Avoiding Hazards for Speed-Independent Logic Design

Igor Lemberski, *Member, IAENG*

Abstract - **In the speed-independent logic, the hazards caused by input inverters are identified. The known methods of the elimination of such hazards are based on avoiding input inverters. In contrast, we propose the method that produces hazard-free circuits with combined (both right and inverse) input signals. It is shown that such circuits are competitive (in terms of complexity) to ones implemented using either right or inverse input signals. The method is intended for incorporation into the current synthesis methodology.**

Index terms- **asynchronous logic, hazards, speed-independent logic, state graph**

I. Introduction

The methodology of speed –independent synthesis is very attractive since it allows designing hazard-free circuits under unbounded gate delays. The current approach [1] is based on the circuit description in the form of signal transition graph (STG) and its transformation into state graph (SG - fig.1a) with following properties: boundedness, consistency, output persistency, complete state encoding. For implementation, RS-based architecture [1] is considered (fig.1b). It consists of two parts: a regular part (homogenous NAND- or NOR-based) flip-flops and control one to implement set-reset functions. For hazard-free implementation, the method of introducing supplementary internal signals to meet so called monotonous cover conditions is proposed. The synthesis example of the frequency divider by 2 was processed in [2] using mentioned methodology and resulting circuit is reproduced in fig.1c. Although the methodology [1] is oriented on speed-independent circuit design it was shown [2] that resulting circuit may be not speed - independent (contain hazards) caused by the presence of the input inverter. It was concluded [2] that the circuit is speed independent only if zero delay of the input inverter is supposed. However, such an assumption is not realistic. In [4, 6], the hazard – free synthesis methodologies were proposed based on avoiding input inverters. In this paper, we will show that for the hazard - free implementation avoiding input inverters is not necessarily. The hazard-free circuits designed using our method may have combined (both right and inverse) input signals. The goal of the paper is twofold: 1) in RS-based implementation, identify the type of the hazards caused by input inverters; 2) propose the method of the elimination of this type of hazards and incorporate it into the current synthesis methodology. For implementation, it is supposed comprehensive CMOS gate library [4]. It is used for Indus trial applications. In this library, each CMOS gate is a complex logic of AND, OR operations (in any order) followed by the inverter. Our method is compact and is based on processing SG simple configurations (called *d-trios*).

II. Essential Hazards and D-trio

It was shown in [2] that hazards in the circuit (fig.1c) occur during transitions (000) -> (110) and (011) -> (101). Based on this analysis one can conclude that the reason of hazards during above mentioned transitions is the presence of SG configurations: (000) -> (110) -> (011), (011)->(101)->(000) (called *d-trio* [3]). A d-trio is the sequence of two transitions between three different states where the first transition occurs when input signal switches: 0 ->1, but the second one- when the same input signal switches: 1 ->0. The hazard caused by d-trio is called an essential hazard [3]. Consider d-trio (000) -> (110) -> (011) and timing diagram (fig.1d). Switching of input signal x: 0 ->1 implies signals yn :1 ->0 (yn means negative of y) and y :0 -> 1. As a result, state (110) is reached. If by this moment signal xn is not yet switched to 0 due to input inverter delay ($\tau(xn) > \tau(yn) + \tau(y)$, where τ – CMOS delay) then signals zn:1 ->0 and z :0 ->1. As a result, d-trio second transition occurs and (wrong) state (111) is reached. The hazard within d-trio (011) -> (101)->(000) can be identified in the similar way.

III. Essential Hazards Elimination

We start from SG with above mentioned properties. Our goal is to eliminate essential hazards in SG and therefore create Essential Hazard Free SG (EHF-SG). For this purpose, either right or inverse input (not both) should imply d-trio transitions. Namely: 1) if signal x: 0 -> 1 implies d-trio first transition, then signal x: 1 -> 0 (not xn: 0 -> 1) should imply d-trio second transition; or 2) if signal xn: 0 -> 1 implies d-trio second transition, then signal xn: 1 -> 0 (not x: 0 -> 1) should imply d-trio first transition. It is achieved by introducing into SG supplementary internal signal c with proper switching discipline.

Manuscript received March 22, 2007.

I. Lemberski is with Baltic International Academy, Lomonosova 4, Riga, LV-1003, LATVIA (phone: +37122 33 29 61; fax: +371 7 241 272; e-mail: Igor.Lemberski@bsa.edu.lv).

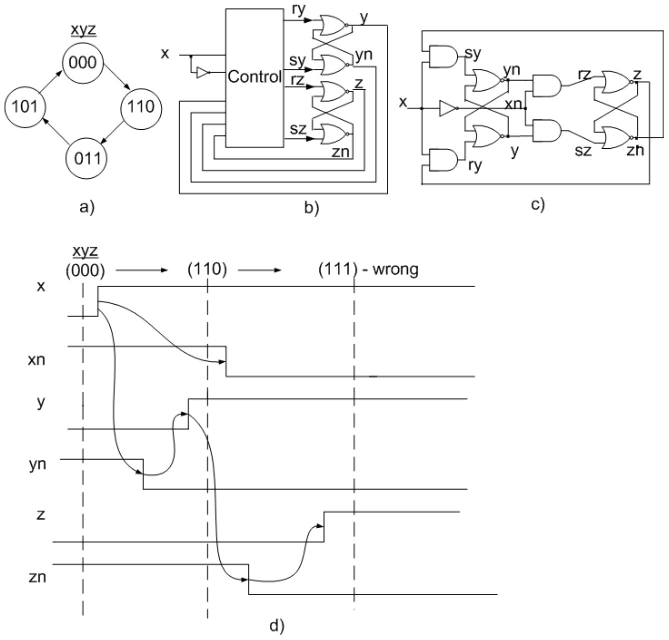


Fig.1. Frequency divider by 2 implementation: a) SG, b) RS-based architecture, c) circuit, d) timing diagram

A. Implementation with Right Inputs

Denote each SG state as couple (x, IS) where x is input signal and IS – Internal State (determined by state signals). Consider d-trio (0,A) -> (1,B) -> (0,C). Let c+ (c-) be signal c that switches: 0 -> 1 (1 -> 0). Our approach is as follows: signal c- should be introduced into d-trio second transition (1,B) -> (0,C) and signal c+ – into any transition that occurs before transition (1,B) -> (0,C). Actually, signal c+ can be introduced into 1) transition (0,A)-> (1,B) (we call it "inside d-trio"); or 2) "outside d-trio" - into the transition between arbitrary states (x,D), (xn,E), provided that there is (possibly, indirect) transition between states (xn,E) and (0,A): (a,D) -> (an,E) ->…-> (0,A) ->…, a∈{0,1}, an – inverse to a. Depending on signal c+ place, circuits with different properties and complexity are generated. Consider case when signal c+ is placed "inside d-trio". The timing diagram and corresponding fragment of EHF SG is given (fig.2a), where each state is described by triple (x,IS,c), c - internal signal c value. One can see that signals c+, c- imply supplementary states (bold) in d-trio transitions. Suppose, the circuit is in state (0,A,0). Actually, signal x+ implies two signals: cn- and c+, (what results in the transition to state (1,A,1)). Signal c+ implies activation of set – reset functions to ensure transition (1,A,1) -> (1,B,1). To eliminate essential hazards one should order signals cn-, c+. Namely, signal cn- should occur first to avoid transition to state (0,C,0) once state (1,B,1) is reached. To order cn- and c+ signal c should be implemented using a gate with a negative output (fig.2a). As required, signal x- (not signal xn+) implies signals cn+ , c- (transition to state (0, B, 0)). Signal cn+ activates set-reset functions to ensure transition (0, B, 0) -> (0, C, 0), Now, consider case when signal c+ is placed "outside d-trio". The timing diagram and corresponding fragment of EHF SG is shown (fig.2b). Signal x+ implies immediate activation of set- reset functions and transition (0,A,1) ->(1,B,1) occurs (fig.2b). The transition to state(0,C,0) can't occur since signal cn=0. Signal x- implies signals cn+, c- (transition to supplementary state (0,B,0)). In turn, signal cn+ implies activation of set-reset functions (transition to state (0,C, 0)). Denote the set of possible supplementary signals as C_r: $c \in C_r$. The formal procedure of the essential hazard elimination is as follows:

1) STG – based description is transformed into SG,
2) d- trios are extracted from SG,
3) for each d-trio, supplementary signal c, $c \in C_r$, is introduced,
4) for each d-trio, the SG local transformation that includes introducing supplementary states is done.

As a result of this procedure, EHF-SG is created.

Note, that the procedure always guarantees the solution. Indeed, once d-trios are extracted, each d-trio processes separately and number of d-trios is finite.

To obtain hazard-free implementation, known methodology [1] is applied to EHF-SG, namely, monotonous covers are generated, after that the minimized functions are produced and implemented using CMOS gate library.

Example. The frequency divider by 2 STG-based description and its transformation into SG were shown in [2]. In SG (fig.1a), we have two d-trios and two supplementary signals c1, c2 are required. Signal c1- should be placed between states 2,3, signal c2- between states 4,1.

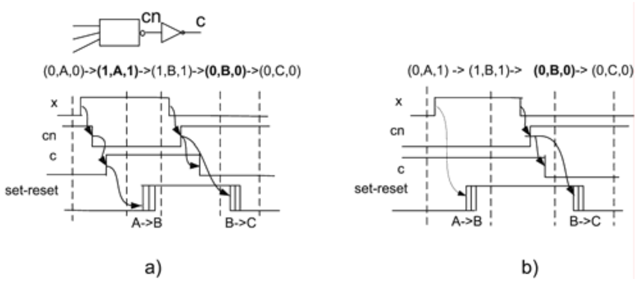
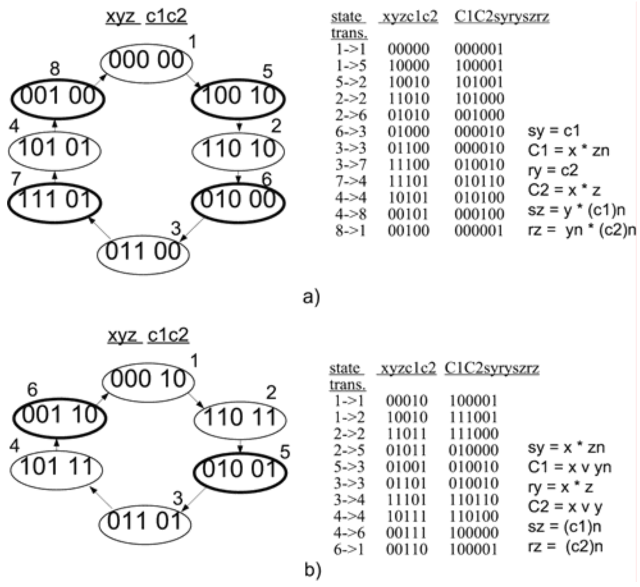

cn
c
(0,A,0)->(1,A,1)->(1,B,1)->(0,B,0)->(0,C,0)
x
cn
c
set-reset
A->B
B->C
a)
(0,A,1) -> (1,B,1)-> (0,B,0)-> (0,C,0)
x
cn
c
set-reset
A->B
B->C
b)

Fig.2. Implementation with right inputs. Timing diagrams if internal signal c+ is placed a) "inside d- trio"; b) "outside d-trio"

The set $C1_r$ is as follows: $C1_r$ = {0100, 0101, 1100, 1101}, the set $C2_r$: $C2_r$ = {0001, 0011, 0101, 0111}. Place signals c1+, c2+ "inside d-trios" first. It is between states 1,2 and 3,4 accordingly. Potentially, the supplementary variables c1: (0101) or (0100) and c2: (0101) or (0001) can be chosen. We checked both cases and found out that the competitive circuit is produced if variables: c1 = (1000), c2 = (0001) are chosen. Supplementary states are shown in bold (fig.3a). Then, place signals c1+, c2+ "outside d-trios". Potentially, the supplementary variables c1: (1100) or (1101) and c2: (0011) or (0111) can be chosen. We choose variables: c1 = (1101), c2 = (0111). The truth tables and resulting functions are presented (fig.3b). Fortunately, no internal signals are required to ensure monotonous covers. One can see that resulting circuits (fig.4) have no inverse input signals and 8

CMOS gates are required to implement the circuit (fig.4a) and 6 CMOS gates are required to implement the circuit (fig.4b). However, the circuit (fig.4a) benefits from lower input signal capacitance load.



state trans.	xyzc1c2	C1C2syryszrz	
1->1	00000	000001	
1->5	10000	100001	
5->2	10010	101001	
2->2	11010	101000	
2->6	01010	001000	
6->3	01000	000010	sy = c1
3->3	01100	000010	C1 = x * zn
3->7	11100	010010	ry = c2
7->4	11101	010110	C2 = x * z
4->4	10101	010100	sz = y * (c1)n
4->8	00101	000100	rz = yn * (c2)n
8->1	00100	000001	

state trans.	xyzc1c2	C1C2syryszrz	
1->1	00010	100001	
1->2	10010	111001	
2->2	11011	111000	
2->5	01011	010000	sy = x * zn
5->3	01001	010010	C1 = x v yn
3->3	01101	010010	ry = x * z
3->4	11101	110110	C2 = x v y
4->4	10111	110100	sz = (c1)n
4->6	00111	100000	rz = (c2)n
6->1	00110	100001	

Fig.3. Implementation with right inputs. Frequency divider by 2 EHF SGs, truth tables and equations if: a) signals c1+ , c2+ are placed "inside d-trios" ; b) signals c1+ ,c2+ are placed "outside d - trios"

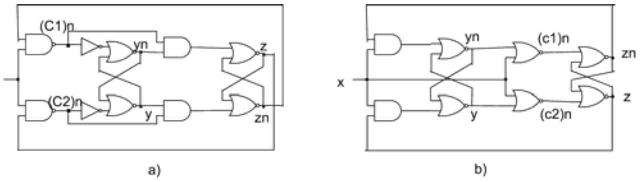


Fig.4. Resulting circuits with right inputs

B. Implementation with Inverse Inputs

Let us introduce signal c- into d-trio first transition (0,A) -> (1,B) and signal c+ – into any transition that occurs before transition (0,A) -> (1,B). Actually, signal c+ can be introduced into 1) the transition between arbitrary states (a,D), (an,E), provided that there is (possibly, indirect) transition between states (an,E) and (0,A): (a,D) -> (an,E) ->…-> (0,A) ->…, a∈{0,1}, an – inverse to a ("outside d-trio") or 2) transition (1,B) -> (0,C) ("inside d-trio"). Depending on signal c+ place, circuits with different properties and complexity are generated. Consider case when signal c+ is placed "outside d-trio" in a transition-predecessor: …-> (0,A) (fig.5a). In a transition-predecessor, signal x- implies signal xn+ and signal xn+ implies signals cn- , c+ . Signal c+ activates set-reset functions to ensure transition to state A. In turn, signal cn+ implies activation of set-reset functions to ensure transition (0,A,1) -> (1,B,0). Therefore, signals cn- and c+ should be ordered to avoid transition to state B before signal cn- occurs. Namely, cn- should occur first. To order, signal c should be implemented using a gate with a negative output (see paragraph III.A). Now consider case when signal c+ is placed "inside d-trio"(fig.5b). One can see that signals c+, c- imply a supplementary state (bold) within d-trio. Suppose, the circuit is in state (0,A,1). Actually, signal xn- (not signal x+) implies signal cn+ (what results in the transition to state (1,A,0)) and signal cn+ implies activation of set – reset functions to ensure transition (1,A,0) -> (1,B,0). Once state B is reached, the transition to state C can't occur since xn=0. Signal xn+ implies immediate activation of set-reset functions and transition (1,B,0) ->(0,C,1) occurs (fig.5b). Denote the set of possible supplementary signals as C_i: $c \in C_i$. The formal procedure of the essential hazard elimination is the same as in paragraph III.A, except step 3: for each d-trio, supplementary signal c, $c \in C_i$, is introduced.

Example. For the frequency divider by 2 (fig.1), signal c1- should be placed between states 1,2, signal c2- between states 3,4. It results in following sets: $C1_i$ = {1000, 1001, 1010, 1011}, $C2_i$= {0010,0110, 1010, 1110}. Place signals c1+, c2+ "outside d-trios" first (between states 4,1 and 2,3 accordingly). Potentially, the supplementary variables c1: (1010) or (1000) and c2: (0010) or (1010) can be chosen. We choose variables: c1 = (1000), c2 = (0010). Supplementary states are shown in bold (fig.6a). Again, no internal signals are required to ensure monotonous covers. One can see that resulting circuit (fig.8a) has inverse input signals and 8 CMOS gates are required for implementation. Then, place signals c1+, c2+ "inside d-trios". Variable c1 may be either (1000) or (1010) and c2: (0010) or (1010). We choose: c1 = (1000) and c2 = (0010). The truth tables and resulting functions are presented (fig.6b).

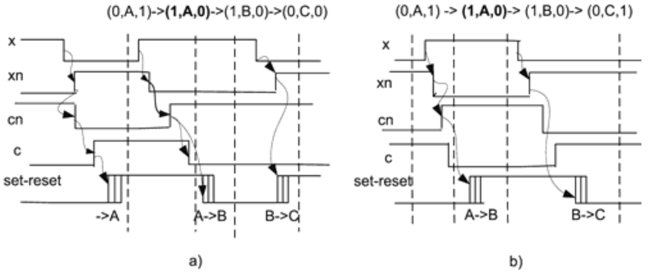
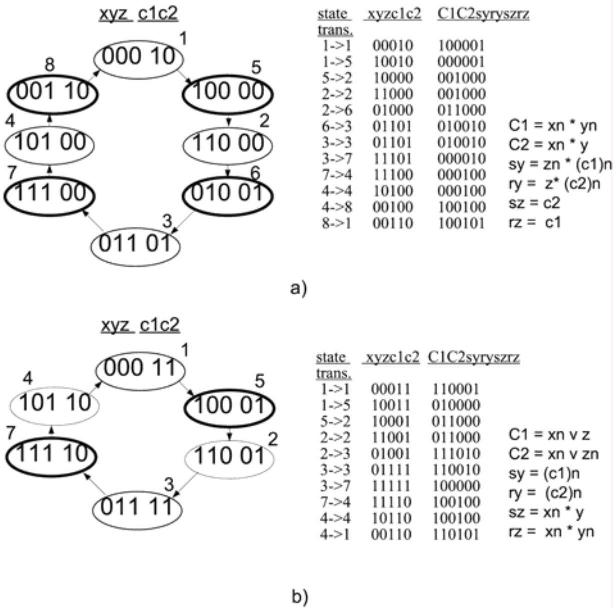


Fig.5. Implementation with inverse inputs. Timing diagrams if internal signal c+ is placed a) "outside d- trio"; b) "inside d-trio"

The method can be used when only inverse inputs (not right ones) are available. Note, that the alternative way is to synthesize the circuit applying the method from paragraph III.A and then invert the input. Once the present method is applied, no input inverter is required.



a)

state trans.	xyzc1c2	C1C2syryszrz
1->1	00010	100001
1->5	10010	000001
5->2	10000	001000
2->2	11000	001000
2->6	01000	011000
6->3	01101	010010
3->3	01101	010010
3->7	11101	000010
7->4	11100	000100
4->4	10100	000100
4->8	00100	100100
8->1	00110	100101

C1 = xn * yn
C2 = xn * y
sy = zn * (c1)n
ry = z* (c2)n
sz = c2
rz = c1

b)

state trans.	xyzc1c2	C1C2syryszrz
1->1	00011	110001
1->5	10011	010000
5->2	10001	011000
2->2	11001	011000
2->3	01001	111010
3->3	01111	110010
3->7	11111	100000
7->4	11110	100100
4->4	10110	100100
4->1	00110	110101

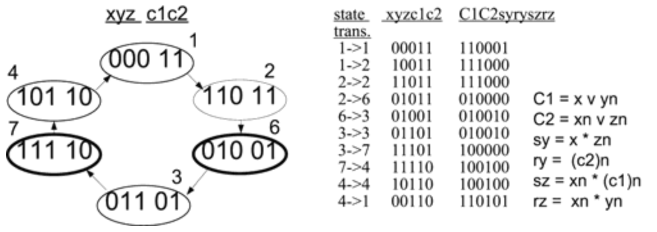
C1 = xn v z
C2 = xn v zn
sy = (c1)n
ry = (c2)n
sz = xn * y
rz = xn * yn

Fig.6. Implementation with inverse inputs. Frequency divider by 2 EHF SGs, truth tables and equations if: a) signals c1+, c2+ are placed "outside d-trios"; b) signals c1+ ,c2+ are placed "inside d - trios"

C. *Implementation with Combined Input Signals*

For hazard-free synthesis, one can combine methods described in paragraphs III.A, III.B. Namely, in the formal procedure (paragraph III.A) step 3 is modified as follows: for each d-trio, supplementary signal c, c$\in C_r \cup C_i$, is introduced. As a result, implementation containing combined input signals (both right and inverse) can be produced. Once implementation with right inputs is supposed and internal signal c+ is place "outside d-trio" the resulting circuit is less complex (in terms of CMOS gates and transistors). Similarly, the implementation with inverse inputs where signal c+ is placed "inside d-trio" leads to resulting circuit of less complexity. However, both circuits have the same drawback: input signal fanout is rather high (4). However, we can combine these two approaches. As a result, we expect to have the circuit with less input signal fanout (load will be distributed between the right and inverse inputs) while preserving low complexity.

Example. Consider again the frequency divider by 2 (fig.1). Potentially, c1 $\in C1_r \cup C1_i$ ={0100, 0101, 1100, 1101, 1000, 1001, 1010, 1011} c2 $\in C2_r \cup C2_i$ ={0001, 0011, 0101, 0111, 0010,0110,1010,1110}. For d-trio 1->2->3, let us choose c1=(1101): implementation with right inputs where signal c+ is placed "outside d-trio", for d-trio 3->4->1, we choose c2=(1110) : implementation with inverse inputs, where signal c+ is placed "inside d-trio". EHF-SG, truth table and equations are presented in fig.7. Again, no internal signals are required to ensure monotonous covers. One can see that resulting circuit (fig.8b) has both right and inverse inputs, requires 6 CMOS gates. However, the circuit (fig.8b) benefits from input signal reduced capacitance (right input signal fanout is 2 and inverse input signal fanout is 3 in contrast to 4 for the circuit (fig.4b)). Therefore, the implementation based on the combined inputs results in reducing capacitance load since the input load is distributed between the right and inverse signals.



state trans.	xyzc1c2	C1C2syryszrz
1->1	00011	110001
1->2	10011	111000
2->2	11011	111000
2->6	01011	010000
6->3	01001	010010
3->3	01101	010010
3->7	11101	100000
7->4	11110	100100
4->4	10110	100100
4->1	00110	110101

C1 = x v yn
C2 = xn v zn
sy = x * zn
ry = (c2)n
sz = xn * (c1)n
rz = xn * yn

Fig.7. Implementation with combined inputs. Frequency divider by 2 EHF SGs, truth tables and equations if: a) the methodology when inverse input implies transitions is applied and supplementary signal c1+ is placed "inside d-trio"; b) the methodology when right input implies transitions is applied and supplementary signal c2+ is placed "outside d - trio"

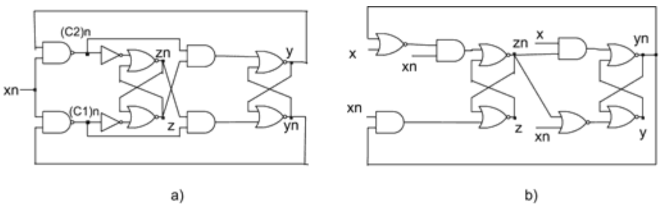


Fig.8. Resulting circuits with: a) inverse inputs; b) combined inputs

IV. Complexity

In table 1, we indicated the complexity of circuits ("Examples") with the right (fig.4a, 4b) and combined inputs (fig.8b) produced using our method. Note, that the circuits with inverse inputs produced have the same complexity as the circuits with the right ones (compare, for example the circuits on fig. 4a and fig.8a). Therefore, the circuits with right inputs only are included into table 1. Following parameters were considered: number of CMOS gates required for implementation ("Gates"), number of CMOS transistors ("Transistors") and input signal fanout ("Fanout").

Table 1. Complexity comparison

Examples	Gates	Transistors	Fanout
Fig.4a	8	32	2
Fig.4b	6	28	4
Fig.8b	6	30	3

No one circuit is an absolute winner. The circuit with the smallest input signal fanout (fig. 4.a) requires the largest number of CMOS gates and transistors. The least number of CMOS gates and transistors are required for the circuit with the largest input signal fanout (fig.4b). Note, that the circuit with combined inputs occupies the intermediate position in terms of input signal fanout and number of transistors and requires the least number of CMOS gates.

V. Conclusion

For RS– based architecture, we identified the type of hazards (essential hazards) caused by input inverters and proposed the method of avoiding essential hazards. It is based on introducing supplementary signals with proper switching discipline and SG local transformations. As a result, EHF-SG is created. In contrast to known approaches, we showed that for hazard- free implementation avoiding the input inverters is not necessarily. Namely, the circuit may contain combined (both right and inverse) input signals. We compared the complexity of circuits with the right (inverse) and combined inputs. The circuit with combined inputs occupies the intermediate position in terms of input signal fanout and number of CMOS transistors required for implementation. However, it requires the least number of CMOS gates to be implemented. Therefore, the circuits with combined inputs are competitive to the circuits with right (inverse) inputs. Our method is intended for incorporation into the current synthesis methodology [1].

References

[1] J. Cortadella, M. Kishinevsky, A. Kondratyev, etc, Logic Synthesis of Asynchronous Controllers and Interfaces, Springer – Verlag Berlin Heideberg, 2002, 274p

[2] C. Piguet, Supplementary condition for STG-designed speed-independent circuits, Electronics Letter, vol. 4, April 2, 1998, pp. 620-622

[3] S. Unger, Asynchronous Sequential Switching Circuits, John Wiley and Sons, 1969, 400p.

[4] C. Piguet, Logic Synthesis of Race-Free Asynchronous CMOS Circuits.- IEEE Journal of Solid- State Circuits, vol. 26, No.3, March, 1991, pp. 371-380

[5] C.Piguet, J.Zahnd, Design of Speed-Independent CMOS Cells from Signal Transition Graphs, PATMOS'98, October, 1998, Copenhagen, Denmark

[6] N. Starodoubtsev, S. Bystrov, Behavior and Synthesis of Two-Input-Gate Asynchronous Circuits, Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'05), March 13-16, 2005, New York City, USA