STUDIA INFORMATICA
Nr 1/2(7) Systemy i technologie informacyjne 2006

Method of Logical Synthesis of Integrated Circuits in basis *K-PLA*

Sergey Novikov
Institute of Computer Science, University of Podlasie
Sienkiewicz Str. 51, 08-110 Siedlce

Modern Computer-Aided Design systems of Very Large Scale Integration allow to use the chips with thousands of components as element's modules of electronic circuits. For example, ULSI - chip has complexity more than 20000 transistors. One of perspective chips (Programmable Logic Chip) has the architecture of a Programmable Logic Array (PLA). PLA can be programmed in a laboratory to perform complex functions with the help of a special equipment called by programmer (similar as PROM blower).

There are different methods of the logical synthesis of circuits in PLA [1,2,3]. These integrated circuits have many advantages (simplicity of layout design, testing and modification), because the circuits are regular.

Let's consider a task of designing of a digital circuit described by the system of partial many-valued logical functions in basis ***K-PLA*** , which is one from modern generalizations of PLA [4].

It is well known, that the mathematical model of functioning of the each code translator (m,r,q) of binary words is a system of partial Boolean functions. If a translator has m inputs, r outputs and reforms q binary input words with length m to q binary output words with length r, and $q < 2^m$, then we have a system (m,r,q) of the partial Boolean functions $G=\{g_1, g_2, ...,g_\gamma\}$, defined on q binary words, where $g_j (\alpha_1,\alpha_2,\ldots,\alpha_m) \in \{0,1\}$, $\alpha_i \in \{0,1\}$, $i\in \{1,2,...,m\}$, $j\in\{1,2,...,r\}$.

The translator (m,r,q) can be realized by a circuit with one $PLA(m',r',q')$, which has not less than m input pins ($m' \geq m$), r output pins ($r' \geq r$) and q intermediate lines ($q' \geq q$). The elementary products (implicants) are realized on the intermediate lines of matrix ***AND*** of ***PLA***, sum-of-products are realized in matrix ***OR*** of ***PLA***.

Otherwise circuits have more chips and the difficulty of the synthesis grows.

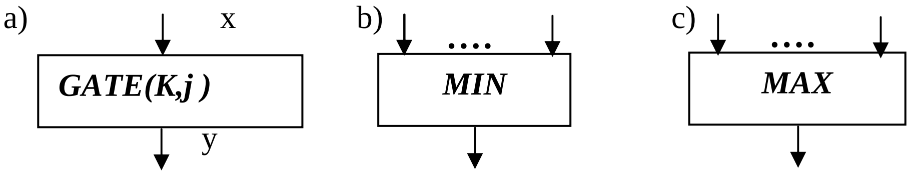
Modern ***PLA*** have $m' < 20$, $r'< 16$, $q' < 100$, as here exist tasks designing of translators for long binary words with $m \geq 64$.

The problem of increase of the main parameters of ***PLA*** reduces to the problem of ***PLA***-area minimization [3]. It is known, that the large chip's area contains 10% the active elements and 90% commutation lines. Inculcation of modern gates and chips with many internal states (***K-PLA***) allows to reduce the ***PLA***-area. However, the corresponding theory and new methods of the synthesis are necessary for reaching of this aim.

The skill to minimize the systems of partial many-valued logical functions [5, 6] allows to suggest the new method for circuit's realization of systems of partial many-valued logical functions in new basis ***K-PLA***, where all gates have ***K*** internal states.

In this paper we propose the new gates with $K>2$ internal states and the new method of the synthesis of circuits with the help of that gates .

We propose the gates:



The gate ***GATE(K,j)*** has one input pin and one output pin, where $y = K-1$ for $x=j$ and $y = 0$ for $x \neq y$. We must be able to modify the main parameters ***K*** and ***j*** for ***GATE(K,j)*** with the help of the special programmer.

The gates ***MIN*** and ***MAX*** realize the generalizations on case $K>2$ of operations "conjunction" and "alternation".

Our method of the synthesis of circuits with the help of our gates allows to reduce a task of the realization of the system ***(m,r,q)*** partial Boolean functions by integrated circuits to the realization's task of the system ***(n,s,q)*** of the partial ***K***-valued logical functions, with $n < m$, $s < r$.

The input information for the synthesis is the system of partial Boolean functions ***(m,r,q)*** and the basis (specific parameters of many-valued gates), which we can use.

First of all, we must choose the value ***K*** (without calculation of difficulty of solution of the technical tasks of realization ***K***-valued elements).

Let's take for instance the system ***(18,6,20)*** of partial Boolean functions (Fig.1), which are reduced to the system ***(6,2,20)*** of partial ***8***-valued logical functions (Fig.2).

Let our translator ***(m,r,q)*** has $r < m$.

Then we must code the binary entrance words with length ***m*** and the binary output words with length ***r*** by ***K***-valued words.

In order to code we shall divide the binary entrance words with length ***m*** on blocks with length ***a***, where $a = \log_2 K$. We can code the each block (binary word with length ***a***) by the corresponding word with length ***1*** in alphabet $\{0,1,2,...,K-1\}$. Hence we can replace the each binary entrance word with length ***m*** by the corresponding word with length ***$n =]\, m/\log_2 K\, [$.*** We can replace the each binary output word of the translator ***(m,r,q)*** with length ***r*** by the corresponding word with length ***$s =]\, r/\log_2 K\, [$*** in alphabet ***{0,1,2,...,K-1}.*** In this way we get the system ***(n,s,q)*** of partial ***K***-valued logical functions in place of the system ***(m,r,q)*** of the partial Boolean functions. The values ***K***, where ***$\log_2 K = [\, \log_2 K]$***, are very interesting. For example, we can reduce the length of entrance and output words in time out of two, if ***K = 4***, and reduce in time out of three, if ***K = 8.***

For circuit's realization of ***(m,r,q)*** we propose a circuit ***S*** with ***K-PLA,*** translator of binary words in ***K***-valued words (***T(2/K)***) and translator of ***K***-valued words in binary words (***T(K/2)***) (Fig.3).

It's easy to explain structures of all blocs showed on Fig. 3 with the help of the concrete examples.

Fig.4 shows the structure of the translator ***T(2/K)*** for three input pins and ***K=8,*** where

⬭ is the gate NOT, ● is the gate AND, ▲ (▼) is the gate for realization of the operation ***Min*** (***Max***), ■ is the gate ***GATE(K,j) .*** The second row from below includes gates: ***MIN(0,x)*** for generation of signal 0, ***MIN(1,x)*** for generation of signal 1, ***GATE(3,1)*** for generation of signal 2, ***GATE(4,1)*** for generation of signal 3, ***GATE(5,1)*** for generation of signal 4, ***GATE(6,1)*** for generation of signal 5, ***GATE(7,1)*** for generation of signal 6, ***GATE(8,1)*** for generation of signal 7.

If input signals of ***T(2/K)*** are equal to ***xi1 = 1, xi2 = 0, xi3 = 1,*** to the input signal of ***GATE(6,1)*** is equal to 1 and it's output signal is equal to 5. Then the output signal of ***T(2/K)*** is equal to 5 also.

Fig.5 shows the structure of the translator ***T(K/2)*** for three input pins and ***K=8,*** where

▼ is the gate OR, ▲ is the gate for realization of the operation ***Min(1,x)***, ■ is the gate ***GATE(K,j) .*** The first row from top of gates includes next gates: two gates at the left ***GATE(1,0)*** and ***GATE(2,1)*** for generation of signals 0 and 1, ***GATE(3,2)*** for generation of signal 2, ***GATE(4,3)*** for generation of signal 3, ***GATE(5,4)*** for generation of signal 4, ***GATE(6,5)*** for generation of signal 5, ***GATE(7,6)*** for generation of signal 6, ***GATE(8,7)*** for generation of signal 7.

For example, if the input signal of ***T(K/2)*** is equal to ***fi = 4,*** the output signal on the fifth at the left gate ***GATE(5,4)*** is equal to 4 so and the output signal on the corresponding gate ***MIN*** is equal to 1. Here we have ***gi1 =1, gi2 = 0, gi3 = 0.***

Figure 6 shows the structure of ***K-PLA(6,2,21)*** for ***K=8*** with 6 input pins, 2 output pins and 20 intermediate lines. This block includes gates ***MIN, MAX, GATE(8,j), MIN(C,x),*** which are described as ▲ **,**▼**,** □ and △ .

The gates □ and △ can turn for realization of the concrete parameters with the help of the special addition equipment for ***K-PLA,*** which was called programmer [2]. The programmer function under control of the matrix for turning of the gates of ***K-PLA***, which in our case (for realization of the system ***(6,2,20)*** on

Fig.2) is shown on Fig.7. This matrix for turning of the gates also allows to describe the structure of ***K-PLA.*** The commutation points between input pins and intermediate lines in ***AND*** matrix of ***K-PLA*** correspond to fixed elements from ***{0,1,…,K-1,K}*** of the matrix for turning ***M(AND),*** where the number ***K*** signifies the absence of the corresponding gait, and commutation points between output pins and intermediate lines in ***OR*** matrix of ***K-PLA*** correspond to elements from ***{0,1,…, K-1,K}*** of the matrix for turning ***M(OR).***

The ***K-PLA*** – programmer can remove unnecessary gates if in correspomding positions of the matrix for turning are numbers ***K*** .

With the help of translators ***T(2/K), K-PLA*** and ***T(K/2)*** we can realize ***(m,r,q)*** by the circuit with blocs showed on Fig.3.

There are the main stages of our method of the logical synthesis of integrated circuits.

1. The choice of the quantity of ***K*** for gates in a circuit's realization of the system of partial Boolean functions ***(m,r,q)***. We take that ***K***, where $log_2 K = [log_2 K].$

2. The coding the binary input and output words of ***(m,r,q)*** by ***K***-valued words with length $n =]m/log_2 K[$ and $s =]r/log_2 K[.$

In this stage we construct the system of partial many-valued logical functions ***(n,s,q),*** which corresponds to ***(m,r,q).***

3. The construction of the circuit with ***T(2/K), K-PLA*** and ***T(K/2)*** showed on Fig. 3 for realization of ***(m,r,q).***

4. The minimization of the system of partial ***K***-valued logical functions ***(n,s,q)***.

With that aim we generate system-intervals, which realize by implicants, and construct SOP-system in the basis $\{\vee, \wedge, v_i^j, 1, 2, ..., K\text{-}2\}$ [5,6].

5. The creation the special matrixes ***M(AND)*** and ***M(OR)*** for turning of the concrete parameters of gates for ***K-PLA*** with the help of the ***K-PLA*** – programmer.

6. The completion of the synthesis with the help of the ***K-PLA*** – programmer.

We continue description of our method with the help of the example.

Let we have ***K=8*** for the system ***(18,6,20)*** on Fig.1 and the system ***(6,2,20)*** on Fig. 2.

In our example we have for minimization two ***8***-valued functions ***(6,2,20)*** on Fig. 2.

For the effective synthesis of ***8-PLA*** we can minimize the obtained ***8*** **–**valued logical functions by our method [5,6].

Creating intervals we must follow that the interval covering row β, where $f(\beta) = j$ can't to cover row γ, where $f(\gamma) = i < j.$ Then we create implicants and the SOP-system in basis $\{\vee, \wedge, v_i^j, 1, 2, ..., K\text{-}2\}$, which was proposed in [5] **.**

Using the tools from [5,6], we have results of the minimization - system's intervals for the system ***(6,2,20)*** :

R1={(4 - - - - - {2}) };
R2={(- - 4 -2- {1}),(- - - -7- {1}),(- - - 7 - - {2}) };
R3={(- - - - - 4 {1}),(7- - - - - {2}),(- - 6 - - - {2}), (- - - - 0 - {1}) };
R4={(- - 0 - - - {2}),(- - - -1 - {1,2}) };
R5={(- - - - - 5 {1,2}),(- 26 - - - {2}),(- - - - 6 - {2}), (- - - - - 0 {1}) };
R6={(- - 7 - - - {1,2}), (04 - - - - {2}), (06 - - - - {1}), (- 7 - - - - {1}), (0 - 4 - - - {2})};
R7={(7 - - - 6 - {1,2}),(1 - - - - - {1}) }.

Then we have the representation (SOP-system):

$$f_1 = P21 \vee P22 \vee P31 \vee P34 \vee P42 \vee P51 \vee P54 \vee P61 \vee P63 \vee P64 \vee P71 \vee P72 = (2 \wedge (v_3^4 \wedge v_5^2)) \vee (2 \wedge v_5^7) \vee (3 \wedge v_6^4) \vee (3 \wedge v_5^0) \vee (4 \wedge v_5^1) \vee (5 \wedge v_6^5) \vee (5 \wedge v_6^0) \vee (6 \wedge v_3^7) \vee (6 \wedge (v_1^0 \wedge v_2^6)) \vee (6 \wedge v_2^7) \vee (v_1^7 \wedge v_5^6) \vee v_1^1;$$

$$f_2 = P11 \vee P23 \vee P32 \vee P33 \vee P41 \vee P42 \vee P51 \vee P52 \vee P53 \vee P61 \vee P62 \vee P65 \vee P71 = (1 \wedge v_1^4) \vee (2 \wedge v_5^7) \vee (3 \wedge v_1^7) \vee (3 \wedge v_3^6) \vee (4 \wedge v_3^0) \vee (4 \wedge v_5^1) \vee (5 \wedge v_6^5) \vee (5 \wedge (v_2^2 \wedge v_3^6)) \vee (5 \wedge v_5^6) \vee (6 \wedge v_3^7) \vee (6 \wedge (v_1^0 \wedge v_2^4)) \vee (6 \wedge (v_1^0 \wedge v_2^6)) \vee (v_1^7 \wedge v_5^6).$$

It's easy to realize our SOP-system by alone ***8-PLA(6,2,21)***. In our case we have the circuit, where the structure of ***8-PLA(6,2,21)*** is showed on Fig. 8 and the matrixes for turning are showed on Fig. 9.

At last we have the circuit for realization our translator ***(18,6,20),*** which is showed on Fig. 10 .

We can apply new results received in this paper to investigation of the problem reducing the chip-area.

The each system ***(m,r,q)*** of ***r*** partial Boolean functions can be easily realized by ***PLA***-circuit with alone ***2-PLA(m,r,q),*** where are ***m*** inputs pins, ***r*** output pins and ***q*** intermediate lines. For realization ***(m,r,q)*** it is necessary one ***2-PLA(m,r,q)***, where usually ***r < m***, containing ***L(m,r,q)*** commutation points between input/output pins and intermediate lines, where

$$L(m,r,q) = (2m + r)*q. \qquad (1)$$

In our example for realization ***(18,6,20)*** by the circuit with one ***2-PLA(18,2,20)*** we need the chip-area, which is equal to ***L(18,6,20) = 840.***

Our new method of synthesis allows to aid the circuit with showed on Fig.3 blocks for realization of that ***(18,6,20),*** where ***T(2/8)***- area is equal to $L_1 = 56$, ***T(8/2)***- area is equal to $L_2 = 32$ and ***8-PLA(6,2,21)***-area is equal to $L_3 = 168$. Then we have

$$L(18,6,20) = 6L_1 + 2L_2 + L_3 = 568.$$

In that way we can greatly reduce the chip-area for realization the system of partial Boolean functions with the help of our results (new ***K***-valued gates and new method of synthesis).

Using our method we get for realization ***(m,r,q)*** the chip-area, which is equal to

$$L(m,r,q)=n*L(T(2/K))+L(K\text{-}PLA(n,s,q'))+s*L(T(K/2)), \qquad (2)$$

where $n =]m/log_2 K[$; $s =]r/log_2 K[$; $q' \leq q$ is the number of intermediate lines, which depends from the result of minimization of the system (n,s,q) of K-valued logical functions; $L(T(2/K)) = (2* log_2 K + 1)*K$; $L(T(K/2)) = (log_2 K + 1)*K.$

Finally, we get

$$L(m,r,q) = n*(2*log_2 K + 1)*K + (n+s)*q' + s*(log_2 K +1)*K=$$

$$= K*((2n +s)*log_2 K + n +s) + (n+s)*q'. \qquad (3)$$

We can propose the more comfortable formula for the estimation of the chip-area, which is needed for realization the system (m,r,q)

$$L(m,r,q) \leq K*((2n+ s)* log_2 K + n +s) + (n+s)* \qquad (4)$$

It is obvious the justice of inequality (4), if $q' \leq q.$

If we have $q' > q$ implicants as result of minimization of K-valued functions (n,s,q) , than for minimization of the chip-area we can ignore these results and make use of the technical possibility of ***K-PLA*** for realization the system of K-valued functions (n,s,q) by a circuit with one ***K-PLA(n,s,q).*** The example of that realization was showed on Fig.6. In this case $L(K\text{-}PLA(n,s,q)) = (n+s)*q$. Thus the justice of inequality (4) is proved in all possible cases.

With the help of (4) we have for $K =4$

$$L(m,r,q) \leq 4(2(2n + s) + n + s) + (n+s)q = n(20 + q) + s(12 + q), \qquad (5)$$

where $n =]m/2[$, $s =]r/2[.$

Then we have

$$L(18,6,20) = 4((18+ 3)*2 + 12) + 12*20 = 216 +240 = 456.$$

Thus the use of the new method of the synthesis allows considerably to reduce the chip-area, which is needed for realization the system of partial Boolean functions $(18,6,20)$.

If the use of traditional ***2***-valued gates allows to realize $(18,6,20)$ by a circuit, where the chip-area is equal to ***840***, to the use ***8***-valued gates allows to realize the right $(18,6,20)$ by the other circuit, where chip-area is equal to ***568***. We can reduce the chip-area considerably more, if we'll realize $(18,6,20)$ by a circuit with ***4***-valued gates. In this case the chip-area for realization of our partial Boolean functions $(18,6,20)$ is equal to ***456***.

On the certain level of the abstraction (without calculation of the difficulty of the solution of the technical tasks of realization K-valued elements) we can conclude, that the new method of synthesis, reported in this paper, have been applied for solution actual technical problems, for example chip-area minimization.

References

1. Zakrevskij A.D., *Logical Synthesis of the Cascade Circuits*. Moscow, Nauka, 1981 (in Russian).

2. Novikov S.V., *Theory of the Regular Structures*. Minsk, Universitetskoje, 1987 (in Russian).
3. Pyotr Bibilo, Natalia Kirienko, *Block synthesis of combinational circuits in the basis of PLA and library gates*. Proceedings of The International Workshop on Discrete-Event System Design. Zielona Góra, Technical University of Zielona Góra, 2001, s. 181-186.
4. Sergey Novikov, *Realization of Many-valued Logical Functions in PLA basis*. Proceedings of Artificial Intelligence Studies. Institute of computer science, University of Podlasie, vol. 2(25)2005, Siedlce, 2005, s. 217-224.
5. Sergiej Nowikow, *Metoda minimalizacji systemów K- znacznych częściowych funkcji logicznych*. Studia Informatica 2, Siedlce, AP, 2004, s. 157-164.
6. Sergey Novikov, *The Methods of Realization of K-Valued Logical Function's Implicants. Mathematica*. Proceedings of the XI Slovak-Czech-Polish Mathematical School. Catholik University in Ruzomberok, Ruzomberok, 2004, s. 51- 56.