КІБЕРНЕТИКА та КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ

УДК 004.274 DOI:10.34229/2707-451X.25.2.8

О.О. БАРКАЛОВ, Л.О. ТІТАРЕНКО, О.М. ГОЛОВІН, О.В. МАТВІЄНКО

ПЕРЕТВОРЕННЯ АДРЕС У КОМПОЗИЦІЙНОМУ МІКРОПРОГРАМНОМУ ПРИСТРОЇ УПРАВЛІННЯ

Запропоновано спосіб зменшення кількості елементів LUT у перетворювачі адрес мікрокоманд. Метод застосовується при реалізації схем композиційних мікропрограмних пристроїв управління у базисі FPGA. Розглядається випадок реалізації схеми композиційного мікропрограмного пристрою управління з використанням елементів LUT та вбудованих блоків пам'яті EMB. Оптимізація досягається з допомогою надмірності EMB по виходам. Наведено приклад синтезу композиційного мікропрограмного пристрою управління із застосуванням запропонованого методу. Показано умови застосування цього методу.

Ключові слова: *композиційний мікропрограмний пристрій управління, LUT, EMB, операторні лінійні ланцюги.*

Вступ. Як правило, цифрові системи складаються з комбінаційних та послідовних блоків [1]. До найважливіших послідовних блоків відносяться пристрої управління [2]. Типові схеми пристроїв управління не належать до бібліотечних компонентів системи автоматичного проєктування. Тому реалізація схеми пристрою управління – більш трудомісткий процес, ніж реалізація таких комбінаційних блоків, як, наприклад, суматори і комутатори. При реалізації цифрових систем виникають проблеми оптимізації їх характеристик. У цій роботі розглядається проблема зменшення апаратурних витрат у схемі композиційного мікропрограмного пристрою управління (КМПУ) [3]. Як елементний базис використовуються ресурси мікросхем FPGA (field-programmable logic array) [4].

Вибір цього елементного базису обумовлений широким застосуванням FPGA для реалізації цифрових систем [5]. Наприклад, у роботі [2] наведено понад 1500 різних проєктів, реалізованих на FPGA. На думку експертів, мікросхеми FPGA домінуватимуть на ринку програмованих користувачем НВІС ще кілька десятиліть [6].

Пропонований метод ґрунтується на адаптації методів оптимізації схем мікропрограмних автоматів [1, 7] до особливостей КМПУ. Метод спрямований на перетворення адрес деяких мікрокоманд у часткові входи. При виконанні певних умов такий підхід дозволяє значно спростити схему адресації мікрокоманд. Такий підхід дає змогу покращити характеристики схеми КМПУ порівняно з іншими відомими методами. Для завдання алгоритму функціонування КМПУ використано мову граф-схем алгоритмів [7].

Аналіз особливостей КМПУ та елементного базису. КМПУ призначений для реалізації лінійних алгоритмів управління. Для лінійних граф-схем алгоритмів операторні вершини становлять 75 % вершин [8]. Методи синтезу КМПУ [3] ґрунтуються на формуванні операторних лінійних ланцюгів. Кожний операторний лінійний ланцюг – це вектор операторних вершин, пов'язаних безумовними переходами. Якщо $B = \{b_1, ..., b_M\}$ – множина операторних

вершин, то кожний операторний лінійний ланцюг визначає один клас розбиття цієї множини. Таким чином, процес синтезу КМПУ пов'язаний із знаходженням розбиття, $C = \{\alpha_1, ..., \alpha_G\}$, де α_g – це операторний лінійний ланцюг. Методи знаходження розбиття C розглянуті, наприклад, у роботі [3].

Схема КМПУ включає блок управляючої пам'яті. Основним завданням КМПУ є генерація мікрооперацій $Y = \{y_1, ..., y_N\}$ під впливом логічних умов $x_l \in X = \{x_1, ..., x_L\}$. Кожна операторна вершина $b_m \in B$ містить набір мікрооперацій $Y(b_m) \subseteq Y$. Кожна вершина $b_m \in B$ відповідає мікрокоманді Y_m, що зберігається в управляючій пам'яті за адресою. Розрядність адреси R визначається за формулою

$$R = \lceil \log_2 M \rceil. \tag{1}$$

У межах кожного операторного лінійного ланцюга виконується природна адресація мікрокоманд [8]. Якщо операторний лінійний ланцюг $\alpha_g \in C$ включає пару вершин $< b_q, b_t >$, то між адресами відповідних мікрокоманд існує залежність:

$$A(b_t) = A(b_q) + 1. \tag{2}$$

Якщо вершини $b_q, b_t \in B$ знаходяться у різних операторних лінійних ланцюгах, то між їх адресами може бути наступна залежність:

$$A(b_t) = f(A(b_q), X). \tag{3}$$

Для реалізації режиму (2) використовується лічильник адреси. Збільшення вмісту лічильника адреси відбувається за певним фронтом сигналу Clock за наявності спеціального сигналу $y_0 = 1$. Для реалізації режиму (3) використовуються функції збудження пам'яті $D_r \in D = \{D_1, ..., D_R\}$. Зміна вмісту лічильника адреси відбувається за сигналом $y_0 = 0$. Виходи лічильника адреси утворюють множину $T = \{T_1, ..., T_R\}$ адресних змінних пам'яті. Для реалізації режиму (3) необхідна схема адресації мікрокоманд, що базується на наступній системі булевих функцій:

$$D = D(T, X). \tag{4}$$

Для початку функціонування КМПУ використовується сигнал Start. Функціонування припиняється за сигналом y_E. Цей сигнал блокує надходження імпульсів Clock. На рис.1 показана структурна схема КМПУ U_1.

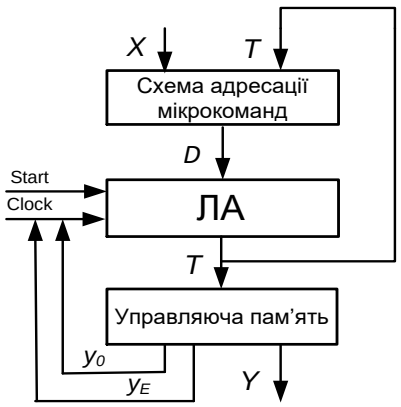


РИС. 1. Структурна схема КМПУ U_1

КМПУ U_1 називається КМПУ із загальною пам'яттю [3]. Принцип функціонування КМПУ U_1 зрозумілий з попереднього матеріалу.

У цій статті ми розглядаємо випадок реалізації схеми КМПУ під час використання програмованих ресурсів FPGA. До цих ресурсів належать [9].

1. Елементи табличного типу (LUT, look-up table).
2. Програмовані тригера пов'язані з виходами елементів LUT.
3. Вбудовані блоки пам'яті EMB (embedded memory blocks).
4. Матриця програмованих міжз'єднань.

Елемент LUT має S_L входів та один вихід. Елемент LUT може реалізувати довільну логічну функцію до аргументів. Число входів LUT обмежене і, як правило, $S_L \le 6$ [10]. Ми використовуємо елементи LUT реалізації комбінаційної частини КМПУ.

Програмованість тригера полягає у тому, що тип його входів може задаватися користувачем [10]. Як правило, лічильники мають інформаційні входи типу D [11]. Саме такі тригери ми використовуємо при реалізації схеми лічильника адреси.

Блок EMB має S_A адресних входів, t_F виходів та V_0 бітів, де

$$V_0 = 2^{S_A} \cdot t_F . \tag{5}$$

Параметри S_A, t_F можуть змінюватись при незмінності ємності V_0. Кожна пара $< S_A, t_F >$ визначає конфігурацію EMB. Для сучасних EMB характерні такі конфігурації [4]: $< 15,1>$, $< 14,2>$, $< 13,4>$, $< 12,8>$, $< 11,16>$, $< 10,32>$, $< 9,64>$. Ми використовуємо EMB для реалізації схеми управляючої пам'яті.

Програмовані міжз'єднання дозволяють реалізувати зв'язок між елементами КМПУ. При необхідності деякі елементи схеми можуть бути пов'язані з програмованими входами-виходами мікросхеми [12].

Основні проблеми, пов'язані з КМПУ U_1, виникають у разі перевищення числа аргументів у функціях (4) числа входів елемента LUT. У цьому випадку схема адресації мікрокоманд має декілька рівнів елементів LUT. Це призводить до наступного:

1) логічні умови $x_l \in X$ виникають на різних рівнях схеми адресації. Це ускладнює систему міжз'єднань. Процес трасування схеми також ускладнюється;

2) тригери лічильника можуть бути пов'язані з елементами LUT різних рівнів схеми. Ускладнюється з'єднання дерева синхронізації з входами тригерів. Чим більша загальна довжина міжз'єднань імпульсів Clock із входами синхронізації, тим більше енергії споживає схема [13];

3) збільшення довжини міжз'єднань призводить до збільшення площі кристала, яку займає схема КМПУ. Як відомо [14], площа схеми значною мірою визначає швидкодію та споживану потужність.

Усі ці негативні явища пов'язані з виконанням умови

$$NA(D_r) > S_L . \tag{6}$$

В умові (6) символ $NA(D_r)$ позначає кількість аргументів (літералів) у диз'юнктивній нормальній формі функції $D_r \in D$.

При реалізації схеми КМПУ у базисі FPGA схема адресації представляється блоком LUTerT (блок з елементів LUT). Цей блок включає лічильник адреси. Управляюча пам'ять реалізується блоком EMBerY (цей блок складається з елементів пам'яті EMB). Будемо використовувати позначення F-КМПУ, щоб показати, що схема КМПУ реалізується в базисі FPGA. Структурна схема F-КМПУ U_1 показана на рис. 2.

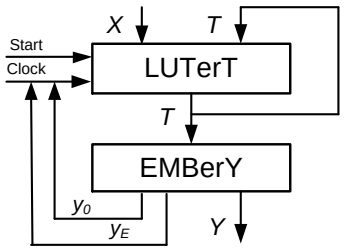


РИС. 2. Структурна схема F-КМПУ U_1

У цій роботі пропонується метод оптимізації блоку LUTerT під час виконання умови (6). Цей метод спрямовано на поліпшення характеристик схеми, запропонованої у роботі [15].

Основна ідея запропонованого методу. Нехай для граф-схеми алгоритмів Г знайдено розбиття $C = \{\alpha_1, ..., \alpha_G\}$. Остання вершина з операторного лінійного ланцюга $\alpha_g \in C$ – це її вихід. Побудуємо множину $O(\Gamma)$ виходів операторного лінійного ланцюга: $O(\Gamma) = \{o_1, ..., o_G\}$. Кожен вихід $o_g \in O(\Gamma)$ характеризується множиною $X_g \subseteq X$. Ця множина складається з логічних умов, що визначають переходи з вершин b_g, з мікрокоманди Y_g.

Якщо M_k виходів об'єднуються в одну множину O^k, то відповідні ним множини об'єднуються в множину X^k, що має L_k елементів. Закодуємо кожен із виходів кодом $K(o_g)$, що має R_k розрядів:

$$R_k = \lceil \log_2(M_k + 1) \rceil . \tag{7}$$

Одиниця додається до M_k, щоб врахувати той факт, що деякі виходи не належать O^k.

Множина O^k є класом сумісних виходів, якщо виконується умова

$$R_k + L_k \le S_L . \tag{8}$$

Нехай елементи множини τ^k використовуються для кодування виходів $o_g \in O^k$. Коди $K(o_g)$ – часткові, оскільки вони кодують лише частину виходів. Ці коди та логічні умови $x_l \in X^k$ визначають систему часткових функцій збудження пам'яті

$$D^k = D^k(\tau^k, x^k) . \tag{9}$$

За умови (8), кожна функція $D_r^k \in D^k$ генерується схемою, що складається з одного елемента LUT. Ці елементи поєднуються в блок LUTerk, що має від одного до R елементів LUT.

Для формування повних функцій необхідно реалізувати диз'юнкції відповідних часткових функцій:

$$D_r = D_r^1 \vee D_r^2 \vee ... \vee D_r^K \ \ (r = \overline{1,R}) . \tag{10}$$

Формування диз'юнктивної нормальної форми (10) викликає необхідність використання елементів LUT. Якщо виконується умова $K \le R$, відповідна схема має R елементів.

Формування змінних $\tau_r \in \tau$ потребує знаходження системи булевих функцій

$$\tau = \tau(T) . \tag{11}$$

При виконанні умови $R \le S_L$ відповідна схема складається з R_O елементів, де

$$R_O = R_1 + R_2 + ... + R_K . \tag{12}$$

На основі системи булевих функцій (9), (10) та (12) можна сформувати наступну схему адресації мікрокоманд рис. 3. Ця схема запропонована у роботі [15].

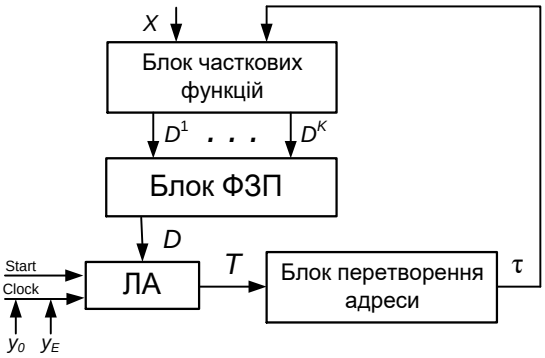


РИС. 3. Схема адресації мікрокоманд

Такий підхід має значний недолік: деякі ресурси мікросхеми використовуються для реалізації схеми перетворення адреси мікрокаманд. Ця схема реалізується на основі системи булевих функцій (12). У цій роботі пропонується метод зменшення апаратурних витрат у схемі перетворення адреси. Метод ґрунтується на використанні надмірності блоків EMB.

Для реалізації управляючій пам'яті необхідно вибрати конфігурацію $< S_A, t_F >$, для якої виконується умова $S_A = R$. Наш метод застосовується, якщо виконується умова

$$t_F \ge N + 2 . \tag{13}$$

Умова (13) означає, що управляюча пам'ять реалізується з використанням одного блоку EMB. При цьому число $N + 2$ визначає розрядність мікрокоманд (N мікрооперацій, сигнали y_O та y_E).

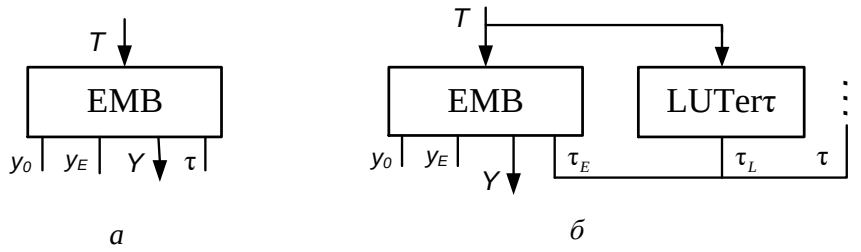
Знайдемо різницю

$$D_F = t_F - (N + 2) . \tag{14}$$

При виконанні умови

$$\Delta_F \ge R_O , \tag{15}$$

блок управляюча пам'ять формує всі функції системи (11). Це призводить до схеми, показаної на рис. 4,*а*.



а *б*

РИС. 4. Реалізація перетворювача адреси

Виконання умови (15) визначає тривіальний варіант, за якого немає необхідності в елементах LUT для реалізації системи булевих функцій (11). Якщо умова (15) порушується, ми пропонуємо наступний підхід:

1. Множина τ розбивається на множини τ_E і τ_L $(\tau_E \cap \tau_L = \varnothing)$.
2. Функції $\tau_r \in \tau_E$ генеруються блоком EMB.
3. Функції $\tau_r \in \tau_L$ представляються у вигляді системи булевих функцій

$$\tau_L = \tau_L(T). \tag{16}$$

4. Система (16) реалізується елементами блоку LUTerτ.

Ця методика призводить до схеми, показаної на рис. 4,*б*. Ґрунтуючись на цьому підході, ми пропонуємо структурну схему КМПУ U_2 (рис. 5).

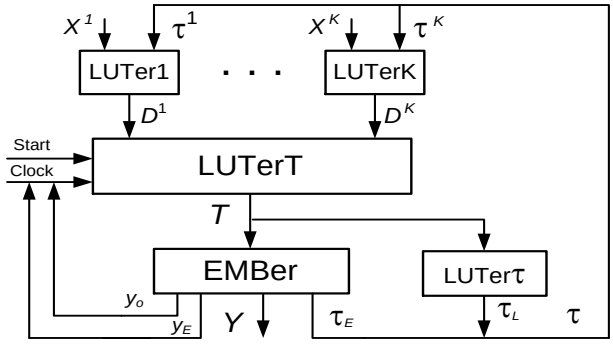


РИС. 5. Структурна схема КМПУ U_2

Блоки КМПУ U_2 виконують наступні функції: блоки LUTer1-LUTerK реалізують часткові функції збудження пам'яті (9); блок LUTerT реалізує систему булевих функцій (10), цей блок включає прихований лічильник адреси, тому виходами LUTerT є розряди адреси мікрокоманди; роботу блоку EMB визначає таблиця істинності функцій Y, τ_E, y_O та y_E; блок LUTerτ реалізує систему булевих функцій (16). Виходи τ_E і τ_L об'єднуються в магістраль, яка є зворотним зв'язком для блоків LUTer1-LUTerK.

Ми пропонуємо наступний метод синтезу КМПУ U2:

1. Формування множини операторних лінійних ланцюгів $C = \{\alpha_1, ..., \alpha_G\}$.
2. Природна адресація мікрокоманд у межах кожного операторного лінійного ланцюга.
3. Розбиття множини O(Γ) на класи сумісних виходів: $\Pi_O = \{O^1, ..., O^K\}$.
4. Кодування виходів частковими кодами $K(o_g)$.
5. Розбиття множини τ на множини τ_E і τ_L.
6. Формування таблиць блоків LUTer1-LUTerK
7. Формування системи булевих функцій (9).
8. Формування таблиці блоку LUTerT та системи булевих функцій (10).
9. Формування таблиці блоку LUTerτ і системи булевих функцій (16).
10. Формування таблиці EMB.
11. Реалізація блоку КМПУ U_2 у заданому елементному базисі.

Розглянемо приклад синтезу схеми КМПУ U_2 для граф-схеми алгоритмів Γ_1 (рис. 6). Синтез провадиться з використанням елементів LUT, що мають $S_L = 4$ входів. Для реалізації управляючої

пам'яті використовується блок ЕМВ із конфігурацією $< 4{,}10>$. Ця конфігурація визначає блок ЕМВ з $S_A = 4$ і $t_F = 10$.

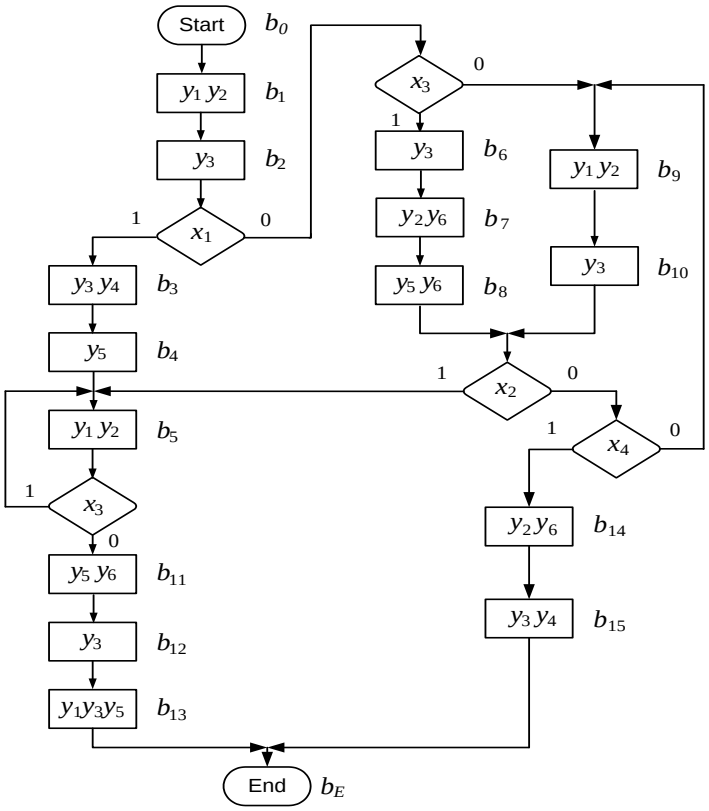


РИС. 6. Граф-схема алгоритму Γ_1

Приклад синтезу схеми КМПУ. Як видно з рис. 6, множина B складається з $M = 15$ операторних вершин. Умовні переходи між мікрокомандами залежить від $L = 4$ логічних умов. У процесі функціонування КМПУ формується $N = 6$ мікрооперацій. Цей аналіз дозволяє отримати множини $B = \{b_1, ..., b_{14}\}$, $X = \{x_1, ..., x_4\}$ і $Y = \{y_1, ..., y_6\}$. Виходячи з (1), для кодування адрес необхідно $R = 4$ розряди, що визначає множини $T = \{T_1, ..., T_4\}$ і $D = \{D_1, ..., D_4\}$.

Використовуючи визначення операторного лінійного ланцюга [3] і метод з роботи [8], отримуємо множину $C = \{\alpha_1, ..., \alpha_G\}$. Це дає такі операторні лінійні ланцюги $\alpha_1 = < b_1, b_2 >$, $\alpha_2 = < b_3, b_4, b_5 >$, $\alpha_3 = < b_6, b_7, b_8 >$, $\alpha_4 = < b_9, b_{10} >$, $\alpha_5 = < b_{11}, b_{12}, b_{13} >$ і $\alpha_6 = < b_{14}, b_{15} >$. Отже, G = 6.

Використовуємо тривіальний підхід до адресації мікрокоманд: адреса мікрокоманди Y_m дорівнює двійковому чотири розрядному еквіваленту числа $(m-1)$. Цей підхід дає адреси $A(b_1) = 0000$, $A(b_2) = 0001$, ..., $A(b_{14}) = 1101$ і $A(b_{15}) = 1110$. Очевидно, умова (2) виконується для сусідніх мікрокоманд із будь-якого операторного лінійного ланцюга $\alpha_g \in C$.

Для розбиття виходів на класи використовуємо метод роботи [16], де розглядається розбиття множини станів автомата на класи сумісних станів. У цій роботі ми замінюємо стани на виходи.

Множина виходів включає такі елементи: $o_1 = b_2$, $o_2 = b_5$, $o_3 = b_8$, $o_4 = b_{10}$, $o_5 = b_{13}$ і $o_6 = b_{15}$. Застосування методу [16] дозволяє отримати розбиття $\Pi_O = \{O^1, O^2\}$ з класами $O^1 = \{o_1, o_2, o_5\}$ і $O^2 = \{o_3, o_4, o_6\}$.

Аналіз цих класів дає $M_1 = M_2 = 3$. Використовуючи (7), отримуємо $R_1 = R_2 = 2$, що дає $\tau^1 = \{\tau_1, \tau_2\}$ і $\tau^2 = \{\tau_3, \tau_4\}$. З граф-схеми алгоритмів Γ_1 можна отримати множини $X^1 = \{x_1, x_3\}$ і $X^2 = \{x_2, x_4\}$. Маємо, $L1 = L2 = 2$. Аналіз умови (8) показує, що вона виконується кожного класу $O^k \in \Pi_O$, отже, розбиття Π_O знайдено.

Закодуємо виходи тривіальним чином: $K(o_1) = K(o_3) = 01$, $K(o_2) = K(o_4) = 10$ і $K(o_5) = K(o_6) = 11$. Код «00» використовуємо для відношення $o_g \notin O^k$.

Аналіз виразу (13) показує, що ця умова перетворюється на $t_F = 10 > 6 + 2$. Використовуючи (14), отримаємо $\Delta_F = 2$. Отже, два з чотирьох розрядів кодів виходів можуть бути реалізовані як виходи EMB.

Спосіб розбиття множини τ не впливає на кількість елементів у блоці LUTerτ. Тому виберемо варіант $\tau_E = \{\tau_1, \tau_2\}$ і $\tau_L = \{\tau_3, \tau_4\}$. Без такого підходу блок LUTerτ складається із $R_0 = 4$ елементів. Врахування особливостей EMB дозволило зменшити це число в 2 рази.

Таблиця блоку LUTerk містить такі стовпці: o_m, $K(o_m)$, Y_S, $A(Y_S)$, X_h, D_h, h. У стовпцях міститься така інформація: o_m – вихід операторного лінійного ланцюга, що належить до класу O^k; $K(o_m)$ – частковий код виходу o_m; Y_S – мікрокоманда, записана у вершині $b_S \in B$; $A(Y_S)$ – адреса мікрокоманди Y_S; x_h – вхідний сигнал, що визначає перехід $< b_m, b_S >$ (перехід може бути безумовним); D_h – набір функцій збудження пам'яті, рівних одиниці записи адреси $A(Y_S)$ у лічильник; h – номер переходу. Таблиці формуються, використовуючи методику роботи [15]. Блок LUTer1 наведений у табл. 1, блок LUTer2 – у табл. 2.

ТАБЛИЦЯ 1. Таблиця блока LUTer1

o_m	$K(o_m)$	Y_S	$A(Y_S)$	x_h	D_h	h
o_1	01	Y_3	0010	x_1	D_3	1
		Y_6	0101	$\overline{x_1}x_3$	$D_2 D_4$	2
		Y_9	1000	$\overline{x_1}\,\overline{x_2}$	D_1	3
o_2	10	Y_5	0100	x_3	D_2	4
		Y_{11}	1010	$\overline{x_3}$	$D_1 D_3$	5
o_5	11	Y_E	0000	1	—	6

З табл.1 можна отримати таку систему часткових функцій збудження пам'яті:

$$D_1^1 = \overline{\tau_1}\tau_2\,\overline{x_1}\,\overline{x_3} \vee \tau_1\,\overline{\tau_2}\,\overline{x_3}; \qquad D_3^1 = \overline{\tau_1}\tau_2 x_1 \vee \tau_1\,\overline{\tau_2}\,\overline{x_3};$$
$$D_2^1 = \overline{\tau_1}\tau_2\,\overline{x_1}x_3 \vee \tau_1\,\overline{\tau_2}x_3; \qquad D_4^1 = \overline{\tau_1}\tau_2\,\overline{x_1}x_3. \tag{17}$$

ТАБЛИЦЯ 2. Таблиця блока LUTer2

o_m	$K(o_m)$	Y_S	$A(Y_S)$	X_h	D_h	h
o_3	01	Y_5	0100	x_2	D_2	1
		Y_{14}	1101	$\overline{x_2}x_4$	$D_1 D_2 D_3$	2
		Y_9	1000	$\overline{x_2}\,\overline{x_4}$	D_1	3
o_4	10	Y_5	0100	x_2	D_2	4
		Y_{14}	1101	$\overline{x_2}x_4$	$D_1 D_2 D_3$	5
		Y_9	1000	$\overline{x_2}\,\overline{x_4}$	D_1	6
o_6	11	Y_E	0000	1	—	7

На основі табл. 2 формується наступна система функцій збудження пам'яті:

$$D_1^2 = \overline{\tau_1}\tau_2\overline{x_2} \vee \tau_1\overline{\tau_2}\,\overline{x_2}; \qquad D_3^2 = \overline{\tau_1}\tau_2\overline{x_2}x_3 \vee \tau_1\overline{\tau_2}x_4;$$
$$D_2^2 = \overline{\tau_1}\tau_2 x_2 \vee \overline{\tau_1}\tau_2\overline{x_2}x_4 \vee \tau_1\overline{\tau_2}x_2 \vee \tau_1\overline{\tau_2}\,\overline{x_2}x_4; \qquad D_4^2 = 0. \tag{18}$$

Блок LUTerT є таблицею зі стовпцями D_r, O^1,..., O^K. При виконанні умови $D_r^k \in O^k$ на перетині рядка D_r і стовпця O^k записується одиниця. В іншому випадку на цьому перетині записується "0". У нашому випадку ця таблиця має стовпці D_r, O^1 і O^2 (табл. 3).

ТАБЛИЦЯ 3. Таблиця блоку LUTerτT

D_r	O^1	O^2	D_r	O^1	O^2	D_r	O^1	O^2	D_r	O^1	O^2
D_1	1	1	D_2	1	1	D_3	1	1	D_4	1	0

З табл. 3 можна знайти наступну систему булевих функцій:

$$D_1 = D_1^1 \vee D_1^2; \qquad D_2 = D_3^1 \vee D_3^2;$$
$$D_2 = D_2^1 \vee D_2^2; \qquad D_4 = D_4^1. \tag{19}$$

Ця система визначає схему блоку LUTerT. Очевидно, схема складається із трьох елементів LUT. Функція D_4 генерується безпосередньо блоком LUTer1.

У цьому прикладі блок LUTerτ генерує лише змінні $\tau_3, \tau_4 \in \tau$. Таблиця блоку LUTerτ містить стовпці: вихід операторного лінійного ланцюга (o_g), адресу виходу $A(o_g)$, змінні (τ_3, τ_4), що формують, номер операторного лінійного ланцюга (g). У нашому випадку цей блок визначається табл. 4.

ТАБЛИЦЯ 4. Таблиця блока LUTerτ

o_g	$A(o_g)$	τ_3	τ_4	g
o_3	0111	0	1	3
o_4	1001	1	0	4
o_6	1110	1	1	6

З табл. 4 отримаємо наступну систему булевих функцій:

$$\tau_3 = o_4 \vee o_6 = T_1\overline{T_2}\,\overline{T_3}T_4 \vee T_1T_2T_3\overline{T_4}; \qquad \tau_4 = o_3 \vee o_6 = \overline{T_1}T_2T_3T_4 \vee T_1T_2T_3\overline{T_4}. \tag{20}$$

Блок управляючої пам'яті представляється таблицею зі стовпцями: Адреса, b_m, y_O, y_E, y_1,...., y_6, τ_1, τ_2 (табл. 5).

Таблиця управляючої пам'яті заповнюється наступним чином:

- якщо $b_m \in B$ не є виходом операторного лінійного ланцюга, то $y_0 = 1$;
- якщо $b_m \in B$ пов'язана з кінцевою вершиною, то $y_E = 1$;
- якщо мікрооперація y_n формується в $b_m \in B$, то $y_n = 1$;
- якщо змінна є кодом $K(o_g)$, то $\tau_r = 1$.

Після формування системи булевих функцій (17) – (20) та табл. 5 можна перейти до реалізації схеми. Для цього необхідно використовувати САПР типу Vivado [17] чи Quartus [18]. Ми не розглядаємо цей етап у деталях. Ці питання представлені, наприклад, у роботі [12].

ТАБЛИЦЯ 5. Таблиця блоку ЕМВ

Адреса	b_m	y_O	y_E	y_1	y_2	y_3	y_4	y_5	y_6	τ_1	τ_2
0000	b_1	1	0	1	1	0	0	0	0	0	0
0001	b_2	0	0	0	1	0	0	0	0	0	1
0010	b_3	1	0	0	0	1	1	0	0	0	0
0011	b_4	1	0	0	0	0	0	1	0	0	0
0100	b_5	0	0	1	1	0	0	0	0	1	0
0101	b_6	1	0	0	0	1	0	0	0	0	0
0110	b_7	1	0	0	1	0	0	0	1	0	0
0111	b_8	0	0	0	0	0	0	1	1	0	0
1000	b_9	1	0	1	1	0	0	0	0	0	0
1001	b_{10}	0	0	0	0	1	0	0	0	0	0
1010	b_{11}	1	0	0	0	0	0	1	1	0	0
1011	b_{12}	1	0	0	0	1	0	0	0	0	0
1100	b_{13}	0	1	1	0	1	0	1	0	1	1
1101	b_{14}	1	0	0	1	0	0	0	1	0	0
1110	b_{15}	0	1	0	0	1	1	0	0	0	0

Результати. Запропонований у роботі метод дозволяє зменшити кількість елементів LUT у схемі адресації КМПУ. Ця мінімізація не потребує додаткових ресурсів мікросхеми FPGA. Зменшення числа елементів LUT досягається за рахунок надмірності блоків ЕМВ. Число виходів змінюється дискретно: зменшення числа входів S_A на одиницю подвоює число виходів [6]. Це випливає із формули (5).

При виконанні умови (15) повністю зникає необхідність використання елементів LUT для генерації кодів виходів операторних лінійних ланцюгів. Якщо умова (15) порушується, але виконується умова (13), то частина кодів виходів генерується блоком управляючої пам'яті.

Такий підхід дозволяє зменшити кількість як елементів LUT, так і загальне число міжз'єднань у схемі. Як зазначено в [14], міжз'єднання відповідальні за 70 % споживаної потужності. З іншого боку, зі зростанням рівня інтеграції FPGA, міжз'єднання починають значно впливати на час поширення сигналів у схемі [13].

Висновки. Запропонований метод дозволяє покращити всі основні характеристики КМПУ порівняно з характеристиками пристрою, запропонованого у роботі [15]. До цих характеристик відносяться: площа кристала, займана схемою КМПУ, максимальна операційна частота, споживана потужність. Очевидно, що менше різниця, то більшою мірою оптимізуються характеристики схеми.

Подальший напрямок наших досліджень пов'язаний із застосуванням запропонованого методу для вирішення двох проблем: перша – оптимізації схем автоматів Мілі, пам'ять станів яких реалізована на лічильниках, друга – адаптація запропонованого методу до комплексного кодування виходів КМПУ.

Авторські внески. Баркалов О.О. – методологія, формальний аналіз, написання – оригінальна чернетка; Тітаренко Л.О. – узагальнення, концептуалізація, методологія; Головін О.М. – формальний аналіз, ресурси, написання – рецензування та редагування, програмне забезпечення; Матвієнко О.В. – формальний аналіз, дослідження, ресурси, візуалізація, написання – рецензування та редагування.

Фінансування. Автори не отримували фінансування для проведення досліджень та написання статті.

Список літератури

1. DeMicheli G. Synthesis and optimization of digital circuits. New York: McGraw-Hill, 1994. 576 p.
2. Ruiz-Rosero J., Ramirez-Gonzalez G., Khanna R. Field Programmable Gate Array Applications – A Scientometric Review. *Computation*. 2019. **7** (4). 63. https://doi.org/10.3390/computation7040063.
3. Barkalov A., Titarenko L. Logic Synthesis for Compositional Microprogram Control Units Lectures Notes in Electrical Engineering. Berlin: Springer (Berlin Verlag Heidelberg), 2008. No. 22. 272 p.
4. Virtex-7 T and XT FPGAs Data Sheet. https://docs.xilinx.com/v/u/en-US/ds183_Virtex_7_Data_Sheet (звернення: 01.01.2025)
5. Sklyarov V., Skliarova I., Barkalov A., Titarenko L. Synthesis and optimization of FPGA-based systems. Berlin: Springer, 2014. 432 p. https://doi.org/10.1007/978-3-319-04708-9_6
6. Kuon I., Tessier R., Rose J. FPGA Architecture: Survey and Challenges. Foundations and Trends in Electronic Design Automation. 2008. Vol. 2, No. 2. P. 135–253.
7. Baranov S. Logic synthesis for control automata. Dordrecht: Kluwer Academic Publishers, 1994. 312 p.
8. Barkalov A. Microprogram control unit as composition of automate with programmable and hardwired logic. *Automatics and Computer Science*. 1983. **17** (4). P. 36–41.
9. Xilinx. http://www.xilinx.com (https://www.amd.com/en/products/adaptive-socs-and-fpgas/fpga.html#overview) (звернення: 01.01.2025).
10. Trimberg S. Three ages of FPGA: A retrospective on the first thirty years of FPGA technology. *IEEE Proceedings*. **10** (2). P. 16–29. http://dx.doi.org/10.1109/mssc.2018.2822862
11. Skliarova I., Sklyarov V., Sudnitson A. Design of FPGA-based circuits using hierarchical finite state machines. Tallinn: TUT Press, 2012. 240 p.
12. Kubica M., Opara A., Kania D. Technology Mapping for LUT- based. FPGA. Berlin: Springer, 2021.
13. Feng W., Greene J., Mishchenko A. Improving FPGA Performance with a S44 LUT Structure. *In Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. FPGA'18, Association for Computing Machinery. 2018. P. 61–66. http://dx.doi.org/10.1145/3174243.3174272
14. Islam M.M., Hossain M., Shahjalal M., Hasan M.K., Jang Y.M. Area-time e_cient hardware implementation of modular multiplication for elliptic curve cryptography. *IEEE Access PP*. 2020. Vol. 8. P. 73898–73906. http://dx.doi.org/10.1109/ACCESS.2020.2988379

15. Barkalov O., Titarenko L., Saburova S., Golovin, O., Matvienko O. Optimization of the microprogram Mealy machine circuit based on LUT and EMB. *Cybernetics and Computer Technologies*. 2024. 4. P. 121–133. https://doi.org/10.34229/2707-451X.24.4.11
16. Barkalov O., Titarenko L., Mielcarek K. Hardware reduction for LUT based Mealy FSMs. *International Journal of Applied Mathematics and Computer Science*. 2018. Vol. 28, No. 3. P. 595–607.
17. Vivado Design Suite. 2020. https://www.xilinx.com/products/design-tools/vivado.html (звернення: 01.01.2025)
18. Quartus II. https://www.intel.com/content/www/us/en/products/details/fpga/development-tools/quartus-prime/resource.html (звернення: 01.01.2025)

Одержано 25.03.2025

Баркалов Олександр Олександрович,
доктор технічних наук, професор Університету Зеленогурського (Польща),
https://orcid.org/0000-0002-4941-3979
A.Barkalov@iie.uz.zgora.pl

Тітаренко Лариса Олександрівна,
доктор технічних наук,професор Університету Зеленогурського (Польща),
професор Харківського національного університету радіоелектроніки,
https://orcid.org/0000-0001-9558-3322
L.Titarenko@iie.uz.zgora.pl

Головін Олександр Миколайович,
кандидат технічних наук, старший науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України, Київ,
https://orcid.org/0000-0002-0279-812X
o.m.golovin.1@gmail.com

Матвієнко Олександр Володимирович,
науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України, Київ.
https://orcid.org/0000-0003-1838-1422
avmatv@ukr.net

УДК 004.274

О.О. Баркалов [1], Л.О. Тітаренко [1, 2], О.М. Головін [3], О.В. Матвієнко [3]

Перетворення адрес у композиційному мікропрограмному пристрої управління

[1] *Університет Зеленогурський, Зелена Гура, Польща*
[2] *Харківський національний університет радіоелектроніки, Харків, Україна*
[3] *Інститут кібернетики імені В.М. Глушкова НАН України, Київ*
[*] *Листування: avmatv@ukr.net*

Вступ. Цифрові системи складаються з комбінаційних та послідовних блоків. До найважливіших послідовних блоків відносяться пристрої управління. Схеми пристроїв управління не відносяться до типових бібліотечних компонентів САПР, тому проєктування схеми пристрою управління – більш трудомісткий процес, ніж реалізація систем з таких звичайних блоків, як, наприклад, регістри, лічильники, арифметичні та логічні блоки.

Мета статті. У процесі реалізації цифрових систем виникають проблеми з оптимізації їх характеристик. У цій роботі розглядається проблема зменшення апаратурних витрат у схемі композиційного мікропрограмного пристрою управління (КМПУ). У якості елементного базису використовуються ресурси мікросхем FPGA (field-programmable logic array).

Пропонований у статті метод ґрунтується на адаптації алгоритмів оптимізації схем мікропрограмних автоматів до особливостей КМПУ. Метод спрямований на перетворення адрес деяких мікрокоманд в часткові входи. У разі виконанні певних умов такий підхід дозволяє значно спростити схему адресації

мікрокоманд. Це дає змогу покращити характеристики схеми КМПУ порівняно з іншими відомими методами. Для завдання алгоритму функціонування КМПУ у статті використано мову граф-схем алгоритмів.

Результати. Розглянуто реалізацію схеми КМПУ з використанням таких ресурсів мікросхем FPGA, як елементи табличного типу LUT та вбудовані блоки пам'яті EMB. Оптимізація досягається за рахунок використання надмірності EMB по виходам.

Запропонований метод дозволяє поліпшити такі основні характеристики КМПУ як площа кристала, займана схемою КМПУ, максимальна операційна частота, загальна кількість з'єднань і споживана потужність.

У статті надано покроковий алгоритм методу синтезу КМПУ за заданою граф-схемою алгоритмів, наведено приклад синтезу КМПУ із застосуванням запропонованого методу, показано умови його застосування.

Висновки. Запропонований метод дозволяє зменшити кількість елементів LUT у схемі адресації КМПУ. Ця мінімізація не потребує додаткових ресурсів мікросхеми FPGA. Зменшення числа елементів LUT досягається за рахунок використання надмірності виходів блоків EMB.

Ключові слова: КМПУ, LUT, EMB, операторні лінійні ланцюги.

UDC 004.274

Alexandr Barkalov [1], Larysa Titarenko [1, 2], Oleksandr Golovin [3], Oleksandr Matvienko [3]

Address Translation in a Compositional Microprogram Control Unit

[1] *University of Zielona Gora, Poland*
[2] *Kharkiv National University of Radio Electronics, Kharkiv, Ukraine*
[3] *V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine, Kyiv*
[*] *Correspondence: avmatv@ukr.net*

Introduction. Digital systems consist of combinational and sequential blocks. The most important sequential blocks include control units. Control unit circuits are not typical library components of CAD systems. Due to it, the designing a control unit circuit is a more labor-intensive process than implementing systems with such common blocks as registers, counters, arithmetic and logic blocks**.**

The purpose of the article. When implementing digital systems, problems arise in optimizing their characteristics. This paper considers the problem of reducing hardware costs in the circuits of compositional microprogram control units (CMCU). The resources of FPGA (field-programmable logic array) chips are used as an element basis. The method proposed in the article is based on the adaptation of algorithms for optimizing microprogram automata circuits to the features of CMCUs. The method is aimed at converting the addresses of some microinstructions into partial inputs. Under certain conditions, this approach can significantly simplify the block of microinstruction addressing. This approach can improve the characteristics of the CMCU circuit in comparison with other known methods. The language of graph-schemes of algorithms (GSA) is used to specify the algorithm for the CMCU operating.

Results. The implementation of the CMCU circuit using such FPGA chip resources as look-up table (LUT) elements and embedded memory blocks (EMB) is considered. Optimization is achieved by using the EMB redundancy at the outputs.

The proposed method allows improving such basic CMCU characteristics as the chip area occupied by the CMCU circuit, the maximum operating frequency, the total number of interconnections and the power consumption.

The article presents a step-by-step algorithm for synthesizing the CMCU for a given GSA. Also, it provides an example of CMCU synthesis using the proposed method. At last, the conditions of the proposed method’s applicability are shown.

Conclusions. The proposed method allows reducing the number of LUT elements in the CMCU addressing circuit. This minimization does not require any additional FPGA chip resources. Reducing the number of LUT elements is achieved by using the redundancy of the EMB block outputs.

Keywords: CMCU, LUT, EMB, operator linear chains.