Trivial Logic Arrays

J. Bokr, V. Jáneš

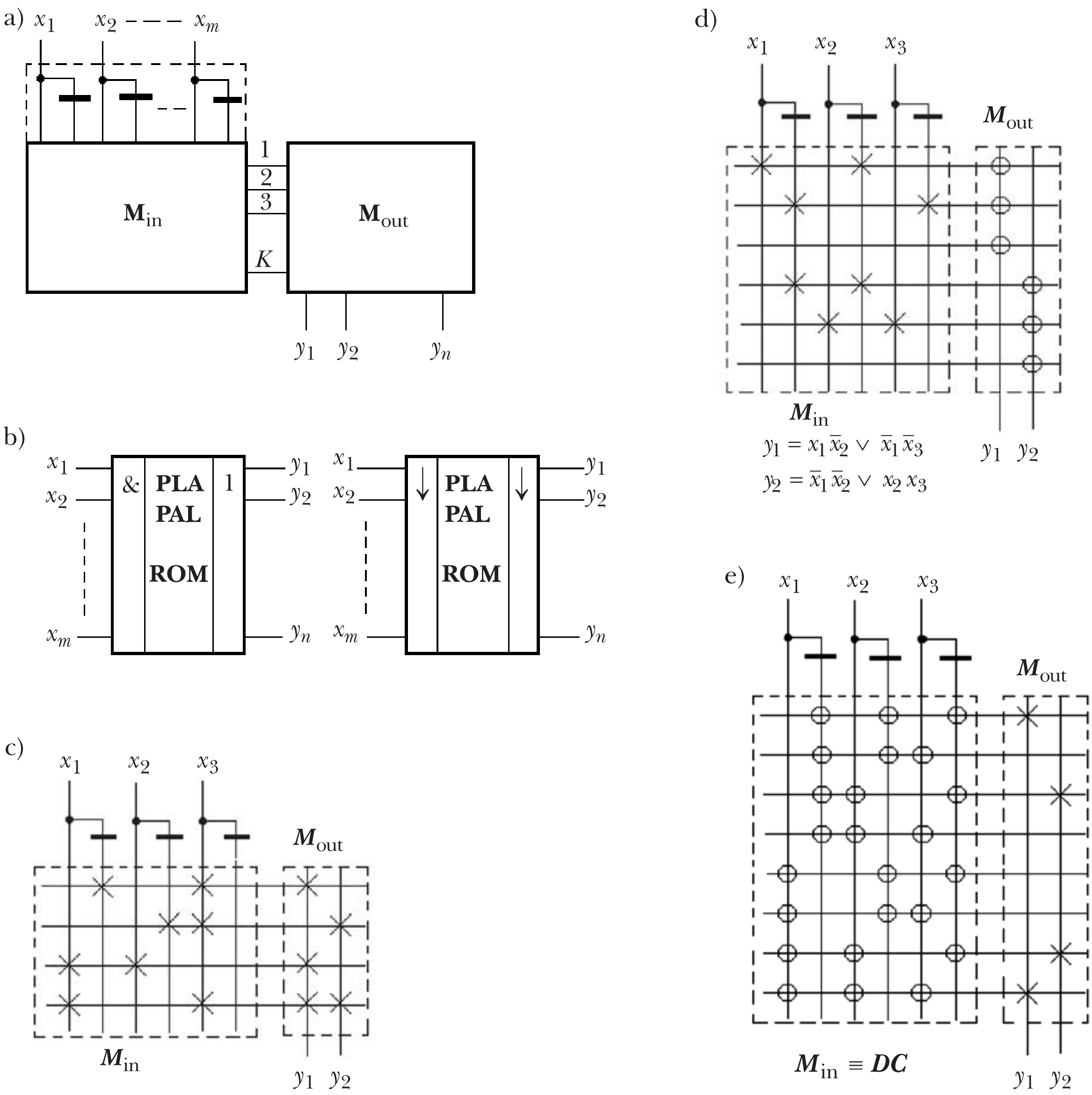
This paper deals with matrix modelling of trivial logic arrays (PLA, PAL, ROM) and the design of the above array as structural models of static and dynamic logic objects.

Keywords: cartesian product of matrices, PLA, PAL, ROM, canonical decomposition, states coding, states coding of Miller or Liu, substitute input variable.

1 Introduction

The trivial logic arrays dealt with here [1,2] (Fig. 1.) are:

- PLA (programmable logic array) with programmable input and output matrices,
- PAL (programmable array logic) with a programmable input matrix and a given output matrix,
- ROM (read only memory) a given input matrix (is given through an address decoder) and a programmable output matrix.



c) $y_1 = \bar{x}_1 x_3 \vee x_1 x_2 \vee x_1 x_3$, $y_2 = \bar{x}_2 x_3 \vee x_1 x_3$

d) $y_1 = x_1 \bar{x}_2 \vee \bar{x}_1 \bar{x}_3$, $y_2 = \bar{x}_1 \bar{x}_2 \vee x_2 x_3$

e) $y_1 = \bar{x}_1 \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3$, $y_2 = \bar{x}_1 x_2 \bar{x}_3 \vee x_1 x_2 \bar{x}_3$

Fig. 1: Trivial logic arrays: a) block diagram, b) labelling; examples of logic arrays: c) PLA, d) PAL, e) ROM, where × and ○ mean the programmable and given positions, respectively

The input matrix has either conjunctors (& - AND) or inverse disjunctors {↓ – NAND). The output matrix has either disjunctors {1 – OR) or inverse disjunctors (NOR).

The paper deals with matrix analysis and synthesis of trivial logic arrays.

2 Cartesian product of Boolean matrices

The ***Cartesian product*** $M\,(OP)(op)\,\mathfrak{M}$ of Boolean matrices

$M:\{1,2,\ldots,q\}\times\{1,2,\ldots,h\}\to\{0,1\}:\langle i,j\rangle\mapsto m_{ij}$

$\mathfrak{M}:\{1,2,\ldots,h\}\times\{1,2,\ldots,p\}\to\{0,1\}:\langle j,k\rangle\mapsto \boldsymbol{m}_{jk}$

denotes a Boolean matrix

$M(OP)(op)\,\mathfrak{M}:\{1,2,\ldots,q\}\times\{1,2,\ldots,p\}\to\{0,1\}:\langle i,k\rangle\mapsto \mu_{ik}$

where $\mu_{ik}=\binom{OP}{j} m_{ij}(op)\boldsymbol{m}_{jk}$ and (OP) and (op) are Boolean operators.

Example 1: Let a system $\{y_1,y_2\}$ of Boolean functions y_1,y_2 be given

$\left(\{y_1,y_2\}\subset\{0,1\}^{\{0,1\}^4}\right)$:

x_1	x_2	x_3	x_4	y_1	y_2
1	0	0	1	0	1
0	1	0	0	1	1
1	0	0	1	1	0
0	0	0	0	0	0
0	1	1	1	1	0

Write the system $\{y_1,y_2\}$ as matrices $\{cndf\,(y_1),\,cndf\,(y_2)\}$, where *cndf* denotes a canonical normal disjunctive formula. Note that $(x\equiv 0)=\bar{x}$ and $(x\equiv 1)=x$:

$$[y_1,y_2]=[x_1,x_2,x_3,x_4]\&\equiv\begin{bmatrix}1&0&0&1\\0&1&0&0\\1&0&0&1\\0&0&0&0\\0&1&1&1\end{bmatrix}^T\vee\&\begin{bmatrix}0&1\\1&1\\1&0\\0&0\\1&0\end{bmatrix}=$$

$$=[x_1,x_2,x_3,x_4]\&\equiv\begin{bmatrix}1&0&1&0&0\\0&1&0&0&1\\0&0&0&0&1\\1&0&1&0&1\end{bmatrix}\vee\&\begin{bmatrix}0&1\\1&1\\1&0\\0&0\\1&0\end{bmatrix}=$$

$$=\begin{bmatrix}(x_1\equiv1)&(x_2\equiv0)&(x_3\equiv0)&(x_4\equiv1)\\(x_1\equiv0)&(x_2\equiv1)&(x_3\equiv0)&(x_4\equiv0)\\(x_1\equiv1)&(x_2\equiv0)&(x_3\equiv0)&(x_4\equiv1)\\(x_1\equiv0)&(x_2\equiv0)&(x_3\equiv0)&(x_4\equiv0)\\(x_1\equiv0)&(x_2\equiv1)&(x_3\equiv1)&(x_4\equiv1)\end{bmatrix}\vee\&\begin{bmatrix}0&1\\1&1\\1&0\\0&0\\1&0\end{bmatrix}=$$

$$=\begin{bmatrix}x_1&\bar{x}_2&\bar{x}_3&x_4\\\bar{x}_1&x_2&\bar{x}_3&\bar{x}_4\\x_1&\bar{x}_2&\bar{x}_3&x_4\\\bar{x}_1&\bar{x}_2&\bar{x}_3&\bar{x}_4\\\bar{x}_1&x_2&x_3&x_4\end{bmatrix}\vee\&\begin{bmatrix}0&1\\1&1\\1&0\\0&0\\1&0\end{bmatrix}=$$

$$=[\,\bar{x}_1x_2\bar{x}_3\bar{x}_4\vee x_1\bar{x}_2\bar{x}_3x_4\vee\bar{x}_1x_2x_3x_4,\;x_1\bar{x}_2\bar{x}_3x_4\vee\bar{x}_1x_2\bar{x}_3\bar{x}_4\,]$$

3 Structural model of a static logic object

Let

$\lambda_j:\{0,1\}^m\to\{0,1\}:\langle x_1,x_2,\ldots,x_m\rangle\mapsto y_j$ be a Boolean function and the literal x^σ be $x^\sigma=x\sigma\vee\overline{x}\overline{\sigma}\;(\sigma\in\{0,1\})$, where $x^0=\bar{x}$ and $x^1=x$. Note that the function λ_j can be expressed as *cnd*

$$cndf(\lambda_j)=\bigvee_{\substack{\langle\sigma_1,\sigma_2,\ldots,\sigma_m\rangle\\ \lambda_j(\sigma_1,\sigma_2,\ldots,\sigma_j)=1}}x_1^{\sigma_1}x_2^{\sigma_2}\ldots x_m^{\sigma_m}$$

or as *ndf* (a normal disjunctive formula)

$$ndf(\lambda_j)=\bigvee_{\langle\sigma_{j1},\sigma_{j2},\ldots,\sigma_{jk}\rangle}x_{j1}^{\sigma_{j1}}x_{j2}^{\sigma_{j2}}\ldots x_{jk}^{\sigma_{jk}}=\bigvee_i K_{ij},$$

where K_{ij} is a normal conjunct on

$X=\{x_1,\bar{x}_1,x_2,\bar{x}_2,\ldots,x_m,\bar{x}_m\}$ or finally expressed as $mndf(\lambda_j)$ (minimal *ndf*). Let the symbol K denote a set of conjuncts K_{ij} from $ndf(\lambda_j)$.

Let

$\langle\{0,1\}^m,\{0,1\}^n,\Lambda\rangle$

be a finite automaton model of a binary static logic object where Λ is the vector output function $\left(\Lambda\in\left(\{0,1\}^n\right)^{\{0,1\}^m}\right)$

$\Lambda:\{0,1\}^m\{0,1\}^n:\langle x_1,x_2,\ldots,x_m\rangle\mapsto\langle y_1,y_2,\ldots,y_n\rangle$

represented by a system $\{\lambda_j\}_{j=1}^n$ of components, output functions

$\lambda_j:\{0,1\}^m\to\{0,1\}:\langle x_1,x_2,\ldots,x_m\rangle\mapsto y_j.$

Let M_{in} (M_{out}) be the input (output) matrix of the given static object, i.e.:

$M_{\text{in}}:\{1,2,\ldots,|K|\}\times\{1,2,\ldots,2m\}\to\{0,1\}:$

$\langle r,s\rangle\mapsto 0\,for\;x_s^{\sigma_s}\notin K_{ij},\;\langle r,s\rangle\mapsto 1\,for\;x_s^{\sigma_s}\in K_{ij},$

$M_{\text{out}}:\{1,2,\ldots,|K|\}\times\{1,2,\ldots,n\}\to\{0,1\}:$

$\langle r,s\rangle\mapsto 0\,for\;K_{ij}\notin ndf(\lambda_j),\;\langle r,s\rangle\mapsto 1\,for\;K_{ij}\in ndf(\lambda_j).$

If $\mathbf{X}^T: \{1,2,\dots,2m\} \to X : s \mapsto x_s$ for $s = 2k+1$, $s \mapsto \bar{x}_s$ for $s = 2k$ ($k = 0,1,2,\dots, m-1$), then for the trivial logic array M the following are valid

$$[y_1, y_2, \dots, y_n] = (M_{\mathrm{in}} \&\& X)^T \vee \& M_{\mathrm{out}},$$

$$[\bar{y}_1, \bar{y}_2, \dots, \bar{y}_n] = \overline{(M_{\mathrm{in}} \&\& X)^T \vee \& M_{\mathrm{out}}},$$

where $X = [x_1, \bar{x}_1, x_2, \bar{x}_2, \dots, x_m, \bar{x}_m]$.

The total capacity (area) $C(M)$ (bit) of trivial logic array M can be written as

$$C(M) = C(M_{\mathrm{in}}) + C(M_{\mathrm{out}}) = (2m+n)|K|.$$

Example 2: Design a $PLA\,(M_{\mathrm{in}}, M_{\mathrm{out}})$ modeled system of output functions

$$y_1 = (x_1 \oplus x_2)\bar{x}_3 \vee (x_1 \equiv x_2)x_3 = \bar{\xi}\bar{x}_3 \vee \xi x_3$$

$$y_2 = x_1 x_2 \vee (x_1 \oplus x_2)x_3 = x_1 x_2 \vee \bar{\xi} x_3$$

and determine its capacity

$$\left(\xi = x_1 x_2 \vee \bar{x}_1 \bar{x}_2, \bar{\xi} = \bar{x}_1 x_2 \vee x_1 \bar{x}_2\right).$$

Hence

$$M_{in} = \begin{array}{c} \begin{array}{cccccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 & \xi & \bar{\xi} \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{array}{cc} x_1 & x_2 \\ \bar{x}_1 & \bar{x}_2 \\ \bar{x}_1 & x_2 \\ x_1 & \bar{x}_2 \\ \bar{\xi} & \bar{x}_3 \\ \xi & x_3 \\ \bar{\xi} & x_3 \end{array}$$

$$M_{\mathrm{out}} = \begin{array}{c} \begin{array}{ccc} y_1 & y_2 & \xi \end{array} \\ \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array} \begin{array}{cc} x_1 & x_2 \\ \bar{x}_1 & \bar{x}_2 \\ \bar{x}_1 & x_2 \\ x_1 & \bar{x}_2 \\ \bar{\xi} & \bar{x}_3 \\ \xi & x_3 \\ \bar{\xi} & x_3 \end{array}$$

and Fig. 2 can be constructed.

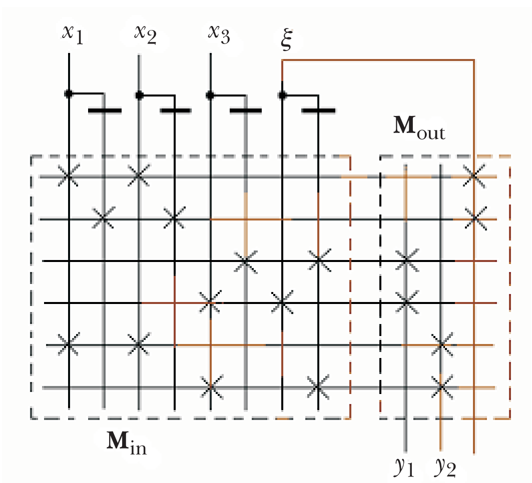


Fig. 2: *PLA* from Example 2

Since

$$[y_1, y_2, \xi] = \left(\begin{array}{c} \begin{array}{cccccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 & \xi & \bar{\xi} \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array} \&\& \begin{bmatrix} x_1 \\ \bar{x}_1 \\ x_2 \\ \bar{x}_2 \\ x_3 \\ \bar{x}_3 \\ \xi \\ \bar{\xi} \end{bmatrix} \right)^T \vee \&$$

$$\vee \& \begin{array}{c} \begin{array}{ccc} y_1 & y_2 & \xi \end{array} \\ \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array} = [\bar{x}_3 \bar{\xi} \vee x_3 \xi,\ x_1 x_2 \vee x_3 \bar{\xi},\ x_1 x_2 \vee \bar{x}_1 \bar{x}_2]$$

and $C(M) = 2 \times 4 \times 6 + 3 \times 6 = 66.$

The starting point for designing M_{in} and M_{out} trivial logic array matrices is a system of tables $\left\{\lambda_j\right\}_{j=1}^{n}$ (*ROM*) or a compressed form [2] (*PLA*, *PAL*) of it, including the corresponding record of the $cndf(\{\lambda_j\})$ of the group function or the $ndf(\{\lambda_j\})$ of the component functions of λ_j.

Example 3: The following system of functions

x_1	x_2	x_3	y_1	y_2	y_3
0	0	0	1	1	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	1	0	0
1	0	0	–	–	–
1	0	1	–	0	1
1	1	0	0	0	0
1	1	1	–	0	1

can also be written as)*

x_1	x_2	x_3	y_1	y_2	y_3
0	–	0	1	1	0
0	–	1	1	0	0
1	1	0	0	0	0
1	–	1	–	0	1

including the group *cndf* or *ndf*:

cndf

$$(y_1, y_2, y_3) = \bar{x}_1 \bar{x}_2 \bar{x}_3 (y_1\ y_2) \vee \bar{x}_1 \bar{x}_2 x_3 (y_1) \vee \bar{x}_1 x_2 \bar{x}_3 (y_1\ y_2) \vee$$
$$\vee \bar{x}_1 x_2 x_3 (y_1) \vee x_1 \bar{x}_2 x_3 (y_3) \vee x_1 x_2 x_3 (y_3)$$

or *ndf*
$(y_1, y_2, y_3) = \bar{x}_1 \bar{x}_3 (y_1\ y_2) \vee \bar{x}_1 x_3 (y_1) \vee x_1 x_3 (y_3)$.

)* Note that the undetermined values of function arguments need to be interpreted as both zeroes and ones, whereas the undetermined values of functions are very difficult, uninteresting, or even impossible, to determine and their *delivery* is motivated by the maximal rate of their utility.

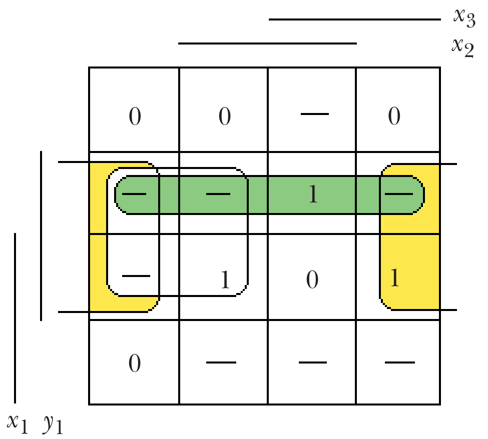
If the system $\{\lambda_j\}_{j=1}^{n}$ is also given by the system $\{cndf(y_j)\}_{j=1}^{n}$, then it is to be decided whether to realise ROM by means of the $cndf(y_j)$ array or the $cndf(\bar{y}_j)$ array. If, therefore, for the Hamming weight $w_H(y_j)$ of the function λ_j – $w_H(y_j) \le 2^m/2$ holds, then naturally we work with use $cndf(y_j)$ or otherwise we do with $cndf(\bar{y}_j)$.

Since $cndf(y_j)$ or $cndf(\bar{y}_j)$ are very complicated systems in practice – the systems of (λ_j) functions are systems consisting of tens or hundreds of functions depending on tens or hundreds of arguments – the classic minimisation procedures are not applicable on $cndf(y_j)$ or $cndf(\bar{y}_j)$. With advantage, however, the Quine and McCluske method of minimising $\{cndf(y_j)\}_{j=1}^{n}$ systems can be used, under the condition that the definition of the undetermined values of functions λ_j of system $\{\lambda_j\}_{j=1}^{n}$ will be suitably defined and the covering table [2] will be not constructed. In this way, we will obtain a subminimal group $ndf(y_1, y_2, \ldots, y_n)$ – see Example 4.

The so called 'Harvard' approach to systems $\{\lambda_j\}_{j=1}^{n}$ [3, 4] can be applied with advantage in designing logic arrays. Let us deal, without loss of generality, only on the system $\{\lambda_1(x_1, x_2, \ldots, x_m), \lambda_2(x_1, x_2, \ldots, x_m)\}$ and regard one of the functions as an argument of the other function and this, of course, affects the complexity of the conception logic array. For instance, let us write

$$y_2 = \hat{\lambda}_2(y_1, x_1, x_2, \ldots, x_m) = \lambda_2(x_1, x_2, \ldots, x_m).$$

Example 4: Let $y_1 = 00010111$ and $y_2 = 00010110$, that is
$y_1 = (x_1 \vee x_2) x_3 \vee x_1 x_2$ *and* $y_2 = \bar{x}_1 x_2 x_3 \vee x_1(\bar{x}_2 x_3 \vee x_2 \bar{x}_3)$
or also $y_2 = y_1 \overline{x_1 x_2 x_3}$ *since:*



Example 5: Let us consider the system $\{\lambda_j\}_{j=1}^{3}$ from Example 3. Then,
$w_H(y_1) = 4$, $w_H(y_2) = w_H(y_3) = 2$, *i.e.*,
$y_1 = \bar{x}_1 \bar{x}_2 \bar{x}_3 \vee \bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 x_3$,

$y_2 = \bar{x}_1 \bar{x}_2 \bar{x}_3 \vee \bar{x}_1 x_2 \bar{x}_3$, $y_3 = x_1 \bar{x}_2 x_3 \vee x_1 x_2 x_3$, *since* $\frac{2^3}{2} = 4$, *and so*

$$y_1 = \bar{x}_1,$$
$$\bar{y}_2 = y_1 \vee \bar{x}_3,$$
$$\bar{y}_3 = \bar{y}_1 \vee \bar{x}_3.$$

Example 6: Let there be a static logic object by means of a product (input) matrix and a sum (output) matrix.

$$M_{\text{in}} = \begin{array}{c} \\ 1\\2\\3\\4\\5\\6\\7\\8\\9\\10\\11 \end{array} \begin{array}{c} \begin{array}{cccccccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 & x_4 & \bar{x}_4 & x_5 & \bar{x}_5 \end{array} \\ \begin{bmatrix} 0&1&1&0&0&0&0&0&0&0\\ 1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&1\\ 0&1&0&0&0&0&0&0&0&0\\ 1&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1\\ 1&0&0&0&1&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&1&1&0&1&0\\ 0&0&0&0&0&0&0&1&0&0 \end{bmatrix} \end{array}$$

$$M_{\text{out}} = \begin{array}{c} \\ 1\\2\\3\\4\\5\\6\\7\\8\\9\\10\\11 \end{array} \begin{array}{c} \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \end{array} \\ \begin{bmatrix} 0&1&0&0&0&0&0\\ 0&0&1&0&0&0&0\\ 0&0&1&1&0&0&0\\ 1&0&0&0&0&0&0\\ 1&1&1&0&0&0&0\\ 0&0&0&0&1&0&1\\ 0&1&0&0&0&0&0\\ 1&0&1&0&0&0&0\\ 0&0&0&1&1&0&0\\ 1&0&1&1&0&0&0\\ 0&0&0&1&0&1&0 \end{bmatrix} \end{array}$$

Let the rows of the input (output) matrix be compatible if each of them contains at least one common input literal (one common output function). Since the relation of compatibility of rows is their relation of tolerance, on the set of rows of the matrix it defines the covering $C_x^{\max}\left(\{i\}_{i=1}^{11}\right)\left(C_y^{\max}\left(\{i\}_{i=1}^{11}\right)\right)$ with all maximal classes:

$$C_x^{\max}(\{i\}) = \left\{ \begin{array}{c} \{1,2,4,5,7,8\}, \{3,6,9,10\} \\ \{3,10,11\}, \{5,7,10\} \end{array} \right\}$$

$$\left(C_y^{\max}(\{i\}) = \{\{2,3,5,8,10\}, \{4,5,8,10\}, \{3,9,10,11\}, \{5,7\}, \{6,9\}\}\right)$$

Since the rows within the input (output) matrix correspond to the sets of input literals (output functions):

Rows	Stimuli	Responses
1	$\{x_1, x_2\}$	$\{y_2\}$
2	$\{x_1\}$	$\{y_3\}$
3	$\{x_4, x_5\}$	$\{y_3, y_4\}$
4	$\{x_1\}$	$\{y_1\}$
5	$\{x_1, x_3\}$	$\{y_1, y_2, y_3\}$
6	$\{x_5\}$	$\{y_5, y_7\}$
7	$\{x_1, x_3\}$	$\{y_2\}$
8	$\{x_1, x_2\}$	$\{y_1, y_3\}$
9	$\{x_5\}$	$\{y_4, y_5\}$
10	$\{x_3, x_4, x_5\}$	$\{y_1, y_3, y_4\}$
11	$\{x_4\}$	$\{y_4, y_6\}$

According to $C_x^{\max}(\{i\})\left(C_y^{\max}(\{i\})\right)$ we get to the covering on the set of input variables (output functions) of all maximum classes:

$$C_x^{\max}\left(\{x_j\}_{j=1}^{5}\right) = \{\{x_1, x_2, x_3\}, \{x_3, x_4, x_5\}\}$$

$$\left(C_y^{\max}\left(\{y_k\}_{k=1}^{7}\right) = \left\{\begin{matrix}\{y_1', y_2, y_3'\}, \\ \{y_1'', y_3'', y_4, y_5, y_6, y_7\}\end{matrix}\right\}\right)$$

i.e., we obviously obtain sparse matrices

$$M_{\text{in}} = \begin{array}{cc} & \begin{array}{cccccccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 & x_4 & \bar{x}_4 & x_5 & \bar{x}_5 \end{array} \\ \begin{array}{r} 2\\4\\7\\5\\1\\8\\10\\3\\11\\9\\6 \end{array} & \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0\\ 1&0&0&0&1&0&0&0&0&0\\ 1&0&0&0&0&1&0&0&0&0\\ 0&1&1&0&0&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&1&1&0&1&0\\ 0&0&0&0&0&0&1&0&0&1\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&1 \end{bmatrix} \end{array}$$

$$M_{\text{out}} = \begin{array}{cc} & \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \end{array} \\ \begin{array}{r} 2\\4\\7\\5\\1\\8\\10\\3\\11\\9\\6 \end{array} & \begin{bmatrix} 0&0&1&0&0&0&0\\ 1&0&0&0&0&0&0\\ 0&1&0&0&0&0&0\\ 1&1&1&0&0&0&0\\ 0&1&0&0&0&0&0\\ 1&0&1&0&0&0&0\\ 1&0&1&1&0&0&0\\ 0&0&1&1&0&1&0\\ 0&0&0&1&0&1&0\\ 0&0&0&1&1&0&0\\ 0&0&0&0&1&0&1 \end{bmatrix} \end{array}$$

The matrices M_{in} (M_{out}) are represented by submatrices

$$M_{\text{in}1} = \begin{array}{cc} & \begin{array}{cccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 \end{array} \\ \begin{array}{r} 2\\4\\7\\5\\1\\8 \end{array} & \begin{bmatrix} 1&0&0&0&0&0\\ 0&1&0&0&0&0\\ 1&0&0&0&1&0\\ 1&0&0&0&0&1\\ 0&1&1&0&0&0\\ 0&1&0&1&0&0 \end{bmatrix} \end{array},$$

$$\left(M_{\text{out}} = \begin{array}{cc} & \begin{array}{ccc} y_1' & y_2' & y_3' \end{array} \\ \begin{array}{r} 2\\4\\7\\5\\1\\8 \end{array} & \begin{bmatrix} 0&0&1\\ 1&0&0\\ 0&1&0\\ 1&1&1\\ 0&1&0\\ 1&0&1 \end{bmatrix} \end{array}\right)$$

$$M_{\text{in}2} = \begin{array}{cc} & \begin{array}{cccccc} x_3 & \bar{x}_3 & x_4 & \bar{x}_4 & x_5 & \bar{x}_5 \end{array} \\ \begin{array}{r} 10\\3\\11\\9\\6 \end{array} & \begin{bmatrix} 0&1&1&0&1&0\\ 0&0&1&0&0&1\\ 0&0&0&1&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1 \end{bmatrix} \end{array}$$

where $y_1 = y_1' \vee y_1''$ and $y_3 = y_3' \vee y_3''$ which leads to saving the capacity of matrices $M_{\text{in}1}$, $M_{\text{in}2}$ ($M_{\text{out}1}$, $M_{\text{out}2}$) compared to the capacity of M_{in} (M_{out}).

4 Structural model of a dynamic logic object

Let us consider a trivial block diagram of canonic composition of a dynamic logic object – like that in Fig. 3, where the substitute Σ [5] is a parallel register consisting of "memory " modules M_k (Fig. 5.) modeled with finite automata $\left\langle \{0,1\}, \{0,1\}^2, \delta_k \right\rangle$ where δ_k are the transition functions

$$\delta_k : \{0,1\} \times \{0,1\}^2 \to \{0,1\} : \langle q_k, e_{1k}e_{2k} \rangle \mapsto q_k',$$

where q_k and q_k' are the affiliate state and the substitute state, respectively.

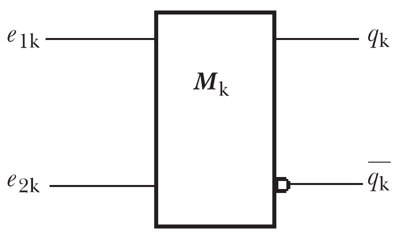


Fig. 3: "Memory" module M_k

Let $[M] = M$ or $\{\varnothing\}$ / $[m] = m$ or $\varnothing$, where M is a set and m its element ($m \in M$). The finite automaton model or the given binary dynamic object is to be an ordered sextuple $\left\langle \{0,1\}^m, \{0,1\}^p, \{0,1\}^n, \Delta, \Lambda \right\rangle$ where the vector of transitional function $\Delta\left(\Delta \in \left(\{0,1\}^p\right)^{\left(\{0,1\}^p \times \{0,1\}^m\right)}\right)$ and the output function are

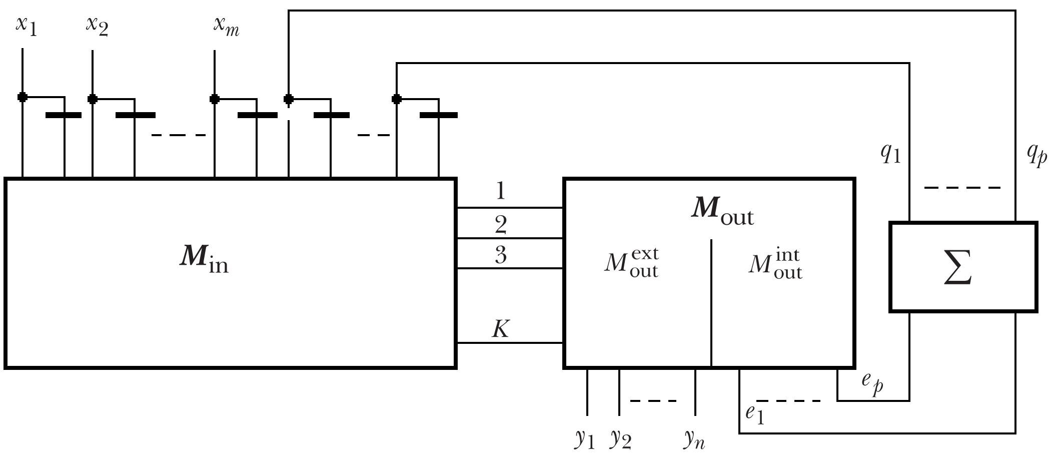


Fig. 4: Block diagram of canonical composition on logic array

$\Lambda\left(\Lambda \in \left(\{0,1\}^n\right)^{\left(\{0,1\}^p \times \left[\{0,1\}^m\right]\right)}\right)$, respectively

$$\Delta : \{0,1\}^p \times \{0,1\}^m \to \{0,1\}^p :$$
$$: \langle q_1 q_2 \dots q_p, x_1 x_2 \dots x_m \rangle \mapsto \langle q'_1 q'_2 \dots q'_p \rangle$$

$$\Lambda : \{0,1\}^p \times \left[\{0,1\}^m\right] \to \{0,1\}^n :$$
$$: \langle q_1 q_2 \dots q_p, [x_1 x_2 \dots x_m] \rangle \mapsto \langle y_1, y_2, \dots, y_n \rangle,$$

when

$$\Delta : \langle q_1 q_2 \dots q_p, x_1 x_2 \dots x_m \rangle \mapsto$$
$$\mapsto \langle \delta_1(q_1, e_{11}, e_{21}),\ \delta_2(q_2, e_{12}, e_{22}), \dots, \delta_p\left(q_p, e_{1p}, e_{2p}\right)\rangle$$

having in mind that the vector exciting function E sought for

$$\left(E \in \left(\{0,1\}^{2p}\right)^{\left(\{0,1\}^p \times \{0,1\}^m\right)}\right)$$
$$E : \{0,1\}^p \times \{0,1\}^m \to \{0,1\}^p :$$
$$: \langle q_1 q_2 \dots q_p, x_1 x_2 \dots x_m \rangle \mapsto \langle e_{11}\, e_{21}\, e_{12}\, e_{22} \dots e_{1p}\, e_{2p} \rangle$$

represented by the system $\{\varepsilon_{12k}\}_{k=1}^{p}$ of components driving functions

$$\varepsilon_{12k} : \{0,1\}^p \times \{0,1\}^m \to \{0,1\}^2 \mapsto$$
$$\mapsto \langle q_1 q_2 \dots q_p, x_1 x_2 \dots x_m \rangle \mapsto e_{1k}\, e_{2k}$$

found according to the function Δ, and Λ and is represented by the system $\{\lambda_j\}_{j=1}^{n}$ of the component input functions

$$\lambda_j : \{0,1\}^p \times \left[\{0,1\}^m\right] \to \{0,1\}^n :$$
$$: \langle q_1 q_2 \dots q_p, [x_1 x_2 \dots x_m] \rangle \mapsto y_j .$$

Let $M_{in}\left(M_{out} = \left[M_{out}^{ext}, M_{out}^{int}\right]\right)$ be the input (output, output external, output internal) matrix, (respectively) of the given dynamic object, i.e.,

$$M_{in} : \{1, 2, \dots, |K|\} \times \{1, 2, \dots, 2m\} \times \{1, 2, \dots, 2p\} \to \{0,1\} :$$
$$\langle r, s, t \rangle \mapsto 0 \text{ for } x_s^{\sigma_s}, q_t^{\sigma_t} \notin K_{ij},$$
$$\langle r, s, t \rangle \mapsto 1 \text{ for } x_s^{\sigma_s}, q_t^{\sigma_t} \in K_{ij}$$

$$M_{out}^{ext} : \{1, 2, \dots, |K|\} \times \{1, 2, \dots, 2n\} \to \{1, 2, \dots, 2n\} \to \{0,1\} :$$
$$\langle r, s \rangle \mapsto 0 \text{ for } K_{ij} \notin ndf(\lambda_j),$$
$$\langle r, s \rangle \mapsto 1 \text{ for } K_{ij} \in ndf(\lambda_j)$$

$$M_{out}^{int} : \{1, 2, \dots, |K|\} \times \{1, 2, \dots, 2p\} \to \{0,1\} :$$
$$\langle r, t \rangle \mapsto 0 \text{ for } K_{ij} \notin ndf(E_{12k}),$$
$$\langle r, t \rangle \mapsto 1 \text{ for } K_{ij} \in ndf(E_{12k}).$$

If in addition, $\boldsymbol{Q}^T : \{1, 2, \dots, 2p\} \to Q : t \to q_t$ for $t = 2k + 1$, $t \mapsto \bar{q}_t$ for $t = 2k\,(k = 0, 1, 2, \dots, p-1)$ for the logic array M of the given dynamic object there holds

$$[y_1, y_2, \dots, y_n] = \left(M_{in} \& \& \begin{bmatrix} X \\ Q \end{bmatrix}\right)^T \vee \& M_{out}^{ext},$$

$$[\overline{y_1}, \overline{y_2}, \dots, \overline{y_n}] = \overline{\overline{\left(M_{in} \& \& \begin{bmatrix} X \\ Q \end{bmatrix}\right)^T} \vee \& M_{out}^{ext}},$$

$$[e_{11}, e_{21}, e_{12}, e_{22}, \dots, e_{1p}, e_{2p}] = \left(M_{in} \& \& \begin{bmatrix} X \\ Q \end{bmatrix}\right)^T \vee \& M_{out}^{int},$$

$$[\overline{e_{11}}, \overline{e_{21}}, \overline{e_{12}}, \overline{e_{22}}, \dots, \overline{e_{1p}}, \overline{e_{2p}}] = \overline{\overline{\left(M_{in} \& \& \begin{bmatrix} X \\ Q \end{bmatrix}\right)^T} \vee \& M_{out}^{int}},$$

where $X = [x_1, \bar{x}_1, x_2, \bar{x}_2, \dots, x_m, \bar{x}_m]^T$ and
$Q = [q_1, \bar{q}_1, q_2, \bar{q}_2, \dots, q_p, \bar{q}_p]^T$.

It can be recommended to use reduced exciting matrices of "memory" modules [2, 5], i.e., a matrix with actual state transitions only (except for the delay element and D flip-flop circuit):

q_k	q'_k	D_k
0	0	0
0	1	1
1	0	0
1	1	1

$$\begin{array}{cc} R_k & S_k \\ \end{array}\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad \begin{array}{cc} D_k & L_k \\ \end{array}\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad \begin{array}{cc} J_k & K_k \\ \end{array}\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad T_k \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

The total capacity (area) $C(M)$ (*bit*) of the logic array M is expressed as

$$C(M) = C(M_{\text{in}}) + C(M_{\text{out}}^{\text{ext}}) + C(M_{\text{out}}^{\text{int}}) = \big(2(m+p) + n + 2p\big)|K|.$$

Example 7: Design a *PLA* by means of an operating table (Table 1a) and use a synchronous flip-flop *RS* as a binary substitutor. It seems suitable to modify the operating table and add the excitation *RS* flip-flop (Table 1b).

Table 1: a) the operating table,
b) the modified operating table from Example 7

a)

	$q'_1 q'_2 / y$	
$q_1 q_2$ \ x	0	1
0 0	01/1	00/0
0 1	01/1	10/0
1 0	00/0	01/1

b)

$q_1\,q_2$	x	$q'_1 q'_2$	y	R_k, S_k
0 0	0	0 1	1	S_2
	1	0 0	0	–
0 1	0	0 1	1	–
	1	1 0	0	$S_1,\ R_2$
1 0	0	0 0	0	R_1
	1	0 1	1	$R_1,\ S_2$

According to Table 1b) we obtain

$$y = \bar{q}_1\,\bar{q}_2\,\bar{x} \vee \bar{q}_1\,q_2\,\bar{x} \vee q_1\,\bar{q}_2\,x$$

as well as

$$R_1 = q_1\,\bar{q}_2 \qquad R_2 = \bar{q}_1\,q_2\,x$$

$$S_1 = \bar{q}_1\,q_2\,x \qquad S_2 = \bar{q}_1\,\bar{q}_2\,\bar{x} \vee q_1\,\bar{q}_2\,x$$

Hence

$$M_{\text{in}} = \begin{array}{c} \begin{array}{cccccc} x & \bar{x} & q_1 & \bar{q}_1 & q_2 & \bar{q}_2 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{array}{ccc} \bar{q}_1 & \bar{q}_2 & \bar{x} \\ \bar{q}_1 & q_2 & \bar{x} \\ \bar{q}_1 & q_2 & x \\ q_1 & \bar{q}_2 & x \\ q_1 & \bar{q}_2 & \end{array}$$

$$M_{\text{out}}^{\text{ext}} = \begin{array}{c} y \\ \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \end{array} \qquad M_{\text{out}}^{\text{int}} = \begin{array}{c} \begin{array}{cccc} R_1 & S_1 & R_2 & S_2 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Determine the capacity *PLA*: $C(M) = (6 + 1 + 4)\,5 = 55$ bits.

Since the number of logical array input ports is usually markedly higher than the number of rows of input as well as exciting columns, the area of the input matrix M_{in} is unnecessarily large. Replace, therefore, the input variables by suitable substitutes.

Example 8: Given a synchronous logic object (Table 2a) by an operating table (Table 2b), create a table with a minimal number of columns denoted by substitution variables $s_i (i = 1, 2, 3)$ – Table 2c [6].

Hence

$$s_i = a\big(p(1) \vee p(2)\big) \vee d\big(p(3) \vee p(5)\big),$$

$$s_2 = c\big(p(4) \vee p(6)\big) \vee e\; p(1) \vee b\; p(2),$$

$$s_3 = c,$$

where $p: \{q\}_{q=1}^{6} \to \{0,1\}: q \mapsto 0$ if the object is not in the state q, in the opposite case $q \to 1$. "Substitution" variables define on the alphabet state $\{q\}_{q=1}^{6}$ partition {{1, 2}, {3, 5}, {4, 6}} (the permutations of row elements of Table 2c can also helpful). If Miller's state coding [7] transferred on synchronous logic objects [2] is used, and if the elementary substitute will be *D* flip-flop, we obtain $w(1) = 0$ for the state.

Table 2: a) flow table, b) operating table, c) table of substitution variables from Example 8

a)

q	x	y	q'
1	a	α	2
	c	β	5
	e	γ	6
2	a	β	2
	b	δ	4
	d	α	–
3	b	α	–
	d	β	5
4	c	γ	3
5	a	β	–
	d	β	4
6	c	γ	5
	e	α	–

b)

q_1	q_2	q_3	s	y_1	y_2	q'_1	q'_2	q'_3	D_k
1	0	1	s_1	0	0	0	0	1	D_3
			c	0	0	0	0	0	–
			s_2	1	0	0	1	1	D_2, D_3
0	0	1	s_1	0	0	0	0	1	D_3
			s_2	1	1	0	1	0	D_2
			–	0	1		–		–
1	0	0	–	0	1		–		–
			s_1	0	0	0	0		–
						0			
0	1	0		1	0	1	0		D_1
			s_2			0			
0	0	0	–	0	0		–		–
			s_1	0	0	0	1	0	D_2
0	1	1		1	0	0	0	0	–
			s_2	0	1		–		–
			–						

c)

q	s_1	s_2	s_3
1	a	e	c
2	a	b	–
3	d	–	–
4	–	c	–
5	d	–	–
6	–	c	–

Weights $w(1)=0, w(3)=w(6)=1, w(2)=w(4)=2, w(5)=3$ $w(q)=\chi(q')$, where $\chi(q')$ is the characteristic function of the set $\{q' | \chi(q')\}$ of the state binary code

$$\{i\}_{i=1}^{6} \to \{0,1\}^3 : 5 \mapsto 000, 3 \mapsto 100, 6 \mapsto 011, 4 \mapsto 010,$$
$$2 \mapsto 001, 1 \mapsto 101,$$

which is a reasonable compromise:

$$\begin{array}{c|c|c|c|c|}
 & & & q_3 & q_3 \\
 & & q_2 & q_2 & \\
\hline
q_1 & 5 & 4 & 6 & 2 \\
\hline
 & 3 & – & – & 1 \\
\hline
\end{array}$$

Note, in addition, that the states of one and the same class {1, 2}, {3, 5}, {4, 6} of the partition {{1, 2}, {3, 5}, {4, 6}} is to be coded also with respect to the neighbouring coding, where a heuristic procedure lead to the reduction number of rows of the array matrices, i.e.:

$$s_1 = (q_1\, q_2\, q_3 \vee \bar{q}_1\, q_2\, q_3)\, a \vee (q_2\, \bar{q}_2\, \bar{q}_3 \vee \bar{q}_1\, \bar{q}_2\, \bar{q}_3)\, d =$$
$$= q_2\, q_3\, a \vee \bar{q}_2\, \bar{q}_3\, d,$$

$$s_3 = c,$$

$$s_2 = (q_1\, q_2\, \bar{q}_3 \vee \bar{q}_1\, q_2\, \bar{q}_3)\, c \vee q_1\, q_2\, q_3\, e \vee \bar{q}_1\, q_2\, q_3\, b =$$
$$= q_2\, \bar{q}_3\, c \vee q_1\, q_2\, q_3\, e \vee \bar{q}_1\, q_2\, q_3\, b.$$

Let the input binary code be

$$\{a, b, c, d\} \to \{0,1\}^3 : a \mapsto 000, b \mapsto 001, c \mapsto 010,$$
$$d \mapsto 011, e \mapsto 100$$

i.e.:

$$s_1 = \bar{x}_1\, \bar{x}_2\, \bar{x}_3\, q_2\, q_3 \vee \bar{x}_1\, x_2\, x_3\, \bar{q}_2\, \bar{q}_3,$$
$$s_2 = \bar{x}_1\, x_2\, \bar{x}_3\, q_2\, \bar{q}_3 \vee x_1\, \bar{x}_2\, \bar{x}_3\, q_1\, q_2\, q_3 \vee \bar{x}_1\, \bar{x}_2\, x_3\, \bar{q}_1\, q_2\, q_3;$$

$$\text{hence} \quad [s_1, s_2] = \left(\begin{array}{c} \begin{array}{cccccccccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 & q_1 & \bar{q}_1 & q_2 & \bar{q}_2 & q_3 & \bar{q}_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array} \&\& \begin{bmatrix} x_1 \\ \bar{x}_1 \\ x_2 \\ \bar{x}_2 \\ x_3 \\ \bar{x}_3 \\ q_1 \\ \bar{q}_1 \\ q_2 \\ \bar{q}_2 \\ q_3 \\ \bar{q}_3 \end{bmatrix} \right)^T \vee \& \begin{array}{c} \begin{array}{cc} s_1 & s_2 \end{array} \\ \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array} =$$

$$= \left[\bar{x}_1\, \bar{x}_2\, \bar{x}_3\, q_2\, q_3 \vee \bar{x}_1\, x_2\, x_3\, \bar{q}_2\, \bar{q}_3 \vee x_1\, \bar{x}_2\, \bar{x}_3\, q_1\, q_2\, q_3 \vee \bar{x}_1\, \bar{x}_2\, x_3\, \bar{q}_1\, q_2\, q_3 \right].$$

Let us code the responses according to Miller again. This can also be used with advantage in coding reactions of asynchronous objects:

$$\{\alpha, \beta, \gamma, \delta\} \to \{0,1\}^2:$$
$$:\beta \mapsto 00, \alpha \mapsto 01, \gamma \mapsto 10, \delta \mapsto 11,$$

since the response weights are $w(\beta) = 5$, $w(\alpha) = 4$, $w(\gamma) = 3$ and $w(\delta) = 1$. For excitation and reactions we can write:

$$D_1 = \bar{q}_1\, q_2\, \bar{q}_3\, s_2$$

$$D_2 = (q_1\, \bar{q}_2\, q_3 \vee \bar{q}_1\, \bar{q}_2\, q_3)\, s_2 \vee \bar{q}_1\, \bar{q}_2\, \bar{q}_3\, s_1 = \bar{q}_2\, q_3\, s_2 \vee \bar{q}_1\, \bar{q}_2\, \bar{q}_3\, s_1$$

$$D_3 = q_1\, \bar{q}_2\, q_3\, s_1 \vee q_1\, \bar{q}_2\, q_3\, s_2 \vee \bar{q}_1\, \bar{q}_2\, q_3\, s_1 = \bar{q}_2\, q_3\, s_1 \vee q_1\, \bar{q}_2\, q_3\, s_2$$

$$y_1 = (q_1\, \bar{q}_2\, q_3 \vee \bar{q}_1\, \bar{q}_2\, q_3 \vee \bar{q}_1\, q_2\, \bar{q}_3 \vee \bar{q}_1\, q_2\, q_3)\, s_2 = (\bar{q}_1\, q_2 \vee \bar{q}_2\, q_3)\, s_2$$

$$y_2 = q_1\, \bar{q}_2\, q_3\, s_1 \vee (\bar{q}_1\, \bar{q}_2\, q_3 \vee q_1\, \bar{q}_2\, \bar{q}_3 \vee \bar{q}_1\, q_2\, q_3)\, s_2 = q_1\, \bar{q}_2\, q_3\, s_1 \vee (\bar{q}_1\, q_3 \vee q_1\, \bar{q}_2\, \bar{q}_3)\, s_2.$$

Hence

$$[y_1, y_2, D_1, D_2, D_3] = \left(\begin{array}{cccccccccc} q_1 & \bar{q}_1 & q_2 & \bar{q}_2 & q_3 & \bar{q}_3 & s_1 & \bar{s}_1 & s_2 & \bar{s}_2 \\ \multicolumn{10}{c}{\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}} \end{array} \&\& \begin{bmatrix} q_1 \\ \bar{q}_1 \\ q_2 \\ \bar{q}_2 \\ q_3 \\ \bar{q}_3 \\ s_1 \\ \bar{s}_1 \\ s_2 \\ \bar{s}_2 \end{bmatrix} \right)^T \vee\& \begin{array}{ccccc} y_1 & y_2 & D_1 & D_2 & D_3 \\ \multicolumn{5}{c}{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \end{array}$$

Example 9: Given an asynchronous logic object (Table 3a) by an operating table (Table 3b), create a table with a minimal number of columns denoted by "substitute" variable σ_i ($i = 1, 2, 3$) – (Table 3c). Since $\sigma_1 = a, \sigma_2 = b$ and $\sigma_3 = c$, let us discuss just the parallel coding of states $\{q\}_{q=1}^{6} \to \{0,1\}^5$ according to Liu [2]. In the case of single stimuli a, b, c, the noncritical pairs of state transitions on the state alphabet $\{q\}_{q=1}^{6}$ define

– $\{\{1,2,3\}, \{4,5,6\}\}$,
– $\{\{1,4\}, \{2,3\}, \{5,6\}\}$,
– $\{\{1,2\}, \{3,4\}, \{5,6\}\}$.

Hence for the separation state components [2] we obtain a state code (Table 3d), again with an obvious effort to achive the least possible number (Table 3e) of state components q ($j = 1, 2, 3$) and the least possible $w_H(D_j)$ exciting weight D_j.

Hence $$[\xi, \zeta, \eta, \vartheta] = \left(\begin{array}{cccccc} x_1 & \bar{x}_1 & x_2 & \bar{x}_2 & x_3 & \bar{x}_3 \\ \multicolumn{6}{c}{\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}} \end{array} \&\& \begin{bmatrix} x_1 \\ \bar{x}_1 \\ x_2 \\ \bar{x}_2 \\ x_3 \\ \bar{x}_3 \end{bmatrix} \right)^T \vee\& \begin{array}{ccccc} \xi & \zeta & \eta & \vartheta & \kappa \\ \multicolumn{5}{c}{\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \end{array}$$

$$
\left[D_1, D_2, D_3\right] = \left(\begin{array}{cccccccccccccccc}
q_1 & \bar{q}_1 & q_2 & \bar{q}_2 & q_3 & \bar{q}_3 & \xi & \bar{\xi} & \zeta & \bar{\zeta} & \eta & \bar{\eta} & \vartheta & \bar{\vartheta} & \kappa & \bar{\kappa} \\
\end{array}
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{bmatrix}
\,\&\&\,
\begin{bmatrix}
q_1 \\ \bar{q}_1 \\ q_2 \\ \bar{q}_2 \\ q_3 \\ \bar{q}_3 \\ \xi \\ \bar{\xi} \\ \zeta \\ \bar{\zeta} \\ \eta \\ \bar{\eta} \\ \vartheta \\ \bar{\vartheta} \\ \kappa \\ \bar{\kappa}
\end{bmatrix}
\right) \vee\&
\begin{array}{ccc} D_1 & D_2 & D_3 \end{array}
\begin{bmatrix}
1 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 1 & 0 \\
0 & 1 & 1 \\
0 & 1 & 0 \\
0 & 1 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 1 \\
0 & 0 & 1 \\
0 & 0 & 1 \\
0 & 0 & 1
\end{bmatrix}
$$

Table 3: a) transition table, b) operating table, c) table of input "substitute" variables, d) state code table, e) state code table with a minimum state code words from Example 9

a)

q	x	q'
1	a	1
	b	4
	c	2
2	a	1
	b	3
	c	2
3	a	1
	b	3
	c	3
4	a	4
	b	4
	c	3
5	a	4
	b	6
	c	5

b)

$q_1\, q_2\, q_3$	$x_1\, x_2$	$q'_1\, q'_2\, q'_3$	D_k
0 0 0	0 0	0 0 0	–
	0 1	0 0 1	D_5
	1 0	0 1 0	D_3
0 1 0	0 0	0 0 0	–
	0 1	0 1 1	D_3, D_5
	1 0	0 1 0	D_3
0 1 1	0 0	0 0 0	–
	0 1	0 1 1	D_3, D_5
	1 0	0 1 1	D_3, D_5
0 0 1	0 0	0 0 1	D_5
	0 1	0 0 1	D_5
	1 0	0 1 1	D_3, D_5
1 0 0	0 0	0 0 1	D_5
	0 1	1 1 0	D_1, D_3
	1 0	1 0 0	D_1

c)

q	σ_1	σ_2	σ_3
1	a	b	c
2	a	b	c
3	a	b	c
4	a	b	c
5	a	b	c
6	a	b	c

d)

x	a	b	c
q	q_1	$q_2\, q_3$	$q_4\, q_5$
1	0	0 0	0 0
2	0	0 1	0 0
3	0	0 1	0 1
4	0	0 0	0 1
5	1	1 –	1 –
6	1	1 –	1 –

e)

q	q_1	q_2	q_3
1	0	0	0
2	0	1	0
3	0	1	1
4	0	0	1
5	1	0	0
6	1	1	0

5 Conclusions

In [8], we find exact, theoretically demanding and quite elaborate optimization algorithms both according to state coding (Miller's economic coding and Liu's parallel coding) and by the number rows of matrices of the matrix structural model of the dynamic logic object.

If the number of states of the dynamic object is large, it is suitable to apply a simple block decomposition [2, 9] and design the structural model of the matrix as a composition of matrix structural models of individual blocks.

References

[1] Liebig, H., Thome, S.: *Logischer Entwurf digitaler Systeme*. Berlin -…- Tokio: Springer, 1996.

[2] Bokr, J., Jáneš, V.: *Logické systémy*. Praha: Vydavatelství ČVUT, 1999.

[3] Pospelov, D. A.: *Logičeskie metody analiza schem*. Moskva: Energija, 1974.

[4] Šalyto, A. A.: *Logičeskoe upravlenie. Metody apparatnoj i programmnoj realizaciji algoritmov*. Sankt Peterburg, 2000.

[5] Bokr, J.: Kánonická dekompozice. *Acta Elektrotechnica et Informatica*, Vol. **4** (2004), No. 1, p. 60–65.

[6] Baranov, S. I., Sinev, V. N.: *Avtomaty i programmiruemye matricy*. Minsk: Vyšejšaja škola, 1980.

[7] Miller, R. E.: "Switching Theory". *Sequential Circuits and Machines*, Vol. II. (Translated into Russian), Moskva: Nauka, 1971.

[8] Ačasova, S. M.: *Algoritmy sinteza avtomatov na programmirujemych matricach*. Moskva: Radio i svjaz, 1987.

[9] Bokr, J.: "Prostá bloková dekompozice". *Slaboproudý obzor*, sv. 52 (1991), č. 3–4, p. 80–83.

Doc. Ing. Josef Bokr, CSc.
e-mail: bokr@kiv.zcu.cz

Department of Information and Computer Science

University of West Bohemia
Faculty of Applied Sciences
Universitní 22
306 14 Pilsen, Czech Republic

Doc. Ing. Vlastimil Jáneš, CSc.
phone: +420 224 357 289
e-mail: janes@fel.cvut.cz

Department of Computer Science and Engineering

Czech Technical University in Prague
Faculty of Electrical Engineering
Karlovo nám.13
121 35 Praha 2, Czech Republic