УДК 004.07

Р.Б. ДУНЕЦЬ, Д.Я. ТИХАНСЬКИЙ

Національний університет «Львівська політехніка», Україна

ПРОБЛЕМИ ПОБУДОВИ ЧАСТКОВО РЕКОНФІГУРОВАНИХ СИСТЕМ НА ПЛІС

Розглянуто проблеми побудови частково реконфігурованих систем на базі ПЛІС, котрі виникають при побудові таких систем. Відсутність певного стандарту проектування часткової реконфігурації, як в межах різних типів ПЛІС одного виробника, так і для різних виробників суттєво ускладнює процес проектування. На прикладі простого проекту проведено аналіз процесу проектування частково реконфігурованої системи та показано шляхи її вдосконалення. Звернено увагу на час реконфігурування системи, який є найбільшим чинником, що гальмує розвиток реконфігурованих систем на базі ПЛІС.

Ключові слова: *ПЛІС, часткова реконфігурація, модульний проект, час реконфігурації.*

Вступ

Відомо [1], що часткова реконфігурація - це здатність перепрограмовувати ПЛІС частинами, в той час коли решта системи залишається незмінною. Тобто, бітова послідовність для часткової реконфігурації завантажує тільки частину проекту в ПЛІС.

Розрізняють статичну та динамічну часткову реконфігурації. Статична часткова реконфігурація - це перепрограмування частини ПЛІС в процесі її перезавантаження [2], а динамічна часткова реконфігурація – це перепрограмування частини ПЛІС, залишаючи решту її частин працюючими. Очевидно, що динамічна часткова реконфігурація є набагато ефективнішою, оскільки вона дозволяє перепрограмовувати ПЛІС не зупиняючи її роботи для перепрограмування. Надалі під частковою реконфігурацією будемо розуміти динамічну часткову реконфігурацію.

Огляд літературних джерел

Фірма Xilinx є однією з тих провідних фірм, ПЛІСи якої, починаючи з Virtex, Spartan-2 та закінчуючи останніми моделями, підтримують часткову реконфігурацію. Треба зазначити, що у ПЛІС фірми Xilinx часткова реконфігурація може бути реалізована одним із двох методів: модульна реконфігурація (module-based) та диференційована реконфігурація (difference-based) [3]. Модульна часткова реконфігурація використовується для окремих, наперед визначених частин ПЛІС, що можуть бути реконфігурованими. Такі частини отримали назву "реконфігурований модуль". Реалізація модульного методу потребує розробки загальної топології кристалу, з подальшим визначенням місць фізичного розташування всіх реконфігурованих модулів. При необхідності проведення реконфігурації, весь реконфігурований модуль перезаписується новою бітовою послідовністю. Диференційована реконфігурація базується на незначних змінах ПЛІС, і використовується тільки для невеликих змін у ПЛІС. Згенерована бітова послідовність для цього методу містить тільки відмінності між станами до і після реконфігурації. Хоч диференційована реконфігурація перепрограмовує ПЛІС за менший час, оскільки перезавантажуються тільки зміни, проте кількість задач, що може бути розв’язана цим методом, є меншою у порівнянні з кількістю задач для модульної реконфігурації. З цих міркувань надалі будемо розглядати тільки модульну реконфігурацію [4].

Постановка задачі

Модулі, зміна алгоритму роботи яких не передбачається, а відтак вони не будуть зазнавати реконфігурації, відносять до статичних модулів. Решта модулів ПЛІС, які можуть бути спроектовані як частково реконфігуровані, відносять до реконфігурованих модулів. Очевидно, що вони потребують додаткової уваги на всіх стадіях розробки – від синтезу до реалізації. Загалом для всієї ПЛІС треба синтезувати і створити окремі бітові послідовності для кожного реконфігурованого та статичного модуля, а тоді бітова послідовність всієї системи буде включати всю логіку статичних модулів та по одному із варіантів реалізації логіки роботи кожного реконфігурованого модуля.

Часткова реконфігурація може бути здійснена

двома шляхами: через JTAG порт з підєднанням до ПК або через внутрішню логіку чи за допомогою процесора на платі, наприклад, через процесор PowerPC у Virtex II Pro чи інших, більш новіших, ПЛІС. Часткова реконфігурація ПЛІС через внутрішню логіку, що часто називається самореконфігуруванням, має перевагу в тому, що, по-перше, не потребує зовнішнього комп'ютера, а, по-друге, має вищу швидкодію, оскільки вся інформація стосовно часткового реконфігурування зберігається в пам'яті і записується через внутрішній конфігураційний порт (Internal Configuration Access Port (ICAP)) у процесі реконфігурування необхідного модуля ПЛІС. Недоліком цього шляху є низька оперативність проведення модифікації алгоритму роботи реконфігурованих модулів, оскільки для модифікації прийдеться все-таки підключати ПК із спеціалізованим програмним забезпеченняя і перезаписувати JTAG порт бітову послідовність.

Стосовно другого шляху здійснення реконфігурації, то у ньому переваги і недоліки по відношенню до першого шляху міняються місцями. Крім того, другий шлях дозволяє більш повно дослідити проблеми, що виникають в процесі реалізації часткової реконфігурації. З цією метою нижче розглянемо приклад елементарного проекту, в якому буде реалізована часткова реконфігурація ПЛІС через JTAG порт за допомогою зовнішнього ПК.

Приклад проекту

Приклад проекту дуже простий - це 8-ми бітний реконфігурований лічильник. При початковому завантаженні ПЛІС лічильник повністю завантажується двома модулями - статичним та однією з двох версій динамічного модуля. Після завантаження ПЛІС лічильник починає роботу, яка полягає в інкрементуванні регістра. Оскільки заданий лічильник реконфігурований, то є можливість змінювати алгоритм його роботи без залучення додаткової логіки на ПЛІС і, найголовніше, не зупиняючи роботи решти складових елементів системи. Звісно, що на реконфігурування затрачається певний час.

На вхід даного лічильника подаються тільки тактові імпульси, а, відтак, не має іншої можливості впливати на логіку його роботи, окрім його реконфігурування.

Результатом роботи лічильника є сигнал на виході Result. Аналізуючи стан сигналу на цьому виході, можна робити висновок про правильну роботу лічильника, а також розрізняти який з реконфігурованих модулів завантажений на даний момент часу.

У процесі реконфігурації лічильника, вихід Result переходить у високо імпедансний стан, оскільки він зв'язаний з реконфігурованим модулем. Вихід Q служить для перевірки працездатності решти системи підчас реконфігурування. На цей вихід подаються дані з іншого лічильника, який не є зв'язаний з реконфігурованим модулем. Оскільки, той лічильник є незалежний від процесу реконфігурації, то він не зупиняється підчас реконфігурування.

На рис. 1 зображено загальну схему проекту. Лічильник складається з двох модулів – статичного і динамічного

Статичний модуль, який називається myRegister - це регістр, який зберігає дані.

Динамічний модуль, який називається incrementer, - це 8-ми бітний суматор, який формує нові дані для регістра. Цей модуль реалізований у двох версіях оскільки він є реконфігурованим. У першій версії - це суматор з константою "1", а у другій - це суматор з константою "10".

У проекті також є два ідентичних модулі bm8, які забезпечують інтерфейс і об'єднують в собі два шинних макроси. Необхідність цих двох модулів обумовлюється тим, що наявні шинні макроси, які пропонує фірма Xilinx, є лише 4-х бітні. Оскільки шинні макроси є одно направленими, тому один з них орієнтований на передачу сигналів з ліва-направо, а другий - з права-на-ліво.

На рис. 2 показано схему проекту в САПР Aldec Aactive-HDL.

Для реалізації проекту був вибраний один із доступних авторам на момент проведення експеременту кристал Xilinx VirtexE, а саме XCV300E_PQ240.

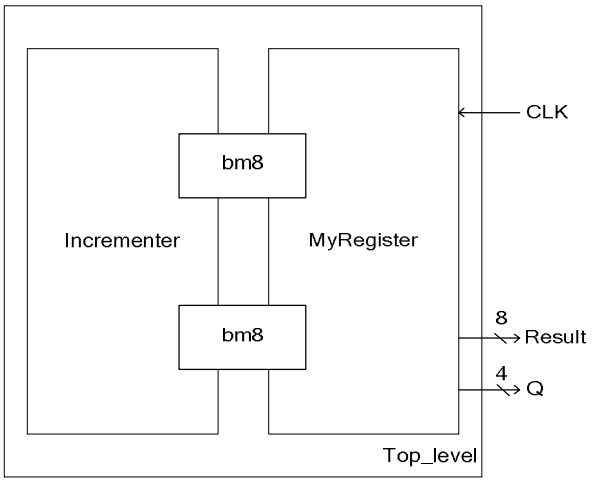

Рис. 1. Загальна схема проекту реконфігурованого лічильника

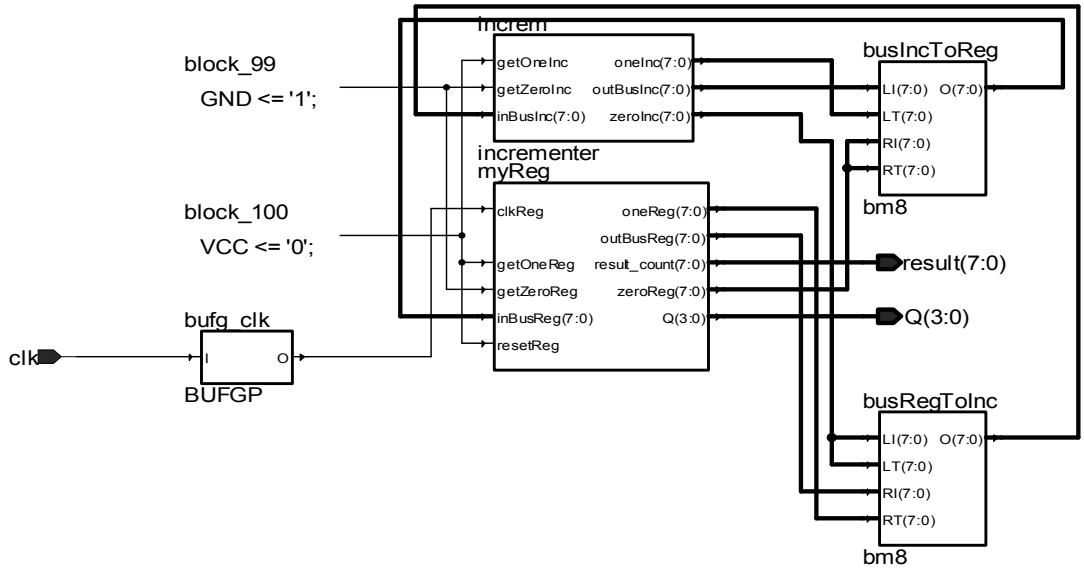

Рис. 2. Схема проекту реконфігурованого лічильника в Aldec Active-HDL

Шинний макрос

У процесі реалізації частково реконфігурованого проекту особлива увага повинна бути приділена шинному макросу, наявність якого в проекті є обов'язковою. Якщо модулі не є частково реконфігурованими, тоді лінії зв'язку між ними можуть бути розведені в кристалі довільним чином, а на їх довжину не накладаються якісь обмеження. Якщо модуль є частково реконфігурованим, то тоді він має бути розведений на кристалі окремо від інших модулів. У цьому випадку, щоб встановити зв'язок між двома частково реконфігурованими модулями, потрібно використовувати одні і ті ж самі фізичні зв'язки, які, як правило, є спеціалізованими лініями зв'язку. Власне таку функцію комутування забезпечує шинний макрос. Такий макрос є апаратним макросом і він завжди при синтезі проекту розміщається в одне і те саме місце на кристалі, утворюючи канал зв'язку для інших модулів.

На рис. 3 наведено схему шинного макроса. Він використовує на один біт один буфер з 3-ма станами (TBUF), який зв'язує лівий вхід з правим. В нашому макросі використано вісім буферів з 3-ма станами, які забезпечують 4-м бітам інформації переміщатись з ліва-на-право або з права-на-ліво.

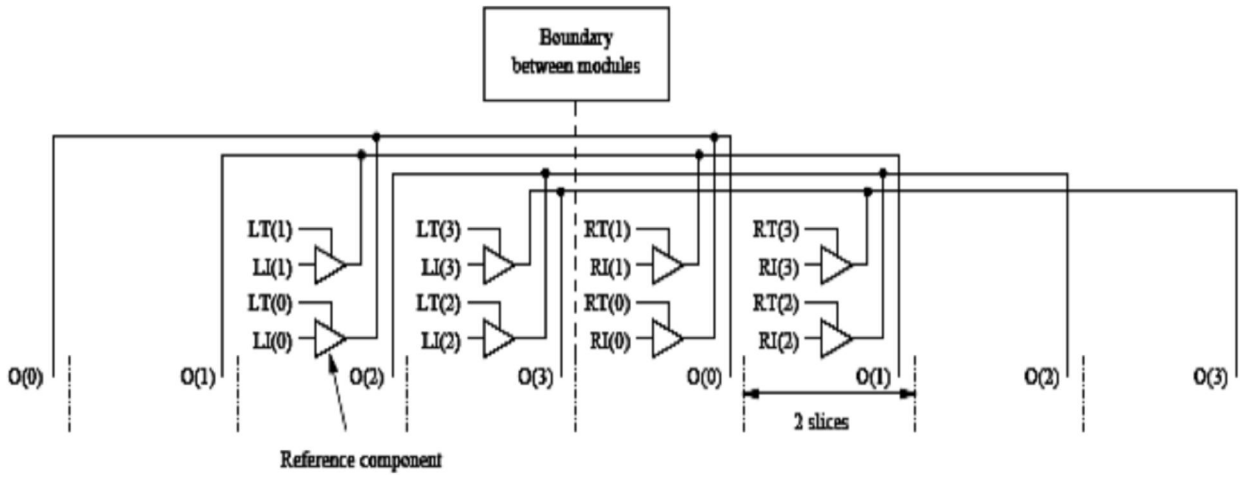

Рис. 3. Шинний макрос

Цей макрос має 5 входів/виходів: LI і RI - ліва і права 4-х бітова вхідна шина, O – 4-х бітова вихідна шина, LT і RT – ліва і права конфігураційна шина, котра налаштовує напрям для кожного зв'язку шинного макроса. Звичайно LT і RT повинні приймати протилежні значення для кожного біта. Якщо цієї умови не буде дотримано, тоді може статись коротке замикання. З цих міркувань було би доцільним створення спеціального інтерфейсу для з'єднання динамічних модулів.

Аналіз програмного забезпечення

Щоб реалізувати цей проект необхідно використовувати спеціалізоване програмне забезпечення Xilinx, а саме Xilinx ISE. Фірма Xilinx досить швидко оновлює цей програмний продукт додаючи до нього нові можливості, що є позитивним моментом. Але при оновлені програмного забезпечення фірма Xilinx не тільки вносить нові можливості, але, на жаль, вилучає старі рішення і знімає їх підтримку у нових ПЛІС. Це не тільки принцип ієрархічної спадковості, але й створює цілу низку труднощів процесу проектування, зокрема затрати часу на освоєння нових можливостей і перевірку їх застосування для кожного конкретного типу ПЛІС. Так у нашому випадку проект часткової реконфігурації для VirtexE передбачає використання модульного підходу підчас проектування [5], але Xilinx в останніх версіях Xilinx ISE вилучила модульний підхід для проектування ПЛІС, вважаючи його застарілим і замінила його методом проектування на основі розділів (Partitions) пори те, що ціла низка ПЛІС цієї фірми, зокрема Virtex-4, достосована до модульного підходу.

Звісно можна не використовувати модульний підхід, а здійснювати часткову реконфігурацію за допомогою потужного продукту PlanAhead, але жодна версія PlanAhead не підтримує чіп VirtexE.

Крім того, в нових САПР Xilinx ISE змінено так звану систему обмежень (Constraint System) і відповідно розміщувати модулі на ПЛІС, користуючись підходами і макросами, які запропоновані в документах опису часткової реконфігурації від Xilinx, в нових САПР не можливо. У результаті, щоб реалізувати проект часткової реконфігурації на чіпі VirtexE необхідно використовувати далеко не нову версію 6.3 Xilinx ISE.

Таким чином, виходить, щоби досліджувати проблеми побудови часткової реконфігурації необхідно відмовитись від використання давніших чіпів і використовувати натомість лише найновіші. На думку авторів, досліджувати проблеми часткової реконфігурації необхідно починати хоча б з Virtex-4, не дивлячись на те, що ця модель є давніша, але виробник досить непогано її підтримує, саме в плані часткової реконфігурації. Всі нові приклади реалізації часткової реконфігурації виробник реалізовує саме на чіпі Virtex-4. Крім того, виробник у анонсованому своєму новому САПР Xilinx ISE 12, який на момент написання статті ще не був випущений, обіцяє підтримку часткової реконфігурації на якісно новому рівні. Підтримка Virtex-4 у пакеті Xilinx ISE 12 також буде здійснюватися.

Отже, враховуючи досить велику кількість вже реалізованих прикладів часткової реконфігурації саме під Virtex-4 і підтримку цього чіпа виробником майбутніми САПР ,а також нижчу ціну на Virtex-4, порівняно з новішими ПЛІС, можна зробити висновок, що, цей чіп є досить непоганим вибором для дослідження та реалізації часткової реконфігурації, і навіть реконфігуровані обчислення (Reconfigurable Computing).

Підсумовуючи сказане, можна зробити висновок, що відсутність певного стандарту проектування часткової реконфігурації, як в межах різних типів ПЛІС одного виробника, так і для різних виробників, приводить до збільшення часу проектування та допущення помилок.

Аналіз бітових послідовностей

Дуже вузьким місцем при використанні реконфігурації, як повної, так і часткової є час реконфігурації. Враховуючи, що сучасні ПЛІС містять мільйони конфігураційних блоків, тому час реконфігурації може бути досить великим (кілька секунд).

Оскільки частково реконфігурована область займає тільки певну частину ПЛІС, то, очевидно, конфігураційний файл для часткової реконфігурації має бути меншим, ніж конфігураційний файл для всієї системи. Відповідно час на часткову реконфігурацію має бути менший ніж на реконфігурацію всієї ПЛІС.

Для VirtexE XCV300E повний конфігураційний файл займає приблизно 235 КБ. Файл частково реконфігурованого модуля займає приблизно 41КБ. Файл статичного модуля займає приблизно 71КБ. Відповідно час на реконфігурування модуля буде приблизно в 5 раз меншим. Також слід зауважити, що розмір файлу для часткової реконфігурації залежить від складності модуля, тобто використаної логіки і від площі, яку займає модуль на кристалі. В свою чергу повний файл конфігурації завжди є однаковим незалежно від складності проекту.

Висновки

1. Відсутність певного стандарту проектування часткової реконфігурації, як в межах різних типів ПЛІС одного виробника, так і для різних виробників суттєво ускладнює процес проектування.

2. Виробник постійно покращуючи використання часткової ре конфігурації, на жаль, змінює підходи до проектування, що призводить до проблем, пов’язаних із використанням програмного забезпеченням, не відповідності документів та з підтримкою різних чіпів.

3. Відсутність спеціального інтерфейсу для з’єднання динамічних модулів ускладнює процес

проектування та збільшує ймовірність прийняття некоректних інженерних рішень.

4. Проектування частково реконфігурованих систем вимагає приділяти особливу увагу часу реконфігурації, оскільки в кінцевому випадку може виявитись, що час реконфігурації не відповідає заданим вимогам до систем.

Отже, при застосуванні часткової реконфігурації виникає багато проблем, які потрібно в подальшому розв'язувати, але можливості, які відкриваються у частково реконфігурованих систем, варті проведення подальшого дослідження.

Література

1. Dynamic partial reconfiguration on FPGA [Електронний ресурс]. – Режим доступу к ресурсу: http://www.vlsi-world.com/content/view/48/47/.

2. Zeineddini A.H.S. Secure Partial Reconfiguration of FPGAs / A.H.S .Zeineddini // George Mason University [Електронний ресурс]. – Режим доступу к ресурсу: http://ece.gmu.edu/courses/Crypto_resources/web_resources/theses/GMU_theses/Zeineddini/Zeineddini_Summer_2005.pdf.

3. Partial Reconfiguration, Xilinx, Development System Reference Guide, Chapter 5 [Електронний ресурс]. – Режим доступу к ресурсу: http://www.xilinx.com/itp/xilinx8/de/dev/partial.pdf.

4. Kao C. Benefits of Partial Reconfiguration / C. Kao // Xcell Journal Fourth Quarter. – 2005. – P. 65-67.

5. Modular Design, Xilinx, Development System Reference Guide, Chapter 4 [Електронний ресурс]. – Режим доступу к ресурсу: http://www.xilinx.com/itp/xilinx10/books/docs/dev/dev.pdf.

6. Custodio E. Self-Healing Partial Reconfiguration of an FPGA / E. Custodio, B. Marsland [Електронний ресурс]. – Режим доступу к ресурсу: http://www.wpi.edu/Pubs/E-project/Available/E-project-042607-133123/unrestricted/GD_D07_MQP_final_draft.pdf.

7. Mermoud G. A Module-Based Dynamic Partial Reconfiguration Tutorial / G. Mermoud // Ecole Polytechnique Fédérale de Lausanne, 2004.

8. Thorvinger J. Dynamic Partial Reconfiguration of an FPGA for Computational Hardware Support / J. Thorvinger // Lund Institute of Technology, 2004.

9. Дунець Р.Б. Дослідження часткової реконфігурації ПЛІС / Р.Б. Дунець, Д.Я. Тиханський // Радіоелектронні і комп'ютерні системи. – №6(40). – 2009. – С. 240-244.

Надійшла в редакцію 28.01.2010

Рецензент: д-р техн. наук, проф. А.А. Мельник, Национальный университет «Львовская политехника», Украина.

ПРОБЛЕМЫ ПОСТРОЕНИЯ ЧАСТИЧНО РЕКОНФИГУРИРУЕМЫХ СИСТЕМ НА ПЛИС

Р.Б. Дунец, Д.Я. Тиханский

Рассмотрены проблемы построения частично реконфигурируемых систем на базе ПЛИС, возникающие при построении таких систем. Отсутствие определенного стандарта проектирования частичной реконфигурации, как в пределах разных типов ПЛИС одного производителя, так и для разных производителей существенно усложняет процесс проектирования. На примере простого проекта проведен анализ процесса проектирования частично реконфигурируемой системы и показаны пути ее усовершенствования. Обращено внимание на время реконфигурирования системы, которое является основным определяющим фактором, тормозящим развитие систем на базе ПЛИС.

Ключевые слова: ПЛИС, частичная реконфигурация, модульный проект, время реконфигурации.

PROBLEMS OF PARTIALLY RECONFIGURABLE FPGA-BASED SYSTEM DESIGN

R.B. Dunets, D.Y. Tykhanskyy

This paper considered design issues of partially reconfigurable FPGA-based systems. The lack of definite standard of partial reconfiguration development both for different types of one vendor FPGA and for different vendors considerably complicates the development process. Partial reconfiguration process analysis has been done using a simple example. Paper shows the ways for improvement the design flow. Special attention devoted to the reconfiguration time, which is the main slow down factor in evolution of reconfigurable FPGA based system.

Keywords: FPGA, partial reconfiguration, modular design, reconfiguration time.

Дунець Роман Богданович – д-р техн. наук, доцент, завідувач кафедри спеціалізованих комп'ютерних систем, Національний університет «Львівська політехніка», Львів, Україна, e-mail: dunets@polynet.lviv.ua.

Тиханський Дмитро Ярославович – аспірант кафедри спеціалізованих комп'ютерних систем, Національний університет «Львівська політехніка», Львів, Україна, e-mail: tyxdima1@gmail.com.