Improvement of Time Constants Method for Hazard Analysis

Asist. ing. Mihai Timiş,
Ş.l. ing. Nicu Galupa
Technical University "Gh.Asachi" Iaşi

ABSTRACT. In this paper we consider the problem of hazard in digital equential structures. First we define a method (based on the time onstants method) to thoroughly analyze the possibility of hazardous functioning for a combinatorial structure. We shall also prove that this new approach is of higher quality than the classical method by pointing out hazards that the time constants method does not find. Secondly we apply this method for the analysis of hazard in the combinatorial structures that implement the input and output group of functions in an automaton. So we find out where and when the input combinatorial structure generates hazard and more important we find out the moment of time when the outputs of this structure are stabilized to the correct value. That means that for each state of the automaton we shall find out the minimum amount of time after which we can force the evolution of the automaton towards the next correct state. That information proves to have tremendous importance in improving the overall speed of the automaton.
Keywords: hazard in combinatorial structures, hazard analysis, Mealy automaton, variable duty factor clock.

1 Foreword

The first step, as mentioned above, is the enhancement of the time constants method for the analysis of hazard in combinatorial structures.

The second step is applying this method for the analysis of hazard presence in the combinatorial structure that implements the input group of functions for an automaton. We remind that we define the automaton in a Mealy meaning being the quintuple (X,Y,Z,F,G) where X is the finite array of the input variables, Y is the finite array of the state variables, Z is the

finite array of the output variables, F:XxY→Y is the transfer group functions (towards state) and G:XxY→Z is the output group of functions.

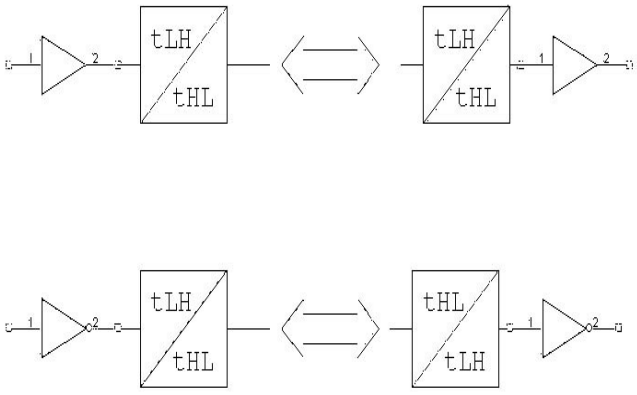
We consider the existing implementation of the transfer group of functions F:XxY→Y. We shall consider, at one time, each function f_i from the group F, and we shall extract the combinatorial logic circuit that implements it. The output vector is being already separated in two distinct vectors – the input variables vector X and the present state vector Y. We shall consider the input vector X as being already synchronized with the main clock of the automaton. We shall trace the output, using the method presented as step one, only for those transitions that are compatible with the states transition graph of the automaton. Next we shall design a circuit that generates a variable clock signal that will force the evolution of the automaton only when the evolution is safe towards a correct next state.

Hazard analysis for combinatorial structures

Basically we shall follow the scheme suggested by [Bei74], known as the time constants method. However we shall use two new notions:

Gate level: we say that gate **k** is characterized by level **n** if the maximum number of gates that must be crossed by at least one primary input variable to reach it is **n**.

Real gates & Ideal gates: we shall maintain the equivalence between a real gate and an ideal gate plus a delay (propagation time), as defined by the time constants method. However because the propagation times for a given gate are not identical for both transitions possible ("0"↑"1" and "1"↓"0") we shall consider the propagation time as a fraction that includes the specific time for each of the two transitions possible (t_{LH}/t_{HL}). Please note the influence that an inverting characteristic has on the propagation times when we translate them towards the input.



2 Method's general description

We'll describe the method by presenting the actions to be followed in a step-by-step manner. Let be a switching function y=f(a,b,c,d) and let be a given combinatorial structure that implements it.

- **STEP 1** Reorganize the combinatorial structure so that on each vertical we have gates characterized by the same gate level. We assign an identificator to each gate (a number for example).
- **STEP 2** Multiply the gates that drive more than one input and define the distinct paths that each primary input crosses towards the output
- **STEP 3** Assign to each path defined in step 2 a secondary input variable which is derived from the primary input variable crossing that path.
- **STEP 4** Rewrite the function equation with respect to the secondary input variables defined in step 3. We'll name secondary function this form of expressing the function.
- **STEP 5** We describe the circuit by aid of a graph. Each gate is a knot in the graph and a vector describes each connection between gates. ATTENTION – we maintain the level organization defined in step 1. Also we mark the gates with an inverting characteristic (by an " * " for example).
- **STEP 6** We calculate the delay times for each path. The following steps are to be repeated for each set of secondary input variables that are derived from one primary input.

  6.1. We generate a rectangular network. We associate the secondary input variables to the network's rows and the gate levels to the network's columns. At each row-column crossing point we write the identificator of the gate that is being crossed by the secondary variable associated to that row, on the level associated with the column. Also we mark the inverting characteristic if it exists.

  6.2. We write the propagation time matrix using a similar matrix with the one in step 6.1. We shall use the fractionary propagation times notation.

  6.3. We write the inverting levels matrix using a similar matrix with the one in step 6.1. The element at the crossing point of row **i** with column **j** has value **k** if the secondary input variable

associated to row **i** crosses up to level **j**, including the gate on level **j**, **k** inverting gates.

6.4. We write the propagated towards input delay matrix using the matrixes written in steps 6.2. and 6.3. So, if the inverting level is odd for the element at the crossing point of row **i** with column **j** we'll transfer from the matrix written in step 6.2. the fraction with the nominator and denominator interchanged (an odd number of interchanges). If the inverting level is even we'll transfer the same fraction unchanged (an even number of interchanges).

6.5. We add in the matrix written in step 6.4., for each row, the nominators and denominators of the fractions, calculating this way the delay times for the respective paths.

- **STEP 7** We study if there is static or dynamic hazard by altering one by one the input variables

7.1. We consider all the possible combinations for the input variables that are not studied at a moment. We calculate the reduced functions by entering in the secondary function equation the values for the variables not studied. So we'll generate a reduced functions table.

7.2. We mark and analyze the reduced functions that are expressed by aid of secondary input variables that are characterized by different propagation times (so have the capability of generating hazard).

7.3. We study hazard for each reduced function marked in step 7.2.

7.3.1. We arrange the propagation times for the secondary input variables in a rising order. So we can determine the combinations that appear when crossing from the initial stable form towards the final stable form.

7.3.2 We map out on a table the combinations that appear and the specific moments of time when that happens. So we can easily see the diagram of the output for a specific transition.

We shall consider an example that will prove the efficiency of the method. Let's consider the circuit presented in fig.1. By crossing steps 1, 2 and 3 we'll reach the circuit presented in fig.2.

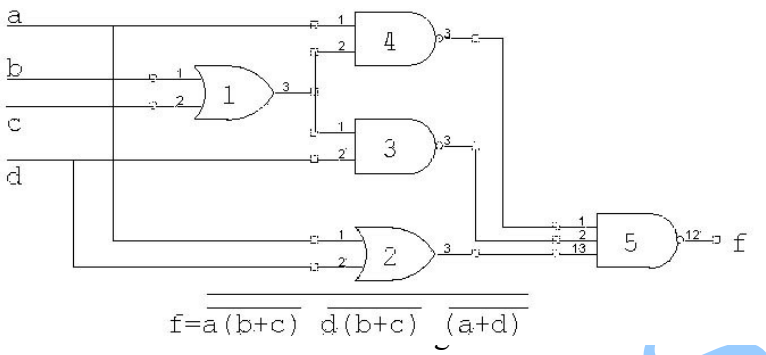


Fig.1

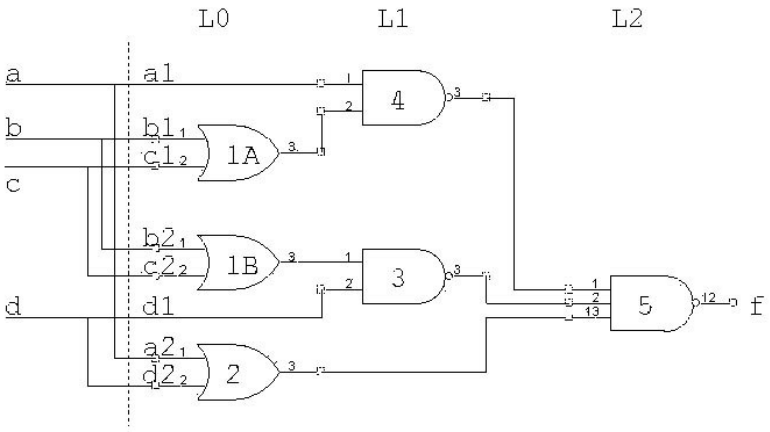


Fig.2

Step 4 will produce the secondary function equation:

$$fs=\overline{a_1(b_1+c_1)}*\overline{d_1(b_2+c_2)}*(a_2+d_2)$$

By crossing step 5 we reach the graph presented in fig.3. Next, for each primary variable we execute step 6, as follows:

a_1	.	3^*	5^*
a_2	2	.	5^*

0/0	45/15	45/15
22/18	0/0	45/15

0	1	2
0	0	1

0/0	15/45	45/15
22/18	0/0	15/45

So we find that t_{a1}=60/60 ns, t_{a2}=37/63 ns

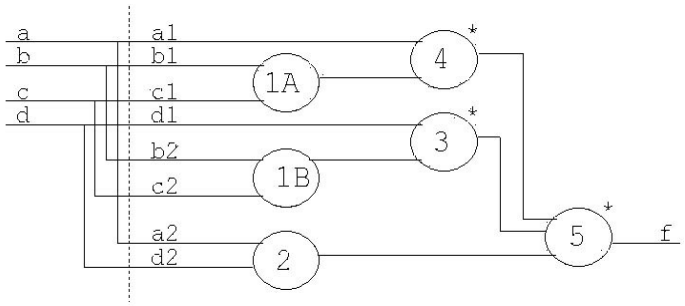


Fig.3

Following the same method we find:

$t_{b1}=t_{b2}=t_{c1}=t_{c2}$=82/78 ns

t_{d1}=60/60 ns

t_{d2}=37/63 ns

After determining the propagation times for each variable we should execute step 7 for each primary variable and for all possible combinations of them. In this paper we'll present the analysis for variable **a** and for a synchronous change of variable **a** and **b**.

Variable a

dcb	f_r
000	a_2
001	$\overline{a_1}a_2$
010	$\overline{a_1}a_2$
011	$\overline{a_1}a_2$
100	0
101	1
110	1
111	1

$f_r=\overline{a_1}a_2$

We shall analyze

1. a "0" → "1" with $t_{a1LH} > t_{a2LH}$

a_1 A_2	0 0	1 0	1 1
f_r	1	0	1
t	0 ns	37 ns	60 ns

We found that we have hazard and the settling time for the output is 60 ns

2. a "1" → "0" with $t_{a2LH} > t_{a1LH}$

a_1 A_2	1 1	1 0	0 0
f_r	1	0	1
t	0 ns	60 ns	63 ns

Hazard is present, $t_{settling}$=63 ns.

Synchronous switch of variables a and b

dc	f_r
00	$\overline{a_1 b_1}\, a_2$
01	$\overline{a_1} a_2$
10	$\overline{a_1 b_1}\, a_2$
11	1

We shall analyze $f_r = \overline{\overline{a_1 b_1}\, a_2}$

1. ba "00"→"11" with $t_{b1LH} > t_{a1LH} > t_{a2LH}$

$a_2a_1b_1$	000	100	110	111
f_r	1	0	0	1
t	0 ns	37 ns	60 ns	82 ns

Hazard present, $t_{settling}$=82 ns

2. ba "01"→"10" with $t_{b1LH} > t_{a2HL} > t_{a1HL}$

$A_2a_1b_1$	110	100	000	001
f_r	1	0	1	1
T	0 ns	60 ns	63 ns	82 ns

Hazard present, $t_{settling}$=63 ns

3. ba "10"→"01" with $t_{b1HL} > t_{a1LH} > t_{a2LH}$

$a_2a_1b_1$	001	101	111	110
f_r	1	0	1	0
T	0 ns	37 ns	60 ns	78 ns

Hazard present, $t_{settling}$=78 ns

4. ba "11"→"00" with $t_{b1HL} > t_{a2HL} > t_{a1HL}$

$A_2a_1b_1$	111	101	001	000
f_r	1	0	1	1
T	0 ns	60 ns	63 ns	78 ns

Hazard present, $t_{settling}$=63 ns

We hope that now it is clear how we determine whether exists or not hazard and how we can draw the wave form of the output. Also we can easily see which is the minimum amount of time requested by the output to stabilize to the correct value.

As we have promised before we'll consider the same analysis using the known time constants method. According to the method we find the circuit presented in fig.4.

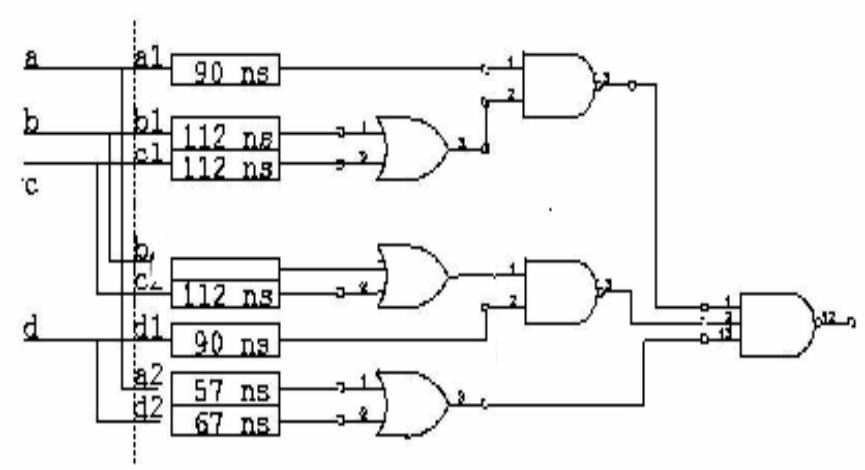


Fig.4

Let's study what happens for the transition 0011→0010, transition that proved to be hazard generating.

$f_r = \overline{a_1} a_2$

and $t_{a1} > t_{a2}$

a_1 a_2	1 1	1 0	0 0
f_r	1	1	1
t	0 ns	60 ns	63 ns

So with this method we conclude that this transition of the input vector is hazard free. However we have proved that this specific transition is hazard generating so we conclude that the improved method suggested is of higher quality.

3 Conclusions

1. The method for hazard analysis proves to more efficient
2. The way the method for hazard analysis is defined allows to map the output of a circuit by means of computer
3. Using a variable clock instead of a fixed one enhances the working speed of the automaton. For the circuit studied using a programmable clock enhanced the performance of the automaton by 36.23% in the case of x1x0=00
4. Last but not least we do not have to eliminate hazard but we can simply avoid it. That means that we do not have to use more gates in order to introduce redundant terms in the process of minimization.

References

[Bei74] **J. Beister** – *A unified approach to combinational hazards* – IEEE_TC, vol C-23, no 6, 1974

[Gal91] **N. Galupa** – *Analysis method for hazard determination in combinatorial structures,* 8th international conference on control systems and computer science, Bucharest, 1991

[VB86] **Al.Valachi, M.Birsan** – *Tehnici numerice si automate (Numerical techniques and automata),* Ed. Junimea – Iasi, 1986

[***] *** – *Analog Devices - Data Book,* 1990