Functional Approach in Self-Timed Circuit Design*

Plekhanov L.P.$^{1)}$, Zakharov V.N.$^{2)}$, Stepchenkov Y.A.$^{3)}$
Department of perspective computer systems architecture
Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences (IPI FRC CSC RAS), IPI RAS, Moscow, Russian Federation
$^{1)}$ lplekhanov@inbox.ru, $^{2,3)}$(VZakharov, YStepchenkov)@ipiran.ru

Abstract

Self-timed circuits have the unique properties of a lack of competition. One of the main problems of such circuits design - the analysis on self-timing (elements switching) and the construction of large circuits. In traditional approach computational complexity is so great, that it does not allow to analyze the most important practical circuits. In the functional approach we propose hierarchical method: on the lower level logic functions of elements are analyzed, on the top - the relationships between blocks. The complexity of calculations here is close to the linear function of the size of circuit. Such approach solves one of the main problems of self-timed circuits design - analysis circuits of any size. The efficiency of the proposed methods was confirmed by the developed SW.

1. Introduction

Self-timed circuits (ST-circuits) belong to the class of asynchronous speed-independent [1] circuits which are functioning properly independently on the delay of elements. The term «speed-independent» is defined for closed circuits with one initial state. More convenient for the practical development seems the term "self-timed circuit", proposed in [2, 3], since it allows the opened representation of circuits.

These properties provide unique effects. They provide the widest range of proper functioning, defined only by the physical (not circuit design) possibilities of elements switching, that is unattainable for synchronous circuits. The correctness of the ST-circuits at low supply voltages allows one to create circuits with low power consumption. A fail-safe feature makes it possible to obtain highly reliable circuits including their self-testing and self-repairing. The development of practical ST-circuits and CAD tools in Russia was started by staffs of V.I. Varshavsky [2, 3]. They developed ST-circuit analysis subsystem TRANAL [3]. It based on the method of transition diagrams (TD) in full states, and provided analysis of small circuits with parallelism order up to 6.

To increase size of analyzed circuits, the group of V.I. Varshavsky has developed the event method of change diagrams (CD) [4] and subsystem of analysis TRASPEC, but for distributive circuits [5] only (Table 1).

Table 1. A comparison of CAD tools [5]

Circuit	Micropipeline control queue (Figure 10 from [5])		Asynchronous switch (Figure 9 from [5])	
CAD Tool	Time, sec	Memory /+ or – disk swapping	Time, sec	Memory
TRANAL	180	400 Kbytes /+	0.28	100 Kbytes
VERDECT	300	22 Mbytes /–	0.99	122 Kbytes
TRASPEC	1	100 Kbytes /–	0.38	100 Kbytes

A complexity and necessity of CD implementation for all ST-circuit classes were noted in [5, 6].

In Russia, the practical methods and tools for creating and analysis of the ST-circuits are developing by specialists in IPIRAN. Initially, the subsystems Big TRANAL (BTRAN) and ASYAN [7] were developed on CD base. This had allowed for increasing parallelism order of analyzed circuits up to 24 (Table 2).

Subsequently a CD based subsystem ASPECT was developed. It has provided a support of designing all ST-circuit classes (Table 3).

Attained complexity level can be considered as an ultimate for analysis programs that do not use a hierarchical analysis. Further decrease of time costs

*The reported study was funded by RFBR according to the research project № 13-07-12068 ofi_m).

Table 2. A comparison of TD CAD tools [7]

CAD tools	Paral-lelism	Circuit level	Memory for state, Gbyte	Number of states per hour, 10^6
TRANAL	5-6	cells	0.64	0.3
BTRAN	12 16	modules	2	1.0
ASYAN	18 24	blocks	48	200

Table 3. A comparison of CAD tools (IPI RAN)

Unit's type –capacity / parallelism	Analysis time, min$^{*)}$		
	BTRAN	ASYAN	ASPECT
Binary counter – 4 / 1	0.01	0.12	0.02
Shift register – 4 / 4	0.21	0.97	0.02
Microcore – 4 / 47	–	0.53	0.02
ALU – 64 / 293	–	–	0.14
Divider – 16 / 330 [8]	–	–	1588
Divider – 64 / 1024 [8]	–	–	27360

$^{*)}$ for single state of inputs and triggers

can be obtained by means of usage of hardware-software tools and parallel programming which facilities are rather limited. Transfer to the hierarchical analysis is the cardinal solution of this problem.

2. The problems of classical analysis

The classical analysis of schemes on the property of semimodularity (self-timing) is based on the study of circuit's states and switching of elements - events. This approach we will call further *event-driven*.

The purpose of the first method of the analysis (TD [1]) is to identify conflicts (potential competitions) in the scheme. Circuits without conflicts have been called *semimodular*. Computational problem in the TD method arises because of the exponential dependence of the number of states on the number of elements, as well as in connection with the degree of parallelism in the circuit. Let N - number of elements in the excited state (degree of parallelism). According to the rules of TD construction this state generates N following states, in each of which N–1 elements are excited. Each of these states, in turn, produces N-1 followers in each of which there are N–2 excited elements. So we see the factorial dependence of computing on the degree of parallelism. For example, if N = 10, which is quite possible, then the number of states generated exceeds 3.6 million.

The CD method allows to submit work of scheme in a more compact way, describing not complete states but only changed outputs of elements. The method is based on an analysis of the equivalent TD. Since all passable states of the scheme should be analyzed in the existing degree of parallelism, computational difficulties, though reduced, but remain large.

The common for event methods is that the scheme should be presented in a closed form, which guarantees its self-generation, as well as the fact that the analysis is carried out from only one given initial state. Both of these circumstances require artificial constructions and impede traditional design of circuits.

For practical purposes it is necessary to analyze all states of the circuit in the real operation that is to provide a complete analysis. Calculations for completeness [9] show that in the event-driven approach to guarantee the fulfillment of this condition it is possible sending to the input 2^{2I+M} sets, where I - the number of data inputs, M - the number of memory variables, and these sets must follow in a particular order. This means that it is necessary to build a very complicated circuit, increasing the number of equations for the analysis.

The above computational difficulties limit the use of event-methods: for TD - for circuits of the two or three dozen elements, for CD - some more.

Event methods analyze the scheme as a whole. In practice, large circuits are designed in part, and using these methods, you must repeatedly make and break pieces of the scheme. In this it is usually impossible to analyze final scheme completely because of its size, that does not allow to guarantee their semimodularity.

The only way to analyze circuits of any size - use a hierarchical method guaranteeing self-timing of compound circuit on the basis of the previously performed analysis of its parts. In the event-driven approach, this method has not yet been offered.

Further a functional approach [9] is described, it essentially uses particularly of ST-circuits and allowed to solve the problems mentioned above.

3. Functional approach to analysis

Features of the functional approach:

1) Analyzes the equations of elements, circuit states in 0 and 1 are not computed.

2) We consider the open loop schemes, which is more natural in practical development.

3) Taking into account the properties of ST-circuits [3]: ST-coding data, two-phase work discipline (striping working phase and intermediate one - spacer), indicating of signals and other.

In the event-driven approach, the criterion of independence on delays is semimodularity. This criterion is a mathematical abstraction, only indirectly related to the practical parameters of the circuit. But the consequences of semimodularity – lack of competitions and failure safety - have a direct practical meaning. These consequences can form the basis of the definition and construction of the ST-circuits. Therefore, for practical purposes of ST-circuits design the following definition has been proposed [9].

ST-circuit - is open loop scheme, for all real initial and passable at work states has two properties: the lack of competitions for all finite elements delays and failure safety with respect to the constant sticking to 0 and 1 of elements outputs.

When implemented on-chip, self-timing may depends on delays in the tracks. Therefore, on the functional-logical level of development the properties stated in the definition, are the necessary conditions and shall be provided with engineering solutions. The issue of delays in the tracks requires additional topological analysis and is not considered here.

Analysis on self-timing in functional approach consists in checking the both properties of circuit separately for each fragment at all levels. To perform the analysis the inputs and outputs of the circuit must be specified (have attributes) by their ST-types: phase signals (control, indicating), data signals (outputs of bistable cells), and some auxiliary signals. In such way open-loop circuit may be represented at any level of the hierarchy.

The condition of failsafe is ensured by the signals indicativeness. Indicativeness is a concept introduced in [3]. It means that in each phase any change of the input or the internal signal of the circuit should be reflected in the change of the phase outputs. If the phase outputs fail to respond to a signal change, then there is no indicativeness, and in the case of a constant sticking it will not be detected - the circuit will continue to operate erroneously.

Thus, for the analysis on self-timing, it is necessary to check two requirements: indicativeness of signals and the lack of competitions. Both requirements must be checked for each fragment. The analysis of the lower level fragments and the upper levels analysis are performed in different ways. On the lower level, the descriptions of fragments in logical equations are used, and on the upper levels - the relationships of fragments and the results of lower levels analysis.

3.1. Lower level analysis

At first, on the lower level checking of each signal's indicativeness is performed (for all circuit's input and all outputs of the inner elements). It is carried out by the direct method. First the normal values of the phase outputs of the fragment are calculated. Next for each signal the sticking is simulated and a new calculation of phase outputs is performed. The comparison of normal and "stuck" values shows the signal's indicativeness.

For providing completeness, not numbers 0 and 1 are sent on the information inputs of the circuit while calculation but the independent variables, which can accept binary values. Thus, all possible values of information inputs of the circuit are taken into account. Some independent variables are assigned also to the memory cells that are identified during calculations.

Calculation of the phase outputs subject to the independent variables is not of any major computational difficulties, since its computational complexity is close to the linear law from the number of equations.

The next step is identification of possible competitions. This step is based on the theorem for open loop circuits [3], stating that the two-phase combination circuit is self-timed if and only if it is indicative (i.e. all internal signals are indicated on the phase outputs). This means that in the combination indicativable ST-circuit there are no competitions.

Thus, potential competition must be sought in the memory elements (bistable cells, BSC) and related signals. Signals that may cause competitions are BSC outputs. And competitions themselves may appear on those elements, to which BSC outputs are sent. Such elements will be called forth BSC followers.

There are two cases in the analysis on the competitions.

The first one involves locking BSC followers. The phase signals are also connected to followers together with BSC outputs. All these signals should be changed according to a specific discipline: phase signals should prohibit (lock) changes in object outputs in those periods when their BSC inputs are changing. In this case the problem comes to the verification of this discipline. It can be checked on base of relationships of elements, i.e. by the structural method. It is necessary to ensure that the locking signal is fed directly to the followers, while locked signals - through other elements.

The second case in the analysis of the competition is the absence of the locks. In this case, it is necessary to check the function of the element, which is BSC follower, on the lack of competitions in the corresponding phase.

After completing the analysis of the fragment on the lower level, one should prepare some data and add them to the fragment's interface for an upper layer.

The computational complexity of the analysis at the lower level is close to linear law on the number of circuit elements. Parallelism of processes does not affect the calculations noticeably.

3.2. Upper level analysis

Circuit description on any upper level should contain only fragments which were successfully tested at lower levels. The analysis is performed only on the relationships of fragments and the descriptions of their interfaces. It is required, as before, to check the signal's indicativeness and the presence of competitions. When testing, the transitive property of indicativeness [3] is essentially used.

For each phase signal, the list of signals indicated by this signal is created. Checks are made from the inputs to the outputs of the circuit. In the course of checking, each fragment gets on its phase inputs the indication lists. By combining obtained lists and parameters of the interface, the output fragment indication lists are formed. The process ends at the phase outputs of the circuit.

Checking on the competitions is carried out similarly to such verification on the lower level. All the necessary information for this is taken from relationships of fragments and descriptions of their interfaces.

Finally, the last step at the top level will be preparing some information and writing it to the interface of analyzed circuit for the next higher level.

The proposed method of hierarchical analysis was implemented and tested using software [10]. Testing of the program on medium complexity circuits - 4-bit ST-microcore (Table3) - has proved high efficiency of the method (Table 3). Analysis on any of the upper level of the hierarchy with the fullness of all the states and transitions takes tenths through hundredths of seconds. The total time of the analysis of all circuit blocks equals to 0.84 second. A one-time event-driven analysis of the same microcontroller with one initial state and one combination of the input values (not ensuring the completeness) took 7 seconds.

4. Conclusions

The existing classic (event-driven) methods of ST-circuits design do not solve many of the problems arising in their practical development. The functional approach has emerged as a way to overcome the practical difficulties.

One of the main challenges of designing ST-circuits of practical size is the analysis on self-timing. Event-driven methods require full disclosure of schemes, i.e. the study of equations of all its elements. If the size of the circuit is increasing, the complexity of the calculations increases sharply, that does not allow analyzing practically important schemes with the necessary completeness.

In the functional approach, we developed a method of hierarchical analysis of ST-circuits. The computational complexity is practically linear on the number of fragments and signals. This procedure has allowed to resolve one of the main problems of designing ST-circuits, greatly inhibiting their development, the analysis of schemes of any size.

5. References

[1] D.E. Muller, and W.C. Bartky, "A Theory of Asynchronous Circuits", *International Symposium on the Theory of Switching*, Part 1, Harvard University Press, 1959, pp. 204–243.

[2] Varshavsky, V.I., et al., *Aperiodic Automatons*, Nauka press, Moscow, 1976 (in Russian).

[3] Varshavsky, V., M. Kishinevsky, V. Marakhovsky, et al. *Self-Timed Control of Concurrent Processes*, Kluwer Academic Publishers, Dordrecht, Netherlands, 1990.

[4] Kishinevsky, M., A. Kondratyev, A. Taubin, and V. Varshavsky, Concurrent Hardware: The Theory and Practice of Self-Timed Design, Wiley Series in Parallel Computing, V.12, 1993.

[5] Kishinevsky, M., A. Kondratyev, A. Taubin, and V. Varshavsky, “Analysis and identification of speed-independent circuits on an event model*”. Formal Methods in System Design,* 4(1), 1994, pp. 3-75.

[6] Yakovlev, A., M. Kishinevsky, A. Kondratyev, L. Lavagno, and M. Pietkiewics-Kountry, “On the models for asynchronous circuit behaviour with OR causality”, *Formal Methods in System Design,* 9, 1996, pp. 189-233.

[7] Self-timed circuitry site (http://www.selftiming.ru).

[8] Y. Stepchenkov, Y. Diachenko, V. Zakharov, et al. "Quasi-delay-insensitive computing device: methodological aspects and practical implementation", *International Workshop on power and timing modeling, optimization and simulation (PATMOS'2009)*, Delft, Netherlands, 2009, pp. 276–285.

[9] Plekhanov, L., *Basics of Self-Timed Electronic Circuits*, Binom, Moscow, 2013 (in Russian).

[10] Plekhanov, L., "Design of Self-Timed Circuits: Structural Methods in Hierarchical Analysis”, *Informatics and applications*, V.8(3), Moscow, 2014, pp. 62–70 (in Russian).