A New Heuristic Test Sequence Generation Algorithm for Sequential Circuits

Mitsukuni Tsuboya,[*] Goro Amamiya,[*] Toshihiro Arima[*] and Jiro Okuda[*]

1. INTRODUCTION

With the advent of LSI, the problem of testing chips and packages with logic elements is becoming increasingly difficult. To overcome the difficulty, there are two approachs; one is to design easily-testable circuits, the other is to develop powerful algorithms for the test data generation of logic circuits. In this paper we are concerned with the test data generation. Up to this time many researchers have worked in this field, and published many papers, the paper about the D-algorithm which has been proposed by J. P. Roth is especially well known. In this paper a new algorithm, named MOM1 algorithm, is introduced. The algorithm is used to compute tests to detect failures in sequential logic circuits. It is a heuristic and iterative procedure which is based upon new logical operations derived from Boolean algebra. This paper describes the basic theory and the procedure, followed by a discussion of experiences with a program of the procedure is given.

2. FAILURES AND TESTS

In most cases a logic circuit that is out of order has a "stuck-line" in a solid failure; stuck-at-1 (s-a-1) or stuck-at-0 (s-a-0). We assume that a logic circuit M0 has no failures and a logic circuit M1 has a solid failure in M0. The problem is to find an input pattern T, such that the output pattern M0 (T) for M0 differs from the output pattern M1 (T) for M1, i.e. M0 (T) $\neq$ M1 (T). Such an input pattern T is called a "test" for M1. We can describe them in the language of Boolean equations.

X: a vector of input variables.

M0 (X): a Boolean representation for the output of M0.

M1 (X): a Boolean representation for the output of M1.

This paper first appeared in Japanese in Joho—Shori (Journal of the Information Processing Society of Japan), Vol. 16, No. 2 (1975), pp. 108~114.

[*] Engineering Department, Electronic Switching Div., Nippon Electric Co., Ltd.

To find a test T for M1 is equivalent to solving the Boolean equation

MO (X) ⊕ M1 (X) = 1 where ⊕ is the exclusive-or operator.

Unfortunately it is impossible to solve Boolean equations with many variables within reasonable computer time.

3. GRAPH REPRESENTATION OF THE LOGIC CIRCUIT

In general, a combinational logic circuit is documented in Boolean equation form, using Boolean operator, namely, the logic sum operator, +, the logic product operator, ., and the logic complement operator, –. A graph is then obtained from the equation by translating logic sum and product into parallel and serial connections, respectively. For example, the Boolean equation

$y = \overline{\overline{(\overline{x1} + x2).(x1 + x3)}} + x2.x3$ is represented to Fig. 1 or Fig. 2. In fact, Fig. 2 is directly derived from Fig. 1, as y-g5-$\overline{g4}$-$\overline{x2}$, we can obtain the edge e5, as shown in Fig. 2. Now, every consistent chain in the graph is a term in disjunctive canonical form. Therefore such a chain derives a solution of the equation y = 1. For instance, a chain e1e4e5 corresponds to a term $\overline{x1}x3\overline{x2}$ = 1, and it derives a solution x1 = 0, x3 = 1, x2 = 0. We can get such a chain by method as shown Fig. 3, like (a)-(b)-(c)-(d) or (a)-(b)-(e)-(f)-(g), in later case a chain e2e3e6 derives another solution x2 = 1, x1 = 1, x3 = 0.

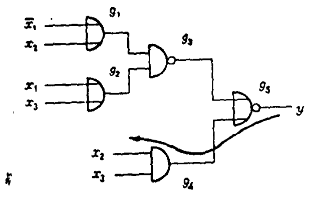

Fig.1 The circuit model

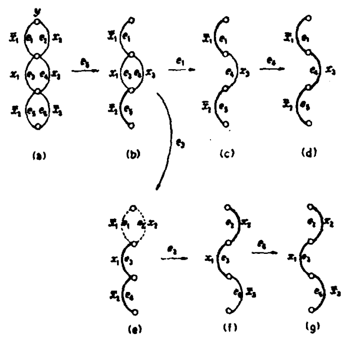

Fig.3 Consistent chains

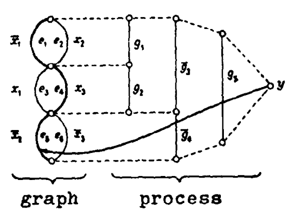

Fig.2 The graph model

4. THE NEW LOGICAL OPERATIONS

In this section, the new logical operations derived from Boolean algebra is briefly discussed, and the method for finding a solution of the equation without a graph is shown. There used to be three values $\{0, 1, u\}$ in logical operations: 0 false, 1 true, u don't care. In our logical operations, six values $\{0, 1, 2, 3, 4, 5\}$ are assumed:

Assumption 1

(1) 0 and 3 indicate false, 1 and 4 indicate true, 2 and 5 indicate don't care.

(2) 2 can be changed to 3 or 4.

(3) 3 and 4 can be changed to 1 and 0, respectively.

To operate these values a definition is introduced.

Definition 1

$A + B = (a1 + b1\ a2 + b2\ a3 + b3\ a4 + b4\ a5 + b5\ a6 + b6)$

$A . B = (a1.b1\ a2.b2\ a3.b3\ a4.b4\ a5.b5\ a6.b6)$

$\overline{A} = (\overline{a4}\ \overline{a5}\ \overline{a6}\ \overline{a1}\ \overline{a2}\ \overline{a3})$

Where A = (a1 a2 a3 a4 a5 a6), B = (b1 b2 b3 b4 b5 b6) indicate Boolean vectors and ai, bi = 0 or 1. Here special vectors are assigned to the six values.

0 = (000000), 1 = (111111), 2 = (111000), 3 = (100000), 4 = (111011), 5 = (010101).

But seven more values must be introduced so that the operations are closed in these values.

6 = (010001), 7 = (110101), 8 = (010000), 9 = (111101), 10 = (110000), 11 = (111001), 12 = (110001).

For example 3 + 5 = (100000) + (010101) = (110101) = 7. For convenience the set $S = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$ is divided into subsets as below:

Definition 2

$S1 = \{1, 4\}$, $S2 = \{2, 9, 11\}$, $S3 = \{3, 7, 10, 12\}$, $S4 = \{0, 5, 6, 8\}$,

$\overline{S1} = \{0, 3\}$, $\overline{S2} = \{2, 8, 10\}$, $\overline{S3} = \{4, 6, 11, 12\}$, $\overline{S4} = \{1, 5, 7, 9\}$.

We can apply operations to these subsets: S1 + S2 = S1, S1.S2 = S2 and so on. Here Assumption 1 can be rewritten as Assumption 2.

Assumption 2

(1) S1 indicates true, S0 indicates false.

(2) S2 and $\overline{S2}$ can be changed 4 and 3, respectively.

(3) S3 and $\overline{S3}$ can be changed 1 and 0, respectively.

Now we show an equivalent method to the method using a graph as mentioned above. Using the circuit as shown in Fig. 1, we take first expressions such as x1 = x2 = x3 = 2, and g1 = g2 = g3 = g4 = g5 = y = 2. Applying Assumption 2 (2), we select the path y-g5-$\overline{g4}$-$\overline{x2}$ which corresponds to selecting of the edge e5 in Fig. 3 (a). And we change $\overline{x2}$ to 4 (true), i.e. x2 = 3. Then g1 = g2 = g3 = g5 = y = 2 and g4 = 3. Similarly applying Assumption 2 (2) we can get x1 = 3, x3 = 4 which correspond to the edges e1, e4, respectively in Fig. 3. Here a solution x1 = false, x2 = false, x3 = true is derivered from Assumption 2 (1). Incidentally we assume x1 = 4, x2 = 3, x3 = 2 and g1 = 3, g2 = 4, g3 = 4, g4 = 3, g5 = 3 as shown in Fig. 4. Applying Assumption 2 (3) because of y = 3∈S3, we select the path y-g5-$\overline{g3}$-g1-x2 which corresponds to selecting of the edge e2 in Fig. 3 (e). We change x2 to 1 (true). In this way another solution x1 = true, x2 = true, x3 = false is derived.

5. MOM1 ALGORITHM

This section discusses MOM1 algorithm which finds a test for a failure of logic circuit. As mentioned above, a circuit M1 has a failure in M0 which has no failures. We assume that a representation a/b indicates a pair of values: a and b are the values of a logic element in M0 and M1, respectively. D and $\overline{D}$ indicate as below:

D = a/b if a = true and b = false, i.e. D = S1/$\overline{S1}$.

$\overline{D}$ = a/b if a = false and b = true, i.e. $\overline{D}$ = $\overline{S1}$/S1.

If the value of a circuit-output is D or $\overline{D}$, it is equivalent to M0 + M1 = 1. Now we show MOM1 algorithm using an example as shown in Fig. 5: the circuit has a s-a-1 failure of element g1 in M1.

(STEP 1) We take first expressions such as x1 = x2 = 2 (don't care), x3 = 5 (don't care) because the value of y is unknown. And g1 has a failure, then g1 = 2/1, g2 = 2/2, g3 = 9/5, g4 = 2/9.

(STEP 2) Since g1 = 2/1∈$\overline{S2}$/S1, we should change g1 to $\overline{D}$. Noticing g1 in M0 we back-trace such as g1-x2 in M0 and change x2 to 4. Then g1 = 3/1, g2 = 4/2, g3 = 1/5, g4 = 3/9.

(STEP 3) Since $g2 = 4/2 \in S1/\overline{S2}$, we should change g2 to D. Noticing g2 in M1 we back-trace such as g2-x1 in M1 and change x1 to 4. Then $g1 = 3/1$, $g2 = 4/3$, $g3 = 1/5$, $g4 = 3/1$. Here $y = 3/1 = \overline{D}$ and a test x1 = true, x2 = true is found.

6. IMPLEMENTATION

The algorithm, called MOM1 algorithm, was implemented by PL/1 using a computer NEAC 2200/500. The system has MOM1 program which generates tests, and also has exact simulation program which guarantees the quality of the tests. Table 1 shows the computed results for some sample circuits. From our experience MOM1 algorithm is effectively applicable to sequential circuits with under 500 logic elements.

7. COCLUDING REMARKS

Comparing with well known D-algorithm, the algorithm is a heuristic and constructive method, while MOM1 algorithm is a heuristic and iterative method. So both have advantages and disadvantages. The most advantageous feature of MOM1 algorithm is that it computes with a difference between don't care values 2 and 5: value 2 indicates don't care for circuit-input, value 5 indicates don't care for unknown of flip-flop or loop.

8. REFERENCE

(1) J. P. Roth: A Heuristic Algorithm for the Testing of Asynchronous Circuits, Computer (IEEE), vol. 20, No. 5, pp. 639 - 647 (1971).

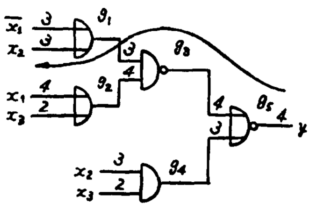

Fig.4 Backtracing

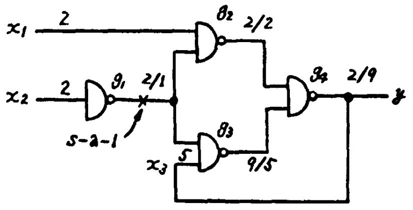

Fig.5 The circuit with a failure

Sample	No.of logic elements	No.of flip-flops	NO.of total failures	No.of total tests	CPU time (min)	Detection ratio (%)	limited time for a failuer (sec)
Sample 1	66	0	194	35	1*	100	2
Sample 2	106	2	278	31	4*	97	10
Sample 3	273	32	892	109	44*	88	10
Sample 4	1464	228	5476	165	130	63**	60

* CPU time including simulator's
** Detection ratio for only 222 failuers

Table 1 Computed results by NEAC 2200/500