2006 IEEE International Conference on
Systems, Man, and Cybernetics
October 8-11, 2006, Taipei, Taiwan

Test Generation and Site of Fault for Combinational Circuits Using Logic Petri Nets

Jui-I Tsai, Ching-Cheng Teng, and Ching-Hung Lee, *Member, IEEE*

Abstract— **In this paper, we propose a novel Petri Net model for solving test generation and site of fault and fired logical value for combinational circuits. In order to improve the logic fault efficiency, the transitions of general Petri Nets (PNs) are modified according to the critical of truth table, called Logic Petri Net LPN. The LPN model can transfer complexity circuit problem to a local adjacent place and transition relational problem. Therefore, the site of fault and fired logical value problem is simplified and clearly. The LPN model has the properties of Boolean algorithm, collapsing fault with clear physical concepts, fast calculation speed, and high veracity. The approach contains site of a fault and fired logical value reasoning algorithm and test vector generation reasoning algorithm. Two examples are shown to demonstrate the effectiveness of our approach.**

I. Introduction

Modeling plays a central role in design, fabrication, and testing of a digital system [1]. Many techniques have been presented for finding the exact sites of fault in combinational circuits [1, 6, 10]. Most of them have been presented by functional modeling at the logic level [1, 6]. Logical faults represent the effect of physical faults on the behavior of modeled system. In general, structural fault models assume that components are fault-free and only their interconnections are affected. Typical faults affecting interconnections are shorts and opens. The fundamental fault model is a stuck-at-fault, which implies the fault effect to be a line segment stuck at logic 0 or 1 (stuck at 0 or stuck at 1). The corresponding logical fault consists of the signal being stuck at a fired logical value x ($x \in \{0,1\}$), and it is denoted by s-a-0 or s-a-1. Note how a single logical fault, namely the line l stuck at $a \in \{0,1\}$, can represent many totally different physical faults: l open, l shorted to power or ground, and internal fault in the component driving l that keeps l at the logic value a [1,6].

Petri Nets were originally proposed by Carl Adam Petri and based on the concept that relationships between the components of a system, which exhibits asynchronous and concurrent activities, could be represented by a net [4, 5,7-9, 10, 12, 14]. Therefore, PN is an excellent tool for modeling asynchronous concurrent system such as computer systems and manufacturing systems, as well as power protection systems [4, 5, 7-9, 10, 12, 14]. In this paper, based on the truth table of combinational circuits, the Petri Nets are modified to solve the test generation and sites of fired value.

In order to improve the logic fault efficiency, the transitions of general Petri Nets are modified according to the critical of truth table, called Logic Petri Net LPN. The LPN model can transfer complexity circuit problem to a local adjacent place and transition relational problem. Therefore, the site of fault and fired logical value problem are simplified and clearly. The LPN model has the properties of Boolean algorithm, collapsing fault with clear physical concepts, fast calculation speed, and high veracity. The approach major contains site of a fault and fired logical value reasoning algorithm and test vector generation reasoning algorithm.

The paper is organized as follows. In Section II, the LPN model is introduced by critical of truth table, and the description of Boolean algorithm and fault collapsing for LPN model. Section III describes the site of a fault and fired logical value reasoning algorithm. Section IV describes test generation reasoning algorithm. Finally, conclusion is given in Section V.

II. The Model and Properties of Logic Petri Net

The purpose of the development of LPN model is that the LPN model holds clear logical property in IC testing. Firstly, the simplest way to represent a combinational circuit is by its truth table. Assuming binary input variable, a circuit realizing a function $X(x_1,x_2,...,x_n)$ of n variables requires a table with 2^n entries. The data structure representing a truth table is usually an array U of dimension 2^n. We arrange the input combinations in their increasing binary order. Then, we obtain $U(0)=X(0,0,...,0)$, $U(1)=X(0,0,...,1)$, ..., $U(2^n-1)=X(1,1,...,1)$. The truth table can be divided into critical and no-critical part. For AND gate, the corresponding critical value is $x_1 \in 1$, $x_2 \in 1$, and $U(2^2-1)=X(1,1)=1$. That is, if $X(x_1,x_2)=1$ then $x_1 \in 1$ and $x_2 \in 1$; no-critical value of AND gate is $x_1 \notin 1$ or $x_2 \notin 1$ and $X(x_1,x_2) \neq 1$, i.e., if $X(x_1,x_2) \neq 1$ then $\{x_1,x_2\} \not\subset \{1,1\}$.

In this paper, we embed the critical value of truth table into transition of PN to develop LPN model. This special transition is called "logic transition". Table I describes the LPN model corresponding to the truth table. Clearly, the LPN model is matched properties of Boolean algorithm and fault collapsing. Based on the embed critical value of truth table in

This work was supported in part by the National Sciences Council under Grant NSC-94-2213-E-155-039.

Jui-I Tsai and Ching-Cheng Teng are with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu, 300, Taiwan, R.O.C. +886-3-5712121, ext: 54345; Fax: +886-3-571599855; e-mail: ccteng@cn.nctu.edu.tw).

Ching-Hung Lee is with the Department of Electrical Engineering, Yuan Ze University, Chung-li, Taoyuan 320, Taiwan, R.O.C. (e-mail: chlee@saturn.yzu.edu.tw).

LPN model, the Boolean algorithm and fault collapsing in LPN representation are shown in Tables II and III.

In general used representation, the LPN model structure can be defined as follows:

$$LPN = (P, T, D, I, O, i, o, f, b, \alpha, m_0)$$

where

$p = \{p_1, p_2, ..., p_m\}$: finite set of places,

$T = \{t_1, t_1, ..., t_n\}$: finite set of logic transitions by critical value of truth table,

$D = \{d_1, d_1, ..., d_m\}$: finite set of propositions,

$P \cap T \cap D = \Phi$,

$|P| = |D|$,

$I : T \to P^{\infty}$: input function (a mapping from transitions to bags of places),

$O : T \to P^{\infty}$: output function (a mapping from transitions to bags of places),

$i : T \to \{\bullet, \circ\}$: logical value of a input transitions,

$o : T \to \{\bullet, \circ\}$: logical value of a output transitions,

$f : p \to i(t_j)$: logical value of a input transitions (a forward mapping from place p to input critical value $i(t_j)$),

$b : p \to o(t_j)$: logical value of a output transitions (a backward mapping from place p to output critical value $o(t_j)$),

$\alpha : P \to \{\bullet, \circ\}$: logic value of place (a mapping from place to logic value, $\alpha(p_i) = \{\bullet, \circ\}$, i.e., $\bullet$ denotes logic 1 and $\circ$ denotes logic 0).

m_0 : Initial mark

Example 1: Herein, the description of LPN model for NOT gate is introduced, as the following Figure 1.

p_1 , p_2 : place, t_k : transition, d_1 : stuck-at-1, $I(t_k) = p_1$, $O(t_k) = p_2$, $i(t_k) = \bullet$, $o(t_k) = \circ$, $f : p_1 \to i(t_k) = \bullet$, $b : p_2 \to o(t_k) = \circ$, $\alpha : P_1 \to \bullet$.

III. A Fault Logic Reasoning Algorithm for Sites and Fired Logic Value

Using the LPN model, we proposed an algorithm to determine sites of a fault fired logical value at combinational circuits.

Algorithm 1

Step 1: Transfer the circuit into the LPN circuit.

Step 2: List the table for transitional state of forward of place $f(p_i)$ and backward of place $b(p_i)$.

Step 3: If $b(p_i) = \phi$ and $f(p_i) \neq \phi$ then place p_i is the primary input, while line of a primary input is fired logical value $f(p_i)$, and it is denoted by $D(p_i)$ =s-a- $f(p_i)$.

Step 4: If $b(p_i) \neq \phi$ and $f(p_i) = \phi$ then place p_i is the primary output, while line of a place of primary output is fired logical value $\overline{b(p_i)}$, and it is denoted by $D(p_i)$ = s-a- $\overline{b(p_i)}$,

Step 5: If $b(p_i) \neq \phi$, $f(p_i) \neq \phi$, and $b(p_i) \neq f(p_i)$ then line of a place p_i is fired logical value $f(p_i)$, and it is denoted by $D(p_i)$ = s-a- $f(p_i)$, else no site of fault.

Using Algorithm 1, the site of fault and fired logic vales can be found. An example of simple circuit is described below.

Example 2: A simple combinational circuit with AND and OR gates are used here (as shown in Fig. 2 (a)).

Step 1: Transfer the combinational circuits to LPN circuit, as Fig. 2 (b).

Step 2: List the transitional state as Table IV.

Step 3: Place p_1 , p_2 , p_3 , and p_4 are primary inputs since $b(p_1) = b(p_2) = b(p_3) = b(p_4) = \phi$. $D(p_1)$, $D(p_2)$, $D(p_3)$ and $D(p_4)$ are stuck-at-1 by $f(p_1) = f(p_2) = f(p_3) = f(p_4) = 1$.

Step 4: Place p_7 is primary output since $f(p_7) = \phi$. $D(p_7)$ is stuck-at 1 since $b(p_7) = 0$.

Step 5: p_5 , p_6 are not terminal place since $b(p_i) \neq \phi$, $f(p_i) \neq \phi$ and $b(p_i) \neq f(p_i)$, then $D(p_5)$ and $D(p_6)$ are stuck-at 0 since $f(p_5) = f(p_6) = 0$.

By the results of above discussion, we can determine the fired logical values (struck-at-fault) of places p_1, ..., p_7 as Table IV.

IV. Forward and Backward Reasoning Algorithm

By the definitions of literature [2, 3], immediate reachability set, reachability set, immediate backward incidence set, backward incidence set, and adjacent place, a forward and backward reasoning algorithm is proposed for test generation of combinational circuits.

Firstly, the PN model for describing the definitions is shown in Fig. 3. For Fig. 3 (a), let t_i and t_j be general transitions, and p_a , p_b , p_c be three places. If $p_a \in I(t_i)$, $p_b \in O(t_i)$, $p_b \in I(t_j)$ and $p_c \in O(t_j)$, then we have

(1) Place p_b is immediately reachable from place p_a ,

(2) Place p_c is immediately reachable from place p_b ,

(3) Place p_a is an immediately backward incidence place of place p_b ,

(4) Place p_b is an immediately backward incidence place of place p_c ,

(5). Places p_b and p_c are reachable from place p_a ,

(6) Places p_a and p_b are backward incidence places of place p_c .

The reachability relationship is the reflexive closure of the immediately reachable relationship. The backward incidence relationship is the reflexive closure of the immediately backward incidence relationship.

The set of places that is immediately reachable from a place p_a is called the immediately reachability set of p_a and is denoted by $IRS(p_a)$. The set of places that is reachable from a place p_a is called the reachability set of p_a and is denoted by $RS(p_a)$. The set of places that contains immediate backward places of p_b is called the immediate backward set of p_b and is denoted by $IBIS(p_b)$. The set of places which contains backward incidence places of p_c is called the backward incidence set of p_c and is denoted by $BIS(p_c)$.

For Fig. 3(b), let t_k be a transition, p_{a1} and p_{a2} be places. If place $p_{a1} \in I(t_k)$ and place $p_{a2} \in I(t_k)$ then p_{a1} and p_{a2} are called adjacent places with respect to t_k.

Next, we have the following forward and backward reasoning algorithm.

Algorithm 2

Step 1: Transfer the combinational circuits to LPN circuit.

Step 2: List the table for immediate reachability set, reachability set, immediate backward incidence set, backward incidence set, and the table for set of adjacent places Ap_{jk} for each place p_j.

Step 3: Find the primary inputs p_i (IBIS(p_i)= ϕ) and primary outputs (IRS(p_j)= ϕ).

Step 4: Select a site of fault and fired logic value from Table IV, activate it and propagate to primary output, i.e., generate a fault effect and sensitized path. Initial mark m_0 are comprised by logical value of fault effect and logical value of a propagation of all adjacent place of sensitized path (i.e., $f: AP_{jk} \rightarrow i(t_i)$ is logical value of a input transitions of all adjacent place of sensitized path).

Step 5: Find the test pattern by initial mark backtracing path and hold the fault effect as below.

(1) Proposition of place p_j - $D(p_j)$ generates a fault effect and forward propagates the error through t_i to proposition of immediate reachability place p_k - $D(p_k)$ until to the primary output p_o. The change of the state of $D(p_k)$ is depended on the input value $i(t_i)$ and output value $o(t_i)$ of transition relation. If $i(t_i) = o(t_i)$ then $D(p_k) = D(p_j)$. Otherwise, $D(p_k) = \overline{D(p_j)}$. Details of $i(t_i)$ and $o(t_i)$ can be found in Table IV.

(2) At the same time, the proposition of place p_j possess a fault effect. The token of adjacent place Ap_{jk} is equal to a forward mapping from Ap_{jk} to $i(t_j)$, i.e., $\alpha(AP_{jk}) = i(t_i)$, the sensitized path is hold. Then we select a back path of immediate backward incidence place Ap_{jk} through transition t_b ($IBIS(Ap_{jk})$) to primary input p_{in}. If $\alpha(Ap_{jk}) = o(t_b)$) then $\{\alpha(p_b)\} = \{i(t_b)\}$. Otherwise, $\{\alpha(p_b)\} \subset \{i(t_b)\}$.

(3) Find the test generation of back path. Place p_j propagate back through transition t_b to p_i until to primary input p_{in}. If $\alpha(p_j) = o(t_b)$ then $\{\alpha(p_b)\} = \{i(t_b)\}$. Otherwise, $\{\alpha(p_b)\} \subset \{i(t_b)\}$.

Step 6: If we can find a token of primary input $\alpha(p_{in})$ set and generate a fault effect then fault f is detectable and test generation is set of a primary input token $\alpha(p_{in})$.

Finally, we use an example to illustrate the LPN reasoning process for test generation.

Example 3: Determine test generation of sat-at-1 at p_4 and sat-at-0 at p_6 in combinational circuit, as shown in Fig. 4(a).

Case (a) $D(p_6)$: sat-at-0.

Step 1: Transfer the combinational circuits to LPN circuits as shown in Fig. 4 (b).

Step 2: List the table for immediate reachability set, reachability set, immediate backward incidence set, backward incidence set table and the table for set of adjacent places Ap_{jk}, as Table V and VI, respectively.

Step 3: Find the primary input $p_{in} = \{p_1, p_2, p_3, p_4\}$ and the primary output $p_o = \{p_7\}$.

Step 4: Select a $D(p_6)$ (which is sat-at-0), $\alpha(p_6) = 1$ is generate a fault effect and $\alpha(p_5) = 0$ is logical value of a propagation of all adjacent place of sensitized path. So $m_0 = \{\alpha(p_6) = b(p_6) = o(t_2) = 1, \alpha(Ap_{67}) = \alpha(p_5) = i(t_3) = 0\}$.

Step 5:

(1) $D(p_6)$ propagates the error through t_3 to $D(p_7) = 1/0$ since $i(t_3) = o(t_3)$.

(2) $\alpha(AP_{67}) = \alpha(p_5) = i(t_3) = 0$, sensitized path is hold. $\alpha(p_5) \neq o(t_1) = 0$ implies $\{\alpha(p_1), \alpha(p_2)\} \neq \{i(t_1), i(t_1)\} = \{1,1\}$, i.e., $\{\alpha(p_1), \alpha(p_2)\} = \{0,0\} or \{0,1\} or \{1,0\}$.

(3) $\alpha(p_6) = b(p_6) = o(t_2) = 1$ implies $\{\alpha(p_3), \alpha(p_4)\} = \{i(t_2), i(t_2)\} = \{1,1\}$.

Step 6: $D(p_6)$: sat-at-0 is detectable. Then, the test generation is $\alpha(p_{in}) = \{\{\alpha(p_1), \alpha(p_2)\} \subset \{1,1\}, \{\alpha(p_3), \alpha(p_4)\} = \{1,1\}\}$.

Case (b) $D(p_4)$: sat-at-1

Step 1: Transfer the combinational circuits to LPN circuit as Fig. 4 (b).

Step 2: List the table for immediate reachability set, reachability set, immediate backward incidence set and backward incidence set table and the table for table set of adjacent places Ap_{jk}, as Table V and VI, respectively.

Step 3: Find the primary input $p_{in} = \{p_1, p_2, p_3, p_4\}$ and the primary output $p_o = \{p_7\}$.

Step 4: Select $D(p_4)$ (sat-at-1) and $\alpha(p_4) = 0$ generate a fault effect. $\alpha(p_3) = 1$, $\alpha(p_5) = 0$ are logical value of a

propagation of all adjacent place of sensitized path. So $m_0 = \{\alpha(p_4) = \overline{f(p_4)} = \overline{i(t_2)} = 0, \alpha(Ap_{46}) = \alpha(p_3) = i(t_2) = 1, \alpha(Ap_{67}) = \alpha(p_5) = i(t_3) = 0\}$

Step 5:

(1) $D(p_4)$ (sat-at-1) and $\alpha(p_4) = 0$ Since $i(t_2) = o(t_2)$, $D(p_4)$ propagates the error through t_2 to $D(p_6) = 0/1$. and $i(t_3) = o(t_3)$, $D(p_4)$ propagates the error through t_2 to $D(p_7) = 0/1$.

(2) $\alpha(AP_{46}) = \alpha(p_3) = i(t_2) = 1$.

(3) $\alpha(AP_{67}) = \alpha(p_5) = i(t_3) = 0$, sensitized path is hold. The result is similar to (2) of case (a)- Step 5. Thus, $\{\alpha(p_1), \alpha(p_2)\} \neq \{i(t_1), i(t_1)\} = \{1,1\}$, i.e., $\{\alpha(p_1), \alpha(p_2)\} = \{0,0\}$ or $\{0,1\}$ or $\{1,0\}$.

Step 6: $D(p_4)$: sat-at-1 is detectable and test generation is $\alpha(p_{in}) = \{\{\alpha(p_1), \alpha(p_2)\} \not\subset \{1,1\}, \{\alpha(p_3), \alpha(p_4)\} = \{1,0\}\}$.

The comparison between LPN model and traditional method (by Kirkland and Mercer [15]) in test generation for combinational circuit is shown in Fig. 5. The major differences are described below. (1) LPN approach is parallel processing, i.e., LPN approach has less operational time than [15]; (2) every back tracing path of LPN is shorter than [15], i.e., complexity of determining test generation LPN is easier; (3) LPN approach needs larger memory than [15], i.e., cost using LPN approach will increase.

V. Conclusion

For solving test generation and site of fault in combinational circuits, we have proposed a so-called Logic Petri Net (LPN) model. The LPN model embeds critical of truth table into transition of Petri Net with clear physical concepts, fast calculation speed and high veracity. It first transfers a complexity circuit problem to a local adjacent place and transition relational one. Thus, the site of fault and fired logical value problem is simplified. Several algorithms were presented for obtaining the test pattern and improved the calculation speed. Two examples were shown to demonstrate the effectiveness of LPN model.

References

[1] M. Abramovici, M.A Breuer, and A.D.Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, New York, 1990.

[2] S. M. Chen, J.S. Ke and J.F. Chang, "Knowledge representation using fuzzy Petri nets," *IEEE Trans. Knowl. Data Eng.*, vol. 2, pp. 311-319, Sept. 1990.

[3] S. M. Chen, "Fuzzy backward reasoning using fuzzy Petri nets," *IEEE. Trans. Syst. Man, Cybern. – Part B: Cybernetics.*, vol. 30, no. 6, pp. 846-856, December 2000.

[4] J.K. Peterson, *Petri Nets, Theory and The modeling of Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[5] S. Y. Huang, "VLSI Testing," Handout of Electrical Engineering of Nation Tsing Hua Universion, 1999.

[6] C. G. Looney, "Logical Control via Boolean Rule Matrix Transformations," *IEEE. Trans. Syst. Man, Cybern.*, vol. SMC-17, no. 6, pp. 1077-1082, Nov./Dec. 1987.

[7] W. Reisig, *Petri Nets, An Introduction*, Springer Verlag, 1985.

[8] K. Jensen, *Colored Petri Nets, Basic Concepts, Analysis Methods and Practical Use*, Springer Verlag, 1993.

[9] C. Girault, R. Valk, *Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications*, Springer Verlag, 2002.

[10] R. David and H. Alla, "Petri Nets for Modeling of Dynamic Systems- A Survey," Automatica,Vol.30, No. 2, pp.175-202,1994.

[11] J. Desel, W. Reisig, G. Rozenberg, *Lectures on Concurrency and Petri Nets, Advances in Petri Nets*, Lecture Notes in Computer Science, Vol. 3098, Springer Verlag, 2004.

[12] K. L. Lo, H. S. Ng, and J. Trecat, "Power Systems Fault Diagnosis Using Petri Nets," *IEE Proc. Generation, Transmission, and Distribution*, Vol. 144, No. 3, pp. 231-236, 1997.

[13] K. L. Lo, H. S. Ng, D. M. Grant, and J. Trecat, "Extended Petri Net Models for Power Systems Fault Diagnosis for Substation Automation," *IEE Proc. Generation, Transmission, and Distribution*, Vol. 146, No. 3, pp. 229-234, 1999.

[14] T. Murata, "Petri Nets: Properties, Analysis, and Applications, *Proceedings of IEEE*, Vol. 77, No. 4, pp. 541-580, 1989.

[15] T. Kirkland and M.R. Mercer, "Algorithms for Automatic Test-pattern Generation," *IEEE Design and Test of Computers*, Vol. 5, No. 3, pp. 43-55, 1988.

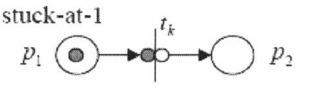


Figure 1. Logic Petri Net model for NOT gate.

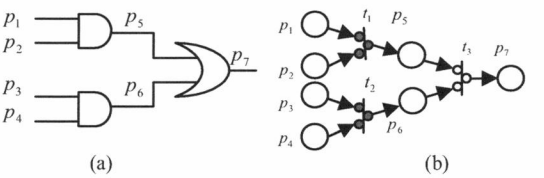


Figure 2. (a) Combinational circuit; (b) LPN circuit

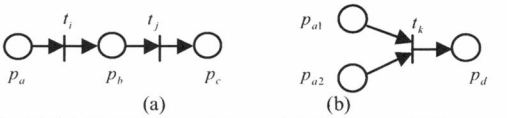


Figure 3. (a) Petr Net for immediate reachability, reachability, immediate backward incidence, and backward incidence sets (b) Petr Net for adjacent place.

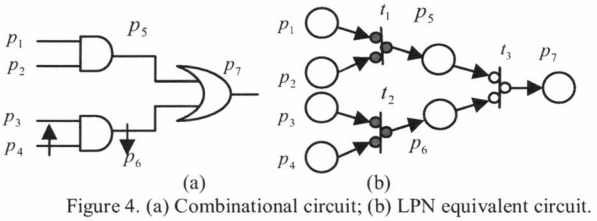


Figure 4. (a) Combinational circuit; (b) LPN equivalent circuit.

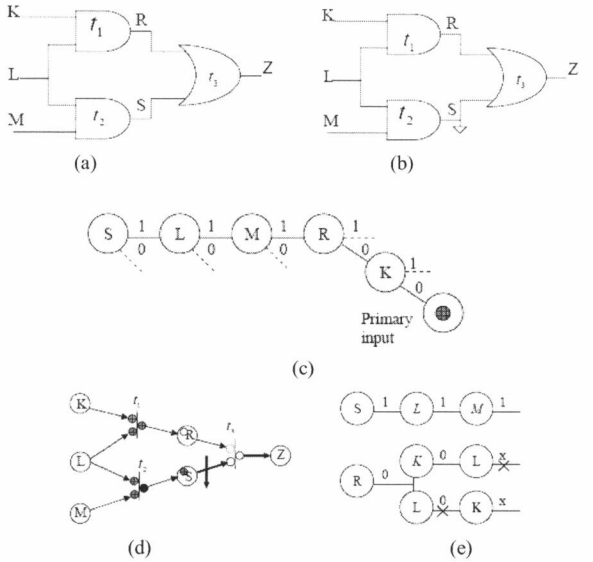


Figure 5. The comparison between LPN model and traditional method [15]: (a) A sample good circuit; (b) A faulty circuit; (c) The search graph for locating the fault; (d) A faulty circuit of LPN; (e) The search graph for locating the fault of LPN.

Table I
Truth table and Logic Petri Nets model

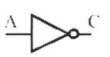
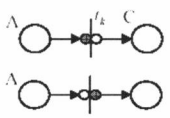
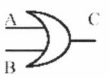
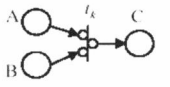
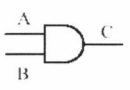
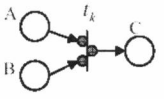
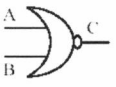
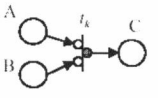
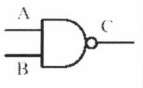
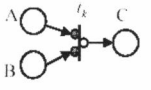
TYPE	True Table	Logic gate	Logic Petri Nets
NOT gate	A | C 0 | 1 1 | 0		
OR gate	A B | C 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1		
AND gate	A B | C 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1		
NOR gate	A B | C 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0		
NAND gate	A B | C 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0		

TABLE II
Boolean algorithm respect to LPN

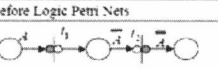
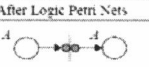
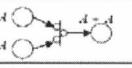
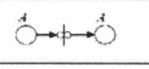
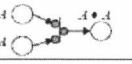
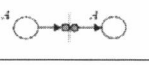
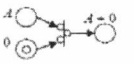
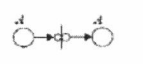
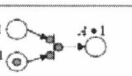
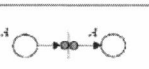
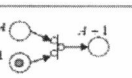
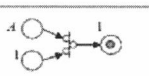
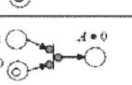
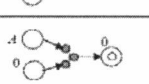
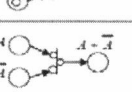
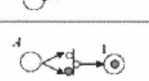
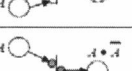
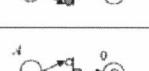
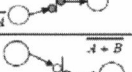
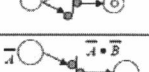
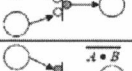
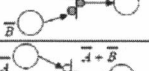
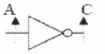
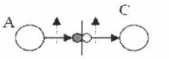
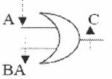
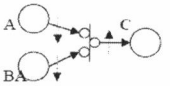
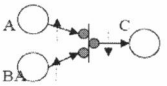
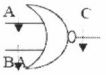
NO	Boolean algorithm	Before Logic Petri Nets	After Logic Petri Nets
(1)	$\overline{(\overline{A})} = A$		
(2)	A+A=A		
(3)	A•A=A		
(4)	A+0=A		
(5)	A•1=A		
(6)	A+1=1		
(7)	A•0=0		
(8)	$A+\overline{A}=1$		
(9)	$A\bullet\overline{A}=0$		
(10)	$\overline{(A+B)} = \overline{A}\bullet\overline{B}$		
(11)	$\overline{(A\bullet B)} = \overline{A}+\overline{B}$		

TABLE III
Fault Collapsing respond to LPN

Type of gate	Logic gate	Logic Petri Nets
Not gate		
OR gate		
AND gate		
NOR gate		
NAND gate		

Note: Fault Collapsing contains Equivalent and Dominance.

Table IV

The transitional state for forward and backward of places.

Place p_i	$b(p_i)$	$f(p_i)$	$D(p_i)$	Sign of stuck-at-fault
p_1	ϕ	$i(t_1)=\bullet$	Stuck-at-1	↑
p_2	ϕ	$i(t_1)=\bullet$	Stuck-at-1	↑
p_3	ϕ	$i(t_2)=\bullet$	Stuck-at-1	↑
p_4	ϕ	$i(t_2)=\bullet$	Stuck-at-1	↑
p_5	$o(t_1)=\bullet$	$i(t_3)=\circ$	Stuck-at-0	↓
p_6	$o(t_2)=\bullet$	$i(t_3)=\circ$	Stuck-at-0	↓
p_7	$o(t_3)=\circ$	ϕ	Stuck-at-1	↑

Table V

Immediate Reachability Set, Reachability Set, Immediate Backward Incidence Set and Backward Incidence Set for each place p_i.

Place	IRS(p_i)	RS(p_i)	IBIS(p_i)	BIS(p_i)
p_1	$\{p_5\}$	$\{p_5, p_7\}$	ϕ	ϕ
p_2	$\{p_5\}$	$\{p_5, p_7\}$	ϕ	ϕ
p_3	$\{p_6\}$	$\{p_6, p_7\}$	ϕ	ϕ
p_4	$\{p_6\}$	$\{p_6, p_7\}$	ϕ	ϕ
p_5	$\{p_7\}$	$\{p_7\}$	$\{p_1, p_2\}$	$\{p_2, p_2\}$
p_6	$\{p_7\}$	$\{p_7\}$	$\{p_3, p_4\}$	$\{p_3, p_4\}$
p_7	ϕ	ϕ	$\{p_5, p_6\}$	$\{\{p_5, p_6\}, \{p_3, p_4\}, \{p_2, p_2\}\}$

Table VI

Set of Adjacent Places Ap_{jk} for each place p_j.

Place p_j	Transition t_i	$i(t_i)$	$o(t_i)$	Place p_k	Ap_{jk}
p_1	t_1	1	1	p_5	p_2
p_2	t_1	1	1	p_5	p_1
p_3	t_2	1	1	p_6	p_4
p_4	t_2	1	1	p_6	p_3
p_5	t_3	0	0	p_7	p_6
p_6	t_3	0	0	p_7	p_5