Semi-modular Latch Chains for Asynchronous Circuit Design

N. Starodoubtsev*, A. Bystrov, and A. Yakovlev

Department of Computing Science, University of Newcastle upon Tyne, NE1 7RU, U.K.

Abstract. A structural discipline for constructing speed-independent (hazard-free) circuits based on canonical chains of set-dominant and reset-dominant latches is proposed. The method can be applied to decompose complex asymmetric C-gate generated by logic synthesis from Signal Transition Graphs, and to map them into a restricted gate array ASIC library, such as IBM SA-12E that consists of logic gates with maximum four inputs and includes AO12, AOI12, OA12 and OAI12. The method is illustrated by new implementations of practically useful asynchronous circuits: a toggle element and an edge-triggered latch controller.

1 Introduction

Asynchronous circuits offer promising advantages for circuit design in deep-submicron technology, amongst which the most attractive are low power, EMC, modularity and operational robustness. As systems-on-chip become a reality, design of asynchronous control circuits that can tolerate variations in timing parameters of components is particularly important. Examples of such circuits are interface controllers [1]. A class of asynchronous circuits that are insensitive to gate delay variations is Muller's speed-independent (SI) circuits [2]. An extensive research has been in methods and algorithms for synthesis of SI circuits in the last decade [3]. A software tool, called Petrify [4], can synthesise a SI circuit from its Signal Transition Graph (STG) specification [5] if the latter satisfies the basic implementability conditions [3]. The result of synthesis is a circuit in which each non-input signal is a *generalised or asymmetric C-gate* [6] (see Section 2).

The property of *acknowledgement* is characteristic to SI circuits compared to their less conservative counterparts, such as Burst-Mode circuits [7] or Timed circuits [8,9]. According to this property, every transition of each gate output is acknowledged by another signal, which allows the circuit to operate correctly for unbounded gate delays. Guaranteeing this property, however, is a difficult task, particularly if the circuit realisation is restricted by a given gate library. Petrify can perform logic decomposition using a gate and latch library, in which components can be restricted to a given number of input literals. In order to preserve

* On leave from: Institute for Analytical Instrumentation, Russian Academy of Science, St. Petersburg, Russia; work in Newcastle supported by EPSRC GR/M94359.

D. Soudris, P. Pirsch, and E. Barke (Eds.): PATMOS 2000, LNCS 1918, pp. 168–177, 2000.

SI property after logic decomposition, Petrify seeks for the newly emerging gate outputs to be acknowledged by other signals, or in the case of complements of signals assumes the delay of input inverters ("bubbles") to be equal to zero. This is a limitation that is not present in our canonical decomposition of generalised C-gates.

This paper addresses the problem of the gate level realisation of SI circuits for CMOS ASIC libraries in which cells may have a limited number of inputs, e.g. three. A regular method for constructing a class of speed-independent circuits composed of 3-input gates AO12, AOI12, OA12 and OAI12 is presented. These gates implement monotonic Boolean functions $d = a+be$, $d = \overline{a + be}$, $d = a(b+e)$ and $d = \overline{a(d + e)}$, respectively. Most CMOS gate libraries include these elements. For example, the IBM SA-12E gate array library [10] offers such gates with high speed and low power consumption, and compared to other 3-input (simple) gates, such as AND3 and OR3, their functional capabilities are greater – one can construct latches out of them. E.g., a simple state-holding element $d = a + bd$ (*set-dominant latch*) can be built out of just one AO12 if output d connected to its input e. We present examples of the application of our construction method, by showing two new implementations of practically useful circuits, one is a toggle element and the other is a pipeline stage (latch) controller. Both circuits are built as chains of the above mentioned positive and negative gates. They are totally speed-independent, they do not have zero delay inverters, and thus compare favourably to the existing solutions.

Negative gate circuits attracted attention about three decades ago since it was noticed that basic CMOS gates have inherent output inverters and thus implement decreasing, or negative monotonic, Boolean function [11,12,13]. Later, interest to negative asynchronous circuits arose when it became clear that they consume less power than their non-negative counterparts [14,15,16].

The rest of the paper is organised as follows. Basic latches are introduced in Section 2. Positive latch chains for the implementation of asymmetric C-gates are described in Section 3. Negative chains and circuit reduction methods are presented in Section 4. Section 5 illustrates applications, a toggle element and edge-triggered latch control circuits. Analysis of behavioural correctness of our circuits is discussed in Section 6. Section 7 contains the conclusion.

2 Basic Latches and Notations

A latch built of a single AO12 element, known as *a set-dominant latch*, is shown in Fig. 1(a). Its behaviour is described by the STG depicted in Fig. 1(b), where "+" denotes the rising signal edge and "-" the falling edge. Signals a and b are inputs, signal d is an output. The solid arcs depict casualty relations within the circuit, whereas the dotted arcs describe the environment behaviour.

Following the STG in Fig. 1(b), transition $a+$ causes transition $d+$ while transition $d-$ is caused by the firing of $a-$ and $b-$. The signalling discipline between the latch and the environment assumes that transitions at a and b inputs may only occur after signal d becomes stable. This latch can be considered as a

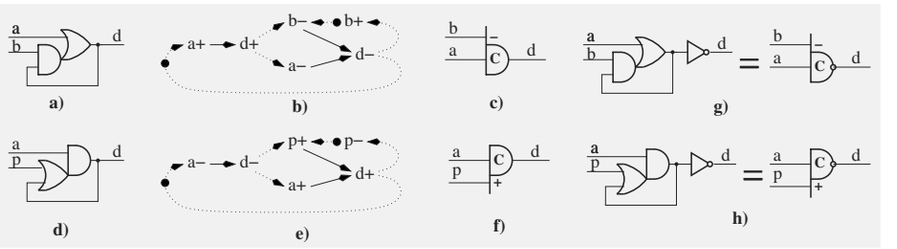

Fig. 1. Basic latches: b-gate(a), its behaviour(b) and notation(c); p-gate(d), its behaviour(e) and notation(f); g, h - negative $\overline{b}$- and $\overline{p}$-gates and their notations

simple case of an asymmetric C-gate [6] as shown in Fig. 1(c). Input a in this drawing, being connected to the main body of the symbol, controls both $d+$ and $d-$, while input b, being connected to the extension marked "-", controls only $d-$.

A dual circuit (*reset-dominant latch*) can be built of OA12 gate. Its schematic, STG and symbolic notation are shown in Fig. 1(d,e,f). In this case, input a controls both edges of d while p controls only the rising edge of d. The shapes of symbols in Fig. 1(c,f) look similar to Latin characters "b" and "p", which we will use to denote the asymmetric C-gates as *b-gate* and *p-gate* respectively. The latches with inverted outputs, shown in Fig. 1(g,h), will be denoted as $\overline{b}$-gate and $\overline{p}$-gate respectively. In the following sections the latches of b, p, $\overline{b}$ and $\overline{p}$ types are used as building blocks to construct more complex components of SI circuits.

3 Generalised Asymmetric C-gates

3.1 Homogeneous Positive Latch Chains: Generalised Latches

A homogeneous chain comprising b-gates only is shown in Fig. 2(a). Such a circuit, denoted as b^n, where n is the number of stages, implements a generalised C-gate with single input a controlling both edges of signal d and n signals $b_1, \ldots b_n$ controlling $d-$ only (set-dominant latch). A dual circuit, denoted as p^m, where m is the number of stages, is shown in Fig. 2(b). Note, that b^n and p^m chains are transitive, so any pair of gates within a chain can be swapped places without affecting the external specification of the chain.

Similar chains of more complex latches can be constructed. An example of a three input p-gate (a, p_1, p_2) is shown in Fig. 2(c). Its transistor-level implementation could be simple, being just one transistor pair larger than a b-gate. Such an element is not present in most gate array libraries and, therefore, will not be considered. However, it can be implemented as a p^2-chain.

3.2 Heterogeneous Positive Latch Chains: C-gates

Any asymmetric C-gate can be constructed as a composition of two generalised b and p-latches (see Fig. 3(a)), which results in a heterogeneous latch chain. Two

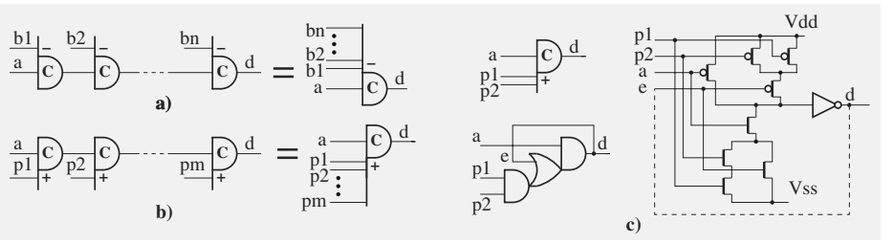

Fig. 2. Homogeneous b^n-chain (a); p^m-chain (b); single gate realisation of p^2-chain (c)

examples of a 3-input asymmetric C-gate, based on two simple b and p-latches, is shown in Fig. 3(b).

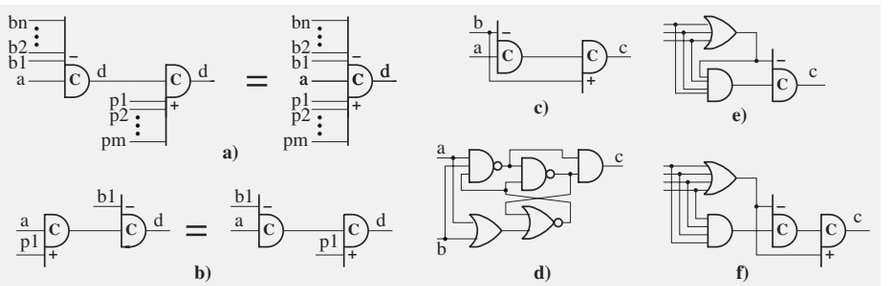

Fig. 3. Heterogenous latch chains: $b^n p^m$-chain for generalised asymmetric C-gate implementation (a); pb and bp-chains (b); 2-input symmetrical bp-chain C-element (c); Mayevsky C-element; 3-input (e) and 4-input (f) C-element

Both chains in Fig. 3(b) are equivalent in their functionality, though having different signal delays from input $b1$ (or $p1$) to output d. The chain function is preserved under any transposition of b and p-gates. Hence, any heterogeneous chain consisting of n b-gates and m p-gates is functionally equivalent to the $b^n p^m$-chain. Both bp and pb-chains can be used to implement a two-input symmetric (Muller) C-element [18] as shown in Fig. 3(c). This realisation favorably compares to the known Mayevsky [19] C-element shown in Fig. 3(d). The following list contains pairs of parameters for comparison of the pb-chain against Mayevsky C-element in the CMOS AMS-3.11 0.6μ realisation: 2/5 gates, 16/25 transistors, 1.51/1.85(ns) cycle time, 11.9/21.5 pJ energy per cycle and 699/1748 μ^2 area.

3.3 Chain Length Reduction

Serial connection of elements in a $b^n p^m$-chain may cause a significant delay. In many cases the chain length can be reduced by using a simpler generalised C-gate (with less inputs) and an expansion circuit comprising AND/OR gates.

A traditional expansion solution [17], shown in Fig. 3(f), uses an OR gate to detect the all-zeroes input state (the condition of switching to zero) and an AND gate to detect the all-ones input state (the condition of switching to one). The outputs of these gates are connected to the inputs of the symmetrical C-gate. It is easy to check that all signals in this circuit are acknowledged under the assumption of wires having no delay (Muller's SI model). This method is applicable only to the symmetrical C-gate inputs, i.e. to those which control both events of output switching to 1 and to 0.

A new compact solution to the expansion problem is shown in Fig. 3(e). It uses a single b-latch instead of the symmetric C-gate. This improvement is achieved at the expense of an additional circuit (connecting the output of the OR-gate to an additional input of the AND gate) providing the acknowledgement of 1 at the output of the OR-gate. A disadvantage of this solution is the number of possible inputs reduced by 1 in comparison with Fig. 3(f).

4 Negative Latch Chains

4.1 General Properties

Note that connecting p or b-gate to the symmetrical input a of a $b^n p^m$-chain implementing a generalised C-gate is equivalent to adding an input to "+" or "-" extension of the C-gate, as shown in Fig. 4(a) for a b-gate. The rule of adding inputs to the extensions for negative latch chains is more complicated.

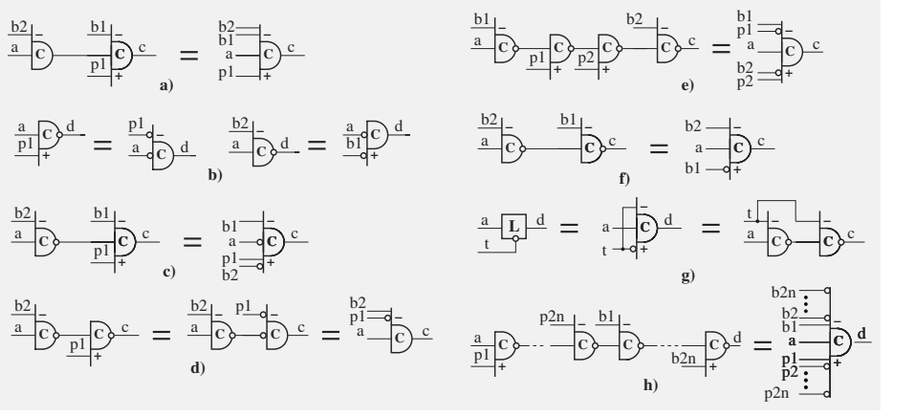

Fig. 4. Negative latch chains transformations: connecting b-gate to C-gate input (a); duality (De Morgan's) rule for $\overline{b}$- and $\overline{p}$-gates (b); connecting $\overline{b}$-gate to C-gate (c), conversion of $\overline{b}\overline{p}$-chain to an asymmetric C-gate, $\overline{b}\overline{p}^2\overline{b}$-chain ($e$), $\overline{b}^2$-chain (f); transparent latch implementation (g), example of complex negative chain (h)

The functionality of a $\overline{p}$-gate is equivalent to that of a b-gate with all inputs inverted. Output d of a $\overline{p}$-gate (see Fig. 4(b)) gets 1 as soon as $a = 0$ and it gets 0 as soon as $a = p1 = 1$. The same for a b-gate with inverted inputs: if $\overline{a} = 1$, then output d gets 1 and if $\overline{a} = \overline{p1} = 0$, then output d gets 0. This corresponds to DeMorgan's laws.

Connecting $\overline{p}$ or $\overline{b}$-gate to the input of a generalised C-gate is equivalent to adding an inverted input to "-" or "+" extensions of the C-gate, respectively, and inverting its input a. The example in Fig. 4(c) illustrates this for the $\overline{b}$-gate.

Let us consider chains consisting only of $\overline{p}$- and $\overline{b}$-gates, starting with the simplest case of a heterogeneous negative $\overline{b}\overline{p}$-chain shown in Fig. 4(d). Using the above transformations one can see that such a chain results in an asymmetric C-gate with two inputs connected to "-" extension. The symmetric chain $\overline{b}\overline{p}^2\overline{b}$ leads to a more symmetric generalised C-gate depicted in Fig. 4(e). In general, each $\overline{p}$ and $\overline{b}$-gate contributes to either "+" or "-" extensions, respectively, without input inverters if the signal path leading from the input of this gate to the chain output includes odd number of inverters. If such a path includes an even number of inverters (bubbles), then $\overline{p}$-gate ($\overline{b}$-gate) contributes an inverted input to the "-" ("+") extension.

The immediate consequence of this claim is that any transposition of odd gates in a negative chain preserves its functionality. The same takes place for even gates. Hence, each negative chain is functionally equivalent to $\overline{p}^n(\overline{p}\overline{b})^m(\overline{b}\overline{p})^k\overline{b}^l$, where $m, k = 0, 1, 2, \ldots$; $n, l = 0, 2, 4, \ldots$

A special case of such a chain, namely $\overline{b}^2$-chain (see Fig. 4(f)), is useful for transparent latch realisations. The transparent latch with "enable" input t, data input a and output d, which is transparent when $t = 0$ and opaque for $t = 1$, can be implemented as a generalised C-gate shown in Fig. 4(g) with t connected to both extensions.

Finally, as a more complex example, a heterogeneous $(\overline{p}\overline{b})^n(\overline{b}\overline{p})^n$-chain for a generalised C-gate with n inputs in both "+" and "-" extensions, with half of them being inverted, is given in Fig. 4(h).

4.2 Reduction of Negative Chains

Negative chains comprising $\overline{b}$ and $\overline{p}$-gates look more complex than those comprising b and p-gates. However, the structure of $\overline{b}$ and $\overline{p}$-gates, if implemented by CMOS circuits, includes two inverters: the first is the inherent inverter of b or p-gate and the second is output inverters depicted in Fig. 1(g,f). These inverters can be removed without changing the chain function in semi-modular applications. Such a circuit is simpler and faster than its positive counterpart.

A new realisation of inverting transparent latch can be derived from Fig. 4(g). The circuit in Fig. 5(a) is obtained by refining the $\overline{b}$ and b-gates. Note, that signal transitions propagate from left to right in that negative gate chain in such a way that, in any cycle which starts after signal d assumes a new value, signal g accepts the value of signal f with some delay. Therefore, the feedback from f to the input of the left-most $\overline{b}$-gate can be replaced by the feedback from output g. Further,

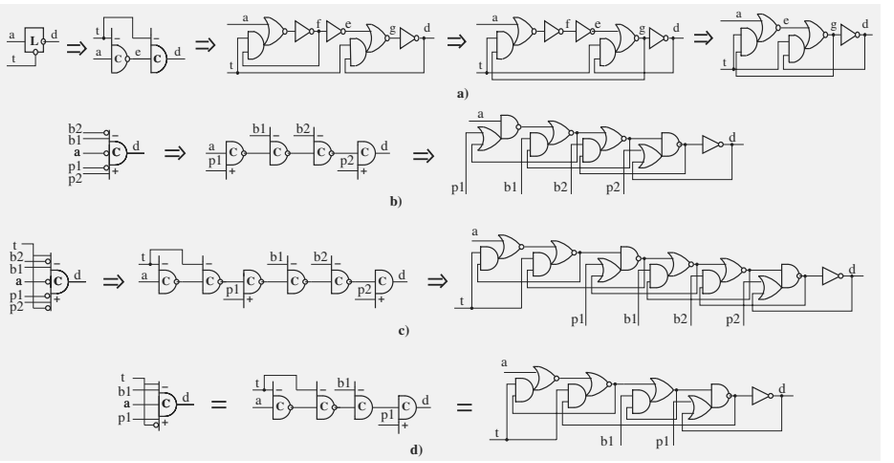

Fig. 5. Reduction of negative chains for: transparent latch application (a), C-gate with a few inputs inverted (b), the same with built-in transparent latch (c), 2-input C-element with built-in transparent latch

the transitions at f and e do not affect any other signal in the circuit. Hence, the inverters at f and e can be safely removed without affecting the circuit function.

This approach can be applied to any negative chain, as shown in examples in Fig. 5(b,c,d). A three-input C-element with two inverted inputs (see Fig. 5(b)) can be realised as $\overline{p}\overline{b}^2\overline{p}$-chain. A similar C-gate with an additional built-in transparent latch can be obtained from a reduced $\overline{b}^2\overline{p}\overline{b}^2p$-chain as shown in Fig. 5(c). A mixed negative-positive $\overline{b}^2bp$-chain, consisting of negative and positive latches, realises a two-input symmetric C-element with a built-in transparent latch (see Fig. 5(d)). This solution can be seen as a Muller C-element enhanced with the enabling/blocking input t.

5 Examples Based on Reduced Negative Chains

We have, so far, considered only applications mapped easily on latch chains. We will now consider other applications, which are compositions of two chains. These implementations present new solutions to the known practical circuit designs. They illustrate the power of the reduction approach applied to a chain that is a backbone of the application.

Toggle. A toggle is one of the key elements in constructing self-timed micropipeline controllers [20], with two-phase signalling discipline. The STG of a toggle element is shown in Fig. 6(a). It responds to each even (odd) transition on input x with a transition on output $y1$ ($y2$).

A known solution for toggle circuit [21] based on two transparent latches with different polarity of the control signal x is shown in Fig. 6(b). We propose

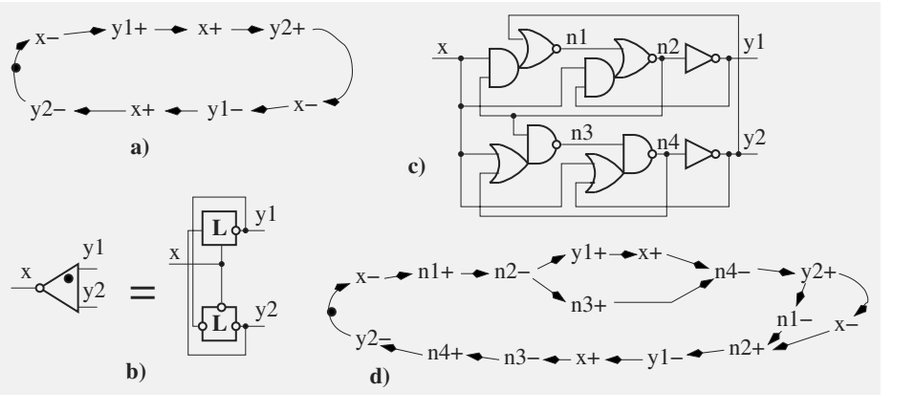

Fig. 6. A toggle circuit: STG (a), transparent latch-based realisation (b), reduced negative circuit(c), refined STG (d)

the implementation shown in Fig. 6(c), which is based on the transparent latch shown in Fig. 5(a). Its STG is given in Fig. 6(d).

Being implemented in IBM SA-12E gate array library (0.25μ, 2.5V), this circuit has the following delays from input x to outputs y_1 and y_2: $d(y_1+) = 0.29ns$, $d(y_2+) = 0.19ns$, $d(y_1-) = 0.30ns$, $d(y_2-) = 0.24ns$.

Edge-triggered latch control circuit. The edge-triggered latch control circuit described in [6] has the STG shown in Fig. 7(a). We refine the implementation based upon asymmetric C-gates [6] using our basic negative chains. The circuit in Fig. 7(b) is obtained by further reduction and simplification.

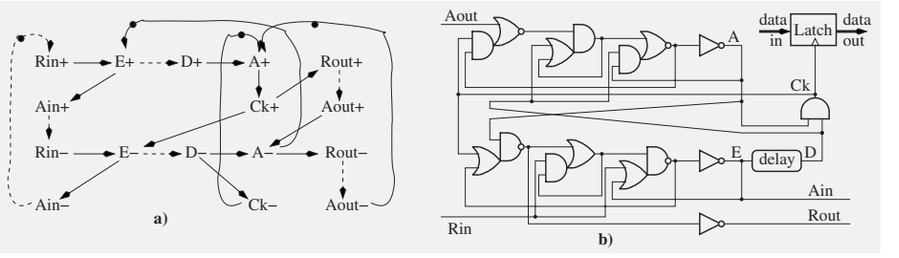

Fig. 7. Edge-triggered latch control: STG (a), circuit (b)

6 Behavioural Correctness

Semi-modularity [2] of two above examples was checked by Versify tool. All other our proposed solutions are also semi-modular, i.e. no hazards are possible

under the correct environment behaviour defined in Fig. 1(b,e). The intuitive reasoning behind this claim (the proof is omitted) is that under such a discipline every transition on the output is followed by a single transition of each input, which in turn eventually acknowledged by the next output transition.
All the properties described above have been obtained under assumption of *monotonic* environment behaviour. That is if the circuit input is set to some particular value, which is the *necessary* condition of the output event, then this value must not change until the output event happens. All our circuits are robust to monotonic environment behaviour. However, there are applications where the environment, being semi-modular (hence SI), is allowed to withdraw such an input, providing that the output is not excited. This environment behaviour is *non-monotonic*.

Such non-monotonic inputs, being applied to a circuit comprising several stages may cause switching of the internal signals. Under the above condition of the output being not excited the events on internal signals are not acknowledged at the circuit output, which may result in hazards.

Latches and chains shown in Fig. 1-5 may produce hazards in a non-monotonic environment. The robustness analysis of the proposed circuits in non-monotonic environments is the subject of the future work.

7 Conclusion

A method of speed-independent asynchronous controllers design, using a limited fan-in gate library, has been developed. It is based on chains of set-dominant and reset-dominant latches. Several regular structures comprising positive and negative chains are studied and a reduction technique is used at the latch level. Our method can be applied to decompose complex asymmetric gate implementations generated by logic synthesis tools (such as Petrify) from Signal Transition Graphs, and to perform mapping into a restricted ASIC gate array library, such as IBM SA-12E (contains logic gates with maximum three-four inputs and includes AO12, AOI12, OA12 and OAI12 logic gates). No assumptions on inverter (bubble) delay are used. The method has been illustrated by the new implementations of practically useful asynchronous building blocks: a toggle element and an edge-triggered latch controller.

References

1. M. Kishinevsky, J. Cortadella, A. Kondratyev and L. Lavagno. Asynchronous interface specification, analysis and synthesis, Proc. DAC'98, pp. 2–7.
2. D. E. Muller and W. S. Bartky. A theory of asynchronous circuits. In Proceedings of an International Symposium on the Theory of Switching, pp. 204-243. Harvard University Press, April 1959.
3. A. Kondratyev, J. Cortadella, M. Kishinevsky, L. Lavagno, and A. Yakovlev. Logic Decomposition of Speed-Independent Circuits, In Proceedings of the IEEE/, Vol.87, No.2, February 1999, pp. 347-362.

4. J. Cortadella, M. Kishinevsky, A. Kondratyev, L. Lavagno and A. Yakovlev. Petrify: a tool for manipulating concurrent specifications and synthesis of asynchronous controllers, IEICE Trans. Inf. and Syst., Vol. E80-D, No.3, March 1997, pp. 315-325.
5. L.Y.Rosenblum, and A.V.Yakovlev. Signal graphs: From self-timed to timed ones, in Proc. Int. Workshop Timed Petri Nets, Torino, Italy, 1985, pp. 199–207.
6. S.B. Furber and J. Liu. Dynamic logic in four-phase micropipelines. Proc. of the Second Int. Symp. on Advanced Research in Asynchronous Circuits and Systems (ASYNC'96) March, 1996 Aizu-Wakamatsu, Japan, pp.11–16.
7. Steven M. Nowick. Automatic Synthesis of Burst-Mode Asynchronous Controllers. PhD thesis, Stanford University, Department of Computer Science, 1993.
8. Chris J. Myers. Computer-Aided Synthesis and Verification of Gate-Level Timed Circuits. PhD thesis, Dept. of Elec. Eng., Stanford University, October 1995.
9. Ken Stevens, Ran Ginosar, and Shai Rotem. Relative timing. In Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC'99), Barcelona, Spain, pages 208-218, April 1999.
10. http://www.chips.ibm.com/techlib/products/asics/databooks.html
11. T.Ibaraki, S.Muroga. "Synthesis network with a minimal number of negative gates", IEEE Trans. on Computers, Vol. C-20. No. 1, January 1971
12. G.Mago, "Monotone function in sequential circuits", IEEE Trans. on Computers, Vol. C-22. No. 10, October 1973, pp.928 - 933.
13. N.A.Starodoubtsev. Asynchronous processes and antitonic control circuits, In: Soviet Journal of Computer and System Science (USA), English translation of Izvestiya Akademii Nauk SSSR. Technicheskaya Kibernetika (USSR), 1885, vol.23, No.2, pp.112-119 (Part I. Description Language), No.6, pp.81-87 (Part II. Basic properties), 1986, Vol.24, No.2, pp.44-51 (part III. Realisation).
14. C.Piguet. Logic synthesis of race-free asynchronous sequential circuits. IEEE JSSC, vol.26, No 3, March 1991. pp. 371-380.
15. C.Piguet. Synthesis of Asynchronous CMOS Circuits with Negative Gates. Journal of Solid State Devices and Circuits, vol.5, No.2, July 1997.
16. C.Piguet, J.Zahnd. Design of Speed-Independent CMOS Cells from Signal Transition Graphs. PATMOS'98. Oct.1998, Copenhagen, pp.357-366.
17. V. Varshavsky (Ed.), Aperiodic Automata, Nauka, Moscow, 1976 (in Russian).
18. D. E. Muller. Asynchronous logic and application to information processing, Switching Theory in Space Technology, H. Aiken and W. F. Main, Eds. Stanford, CA; Stanford Univ. Press, 1963, pp. 289-297.
19. J. A. Brzozowski and K. Raahemifar. Testing C-elements is not elementary. In Asynchronous Design Methodologies, pp. 150-159. IEEE Computer Society Press, May 1995.
20. Ivan E. Sutherland. Micropipelines. In: Communications of the ACM. June 1989, vol.32, N6, pp.720-738.
21. M. Josephs. Speed-independent design of a Toggle. Handouts of ACiD-WG/EXACT Workshop on Asynchronous Controllers and Interfacing. IMEC, Leuven, Belgium, September 1992.