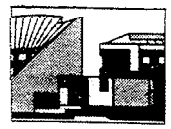
TEKNILLINEN KORKEAKOULU
DIGITAALITEKNIIKAN LABORATORIO
OTANIEMI

TEKNISKA HÖGSKOLAN
LABORATORIET FÖR DIGITALTEKNIK
OTNÄS



HELSINKI UNIVERSITY OF TECHNOLOGY
DIGITAL SYSTEMS LABORATORY
Series **B: Technical** Reports
No. 7; November 1989

ISSN **0783-540X**
ISBN **951-22-0345-6**

CIRCUITS INSENSITIVE TO DELAYS IN TRANSISTORS AND WIRES

VICTOR I. VARSHAVSKY

Professor, Doctor of Engineering
Computer Science Department
V.I. Ulyanov (Lenin) Electrical Engineering Institute
Leningrad, USSR

Research Director
R&D Coop "TRASSA"
Leningrad, USSR

Helsinki University of Technology
Department of Computer Science
Digital Systems Laboratory
Otaniemi, Otakaari 1
SF-02150 ESPOO, FINLAND

HELSINKI UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF COMPUTER **SCIENCE**
Reports of the DIGITAL SYSTEMS LABORATORY

Principal Editor: *Leo Ojala*
Professor in Digital Systems Science
Head of the Laboratory

Editorial Assistant: *Ulla Kangasniemi*
Mrs., Laboratory Secretary

Series **A** : Research Reports ISSN 0783–5396

Series B : Technical Reports ISSN 0783–540X

Offset printing: TKK Monistamo, Otaniemi
SF-02150 ESPOO 15, FINLAND

HELSINKI UNIVERSITY OF TECHNOLOGY
DIGITAL SYSTEMS LABORATORY
Series B: Technical Reports ISSN 0783–540X
No. 7; November 1989 ISBN 951–22–0345–6

Circuits Insensitive to Delays in Transistors and Wires

VICTOR I. **VARSHAVSKY**
Professor, Doctor of Engineering

Abstract: Traditionally, analysis and synthesis of self-timed circuits is based on the D. **Muller's** delay assumption according to which delays are attached to the outputs of the circuit elements. It neglects the delay distribution inside the element and its output wires after the branching. The self-checking property of delay-insensitive circuits with respect to **stuck-at** faults and the growth in topological complexity of elements and modules requires a deeper and more accurate accounting of the delays in the circuit's behaviour. In this paper, we show the possibility of implementing the distributive circuits whose behaviour is insensitive to the delays in transistors and wires, **i.e.** the **circuits** with the delays attached to the element inputs. At the lowest level of the modular description hierarchy, we consider transistors and wires, in the NMOS, CMOS and ECL technologies, as basic components.

Keywords: self-timing, speed-independence, delay-insensitivity, Muller circuits; dead-beat (aperiodic) automata, change diagrams.

Printing: TKK Monistamo; Otaniemi 1990

Helsinki University of Technology
Department of Computer Science
Digital Systems Laboratory
Otaniemi, Otakaari 1
SF–02150 ESPOO, FINLAND

Phone: $\frac{90}{+358-0}$ 4511
Telex: 125 161 htkk sf
E-mail: lab@hutds.hut.fi

Contents

1 Introduction

Synchronization, or timing, is one of the most acute problems in creating new generations of VLSI circuits and VLSI systems [1,2,3]. The growth of integration scale, both due to the size of circuit components scaling down and increase in circuit area, results in a substantially stronger effect of wires on structural and behavioural characteristics of a VLSI chip. The relative part of a chip area occupied by interconnections considerably increases. As λ scales down so does the gate delay while the propagation delay per the wire length unit remains unchanged (reportedly, may even grow [4]). If, previously, a circuit has been defined as a collection of the active decision elements interconnected by wires, then, without exaggeration, one may define it now as a collection of the wires connected by transistors.

With the growth of speed and larger distances (both relative and absolute) that the signals must be transferred dong, one can see that the requirements for the delivery (through wires) of timing and data signals to a particular place at a particular time become much stricter. The "skew" between timing and data signals, which is due to a spread in delay values, leads to the electronic arbitration condition [5,6] arising in flipflops and other feedbacked circuits. This condition may then be followed by a complete "break-down" of timing order in the system.

After analysing the requirements for timing conditions the researchers suggested [2,7] a concept of the equichronic region, a VLSI chip area inside which a reliable timing can be achieved. The size of the equichronic region becomes less with smaller gate delays and a higher clock frequency. Rather rough estimates show that a linear size of the equichronic region must not exceed the distance at which the propagation delay is equal to the gate switching delay.

To tackle the above-mentioned problems one may try to slow down the o p erating clock rate, introduce more clock sequences, make certain geometrical improvements, and, hence, complications, in the circuit design. However, all these measures can only mitigate but not obviate the principal difficulties.

One of the clear ways to overcome these problems is to build self-timed circuits [7,8]. The organization of a self-timed behaviour is based on the use of circuits that are insensitive to delays. Such circuits have also been known as Muller circuits or aperiodic automata [9,10,11].

The algebraic background of delay-insensitive circuits and their subsequent investigations [8,9,10,11,12] demonstrate the possibility of constructing an arbitrary discrete circuit (finite state-machine) whose behaviour is invariant to the delays in its components. However, though it may seem rather odd, all the known theoretical and practical results do not guarantee complete inde-

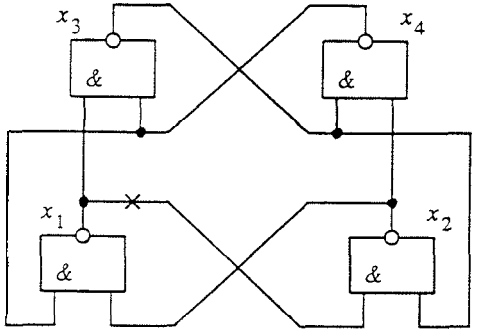
pendence of component delays. This statement does by no means contradict to the precise formal results by Muller and his successors because all these results stem from the Muller's assumption of delays. This assumption, on the today's technological level, certainly requires some special geometrical constraints.

According to the Muller's assumption the entire delay in an element is associated with its output, and the element is considered as a single unit. As a consequence, the spread in delays of the output wire after its branching can be neglected. This assumption, which has not been for years doubted, appears to be the background for the whole theory of delay-independent circuits. Certain attempts to study the problem of circuits whose behaviour is insensitive to wire delays [12,13,14] has given somewhat pessimistic conclusions, though led also to some particular solutions and assessments.

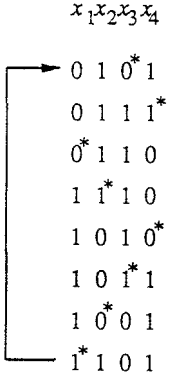
In this paper, we allow ourselves to analyse the possibility of abandoning the Muller's assumption. We begin our discussion with a very simple example.

An autonomous circuit consisting of two RS-flip-flops and the corresponding Muller diagram describing its behaviour are shown in Fig. 1. Within the framework of the Muller's assumption this circuit is semimodular, totally sequential, and its behaviour does not depend on element delays. Now, if we insert a delay into the feedback marked by a cross in Fig. 1, the corresponding Muller diagram will be as shown in Fig. 2, with the fifth variable defining the state of the delay's output. The main operating cycle of the circuit is designated in Fig. 2 by thick arrows. If the inserted delay is greater than the delay in gate 23, then in state 010*11* there may arise a condition for the circuit to leave the main cycle (as shown the dashed arrow in Fig. 2) to the 01'11'1' state in which semimodularity is violated, with subsequent transition of flip-flop (x_1, x_2) to the metastable state. The reader may easily see that there are such relationships between delays that under them the circuit would be susceptible to all imaginable troubles.

It might seem that the problems in the given example could be remedied by ordering the delays — a signal is firstly applied to the feedback connection, and only then to the other flip-flop input, as it is shown in Fig. 1 for flip-flop (x_3, x_4). This does not however exclude a wire branching, and due to some reasons that will further be considered does not eliminate the trouble. Speaking generally, we could avoid wire branches in the circuit of Fig. 1 by arranging the 'field branching" instead if the gate of the transistor (in MOS-technology) had been connected in series with the output wire. This, however, does not exclude the delay of the transistor itself. Besides, even in a complementing flip-flop (the so-called Harvard circuit), let alone any circuits with feedbacks of higher complexity, one should demand that the signal at the complementing input be ordered differently depending a particular state



a)



b)

Figure 1.

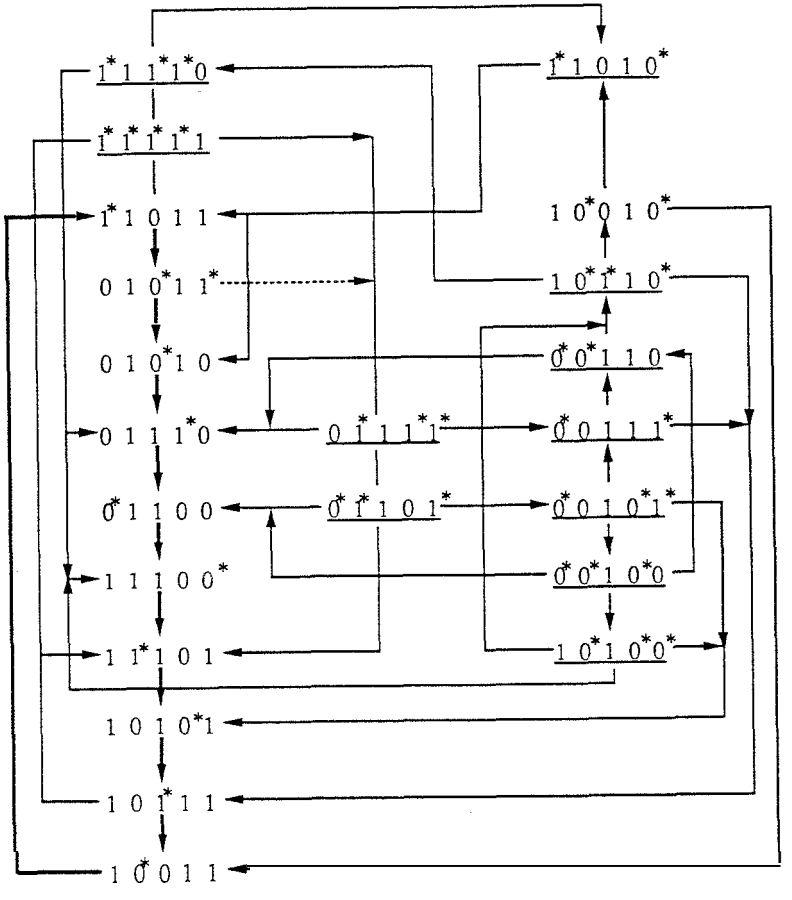


Figure 2.

of the flip-flop.

It can be seen as an established fact [12,14,15] that the wire branches, with rare exceptions, lead to the violation of delay-independence. Consequently, the question regarding the possibility of constructing a circuit invariant to wire and transistor delays is reduced to the question of the possibility of constructing a circuit having no branches in wires or having the branches which are insensitive to the delays.

In other words, while the Muller's assumption reduces all the delays in elements to the element's outputs, our task is to investigate the model in which all the delays are associated with the inputs, and not of the logical elements, which are circuits themselves, but of the transistors. Such investigations would certainly require using not just the automata- logical approaches, but also the topological findings such as using the field branches rather than the wire ones.

It seems that the first attempt to solve the problem of constructing a circuit insensitive to wire delays was undertaken by Keller [13]. He proposed a set of universal modules allowing to build circuits whose behaviour is invariant to delays in modules and intermodular connections. Unfortunately, the Keller's modules were not themselves invariant to their element delays and intramodular wires. However, by and large, the Keller's idea — to confine all the branches inside circuit elements — appears to be productive, and provided that the intraelement problem are solved the result may be successful.

Now it would be expedient to ask: should one 'enter the lists[n] in achieving the intraelement delay-independence when this problem can be effectively and fully solved on the layout design level?

There are at least two arguments in favour of efforts in this direction.

First of all it is a natural scientific curiousity that makes us interested in finding the answer to the following question: Is it possible to construct a circuit insensitive to wire and transistor delays?

The other reason has a direct and clear practical consequence. The fact that delay-independent circuits are totally self-checking with respect to stuck-at faults [12,15,16] is obviously very important and promising for their application. This property allows to implement a test-free fault localizatior during the on-line circuit operation. It is however the case that only those stuck-at faults of elements (modules, components) with respect to which the circuit is delay-independent. Thus, for example, the break and short-circuit (stuck-at faults) faults of a transistor in a NAND element do not mimic as stuck-at faults of the element itself. As a consequence, the step, with respect to the delay-independence, from the gate level to the transistor level considerably

increases the number of potentially detectable faults.

Generally speaking, the basic circuit element of an integrated circuit is a topological construct, and the central problem of a short term scale would apparently be development of synthesis methods for a base of topological constructs. This goal is however just a sweet dream today.

In this paper we avoid formal proofs, which are evidently implied from the known theoretical results, and undertake the way of showing, at the constructive level, the possibility to construct circuits whose behaviour is independent of delays in wires and transistors. We try to draw the reader's attention to all our assumptions to such an extent, of course, that we see and understand them.

The major part of work done in the area of self-timing assumes, either implicitly or explicitly, certain constraints on relations between delays. For example, it can be foreseen that the delay of a 1 micron wire would be less than that of a 1 mm wire. But as we have already mentioned, the property of delay-independence would interest us, among other things, with regard to that of self-checkability. And in this case the break of the 1 micron wire would make the wire delay infinitely large and hence greater than the 1 mm wire delay. Therefore, when we are unable to reach the point of absolutely pure delay-independence we shall be assessing the correct operation with respect to certain limit determined by the possibility of occurrence of a stuck-at fault.

The results obtained in the following sections hold for a restricted, but still rather wide, class of circuits which will be defined below.

The solutions proposed may appear quite heavy. Furthermore, every time we faced a dilemma as to whether to prefer a more economical construction or more descriptively simple one, we made our choice in favour of the latter because the objective of this work was to establish the fact that circuits insensitive to both wire and transistor delays are implementable, the fact that for many years had been doubtful.

However, understanding the complexity of the suggested constructions we should make a preliminary note. In this paper, we shall be using a language for initial specifications of circuits, the language of change diagrams. The complexity of implementations will be linear to that of initial specification, and we would like the reader to bear this in mind and hence not to be "feared" of the complexity and heaviness of the constructions.

2 Circuit Elements

As we are interested in solutions invariant to the delays in wires and transistors the transistors and wires should therefore be basic building blocks for a circuit. Such blocks, or circuit modules, will be taken in the NMOS, CMOS and bipolar (ECL) technologies.

2.1 Invertor

Two serially interconnected NMOS invertors with a D-load are shown in Fig. 3a, and the corresponding layout for this circuit with polysilicon self-coupled gates [17] is shown in Fig. 3b. The circuit has a wire branch at the point marked "c". In order to eliminate this branch let us modify the circuit layout (Fig. 3c and 3d). After doing so we have a poly wire connecting the source of transistor T_1 with its load being also the gate of transistor T_2. Such a layout provides for branching not by a wire but rather by the electrical field. The signal transfer from the first invertor to the second is done through the charging of a capacitance "conductor - undergate oxide - channel - substrate". The charging time for the capacitance is a part of the whole delay of signal propagation along the wire. The similar topological modification will be used elsewhere in this paper for the interconnection of elements.

It may seem that from the viewpoint of delay-independence the elimination of branching at point c, at least in the invertor, has no particular sense. However, let us look at the possibility of breaking the wire connecting the c point with the gate of transistor T_2. In the case of the layout of Fig. 3b this fault is not stuck-at and the state of transistor T_2 is determined by the distribution of capacitances in the circuit and by the state of the adjacent wires. For the topology of Fig. 3c, the break of the wire connecting the transistor's source with its load, below transistor T_2, leads to the stuck-at condition of $Y = 0$, and, above T_2 (in 'alive" circuit), $Y = 1$ due to a periodic subcharge of the gate of T_2 through transistor T_1.

In the CMOS invertor shown in Fig. 4a the input signal is applied on the two transistors which, without coupling the wire with the gates (Fig. 4b), will inevitably result in an additional branch.

The ECL-invertor shown in Fig. 5 contains at least three branching points, one of which, point a, provides the branching of current and hence must be kept, while the other's influence can be "abolished", for achieving the stuck-at character of faults, by the layout design measures.

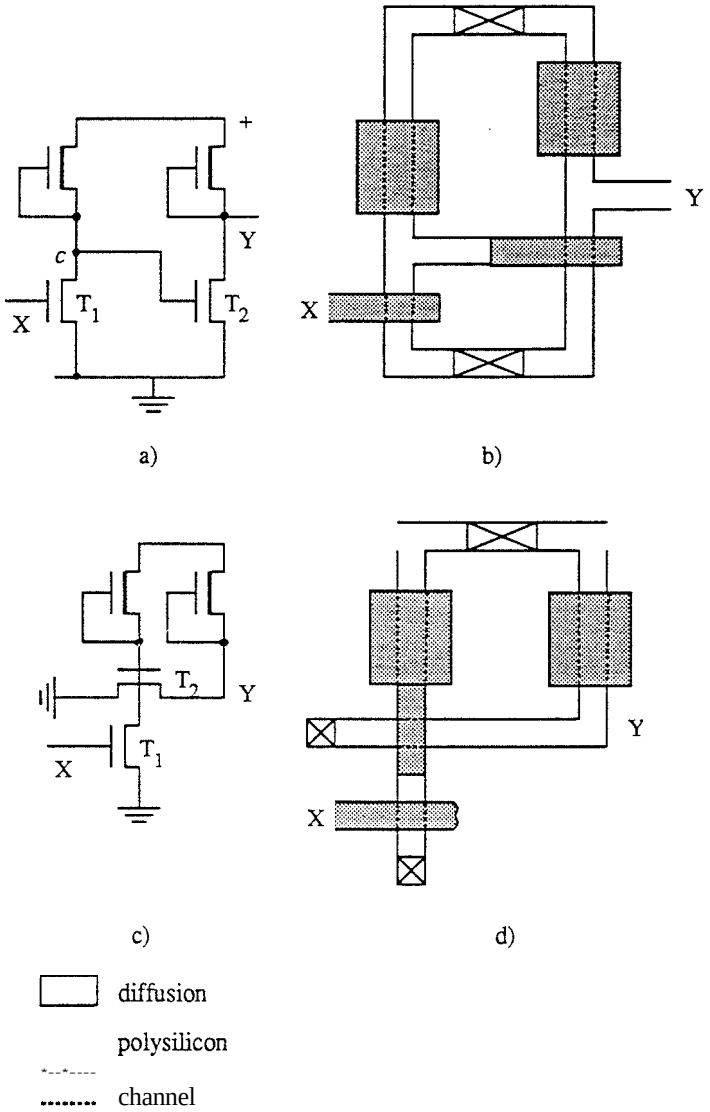


Figure 3.

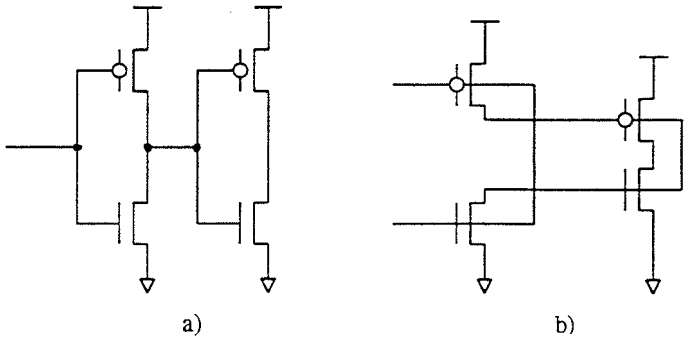


Figure 4.

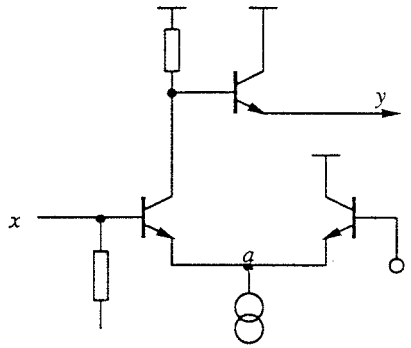


Figure 5.

2.2 Gate

An NMOS gate representing the two-input NAND (further denoted as 2NAND) function is shown in Fig. 6a. In the above we have already given the way to tackle the branching at the point c, but here we are facing another obstacle. When $x_1 = 0$ the change of input x_2 is not translated (12,151 to the circuit's output, and as a consequence can not be indicated (observed). Therefore, for $x_1 = 0$ the circuit's behaviour depends on the value of the delay of wire x_2 and transistor T_2. It is *an* apparent conclusion that a 2NAND can be used for building delay-independent circuits with respect to the delays which are bound to the inputs if and only if we have an additional output, translating changes of signals at the x_2 input (through transistor T_2!).

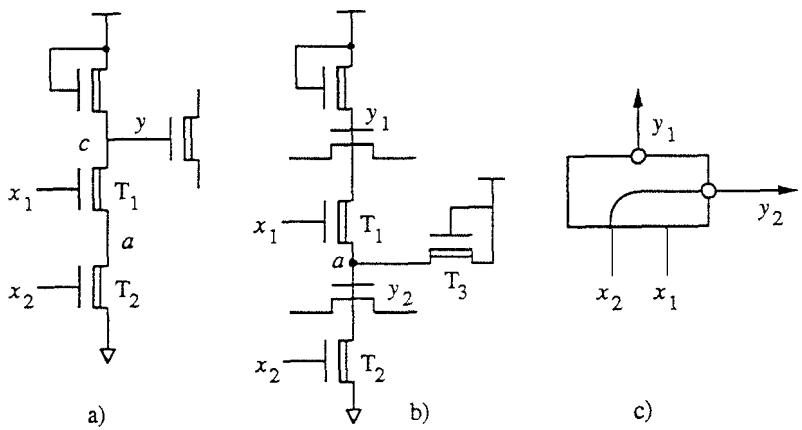


Figure 6.

In principle, we could disregard the delay of transistor T_2 and translate signal x_2 onto the output by simply expanding the corresponding wire, all the more that if $x_1 = 0$ then there is no current through T_2 and the change of its state is reduced only to the change of its gate. However this is not the case when we are interested in self-checkability. In fact, if the "source-drain" path of transistor T_2 is short-circuited, the delay of closing the transistor off becomes infinitely large and the circuit stops its operation. In order to do this, the information about transistor T_2 switching off is to be translated over to the output of the gate, which is provided by adding an extra pull-up transistor at the a point (Fig. 6b). Having so the circuit is in a natural way partitioned

into two subcircuits: the lower invertor $a = x_2'$, and the upper subcircuit with two inputs, at the gate and at the drain, $y = a \vee x_1'$.

The introduction of a pull-up transistor at point a has resulted in the branch arisen, and besides when $x_2 = 0$ (a = 1) the x_1 wire and transistor T_2 are not indicated. To overcome the latter problem we suggest principal way — to fix up the discipline of signal changes at the gate input: if $x_1 = x_2 = 1$, then the change of x_2 from 1 to 0 must be preceded by the change of x_1 from 0 to 1. Actually, if such an ordering is not provided by a natural way of signal sequencing, there can be built a composite gate with an automatic preodering of the input signal changes.

The gate of Fig. 6b will be designated as shown in Fig. 6c. Then the circuit for an auto-ordered-input gate will be represented as shown in Fig. 7a. In this figure the values of outputs are marked near the outputs. It can be seen that the circuit does not respond to the changes of signal x_2 until the x_1 signal becomes equal to 0 and y_1 has changed from 1 to 0. Necessary order of signals on inputs of gate z is provided by the semimodularity of the circuit realized. The general symbolics for an auto-ordered gate is shown in Fig. 7b.

The situation with the branch at point a is more difficult. The circuit's behaviour depends on the delay in the wire between point a and the drain of transistor T_1. The constructive coupling of the drains of transistors T_1 and T_3 makes this delay negligibly small, and in a faultless circuit it can really be neglected. In a faulty circuit the following double defect is not indicated (i.e. doesn't result in a stuck-at fault) — the break of a link between point a with transistor T_1 and the short-circuit of the drain of transistor T_1 onto the ground. It is easily seen that simple layout efforts make the probability of such a defect negligibly small.

The implementation of a CMOS gate is a bit more complex because the gate (see Fig. 8a) contains two ptransistors connected in parallel, and in a conventional implementation the opening of one of them, while the other is already open, is not indicated. The step to the circuit of Fig. 8b requires using a pass transistor, T_2, a transistor with a lower threshold, which is technologically achieved by having an additional lithography — unfortunately, we could not find a better solution (in the case of a high degree of circuit "liveness", i.e. when a guaranteed gate switching rate must be provided, and in high fault diagnosis rate requirements, we can abandon pass transistors and resort to using an intracircuit element memory). The breaks of wires into which the gates of receivers are inserted are designated by crosses, the designation to be also used in the sequel. The situation with the branch in point a is similar to that of an NMOS gate, although the consideration of the collisions arisen in it would be a useful, but not so simple, exercise.

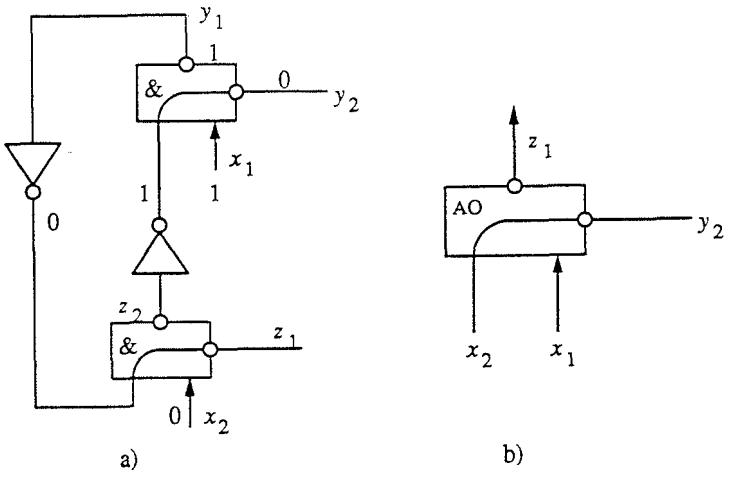


Figure 7.

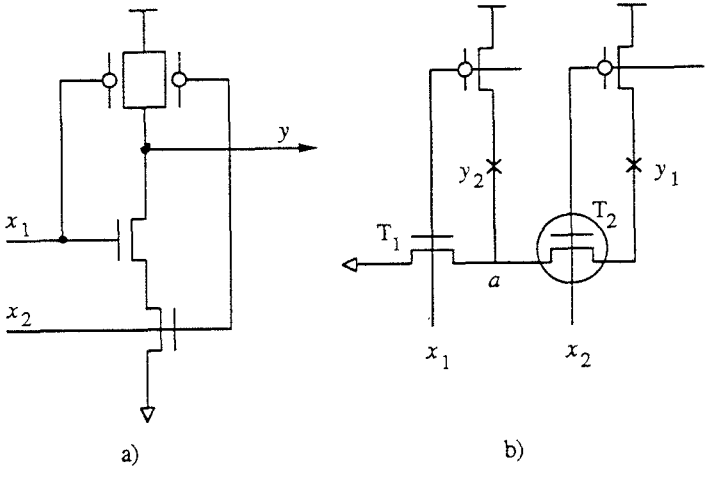


Figure 8.

An **ECL** gate is shown in Fig. 9a, and its symbolics is in Fig. 9b. The variant of its implementation demonstrating the possibility to complement the variables at the output is shown in Fig. 9c.

2.3 Valve Gate

The introduction of a circuit element called a valve gate, or more briefly a valve, has been caused only by the reasons of suitability for the further presentation. In fact, a valve can be implemented using the above-described gate **as** a building block. this is shown in Fig. 10. The basic idea of valve operation is in providing the 1-0-1 change cycle of signal x_2 if $x_2 = 1$.

It should be noted that the valve implemented as shown in Fig. 10 preserves the requirements to the signal change discipline which were necessary for a gate, **i.e.** the valve (control) signal may change its allowing value, $x_2 = 1$, only after the end of the change cycle of signal $x_1, y_1 = 1$. Using the auto-ordering removes such a constraint.

On the other hand, we could also use **as** a valve just a part of the gate circuit which forms an analogue of a **Manchester** carry chain [17], **as** shown in Fig. 11, with similar constraints on the chain length. The auto-ordering in this situation can be governed by signal y, common for all valve control variables.

2.4 RS-Flip-Flop

In principle an RS-flip-flop is built as a conventional interconnection of two NAND elements, the gates, as shown in Fig. 12. The distinction reflecting the present approach lies in the fact that the output signals of the flip-flop are taken from the additional outputs of the gates thereby providing that the change of the flip-flop output is possible strictly after the completion of transitions in all of its transistors.

We should point out that the **RS-flip-flop** considered here has, at the same time, a substantial distinction against a conventional circuit. In a commonly used RS-flipflop, built of **NANDs**, the change of state proceeds through the intermediate, transient, state (1,1). The same, (1,1), state can also be reached when the two zero values, R = S = 0, are applied to the flip-flop's inputs. In the **flip-flop** of Fig. 12 the transient state of its inputs is (0,0) that **makes** it impossible to build a generator circuit by means of a **serial** interconnection of two **RS-flip-flops** (likewise Fig. 1a).

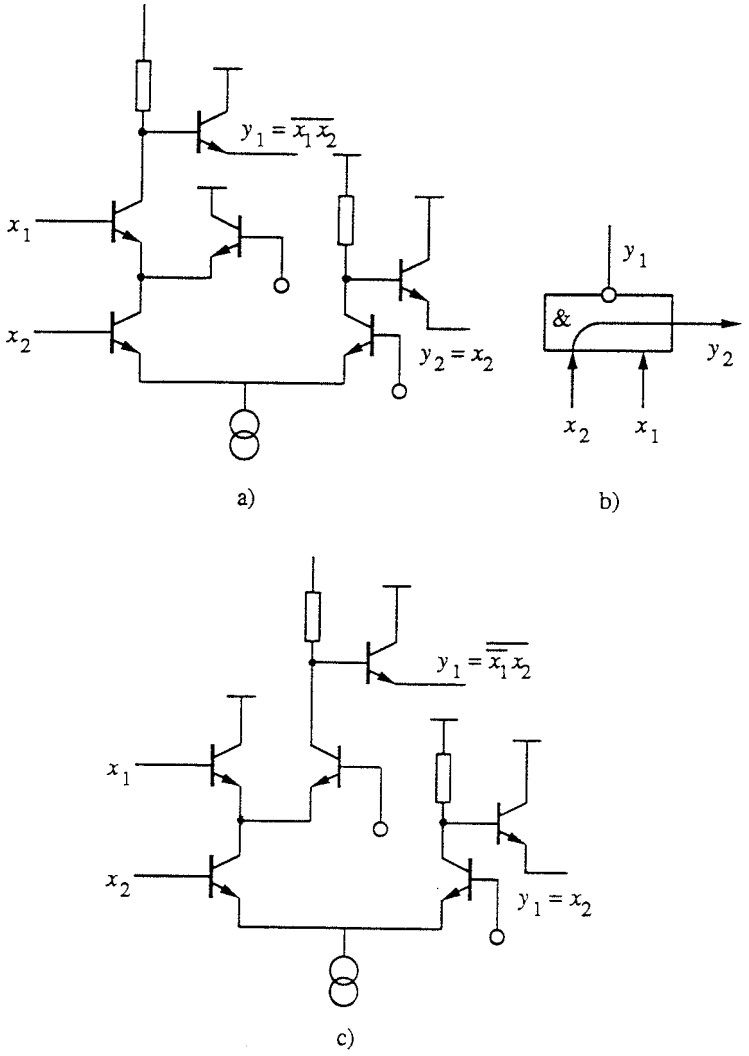


Figure 9.

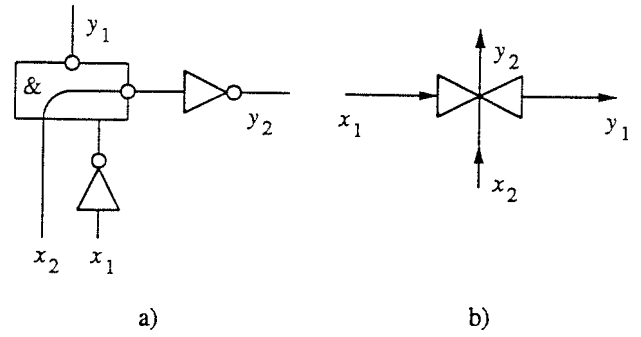


Figure 10.

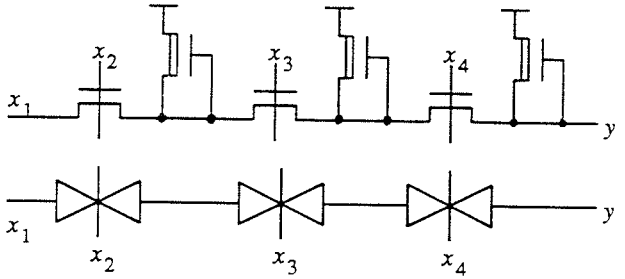


Figure 11

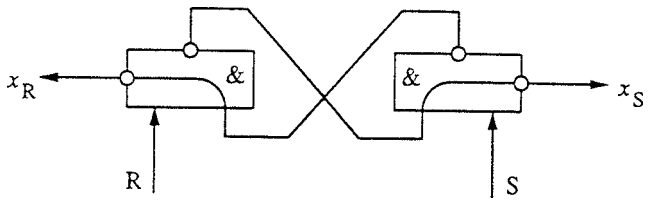


Figure 12.

2.5 Asynchronous Distributor **Cell**

A cell of asynchronous distributor of signals is shown in Fig. 13. It is a modification of the so-called David element [11,12,15,18,19]. Let us consider the behaviour of the cell using the Muller diagram notation as shown in Fig. 14.

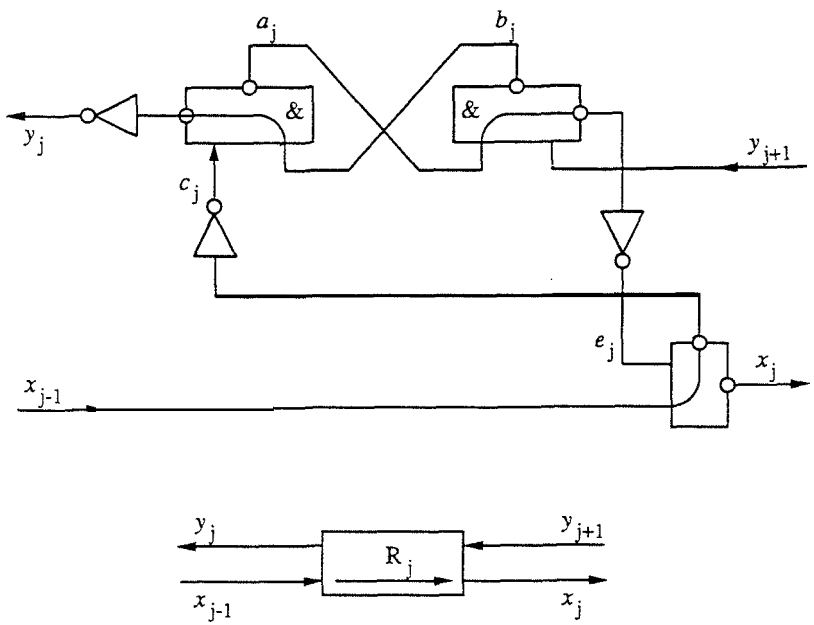
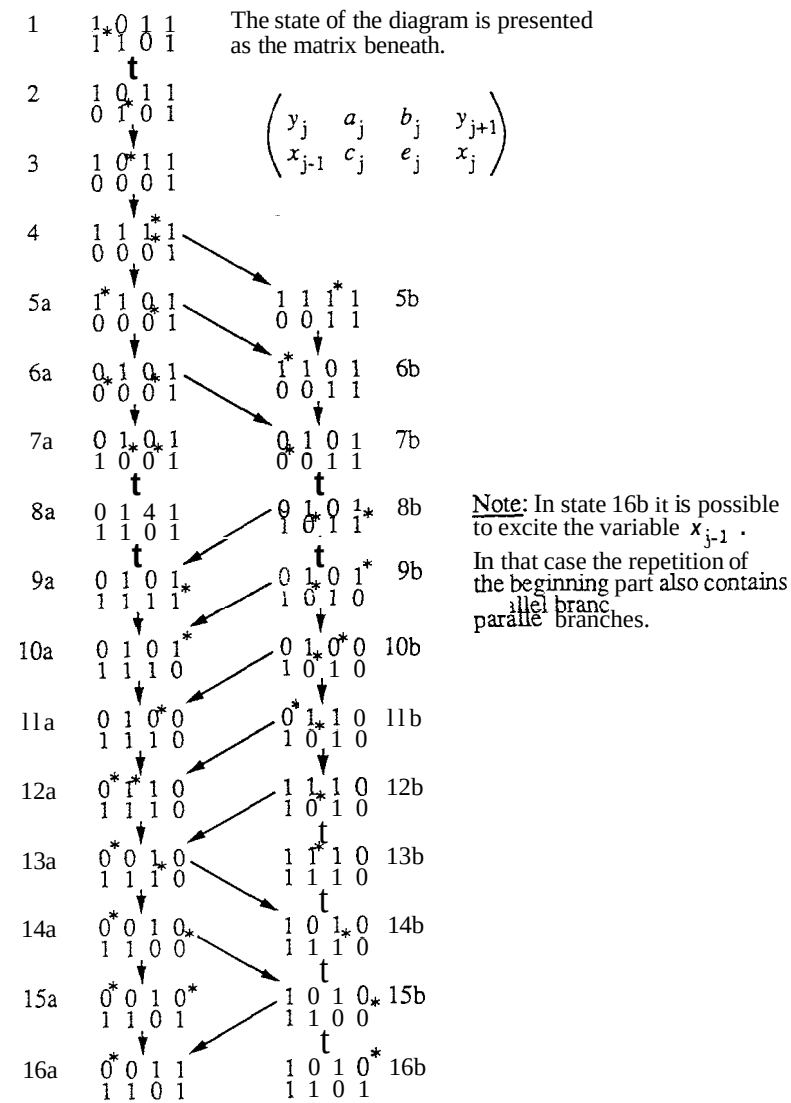


Figure 13.

In the initial state all the input and output signals of the cell are equal to 1 and the internal flip-flop has the (0,1) state. The cell is activated by the change of input signal x_{j-1}. After that the input gate, x_j, becomes non-conductive at the lower transistor (at the upper transistor it has already been cut off by signal e_j) and makes the flip-flop to change its state through invertor c_j. The flip-flop transient process begins with closing the gate a_j and, as a consequence, opening the lower transistor of gate b_j. Starting thereupon processes inside the cell proceed their ways concurrently. This is shown in Fig. 14 by chains a and b.

The change of signal y_j, beginning from state 6a serves as a reply to signal



The state of the diagram is presented as the matrix beneath.

Note: In state 16b it is possible to excite the variable x_{j-1}. In that case the repetition of the beginning part also contains parallel branches.

Figure 14.

x_{j-1} and allows the change of the letter. The transition of the circuit to state $8b$ and to subsequent states, $x_{j-1} = e_j = 1$, allow for the change of the input signal x_j which is acknowledged by signal $y_{j+1} = 0$. From this moment, the cell carries out the backward transitions towards the initial state.

It should be noted that due to the requirement of delay-independence the cell's behaviour must be such that the next activation of the cell can not be realized before the input signal has changed back to the initial state, $x_j = 1$. On the other hand, the next activation may be started before the removal of the acknowledge signal from the next cell has occurred because signal $y_{j+1} = 0$ holds up the change in the flip-flop. The issue related to the automatic blocking of unsafe collisions will be discussed later.

2.6 Controlled Flip-Flop

The RS-flip-flop discussed in Section **2.4** had only one control input per each of its arms. In our below discussions we would rather need a flip-flop whose transitions are controlled by the conjunction of arbitrary length.

But before we proceed to the description of such a flip-flop we should recall that in order to provide the circuit's delay-independence with respect to wire and transistor delays we must eliminate all branches in wires. Besides, we must also make each signal acknowledged. Such acknowledgement must generally be organized at the global circuit level, but it should also have a proper local support. With respect to the input signal, each transistor, the signal's acceptor, can and must be regarded as a delay inserted in series. In this case, the acceptor is inserted into the break of the corresponding wire. A particular point for such an insertion will again further be labeled in figures by a cross on a necessary wire.

Let us refer to Fig. 15. Let both of the valved chains transferring the output signals of RS-flip-flop to its input be closed, $a_1 a_2 = 0$, $b_1 b_2 = 0$. Then RS-flip-flop can be in one of the two steady states, e.g., in $x_1 = 0$, $x_2 = 1$. Conjunction $a_1 a_2 = 1$ allows for the transfer of value $x_1 = 0$ onto the z_1 input of RS-flip-flop that leads to the switching of the flip-flop with the result of x_1 being changed to 1 which is translated onto the z_1 input. The change of the values of control signals a_i, b_i is allowed when $z_1 = z_2 = 1$. The change of the values of control signals must here provide the transient state of conjunctions $a_1 a_2 = b_1 b_2 = 0$.

We should point out, for a meticulous reader, that the opening of the transistor controlled by signal z_1 is checked under the next switching of the flip-flop. Note also that the control conjunction may be of arbitrary length.

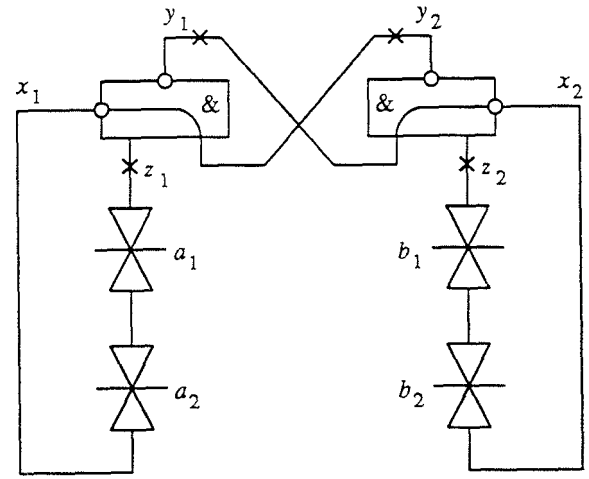


Figure 15.

3 The Synthesis Problem

It has become traditional to use Muller diagrams in the specification an analysis of delay-independent circuits [9,10]. The Muller diagram can be uniquely built for a set of functions describing the behaviour of circuit elements. On the other hand, for any non-contradictory Muller diagram one can build, with an accuracy of reassignment of the functions on the combinations corresponding to the states unused in the diagram, a set of circuit functions. However, the functions thus derived may happen to be, as they fairly often are, not realizable by a single element in a chosen circuitry basis. To overcome this problem, during the synthesis of circuit from a Muller diagram one should introduce additional variables into the latter, and the resulting circuit is described by some other, extended, diagram. For example, the method of the so-called perfect implementation [11,12,15] is concerned with twice the number of variables in a diagram. Therefore, first of all, we should make certain refinements on the statement of the synthesis problem.

3.1 The Projection of Muller Diagram

Let us have an autonomous circuit described by the following equations:

$x_1 = x_3$,
$x_2 = z$;,
$x_3 = z$;,
$x_4 = x_1' x_2' \vee x_1' x_4 \vee x_2' x_4$.

The circuit's behaviour is described by the diagram shown in Fig. 16a. Suppose that only variables x_1 and x_2 are available for observation. Their behaviour is shown in Fig. 16b. It is natural that the observer can not distinguish the variable state 0 from state 0* because the presence or absence of the element's excitation is concerned with the processes internal for the element. Consequently, the sequences of the states differing from each other only in the presence or absence of variable excitations can not be seen by the external observer. However, such an observer may be certain in that the change of the variable's value is preceded by its excitation. Therefore, the above circuit, denoted as $S(x_1, x_2, x_3, x_4)$ for the observer having access only to variables x_1 and x_2, will be expressed by the diagram shown in Fig. 16c. This diagram is no Muller diagram, or is a contradictory Muller diagram, because it has two pairs of contradictory states, viz. $(0^*1, 01^*)$ and $(10^*, 1^*0)$, the states with identical code combinations but different excitations.

We call the diagram of Fig. 16c the projection of the diagram of Fig. 16a onto variables x_1, x_2. In sequel, the Muller diagram for circuit $S(X, Y)$ will be denoted as $D[S(X,Y)]$, and its projection onto subset X of variables as $Px[S(X, Y)]$. It is natural that

$$D[S(X)] = Px[S(X)]$$

It follows from the above that $Px[S(X, Y)]$ is obtained from $D[S(X, Y)]$ by deleting the variables in subset Y from the state encodings and "glueing up" the adjacent states differing only in prsence or absence of excitation asterisks.

Let us make use of the example just to make several preliminary notes.

First, we can avoid contradictions in the diagram shown in Fig. 16c by introducing a multi-valued representation of variables, i.e. by constructing a semi-cumulative diagram (Fig. 17a). This means that the external observer counts the number of changes of each variable (in this case by modulo 4). It is clear that this counting procedure can be "mechanized" through the inclusion of a variable that will distinguish even and odd values of variables (Fig. 17b). Note that the diagrams in Fig. 16a and 17b have the same projection on variables x_1, x_2.

Second, we can see that the excitation of variable x_2 in Fig. 16a occurs before that of x_1 whereas in the diagram of Fig. 16c this is not the case. Furthermore,

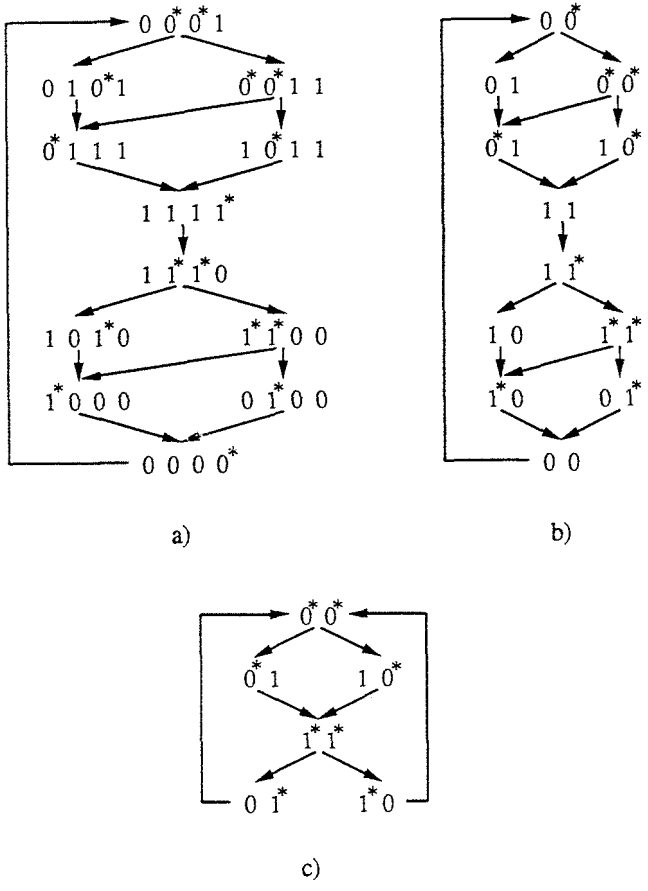


Figure 16.

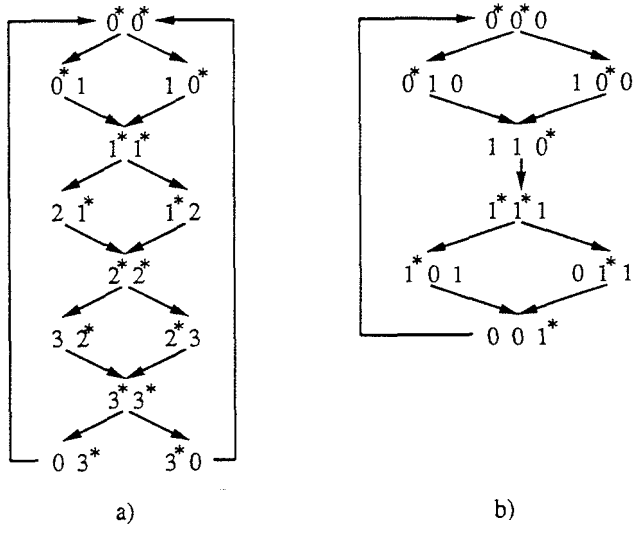


Figure 17.

the above-mentioned ordering can not be noticed by the external observer. Such an effect will further be called an implicit, or hidden, ordering.

3.2 Synthesis based on Muller diagrams

Let a behaviour of circuit $S_1(X)$ be defined by semimodular Muller diagram $D[S_1(X)]$. *The problem ofaynthesia of the circuit in a given functional base (a given set of functions implemented by the circuit elements) consists in finding a circuit* $S_2(X,Y)$ *such that:*

a) all the functions $Fj(X,Y)$ describing the circuit belong to the given base;

b) $Px[S_2(X,Y)] = D[S_1(X)]$, i.e. the projection of the Muller diagram of circuit $S_2(X,Y)$ onto subset of variables X is identical with the original diagram;

c) the behaviour of circuit $S_2(X,Y)$ does not depend on delays.

Here, we can actually start discussion on how to build circuits from simple Muller diagrams. Consider as an example the diagram shown in Fig. 17b. The switching conditions for the variables are as follows:

	Transition from 1 to 0	*Transition from 0 to 1*
x_1	Y	y'
x_2	Y	y'
Y	$x'_1x'_2$	x_1x_2

The conjunction determining the switching condition for x_j from 1 to 0 and from 0 to 1 will be called the zero transition term and the one transition term, respectively, and denote as Tj_0 and Tj_1. If the switching conditions of variables consist each exactly of one term (single-term diagram), then the corresponding circuit is implemented using controlled flipflops as shown in Fig. 18.

It should be kept in mind that the change of variable values must, for the values of z j and x'_j, have the (0,0) transient state, and the control conjunction must include, together with the variables, the outputs of their valved chains.

Note, however, that a single-term diagram is just a particular, rather limited, case of Muller diagrams. The number of terms in the switching condition depends on the number of variable changes in the circuit's operation cycles and on the presence of the so-called overtake terms. We first consider the concept of the excitation overtake. Let a circuit be given by the following equation system:

$$x_1 = x_2 \vee x_3 \vee x_4,$$
$$x_2 = x_2x_3 \vee x_4,$$
$$x_3 = x_2x'_4 \vee x'_1x_3,$$
$$x_4 = x'_2x_4 \vee x'_1x'_2x'_3.$$

The corresponding Muller diagram is shown in Fig. 19a. For this diagram, the variable switching conditions are as follows:

	Transition from 1 to 0	*Transition from 0 to 1*
x_1	$x'_2x'_3x'_4$	$x_2 \vee x_3 \vee x_4$
x_2	$x_3x'_4$	x_4
x_3	$x_1x'_2$	$x_2x'_4$
x_4	x_2	$x'_1x'_2x'_3$

The diagram is single-term with respect to all variables but x_1. The excitation of variable x_1 occurs when $x_4 = 1$. However, under this condition variable x_2 is also excited, and its transition to 1 switches variable x_4. If by the time that x_4 has changed to 0 variable x_1 has not yet changed, then variable x_2 must "overtake" the excitation of x_1. Similarly, the excitation of x_1 is further "overtaken" by variable x_3.

The general idea of the exclusion of overtake terms is quite trivial — prior to resetting the excitation term one must store the excitation in an additional

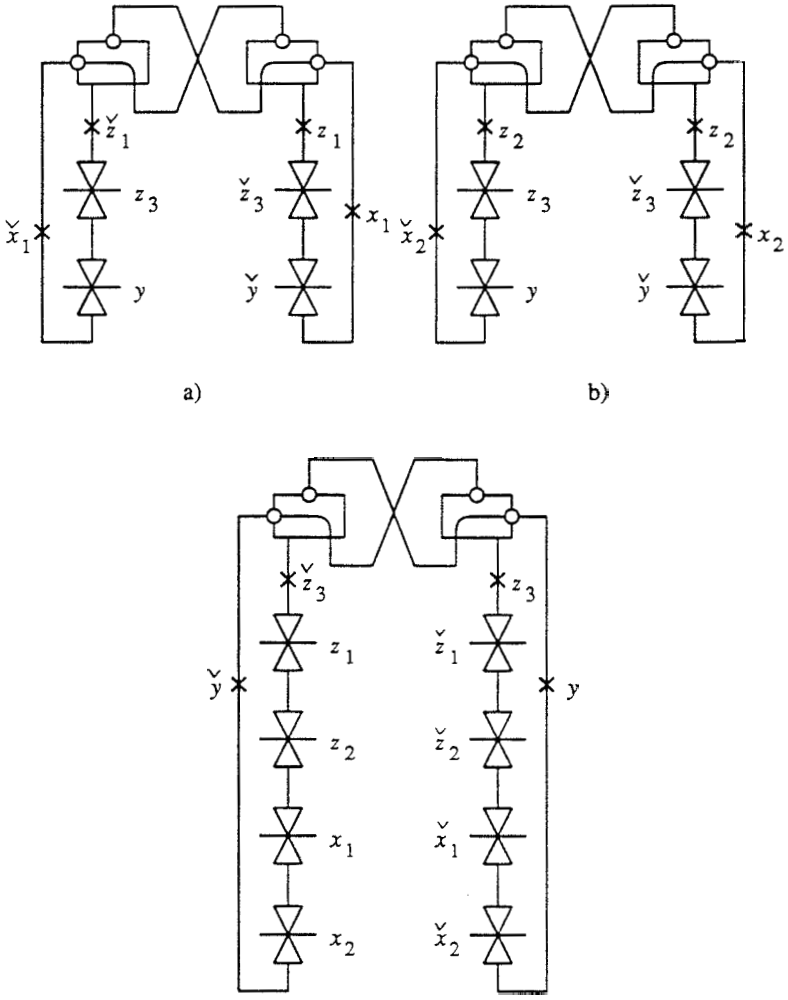


Figure 18.

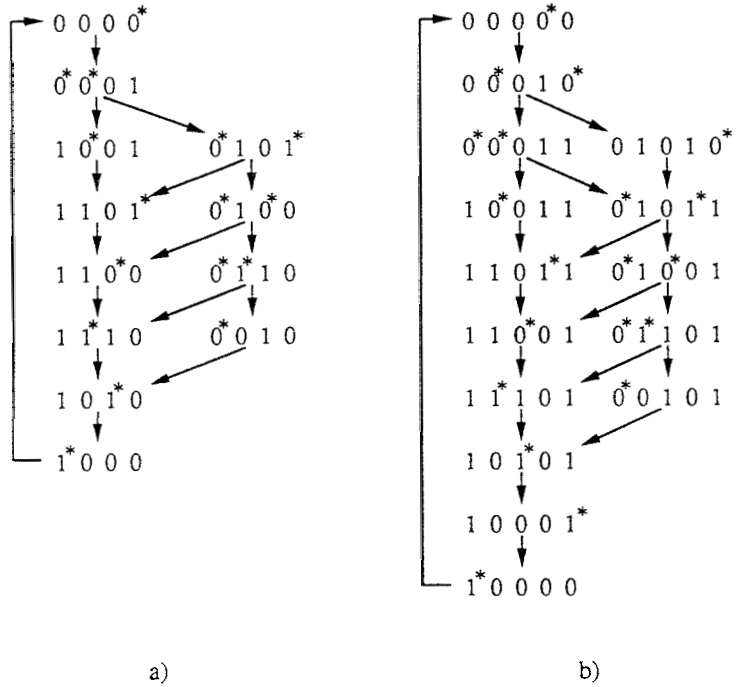


Figure 19.

variable. This technique is shown in the diagram of Fig. 19b. Note that the diagram of Fig. 19a is the projection of that of Fig. 19b onto variables x_1, x_2, x_3, x_4. The circuit corresponding to the latter diagram has the system of elements' inherent functions as follows:

$$x_1 = y,$$
$$x_2 = x_4 \vee x_2 x_3',$$
$$x_3 = x_2 x_4' \vee x_1' x_3,$$
$$x_4 = x_1' y' \vee x_2' x_4,$$
$$y = x_2 \vee x_3 \vee x_4.$$

where y is an additional variable. The variable switching conditions are as follows:

	Transition from 1 to 0	*Transition from 0 to 1*
x_1	y'	y
x_2	x_3	x_4
x_3	$x_1 x_2'$	$x_2 x_4'$
x_4	$x_2 y$	$x_1' y'$
y	$x_2' x_3'$	x_4

and the diagram shown in Fig. 19b is single-termed. Note that we have again faced with the hidden ordering. Furthermore, the elimination of overtake terms may require, in the worst case, twice the number of variables. This however does not make any allowance for the procedure of obtaining a single-term diagram in the case of variables with multiple changes.

In the diagram shown in Fig. 20a, during the operational cycle, variable x_1 changes twice. The corresponding circuit has the following element's equations:

$$x_1 = (x_2 \oplus x_3)',$$
$$x_2 = x_1 x_3' \vee x_1' x_2,$$
$$x_3 = x_1 x_3 \vee x_1' x_2.$$

and the variable switching conditions are:

	Transition from 1 to 0	*Transition from 0 to 1*
x_1	$x_2 x_3' \vee x_2' x_3$	$x_2' x_3' \vee x_2 x_3$
x_2	$x_1 x_3$	$x_1 x_3'$
x_3	$x_1' x_2'$	$x_1' x_2$

The switching of variable x_1 is controlled by four terms. The general idea of the elimination of multitermality consists in the fixation of each excitation term and representation of variable x_1 in a positional notation, with

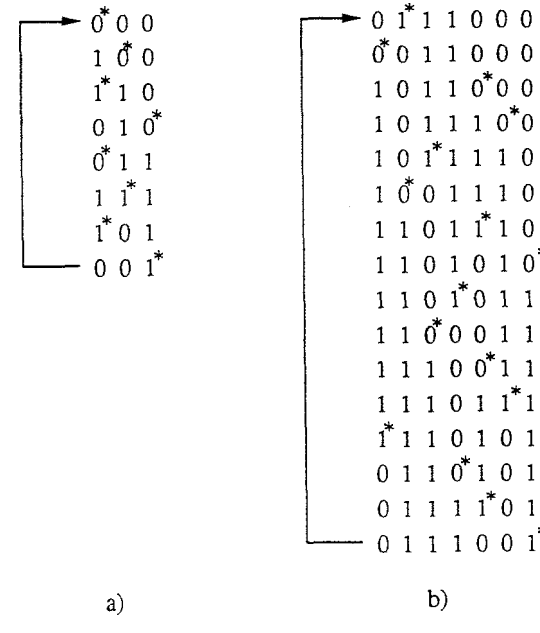


Figure 20.

subsequent transformation into a binary variable. The idea of such an implementation is clearly seen from the diagram of Fig. 20b where the first four variables, y_1 to y_4, constitute the positional representation of variable x_1.

The subcircuit forming variable x_1 has the following switching terms:

	Transition from 1 to 0	*Transition from 0 to 1*
y_1	$x_2' x_3$	y_2'
y_2	$x_2' x_3'$	y_3'
y_3	$x_2 x_3'$	y_4'
y_4	$x_2 x_3$	y_1'
x_1	$y_2 y_4$	$y_1 y_3$

It can easily be seen that the control of the changes of variables x_2 and x_3 does not alter.

All the above said provides for a transition towards single-termed diagrams and enables synthesis of circuits whose behaviour is independent of delays in wires and transistors for distributive circuits in the base of controlled flip-flops of the type shown in Fig. 15.

Nondistributive circuits involve one more type of multitermality the method for the elimination of which, in a given functional base, has not been found. Moreover, it seems very likely that nondistributive circuits can not be implemented in fully delay-independent way with respect to wire and transistor delays though we have not been able to find any satisfactory proof for this fact[1]. For clarifying the above we use an example. Consider a simple, but rather instructive, circuit:

$$
\begin{aligned}
x_1 &= x_2 \vee x_3, \\
x_2 &= x_4', \\
x_3 &= x_4', \\
x_4 &= x_1 x_2 x_3 \vee x_4 (x_1 \vee x_2 \vee x_3)
\end{aligned}
$$

whose Muller diagram is given in Fig. 21. It is important here that variable x_1 being in state 0 is excited by the two independent processes — the switchings of x_2 and x_3, which is exactly the inherent point in a violation of distributivity.

Note that the above circuits have a characteristic property that can be called the conservativity of the links, i.e. each signal incoming to the element is associated with the corresponding acknowledgement signal. The separation of acknowledgement signals in a gate is done by the procedure of ordering the signals. In order to find a solution to the problem of synthesis of nondistributive circuits of the class considered here we need either to find a technique

[1]It was just the search for this proof that caused the two years between it had been reported at the seminar at Helsinki University of Technology and its appearance in print.

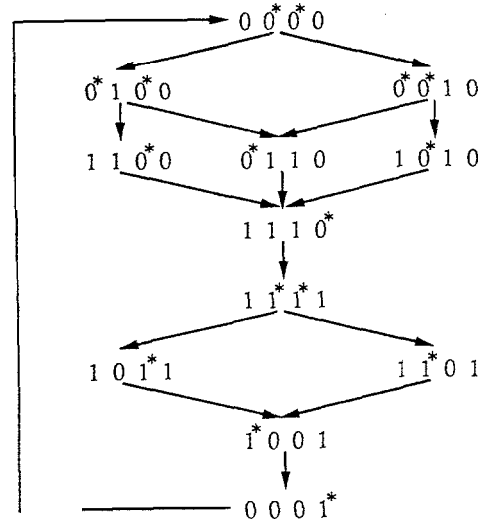


Figure 21.

for the implementation of a related hidden ordering or to prove that such a procedure does not exist.

This could be a good place to end the paper if it had not been leaving the reader with a bit of regret because of complexity of the proposed procedures, which are concerned with a considerable growth in the number of variables, and hence, that of the Muller diagram size. Actually, the increased number of variables causes the increase in the ranks of switching terms though we are unable to give a precise evaluation of such an increase under the transformation of a diagram to the single-termed one. A natural objective would be to delete the redundant variables from the general functional consideration by means of the structural synthesis which enables to construct a circuit through a direct translation of an initial specification into the circuit's representation.

3.3 Event frames

A Muller diagram is capable of demonstrating the logical possibilities of interactions between the variables and all the potential trajectories[2] of the state changes.

At the same time, a circuit's operation is a collection of events associated with the changes of states of the circuit components. On the one hand, the Muller diagram contains all the information regarding the ordering and coordination of these events but, on the other hand, this information is intermeshed with a large number of events (the changes of additional variables) playing rather an auxiliary role in the circuit's operation. We inserted these variables in accordance to our definition of the synthesis problem, and without them the circuit can not work as we wish, but it would nevertheless be a nice thing to avoid their insertion in explicit way, by having them being added to the circuit by a mechanical procedure hidden from the designer's view.

In fact, by eliminating the takeover terms and the multitermality with respect to the switchings we temporarily clear off the way the original logical variables of the circuit and only take care of the event "framework" of its behaviour. We shall call the latter an event frame.

As far as the description of the system of interacting events is concerned the languages used for the specification of event frames must also be the languages for the description of systems of interacting events, and the events themselves, according to the synthesis problem requirements, must be asynchronous. Such

[2] It is expedient to exploit here the term "trajectory" rather than a more popular now "trace" because though understanding the common between them to underline the distinction between a Muller diagram and a system of traces.

languages are for example the Petri nets, signal graphs, change diagrams. Regarding the class of our problems all these languages have tbe same descriptive power, the same hardship of the correctness checking, and preference to some of them can be given only from rather personal attitudes of the users.

A change diagram is a graph whose vertices are associated with prescribed events, the changes of variable values, and arcs correspond to the causal relations between events. Marking an arc with a token shows the presence of a cause. The set of arcs incoming to a vertex defines the conjunction of causes for the event associated with the vertex. The changing of markings which reflects the causal relationship in a change diagram is an indivisible operation. It consists of the simultaneous removal of tokens from the incoming arcs and adding them to each of the outgoing arcs. Note that the simultaneity requirement can not be implemented in an asynchronous circuit because of the natural physical phenomena in the implementation. As a consequence, in a synthesis from change diagrams the indivisibility of marking changes will be held with accuracy of hidden ordering. We should also briefly note that the absence of disjunctive vertices restricts our consideration to distributive event systems, by analogy with the distributive circuits.

The behavioural correctness of change diagrams can actually be reduced to safeness, the impossibility to have more than one token on an arc, and to correctness with respect to switching changes, the impossibility to have concurrent activation of two events that are associated with the change of one and the same variable and the impossibility of violation of the variable switching order (according to that order, the 0-1 transition of variable must strictly be followed by the 1-0 transition and vice versa). The persistence of the change diagrams is provided by their structural organization — any outgoing arc has one and only one goal vertex. Liveness and reachability depend on the initial marking and the circuit's behavioural semantics, and here are out of concern.

For every Muller diagram we can construct exactly one change diagram, and, furthermore, for a semimodular Muller diagram there is a correct change diagram. Thus, for example, the Muller diagram shown in Fig. 16a corresponds to the change diagram shown in Fig. 22. The converse is not true. A correct change diagram may have no correspondent Muller diagram. For example, the change diagram of Fig. 23 generates the diagram shown in Fig. 16c, which is not a Muller diagram. As has already been mentioned, the transition from the diagram of Fig. 16c to the Muller diagram (Fig. 17b) demands a more or less evident insertion of one additional variable. However this procedure is not always so simple and clear. Thus for the event frame shown in Fig. 24a the state diagram presented in Fig. 24b[3] contains contradictions in ail of the

[3] The meticulous reader may again try to convince him/herself in this by constructing a marking diagram for the change diagram of Fig. 24.

states and its transformation to a Muller diagram won't be a trivial task, let alone any event frames of a higher complexity.

The possibility of constructing circuits invariant to delays in wires and transistors, which model the event frame behaviour is very nice. We shall refer ourselves to this problem leaving aside the problem of transformation of event signals into the states of flip-flops representing the variables.

3.4 Synthesis of circuits modelling event frames

Consider a simple linear event frame shown in Fig. 25a and the circuit consisting of a series of asynchronous distributor cells shown in Fig. 25b. The flip-flop of the cell holds a "token", i.e. $a_j = 1, b_j = 0$ is the presence of a token in the flip-flop, and $a_j = 0, b_j = 1$ is the absence of a token. The event activated by a token consists in the cyclic, 1-0-1, change of the input signal of cell x_j. Note again that the behaviour of the cell under the interaction discipline described in Section 2.5 does not depend on the values of delays in wires and transistors.

In the initial state, a token is in cell R_1, and $x_1 = 0$. This is the beginning phase of event 1. The token is being written into cell R_2. The end of the token writing is acknowledged by signal $y_2 = 0$ that leads to the removal of the token in cell R_1 and, as a consequence, to the ending phase of event 1, with $x_1 = 1$, which indicates the beginning phase of event 2, $x_2 = 0$. Then the cycle recurs with the shift at one cell. It is clear that a hidden ordering of marking changes has taken place in the above process.

One can easily notice a constraint arisen in the context of the need for a hidden ordering. The length of the loop must be greater or equal to three. When having in an event frame a loop whose length is equal to two the loop must be augmented with a dummy event.

Now consider an arbitrary change diagram that may include vertices with several incoming and outgoing arcs. The circuits modelling the events with several incoming and several outgoing arcs are shown in Fig. 26 and 27 respectively.

Event S in Fig. 26a, the state change of wire x_3, can occur only after the occurrence of events marking the arcs d_1, d_2, d_3. However, the circuit of Fig. 26b providing the sequential writing of tokens onto arcs, the switching of the flip-flops of cells R_1, R_2, R_3, introduces interaction between events thereby "holding up" the writing of a token in cell R_2 until the complete termination of the first event and the writing of a token to cell R_3 until the complete termination of the second event. With this, the concurrency of the input events

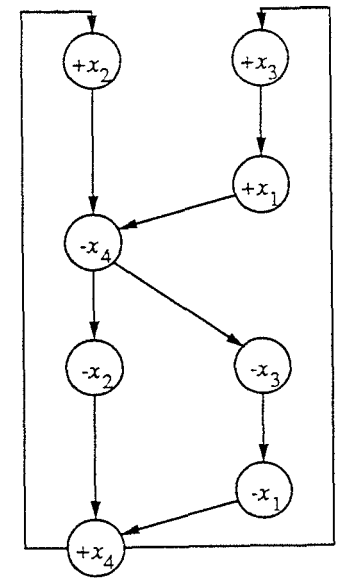


Figure 22.

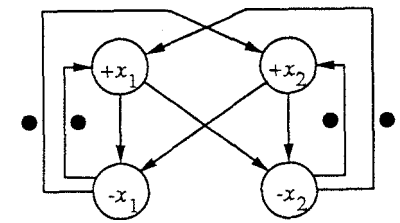
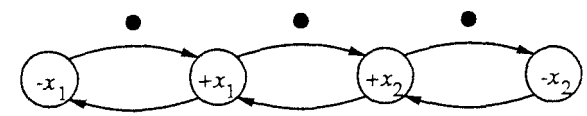
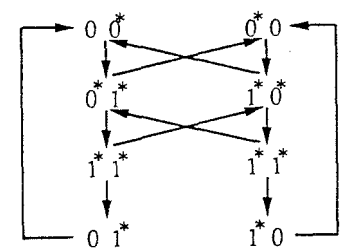


Figure 23.



a)



b)

Figure 24.

is violated, and they become explicitly ordered. To enable a hidden ordering several dummy cells, F_1, F_2, F_3, are added to the circuit thereby providing a necessary token buffering as shown in Fig. 26c.

It may seem that such a buffering has no practical sense because it still does not change the order of dummy events while adding more stages just complicates and slows down the circuit. But one should bear in mind that an event taken in an event frame may itself be rather complex as, for example, in control circuits or modular structures, and the length of the controlled event may be orders of magnitude greater than the switching time of a distributor cell. In such circumstances, concurrent implementation of the events themselves is more important than the ordering of circuit element switchings. A simple example of a complex event, which we look at below, will be the event consisting in the forming of a value of a variable.

Event S in Fig. 27a puts tokens onto several outgoing arcs. The circuit shown in Fig. 27b carries out this process sequentially. And, again, the insertion of buffering cells F_1, F_2, F_3 masks the ordering as shown in Fig. 27c.

From the above one may see that a circuit that models an event frame can be built directly from its change diagram by substituting the respective structures of asynchronous distributor cells. The complexity of the resulting circuit would be linear to the change diagram size, and the hardware costs is not greater than two cells per a change diagram arc (with accuracy to invertors implemented as shown in Fig. 26).

We end this section demonstrating the possibility of a rather simple pipelinization of a circuit modelling an event frame by means of making it safe (19,201.

The violation of safeness consists in the potentiality of having more than one token on an arc. To block this condition the writing of the next token in a distributor cell must be held up until the previous token has been removed and the state change cycle of the output wire has terminated. This is implemented by the circuit shown in Fig. 28.

3.5 State construction

An event frame defines the interaction and coordination of events while a modelling circuit models by its signals this interaction and coordination. The synthesis problem is concerned with the implementation of a state configuration, and hence we should realize the changing of states of variables in accordance with the events in the event frame.

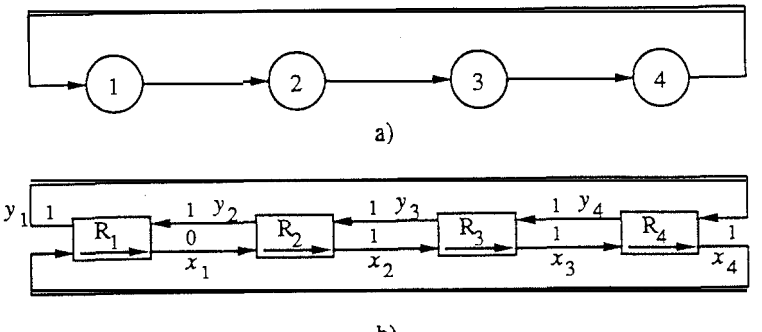


Figure 25.

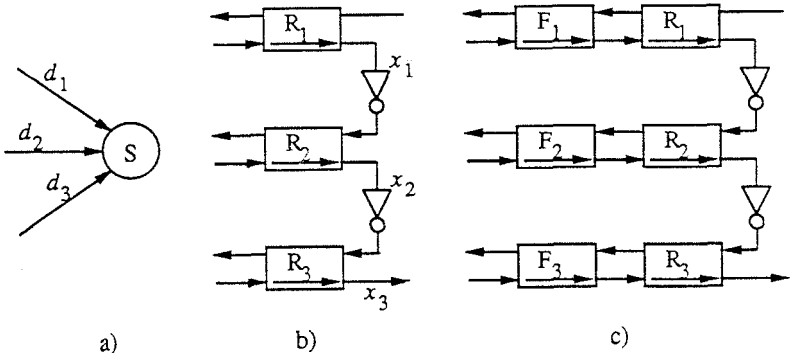


Figure 26.

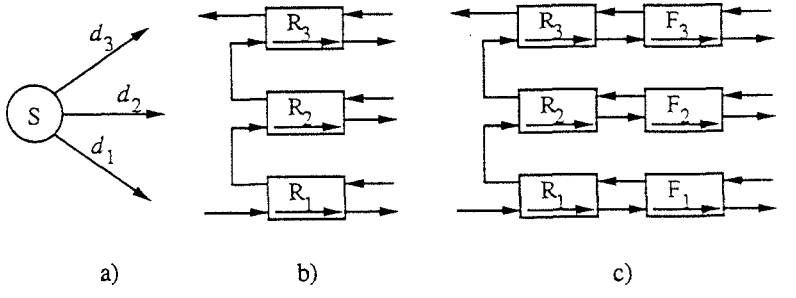


Figure 27.

If an event frame has only two events associated with variable x_j, $+zj$ and $-x_j$, i.e. during the circuit operation cycle variable x_j has one cycle of changes, then the problem of construction of signal x_j can be solved rather easily. This solution is clearly seen from Fig. 29. In the case of a multiple change of a signal in the operation cycle, the transformation work becomes harder.

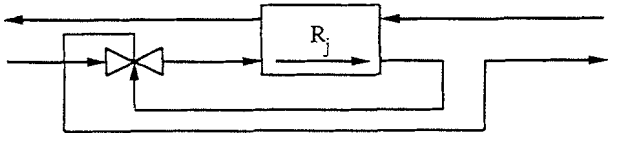


Figure 28.

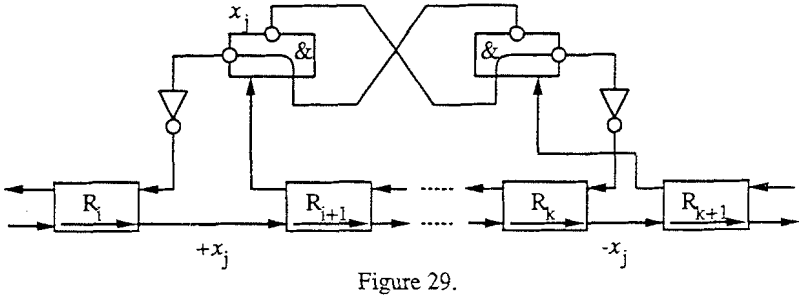


Figure 29.

We have not been able to find a more simple and clear circuit solution for the implementation of a multiple-change variable than a circuit based on the positional (as in Section 3.2), intermediate, representation of a variable. The technique for such a representation, with the "double zero" transient state on the informational variables, is clear from Fig. 30. In the circuit of Fig. 30 the insertion into an event frame is done similar to the technique shown in Fig. 29, i.e. into the break of the acknowledgement signal of the distributor cell.

It is also seen from Fig. 30 that the switching conditions of the flip-flop implementing variable x_j are as follows:

$X_1X_3 = 1$ for the transition of x_j from 0 to 1,

$X_2X_4 = 1$ for the transition of x_j from 1 to 0.

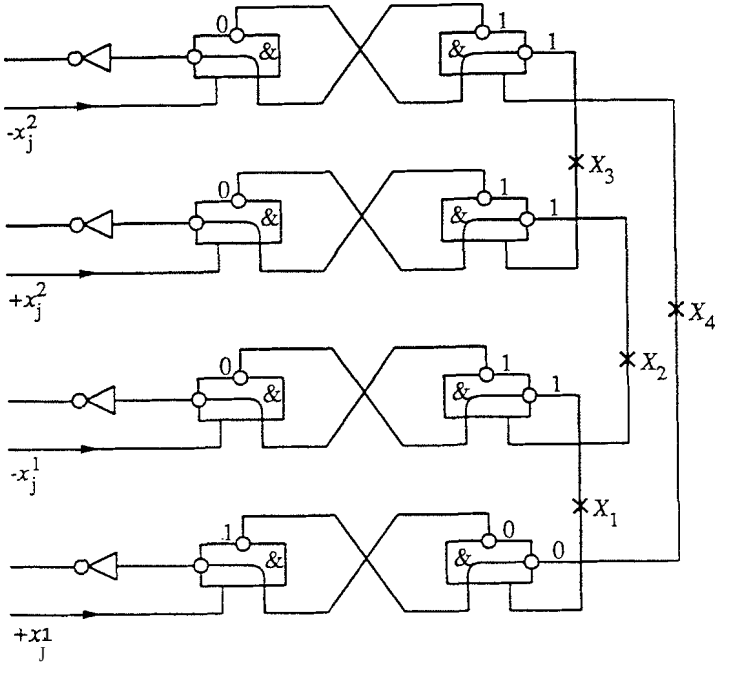


Figure 30.

The final point in our construction is to include into the circuit modelling the event frame the information about the completion of the change of a flip flop implementing variable values. Such a flip-flop is shown in Fig. 31a, and the additional valves allowing the transfer of tokens in the circuit modelling the event frame only after the informational flip-flop switching completion is shown in Fig. 31b.

4 Conclusions

It is seen from the above discussion that within the assumptions about the element functioning (see Section 2) we can implement distributive Muller circuits, the autonomous circuits with a single-term excitation on each variable change, with delays attached to the inputs of transistors, and therefore the circuits whose behaviour is independent to delays in wires and transistors. This is the precise assessment of the results discussed here.

A natural question arises regarding the connection between these results and the practice of self-timed circuit design. In fact, this is a general question about the relationship between Muller circuits [9,10], on the one hand, and self-timed circuits [1,2,3] and aperiodic automata [11,12,15], on the other.

As has already been noted, Muller circuits are autonomous automata whose connection with the environment is done through the insertion of the environment into the breaks of corresponding wires. If we also consider the environment as an automaton, then the interface of such kind would be a direct consequence of the Muller-Bartkey interconnection theorem. A simple example of such an organization can be the circuit with the Muller diagram shown in Fig. 20a. If we break the x_1 wire, i.e. insert the environment in series with the x_1 gate, then the output of the environment can be regarded as a control input of the automaton, which activates the current operation phase of the latter. The signal at the output of the x_1 gate will be the indicator of the fact of transient process completion in the current phase. One can easily see that the circuit described by the diagram of Fig. 20a is a modulo 2 counter, the complementing flipflop, with the indication of the transient process completion — to each two input signal changes there is a single change of each of the variables x_2 and x_3.

The above implies that the so-organized interaction between a Muller circuit and the environment requires a handshake on each signal, which is "affordable" for control signals but would be rather wasteful for informational ones.

Here, we face a natural question about the information interface, i.e. the connection through "long" lines, which having been doubled for the organization of a separate handshake on each signal, would make the provision of delay-independence a too cost venture.

Aperiodic automata and some other types of self-timed circuits solve the problem of delay-independence with respect to wire delays by means of using self-synchronizing codes [12,15,21], which have, except for the double-rail code, a logarithm redundancy. A typical way, in this case, would be using the one signal acknowledging the data transfer. To implement such a signal one may use various types of hysteresis flipflops (H-flip-flop), the simplest variant of which is known as a Gelement of Muller.

The situation with the Muller's Gelement leads to rather pessimistic assessment of the possibility of the implementation of a group handshake independently of delays in wires and transistors.

The Muller diagram shown in Fig. 17b is distributive and, therefore, it can be used for synthesizing a circuit whose behaviour is invariant to the delays in wires and transistors. Within the limits of the Muller's assumption about delays it can be implemented by NAND elements (for non-distributive circuits this fact has not been established) (12,151.

On the other hand, variable x_3 in this diagram corresponds to the Muller's Celement and the circuit of Fig. 32 allows to organize a group handshake on the concurrent change of variables x_1 and x_2. However, the implementation of the Celement, even under the assumption of delays attached to the element outputs, using only NANDs is unknown (!) and is unlikely to be possible, though we have already had a chance to see that such conclusions are quite risky — it had been stated in [11] that for the construction of Muller circuits one must have AND-OR-NOT gates.

It seems that here lies some principal rub causing the difference between aperiodic circuits and Muller circuits. The question regarding the establishment of a precise link between the classic Muller theory and the practice of modern self-timing is still open.

Acknowledgements

This work has been done under contract N 0103/88, Research work "TRASSA-PIN" with the Institute for Informatics of USSR Academy of Sciences. Tbe major principles of this work were also reported at the seminar of the Digital Systems Laboratory, Helsinki University of Technology, in January 1986.

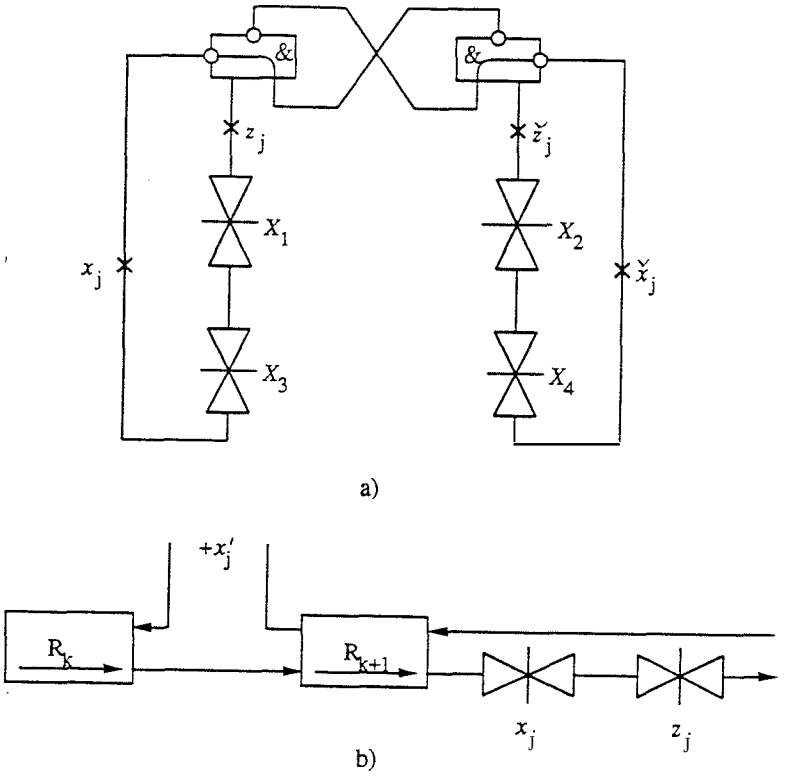


Figure 31.

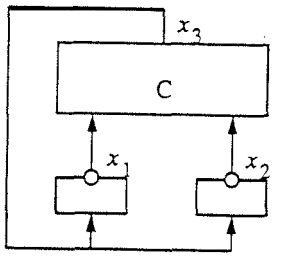


Figure 32.

References

[1] Bryant, R.E., *Report on the Workshop on Self-Timed Systems*. Cambridge 1980, MIT Laboratory of Computer Science Technology, Memo no. 166, 21 p.

[2] Seitz, C.L., System Timing. In Mead, C. and Conway, L., *Introduction to VLSI Systems*. Addison-Wesley, Reading 1980, ch. 7, pp. 218–262.

[3] Barney, C., *Logic Designers Toss out the Clock — They Moue to Self-Timed Circuits as Timing Problems Multiply*. Electronics 58(1985), no. 49, pp. 42–45.

[4] Meindl, J.D., *Interconnection Limits on Ultra Large Scale Integration*. In Horbst, E. (ed.), VLSI 85: VLSI Design of Digital Systems: Proceedings of the IFIP TC 10/WG 10.5 International Conference on Very Large Scale Integration, Tokyo, 1985, pp. 13–19.

[5] Catt, I., *Time Loss Through Gating of Asynchronous Logic Signal Pulses*. IEEE Transactions on Electronic Computers 15(1966), pp. 108–111.

[6] Mayne, D., *Minimize Computer 'Crashes "*. Electronic Design 22(1974), pp. 168–172.

[7] Seitz, C.L., *Self-Timed VLSI Systems*. Proceedings of the Caltech Conference on Very Large Scale Integration, Pasadena, 1979, pp. 345–355.

[8] Varshavsky, V.I., *Aperiodic Automta with Self-Synchronisation*. Proceedings of the IFAC International Symposium on Discrete Systems, Pt.I, Riga, 1974, pp. 9–25. (in Russian)

[9] Muller, D.E., *A Theory of Asynchronous Circuits*. Illinois 1955, Report of University of Illinois, no. 66.

[10] Muller, D.E., Bartky, W.C., *A Theory of Asynchronous Circuits I,II*. Illinois 1956, Illinois 1957, Reports of University of Illinois, no. 75, no. 78.

[11] Varshavsky, V. (ed.), *Aperiodic Automata*. Nauka, Moscow 1976, 420 p. (in Russian)

[12] Varshavsky, V.I. (ed.), *Automata Control of Asynchronous Processes in Computer and Discrete Systems*. Nauka, Moscow 1986, 400 p. (in Russian) English translation: Self-Timed Control of Concurrent Processes — to be published by Kluwer Academic Publishers in 1990.

[13] Keller, R.M., *Towards a Theory of Universal Speed-Independent Modules*. IEEE Transactions on Computers 23(1974), pp. 21–33.

[14] Udding, J.T., *A Formal Model for Defining and Classifying Delay-Insensitive Circuits and Systems*. Distributed Computing 1(1986), pp. 197–204.

[15] Varshavsky V.I. *Hardware Support of Parallel Asynchronous Processes.* Espoo 1987, Helsinki University of Technology, Digital Systems Laboratory, Series A: Research Reports, no. 2.

[16] Varshavsky, V.I., Rozenblum, L.Ya., and Taubin, A.R., *Totally Self-Checking Asynchronous Combinational Circuits and the Indicatability Property.* Avtomatika i Telemekhanika, 43(1982), pp. 138–146. (in Russian) English translation: Automation and Remote Control 43(1982), pp. 489–496.

[17] Weste, N. and Eshraghian, K., *Principles of CMOS VLSI Design, a Systems Perspective.* Addison-Wesley, Reading 1985, 531 p.

[18] Blanchard, M. et al., *Automatismes a sequences.* Tuillet 1983, Rapport final du contract DGRST H.7.2912/DERA, 420 p.

[19] Varshavsky, V. and Tiusanen, M., *Hardware Support of Concurrent Process Interaction and Synchronization: On the Principle of Autocorrect Implementation.* Espoo 1988, Helsinki University of Technology, Digital Systems Laboratory, Series B: Technical Report, no. 4.

[20] Varshavsky, V., *Hardware Support of Concurrent Process Interaction and Synchronization. The Principle of Autocorrect Implementation.* Proceedings of the Workshop on Formal Models of Parallel Computations, Novosibirsk, 1988, pp. 69–82. (in Russian)

[21] Verhoeff T., *Delay-Insensitive Codes. An Overview.* Eindhoven 1987, Technical Report, Eindhoven University of Technology, 19 p.

HELSINKI **UNIVERSITY** OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE
Reports of **the DIGITAL** SYSTEMS LABORATORY

Series A : **Research** Reports ISSN **0783–5396**

No. 5 *Heikki Tuominen:* Logic *in* Petri Net Analysis, January **1988; 53 p.**
ISBN **951–754–342–5**

No. 6 *Abbas Moslemie:* Towards Automatic Interpretation of **S-Invariants** of **Predicate/Transition Nets, August 1988;** ix+55 p. ISBN **951–754–569–X**

NO. 7 *Petri Krohn and Marko Rauhamaa:* Reduction **Transformations** of PrT-nets, October 1988; 40 p. ISBN **951–754–757–9**

No. 8 *Kaj Johansson:* Modelling **Message** Protocols for the Byzantine **Generals** Problem with **PrT-Nets,** November **1988; 65** p. ISBN **951–754–819–2**

No. 9 *Tapio Halkola, Kaj Johansson, and Leo Ojala:* Net **Theoretical Methods** in Modelling Byzantine **Consensus** Protocols in Agreement Problems, December **1988; 23** p.
ISBN **951–22–0031–7**

NO. 10 *Nisse Husberg: A* Category of Distributed Transition Systems, May **1989; 20** p.
ISBN **951–22–0023–6**

NO. 11 *Mikko Tiusanen:* A Survey of Fault-Tolerant **Clock Synchronization, June 1989;** vii+89 p. ISBN 951-22-0095-3

No. 12 K q *Johansson:* Modelling **Solutions** for Agreement Problems, June **1989; 19** p.
ISBN 951-754-984-9

Series B : Technical Reports ISSN 0783–540X

No. 1 *Ilkka Niemelä and Heikki Tuominen:* Helsinki Logic Machine: **a** System for Logical **Expertise,** December **1987; 57** p. ISBN **951–754–343–3**

No. 2 *Mikko Tiusanen:* Some Unsolved Problems in Modelling Self-timed Circuits Using Petri Nets, January **1988; 10** p. ISBN **951–754–344–1**

No. **3** *Esa Kcttuncn, Esa Montonen, and Timo Tuuliniemi: An* Interactive PrT-Net Tool for Verification of SDL-Specifications, February **1988; 43** p. ISBN 951–754–345–X

No. 4 *Victor Varshavsky and Mikko Tiusanen:* Hardware Support of Concurrent Process Interaction and Synchronization: On the Principle of Autocorrect Implementation. May **1988; 13** p. ISBN 951–754–496–0

No. 5 *Nisse Husberg:* Petri Nets in Algebraic Theories - a Category Theory Approach, June **1988; 20** p. ISBN 951–754–639–4

No. 6 *Marko Rauhamaa:* Design and Implementation of a Reduction Tool for PrT-Nets, December **1988; 13** p. ISBN 951–754–805–2

No. 7 *Victor Varshavsky:* Circuits Insensitive to Delays in Transistors and Wires, November **1989; 42** p. ISBN 951–22–0345–6