УДК 004.3

Approach to Realization of Compositional Microprogramming Control Unit with Code Converter on Custom-Made Matrix

A. A .Barkalov[1], I.J. Zelenyova[2], A.N. Miroshkin [2], Hathot Biayrek[2]
[1]University of Zielona Gora, Poland
A.Barkalov@iie.uz.zgora.p1
[2]Donetsk National Technical University, Donetsk, Ukraine
miroshkin@cs.dgtu.donetsk.ua, allah_karem90@yahoo.com

Abstract

***Barkalov O., Zelenyova I.J., Miroshkin A.N., Hathot Biayrek. Approach to Realization of Compositional Microprogramming Control Unit with Code Converter on Custom-Made Matrix**. The method of compositional microprogramming control unit realization on custom matrix is proposed. Using of code converter allows reducing the area of custom-made VLSI in logic circuit of control unit. Analysis of proposed method and results of modeling for flow-charts of control algorithm is carried out.*

Introduction

The properties of the interpreted control algorithm have great influence on the hardware amount of corresponding control units. One of such properties is the existence of operational linear chains (OLC) corresponding to the graph-scheme of algorithm (GSA) paths, which include operator vertices only [1]. There are many of the approaches for linear graph-scheme of algorithm interpretation and one of them is the use of compositional microprogram control units (CMCU), which can be viewed as a composition of the finite state machine and microprogram control unit [2]. In this case control unit can be implemented as composition of automaton with "hardware" logic, realizing addressing of transitions, and automaton with "programmable" logic. The given article explains the method of CMCU synthesis with microinstructions code converter.

The purpose of the research is realization of logic circuit of control unit on custom-made VLSI as interpretation of linear control algorithm.

The task of the research is development of method for CMCU with microinstructions codes converter synthesis of microinstructions addresses that allows reducing the area of custom-made matrix for its logic circuit. The control algorithm is represented in the form of GSA.

Basic definitions and general provisions

Transformation of microinstruction addresses is oriented at the reduction of hardware amount in CMCU logic circuit. This method uses the transformation of microinstruction addresses of operational linear chains outputs into codes of the classes of pseudoequivalent OLC.

Let initial GSA have starting vertex b_0, final vertex b_E, the set of the operational vertices B_1 (the set of states of Moore automata), and conditional vertices set B_2. Vertices $b_q \in B_1$ contains microoperations $Y(b_q) \subseteq Y$, where $Y = \{y_1,....,y_N\}$ is the set of microoperations. In the vertices $b_q \in B_2$ elements from the set of logic conditions $X = \{x_1,....,x_N\}$ are contained. Vertices GSA are connected by arches $\langle b_t, b_q \rangle$, forming the set E. $B_j = \{B_0,...,B_{J-1}\}$ is the set of the classes of pseudoequivalent OLC [2]. Let us introduce some of definitions necessary for the further material explanation.

Definition 1: An operational linear chain of GSA is a finite vector of operator vertices $\alpha_g = \{b_{g(1)},...,b_{g(Fg)}\}$, such that an arch $\langle b_{g(i)}, b_{g(i+1)} \rangle \in E$ corresponds to each pair of adjacent vertices $b_{g(i)}$, $b_{g(i+1)}$, where i is the component number of vector α_g.

Definition 2: An operator vertex $b_q \in \alpha^g$ is called the input of OLC α_g, if there is an arch $\langle b_t, b_q \rangle \in E$, such that $b_t = b_0$ or $b_t \in B_2$ or $b_t \notin \alpha^g$.

Definition 3: An operator vertex $b_t \in D^g$ is called the output of OLC α_g, if there is an arch $\langle b_t, b_q \rangle \in E$, where $b_q = b_E$ or $b_q \in B_2$ or $b_q \notin \alpha^g$.

Any OLC α_g can have more than one input, let us designate j input OLC α_g like I_g^j, every OLC contain only one output O_g, entering in the set $O=\{O_1,...,O_G\}$ of OLC outputs.

The set of operator vertices of GSA is OLC $C=\{\alpha_1,....,\alpha_G\}$ satisfying the following conditions:

$$\begin{cases}\alpha^i\cap\alpha^j=\varnothing;\ \ (i\neq j;\ i,\ j\in[1..G]),\\ \alpha^1\cup\alpha^2\cup...\cup\alpha^G=B_1\end{cases} \qquad (1)$$

At performance of conditions (1) every vertex $b_q\in B_1$ enters exactly into one OLC $\alpha_g\in C$. A set of inputs $I(\Gamma)$ of the operational linear chains of GSA Γ is

$$I(\Gamma)=\bigcup_{g=1}^{G}I(\alpha_g). \qquad (2)$$

Also we have a set of outputs $O(\Gamma)$ of the operational linear chains of GSA:

$$O(\Gamma)=\bigcup_{g=1}^{G}O(\alpha_g). \qquad (3)$$

Let $A(b_q)$ be a binary address of microinstructions corresponding $b_q\in B_1$ includes

$$R_A=\lceil\log_2 M\rceil \qquad (4)$$

bits, where the total number of microinstructions is equal to $M=|B_1|$. The natural microinstruction addressing can be executed for microinstructions corresponding to the adjacent components of each OLC $\alpha_g\in C$:

$$A(b_{g(i+1)})=A(b_{g(i)})+1; \qquad (5)$$

Let us encode each class $B_i\in\Pi_C$ by a binary code $K(B_J)$ of

$$R_B=\lceil\log_2 I\rceil \qquad (6)$$

bits, using variables $\tau_r\in\tau$, where $|\tau|=R_B$, for encoding the classes $B_i\in\Pi_C$.

It is possible now the initial GSA be interpreted using the model of CMCU with microinstructions codes converter [2], designated in the further symbol U_1 (Fig.1), where an address converter AT converts output address of OLC $\alpha_g\in B_i$ into the codes of classes $B_i\in\Pi_C$.

In compositional microprogram control unit, combinational circuit CC implements the input memory functions

$$D=F(\tau,x). \qquad (7)$$

And address converter AT implements functions of the system

$$\tau=F(T). \qquad (8)$$

The CMCU with converter of microinstructions (Fig. 1) operates as follows:

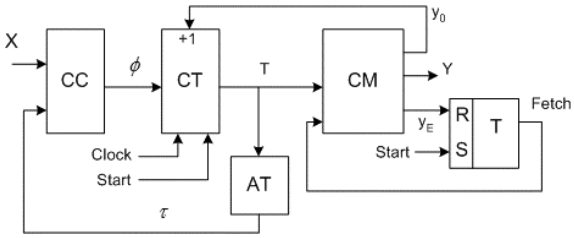

Figure 1 – Structural diagram of CMCU U_1

Pulse Start is used to load a zero address into the counter CT and to set up the flip-flop T (it gives $Fetch=1$). If microinstruction $Y(b_q)$, where $b_q\neq Q_g$ $g=[1..G_1]$, is read out of the CM, signal y_0 is generated and content of the counter is incremented, in order to address next microinstruction corresponds to next component of the current OLC. If $b_q=O_g$, a transition address is generated by combinational circuit CC using outputs of the address transformer AT and logical conditions. If microinstruction with $y_E=1$ is read out of the CM and flip-flop T is cleared and operation of CMCU terminated.

Method of synthesis of CMCU U includes the following stages:

1. Creating the transformation of initial flow-chart.
2. Addition of internal control microoperations y_0 and y_E.
3. Creation of the set $C=\{\alpha_1,....,\alpha_G\}$ of OLC.
4. Natural addressing of microinstructions.
5. Forming the content of control memory (CM).
6. Splitting the set C of OLC on classes of pseudoequivalent OLC: $B_j=\{B_0,...,B_{J-1}\}$.
7. Encoding of classes $B_j\in B$.
8. Construction the table of address converter AT.
9. Construction the table of code converter.
10. Creation the logical circuit of CMCU U in the given element basis.

Matrix realization of CMCU with microinstruction addresses converter

For implementation of the CMCU scheme on custom-made matrices we proposed to represent the CC circuit in the form of conjunctive matrix M_1 and disjunctive matrix M_2, CM is presented as matrix

M_3 and disjunctive matrix M_4 while the converter device is introduced as M_5 matrix. (Fig. 2):

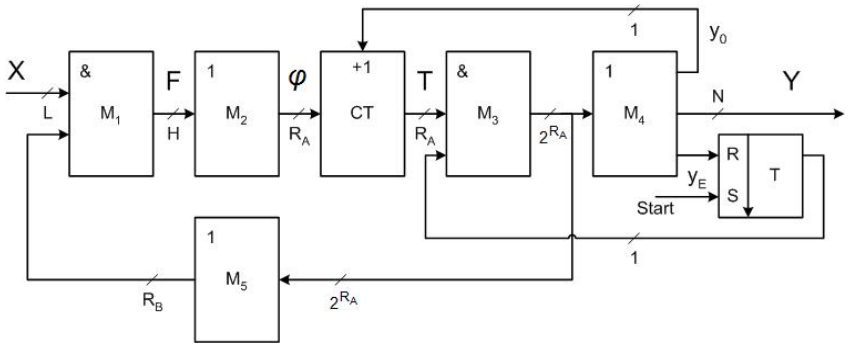

Figure 2 —Matrix realization of CMCU U_1

From Fig. 2 we can calculate S^{CC}_{CMCU}, where S^{CC}_{CMCU} is the area of CMCU U:

$$S^{CC}_{CMCU} = S_{M_1} + S_{M_2} + S_{M_3} + S_{M_4} + S_{M_5}; \quad (9)$$

$$S_{M_1} = 2*(L + R_B)*H; \quad (10)$$

$$S_{M_2} = H*R_A; \quad (11)$$

$$S_{M_3} = 2*(R_A + 1)*2^{R_A}; \quad (12)$$

$$S_{M_4} = 2^{R_A}*(N + 2); \quad (13)$$

$$S_{M_5} = 2^{R_A}*R_B. \quad (14)$$

Example of application of CMCU with converter of microinstructions

Let us consider the example of CMCU synthesis method application with using initial GSA (figure 3).

At first we add internal microoperations y_0 and y_E to the operational vertices and define the set of operational vertices $B_1 = \{b_1,...,b_6\}$; $M = 6$, $R_A = \lceil \log_2 M \rceil = 3$. Then we try to form the set of operator vertices of initial GSA OLC, $C = \{\alpha_1,...,\alpha_4\}$ from the set we can now determine $G = 4$, $\alpha_1 = \{b_1\}$, $\alpha_2 = \{b_2, b_3\}$, $\alpha_3 = \{b_4\}$, $\alpha_4 = \{b_5, b_6\}$.

Then we will create the partition $K(B_i)$ for our example, which contains blocks $\{B_0,...,B_2\}$, where $B_0 = \{\alpha_1\}$, $B_1 = \{\alpha_2, \alpha_3\}$, $B_2 = \{\alpha_4\}$.

So, for encoding of pseudoequivalent OLC classes $B_j \in \Pi_C$ it is enough $R_B = 2$ variables $\tau = \{\tau_1, \tau_2\}$. Let us encode classes $B_i \in \Pi_C$ in such way: $K(B_0) = 00, \ldots, K(B_2) = 10$.

Table of transitions CMCU has the following columns: B_i – class of pseudoequivalent OLC, $K(B_i)$ – code of class B_i, I_g^j, $A(I_g^j)$, X_h, Φ_h, h. Table of transitions for CMCU U_1 is given (Table 2).

We don't need to consider transition from OLC $\alpha_4 \in B_2$ as it goes to ending node. Applying of method of address converter is similar to applying of converter of states' codes in codes of classes of pseudoequivalent states of Moore automaton.

Table 1. Addressing of control memory words for CMCU

Address	Content						B_I	Π_C	I_g^i	O_g
$D_3D_2D_1$	y_0	y_1	y_2	y_3	y_4	y_E				
000	1	0	0	0	0	0	b0	-	-	-
001	0	1	0	1	0	0	b1	α_1	I_1	O_1
010	1	0	1	1	0	0	b2	α_2	I_2	-
011	0	1	0	0	1	0	b3	α_2	-	O_2
100	0	0	1	0	0	0	b4	α_3	I_3	O_3
101	1	0	0	1	0	0	b5	α_4	I_4^1	-
110	0	1	0	1	1	1	b6	α_4	I_4^2	O_4

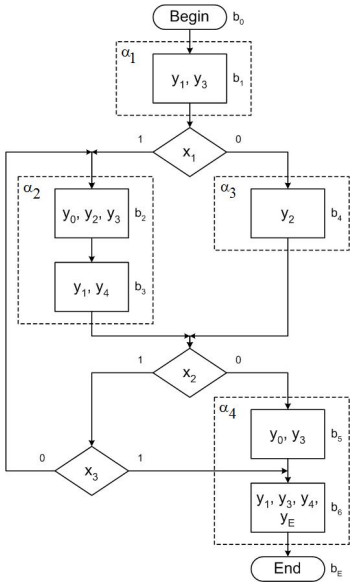

Figure 3 – Initial GSA for CMCU U_1

Table 2. Transitions for CMCU with code converter

B_i	$K(B_i)$ $\tau_1\ \tau_2$	I_g^j	$A(I_g^j)$	X_h	Φ_h	h
B_0	0 0	I_2	010	x_1	D_1	1
		I_3	100	$\overline{x_1}$	D_2	2
B_1	0 1	I_2	010	$x_2\overline{x_3}$	D_1	3
		I_4^1	101	$\overline{x_2}$	D_2, D_0	4
		I_4^2	110	x_2x_3	D_2, D_1	5

With the table we can build system function (2), for example:

$$D_1 = F_1 \vee F_3 \vee F_5 = \overline{\tau_1}\overline{\tau_2}x_1 \vee \overline{\tau_1}\tau_2 x_2 \overline{x_3} \vee \overline{\tau_1}\tau_2 x_2 x_3 . \quad (15)$$

Table of the converter of addresses of microinstructions contains columns: O_g, $A(O_g)$, B_i, $K(B_i)$, τ_j, g. If $\alpha_g \in B_j$, then row g of table contains corresponded information. For our example address converter is given in table 3:

Table 3 – Table of code converter

O_g	$A(O_g)$	B_i	$K(B_i)$	τ_j	g
O_1	001	B_0	0 0	-	1
O_2	011	B_1	0 1	τ_1	2
O_3	100	B_1	0 1	τ_1	3
O_4	110	B_2	1 0	τ_2	4

With the help of this table we can form system of functions

$$\tau_r = \bigvee_{g=1}^{G} C_{rg} A_g, (r = 1,..., R_0), \quad (16)$$

where C_{rg} is boolean variable. It is equal to "1", if and only if $\alpha_g \in B_i$ and bit r of $K(B_i)$ is equal to "1", A_g is a conjunction of variables $Q_r \in Q$, which corresponds to address $A(O_g)$ of OLC output $\alpha_g \in B_i$.

Results of the modelling

Dependence custom-made matrix area on parameters of initial GSA is explored using expressions (9)-(14) and statistics of EXCEL (fig.4).

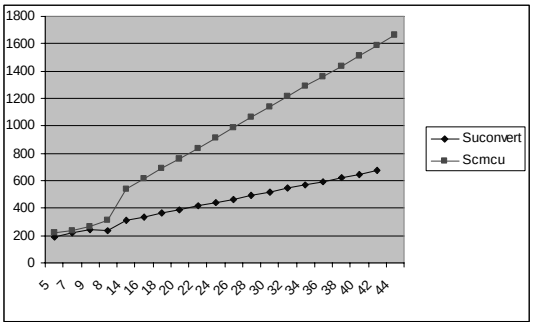

Figure 4 –Dependence matrix area on number of transitions

Results of research shows that effectiveness of code converter using depends most of all on possibility of classes forming and transition number reducing .

Conclusion

The proposed CMCU synthesis method targets to decrease in hardware amount, used area of the custom-made matrices. The method is based on encoding of the classes of pseudoequivalent OLC permitting decrease of the transition table lines in comparison with CMCU with base structure.

Results of experiments show that the area of matrices is decreased up to 15%. Of course, application of proposed method is possible only for interpretation of linear GSA.

The next step in research is exploration of possibility for given method application in case of standard elementary basis (CPLD, FPGA).

References

1. Barkalov A.A., Titarenko L.A. Synthesis Of Operational And Control Automata. – Donetsk: Unitech, 2005. – 256 pp.
2. Barkalov A.A., Titarenko L.A. Logic Synthesis for Compositional Microprogram Control Units,2008. – 272 pp.
3. Smith M. Application – Specific Integrated Circuits. – Boston: Addison Wesley, 1997. – 836 pp.
4. Baranov S. Logic Synthesis For Control Automata. – Boston: Kluwer Academic Publisher, 1954. – 312 pp.

Received 29.03.2010