Remigiusz WIŚNIEWSKI, Alexander BARKALOV, Anna JANIAK
UNIWERSYTET ZIELONOGÓRSKI

Methods of designing of compositional microprogram control units with mutual memory

Mgr inż. Remigiusz WIŚNIEWSKI

Mgr inż. Remigiusz Wiśniewski jest absolwentem Uniwersytetu Zielonogórskiego (2003). Ukończył studia o specjalności Inżynieria Komputerowa. W roku 2000 odbył przemysłową praktykę studencką w firmie Aldec Inc. w Stanach Zjednoczonych. Od roku 2003 pracuje jako asystent na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego.

e-mail: R.Wisniewski@iie.uz.zgora.pl

Prof. dr hab. inż. Alexander A. BARKALOV

W latach 1976-1996 był pracownikiem dydaktycznym w Instytucie Informatyki Narodowej Politechniki Donieckiej. Współpracował aktywno z Instytutem Cybernetyki im. V.M. Glushkova w Kijowie, gdzie uzyskał tytuł doktora habilitowanego ze specjalnością informatyka. W latach 1996-2003 pracował jako profesor w Instytucie Informatyki Narodowej Politechniki Donieckiej. Od 2004 pracuje jako profesor na Wydziale Elektrotechniki, Informatykii Telekomunikacji Uniwersytetu Zielonogórskiego.

e-mail: a.barkalov@iie.uz.zgora.pl

Abstract

In the article four designing methods of Compositional Microprogram Control Unit (CMCU) will be described and compared. The first one - with mutual memory - is traditional way of synthesis of CMCU. Here operational vertices of the initial flow chart that describes the functionality of control units are replaced with operational linear chains that permit to minimize the number of internal states of the controller. Three remaining methods are based on the CMCU with mutual memory; however there are additional improvements that allow reducing the number of logic elements that are required for implementation of CMCU on programmable device. Detailed results of investigations will be shown in the paper. Authors have performed researches where over 100 benchmarks (descriptions of CMCU) were designed with all four methods and implemented on an FPGA. Results of implementation will be studied and analyzed in detail and described in the paper.

Keywords: Compositional Microprogram Control Unit (CMCU), Field-Programmable Gate Arrays (FPGAs).

Metody projektowania mikroprogramowanych jednostek sterujących o adresowaniu wspólnym

Streszczenie

W referacie zaprezentowane zostaną cztery metody projektowania mikroprogramowanych jednostek sterujących. Pierwsza metoda to tradycyjny sposób syntezy sterownika o adresowaniu wspólnym. Na jej podstawie opracowane zostały trzy inne metody projektowania mikroprogramowanych układów sterujących. Wprowadzono modyfikacje w strukturze sterownika, których głównym celem była redukcja liczby wykorzystanych elementów logicznych podczas implementacji systemu w matrycach FPGA. W artykule przedstawione zostaną szczegółowe wyniki badań przeprowadzonych przez autorów. Każdy sterownik zaprojektowano wszystkimi czterema metodami, a następnie przeprowadzono operacje syntezy oraz implementacji. Końcowe wyniki zajętości poszczególnych wersji w programowalnych matrycach FPGA zostaną szczegółowo przeanalizowane.

Słowa kluczowe: mikroprogramowany układ sterujący, programowalny układ FPGA.

Mgr inż. Anna JANIAK

Ukończyła studia magisterskie na Uniwersytecie Zielonogórskim. Pracę magisterską obroniła w 2005 r. Od 2005 r. jest słuchaczem studiów trzeciego stopnia (doktoranckich) prowadzonych w Uniwersytecie Zielonogórskim.

e-mail: A.Janiak@wh.uz.zgora.pl

1. Introduction

A control unit (CU) is one of the main parts of any digital system [1, 3, 4, 5]. Traditional design methods of digital systems implement CU as a finite-state-machine (FSM). Such a solution very often consumes many logic blocks of the device which can be more effectively used by other parts of the prototyped system. Therefore compositional microprogram control units (CMCUs) can be more effective [2, 3]. In a CMCU the control unit is decomposed into two main parts. The first is responsible for addressing of microinstructions that are kept in the control memory. It is a simple finite-state-machine. The role of the second part is to hold and generate adequate microinstructions. Such a solution permits to minimize the number of logic elements that are used to implement the CU. Therefore, wider areas of the target device can be accessed by other modules of the designed system. The memory of the prototyped control unit can be implemented using either logic elements or dedicated memory blocks of a chip. The rest of the system is realized by the logic blocks of the FPGA [2, 3]. All logic functions are performed by the Look-Up Tables (LUTs) which have limited amount of inputs [2, 3]. Therefore very often the Boolean functions of the design ought to be decomposed [4, 5]. Such a process very often consumes additional area of the destination FPGA.

In the paper four designing methods of CMCUs are shown. The first one – with mutual memory - is traditional way of synthesis of CMCUs. The structure of CMCU with mutual memory was an inspiration for three remaining ways of CMCU designing. The main aim was to reduce the number of logic elements that are required in order to implement the controller on an FPGA. The aim of this article is to compare the effectiveness of presented methods. Detailed designing process of each CMCU was described in literature [3, 6, 7, 8] thus it will be omitted and assumed as already known.

2. Main definitions and current state of the art

Let a control algorithm of a digital system is represented as a flow-chart Γ [2] with a set of operational vertices $B=\{b_1,\ldots,b_K\}$ and a set of the arcs E. Each vertex $b_k \in B$ contains microoperations $Y(b_k) \subseteq Y$, where $Y=\{y_1, \ldots y_N\}$ is a set of microoperations. Each conditional vertex of the flow-chart contains one element of the set of logic conditions $X=\{x_1,\ldots,x_L\}$. Any flow-chart includes initial vertex b_0 and final vertex b_E.

Definition 1

The operational linear chain (OLC) of the flow-chart Γ is a finite sequence of operational vertices $\alpha_g=<b_{g1},\ldots,b_{gFg}>$ such that for any pair of adjacent components of the vector α_g there is

an arc $<b_{gi}, b_{gi+1}> \in E$, where i is a component number of the vector $\alpha_g(i=1, \ldots F_g\text{-}1)$.

Definition 2

A vertex $b_q \in B$ is named as an *input* of the OLC α_g if there is an arc $<b_t, b_q> \in E$, here b_t is a vertex that does not belong to the OLC α_g.

Definition 3

A vertex $b_q \in B$ is named an *output* of OLC α_g if there is an arc $<b_q, b_t> \in E$, where b_t is a vertex that does not belong to OLC α_g.

Let D_g is a set of operational vertices that are included in the chain α_g. Let $C=\{\alpha_1, \ldots, \alpha_G\}$ is the set of OLCs of flow-chart Γ satisfied to condition:

$$\begin{aligned} &D^g \cap D^q = \emptyset\,(g \neq q; g, q \in \{1,\ldots,G\});\\ &B = D^1 \cup D^2 \cup \ldots \cup D^G;\\ &D^g \neq \emptyset (g = 1,\ldots,G). \end{aligned} \tag{1}$$

Let natural addressing of microinstructions [2, 3] is executed for each OLC $\alpha_g \in C$:

$$A(b_{gi+1}) = A(b_{gi}) + 1 (i = 1,\ldots,F_{g-1}), \tag{2}$$

where $A(b_g)$ is an address of microinstruction corresponding to the vertex $b_g \in B$. In this case flow-chart Γ can be interpreted by CMCU U_{MM} with mutual memory (Fig. 1).

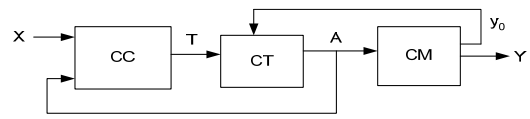

Fig. 1. Compositional microprogram control unit with mutual memory
Rys. 1. Mikroprogramowany układ sterujący o adresowaniu wspólnym

There are three main modules in the control unit presented in the fig. 1: combinational circuit CC, counter CT and control memory CM. The combinational circuit CC is responsible for generating excitation functions for the counter CT. The CT keeps an address of microinstructions and variables $A_r \in A$ are used for representation of addresses $A(b_k)$, $b_k \in B$. Microinstructions are kept in the CM and each word (microinstruction) has N+2 bits in the case of unitary encoding of microoperations [2]. One of the additional bits is used to keep a variable y_0 to organize the mode of addressing (2). The second additional bit keeps a variable y_K to organize the fetching of microinstructions from the CM. At the beginning the CT is set to the value that corresponds to an address of the first microinstruction. If the transition is executed inside OLC $\alpha_g \in C$, then y_0=1 which causes increment of the CT. When output of OLC $\alpha_g \in C$ is reached, y_0=0 and the circuit CC forms excitation functions for the counter:

$$T = f(X,A), \tag{3}$$

where X is a set of conditional vertices, A is an address of microinstruction. If the CT contains an address of microinstruction $Y(b_k)$ such as $<b_k, b_E> \in E$, then y_K=1. In this case operation of CMCU is finished.

Such an organization of the control unit permits to use only minimal amount of microinstructions and each microinstruction does not contain information for calculation of an address of the transition. Thus it permits to minimize the size of the control memory in comparison with any microprogram control unit [2, 3]. The investigations have shown that CMCU U_{MM} is always better than traditional FSM in case of systems, where the total number of vertices in the initial flow-chart exceeds 70-80 (outputs of the FSM were also implemented with memory blocks). However in case of smaller controllers, the CMCU became less effective and often it consumes wider areas of an FPGA than traditional automata. To eliminate this problem and to improve the functionality of the CMCU three methods of the CMCU designing were proposed in [6, 7, 8] where the traditional way of synthesis of the controller was modified. Next sections deal with brief description and main ideas of proposed methods.

3. CMCU with function decoder

The CMCU with function decoder U_{FD} is an extended structure of the CMCU with mutual memory. In comparison to the U_{MM} there is an additional block (function decoder) introduced. Figure 2 illustrates the CMCU with function decoder [7].

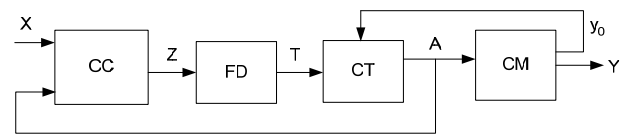

Fig. 2. Compositional microprogram control unit with function decoder
Rys. 2. Mikroprogramowany układ sterujący z dekoderem funkcji

The main idea of the method is to reduce the number of logic blocks of the destination FPGA due to the usage of an additional block (function decoder) which is realized using dedicated memories. Therefore less LUT elements are used during implementation of control unit in comparison with CMCUs with mutual memory [7].

In the CMCU U_{FD} the set of excitation functions for counter is encoded with the minimum number of bits. To reach it all inputs of operational linear chains ought to be encoded. Therefore circuit CC generates the set of functions Z:

$$Z = f(X,A). \tag{4}$$

Functions Z contain encoded addresses A of all inputs I in the set of OLCs. They are further decoded by the circuit FD which indicates the proper code for the counter:

$$T = f(Z). \tag{5}$$

Presented solution permits to reduce the number of outputs and excitation functions generated by the circuit CC. An additional block of function decoder is implemented with dedicated memories of FPGAs. Therefore the number of logic elements that are needed to implement whole controller is reduced. As it was mentioned designing method of CMCU with function decoder was described in [7] and will not be presented in this paper.

4. CMCU with outputs identification

The structure of the CMCU U_{OI} with outputs identification is illustrated in the fig. 3. The main idea is to use the part of an address A for the identification of operational linear chains. Now the set of variables Q ($Q \subset A$) keeps a code of the current state of the controller.

In the CMCU U_{OI} the set of feedback variables Q that are used for the identification of the current state of the controller are reduced to the minimum.

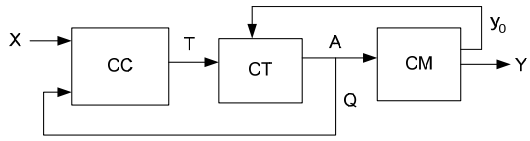

Fig. 3. Compositional microprogram control unit with outputs identification
Rys. 3. Mikroprogramowany układ sterujący z identyfikacją wyjść

Outputs of the operational linear chains may be recognized using R_{OI} bits thanks to the special encoding of microinstruction [6, 8]. Therefore now the combinational circuit generates the set of functions T for the counter:

$$T = f(X,Q). \tag{6}$$

The main advantage of presented solution is reduction of the number of feedback functions that keep the actual code of state of the controller. Thus the number of logic functions is reduced in comparison with traditional CMCU with mutual memory. It should be pointed out that special addressing of microinstruction is required [6]. Therefore additional designing step has to be performed in comparison with CMCU U_{MM}.

5. CMCU with outputs identification and function decoder

The last control unit presented in this paper describes the CMCU UOF with outputs identification and function decoder. Such a controller is a conjunction of two structures presented in previous sections. There is a special addressing of microinstruction used in CMCU UOF. Moreover maximal encoding of the set of excitation functions for counter is performed as well.

To improve the minimization of the LUT elements of the implementation of CMCUs UFD and UOI both methods may be combined. Now the combinational circuit generates the set Z of excitation functions for the circuit FD and the function decoder generates the proper address of the microinstruction:

$$Z = f(X,Q), \tag{7}$$

$$T = f(Z). \tag{8}$$

The CMCU with outputs identification and function decoder permits to implement both the FD and the CM with dedicated memory blocks. Moreover thanks to the outputs identification the number of feedback functions for the combinational circuit decreases. Therefore implementation of the CMCU UOF consumes the least logic elements of programmable devices in comparison with CMCUs UMM, UFD and UOI. However presented controller uses at least two dedicated memory blocks of an FPGA.

6. Results of the investigations

To compare the effectiveness of presented methods, detailed investigations were performed. Benchmarks used for tests were prepared as descriptions of control units in a common project of University of Zielona Gora and Donetsk National Technical University. Each benchmark describes CMCU as a flow-chart where exists at least one operational linear chain that contains two or more operational vertices. Additionally for each benchmark traditional FSM was also prepared – here microinstructions were also implemented with dedicated memory blocks. Table 1 shows results of performed investigations. There are most important benchmarks presented. In brackets there is also the number of used dedicated memory blocks shown. As the target Xilinx xc3s500e (Spartan 3E family) device was used.

Tab. 1. Results of performed investigations
Tab. 1. Wyniki przeprowadzonych badań

Name of the benchmark	No of vertices	No of OLCs	Total LUTs (and BRAMs) usage				
			FSM	U_{MM}	U_{FD}	U_{OI}	U_{OIFD}
MK_02	99	15	91 (1)	85 (1)	53 (2)	54 (1)	43 (2)
MK_08	56	12	39 (1)	44 (1)	34 (2)	41 (1)	28 (2)
MK_12	82	23	100 (1)	96 (1)	67 (2)	64 (1)	49 (2)
MK_17	81	19	81 (1)	88 (1)	57 (2)	54 (1)	48 (2)
MK_15	76	18	49 (1)	71 (1)	50 (2)	48 (1)	40 (2)
Test025	218	54	642 (1)	283 (1)	229 (2)	309 (1)	224 (2)
Test027	135	29	361 (5)	176 (5)	138 (6)	181 (5)	142 (6)

From presented results we can see that in case of small controllers (MK_08, MK_17) the area needed for FSMs implementation is smaller than CMCUs with mutual memory. Moreover FSMs may consume less logic than CMCUs with function decoder or CMCUs with outputs identification. In all cases CMCUs that combines output identification and function decoder required the least resources of an FPGA. In case of bigger designs where the number of vertices in the flow-chart exceeds 100 vertices all four CMCUs required much less LUTs than traditional FSM.

7. Conclusions

In the paper four designing methods of compositional microprogram control units were shown and compared. Performed investigations have shown that either application of the function decoder and reduction of the feedback function always reduced the area used for implementation of the CMCU on an FPGA. Thus the best results of the researches were achieved in case conjunction of those two ideas. CMCU with outputs identification and function decoder is always better compared to the traditional CMCU with mutual memory. It should be also pointed out that such a CMCU almost always consumes less logic blocks than controller realized as a traditional FSM.

8. References

[1] Baranov S.: Logic Synthesis for Control Automata, Kluwer Academic Publishers, 1994.
[2] Barkalov A.A., Palagin A.V.: Synthesis of Microprogram Control Units, IC NAS of Ukraine, Kiev, Ukraine, 1997.
[3] Adamski M., Barkalov A.A.: Architectural and sequential synthesis of digital devices, University of Zielona Góra Press, 2006.
[4] DeMicheli G.: Synthesis and Optimization of Digital Circuits, McGraw Hill, New York, 1994.
[5] Łuba T (Praca zbiorowa pod redakcją prof. Tadeusza Łuby): Synteza układów cyfrowych, WKŁ, Warszawa, 2003.
[6] Wiśniewski R., Barkalov A., Titarenko L.: Optimization of address circuit of CMCU, EWDTW '06. Kharkov, 2006, pp. 167-170
[7] Wiśniewski R., Barkalov A.: Synthesis of CMCUs with function decoder, IWCIT 2007:VSB - Technical University of Ostrava, 2007, pp. 229-232.
[8] Wiśniewski R., Barkalov A., Titarenko L.: Synthesis of compositional microprogram control units with OLC output identification, CAD DD '07 : proceedings of the 6th International Conference, Minsk, Bielarus, 2007 .- Minsk, 2007 .- T. 2, pp. 81-86.

Artykuł recenzowany