УДК 681.326

І.Я. Зеленьова, О.М. Мірошкін, А.В. Казачанський
Донецький національний технічний університет, м. Донецьк,
кафедра електронних обчислювальних машин
E-mail: irina@cs.dgtu.donetsk.ua

МОДИФІКАЦІЯ АЛГОРИТМУ ШВАРЦА ДЛЯ ОПТИМІЗАЦІЇ ОПЕРАЦІЙНОЇ ЧАСТИНИ КОМПОЗИЦІЙНИХ МІКРОПРОГРАМНИХ ПРИСТРОЇВ КЕРУВАННЯ З УРАХУВАННЯМ ОБМЕЖЕНЬ БАЗИСУ FPGA

Abstract
Zeleneva I.Y., Kazachanskyy A.V., Miroshkin A.N. Shvarts algorithm modification for compositional microprogram control unit operational part optimization granting FPGA basis scopes. *The compatible microoperations fields coding method for hardware expenses minimisation in an operational part of compositional microprogram control units (CMCU) is considered in article. For microoperations placing on separate fields Schwarz's modified algorithm is used. By working out of the modified algorithm of Schwarz restrictions of FPGA (Field-Programmable Gate Arrays) basis in which the CMCU structure is realised are considered.*
Keywords: *Shvarts algorithm, control unit, FPGA basis, microoperations, operational part, modified algorithm.*

Анотація
Зеленьова І.Я., Мірошкін О.М., Казачанський А.В. Модифікація алгоритму Шварца для оптимізації операційної частини композиційних мікропрограмних пристроїв керування з урахуванням обмежень базису FPGA. *У статті описується та досліджується метод кодування полів сумісних мікрооперацій для мінімізації апаратурних витрат при реалізації операційної частини композиційних мікропрограмних пристроїв керування (КМПК). Для розподілу мікрооперацій по окремим полям використовується модифікація алгоритму Шварца. Модифікація алгоритму Шварца враховує особливості базису FPGA (Field-Programmable Gate Arrays), у якому пропонується реалізовувати структуру КМПК.*
Ключові слова: *алгоритм Шварца, пристрій керування, базис FPGA, мікрооперації, операційна частина, модифікація алгоритму.*

Аннотация
Зеленева И.Я., Мирошкин А.Н., Казачанский А.В. Модификация алгоритма Шварца для оптимизации операционной части композиционных микропрограммных устройств управления с учетом ограничений базиса FPGA. *В статье описывается и исследуется метод кодирования полей совместимых микроопераций для минимизации аппаратурных затрат при реализации операционной части композиционных микропрограммных устройств управления (КМУУ). Для распределения микроопераций по отдельным полям применяется модификация алгоритма Шварца. Модификация алгоритма Шварца учитывает особенности базиса FPGA (Field-Programmable Gate Arrays), в котором предлагается реализовывать структуру КМУУ.*
Ключевые слова: *алгоритм Шварца, устройство управления, базис FPGA, микрооперации, операционная часть, модификация алгоритма.*

Вступ. Розвиток цифрової техніки та електроніки, їх широке застосування на промислових підприємствах у системах керування технологічними процесами обумовили швидкий розвиток

елементної бази для реалізації цифрових систем. Різноманітні цифрові системи зручно реалізовувати на програмованих логічних інтегральних схемах (ПЛІС). Фірми, що спеціалізуються на виробництві компонентів для електронної техніки, випускають широкий спектр ПЛІС: PLA (Programmable Logic Arrays), PAL (Programmable Array Logics), GAL (Generic Array Logics), PLS (Programmable Logic Sequencers), CPLD (Complex Programmable Logic Devices) та FPGA (Field-Programmable Gate Arrays) [1]. Але за статистикою найбільшу популярність у цифровій техніці отримали FPGA. У цих ПЛІС у якості програмованих елементів використовуються комірки статичного ОЗП (оперативний запам'ятовуючий пристрій), мікросхеми FPGA можна багаторазово перепрограмувати. Конфігурація FPGA може зберігатися у спеціальних конфігураційних ПЗП (постійний запам'ятовуючий пристрій).

Сучасні алгоритми керування цифровими системами зручно представляти у вигляді кінцевих автоматів і реалізовувати на ПЛІС у вигляді мікропрограмних пристроїв керування, бо такі пристрої мають регулярну структуру. Але мікропрограмні пристрої керування мають ряд недоліків, основними з яких є: необхідність вказувати у форматі мікрокоманд принаймні одну адресу переходу, що збільшує довжину мікрокоманд, а, отже, і необхідну ємність керуючої пам'яті; багатоспрямовані умовні переходи у алгоритмах керування виконуються за декілька тактів роботи пристрою. Ці недоліки нівелюються у композиційних мікропрограмних пристроях керування (КМПК), які представляють собою композицію структур керуючих автоматів із програмованою та "жорсткою" логікою. Але КМПК мають менш регулярну структуру, ніж прості мікропрограмні пристрої керування.

Існує велика кількість різноманітних методів оптимізації апаратних витрат у структурах КМПК, але усі ці методи передбачають реалізацію КМПК у базисі PLA або PAL. Через те, що вказані методи не можна застосовувати без змін при синтезі та реалізації структур КМПК у базисі сучасних ПЛІС типу FPGA, створення нових методів є актуальною задачею. Нові методи повинні враховувати особливості сучасних елементних базисів.

Метою розробки є мінімізація апаратних витрат у операційній частині КМПК за рахунок використання модифікованого методу кодування полів сумісних мікрооперацій.

Задачею досліджень є розробка модифікованого алгоритму для розподілу мікрооперацій по окремих полях на основі модифікованого алгоритму Шварца із використанням елементів алгоритму гілок та меж. При розподілі мікрооперацій повинні враховуватись обмеження базису FPGA.

Аналіз базису FPGA та обґрунтування задачі оптимізації операційної частини КМПК. Усі сучасні FPGA мають у своєму складі чотири основних структурних елемента [2]:

1. Блоки введення-виведення даних (БВВ), які забезпечують введення зовнішніх сигналів до FPGA та виведення інформації на зовнішні контакти мікросхеми. БВВ вміщують набір елементів пам'яті, мультиплексорів для забезпечення різноманітних способів комутації сигналів та елементів із трьома станами.
2. Логічні блоки, що конфігуруються (ЛБК) — це основні елементи логіки у FPGA. Основними елементами у ЛБК є логічні комірки — ЛК (Logic Cell — LC). ЛК уміщують у своєму складі наступні елементи: функціональний генератор, який реалізовано у вигляді 4-х або 5-ти входової таблиці перетворення (Look-Up Table — LUT); логічні елементи для забезпечення прискореного перенесення та контролю даних; елементи пам'яті.
3. Блочна пам'ять (Block Select RAM), яка створена додатково до розподіленої пам'яті невеликої ємності (Select RAM), що реалізується на LUT-елементах.
4. Програмована матриця трасування, яка забезпечує зв'язок між окремими ЛБК і між ЛБК та БВВ.

Існує чотири основних метода мінімізації апаратних витрат у операційній частині КМПК: унітарне кодування мікрооперацій, максимальне кодування наборів мікрооперацій, кодування полів сумісних мікрооперацій та вертикалізація граф-схеми алгоритму [3].

При застосуванні унітарного кодування мікрооперацій кожній мікрооперації у мікропрограмі відповідає окремий розряд шини даних керуючої пам'яті (КП). Реалізація КП

у базисі FPGA можлива із застосуванням або блочної пам'яті, або набору LUT-елементів. При реалізації КП із застосуванням блочної пам'яті вона повинна мати наступні параметри:

$$N^{A}_{BRAM} =]\log_2 N_{ST}[, \quad (1)$$

$$N^{D}_{BRAM} = N_y + 1, \quad (2)$$

де N^{A}_{BRAM} — кількість адресних входів блочної пам'яті;

N^{D}_{BRAM} — кількість ліній шини даних блочної пам'яті;

N_{ST} — кількість операторних вершин у алгоритмі керування;

N_y — кількість мікрооперацій у мікропрограмі керування.

У формулі розрахунку кількості ліній шини даних необхідно додавати одиницю для врахування службового сигналу керування y_0 у структурі КМПК. При реалізації КП із застосуванням LUT-елементів за умови:

$$]\log_2 N_{ST}[> N^{IN}_{LUT}, \quad (3)$$

де N^{IN}_{LUT} — кількість входів у елементі LUT, необхідно синтезувати КП із застосуванням складного каскадування елементів LUT. У цьому випадку кількість LUT-елементів, які необхідні для реалізації КП, залежить від необхідної розрядності шин адреси та даних КП і розрахунок цієї кількості важко піддається формалізації. За умови:

$$]\log_2 N_{ST}[\le N^{IN}_{LUT} \quad (4)$$

кількість елементів LUT (N_{LUT}), що необхідна для реалізації КП, розраховується за формулою:

$$N_{LUT} = N_y + 1. \quad (5)$$

Отже, використання LUT-елементів для реалізації керуючої пам'яті доцільне тільки при виконанні умови (4) та коли необхідна кількість розрядів шини даних КП відносно мала (100–200 розрядів).

При застосуванні методу максимального кодування наборів мікрооперацій керуюча пам'ять може бути реалізована як на блочній пам'яті FPGA, так і на LUT-елементах. Розрахунок параметрів КП відбувається так само, як і у методі унітарного кодування мікрооперацій, з тією лише різницею, що розрядність шини даних блочної пам'яті та необхідна кількість елементів LUT для реалізації КП при виконанні умови (4), розраховуються за формулами:

$$N^{D}_{BRAM} =]\log_2 N_Y[, \quad (6)$$

$$N_{LUT} =]\log_2 N_Y[, \quad (7)$$

де N_Y — кількість мікрокоманд у мікропрограмі керування.

Дешифратор та логічна схема формування мікрооперацій за умови застосування методу максимального кодування наборів мікрооперацій реалізуються у базисі FPGA на блочній пам'яті або на елементах LUT. При реалізації дешифратора та логічної схеми формування мікрооперацій на блочній пам'яті її параметри розраховуються за допомогою наступних формул:

$$N^{A}_{BRAM} =]\log_2 N_Y[, \quad (8)$$

$$N^{D}_{BRAM} = N_y + 1. \quad (9)$$

При використанні LUT-елементів при виконанні умови:

$$]\log_2 N_Y[> N^{IN}_{LUT} \quad (10)$$

потрібно виконувати складне каскадне з'єднання LUT-елементів, кількість яких залежить від необхідної кількості входів дешифратора та кількості мікрооперацій у мікропрограмі керування (N_y). При виконанні умови $]\log_2 N_Y[\le N^{IN}_{LUT}$ необхідна кількість елементів LUT для реалізації дешифратора та логічної схеми розраховується за формулою:

$$N_{LUT} = N_y + 1. \quad (11)$$

Отже, використання методу максимального кодування наборів мікрооперацій доцільне тільки при виконанні умови:

$$]\log_2 N_Y[\le N_{LUT}^{IN} \quad (12)$$

та при реалізації КП на блочній пам'яті, а дешифратора та логічної схеми формування мікрооперацій — на LUT-елементах.

При застосуванні методу кодування полів сумісних мікрооперацій у базисі FPGA керуючу пам'ять можна реалізовувати або на блочній пам'яті, або на наборі LUT-елементів. Параметри блочної пам'яті при реалізації на її основі КП розраховуються за наступними формулами:

$$N_{BRAM}^{A} =]\log_2 N_{ST}[, \quad (13)$$

$$N_{BRAM}^{D} = 1 + \sum_{m=1}^{M}]\log_2 (N_{FIELDm} + 1)[, \quad (14)$$

де N_{FIELDm} — кількість мікрооперацій y у полі m;

M — кількість полів.

Одиниця у формулі розрахунку розрядності шини даних блочної пам'яті додається з тієї ж причини, як і у формулі (2). До загальної кількості мікрооперацій y у полі m додається одиниця для врахування випадку, коли усі мікрооперації у полі m мають неактивний стан.

При реалізації КП на елементах LUT за умови (3) потрібно також виконувати складне каскадне з'єднання цих елементів. Кількість LUT-елементів, що необхідна для побудови керуючої пам'яті, у цьому випадку залежить від кількості адресних розрядів КП та розрядності шини даних і розрахунок цієї кількості доволі важко формалізувати. При виконанні умови (4) кількість LUT-елементів, необхідних для реалізації КП, розраховується за наступною формулою:

$$N_{LUT} = 1 + \sum_{m=1}^{M}]\log_2 (N_{FIELDm} + 1)[. \quad (15)$$

Отже, реалізацію керуючої пам'яті при використанні методу кодування полів сумісних мікрооперацій доцільно робити на блочній пам'яті при виконанні умови (3), або на LUT-елементах при виконанні умов (4) та

$$1 + \sum_{m=1}^{M}]\log_2 (N_{FIELDm} + 1)[< 100. \quad (16)$$

Реалізація окремих дешифраторів полів мікрооперацій у базисі FPGA можлива, аналогічно з реалізацією КП, як на блочній пам'яті, так і на елементах LUT. При реалізації дешифратора на блочній пам'яті вона повинна мати наступні параметри:

$$N_{BRAM}^{A} =]\log_2 (N_{FIELD} + 1)[, \quad (17)$$

$$N_{BRAM}^{D} = N_{FIELD}, \quad (18)$$

де N_{FIELD} — кількість мікрооперацій у полі.

При реалізації дешифраторів на елементах LUT за умови:

$$]\log_2 (N_{FIELD} + 1)[> N_{LUT}^{IN} \quad (19)$$

необхідно виконувати складне каскадне з'єднання LUT-елементів. Кількість елементів LUT у цьому випадку залежить від кількості адресних входів дешифратора, що реалізується на цих елементах, та від кількості мікрооперацій у полі (N_{FIELD}) і розрахунок цієї кількості доволі важко формалізувати. При виконанні умови:

$$]\log_2 (N_{FIELD} + 1)[\le N_{LUT}^{IN} \quad (20)$$

кількість LUT-елементів, що необхідна для реалізації дешифратора, дорівнює N_{FIELD} ($N_{LUT} = N_{FIELD}$).

Таким чином, для реалізації дешифраторів полів мікрооперацій доцільно використовувати елементи LUT, при цьому бажано, щоб при розміщенні мікрооперацій по окремих полях виконувалася умова (20).

При застосуванні методу вертикалізації граф-схеми алгоритму алгоритм керування перетворюється таким чином, що у кожній мікрокоманді знаходиться не більше за одну мікрооперацію:

$$\left|Y(b_q)\right| \le 1, \tag{21}$$

де $Y(b_q)$ — мікрокоманда, що записана у вершині b_q. Ця формула передбачає варіант, коли у алгоритмі керування присутні порожні вершини.

При використанні методу вертикалізації граф-схеми алгоритму мікрооперації розміщуються у одному полі (бо вони усі будуть сумісні між собою) та кодуються за допомогою методу максимального кодування, при цьому шина даних у КП повинна мати розрядність $\left]\log_2(N_y+1)\right[+1$. Параметри блочної пам'яті, за умови реалізації на її основі КП, розраховуються наступним чином:

$$N^A_{BRAM} = \left]\log_2 N_{ST}\right[, \tag{22}$$

$$N^D_{BRAM} = \left]\log_2(N_y+1)\right[+1. \tag{23}$$

При реалізації КП на LUT-елементах при виконанні умови (3) необхідно виконувати каскадне поєднання LUT-елементів. При виконанні умови (4) для реалізації КП знадобиться лише $\left]\log_2(N_y+1)\right[+1$ елемента LUT ($N_{LUT} = \left]\log_2(N_y+1)\right[+1$).

Таким чином, при використанні цього методу КП доцільно реалізовувати на блочній пам'яті при виконанні умови (3), або на LUT-елементах при виконанні умови (4).

Реалізація дешифратора поля у базисі FPGA на блочній пам'яті або на LUT-елементах виконується аналогічно з реалізацією дешифраторів полів при використанні методу кодування полів сумісних мікрооперацій.

При реалізації дешифратора на блочній пам'яті вона повинна мати наступні параметри:

$$N^A_{BRAM} = \left]\log_2(N_y+1)\right[, \tag{24}$$

$$N^D_{BRAM} = N_y. \tag{25}$$

При реалізації дешифратора поля на елементах LUT розрахунок кількості цих елементів, яка знадобиться для реалізації дешифратора, відбувається згідно із наступною формулою:

$$N_{LUT} = N_y * \left]\frac{\left]\log_2(N_y+1)\right[-1}{N^{IN}_{LUT}-1}\right[. \tag{26}$$

При використанні методу вертикалізації граф-схеми алгоритму значно збільшується час виконання мікропрограми та необхідна розрядність шини адреси КП. У якості альтернативного варіанту вертикалізації граф-схеми пропонується перекодування сигналів керування у об'єкті керування так, щоб у кожній непорожній операторній вершині мікропрограмного пристрою керування об'єктом формувався один унікальний сигнал керування. При цьому кількість мікрооперацій у мікропрограмі керування буде дорівнювати:

$$N_y = N_{ST} - N_\varnothing. \tag{27}$$

де $N_\varnothing$ — кількість порожніх операторних вершин у алгоритмі керування. При використанні такого методу вертикалізації граф-схеми алгоритму не збільшується час роботи мікропрограми, але для цього необхідно виконувати складний процес перекодування сигналів керування у об'єкті керування, при цьому збільшуються апаратні витрати при реалізації об'єкта керування.

Розробка модифікованого алгоритму Шварца. Загальна структура схеми формування мікрооперацій із використанням методу кодування полів сумісних мікрооперацій наведена на рис.1.

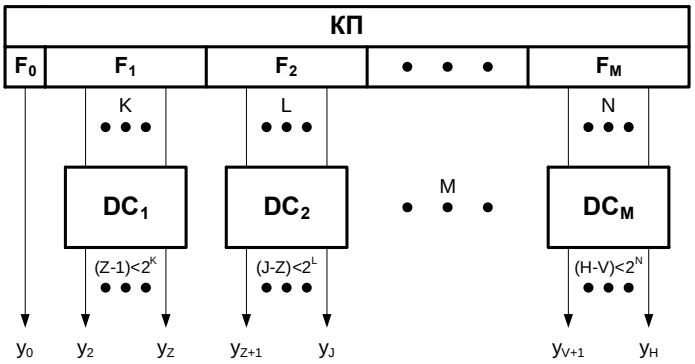

Рисунок 1 — Структура схеми формування мікрооперацій із використанням методу кодування полів сумісних мікрооперацій

Службовий сигнал керування y_0, що керує автоматами з програмованою та "жорсткою" логікою у складі структури КМПК, при розміщенні мікрооперацій по окремих полях не включається в жодне з полів і має прямий вихід з блоку КП, бо включення цього сигналу до якогось з полів ускладнює процес розподілення мікрооперацій і не є ефективним.

При кодуванні мікрооперацій у окремих полях за допомогою максимального кодування необхідно враховувати випадок, коли усі мікрооперації в окремому полі мають неактивний стан. Це досягається введенням додаткового коду неактивного стану усіх мікрооперацій у полі. При цьому у процесі розрахунку необхідної розрядності коду мікрооперацій до загальної кількості мікрооперацій необхідно додавати одиницю:

$$K_{FIELD}^{ADR} = \left\rceil \log_2 (N_y + 1) \right\lceil, \tag{28}$$

де K_{FIELD}^{ADR} — необхідна розрядність коду при кодуванні мікрооперацій у полі;

N_y — кількість мікрооперацій у полі.

Пропонується модифікація алгоритму Шварца [3], яка складається з двох основних частин.

Перша частина модифікованого алгоритму Шварца полягає у знаходженні та видаленні з подальшого розглядання еквівалентних мікрооперацій.

Еквівалентними звуться такі мікрооперації $y_i,...,y_k$, які в усіх мікрокомандах $Y_1,...,Y_Z$ мікропрограми керування використовуються тільки разом і жодна з них в жодній мікрокоманді не використовується окремо від інших.

Якщо G мікрооперацій є еквівалентними, то $G-1$ з них виключаються з подальшого розглядання, а при реалізації схеми формування мікрооперацій вони реалізуються простим розгалуженням одного виходу схеми на G напрямів. Це дозволяє значно скоротити апаратні витрати у схемі формування мікрооперацій.

Друга частина модифікованого алгоритму Шварца призначена для кінцевого розміщення сумісних мікрооперацій по окремих полях. При реалізації полів на елементах LUT у базисі FPGA бажано, щоб виконувалась умова (20). Тоді для реалізації одного поля знадобиться N_{FIELD} елементів LUT ($N_{LUT} = N_{FIELD}$). Але не завжди можливо розподілити мікрооперації між полями так, щоб виконувалась вищезазначена умова. При виконанні умови (19) для реалізації дешифратора поля знадобиться наступна кількість елементів LUT (при реалізації дешифратора без спрощення його структури):

$$N_{LUT} = N_{FIELD} * \left\rceil \frac{\left\rceil \log_2 (N_{FIELD} + 1) \right\lceil - 1}{N_{LUT}^{IN} - 1} \right\lceil. \tag{29}$$

З вищезазначених формул виходить, що реалізація дешифратора поля за допомогою N_{FIELD} LUT-елементів (за умови $]\log_2(N_{FIELD}+1)[\le N_{LUT}^{IN}$) більш доцільна з точки зору мінімізації апаратних витрат у схемі формування мікрооперацій у складі структури КМПК.

Для оцінки максимально можливої наповненості окремого поля вводяться поняття потужності та класу поля.

Визначення 1. Класом поля m (i_m) називається кількість LUT-елементів, необхідна для реалізації одного розряду дешифратора поля, яке вміщує N_{FIELD} мікрооперацій.

Клас поля m (i_m) розраховується за наступною формулою:

$$i_m = \left]\frac{]\log_2(N_{FIELD}+1)[-1}{N_{LUT}^{IN}-1}\right[. \quad (30)$$

Визначення 2. Потужністю поля m (P_m) називається максимальна кількість мікрооперацій, яку можливо увести до поля, для реалізації якого використовується завдана кількість LUT-елементів.

Потужність поля m (P_m) визначається за формулою:

$$P_m = 2^{N_{LUT}^{IN}+(i_m-1)*(N_{LUT}^{IN}-1)} - 1, \quad (31)$$

де i — клас поля m.

При розміщенні мікрооперацій по окремих полях бажано досягти максимального наближення кількості мікрооперацій у полі m (N_{FIELD}) до потужності поля (P_m). Це необхідно, щоб зменшити кількість комірок пам'яті у елементах LUT, що не використовуються.

Також необхідно прагнути того, щоб кількість мікрооперацій у окремому полі була більша за 2 ($N_{FIELD} > 2$), бо в іншому випадку кодування такого поля вироджується в унітарне кодування мікрооперацій і зменшення необхідних розрядів КП для такого поля не відбувається.

Для ефективного розміщення сумісних мікрооперацій по окремих полях необхідно ввести поняття показника сумісності окремої мікрооперації y_n (E_n) [3]:

$$E_n = |Q_n|, \quad (32)$$

де Q_n — масив мікрооперацій, що сумісні із мікрооперацією y_n. При цьому першою повинна розміщуватися ще не розміщена в жодному полі мікрооперація y_n, яка має найменше значення показника сумісності E_n серед нерозміщених мікрооперацій. Інші нерозміщенні мікрооперації розміщуються у порядку зростання показника своєї сумісності. Такий порядок розміщення мікрооперацій по полях дає змогу досягнути оптимального розміщення.

Мінімально можлива кількість полів сумісних мікрооперацій M дорівнює:

$$M = \max(|Y_1|,|Y_2|,...,|Y_Z|), \quad (33)$$

де $Y_1,...,Y_Z$ — набір мікрокоманд, що присутні у мікропрограмі керування.

Реалізація дешифратора окремого поля у базисі FPGA на елементах LUT з чотирма входами при виконанні умови $]\log_2(N_{FIELD}+1)[\le N_{LUT}^{IN}$ наведена на рис. 2.

Пропонується модифікація методики розміщення сумісних мікрооперацій по окремих полях за допомогою модифікованого алгоритму Шварта із завданням значення максимального класу поля (I_{MAX}).

Основні етапи модифікованого алгоритму Шварца розміщення сумісних мікрооперацій по окремих полях із завданим значенням максимального класу полів (I_{MAX}) мають наступний вигляд:

1. Визначення мінімально можливої кількості полів розбиття мікрооперацій: $M = \max(|Y_1|,|Y_2|,...,|Y_Z|)$.

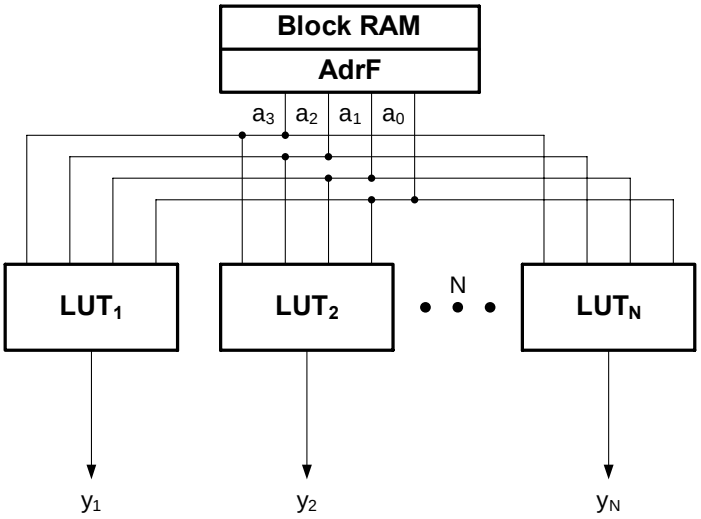

Рисунок 2 — Реалізація дешифратора поля сумісних мікрооперацій у базисі FPGA

2. Мікрооперації з мікрокоманди Y_{MAX} ($|Y_{MAX}| = M$) розміщуються по одній за порядком слідкування у M одноелементних полів $F_1,...,F_M$.

3. Розраховується потужність $P_m = 2^{N_{LUT}^{IN}+(i_m-1)*(N_{LUT}^{IN}-1)} - 1$ (де N_{LUT}^{IN} — кількість входів у елементі LUT, i_m — клас поля F_m) кожного поля з масиву $M_F = \{F_1,...,F_M\}$ за умови, що класи цих полів дорівнюють одиниці ($i_m = 1$).

4. Для усіх нерозміщених мікрооперацій будуються масиви мікрооперацій, що з ними сумісні (Q_n) та розраховуються їх показники сумісності (E_n). Першою розміщується мікрооперація із найменшим значенням показника сумісності.

5. Ще нерозміщена мікрооперація y_n у першу чергу розміщується у поле, яке не містить мікрооперацій, що не сумісні із y_n, яке задовольняє умові $|F_m| \le 2$ та яке має при цьому максимальну потужність ($P_m \to \max$).

6. У другу чергу мікрооперація y_n розміщується у поле, яке не містить мікрооперацій, що не сумісні із y_n, яке має мінімальний клас i_m, яке задовольняє умові $|F_m| < P_m$, та яке при цьому має мінімальну потужність ($P_m \to \min$).

7. У третю чергу мікрооперація y_n розміщується у поле, яке не містить мікрооперацій, що не сумісні із y_n та яке має мінімальний клас i_m ($i_m \to \min$).

8. Якщо є декілька альтернативних варіантів розміщення мікрооперації y_n у поля, які мають однакові параметри, то мікрооперація розміщується у те поле, яке має найменший показник сумісності усіх мікрооперацій, які вже знаходяться у цьому полі, із ще нерозміщеними мікроопераціями (Z_m).

Дослідження модифікованого алгоритму Шварца. Ефективність застосування модифікованого алгоритму Шварца загалом залежить від кількості мікрооперацій у мікропрограмі керування, від кількості мікрокоманд, від кількості мікрооперацій у окремих мікрокомандах та від частоти застосування окремих мікрооперацій у мікрокомандах.

У алгоритмах керування можливі випадки, коли розбиття сумісних мікрооперацій по окремих полях не є ефективним, і основні з них такі:

1. Якщо у мікропрограмі присутні $M_{MO} = \{y_0,...,y_Q\}$ мікрооперацій та якщо у якійсь з мікрокоманд $M_{MC} = \{Y_1,...,Y_Z\}$ у мікропрограмі використовуються $|M_{MO}| - 1$ з цих

мікрооперацій, то метод кодування полів сумісних мікрооперацій вироджується у метод унітарного кодування і немає сенсу у його застосуванні.

2. Якщо у алгоритмі керування у кожній мікрокоманді $M_{MC} = \{Y_1,...,Y_Z\}$ використовується лише одна мікрооперація з набору $M_{MO} = \{y_0,...,y_Q\}$, то усі ці мікрооперації сумісні одна з одною і метод кодування полів сумісних мікрооперацій вироджується у метод максимального кодування мікрооперацій, які розміщуються у одному полі.

Але у більшості алгоритмів, що застосовуються на практиці, використання модифікованого алгоритму Шварца для розподілення сумісних мікрооперацій по окремих полях і подальше кодування цих полів дозволяють значно скоротити апаратні витрати блочної пам'яті при реалізації операторної частини КМПК у базисі FPGA у порівнянні із унітарним кодуванням мікрооперацій. При цьому запропонований метод має більшу гнучкість при реалізації у базисі FPGA, аніж метод максимального кодування мікрокоманд.

Метод кодування полів сумісних мікрооперацій має більший ефект у композиції із методом вертикалізації граф-схеми алгоритму керування. У цьому випадку усі мікрооперації у мікропрограмі керування сумісні одна з одною і їх можна оптимально розподілити між окремими полями.

Вертикалізація граф-схеми алгоритму керування може бути здійснена із використанням двох способів:

1. Розбиття кожної операторної вершини у алгоритмі керування (за умови, коли $|Y(b_d)| > 1$) на $|Y(b_d)|$ (де b_d — операторна вершина з набору $B = \{b_1,...,b_D\}$ операторних вершин, що присутні у алгоритмі керування; $Y(b_d)$ — мікрокоманда, що записана у вершині b_d) операторних вершин, у кожній з яких розміщується одна мікрооперація з мікрокоманди $Y(b_d)$. При використанні цього способу збільшується загальна кількість операторних вершин у алгоритмі керування (за виключенням випадку, коли довжина мікрокоманд у кожній операторній вершині b_d з набору $B = \{b_1,...,b_D\}$ не перевищує одиниці ($|Y(b_d)| \le 1$)). При цьому кількість мікрооперацій, які необхідно розподілити по окремих полях, не змінюється.

2. Перекодування сигналів керування у об'єкті керування таким чином, щоб кожній не порожній операторній вершині у алгоритмі керування відповідав свій унікальний сигнал керування. При використанні цього способу вертикалізації граф-схеми алгоритму у ньому не збільшується кількість операторних вершин, але збільшуються апаратні витрати при реалізації об'єкту керування. Кількість мікрооперацій, які необхідно розподілити по окремих полях, розраховується за формулою (27).

При використанні композиції методів вертикалізації граф-схеми алгоритму та кодування полів сумісних мікрооперацій вдається розподілити мікрооперації на оптимальну кількість полів із завданою кількістю LUT-елементів, що повинні використовуватися для реалізації дешифраторів кожного поля. Загальна кількість полів, на які розбивається масив мікрооперацій, у цьому випадку розраховується за допомогою наступної формули:

$$M = \left\lceil \frac{N_y}{2^{N_{LUT}^{IN}+(i-1)*(N_{LUT}^{IN}-1)} - 1} \right\rceil, \qquad (34)$$

де M — необхідна кількість полів;

N_y — загальна кількість мікрооперацій у мікропрограмі керування;

N_{LUT}^{IN} — кількість входів у елементі LUT;

i — максимальна кількість елементів LUT, яка необхідна для реалізації одного розряду дешифратора полів (клас полів).

За умови $i = 1$ апаратні витрати на реалізацію дешифраторів полів досягають мінімального значення, бо за цієї умови для реалізації усіх полів знадобиться наступна кількість елементів LUT (N_{LUT}):

$$N_{LUT} = N_y . \quad (35)$$

А за умови $i > 1$ для реалізації усіх полів знадобиться наступна кількість елементів LUT:

$$N_{LUT} = i * N_y . \quad (36)$$

За умови $i = 1$ також мінімізується часова затримка на схемі формування мікрооперацій (t_{SFMO}):

$$t_{SFMO} = \max(t_{LUT1},...,t_{LUTN}), \quad (37)$$

де $t_{LUT1},...,t_{LUTN}$ — часові затримки на елементах LUT;

N — кількість мікрооперацій у мікропрограмі керування.

Нижче наведені діаграми дослідження розбиття сумісних мікрооперацій по окремих полях за наступних умов:

$M_{MO} = \{y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{10}\}$;

$N_{LUT}^{IN} = 2$;

$I_{MAX} = 1$.

На рис. 3 наведена діаграма залежності кількості виходів керуючої пам'яті (КП) від кількості мікрооперацій у найдовшій мікрокоманді (Y_{MAX}), за умови, що у інших $|M_{MO}| - |Y_{MAX}|$ мікрокомандах розташовано по одній мікрооперації з набору $M_{MO} \setminus Y_{MAX}$.

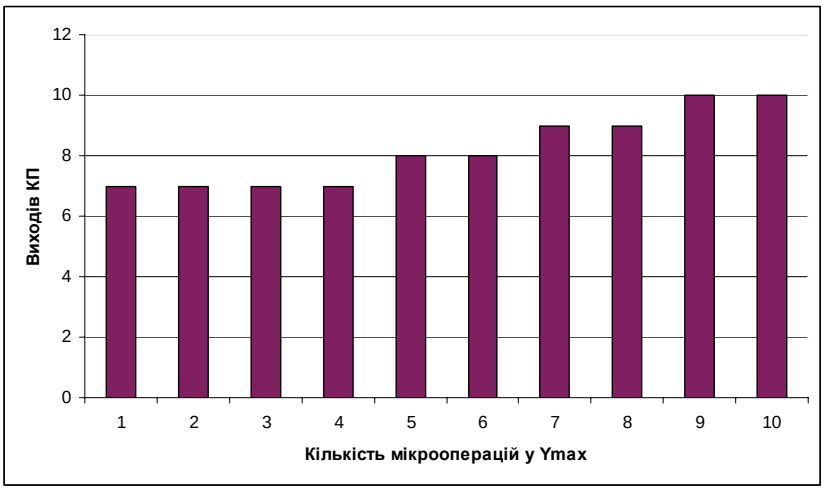

Рисунок 3 — Діаграма залежності кількості виходів КП від кількості мікрооперацій у найдовшій мікрокоманді (Y_{MAX})

На рис. 4 наведена діаграма залежності кількості полів сумісних мікрооперацій від кількості мікрооперацій у найдовшій мікрокоманді (Y_{MAX}).

На рис. 5 наведена діаграма залежності кількості елементів LUT, що необхідна для реалізації дешифраторів полів, від кількості мікрооперацій у найдовшій мікрокоманді (Y_{MAX}).

Висновки.

Розроблений модифікований алгоритм Шварца для розподілу сумісних мікрооперацій по окремих полях з урахуванням обмежень базису FPGA має велику швидкість розподілу мікрооперацій порівняно з іншими алгоритмами і при цьому по оптимальності розподілу наближується до показників цих алгоритмів. Велика швидкість розподілу сумісних мікрооперацій по полям досягається за рахунок того, що в цьому алгоритмі не перебираються усі існуючи варіанти розподілу, а оптимальність розбиття, що отримується у результаті роботи алгоритму, забезпечується введенням до базового алгоритму Шварца

елементів з методу гілок і меж та іншими додатковими етапами, які покращують оптимальність розбиття сумісних мікрооперацій по окремих полях. Швидкість функціонування модифікованого алгоритму Шварца менша, аніж базового алгоритму. Але при цьому ефективність розподілу мікрооперацій за цим алгоритмом значно вища, аніж за базовим алгоритмом Шварца.

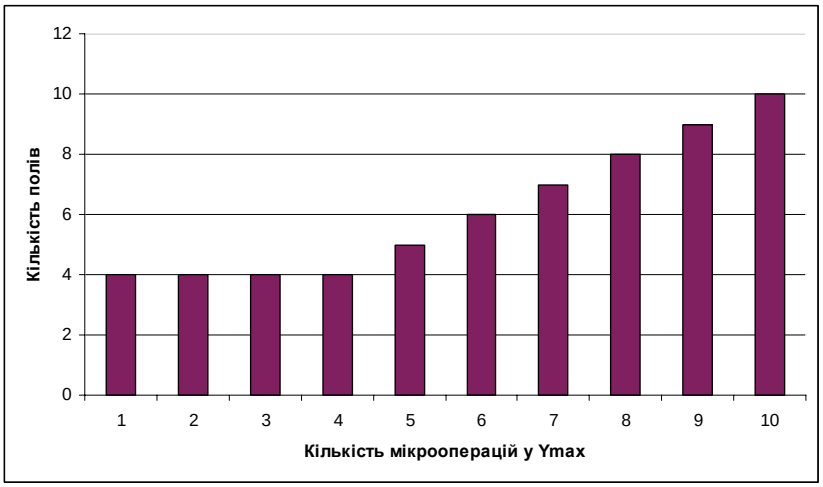

Рисунок 4 — Діаграма залежності кількості полів сумісних мікрооперацій від кількості мікрооперацій у найдовшій мікрокоманді (Y_{MAX})

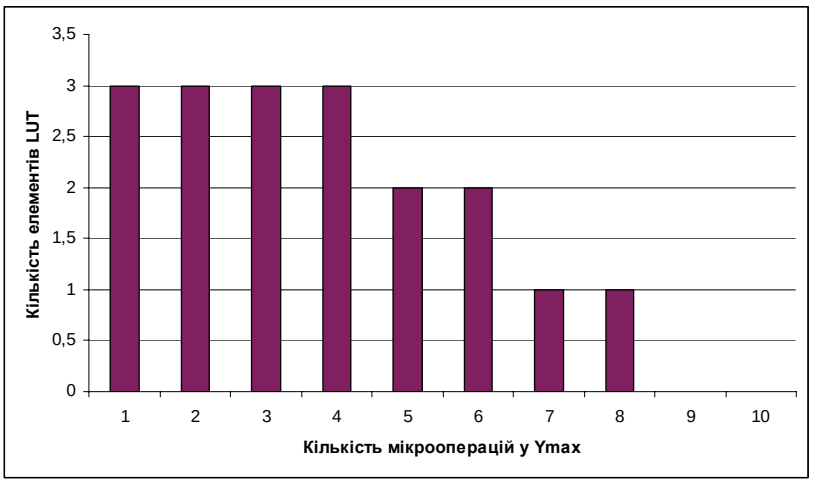

Рисунок 5 — Діаграма залежності кількості елементів LUT, що необхідна для реалізації дешифраторів полів, від кількості мікрооперацій у найдовшій мікрокоманді (Y_{MAX})

Література

1. Соловьев В.В. Проектирование цифровых систем на основе программируемых логических интегральных схем. — М.: Горячая линия-Телеком, 2001. — 636 с. ил.
2. Кузелин М.О., Кнышев Д.А., Зотов В.Ю. Современные семейства ПЛИС фирмы Xilinx. Справочное пособие. — М.: Горячая линия-Телеком, 2004. — 440 с.: ил.
3. А.А. Баркалов, А.В. Палагин. Синтез микропрограммных устройств управления. — К., 1997. — 136 с.: ил.

Здано в редакцію:
12.03.2009р.

Рекомендовано до друку:
д.т.н, проф. Башков Є.О.