ПРИКЛАДНАЯ ТЕОРИЯ АВТОМАТОВ

УДК 519.711 DOI 10.17223/20710410/70/2

DECOMPOSITION OF A PARALLEL AUTOMATON INTO A NET OF SEQUENTIAL AUTOMATA

Yu. V. Pottosin

United Institute of Informatics Problems, National Academy of Sciences of Belarus, Minsk, Belarus

E-mail: pott@newman.bas-net.by

A method to construct a net of sequential automata that realizes the given parallel automaton is described. The parallelism relation of partial states is used to decompose a given parallel automaton. Each component automaton's set of states is based on mutually nonparallel partial states of the given parallel automaton. The state assignment of a component automaton provides decreasing power consumption of the designed device based on reducing the switching activity of memory elements. The joint low power assignment of states of component automata takes into consideration the conditional compatibility of states. The component automata exchange with binary signals. The communication between component automata is minimized.

Keywords: *parallel automaton, partial state, state assignment, complete bipartite sub-graph, weighted cover problem.*

ДЕКОМПОЗИЦИЯ ПАРАЛЛЕЛЬНОГО АВТОМАТА В СЕТЬ ПОСЛЕДОВАТЕЛЬНЫХ АВТОМАТОВ

Ю. В. Поттосин

Объединенный институт проблем информатики НАН Беларуси, г. Минск, Беларусь

Описан способ построения сети из последовательных автоматов, реализующей заданный параллельный автомат. При декомпозиции используется отношение параллельности частичных состояний заданного параллельного автомата. Множество состояний каждого из компонентных последовательных автоматов образуется на основе множества взаимно непараллельных частичных состояний заданного параллельного автомата. Кодирование состояний компонентного автомата предусматривает уменьшение энергопотребления проектируемого устройства на основе снижения интенсивности переключений элементов памяти. При совместном энергосберегающем кодировании состояний компонентных автоматов учитывается условная совместимость состояний. Компонентные автоматы обмениваются двоичными сигналами. Число межкомпонентных связей минимизируется.

Ключевые слова: *параллельный автомат, частичное состояние, декомпозиция автоматов, энергосберегающее кодирование состояний автомата, задача взвешенного покрытия.*

1. Introduction

A parallel automaton is a functional model of a discrete device that gives a concise representation of the parallelism of controlled interactive processes [1]. This model is close to widely known Petri net [2]. Unlike the classical finite sequential automaton, the parallel automaton can be in several states simultaneously. They are called *partial*. The set of all partial states that a parallel automaton can be in at some time, not being in any other state, is called *global state*.

The application of the decomposition way gives the possibility to lower the dimension of laborious tasks that appear in designing discrete devices. By decomposing a designed device into separate units, it becomes possible to decrease its power consumption by blocking clocks supplied to some units [3]. The decomposition of a parallel automaton into a net of sequential automata is worthwhile because the model of a sequential automaton allows using wide known effective methods for logical design of discrete devices. This paper considers, as well, such problems of logical design as state assignment and constructing Boolean functions of memory element excitation. The state assignment of component automata is fulfilled jointly. The problem of minimizing interconnections is considered as well.

The problem of state assignment takes a special place in logical design of discrete devices. Its solving influences considerably on complexity and power consumption of a designed device. The amount of power consumption is one of the main optimization criteria in designing discrete devices. It is caused by the tendency to increase the working time of power supply for portable devices and, on the other hand, by the tendency to lower acuity of the problem of heat rejection in designing VLSI circuits. As it is said in [4, 5], the power consumption of a circuit built on the base of CMOS technology is proportional to switching activity of its logical and memory elements. It allows solving this problem at the level of logical design. In particular, decreasing power consumption can be achieved at the stage of state assignment of an automaton.

2. The used model

A parallel automaton consists of the following objects: a set of partial states $Q = \{q_1, q_2, \ldots, q_\gamma\}$, a set of input Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$, a set of output Boolean variables $Y = \{y_1, y_2, \ldots, y_m\}$ and a set of transitions $\{\tau_1, \tau_2, \ldots, \tau_t\}$, which is a sequence of lines of the following form [6]:

$$\tau_i = S_i : -K_i \to K_i' \to S_i', \tag{1}$$

where S_i and S_i' are subsets of Q, K_i is an elementary conjunction of variables from the set X, and K_i' is an elementary conjunction of variables from the set Y.

The sense of the line (1) is the following. If a partial automaton is in states forming the set S_i and Boolean variables took values that convert K_i to 1, then K_i' takes value 1 and the automaton comes to partial states in S_i' from the states composing S_i. In other words, let $P = \{P_1, P_2, \ldots, P_p\}$ be the set of all reachable global states of a given parallel automaton. Then if $S_i \subseteq P_g$, where P_g is a current global state of the automaton, and the automaton receives binary signals that turn the conjunction K_i into 1, then the global state will be $P_h = (P_g \setminus S_i) \cup S_i'$ at the next time and the automaton will produce binary signals that turn K_i' into 1. Any conjunction, K_i and K_i', can be absent in the line. The absence of K_i means its identical equality to 1. The absence of K_i' means, according to interpretation of the model, that either all the variables in Y are equal to 0 or the values of them do not change. In this paper, we do not consider output signals and omit K_i'. As well as for a sequential automaton, the synchronous or asynchronous implementation can be for

a parallel automaton. In a synchronous implementation, the time is divided into fixed units, during which the automaton goes from one state to another. In an asynchronous implementation, the duration of the unit is not fixed and is determined by the points of changing external input signals. When a signal comes to input, the automaton goes to stable partial states and do not go from them up to the next changing input. This changing must not occur until the partial states being stable.

The following restrictions are introduced in the model:

1) The initial global state is a one-element set. For the sake of determinacy, it can be $\{q_1\}$.
2) For two different lines, i-th and j-th, $S_i = S_j$ if $S_i \cap S_j \neq \varnothing$.

There are a number of other restrictions given in [1] and connected with correctness of an automaton description. They will not be considered here, as the correctness problem is not regarded here. Also, the output signals will not be considered here.

Example 1. Let a parallel automaton be defined by the following sequence of lines:

$$\begin{aligned}
&\tau_1 = 1 : - \overline{x}_1 x_2 \to 10; && \tau_5 = 4 : - \overline{x}_1 \to 7;\\
&\tau_2 = 10 : - \overline{x}_2 \to 2.3.4; && \tau_6 = 7 : - \overline{x}_2 \to 9;\\
&\tau_3 = 2 : - x_1 \to 5; && \tau_7 = 8.9 : - \overline{x}_2 \to 6;\\
&\tau_4 = 3.5 : - x_2 \to 8; && \tau_8 = 6 : - x_1 \to 1.
\end{aligned}$$

Here $Q = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ and $X = \{x_1, x_2\}$. Let us take one-element set $\{1\}$ as an initial global state. The first line (transition τ_1) means that the automaton goes from state $\{1\}$ and comes to state $\{10\}$ in the next time unit if $x_1 = 0$ and $x_2 = 1$. The state $\{10\}$ is a global one, as well. The automaton stays in state $\{1\}$ at any other value combination of x_1 and x_2. The automaton goes from global state $\{10\}$ to partial states 2, 3 and 4 at $x_2 = 0$ (transition τ_2). Those partial states constitute the global state $\{2, 3, 4\}$. At the next time unit, the automaton changes the partial state 2 for partial state 5 (transition τ_3) and the automaton will be in global state $\{3, 4, 5\}$. Having observed the functioning of the automaton in this way, we get global states $\{2, 3, 7\}$, $\{2, 3, 9\}$, $\{3, 5, 7\}$, $\{3, 5, 9\}$, $\{7, 8\}$, $\{4, 8\}$, $\{8, 9\}$ and $\{6\}$.

3. Constructing a set of sequential automata implementing a given parallel automaton

A way of constructing a set of sequential automata implementing a given parallel automaton is described in [7, 8]. Let a parallel automaton B with a set Q of partial states be given. Its description is a sequence of lines of the form (1). Let us consider a set $N = (X, A_1, A_2, \ldots, A_n)$, where X is the set of input Boolean variables, the same as in the description of B, and $A_1, A_2, \ldots, A_n$ are component automata with sets $Q_1, Q_2, \ldots, Q_n$ of states. Each automaton A_j is one without outputs, i.e., only transitions are given for them in the form similar to (1):

$$q_j : - \alpha_j \to q^{j'}, \tag{2}$$

where α_j is a predicate of variables in X and states of component automata A_1, A_2, …, A_{j-1}, A_{j+1}, …, A_n. Its value is 1 at a certain value combination of some input variables and a certain set of states of some component automata. The automaton A_j goes to state $q^{j'}$ from q^j if $\alpha_j = 1$, otherwise it remains in q^j.

A net N *implements* an automaton B if there exists a mapping φ of $D \subseteq Q_1 \times Q_2 \times \ldots \times Q_n$ into a set of subsets of Q, such that for any transition (1) of B at $S_i \subseteq \varphi(q^1, q^2, \ldots, q^n)$

it goes to $S'_i \subseteq \varphi(q^{1'}, q^{2'}, \ldots, q^{n'})$ if $\alpha_j = 1$, and $S'_i \subseteq \varphi(q^1, q^2, \ldots, q^n)$ if $\alpha_j = 0$, according to (2). The described net N implements in the sense of [9, 10] a sequential automaton A, modelling B with global states of B.

The set of states Q_j of a component automaton A_j $(j = 1, 2, \ldots, n)$ of N is specified as follows. Every state $q^j \in Q_j$ is in a mutually one-to-one mapping with a partial state q of a parallel automaton B. Among the partial states that correspond to states in Q_j, no pair can be parallel. Partial states of a parallel automaton are called parallel if the automaton can be in them at the same time. The mapping is denoted as $f_j(q^j) = q$. Let the set $\{A_1, A_2, \ldots, A_n\}$ of component automata of N be enough to exist at least one component A_j with such a state q^j that $f_j(q^j) = q$.

The transitions in the component automata are specified according to (2) as follows. Each line τ_i of the form (1) from the description of B corresponds to the set of transitions in those component automata A_j, each of which has such a state q^j that $f_j(q^j) \in S_i \cup S'_i$. The left part of (2) for the component automaton A_j is q^j, for which $f_j(q^j) \in S_i$. There is exactly one such a state in Q_j because, according to determination of Q_j, no pair of states in Q_j corresponds to a pair of parallel partial states of B. The state $q^{j'}$ in (2) is determined such that $f_j(q^{j'}) \in S'_i$. At that, $\alpha_j = 1$ if and only if $K_i = 1$ and $\{f_1(q^1), f_2(q^2), \ldots, f_{j-1}(q^{j-1}), f_{j+1}(q^{j+1}), \ldots, f_n(q^n)\} \supseteq S'_i$.

The set of sequential automata $\{A_1, A_2, \ldots, A_n\}$, which constitutes the net N implementing the given parallel automaton B, is constructed in the following way. We obtain all maximal sets of mutually non-parallel partial states of B. The set is maximal in the sense that any partial state not belonging to it is parallel to some state belonging to it. Then, a set of these sets covering all the pairs of partial states connected in transitions must be obtained. Each set in the obtained cover corresponds to the set of states of one of the component sequential automaton constituting the desired net N.

In the case of low power assignment of a parallel automaton, the global states should be considered and the parallelism of partial states can be easily determined according to the global states. For the parallel automaton from the Example 1, the matrix of parallelism relation of partial states is as follows, where only the elements above the main diagonal are present because of symmetry of the relation:

$$\begin{array}{cccccccccc} 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & \\ \left[\begin{array}{ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ & & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ & & & 1 & 0 & 0 & 1 & 0 & 0 \\ & & & & 0 & 1 & 0 & 1 & 0 \\ & & & & & 0 & 0 & 0 & 0 \\ & & & & & & 1 & 0 & 0 \\ & & & & & & & 1 & 0 \\ & & & & & & & & 0 \end{array}\right] & \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array} \end{array} .$$

We consider this matrix as the adjacency matrix of the graph representing parallelism relation. The mentioned sets of non-parallel partial states correspond to the independent sets in the graph. The methods to find independent sets in a graph are described in [11]. The sets $\{1, 2, 5, 6, 8, 10\}$, $\{1, 3, 6, 8, 10\}$ and $\{1, 4, 6, 7, 9, 10\}$ are independent sets in the graph under consideration. All the sets constitute the only shortest cover, and so, the cover problem must not be solved as it is done in [7, 8].

The states of the component automata are convenient to be denoted by the same symbols that the corresponding partial states of the given parallel automaton are denoted. So three

automata will constitute the desired net: A_1 with set of states $Q_1 = \{1, 2, 5, 6, 8, 10\}$, A_2 with set of states $Q_2 = \{1, 3, 6, 8, 10\}$ and A_3 with set of states $Q_3 = \{1, 4, 6, 7, 9, 10\}$. The transitions between states are determined as it is described above according to (2). We give the behavior of the obtained net N as a system of discrete functions $q^{1'}$, $q^{2'}$ and $q^{3'}$ that take values from Q_1, Q_2 and Q_3, respectively. Their arguments are Boolean variables x_1, x_2 and multivalued variables q^1, q^2 and q^3 that take values from Q_1, Q_2 and Q_3. Table 1, which has the generally accepted form of flow tables for an automaton, represents this system. The rows of Table 1 correspond to the states of the component automata, and the columns to the triplet $x_i, x_j, q^{k'}$ $(i, j = 1, 2;\ k = 1, 2, 3)$. Any element of Table 1 represents the state where the automaton goes under the conditions shown in the row and column.

Table 1

q^1	q^2	q^3	x_1x_2											
			00			01			11			10		
			$q^{1'}$	$q^{2'}$	$q^{3'}$	$q^{1'}$	$q^{2'}$	$q^{3'}$	$q^{1'}$	$q^{2'}$	$q^{3'}$	$q^{1'}$	$q^{2'}$	$q^{3'}$
1	1	1	1	1	1	10	10	10	1	1	1	1	1	1
10	10	10	2	3	4	10	10	10	10	10	10	2	3	4
2	3	4	2	3	7	2	3	7	5	3	4	5	3	4
2	3	7	2	3	9	2	3	7	5	3	7	5	3	9
5	3	4	5	3	7	8	8	7	8	8	4	5	3	4
2	3	9	2	3	9	2	3	9	5	3	9	5	3	9
5	3	7	5	3	9	8	8	7	8	8	7	5	3	9
5	3	9	5	3	9	8	8	9	8	8	9	5	3	9
8	8	7	8	8	9	8	8	7	8	8	7	8	8	9
8	8	4	8	8	7	8	8	7	8	8	4	8	8	4
8	8	9	6	6	6	8	8	9	8	8	9	6	6	6
6	6	6	6	6	6	6	6	6	1	1	1	1	1	1

As a result of state assignment, when any variable q^j (relatively, $q^{j'}$) is replaced by a Boolean vector with components z_i^j (relatively, $z_i^{j'}$) representing states of memory elements, the system of functions given by Table 1 is transformed into a system of Boolean functions.

4. The method for state assignment of a sequential automaton

The iterative way described in [11] for state assignment is used. The search for the maximum cut in a weighted graph is based on it. Let the state assignment of an automaton A be required. That is, the Boolean vectors $(z_1, z_2, \ldots, z_k)$, called state codes, must be assigned to the states q of A, and different states are assigned with different codes. Partial codes $(z_1, z_2, \ldots, z_j)$, $j < k$, and a weighted graph $G = (V, E)$, whose vertices correspond to the states of A, describe the current situation in the way fulfilling. An edge connects two vertices in G if and only if the corresponding states have the same partial code. The partial codes are empty and G is a complete graph in the initial situation. Every edge $v_s v_t \in E$ has a weight w_{st} proportional to $1 - p_{st}$, where p_{st} is the probability of transition between states q_s and q_t (independently of direction) corresponding to vertices v_s and v_t. Evidently, the probability p_{st} is equal to the sum of the probabilities of transitions from q_s to q_t and from q_t to q_s. To lower the switching activity of memory elements, the Hamming distance between the codes of states q_s and q_t in the space of Boolean variables $z_1, z_2, \ldots, z_k$ must be made shorter if p_{st} is high.

The process of state assignment of a given automaton is a sequence of steps. At the i-th step, a partition of the vertex set V of G into two subsets, V_1 and V_2, is obtained. The variable z_i is introduced and receives the value 0 (or 1) for the states corresponding

to vertices in V_1 and value 1 (or 0) for the states corresponding to vertices in V_2 if those vertices are ends of any edge. Then, the edges connecting vertices in V_1 with vertices in V_2 are removed, and the next step, $(i+1)$-th one, is fulfilled. The process is over when graph G becomes empty.

The problem to partition V into V_1 and V_2 is reduced to finding the maximum cut in G, i.e., finding such a partition that the sum of weights of the edges connecting the vertices of V_1 to the vertices of V_2 would be maximum. At the last step, only those edges remain that correspond to the pairs of states connected with transitions of comparatively large probabilities. The Hamming distances between the codes of those states are equal to one. To find the cut, the method described in [12] can be applied. It is a sequence of steps, at each of which a vertex v is selected in V_1 and carried to V_2. The initial meanings are $V_1 = \varnothing$ and $V_2 = V$, and the vertex v is selected in the following way.

Let d be the sum of weights of the edges incident with $v \in V_2$, and c be the sum of weights of edges connecting v with the vertices in V_1. The transfer of the vertex v from V_2 to V_1 increases the sum of weights of edges connecting the vertices from V_1 with the vertices in V_2 by $h = d - 2c$ if it is positive. At the first step, h is equal to d. At any step, the vertex v with maximum value of h is selected. The process comes to the end when h is not positive for all the vertices in V_2.

5. Calculating probabilities of transitions

The following assumptions are accepted in calculating probabilities of transitions between states of a sequential automaton. The automaton is completely specified; all the states are mutually reachable, i.e., for every two states there exists a sequence of input signals transferring one of them to the other; the automaton works a long time enough.

The probability of transition of a sequential automaton from a state q_i to a state q_j caused by an input signal $x = (x_1, x_2, \ldots, x_n)$ is equal to the probability of coming x. If there are several input signals transferring the automaton from q_i to q_j, the conditional probability p'_{ij} of such a transfer is equal to the sum of the probabilities of those signals as a probability of incompatible events. The condition is that the automaton is in the state q_i. The absolute probability p_{ij} of the transition from q_i to q_j for all the time of the automaton working is equal to the product $p(q_i)p'_{ij}$, where $p(q_i)$ is the probability that the automaton is in the state q_i — this event and the incoming signals that transfer the automaton from q_i to q_j are independent events.

To calculate the probabilities $p(q_i)$, $i = 1, 2, \ldots, m$, where m is the number of states of the automaton, the Chapmann — Kolmogorov equations for discrete-time Markov Chains [13] can be applied. That method was used in [7, 8, 14]. Similarly to Kirchhoff's low in electrical engineering, one may say that the sum of the probabilities of transitions to some state is equal to the sum of the probabilities of transitions from this state. Based on the considerations above, the following equations with unknown quantities $p(q_i)$ $(i = 1, 2, \ldots, m)$ can be derived:

$$\sum_{i=1}^{m} p(q_i)p'_{ij} = p(q_j), \quad j = 1, \ldots, m,$$

$$\sum_{i=1}^{m} p(q_i) = 1.$$

The probabilities p'_{ij} must be known. So, having solved this system of equations, the probabilities $p(q_i)$ will be obtained. As it was said above, the absolute probability p_{ij} is defined as $p_{ij} = p(q_i)p'_{ij}$.

An input signal of a component automaton in a net consists of an input signal of the net and signals on the states of the other components of the net. Due to interconnection in the network, the state of a component automaton and the input signal it receives are not independent events. To calculate the probabilities of transitions between component automata, the modeling a parallel automaton by a sequential automaton is suggested. The states of the modeling sequential automaton are the global states of the given parallel automaton. The probabilities of transitions between the global states are defined as it is shown above for the modeling sequential automaton. The probabilities of transitions between the partial states and, relatively, between the states of component automata are determined using the probabilities of transitions between global states. The probability of the transition between partial states q_i and q_j is equal to the sum of the probabilities of the transitions between those global states P_s and P_t, for which $q_i \in P_s$ and $q_j \in P_t$, or $q_i \in P_t$ and $q_j \in P_s$.

The transitions between the global states of the parallel automaton in the considered example is shown in Table 2, where rows and columns correspond to global states, and its entry is the condition of the transition from the global state corresponding to the row into the global state corresponding to the column. Using Table 2, the conditional probabilities are determined easily. Table 3 shows them with common denominator. Here, we regard binary input signals x_1 and x_2 as independent and equally probable.

Table 2

States	$\{1\}$	$\{10\}$	$\{2,3,4\}$	$\{2,3,7\}$	$\{3,4,5\}$	$\{2,3,9\}$	$\{3,5,7\}$	$\{3,5,9\}$	$\{7,8\}$	$\{4,8\}$	$\{8,9\}$	$\{6\}$
$\{1\}$	$x_1 \vee \overline{x}_2$	$\overline{x}_1 x_2$										
$\{10\}$		x_2	$\overline{x}_2$									
$\{2,3,4\}$				$\overline{x}_1$	x_1							
$\{2,3,7\}$				$\overline{x}_1 x_2$		$\overline{x}_1 \overline{x}_2$	$x_1 x_2$	$x_1 \overline{x}_2$				
$\{3,4,5\}$					$x_1 \overline{x}_2$		$\overline{x}_1 \overline{x}_2$		$\overline{x}_1 x_2$	$x_1 x_2$		
$\{2,3,9\}$						$\overline{x}_1$		x_1				
$\{3,5,7\}$								$\overline{x}_2$	x_2			
$\{3,5,9\}$								$\overline{x}_2$			x_2	
$\{7,8\}$									x_2		$\overline{x}_2$	
$\{4,8\}$									$\overline{x}_1$	x_1		
$\{8,9\}$											x_2	$\overline{x}_2$
$\{6\}$	x_1											$\overline{x}_1$

Table 3

States	$\{1\}$	$\{10\}$	$\{2,3,4\}$	$\{2,3,7\}$	$\{3,4,5\}$	$\{2,3,9\}$	$\{3,5,7\}$	$\{3,5,9\}$	$\{7,8\}$	$\{4,8\}$	$\{8,9\}$	$\{6\}$
$\{1\}$	3/4	1/4										
$\{10\}$		2/4	2/4									
$\{2,3,4\}$				2/4	2/4							
$\{2,3,7\}$				1/4		1/4	1/4	1/4				
$\{3,4,5\}$					1/4		1/4		1/4	1/4		
$\{2,3,9\}$						2/4		2/4				
$\{3,5,7\}$								2/4	2/4			
$\{3,5,9\}$								2/4			2/4	
$\{7,8\}$									2/4		2/4	
$\{4,8\}$									2/4	2/4		
$\{8,9\}$											2/4	2/4
$\{6\}$	2/4											2/4

The following system of equations is obtained for the probabilities of global states $\{1\}$, $\{10\}$, $\{2,3,4\}$, $\{2,3,7\}$, $\{3,4,5\}$, $\{2,3,9\}$, $\{3,5,7\}$, $\{3,5,9\}$, $\{7,8\}$, $\{4,8\}$, $\{8,9\}$ and $\{6\}$ according to Table 3:

$$
\begin{aligned}
p(1) &= 3/4\,p(1) + 2/4\,p(6),\\
p(10) &= 2/4\,p(10) + 1/4\,p(1),\\
p(2.3.4) &= 2/4\,p(10),\\
p(2.3.7) &= 2/4\,p(2.3.4) + 1/4\,p(2.3.7),\\
p(3.4.5) &= 2/4\,p(2.3.4) + 1/4\,p(3.4.5),\\
p(2.3.9) &= 2/4\,p(2.3.9) + 1/4\,p(2.3.7),\\
p(3.5.7) &= 1/4\,p(3.4.5) + 1/4\,p(2.3.7),\\
p(3.5.9) &= 2/4\,p(2.3.9) + 1/4\,p(2.3.7) + 2/4\,p(3.5.7) + 2/4\,p(3.5.9),\\
p(7.8) &= 2/4\,p(7.8) + 1/4\,p(3.4.5) + 2/4\,p(3.5.7) + 2/4\,p(4.8),\\
p(4.8) &= 2/4\,p(4.8) + 1/4\,p(3.4.5),\\
p(8.9) &= 2/4\,p(7.8) + 2/4\,p(8.9) + 2/4\,p(3.5.9),\\
p(6) &= 2/4\,p(6) + 2/4\,p(8.9),\\
p(1) + p(10) + p(2.3.4) + p(2.3.7) + p(3.4.5) + p(2.3.9) + p(3.5.7) + p(3.5.9) +&\\
+ p(7.8) + p(4.8) + p(8.9) + p(6) &= 1.
\end{aligned}
$$

The solution of this system gives the probabilities

$$
\begin{aligned}
p(1) &= 12/46, & p(10) &= 6/46, & p(2.3.4) &= 3/46, & p(2.3.7) &= 2/46,\\
p(3.4.5) &= 2/46, & p(2.3.9) &= 1/46, & p(3.5.7) &= 1/46, & p(3.5.9) &= 3/46,\\
p(7.8) &= 3/46, & p(4.8) &= 1/46, & p(8.9) &= 6/46, & p(6) &= 6/46.
\end{aligned}
$$

The absolute probabilities of transitions between global states according to $p_{ij} = p(q_i)p'_{ij}$ are in Table 4.

Table 4

States	$\{1\}$	$\{10\}$	$\{2,3,4\}$	$\{2,3,7\}$	$\{3,4,5\}$	$\{2,3,9\}$	$\{3,5,7\}$	$\{3,5,9\}$	$\{7,8\}$	$\{4,8\}$	$\{8,9\}$	$\{6\}$
$\{1\}$	18/92	6/92										
$\{10\}$		6/92	6/92									
$\{2,3,4\}$				3/92	3/92							
$\{2,3,7\}$				1/92		1/92	1/92	1/92				
$\{3,4,5\}$					1/92		1/92		1/92	1/92		
$\{2,3,9\}$						1/92		1/92				
$\{3,5,7\}$								1/92	1/92			
$\{3,5,9\}$								3/92			3/92	
$\{7,8\}$									3/92		3/92	
$\{4,8\}$									1/92	1/92		
$\{8,9\}$											6/92	6/92
$\{6\}$	6/92											6/92

As it is said above, the probabilities of transitions between partial states are determined using the probabilities of transitions between global states. For example, the automaton in the considered example goes from partial state 2 to partial state 5 (transition τ_3) when it goes from global states $\{2,3,4\}$, $\{2,3,7\}$ and $\{2,3,9\}$ to global states $\{3,4,5\}$, $\{3,5,7\}$ and $\{3,5,9\}$. The sum of the probabilities of those incompatible events is 6/92. It is the probability of the transition from partial state 2 to partial state 5. The component automaton A_1 goes from state 2 to state 5 with the same probability.

6. Joint state assignment of component automata

As the weight of an edge in the graph G_i for the automaton A_i, $i = 1, 2, 3$, we take the numerator of fraction $1 - p^*_{st}$, where p^*_{st} is the probability of the transition in any direction between states q_s and q_t that correspond to the end of the edge. The adjacency matrices of weighted graphs G_1, G_2 and G_3 for automata A_1, A_2 and A_3, where the values below the main diagonal are omitted because of symmetry of the matrices, are the following:

$$A_1 = \begin{array}{c} \begin{array}{ccccc} 2 & 5 & 6 & 8 & 10 \end{array} \\ \left[\begin{array}{ccccc} 92 & 92 & 86 & 92 & 86 \\ & 86 & 92 & 92 & 86 \\ & & 92 & 86 & 92 \\ & & & 86 & 92 \\ & & & & 92 \end{array}\right] \end{array} \begin{array}{c} 1 \\ 2 \\ 5 \\ 6 \\ 8 \end{array}, \quad A_2 = \begin{array}{c} \begin{array}{cccc} 3 & 6 & 8 & 10 \end{array} \\ \left[\begin{array}{cccc} 92 & 86 & 92 & 86 \\ & 92 & 86 & 86 \\ & & 86 & 92 \\ & & & 92 \end{array}\right] \end{array} \begin{array}{c} 1 \\ 3 \\ 6 \\ 8 \end{array},$$

$$A_3 = \begin{array}{c} \begin{array}{ccccc} 4 & 6 & 7 & 9 & 10 \end{array} \\ \left[\begin{array}{ccccc} 92 & 86 & 92 & 92 & 86 \\ & 92 & 86 & 92 & 86 \\ & & 92 & 86 & 92 \\ & & & 86 & 92 \\ & & & & 92 \end{array}\right] \end{array} \begin{array}{c} 1 \\ 4 \\ 6 \\ 7 \\ 9 \end{array}.$$

The method described above is applied for the search for the maximum cuts in G_1, G_2 and G_3. The maximal values of d in G_1, G_2 and G_3 determine the result of the first step of the partition. They are $d_1(1) = 448$, $d_2(1) = 356$ and $d_3(1) = 448$. According to them, the following partitions are obtained: $\{\{1\}, \{2, 5, 6, 8, 10\}\}$, $\{\{1\}, \{3, 6, 8, 10\}\}$ and $\{\{1\}, \{4, 6, 7, 9, 10\}\}$. The next partitions are determined by the maximal values $h_1(6) = 276$, $h_2(6) = 184$ and $h_3(6) = 276$: $\{\{1, 6\}, \{2, 5, 8, 10\}\}$, $\{\{1, 6\}, \{3, 8, 10\}\}$ and $\{\{1, 6\}, \{4, 7, 9, 10\}\}$. Finally, according to $h_1(8) = 92$, $h_2(10) = 0$ and $h_3(9) = 92$, we obtain the cuts of the graphs in the form of partitions $\{\{1, 6, 8\}, \{2, 5, 10\}\}$, $\{\{1, 6\}, \{3, 8, 10\}\}$ and $\{\{1, 6, 9\}, \{4, 7, 10\}\}$.

The values of the coding variables z^1_i, z^2_i and z^3_i, which are introduced after having found the recurrent cuts, are determined taking into consideration the state compatibility. After i-th step in the process of state assignment, the variable z^j_i must take different values for the states of A_j corresponding to the vertices from different parts of the cut of G_j if these states are incompatible, and must not take different values if they are compatible. The state compatibility of component automata is defined as it is described in [15], in the following way. Let $f_j(q^j_l) = f_k(q^k_m) = q_s$ and $f_j(q^j_u) = f_k(q^k_v) = q_t$, where q^j_l and q^j_u are the states of the component automaton A_j, q^k_m and q^k_v are the states of A_k ($j \neq k$), q_s and q_t are partial states of the given parallel automaton B, f_j and f_k are mappings of the states of A_j and A_k into partial states of B. One of the pairs, (q^j_l, q^j_u) and (q^k_m, q^k_v), can be regarded as compatible, while the other is incompatible. The optimal choice of it requires special investigation is not considered in this paper.

The following matrices give the values of z^1_1, z^2_1 and z^3_1:

$$\begin{array}{cccccc|c} 1 & 2 & 5 & 6 & 8 & 10 & q^1 \\ \hline 0 & 1 & 1 & 0 & 0 & 1 & z^1_1 \end{array}, \quad \begin{array}{cccccc|c} 1 & 3 & 6 & 8 & 10 & & q^2 \\ \hline 0 & 1 & 0 & 1 & - & & z^2_1 \end{array}, \quad \begin{array}{cccccc|c} 1 & 4 & 6 & 7 & 9 & 10 & q^3 \\ \hline 0 & 1 & 0 & 1 & 0 & - & z^3_1 \end{array}.$$

Here, the states 1 and 10 of the automaton A_1 are incompatible, and the variable z^1_1 takes different values for them, while the values of z^2_1 and z^3_1 remain indefinite for the states of A_2 and A_3 mapped into the same partial state of B.

The following matrices are obtained after removing the edges corresponding to the pairs of states with different values of coding variables and to the pairs of compatible states:

$$A_1 = \begin{array}{c} \begin{matrix} 2 & 5 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{ccccc} 0 & 0 & 86 & 92 & 0 \\ & 86 & 0 & 0 & 86 \\ & & 0 & 0 & 92 \\ & & & 86 & 0 \\ & & & & 0 \end{array}\right] \end{array} \begin{matrix} 1 \\ 2 \\ 5 \\ 6 \\ 8 \end{matrix}, \quad A_2 = \begin{array}{c} \begin{matrix} 3 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{cccc} 0 & 86 & 0 & 0 \\ & 0 & 86 & 86 \\ & & 0 & 0 \\ & & & 0 \end{array}\right] \end{array} \begin{matrix} 1 \\ 3 \\ 6 \\ 8 \end{matrix}, \quad A_3 = \begin{array}{c} \begin{matrix} 4 & 6 & 7 & 9 & 10 \end{matrix} \\ \left[\begin{array}{ccccc} 0 & 86 & 0 & 92 & 0 \\ & 0 & 86 & 0 & 86 \\ & & 0 & 86 & 0 \\ & & & 0 & 92 \\ & & & & 92 \end{array}\right] \end{array} \begin{matrix} 1 \\ 4 \\ 6 \\ 7 \\ 9 \end{matrix}.$$

The partitions $\{\{1,5\},\{2,6,8,10\}\}$, $\{\{1,3\},\{6,8,10\}\}$, $\{\{7,9,10\},\{1,4,6\}\}$ and, relatively, the values of z_2^1, z_2^2 and z_2^3 are obtained by the same way:

$$\begin{array}{c} \begin{matrix} 1 & 2 & 5 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{cccccc} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & - & 0 \end{array}\right] \end{array} \begin{matrix} q^1 \\ z_1^1 \\ z_2^1 \end{matrix}, \quad \begin{array}{c} \begin{matrix} 1 & 3 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{ccccc} 0 & 1 & 0 & 1 & - \\ - & 0 & - & 1 & 1 \end{array}\right] \end{array} \begin{matrix} q^2 \\ z_1^2 \\ z_2^2 \end{matrix}, \quad \begin{array}{c} \begin{matrix} 1 & 4 & 6 & 7 & 9 & 10 \end{matrix} \\ \left[\begin{array}{cccccc} 0 & 1 & 0 & 1 & 0 & - \\ 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right] \end{array} \begin{matrix} q^3 \\ z_1^3 \\ z_2^3 \end{matrix}.$$

The values of z_2^1 are given by vector $(0\,0\,1\,1-0)$ according to the partition $\{\{1,5\}, \{2,6,8,10\}\}$ and taking into consideration the state compatibility. Vector $(1\,0\,1\,0-0)$ could give the values of z_2^1 as well, but then, the probability of the transitions between states with changing values of z_2^1 is 6/92, while there are no such transitions at the first variant. Therefore, the first variant gives less switching activity.

After removing edges, the graph G_2 has become empty and state assignment of A_2 has fulfilled. The matrices for A_1 and A_3 are

$$A_1 = \begin{array}{c} \begin{matrix} 2 & 5 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 86 \\ & & 0 & 0 & 0 \\ & & & 0 & 0 \\ & & & & 0 \end{array}\right] \end{array} \begin{matrix} 1 \\ 2 \\ 5 \\ 6 \\ 8 \end{matrix}, \quad A_3 = \begin{array}{c} \begin{matrix} 4 & 6 & 7 & 9 & 10 \end{matrix} \\ \left[\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 \\ & & 0 & 0 & 0 \\ & & & 0 & 92 \\ & & & & 92 \end{array}\right] \end{array} \begin{matrix} 1 \\ 4 \\ 6 \\ 7 \\ 9 \end{matrix}.$$

They make us to introduce variables z_3^1 and z_3^3. The following matrices represent the result of state assignment of the component automata:

$$\begin{array}{c} \begin{matrix} 1 & 2 & 5 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{cccccc} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & - & 0 \\ - & 0 & - & - & - & 1 \end{array}\right] \end{array} \begin{matrix} q^1 \\ z_1^1 \\ z_2^1 \\ z_3^1 \end{matrix}, \quad \begin{array}{c} \begin{matrix} 1 & 3 & 6 & 8 & 10 \end{matrix} \\ \left[\begin{array}{ccccc} 0 & 1 & 0 & 1 & - \\ - & 0 & - & 1 & 1 \end{array}\right] \end{array} \begin{matrix} q^2 \\ z_1^2 \\ z_2^2 \end{matrix}, \quad \begin{array}{c} \begin{matrix} 1 & 4 & 6 & 7 & 9 & 10 \end{matrix} \\ \left[\begin{array}{cccccc} 0 & 1 & 0 & 1 & 0 & - \\ 0 & 0 & 0 & 1 & 1 & 1 \\ - & - & - & - & 0 & 1 \end{array}\right] \end{array} \begin{matrix} q^3 \\ z_1^3 \\ z_2^3 \\ z_3^3 \end{matrix}.$$

Substituting the vectors of values of the components z_i^j to the symbols of states in Table 1, we obtain the interval representation of the system of incompletely specified Boolean functions $z_i^{j'}$ of excitation of memory elements. It is convenient to represent the system in Table 5.

Table 5

x_1x_2	$z_1^1z_2^1z_3^1$	$z_1^2z_2^2$	$z_1^3z_2^3z_3^3$	$z_1^{1'}z_2^{1'}z_3^{1'}$	$z_1^{2'}z_2^{2'}$	$z_1^{3'}z_2^{3'}z_3^{3'}$
0 1	0 0 –	0 –	0 0 –	1 0 1	– 1	0 1 1
– 0	0 0 –	0 –	0 0 –	0 0 –	0 –	0 0 –
1 –	0 0 –	0 –	0 0 –	0 0 –	0 –	0 0 –
– 1	1 0 1	– 1	0 1 1	1 0 1	– 1	0 1 1
– 0	1 0 1	– 1	0 1 1	1 0 0	1 0	1 0 –
1 –	1 0 0	1 0	1 0 –	1 1 –	1 0	1 0 –
0 –	1 0 0	1 0	1 0 –	1 0 0	1 0	1 1 –
0 0	1 0 0	1 0	1 1 –	1 0 0	1 0	0 1 0
0 1	1 0 0	1 0	1 1 –	1 0 0	1 0	1 1 –
1 1	1 0 0	1 0	1 1 –	1 1 –	1 0	1 1 –
1 0	1 0 0	1 0	1 1 –	1 1 –	1 0	0 1 0
1 0	1 1 –	1 0	1 0 –	1 1 –	1 0	1 0 –
0 0	1 1 –	1 0	1 0 –	1 1 –	1 0	1 1 –
0 1	1 1 –	1 0	1 0 –	0 – –	1 1	1 1 –
1 1	1 1 –	1 0	1 0 –	0 – –	1 1	1 0 –
0 –	1 0 0	1 0	0 1 0	1 0 0	1 0	0 1 0
1 –	1 0 0	1 0	0 1 0	1 1 –	1 0	0 1 0
– 0	1 1 –	1 0	1 1 –	1 1 –	1 0	0 1 0
– 1	1 1 –	1 0	1 1 –	0 – –	1 1	1 1 –
– 0	1 1 –	1 0	0 1 0	1 1 –	1 0	0 1 0
– 1	1 1 –	1 0	0 1 0	0 – –	1 1	0 1 0
– 1	0 – –	1 1	1 1 –	0 – –	1 1	1 1 –
– 0	0 – –	1 1	1 1 –	0 – –	1 1	0 1 0
1 –	0 – –	1 1	1 0 –	0 – –	1 1	1 0 –
0 –	0 – –	1 1	1 0 –	0 – –	1 1	1 1 –
– 1	0 – –	1 1	0 1 0	0 – –	1 1	0 1 0
– 0	0 – –	1 1	0 1 0	0 1 –	0 –	0 0 –
0 –	0 1 –	0 –	0 0 –	0 1 –	0 –	0 0 –
1 –	0 1 –	0 –	0 0 –	0 0 –	0 –	0 0 –

7. Reduction of interconnection

The component sequential automata in the net implementing the given parallel automaton exchange with binary signals as the Boolean variables z_i^j. The problem of reducing these connections makes sense when the reduction of the number of input pins of a logic circuit is necessary. Solving this problem, as it is described in [16], by decreasing the number of arguments of Boolean excitation functions of memory elements for each component separately is suggested.

Let a pair of matrices $(\mathbf{X}, \mathbf{Y})$ be an interval representation of a system of incompletely specified Boolean functions, where the rows of $\mathbf{X}$ represent the intervals of Boolean space of the arguments, and the rows of $\mathbf{Y}$ the values of the functions at the corresponding intervals. The condition of correctness requires the row orthogonality of $\mathbf{X}$ and the orthogonality of the corresponding rows of $\mathbf{Y}$. The *distinction matrix* of the rows of $\mathbf{X}$ is constructed. For each pair of rows of $\mathbf{X}$ corresponding to orthogonal rows of $\mathbf{Y}$, the distinction matrix has a row obtained by modulo 2 addition of those rows of $\mathbf{X}$. The modulo 2 addition of don't care "–" with zero or one gives zero. The shortest column cover must be found in the distinction matrix, that is the least set of columns such that each row of the matrix has one in these columns. The obtained cover shows the set of essential arguments, and when selecting the columns to form the cover, we prefer those that correspond to the variables coding the states of the automaton under consideration. Thus, we decrease interconnection of components.

For the system of excitation functions in automata A_1, A_2 and A_3, the distinction matrices are as follows after applying the reduction rule in solving the cover problem (if row i has 1s everywhere, where row j has 1s, row i can be removed [17]):

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & z_1^1 & z_2^1 & z_3^1 & z_1^2 & z_2^2 & z_1^3 & z_2^3 & z_3^3 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}, \quad \begin{array}{c} \begin{matrix} x_2 & z_1^1 & z_2^1 & z_3^1 & z_1^2 & z_2^2 & z_1^3 & z_2^3 & z_3^3 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array},$$

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & z_1^1 & z_2^1 & z_3^1 & z_2^2 & z_1^3 & z_2^3 & z_3^3 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

The process of forming the desired covers begins by introducing into them the columns z_1^1, z_2^1, z_3^1 from the first matrix, columns z_1^2, z_2^2 from the second matrix and z_1^3, z_2^3, z_3^3 from the third matrix. Relatively, the column covers $\{x_1, x_2, z_1^1, z_2^1, z_3^1, z_1^2\}$, $\{x_2, z_1^1, z_2^1, z_1^2, z_2^2, z_1^3\}$ and $\{x_1, x_2, z_1^1, z_2^1, z_1^3, z_2^3, z_3^3\}$ are obtained.

Figure 1 shows the net implementing the given parallel automaton. The following matrices represent the systems of disjunctive normal forms of the excitation functions in the component automata that are obtained separately for each automaton by the minimization method described in [18]:

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & z_1^1 & z_2^1 & z_3^1 & z_1^2 \end{matrix} \\ \begin{bmatrix} 0 & - & 0 & 1 & - & - \\ 0 & 1 & - & 0 & - & 0 \\ - & 0 & 0 & - & - & 1 \\ - & 1 & - & - & 1 & - \\ - & - & 1 & 0 & - & - \\ 1 & - & - & - & 0 & 1 \\ - & 0 & 1 & 1 & - & - \end{bmatrix} \end{array}, \quad \begin{array}{c} \begin{matrix} z_1^{1'} & z_2^{1'} & z_3^{1'} \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \end{array}; \quad \begin{array}{c} \begin{matrix} x_2 & z_1^1 & z_2^1 & z_1^2 & z_2^2 & z_1^3 \end{matrix} \\ \begin{bmatrix} - & 1 & - & - & - & - \\ 1 & - & - & - & 1 & - \\ - & 0 & - & - & - & - \\ 1 & - & 1 & - & - & - \\ - & - & - & - & - & 1 \\ 1 & - & - & 1 & - & - \end{bmatrix} \end{array}, \quad \begin{array}{c} \begin{matrix} z_1^{2'} & z_2^{2'} \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \end{array};$$

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & z_1^1 & z_2^1 & z_1^3 & z_2^3 & z_3^3 \end{matrix} \\ \begin{bmatrix} 0 & - & - & 0 & - & 0 & - \\ - & 1 & 1 & - & - & 1 & 1 \\ - & - & - & - & 1 & 1 & - \\ - & 0 & 1 & - & - & - & 1 \\ 0 & - & - & - & 1 & 0 & - \\ - & 1 & - & - & - & 1 & - \\ - & 0 & 1 & - & - & 1 & 0 \\ - & 1 & - & - & 1 & - & - \\ - & - & - & - & 1 & 0 & - \end{bmatrix} \end{array}, \quad \begin{array}{c} \begin{matrix} z_1^{1'} & z_2^{1'} & z_3^{1'} \end{matrix} \\ \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \end{array}.$$

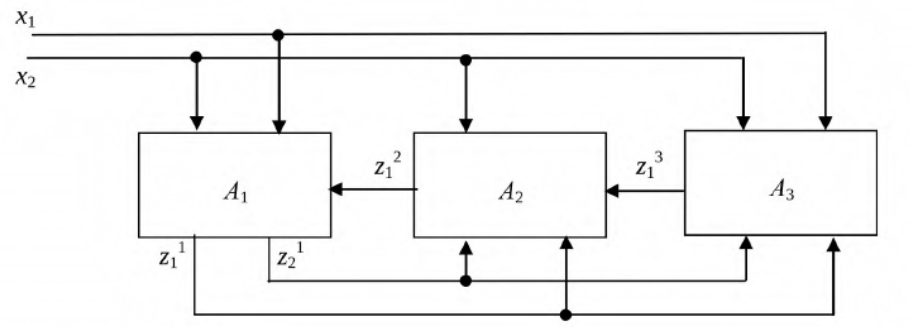

Fig. 1. Net of sequential automata

8. Asynchronous implementation

The probabilities of transitions between partial states of a parallel automaton were determined above with the help of modelling its behavior by a sequential automaton with the states corresponding to the global states of the parallel automaton. Not any parallel automaton admits such modelling under asynchronous implementation [1]. Here, we restrict ourselves to consideration of parallel automata that admit modelling by sequential automaton. The parallel automaton in the Example 1 is an example of such an automaton. We consider only transitions between stable states (partial and global) and regard the probability of unstable state to be negligibly small. The stable global states of the automaton in the Example 1 are $\{1\}$, $\{10\}$, $\{3,4,5\}$, $\{2,3,9\}$, $\{3,5,9\}$, $\{7,8\}$, $\{4,8\}$, $\{8,9\}$ and $\{6\}$.

The component sequential automata are determined by the same way that was applied above. The given parallel automaton is decomposed into asynchronous automata A_1, A_2 and A_3 with the sets of states $Q_1 = \{1,2,5,6,8,10\}$, $Q_2 = \{1,3,6,8,10\}$ and $Q_3 = \{1,4,6,7,9,10\}$. Table 6 shows transitions between states of the component automata, where the stable states are marked with bold.

Table 6

q^1	q^2	q^3	x_1x_2											
			00			01			11			10		
			$q^{1'}$	$q^{2'}$	$q^{3'}$	$q^{1'}$	$q^{2'}$	$q^{3'}$	$q^{1'}$	$q^{2'}$	$q^{3'}$	$q^{1'}$	$q^{2'}$	$q^{3'}$
1	1	1	**1**	**1**	**1**	10	10	10	**1**	**1**	**1**	**1**	**1**	**1**
10	10	10	2	3	9	**10**	**10**	**10**	**10**	**10**	**10**	5	3	4
5	3	4	5	3	9	8	8	7	8	8	4	**5**	**3**	**4**
2	3	9	**2**	**3**	**9**	**2**	**3**	**9**	8	8	9	5	3	9
5	3	9	**5**	**3**	**9**	8	8	9	8	8	9	**5**	**3**	**9**
8	8	7	6	6	6	**8**	**8**	**7**	**8**	**8**	**7**	1	1	1
8	8	4	6	6	6	8	8	7	**8**	**8**	**4**	**8**	**8**	**4**
8	8	9	6	6	6	**8**	**8**	**9**	**8**	**8**	**9**	1	1	1
6	6	6	**6**	**6**	**6**	**6**	**6**	**6**	1	1	1	1	1	1

For low power race-free state assignment of component automata, the approach described in detail in [19] is applied. The pairs of transitions between states at the same input signal are considered. For example, Table 6 shows that when $x_1 = 0$, $x_2 = 1$, the automata A_1, A_2 and A_3 have the pairs of transitions, $(1 \to 10, 5 \to 8)$, $(1 \to 10, 3 \to 8)$ and $(1 \to 10, 4 \to 7)$, relatively. The condition for absence of critical races in a pair of transitions is formulated by a ternary vector whose components correspond to the states of the automaton and have values 1 or 0 depending on what the transition of the pair the corresponding states belong to. For the pairs named above, (0−1−10), (01−10) and

(01−1−0) are such vectors where 0s and 1s can change places. Only one of such conditions is sufficient to fulfill. We call it obligatory. The optimal choice of obligatory conditions requires a special research.

All the obligatory conditions in the form of the vectors above constitute the condition matrix that has not implied rows [6]. A ternary vector **a** implies a ternary vector **b** if **b** is obtained from **a** by replacing some 0s or 1s by the value "−" and, perhaps, by inverting the obtained result. For example, vector (10−101) implies (10−−01) and (01−−1−). The sense of this relation is that the condition represented by **b** is satisfied if the condition represented by **a** is satisfied. For component automata of the considered net, the condition matrices can be as follows:

$$\begin{array}{cccccc} 1 & 2 & 5 & 6 & 8 & 10 \\ \left[\begin{array}{cccccc} - & 0 & 1 & - & - & 0 \\ 0 & - & - & 1 & 1 & - \\ 0 & 1 & 1 & - & 0 & - \\ 0 & - & 1 & - & 1 & 0 \\ - & 0 & - & - & - & 1 \end{array}\right] & \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \end{array},\quad \begin{array}{ccccc} 1 & 3 & 6 & 8 & 10 \\ \left[\begin{array}{ccccc} - & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & - & 1 \\ 0 & 1 & 0 & 1 & - \\ 0 & - & 1 & - & 0 \end{array}\right] & \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \end{array},\quad \begin{array}{cccccc} 1 & 4 & 6 & 7 & 9 & 10 \\ \left[\begin{array}{cccccc} 0 & 1 & - & 0 & - & 1 \\ 0 & 1 & - & - & 0 & 1 \\ - & 0 & - & 0 & 1 & - \\ - & - & 0 & 0 & 1 & - \\ 0 & - & 0 & - & - & 1 \\ 0 & 1 & - & 0 & - & 1 \\ - & 1 & - & - & - & 0 \end{array}\right] & \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \end{array}.$$

A ternary matrix **R** implies a matrix **S** if for every row of **S** there is a row in **R** that implies it. The problem of race-free state assignment is reduced to finding a matrix with the minimal number of rows that implies the condition matrix and is called *shortest implying form* of the condition matrix. The rows of this matrix represent the desired codes of the states.

The shortest implying form of a ternary matrix is found in the following way. A set of rows of a matrix is called *compatible* if there is a vector implying each row of this set. A compatible set is the *maximal* one if it is not a proper subset of any other compatible set. We should find all the maximal compatible sets of the rows of the condition matrix and then obtain the shortest cover of the rows by these sets. Every compatible set correspond to the vector implying all the rows belonging to this set. The vectors corresponding to the elements of the obtained cover constitute the shortest implying form of the condition matrix under consideration. Below, the matrices are given whose rows imply the elements of the maximal compatible sets from the obtained shortest cover; the numbers of the rows of the condition matrix implied by each row of the given matrices are to their right:

$$\begin{array}{cccccc} 1 & 2 & 5 & 6 & 8 & 10 \\ \left[\begin{array}{cccccc} 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & - & 0 & 0 \end{array}\right] & \begin{array}{c} 1,2,4 \\ 3,5 \end{array} \end{array},\quad \begin{array}{ccccc} 1 & 3 & 6 & 8 & 10 \\ \left[\begin{array}{ccccc} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{array}\right] & \begin{array}{c} 1,2 \\ 1,4 \\ 2,3 \end{array} \end{array},\quad \begin{array}{cccccc} 1 & 4 & 6 & 7 & 9 & 10 \\ \left[\begin{array}{cccccc} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right] & \begin{array}{c} 1,2,5,6 \\ 1,4,5,6 \\ 2,3,4 \\ 3,4,5,7 \end{array} \end{array}.$$

We determine the relation for each variable z_i and the set of transitions between states with codes, where z_i changes its value at those transitions. That is the i-th memory element in the real circuit implementing the given automaton changes its state. The switching activity is connected with the probabilities of transitions between states. The probability of transition when z_i changes its value is put into the correspondence to z_i. This probability is equal to the sum of the probabilities of all the transitions when z_i changes its value, because those transitions are incompatible events.

The probabilities of the transitions (independently of the direction) between the partial states of the given parallel automaton coinciding with the probabilities of the transitions in component automata are obtained by the method described above. Table 7 shows them,

where the rows and columns correspond to the states and empty entries mean that there are no corresponding transitions.

Table 7

States	2	3	4	5	6	7	8	9	10
1					13/282	3/282	8/282	8/282	24/282
2				6/282			6/282		12/282
3						4/282	24/282		24/282
4						6/282		4/282	12/282
5							18/282		12/282
6						3/282	13/282	8/282	
7									
8									
9									12/282

Every compatible set of rows of the condition matrix and, correspondingly, vector implying all the rows in the set have the weight as the value proportional to the sum of the probabilities of the transitions connected with this vector. The minimal weight cover of the row sets of the condition matrix with the maximal compatible sets is obtained. The weight of a cover is the sum of the weights of its elements. The problem of minimal weighted cover is investigated in detail in [20].

All the family of maximal compatible sets for A_1 coincides with its shortest cover. According to the matrices above, the sets $\{1,4\}$ and $\{2,3\}$ constitute the shortest cover with the minimal weight for A_2, and $\{1,2,5,6\}$ and $\{3,4,5,7\}$ for A_3. The following matrices give the codes of the states of A_1, A_2 and A_3, respectively:

$$\begin{array}{cccccc} 1 & 2 & 5 & 6 & 8 & 10 \\ \left[\begin{array}{cccccc} 0 & 0 & 1 & 1 & 1 & - \\ 0 & 1 & 1 & - & 0 & 0 \end{array}\right] & \begin{array}{c} q^1 \\ z_1^1 \\ z_2^1 \end{array} \end{array}, \quad \begin{array}{ccccc} 1 & 3 & 6 & 8 & 10 \\ \left[\begin{array}{ccccc} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{array}\right] & \begin{array}{c} q^2 \\ z_1^2 \\ z_2^2 \end{array} \end{array}, \quad \begin{array}{cccccc} 1 & 4 & 6 & 7 & 9 & 10 \\ \left[\begin{array}{cccccc} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right] & \begin{array}{c} q^3 \\ z_1^3 \\ z_2^3 \end{array} \end{array}.$$

Table 8 is the interval specification of the system of incompletely specified Boolean functions $z_i^{j'}$ of excitation of memory elements. In this specification, the intervals of the space of internal variables that are determined by the transitions between states of the component automata are used [6].

After decreasing the number of arguments and minimization of the system of the excitation functions, the following matrices representing the systems of disjunctive normal forms are obtained:

$$\begin{array}{ccccccc} x_1 & x_2 & z_1^1 & z_2^1 & z_1^2 & z_2^2 & z_1^3 \\ \left[\begin{array}{ccccccc} - & 1 & 1 & - & - & 1 & - \\ 0 & - & 0 & 1 & - & - & - \\ 1 & 1 & - & - & - & 1 & 0 \\ - & - & 1 & 0 & - & - & 1 \\ 0 & - & 1 & - & 1 & - & - \\ 0 & 0 & - & - & - & 1 & - \\ 1 & 0 & - & - & 0 & - & 1 \\ - & 0 & 1 & 1 & - & 1 & - \\ 1 & 0 & - & 1 & - & 1 & - \end{array}\right] \end{array}, \quad \begin{array}{cc} z_1^{1'} & z_2^{1'} \\ \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{array}\right] \end{array}; \quad \begin{array}{ccccccc} x_1 & x_2 & z_1^1 & z_2^1 & z_1^2 & z_2^2 & z_1^3 \\ \left[\begin{array}{ccccccc} 0 & - & - & - & 1 & - & - \\ 0 & - & - & - & 0 & 1 & - \\ 1 & - & - & - & - & - & 1 \\ 0 & 1 & 0 & - & - & - & - \\ - & 1 & 1 & - & - & 1 & - \\ 1 & 1 & - & - & - & 1 & 0 \\ - & - & - & - & 1 & - & 1 \\ 1 & - & 0 & 1 & - & 1 & - \\ - & 0 & - & 1 & - & 1 & - \end{array}\right] \end{array}, \quad \begin{array}{cc} z_1^{2'} & z_2^{2'} \\ \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{array}\right] \end{array};$$

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & z_1^1 & z_2^1 & z_1^2 & z_1^3 & z_2^3 \end{matrix} \\ \begin{bmatrix} 1 & - & - & - & - & 1 & - \\ - & 1 & 0 & - & - & 1 & - \\ 0 & 1 & 0 & 0 & - & - & - \\ - & 0 & - & 1 & 0 & 0 & - \\ - & 1 & 1 & - & - & - & 1 \\ 1 & 1 & - & - & - & - & 1 \\ 0 & - & - & - & - & - & 1 \\ 0 & 0 & - & 1 & 0 & - & - \end{bmatrix} \end{array}, \quad \begin{array}{c} \begin{matrix} z_1^{3'} & z_2^{3'} \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array}.$$

Table 8

x_1x_2	$z_1^1z_2^1$	$z_1^2z_2^2$	$z_1^3z_2^3$	$z_1^{1'}z_2^{1'}$	$z_1^{2'}z_2^{2'}$	$z_1^{3'}z_2^{3'}$
− 0	0 0	0 0	0 0	0 0	0 0	0 0
1 −	0 0	0 0	0 0	0 0	0 0	0 0
0 1	0 0	0 −	0 −	0 0	0 1	1 1
− 1	0 0	0 1	1 1	0 0	0 1	1 1
0 0	0 −	0 1	− 1	0 1	0 1	0 1
1 0	− −	0 1	1 −	1 1	0 1	1 0
1 0	1 1	0 1	1 0	1 1	0 1	1 0
0 0	1 1	0 1	− −	1 1	0 1	0 1
0 1	1 −	− 1	− 0	1 0	1 1	0 0
1 1	1 −	− 1	1 0	1 0	1 1	1 0
0 −	0 1	0 1	0 1	0 1	0 1	0 1
1 1	− −	− 1	0 1	1 0	1 1	0 1
1 0	− 1	0 1	0 1	1 1	0 1	0 1
− 0	1 1	0 1	0 1	1 1	0 1	0 1
− 1	1 −	− 1	0 1	1 0	1 1	0 1
− 1	1 0	1 1	0 0	1 0	1 1	0 0
0 0	1 0	1 −	0 0	1 −	1 0	0 0
1 0	− 0	− −	0 0	0 0	0 0	0 0
1 −	1 0	1 1	1 0	1 0	1 1	1 0
0 1	1 0	1 1	− 0	1 0	1 1	0 0
0 0	1 0	1 −	− 0	1 −	1 0	0 0
− 1	1 0	1 1	0 1	1 0	1 1	0 1
1 0	− 0	− −	0 −	0 0	0 0	0 0
0 0	1 0	1 −	0 −	1 −	1 0	0 0
0 −	1 −	1 0	0 0	1 −	1 0	0 0
1 −	− −	1 0	0 0	0 0	− 0	0 0

Figure 2 shows the net of asynchronous sequential automata implementing the given parallel automaton.

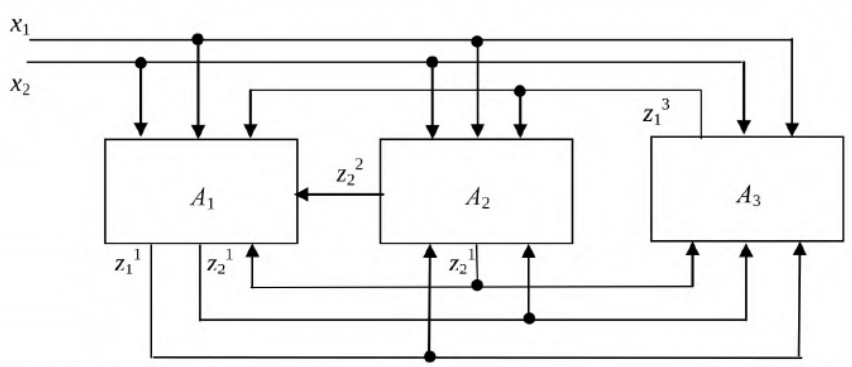

Fig. 2. Net of asynchronous sequential automata

9. Conclusion

The suggested approach to decomposition of a parallel automaton can be applied in the synthesis of distributed control systems. In such a system controlling a set of objects distant from each other, all the blocks are connected in a network, with each block located at one of the controlled objects. So the problem of minimization of interconnection considered in the paper is interesting from the point of view of reliability. Using the decomposition of a parallel automaton allows decreasing the dimension of laborious problems of logical design. The suggested approach is intended for using in a computer aided logical design system.

REFERENCES

1. *Zakrevskiy A. D.* Parallel'nye Algoritmy Logicheskogo Upravleniya [Parallel Algorithms for Logical Control]. Moscow, URSS, 2003, 304 p. (in Russian)
2. *Peterson J. L.* Petri Net Theory and the Modeling of Systems. Englewood Cliffs, N.J., Prentice-Hall Inc., 1981.
3. *Piguet C.* Low-power and low-voltage CMOS digital design. Microelectronic Eng., 1997, vol. 39, no. 1–4, pp. 179–208.
4. *Muroga S.* VLSI System Design. When and How to Design Very-Large-Scale Integrated Circuits. N.Y., John Wiley & Sons, 1982.
5. *Pedram M.* Power minimization in IC design: Principles and applications. ACM Trans. Design Automat. Electron. Syst., 1996, vol. 6, pp. 3–56.
6. *Zakrevskij A., Pottosin Yu., and Cheremisinova L.* Design of Logical Control Devices. Tallinn, TUT Press, 2009.
7. *Pottosin Yu. V.* Sovmestnoe energosberegayushchee kodirovanie sostoyaniy posledovatel'nykh avtomatov seti, realizuyushchey parallel'nyy avtomat [Joint low power state assignment of sequential automata of a net implementing a parallel automaton]. Informatika, 2023, vol. 20, no. 1, pp. 75–90. (in Russian)
8. *Pottosin Yu. V.* Dekompozitsiya parallel'nogo avtomata v set' posledovatel'nykh avtomatov i energosberegayushchee kodirovanie ikh sostoyaniy pri asinkhronnoy realizatsii [Decomposition of a parallel automaton into a net of sequential automata and low power state assignment of them at asynchronous implementation]. Informatika, 2024, vol. 21, no. 3, pp. 7–22. (in Russian)
9. *Hartmanis J. and Stearns R. E.* Algebraic Structure Theory of Sequential Machines. Englewood Cliffs, N.J., Prentice Hall Inc., 1966.
10. *Keevallik A. E.* Teorema dekompozitsii konechnykh avtomatov [A theorem of decomposition of finite automata]. Avtomatika i Vychislitel'naya Tekhnika, 1974, no. 1, pp. 17–24. (in Russian)
11. *Pottosin Yu. V.* Kombinatornye Zadachi v Logicheskom Proektirovanii Diskretnykh Ustroystv [Combinatorial Problems in Logical Design of Discrete Devices]. Minsk, Belaruskaja Navuka, 2021, 175 p. (in Russian)
12. *Zakrevskiy A. D.* Raskraska grafov pri dekompozitsii bulevykh funktsiy [Coloring graphs in decomposition of Boolean functions]. Logicheskoe Proektirovanie, 2000, iss. 5, pp. 83–97. (in Russian)
13. *Macii E., Pedram M., and Somenzi F.* High-level power modeling, estimation and optimization. IEEE Trans. CADICS, 1998, vol. 17, no. 11, pp. 1061–1079.
14. *Pottosin Yu. V.* Low power assignment of partial states of a parallel automaton. Prikladnaya Diskretnaya Matematika, 2022, no. 56, pp. 113–122.
15. *Pottosin Yu. V. and Shestakov E. A.* Dekompozitsiya avtomata v dvukhkomponentnuyu set' s ogranicheniem na vnutrennie svyazi [Decomposition of an automaton into a two-component net with restrictions on internal communication]. Avtomatika i Vychislitel'naya Tekhnika, 1982, no. 6, pp. 25–32. (in Russian)

16. *Zakrevskij A., Pottosin Yu., and Cheremisinova L.* Optimization in Boolean Space. Tallinn, TUT Press, 2009.
17. *Zakrevskij A., Pottosin Yu., and Cheremisinova L.* Combinatorial Algorithms of Discrete Mathematics. Tallinn, TUT Press, 2008.
18. *Pottosin Yu. V., Toropov N. R., and Shestakov E. A.* Metod minimizatsii sistemy ne polnost'yu opredelennykh bulevykh funktsiy [A method for minimization of a system of incompletely specified Boolean functions]. Informatika, 2009, no. 3(23), pp. 16–26. (in Russian)
19. *Pottosin Yu.* Race-free state assignment for low power asynchronous automaton. Further Improvements in the Boolean Domain. Ed. B. Steinbach. Cambridge Scholars Publ., 2018, pp. 253–267.
20. *Zakrevskiy A. D.* Optimizatsiya pokrytiy mnozhestv [Optimization of set covering]. Logicheskiy Yazyk dlya Predstavleniya Algoritmov Sinteza Releynykh Ustroystv [Logical Language for Representation of Algorithms for Relay Devices Synthesis]. Ed. M. A. Gavrilov. Moscow, Nauka, 1966, pp. 136–148. (in Russian)