

Testing for Threshold Logic Circuits Based on Resonant Tunneling Diodes

Weidong Kuang* and Edward Banatoski

Department of Electrical Engineering, University of Texas – Pan American, Edinburg, TX 78541, U.S.A.

*Email: kuangw@panam.edu

Abstract — This paper proposes comprehensive fault models to accommodate the typical manufacturing defects in resonant tunneling diode (RTD) threshold logic integrated circuits. The defects, such as device (field-effect transistor or RTD) short or open, can be modeled as conditional stuck-at fault models. The errors due to the RTD size variations are abstracted as threshold-change fault models. A testing methodology based on the proposed fault models is presented.

Index Terms — RTD, fault model, testing

I. INTRODUCTION

Although research on threshold logic design goes back to the 1960s [1] [2] and has received varied levels of attention over the decades [3] [4], there is now a major and ongoing resurgence of interest due to the tremendous progress of nanoscale logic devices [5] [6] [7]. The scaling of CMOS device and process technology likely will end by the 6 nm node (7 nm physical channel length) by 2019 [8]. This end of scaling will be due to several concurrent fundamental and practical limits related to transistor operation and manufacturability. Recent research demonstrates that nanoscale devices offer the potential to extend CMOS devices [9].

Resonant tunneling diode (RTD) and quantum cellular automata (QCA) are two promising nanoscale devices for future digital systems. RTDs are the most mature type of quantum-effect devices which exhibit negative differential resistance (NDR) at room temperature. In [10], the monstable-bistable transition logic element (MOBILE) is introduced to implement threshold logic by combining RTDs and heterostructure field-effect transistors (HFETs). Quantum cellular automata, an array of quantum dots, can be used to implement a three-input majority gate [11], from which more complex circuits can be built [12]. A tool called TELS [7] has been developed recently to synthesize combinational threshold networks. In [13], an automatic test pattern generation framework for combinational threshold logic networks is proposed to kick off the testing research on threshold logic. There has been no extensive work done to date in the area of testing that the authors are aware of. However, it is critical that testing methodology be established for digital systems based on threshold logic.

This paper presents fault models associated with the defects that are most likely to occur in RTD-HFET threshold gates. Section II introduces the mathematic definition of threshold logic gates and their implementation based on RTDs. The fault models are proposed in Section III. Section IV presents the test generation method for these threshold logic networks. Section V summarizes this paper and proposes the future work.

II. THRESHOLD LOGIC GATE BASED ON RTDS

A threshold logic gate is defined as a logic gate in which each variable input x_i ($i = 1, 2, \dots, n$) is 1 or 0 and for which there is a set of real number w_1, w_2, \dots, w_n and t such that the output f of the gate is

$$f(\bar{X}^{(j)}) = 1$$
$$\text{for } \sum_{i=1}^n w_i x_i^{(j)} \geq t \quad (j = 1, 2, \dots, \alpha)$$

and

$$f(\bar{X}^{(j)}) = 0$$
$$\text{for } \sum_{i=1}^n w_i x_i^{(j)} < t \quad (j = \alpha + 1, \dots, 2^n)$$

Here w_1, w_2, \dots, w_n are called weights, t is the threshold and j is the index of input vectors. The switching function $f(\bar{X})$ is called a threshold function. The structure of a threshold logic gate is denoted by the weight-threshold vector $[w_1, w_2, \dots, w_n; t]$.

A threshold logic gate can be physically implemented through various technologies. A comprehensive survey of threshold logic implementations is given in [5]. The discussion about fault models in this paper is limited to the RTDs implementation. Before presenting the fault models for threshold logic gates, let us review the threshold logic gate implementation based on RTDs.

To illustrate threshold logic gate design based on RTDs, Fig. 1(a) shows a threshold logic gate with two positive weighted inputs (x_1, x_2) and two negative weighted inputs (x_3, x_4). The values of weights (w_1, w_2, w_3, w_4)

and threshold t are determined by the relative sizes of the RTDs. The symbol of a general threshold logic gate is showed in Fig. 1(b). When the power clock V_{clk} is increased and exceeds a certain switching voltage, the gate evaluates the output. The resulting logic output depends on the difference of the total peak currents between the top half circuit and the bottom half circuit, which is determined by the sizes of RTDs and the logic value of each input. A detailed description of the principle of this circuit can be found in [14].

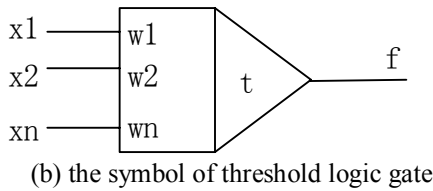
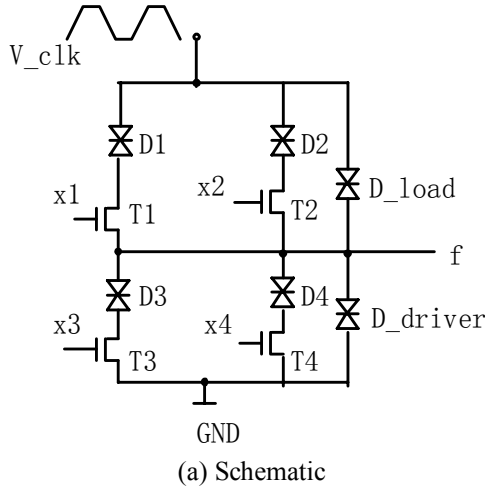


Fig.1 RTD-based implementation of threshold logic gate.

III. FAULT MODELS

In order to develop a testing methodology for logic circuits, it is important to develop a proper fault model. A fault model is a representation of a defect at the abstracted function level while a defect is the unintended difference between the implemented hardware and its intended design. Due to the differences of devices and circuit configurations between threshold logic gates and conventional CMOS gates, current fault models for CMOS gate can not represent the faulty behavior for these threshold gates. Therefore, new fault models have to be developed.

First, it is necessary to identify the transistor-level defects that are most likely to occur in a threshold logic gate. The typical defects of HFETs and RTDs in Fig. 1 include cuts and shorts. For HFETs, *cut* means that there is always no current between the source and the drain, while *short* means that the source and the drain are always connected together, regardless of the voltage of the gate. For RTDs, *cut* means that the two terminals are totally disconnected which *short* means that the equivalent resistance between the two terminals is

zero. At logic gate level, the malfunctions caused by these defects are different from those caused by similar defects in CMOS gates. Furthermore, an RTD area variation, caused by the increasing impact of lithography and etching on lateral dimensions [15], will lead to peak current fluctuations which consequently cause the change of the logic threshold of the gate.

Table 1. Fault models due to device defects

Fault model	Defect	Examples (fault: defect)
Input stuck-at-0 (SA0)	HFET or/and RTD cut	x1 SA0: T1 or/and D1 cut x4 SA0: T4 or/and D4 cut
Input stuck-at-1 (SA1)	HFET short	x1 SA1: T1 short x3 SA1: T3 short
Output stuck-at-0 (SA0)	Both HFET and RTD short in negative weight branch; or D_driver short	f SA0: T4 and D4 short f SA0: D_driver short
Output conditional stuck-at-0 (CSA0)	Only RTD short in negative weight branch	f CSA0 x3=1: D3 short with x3=1
Output stuck-at-1 (SA1)	Both HFET and RTD short in positive weight branch; or D_load short	f SA1: T2 and D2 short f SA1: D_load short
Output conditional stuck-at-1 (CSA1)	Only RTD short in positive weight branch	f CSA1 x2=1: D2 short with x2=1
Threshold increased by 1	RTD area variations*	t+1
Threshold decreased by 1	RTD area variations*	t-1

*Note that different combinations of RTD sizes may result in the same fault – threshold increase or threshold decrease.

All fault models associated with the above defects are proposed in Table 1. These fault models are different the models used in CMOS gates. In CMOS gates, the input-output behavior based on the MOSFET cut or short defects can not be exactly represented by the stuck-at-fault model. Two test vectors are required to test a MOSFET cut fault while a measurement of the device current will detect a MOSFET short fault for CMOS gates. For threshold gates in Fig. 1, any cut of HFET or RTD in an input branch can be modeled as a single input stuck-at-0. A short of HFET associated with input can be modeled as a single input stuck-at-1. An RTD short in input branch is modeled as an output conditional stuck-at fault with the input set to 1. Depending on the amount of RTD area variations, the threshold may increase/decrease by more than one. For simplicity, one unit change on threshold is assumed in this paper.

IV. METHODOLOGY OF TEST GENERATION

A test vector for a given fault in a network of threshold gates is expected to be generated automatically. D-algorithm [16] can be used for this purpose even though new faults could exist in the network, such as the threshold change. For simplicity, we make the following two assumptions: 1) Single fault – only one fault occurs at a time; 2) no interconnection faults – only faults associated with device defects are considered. The circuit in Fig.2 will be used as an example to illustrate D-algorithm for automotive test generation. This circuit, consisting of four threshold gates, produces the sum of a full adder.

A) Primitive D-Cubes of Failure

A Primitive D-Cube of Failure (PDCF) is a fault model for a primitive, which is used in D-algorithm. The primitives to be addressed in this paper are threshold logic gates. Boolean cube theory can be employed to create a PDCF for a primitive with on possible fault listed in Table 1. Specifically, the following general method for find a PDCF is suggested:

- 1) Create a cover consisting of extremals for both the fault-free and faulted gates.
- 2) Intersect members of the set of 0-points of the faulted gate, f_0 with members of the set of 1-points of the fault-free gate, p_1 .
- 3) Intersect members of the set of 1-points of the faulted gate, f_1 with members of the set of 0-points of the fault-free gate, p_0 .

Example 1: Consider the gate G3 in Fig. 2 with a fault of threshold change-to-1. A cover for both fault-free and faulted circuits is shown in Fig. 3. Intersecting f_0 with p_1 results in an empty set. Intersecting f_1 with p_0 results in three cubes $(0, 1, 0, \overline{D})$, $(1, 0, 0, \overline{D})$ and $(1, 1, 1, \overline{D})$. This implies that one of these three input vectors $(0, 1, 0)$, $(1, 0, 0)$ and $(1, 1, 1)$ generate a discrepancy at the primitive output between fault-free and faulted circuits. This discrepancy is expected to propagate to the primary output S for testing purpose.

B) Propagation D-Cubes

The gate associated with a PDCF generates a sensitized signal, D or \overline{D} . This sensitized signal should propagate to an observable output of the circuit. It is also assumed, in keeping with the single-fault assumption, that the primitive through which the sensitized signal is propagating be fault-free. A propagation D-cube of the fault-free primitive (for example, G4 in Fig. 2) must be created to propagate the fault elsewhere (for example, G3 in Fig. 2). The propagation D-cube of any threshold logic gate can be derived by the following steps:

- 1) Create a cover consisting of extremals for the fault-free threshold logic gate.
- 2) Intersect members of the set of 0-points (p_0) with members of the set of 1-points (p_1).

- 3) Intersect members of the set of 1-points (p_1) with members of the set of 0-points (p_0).

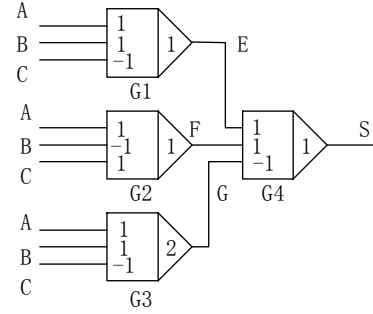


Fig. 2 A circuit for the sum of full adder

fault-free				fault: threshold change to 1			
A	B	C	G	A	B	C	G(faulted)
x	0	0	0	0	0	0	0
x	x	1	0	x	0	1	0
0	x	0	0	0	x	1	0
1	1	0	1	x	1	0	1
				1	x	0	1
				1	1	1	1

Fig. 3 A cover for the gate G3 in Fig. 2

Example 2: the propagation D-cube of G4 in Fig.2. A cover of G4 is given in Fig. 4(a). As shown in Fig. 4(b), nine propagation D-cubes are obtained by intersecting p_0 with p_1 . There are actually 18 propagation D-cubes. The other nine can be obtained by intersecting p_1 with p_0 , or changing D to \overline{D} and \overline{D} to D in Fig. 4(b). In actual practice it is often necessary to restrict the propagation D-cube tables to contain only those propagation D-cubes having a single D or \overline{D} among the inputs. It is obvious that the propagation D-cubes $(1, 0, \overline{D}, D)$ and $(0, 1, \overline{D}, D)$ can be used to propagate \overline{D} at input G,

E	F	G	S
1	\overline{D}	1	\overline{D}
1	0	D	\overline{D}
x	\overline{D}	D	\overline{D}
\overline{D}	1	1	\overline{D}
\overline{D}	x	D	\overline{D}
0	1	D	\overline{D}
\overline{D}	D	D	\overline{D}
\overline{D}	0	0	\overline{D}
0	\overline{D}	0	\overline{D}

Fig.4(a) A cover of G4

intersecting p_0 with p_1

(b) 9 propagation D-cubes

C) Justification

To activate the propagation D-cubes of G4, specific values should be assigned at the inputs of G1 and G2. This can be done by properly intersecting covers of G1 and G2. The possible values obtained should be consistent with the PDCF of G3.

Example 3: To activate the propagation D-cube (1, 0, \overline{D} , D), the inputs of G1 and G2 can be determined by intersecting p1 of G1 with p0 of G2 (shown in Fig. 5), resulting in (A, B, C) = (x, 1, 0). Similarly, to activate the propagation D-cube (0, 1, \overline{D} , D), the inputs of G1 and G2 can be found by intersecting p0 of G1 with p1 of G2 (shown in Fig. 5), resulting in (A, B, C) = (x, 0, 1). Finally, the test for the fault of G3 specified in *Example 1* is obtained by intersecting these (A, B, C) with those (A, B, C) in PDCFs in *Example 1*. In this case, only one test for the fault of threshold change-to-1 in G3 exists, which is ABC=010.

Gate G1					Gate G2				
A	B	C	E		A	B	C	F	
x	0	1	0	} p0	x	1	0	0	} p0
0	x	1	0		0	1	x	0	
0	0	0	0		0	0	0	0	
1	1	1	1	} P1	1	1	1	1	} P1
1	x	0	1		1	0	x	1	
x	1	0	1		x	0	1	1	

Fig. 5 Cover for G1 and G2

V. SUMMARY AND FUTURE WORK

The fault models, accommodating various manufacturing defects in RTD threshold logic gates, have been proposed, and the corresponding test generation method is presented through an example. The future work includes formalization of the test generation and software implementation of automatic test pattern generation (ATPG).

REFERENCES

- [1] R. O. Winder, "Threshold logic," Ph.D. dissertation, Princeton University, NJ, 1962.
- [2] B. D. Carroll, "Two-level realization of switching functions with minimum number of threshold gates," Ph.D. dissertation, University of Texas at Austin, 1969.
- [3] S. Muroga, *Threshold Logic and its Applications*. New York, NY: John Wiley, 1971.
- [4] A. L. Oliveira, *et al.*, "LSAT - An Algorithm for the Synthesis of Two-Level Threshold Gate Networks," *Proc. of Int. Conf. of CAD*, PP. 130-133, 1991.

- [5] Valeriu Beiu, *et al.*, "VLSI Implementations of Threshold Logic - A Comprehensive Survey," *IEEE Trans. on Neural Networks*, Vol. 14, no. 5, Sept. 2003.
- [6] Maria J. Avedillo, *et al.*, "A threshold Logic Synthesis Tools for RTD Circuits," *Proc. the EUROMICRO Systems on Digital System Design*, pp. 624-627, 2004.
- [7] Rui Zhang *et al.*, "Synthesis and Optimization of Threshold logic Net works with Application to Nanotechnologies," *Proc. the Design, Automation and Test in Europe Conference*, pp. 904-909, 2004.
- [8] "Semiconductor Industries Association Roadmap." <http://public.itrs.net>.
- [9] D. Goldhaber-Gordon, *et al.*, "Overview of nanoelectronic devices," *Proc. IEEE*, vol. 85, no. 4, pp. 521-540, April 1997.
- [10] K. Maezawa, *et al.*, "High-speed and low-power operation of a resonant tunneling logic gate MOBILE," *IEEE Electron Device Lett.*, vol. 19, no. 3, pp. 80-82, March 1998.
- [11] C. S. Lent, *et al.*, "A device architecture for computing with quantum dots," *Proc. IEEE*, vol. 85, pp. 541-557, April 1997.
- [12] Rumi Zhang, *et al.*, "A method of majority logic reduction for quantum cellular automata," *IEEE Trans. on Nanotechnology*, vol. 3, no. 4, December 2004.
- [13] P. Gupta, *et al.*, "An Automatic Test Pattern Generation Framework for Combinational Threshold Logic Networks," *Proc. of the IEEE International Conference on Computer Design*, 2004.
- [14] C. Pacha, *et al.*, "Threshold logic circuit design of parallel adders using resonant tunneling devices," *IEEE Trans. on VLSI systems*, vol. 8, no. 5, October 2000.
- [15] W. Prost, *et al.*, "Manufacturability and robust design of nanoelectronic logic circuit based on resonant tunneling diodes," *Int. J. Circ. Appl.*, 2000; 28: 537-552.
- [16] Alexander Miczo, "Digital Logic Testing and Simulation," Wiley-Interscience, 2003.