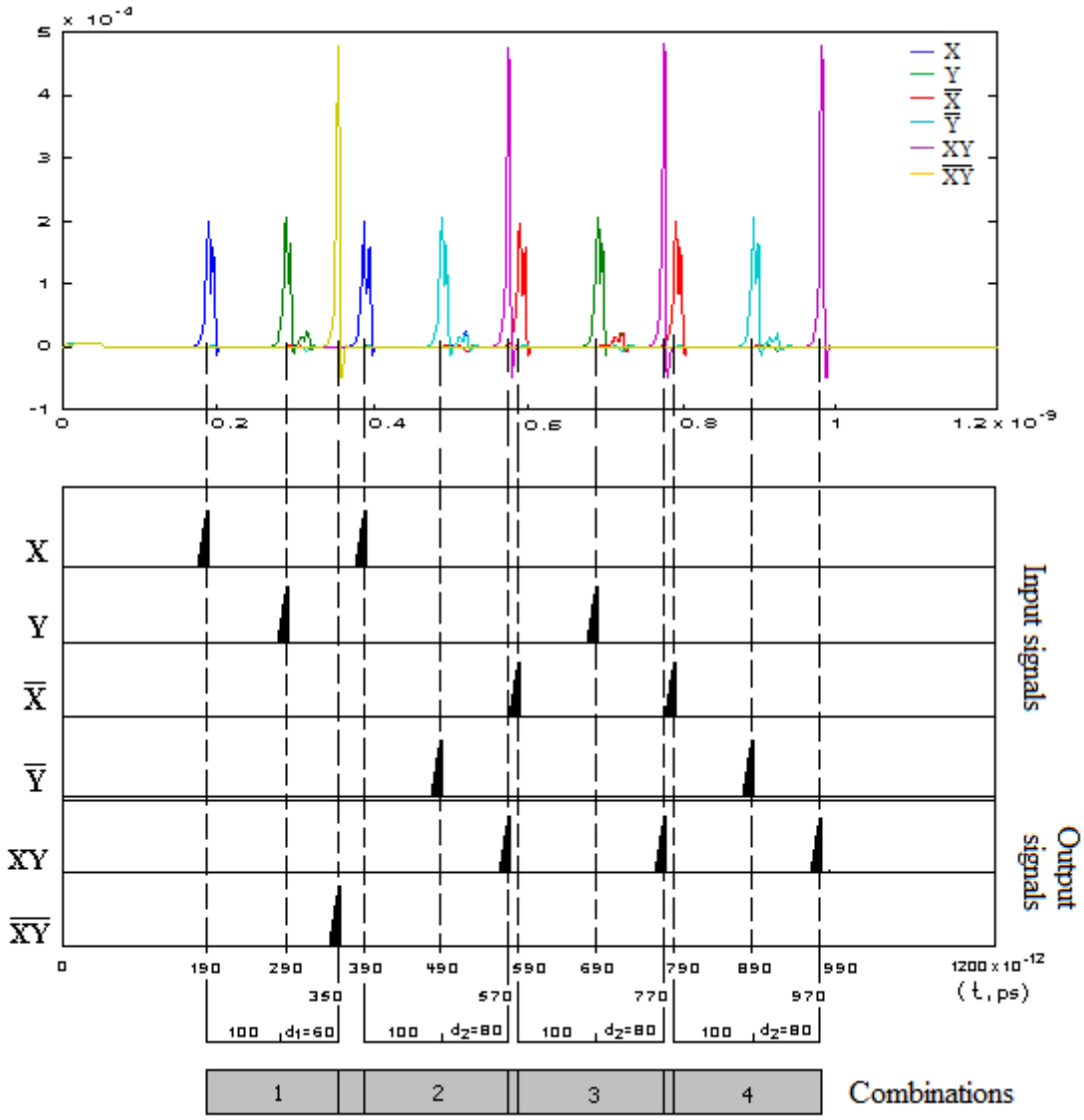

Fig. 3. JSIM time diagram of the input-output signals of DRNAND.

```
            IF (t1=0 ps) THEN t1 := NOW;
            ELSE t2 := NOW; end IF;
        end IF;
        IF ((x_0‘EVENT) and (x_0‘LAST_
VALUE=‘0‘) and (x_0=‘1‘))
            THEN k := k+1; C := ,1‘;
            IF (t1=0 ps) THEN t1 :=
NOW;ELSE t2 := NOW; end IF;
        end IF;
        IF ((y_0‘EVENT) and (y_0‘LAST_
VALUE=‘0‘) and (y_0=‘1‘))
            THEN k := k+1; D := ,1‘;
            IF (t1=0 ps) THEN t1 :=
NOW;ELSE t2 := NOW; end IF;
        end IF;
    CASE k is
        WHEN 0 => NULL;
        WHEN 1 => NULL;
        WHEN 2 =>
        di:=t2-t1;
```

```
            IF ((A='1') and (B='1')) THEN
WAIT FOR d1;
                xy <= ,1'; WAIT FOR dp; xy <=
,0';
            ELSIF ((A='1') and (D='1'))
THEN WAIT FOR d2;
                xy_0 <= ,1'; WAIT FOR dp;
xy_0 <= ,0';
            ELSIF ((B='1') and (C='1'))
THEN WAIT FOR d2;
                xy_0 <= ,1'; WAIT FOR dp;
xy_0 <= ,0';
            ELSIF ((C='1') and (D='1'))
THEN WAIT FOR d2;
                xy_0 <= ,1'; WAIT FOR dp;
xy_0 <= ,0';
            end IF;t1:=0 ps; t2:=0 ps; k:=0;
A:='0'; B:='0'; C:='0'; D:='0'; di := 0 ps;
          end CASE;
    WAIT ON x, y, x_0, y_0;
  end process;
end behave;
-- End Description file.
```

2. Test (test-bench) fail of DRNAND

```
-- The standard library with basic definitions
    library ieee;
    use ieee.std_logic_1164.all;

-- The entity structure of the test file does not
describe ports, as it uses those of the component
for which it is intended
    entity test_DRNAND is
    end test_DRNAND;

-- Architecture part
    architecture test of  test_DRNAND is
    component DRNAND
       port (x, y, x_0, y_0 : IN std_logic;
                   xy, xy_0  : OUT std_logic);
    end component;
    constant PW : TIME := 5 ps;
    signal x, y, x_0, y_0,xy, xy_0 : std_logic ;
    begin
-- Connections between the signals of the test
file and the ports of the component
    uut: DRNAND
       port map (x => x, y => y, x_0 => x_0,
y_0 => y_0, xy => xy, xy_0 => xy_0);

-- Define a number of processes corresponding
to the number of inputs of the component
x_pr:process
       begin
          x <= ,0'; WAIT FOR 190 ps;
          x <= ,1'; WAIT FOR PW;
          x <= ,0'; WAIT FOR 195 ps;
          x <= ,1'; WAIT FOR PW;
          x <= ,0'; WAIT FOR 605 ps;
end process;
    y_pr:process
       begin
          y <= ,0'; WAIT FOR 290 ps;
          y <= ,1'; WAIT FOR PW;
          y <= ,0'; WAIT FOR 395 ps;
          y <= ,1'; WAIT FOR PW;
          y <= ,0'; WAIT FOR 305 ps;
end process;
    x_0_pr:process
       begin
          x_0 <= ,0'; WAIT FOR 590 ps;
          x_0 <= ,1'; WAIT FOR PW;
          x_0 <= ,0'; WAIT FOR 195 ps;
          x_0 <= ,1'; WAIT FOR PW;
          x_0 <= ,0'; WAIT FOR 205 ps;
end process;
    y_0_pr:process
       begin
          y_0 <= ,0'; WAIT FOR 490 ps;
          y_0 <= ,1'; WAIT FOR PW;
          y_0 <= ,0'; WAIT FOR 395 ps;
          y_0 <= ,1'; WAIT FOR PW;
          y_0 <= ,0'; WAIT FOR 105 ps;
    end process;
end test;
-- End Test  fail.
```

To verify the correct operation of the compiled

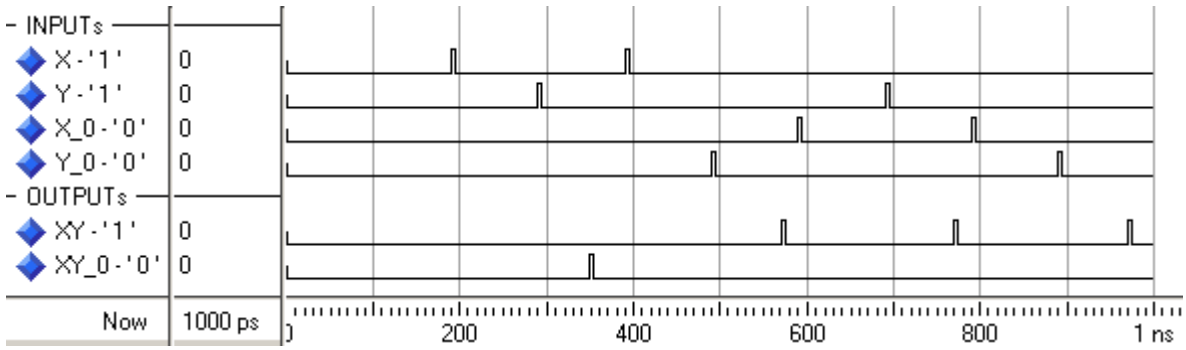

Fig. 4. Input-output signals from DRNAND simulations with ModelSim.

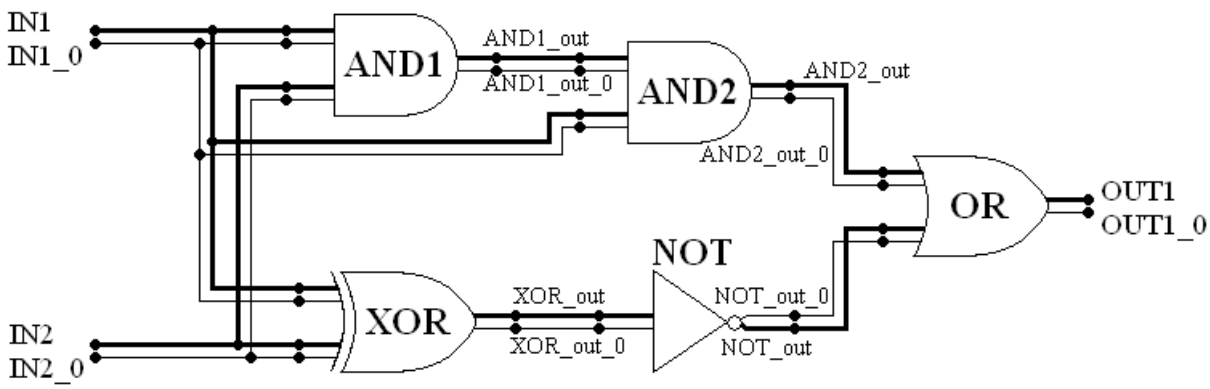

Fig. 5. Complex DR logic circuit for testing VHDL models.

VHDL model, a simulation was performed with ModelSim [11]. The graphs of the input-output signals from DRNAND simulations are given in Fig. 4.

The comparison with the simulations of the same gate with JSIM shows the correct operation of the presented VHDL model. Similar VHDL models of DRAND, DROR, DRXOR, DRNOT,

DR Multiplexer 2x1 and DR Demultiplexer 1x2 are described in [12].

SIMULATION OF A COMPLEX CIRCUIT WITH VHDL MODELS

To test the applicability of the presented VHDL models for practical purposes, a simulation of a complex DR logic circuit was performed, the circuit of which is shown in Fig. 5. The circuit architecture is formal and aims to test the approach to describing asynchronous dual-rail gates with VHDL descriptions.

The time diagram from the simulation of a complex logic circuit described with VHDL models is shown in Fig. 6. The result shows correct operation and allows the use of the presented VHDL models for high-level design of asynchronous dual-rail digital circuits [13].

CONCLUSIONS

The main asynchronous DR RSFQ gates are discussed in detail, as well as creating acceptable simulation models. VHDL models are described in connection with the later implementation of complex logic cell simulation and parameter optimization. Functional testing of RSFQ logic circuits at their high speeds is essential to improve circuit design, but due to the low amplitude of SFQ pulses, off-chip testing is a challenge. Using VHDL models is one of the most common solutions, especially for high integrated logic circuits.

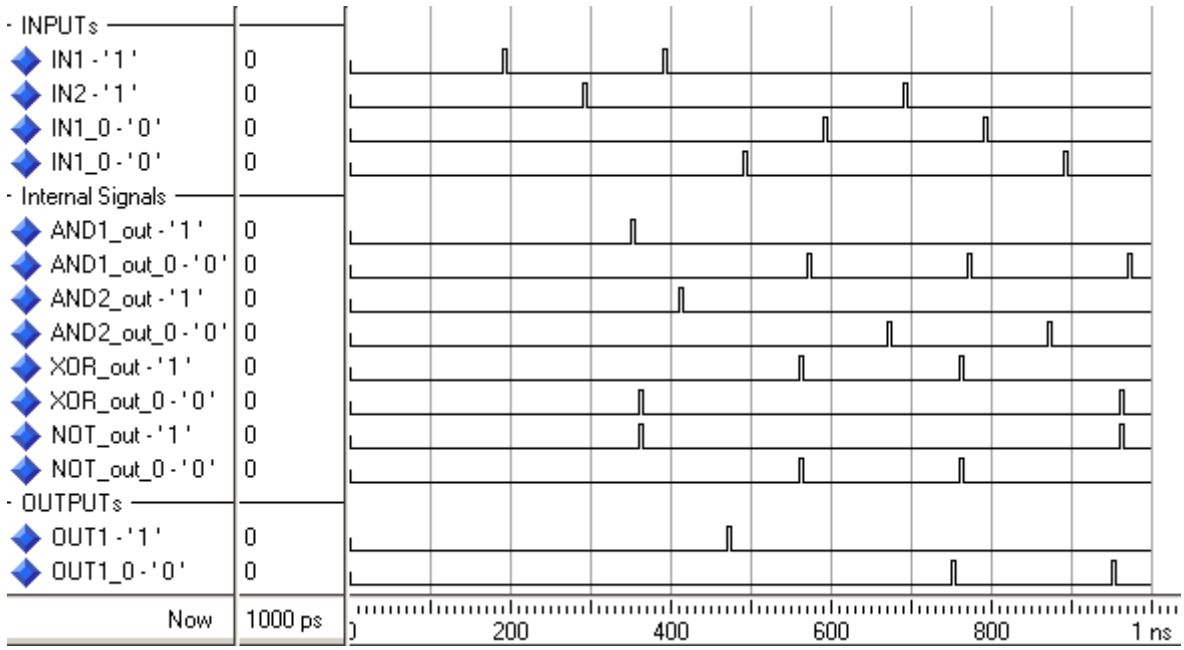

Fig. 6. Time diagram of a complex DR logic circuit described with VHDL models.

The VHDL model of an asynchronous RSFQ NAND logic element is described in detail. To test the correct operation of the described model, a ModelSim simulation was performed, and the results were compared to JSIM simulations on the same gate. The comparison demonstrates that the VHDL model is operating correctly.

The main asynchronous dual-rail RSFQ gates are described similarly. To evaluate the approach for describing asynchronous dual-rail gates with VHDL models, a complex DR logic circuit was simulated. The result showed proper operation and the ability to design asynchronous two-rail digital circuits at a high-level.

REFERENCES

1. K.K. Likharev, V.K. Semenov, RSFQ Logic/ Memory Family: A New Josephson-Junction Digital Technology for Sub- Terahertz Clock-Frequency Digital Systems, IEEE Trans. Applied Supercond., 1, 1991, 3-28.
2. J.A. Brzozowski, C.-J.H. Seger, Asynchronous Circuits, Springer-Verlag, 1995.
3. M. Maezawa, I. Kurosawa, M. Aoyagi, H. Nakagawa, Rapid Single-Flux-Quantum Dual-Rail Logic for Asynchronous circuits, IEEE Trans. Applied Supercond., 7, 1997, 2705-2708.
4. Z.J. Deng, N. Yoshikawa, S.R. Whiteley, T. Van Duzer, Data-Driven Self-Timed RSFQ High-Speed Test System, IEEE Trans. Applied Supercond., 7, 2, 1997, 3830-3833.
5. W. Chen, A.V. Rylyakov, V. Patel, J.E. Lukens, K.K. Likharev, Rapid Single Flux Quantum T-Flip Flop Operating up to 770GHz, IEEE Trans. Applied Supercond., 9, 1999, 3212-3215.
6. H. Akaike et al., Demonstration of a 120 GHz single-flux quantum shift register circuit based on a 10 kA/cm² Nb process, Superconductor Science and Technology, 19, 5, 2006, S320-S324.
7. B. Dimov, M. Khabipov, D. Balashov, C.M. Brandt, Th. Ortlepp, D. Hagedorn, F-Im Buchholz, J. Niemeyer, F.H. Uhlmann, Universal asynchronous RSFQ gate for realization of Boolean functions of dual-rail binary variables, Journal of Physics, Conference Series 43, 2006, 1183-1186.

8. B. Dimov, General Restrictions and Their Possible Solutions for the Development of Ultra-High-Speed Integrated RSFQ Digital Circuits, Wissenschaftsverlag Ilmenau, Germany, 2005.
9. D. Perry, VHDL: Programming by Example, 4th Edition, McGraw-Hill, 2002.
10. http://www.wrcad.com/freestuff.html, jsim-2.0e.tar.gz, JSIM Release 2.0e, 2014.
11. https://modelsim.informer.com/versions, ModelSim 11.0, 2012.
12. V. Todorov, Ultra-low power consumption circuits based on RSFQ technologies, Phd Thesis, TU-Sofia, 2009, (in Bulgarian).
13. V. Mladenov, V. Todorov, B. Dimov, Th. Ortlepp, F. H. Uhlmann, High-Level Design of Asynchronous RSFQ Digital Circuits, Proceedings of 51st Internationales Wissenschaftliches Kolloquium, Technische Universität Ilmenau, 2006, 149-151.