УДК 004.274, 004.383

В. А. Мельник
Національний університет “Львівська політехніка”,
кафедра безпеки інформаційних технологій

АПАРАТНА БАГАТОЗАДАЧНІСТЬ У КОМП’ЮТЕРНИХ СИСТЕМАХ НА ОСНОВІ ЧАСТКОВО РЕКОНФІГУРОВНИХ ПЛІС

Визначено базові принципи реалізації апаратної багатозадачності в реконфігуровних комп’ютерних системах, побудованих на основі ПЛІС з динамічним частковим реконфігуруванням. Запропоновано структуру платформи для реалізації апаратної багатозадачності в частково реконфігуровній ПЛІС. Розглянуто концепцію віртуальних апаратних засобів, а також питання перемикання контексту і переміщення задач у частково реконфігуровній ПЛІС.

Ключові слова: апаратна багатозадачність, ПЛІС, часткове реконфігурування, віртуальні апаратні засоби.

HARDWARE MULTITASKING IN COMPUTER SYSTEMS BASED ON PARTIALLY RECONFIGURABLE FPGAS

In the article the basic principles of hardware multitasking in the reconfigurable computer systems, based on partially reconfigurable FPGAs, are identified. The structure of the platform to implement hardware multitasking in partially reconfigurable FPGA is proposed. The concept of Virtual Hardware and the questions of the context switch and task relocation in partially reconfigurable FPGA are disclosed.

Key words: hardware multitasking, FPGA, partial reconfiguration, Virtual Hardware.

Вступ

Переважна більшість сучасних комп’ютерів загального призначення підтримують багатозадачний режим роботи. В операційних системах багатокористувацьких комп’ютерів багатозадачність необхідна для організації паралельної роботи користувачів. Навіть портативні мобільні пристрої – планшети і смартфони є багатозадачними.

Фактично всі сучасні операційні системи (ОС) є багатозадачними. Зокрема, такими є всі наведені нижче ОС, причому, за даними рейтингу *Top 500* [1], частка використання у суперкомп’ютерах ОС *Linux* та *Unix* станом на листопад 2014 р. становить 99,6 %, а *Microsoft Windows* – 0,2 %. Серед ОС персональних комп’ютерів, за даними *netapplications.com*, у 2014 р. лідером був *Microsoft Windows* (7, 8, *XP*, *Vista*) з часткою 91,45 %, а решта 8,55 % належали *Linux* та *Unix*. У вбудованих комп’ютерних системах, за даними 2012 р. [2], у використанні переважають *Linux*-подібний *Android* (29,44 %), *Unix*-подібний *QNX* (4,29 %) та *Microsoft Windows Embedded Compact* (11,65 %). У системах реального часу [3] – *Android* (19,3 %) та системи від *Microsoft*: *Windows XP Embedded* та *Windows Embedded Compact* (разом 35,8 %).

Багатозадачність забезпечується як на рівні апаратних засобів, так і на рівні системного програмного забезпечення комп’ютера. На рівні апаратних засобів багатозадачність забезпечується використанням одного чи більшої кількості універсальних процесорів, які розв’язують задачі за вказівками програм почергово, виділяючи кожній з них короткий проміжок часу і цим створюючи ефект їх одночасного виконання.

Підвищення продуктивності в багатозадачних комп'ютерних системах досягають екстенсивно – збільшуючи кількість процесорів, ємність пам'яті тощо, що, очевидно, пов'язано зі збільшенням їхньої вартості та споживаної потужності. Іншими підходами до підвищення продуктивності є спеціалізація комп'ютерних засобів і апаратна інтерпретація виконуваних ними алгоритмів. Відомо, що ці підходи забезпечують зростання продуктивності на 1–4 порядки. Однак їх застосування забезпечує високі показники продуктивності комп'ютерних систем лише на охоплених ними класах задач.

Вирішенням проблеми ефективного поєднання універсальності й апаратної інтерпретації виконуваних алгоритмів є побудова комп'ютерних систем з використанням реконфігуровних компонентів. Такі комп'ютерні системи називають реконфігуровними. Їхні структуру та функції можна заданим способом переналаштувати з метою врахування структурних та обчислювальних характеристик виконуваних алгоритмів. Практичне впровадження технології реконфігуровних обчислень уможливилось з появою ПЛІС – програмовних логічних інтегральних схем (англ. *FPGA – Field Programmable Gate Array*) високого ступеня інтеграції. Фірма *Xilinx* на початку 80-х років минулого століття першою почала масове виготовлення ПЛІС, спочатку складністю близько тисячі, а сьогодні – десятки мільйонів логічних вентилів.

Впровадження багатозадачності в реконфігуровні комп'ютерні системи (РККС) все ще залишається проблемним питанням. Це пов'язано з особливістю процесу зміни в них розв'язуваної задачі.

Загалом процес зміни розв'язуваної задачі в комп'ютерній системі передбачає: 1) зупинку розв'язування поточної задачі й запам'ятовування стану комп'ютерних засобів з тим, щоб пізніше можна було продовжити її розв'язування з місця зупинки; і 2) встановлення в початковий стан або відновлення стану комп'ютерних засобів відповідно для початку чи продовження розв'язування наступної задачі. Стан комп'ютерних засобів у момент зупинки розв'язування задачі називають контекстом задачі. В традиційних комп'ютерних системах він складається з вмісту регістрів процесора, стану програмного лічильника, режиму роботи процесора, захисту пам'яті тощо. В РККС контекст також містить конфігурацію ПЛІС і поточний вміст її запам'ятовуючих елементів.

Проблема зміни розв'язуваної задачі в РККС зумовлена передусім значною тривалістю реконфігурування ПЛІС. Зміна активного процесу (програми в стані виконання) в універсальному процесорі здійснюється перемиканням контексту і триває 1–4 мкс, тоді як реконфігурування ПЛІС – від кількох десятків до кількох сотень мс. І більше, залежно від завантаженості комп'ютерної системи перемикання контексту в процесорі може виконуватись десятки чи сотні разів за секунду [4–6], тобто кожні 100–10 мс, а це є співмірним з тривалістю реконфігурування ПЛІС. Отже, реконфігурування ПЛІС не може бути виконане разом із перемиканням контексту під час зміни розв'язуваної задачі в РККС, а в деяких випадках нівелює можливе прискорення виконання програми, отримане завдяки застосуванню ПЛІС.

Разом з тим, останніми роками з'явились та набувають все більшого поширення ПЛІС, що дають змогу виконувати реконфігурування не тільки повністю, але й у деякій своїй частині, не змінюючи конфігурації решти обладнання. Такі ПЛІС отримали назву частково реконфігуровних (ЧР, анг. *Partially Reconfigurable*). Тривалість часткового реконфігурування ПЛІС значно менша, ніж повного, тому їх застосування відкриває низку нових можливостей у комп'ютерній техніці, зокрема дає змогу організувати апаратну багатозадачність у межах ПЛІС, а отже, й у реконфігуровних комп'ютерних системах.

Аналіз досліджень та публікацій

До перших теоретичних праць, в яких висвітлено підхід до реалізації апаратної багатозадачності із застосуванням пристроїв реконфігуровної логіки, належать роботи [7] і [8], опубліковані ще в кінці 90-х років минулого століття. У цих самих роботах також описано концепцію віртуальних апаратних засобів (анг. *Virtual Hardware*), яка безпосередньо пов'язана з апаратною багатозадачністю. Приклади реалізації концепції віртуальних апаратних засобів у багатозадачних реконфігуровних комп'ютерних системах на основі ЧР ПЛІС описано в роботах [9–11]. До інших питань, пов'язаних з реалізацією апаратної багатозадачності в ЧР ПЛІС, також належать перемикання контексту [12–15] і переміщення задач [16–22].

Постановка проблеми

Як показує аналіз досліджень та публікацій за тематикою апаратної багатозадачності в РККС, переважна більшість праць зосереджують увагу на окремих аспектах її реалізації. Проте комплексно питання організації апаратної багатозадачності, формулювання принципів її реалізації і завдань, з цим пов'язаних, досі не розглядались. Тому актуальним завданням є висвітлення підґрунтя концепції апаратної багатозадачності й, на основі узагальнення і систематизації результатів наукових досліджень у цьому напрямі, визначення принципів її реалізації в РККС, основаних на ЧР ПЛІС, чого й стосується ця стаття.

1. Поняття апаратної багатозадачності

У комп'ютерній техніці багатозадачністю називають таку організацію функціонування комп'ютерної системи, яка забезпечує паралельне розв'язування нею багатьох задач упродовж певного проміжку часу. Паралельне розв'язування задач не завжди означає, що вони розв'язуються одночасно, адже одночасне розв'язування потребує наявності в комп'ютері кількох процесорів. Тому розділяють справжній (дійсний) паралелізм, коли декілька процесорів одночасно розв'язують різні задачі, та псевдопаралелізм, коли один процесор розв'язує різні задачі по черзі, виділяючи кожній з них короткий проміжок часу і, отже, створюючи ефект їх одночасного розв'язування. Останній підхід дає можливість паралельно розв'язувати більшу кількість задач, ніж є процесорів у системі. Обидва підходи сьогодні часто суміщають і застосовують не тільки у високопродуктивних, але й у персональних і навіть у планшетних комп'ютерах.

Багатозадачність називають апаратною, якщо для розв'язування задач використовують спеціалізовані апаратні засоби, які працюють паралельно[1]. Базовий підхід до реалізації апаратної багатозадачності передбачає застосування ПЛІС і програмних засобів імплементації у них електронних цифрових пристроїв. Теоретичні праці (серед яких передусім варто згадати роботи [7] і [8]) з цього питання з'явилися ще в кінці 90-х років минулого століття. Якісно новий етап практичного втілення апаратної багатозадачності настав з появою ПЛІС, що підтримують динамічне часткове реконфігурування (далі в тексті під терміном ЧР розумітимемо саме динамічний тип ЧР). Сьогодні цей напрям активно розвивається.

За результатами опрацювання низки літературних джерел, зокрема [12–15, 17, 22, 24, 25], та узагальнення і систематизації результатів наукових досліджень, можна виокремити такі основні принципи реалізації апаратної багатозадачності в РККС, побудованих на основі ЧР ПЛІС:

1) у ЧР ПЛІС розміщено множину реконфігуровних регіонів, у кожному з яких у визначений момент часу може функціонувати один замінний модуль;

2) задачу (або частину задачі) подають у РККС у вигляді програмної моделі замінного модуля, який компілюють у файл часткової конфігурації ПЛІС і завантажують до неї у визначений реконфігуровний регіон за командою ініціалізації розв'язування цієї задачі;

3) замінний модуль може використовувати від одного до N_{RP} реконфігуровних регіонів, де N_{RP} – кількість реконфігуровних регіонів у ЧР ПЛІС;

4) у ЧР ПЛІС може одночасно функціонувати до N_{RP} замінних модулів;

5) у разі потреби замінні модулі можуть використовувати реконфігуровний регіон RP_k у режимі часового мультиплексування;

6) у РККС на основі ЧР ПЛІС реалізовано:
   а) засоби завантаження часткових конфігурацій;
   б) механізми перемикання контексту в ПЛІС і переміщення задач між реконфігуровними регіонами;
   в) концепцію віртуальних апаратних засобів.

[1] Не треба плутати терміни «апаратна багатозадачність» і «апаратна багатопотоковість» (анг. *Hardware Multithreading*) – останнім позначають спосіб реалізації багатозадачного режиму роботи в універсальних процесорах [23].

2. Платформа для реалізації апаратної багатозадачності в ЧР ПЛІС

Крім множини реконфігуровних регіонів, у ЧР ПЛІС також потрібно розмістити допоміжні засоби для підтримки апаратної багатозадачності, які дадуть можливість швидко завантажувати часткові конфігурації і перемикати контекст. Разом із множиною реконфігуровних регіонів ці засоби утворюють платформу для реалізації апаратної багатозадачності в ЧР ПЛІС. Структуру такої платформи, що основана на перерахованих вище принципах реалізації апаратної багатозадачності, наведено на рис. 1. Тут засоби підтримки апаратної багатозадачності розміщено у статичному регіоні ПЛІС. Це дає можливість створити платформу за методологією проектування електронних пристроїв з невизначеними замінними модулями.

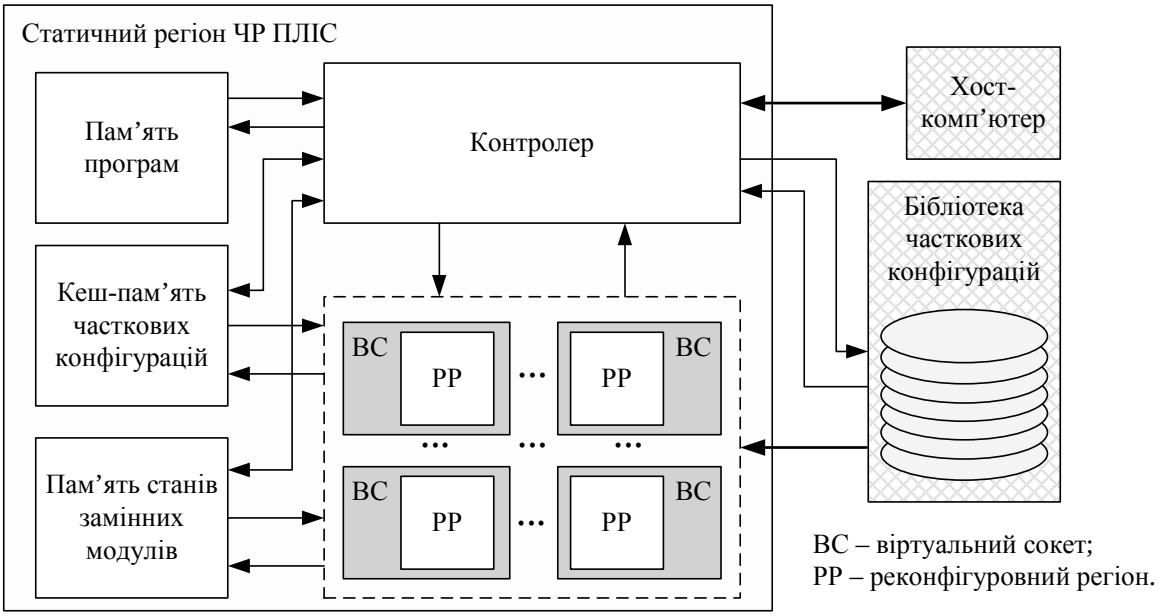

Рис. 1. Структура платформи для реалізації апаратної багатозадачності в ЧР ПЛІС

Реконфігуровні регіони тут мають заздалегідь розплановане розміщення, визначені розміри й інтерфейси. Кожен реконфігуровний регіон розташований в окремому віртуальному сокеті (англ. *Virtual Socket*), під'єднаному до каналів завантаження і вибірки часткових конфігурацій. Під віртуальним сокетом розуміють частину обладнання ПЛІС, що містить реконфігуровний регіон з ресурсами статичного регіону, які забезпечують його часткове реконфігурування [26].

Для розв'язування потрібних задач розробник створює замінні модулі й генерує часткові конфігурації для кожного з них, враховуючи обмеження на кількість обладнання та специфікації інтерфейсів реконфігуровних регіонів. Для зберігання файлів часткових конфігурацій використано спеціальну пам'ять, позначену на рисунку як «бібліотека часткових конфігурацій». Це енергонезалежна пам'ять для довготермінового зберігання інформації, в якій конфігурації зберігаються постійно і зчитуються та передаються до реконфігуровних регіонів після ініціалізації розв'язування відповідних задач.

Для організації перемикання контексту, однією з головних вимог до якого є мінімальний час виконання, використано швидкі блоки вбудованої пам'яті для короткотермінового зберігання інформації – кеш-пам'ять часткових конфігурацій і пам'ять станів замінних модулів. Перемикання контексту, як і завантаження часткових конфігурацій з бібліотеки, виконує контролер.

Пам'ять програм містить програми контролера для завантаження часткових конфігурацій, перемикання контексту, які передбачають збереження і відновлення контексту, переміщення задач у ПЛІС (ці питання висвітлено нижче). Окрім виконання зазначених програм, контролер здійснює загальне керування роботою компонентів платформи і взаємодіє з хост-комп'ютером.

Нижче розглянемо детальніше базові аспекти організації апаратної багатозадачності в РККС на основі ЧР ПЛІС. Розгляд почнемо із концепції віртуальних апаратних засобів.

3. Концепція віртуальних апаратних засобів

Концепція віртуальних апаратних засобів (англ. *Virtual Hardware*) [7, 8] націлена на організацію апаратної багатозадачності в РККС і подібна до концепції віртуальної пам'яті в комп'ютерах зі сторінковою, сегментною і сегментно-сторінковою організацією пам'яті. Одиницею інформації, якою оперує підсистема віртуальних апаратних засобів, є підвантажувальний логічний модуль (ПЛМ, англ. *Swappable Logic Unit*). ПЛМ є логічним представленням комп'ютерного пристрою для розв'язування задачі або її частини на деякому етапі її розв'язування. ПЛМ представляє комп'ютерний пристрій на рівні коду конфігурації ПЛІС. Стан пристрою на деякому етапі розв'язування ним задачі є відображенням поточного вмісту усіх запам'ятовуючих елементів у його складі.

Суть концепції віртуальних апаратних засобів така:

1. У пам'яті РККС зберігаються програмні моделі комп'ютерних пристроїв для розв'язування задач. Ці моделі представлені на рівні кодів конфігурації ПЛІС реконфігуровного середовища.

2. Після ініціалізації розв'язування задачі в реконфігуровному середовищі створюється відповідний їй комп'ютерний пристрій. Упродовж розв'язування задачі цей пристрій може, в разі потреби, бути призупиненим і тимчасово вивантаженим із реконфігуровного середовища у пам'ять. Під тимчасовим вивантаженням пристрою розуміємо запис коду конфігурації ПЛІС і поточного вмісту усіх запам'ятовуючих елементів пристрою до пам'яті. Вивантажений комп'ютерний пристрій продовжує існувати в системі «віртуально» у вигляді ПЛМ.

3. Пам'ять, в якій зберігають ПЛМ – тимчасово вивантажені з реконфігуровного середовища комп'ютерні пристрої, назвемо простором підвантаження.

4. У потрібний момент часу ПЛМ можна повторно завантажити із простору підвантаження до реконфігуровного середовища для продовження/закінчення чи повторного розв'язування задачі.

5. Всі комп'ютерні пристрої, що працюють в реконфігуровному середовищі або існують віртуально як ПЛМ у просторі підвантаження, з погляду користувача функціонують паралельно.

6. Максимальний сумарний розмір ресурсів реконфігуровної логіки, який може бути наданий для імплементації усіх комп'ютерних пристроїв, що функціонують у РККС паралельно, визначається кількістю ресурсів реконфігуровного середовища та об'ємом простору підвантаження. Невелику кількість ресурсів реконфігуровного середовища можна компенсувати збільшенням простору підвантаження.

7. Продуктивність РККС залежить від кількості ресурсів реконфігуровного середовища і не залежить від об'єму простору підвантаження.

З огляду на сказане вище, термін «віртуальні» стосовно апаратних засобів може мати два тлумачення:

1) віртуальними є комп'ютерні пристрої, що існують у РККС у вигляді ПЛМ;

2) віртуальними є ресурси реконфігуровної логіки, що існують у РККС у вигляді простору підвантаження.

Для забезпечення високої швидкості підвантаження ПЛМ пам'ять простору підвантаження може бути багаторівневою і розподіленою по різних запам'ятовуючих пристроях в ПЛІС і за її межами.

Втілення концепції віртуальних апаратних засобів у РККС до останніх років було проблематичним з огляду на технічні характеристики наявних ПЛІС. По-перше, тривалість їх реконфігурування не давала змогу здійснити в прийнятний час підвантаження потрібного ПЛМ. По-друге, відсутність можливості реконфігурування в частині не дозволяла реалізувати апаратну багатозадачність в межах ПЛІС. З появою і подальшим розвитком технології динамічного часткового реконфігурування ПЛІС ситуація змінилася, і сьогодні концепція віртуальних апаратних засобів реалізується в багатозадачних РККС, маючи підтримку не тільки на рівні апаратури, але й на рівні їх операційних систем [9–11].

Трирівневу пам'ять простору підвантаження у платформі для реалізації апаратної багатозадачності в ЧР ПЛІС показано на рис. 2.

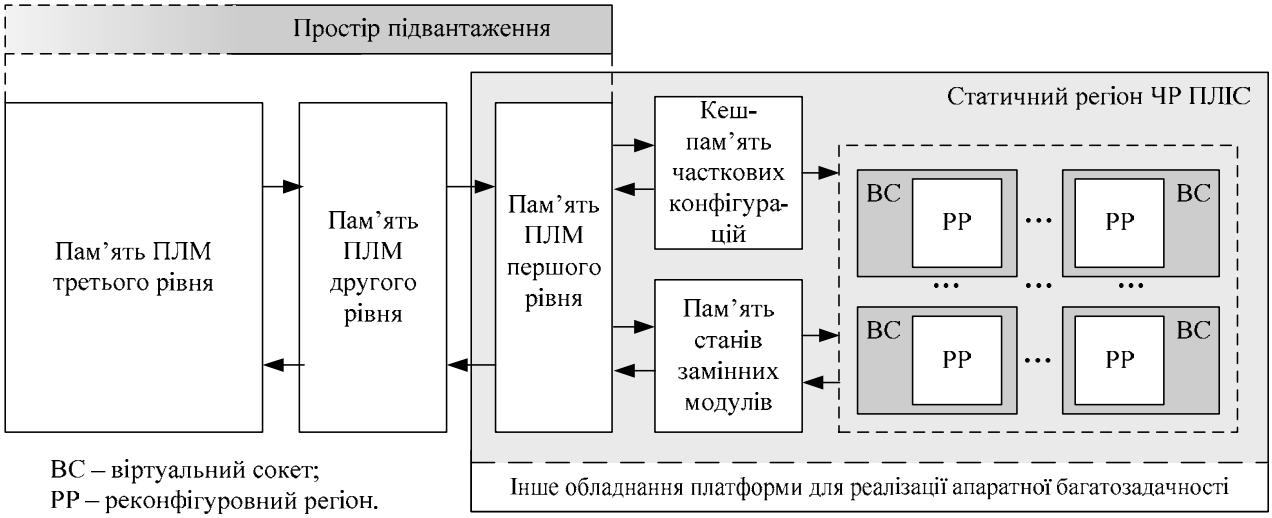

Рис. 2. Трирівнева пам'ять простору підвантаження у платформі для реалізації апаратної багатозадачності в ЧР ПЛІС

Як можемо зауважити, кеш-пам'ять часткових конфігурацій та пам'ять станів замінних модулів, розміщених у межах ПЛІС, мають логічне продовження у просторі підвантаження. Фізично пам'ять ПЛМ першого рівня може бути реалізована у вбудованій пам'яті ПЛІС, пам'ять ПЛМ другого рівня – в оперативному запам'ятовуючому пристрої, який тісно взаємодіє з ПЛІС. Пам'ять ПЛМ третього рівня може використовувати деяку частину бібліотеки конфігурацій, теоретично аж до усього її об'єму. Зрозуміло, що швидкість підвантаження ПЛМ зі збільшенням простору підвантаження знижується, особливо із залученням до його складу бібліотеки конфігурацій, що, як зазначено вище, реалізується в енергонезалежній пам'яті, може мати великий об'єм і невисоку швидкість читання/запису. Все ж переваги застосування концепції віртуальних апаратних засобів беззаперечні, і найважливішими з них є такі:

– можливість паралельно розв'язувати більшу кількість задач, ніж кількість реконфігуровних регіонів у ЧР ПЛІС;

– можливість виконувати задачі, які потребують більшої кількості ресурсів реконфігуровної логіки, ніж доступно в ЧР ПЛІС, завдяки завантаженню до неї окремих модулів відповідних пристроїв.

Першочерговими питаннями, які виникають під час реалізації концепції віртуальних апаратних засобів, є перемикання контексту і переміщення задач з одного реконфігуровного регіону ПЛІС до іншого. Ці питання розглянемо нижче.

3. Перемикання контексту в частково реконфігуровних ПЛІС

У багатозадачних операційних системах під терміном «перемикання контексту» розуміють комплекс дій, необхідних для заміни однієї розв'язуваної процесором задачі на іншу, який дає можливість пізніше відновити розв'язування попередньої задачі з того самого місця, на якому воно було зупинено. У книзі [27] перемикання контексту визначено як передача керування від одного процесу до іншого (процес – це програма під час виконання з її власними даними) зі збереженням стану процесора. Відповідно контекстом називають стан процесу в деякий момент часу його виконання. Хоча реалізація перемикання контексту залежить від архітектури комп'ютера і операційної системи, узагальнена послідовність виконуваних дій така:

1) зупинка виконання поточного виконуваного процесу;
2) збереження контексту поточного виконуваного процесу в область його зберігання в пам'яті;
3) визначення наступного виконуваного процесу;
4) завантаження контексту наступного виконуваного процесу із області його зберігання в пам'яті;
5) початок або продовження виконання наступного виконуваного процесу.

Контекст є сукупністю значень регістрів процесора, причому регістрів не тільки загального призначення, але й системних, зокрема лічильника команд, регістрів захисту пам'яті, слова стану програми й інших, залежно від архітектури процесора [28]. По суті, перемикання контексту є втратою часу, оскільки протягом його виконання процесор не виконує жодної іншої роботи. Для мінімізації цієї втрати перемикання контексту часто виконують із застосуванням засобів апаратної підтримки. Наприклад, в архітектурі *IA*-32 для збереження контексту використовують спеціальну ділянку пам'яті – сегмент стану задачі (англ. *TSS – Task State Segment*). Адресу цієї ділянки зберігають в системному регістрі задачі *TR*.

Реалізація перемикання контексту в ЧР ПЛІС істотно відрізняється від його реалізації в універсальних процесорах. Це зумовлено застосуванням принципово різних підходів до організації опрацювання інформації, покладених в основу їх функціонування, а саме:

– в універсальному процесорі алгоритм розв'язування задачі подають програмою, а в ПЛІС – конфігурацією обладнання;

– структура універсального процесора незмінна, для зміни алгоритму розв'язування задачі змінюють виконувану програму. Зміну алгоритму в ПЛІС здійснюють зміною конфігурації;

– в універсальному процесорі контекст є сукупністю поточних значень його регістрів. У ПЛІС контекст – це конфігурація і поточний вміст всіх запам'ятовуючих елементів пристрою в її складі – тригерів, регістрів, блоків пам'яті. Нагадаємо, що в ЧР ПЛІС розробник реалізує пристрій як замінний модуль.

З урахуванням вказаних відмінностей визначимо та опишемо послідовність дій, які потрібно виконати для перемикання контексту в разі реалізації апаратної багатозадачності в ЧР ПЛІС. Цю послідовність ілюструє рис. 3.

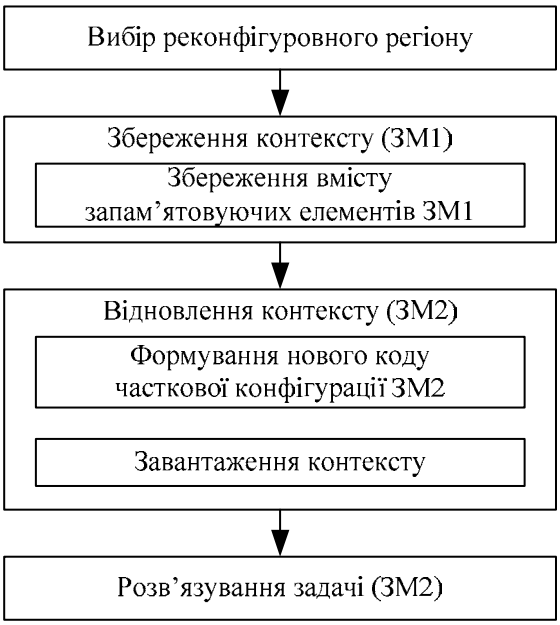

Рис. 3. Послідовність дій для перемикання контексту в разі реалізації апаратної багатозадачності в ЧР ПЛІС

Оскільки в ЧР ПЛІС може одночасно працювати деяка кількість пристроїв, відповідно до наявної кількості реконфігуровних регіонів, питання перемикання контексту в ній виникає тоді, коли всі реконфігуровні регіони зайняті. В іншому випадку часткову конфігурацію завантажують до вільного реконфігуровного регіону. Тому першою операцією, яку потрібно виконати, є вибір реконфігуровного регіону, в якому буде змінено контекст. Далі потрібно зберегти контекст – часткову конфігурацію і поточний вміст усіх запам'ятовуючих елементів замінного модуля ЗМ1, що працює у цьому реконфігуровному регіоні. Сформувати дані про стан ЗМ1 можна способом

зчитування вмісту запам'ятовуючих елементів і його накладанням на вміст відповідних елементів у початковому коді конфігурації, як це запропоновано в роботі [12]. На відміну від описаного в роботі [12] рішення, робити це пропонується не під час збереження, а під час відновлення контексту, якщо воно буде потрібне (нема сенсу формувати контекст, якщо замінний модуль закінчив свою роботу). Тому до пам'яті станів замінних модулів (рис. 1) записуємо вміст запам'ятовуючих елементів замінного модуля.

Наступною операцією є відновлення контексту – часткової конфігурації та вмісту усіх запам'ятовуючих елементів замінного модуля ЗМ2. Якщо ЗМ2 тільки починає свою роботу, то контролер вибирає з бібліотеки його часткову конфігурацію і завантажує її до реконфігуровного регіону, а також копіює у кеш-пам'ять (рис. 1). Якщо ж поставлене завдання відновлення раніше збереженого контексту, то контролер повинен сформувати новий код часткової конфігурації ЗМ2 із його початкового коду і поточного вмісту запам'ятовуючих елементів та завантажити цей код до реконфігуровного регіону. Початковий код конфігурації контролер зчитує з кеш-пам'яті. Процес формування нового коду часткової конфігурації можна сумістити з процесом часткового реконфігурування ПЛІС.

Вирізняють два методи передавання даних під час збереження та відновлення контексту – з використанням стандартного інтерфейсу реконфігурування (англ. *reconfiguration-based*) [22, 13], наприклад, *ICAP*, і з використанням інтерфейсу пристрою (англ. *design-based* [29, 30], або *Task-Specific Access Structures* [22]), який проектується так, щоб максимально швидко виконати зчитування/запис вмісту запам'ятовуючих елементів. Природно, що другий метод забезпечує високу швидкість, але, на відміну від першого, вимагає від розробника безпосередньої реалізації засобів швидкого зчитування/запису даних у запам'ятовуючих елементах на рівні архітектури пристрою, що, своєю чергою, збільшує затрати обладнання на його реалізацію та ускладнює проектування.

Програмну реалізацію процедур формування, збереження та відновлення контексту запропоновано в роботі [12]. Там передбачено, що ці процедури виконує вбудований мікропроцесор, розміщений у статичному регіоні. На нашу думку, такий підхід суттєво уповільнює відновлення контексту, оскільки формування нового коду конфігурації мікропроцесор виконує послідовно у форматі 32-розрядних слів, а формування кожного слова передбачає звернення до кеш-пам'яті часткових конфігурацій, звернення до пам'яті контекстів і передавання результату в реконфігуровний регіон. Зменшити тривалість відновлення контексту можна внесенням до складу віртуального сокета, в якому міститься реконфігуровний регіон, пристрою формування контексту. Враховуючи низьку обчислювальну складність цієї операції ($context = state_{actual} \cdot msk + conf_{init} \cdot \overline{msk}$, де msk – маска, яка вказує на належність біта до даних контексту [12]), її апаратна реалізація не потребуватиме значних затрат обладнання. І більше, якщо такий пристрій розмістити в кожному віртуальному сокеті, то перемикання контексту можна буде виконувати в різних реконфігуровних регіонах одночасно. Укрупнену структуру утворених у такий спосіб засобів апаратної підтримки перемикання контексту в складі платформи для реалізації апаратної багатозадачності в ЧР ПЛІС ілюструє рис. 4. Щоб забезпечити максимальну швидкість і можливості одночасного виконання перемикання контексту в різних реконфігуровних регіонах, кеш-пам'ять часткових конфігурацій і пам'ять станів замінних модулів тут розподілені на окремі для кожного реконфігуровного регіону блоки.

5. Переміщення задач у ЧР ПЛІС

Як вже сказано вище, забезпечення можливості переміщення задач (англ. *Task Relocation*) належить до принципових завдань, необхідних для реалізації апаратної багатозадачності в РККС на основі ПЛІС з динамічним частковим реконфігуруванням. Потреба виконання переміщення задач у ЧР ПЛІС виникає після ініціалізації чи продовження розв'язування задачі, коли відповідні їй ПЛМ (один чи більше) потрібно завантажити до вибраних засобами комп'ютерної системи реконфігуровних регіонів, розміщення яких у ЧР ПЛІС є іншим, ніж визначено на етапі планування.

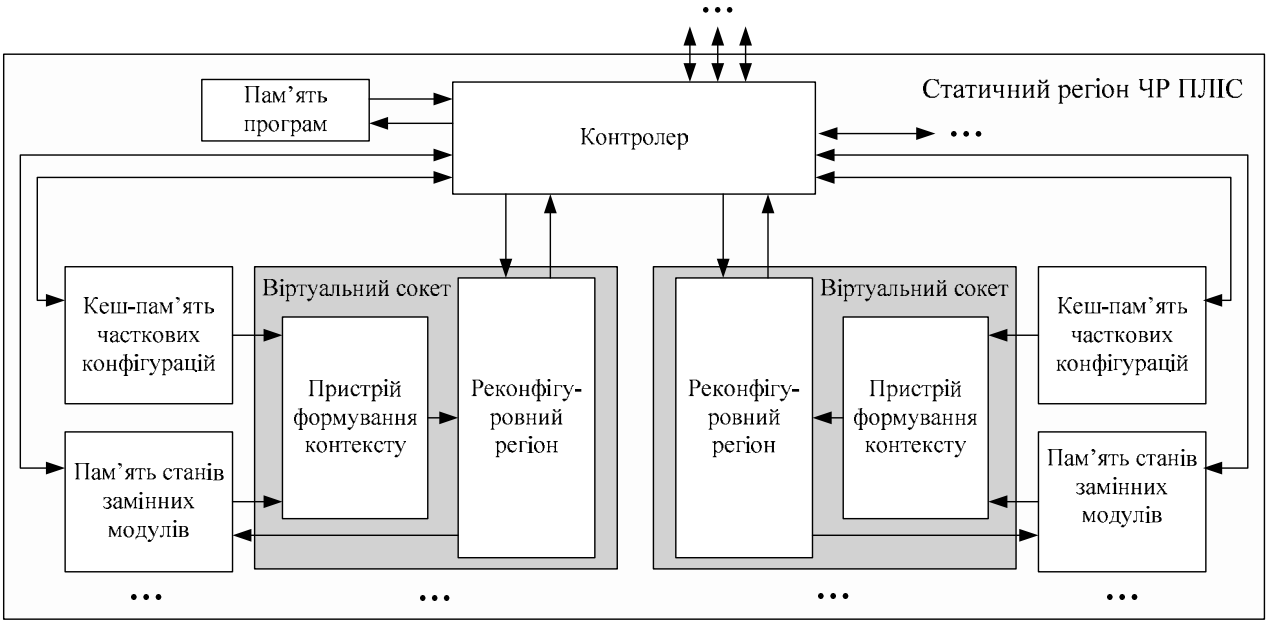

Рис. 4. Укрупнена структура засобів апаратної підтримки перемикання контексту в складі платформи для реалізації апаратної багатозадачності в ЧР ПЛІС

Проблемним питанням переміщення задач є те, що методологія проектування комп'ютерних пристроїв із замінними модулями для ЧР ПЛІС передбачає визначення місця розташування кожного замінного модуля в ПЛІС на етапі планування розміщення реконфігуровних регіонів. У результаті ПЛМ мають визначене і фіксоване розміщення в ПЛІС, вказане адресами у відповідних полях коду конфігурації.

Тривіальним підходом до вирішення питання переміщення задачі є створення окремих ідентичних часткових конфігурацій одного замінного модуля для різних реконфігуровних регіонів, але це супроводжується потребою зберігання в пам'яті великої кількості надлишкових даних. Реалізація ж переміщувальних замінних модулів поки що не підтримується системами автоматизованого проектування, що пропонують виробники ПЛІС. Тому науковці й інженери, які стикаються із питанням переміщення задач у ЧР ПЛІС, розробляють для цього власні засоби. Це, наприклад, *REPLICA* [19], який модифікує початковий код часткової конфігурації під час його завантаження до ПЛІС. Такий підхід – модифікацію коду «на льоту» – називають динамічним переміщенням (англ. *dynamic relocation*). Засіб *PBITPOS* [31] розроблений для ПЛІС *Virtex* II і *Virtex* II *Pro* фірми *Xilinx* і тому уможливлює переміщення лише в межах стовпців (так зване одновимірне переміщення). Ще кілька методів реалізації переміщення описано в роботах [16–18] і [20]. На наш погляд, функціонально найповнішим з погляду проектування комп'ютерних пристроїв із переміщувальними замінними модулями для ЧР ПЛІС є засіб *GoAhead* [32], розроблений на факультеті інформатики Університету м. Осло.

В ЧР ПЛІС можливість переміщення задач забезпечується тим, що початковий, з якого виконується переміщення, і цільовий, у який виконується переміщення, реконфігуровні регіони мають однакові:

1) розміри;
2) кількість, тип і розміщення ресурсів реконфігуровної логіки і каналів передавання даних;
3) розміщення інтерфейсних ліній.

Додатковою вимогою є незастосування транзитних статичних сигналів, що проходять крізь реконфігуровний регіон.

Загалом розрізняють переміщення одновимірне (1-D), можливе виключно в межах стовпців, і двовимірне (2-D), у межах рядків і стовпців ПЛІС. Можливість здійснення 1-D чи 2-D переміщення залежить від розміру реконфігуровного регіону й архітектури ПЛІС. Зважаючи на те, що більшість сучасних ПЛІС містять блоки різного функціонального призначення, а переміщення модуля у

новий регіон можливе за умови, що цей регіон має ідентичний із попереднім склад і розташування ресурсів, 2-D переміщення характерне для регіонів малого і середнього розміру.

Потрібно зазначити, що, оскільки переміщення замінного модуля передбачає маніпуляції над конфігураційним кодом, спосіб його здійснення абсолютно прив'язаний до типу ПЛІС, а тому нема єдиного алгоритму виконання переміщення задач для різних ПЛІС.

Висновки

1. Висвітлено підхід до реалізації апаратної багатозадачності в реконфігуровних комп'ютерних системах на основі ПЛІС, що підтримують динамічне часткове реконфігурування.

2. Виокремлено такі основні принципи реалізації апаратної багатозадачності в РККС, побудованих на основі ЧР ПЛІС:

2.1) в ЧР ПЛІС розміщено множину реконфігуровних регіонів, в кожному з яких у визначений момент часу може функціонувати один замінний модуль;

2.2) задачу (або частину задачі) подають в РККС у вигляді програмної моделі замінного модуля, який компілюють у файл часткової конфігурації ПЛІС і завантажують до неї у визначений реконфігуровний регіон за командою ініціалізації розв'язування цієї задачі;

2.3) замінний модуль може використовувати від одного до N_{RP} реконфігуровних регіонів, де N_{RP} – кількість реконфігуровних регіонів у ЧР ПЛІС;

2.4) у ЧР ПЛІС може одночасно функціонувати до N_{RP} замінних модулів;

2.5) у разі потреби замінні модулі можуть використовувати реконфігуровний регіон RP_k в режимі часового мультиплексування;

2.6) у РККС на основі ЧР ПЛІС реалізовано:

а) засоби завантаження часткових конфігурацій;

б) механізми перемикання контексту в ПЛІС і переміщення задач між реконфігуровними регіонами;

в) концепцію віртуальних апаратних засобів.

3. Запропоновано структуру платформи для реалізації апаратної багатозадачності в ЧР ПЛІС, яка відповідає вказаним вище принципам.

4. Викладено суть концепції віртуальних апаратних засобів. Логічною одиницею інформації, якою оперує підсистема віртуальних апаратних засобів, є підвантажувальний логічний модуль. ПЛМ є логічним представленням комп'ютерного пристрою для розв'язування задачі або її частини на деякому етапі її розв'язування. ПЛМ представляє комп'ютерний пристрій на рівні коду конфігурації ПЛІС. Стан пристрою на деякому етапі розв'язування ним задачі є відображенням поточного вмісту усіх запам'ятовуючих елементів у його складі.

5. Розглянуто особливості виконання та визначено послідовність дій для перемикання контексту в разі реалізації апаратної багатозадачності в ЧР ПЛІС. Запропоновано структуру засобів апаратної підтримки перемикання контексту.

6. Розглянуто питання переміщення задач у ЧР ПЛІС, подано огляд підходів та інструментальних засобів, що застосовуються для його виконання.

1. Top 500 project. "Operating system Family share for 11/2014". [Електронний ресурс]. – Режим доступу: http://www.top500.org/statistics/overtime/. 2. "Embedded market study – Mars, 2012". [Електронний ресурс]. – Режим доступу: http://seminar2.techonline.com/~additionalresources/esd_apr2012/ubme_embeddedmarket2012_full.pdf. 3. "RTOS market". NewTechPress. November 2011. [Електронний ресурс]. – Режим доступу: http://www.newtechpress.net/2011/11/08/rtos-market-in-turmoil/. 4. [Електронний ресурс]. – Режим доступу: http://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html. 5. [Електронний ресурс]. – Режим доступу: http://www.linfo.org/context_switch.html. 6. [Електронний ресурс]. – Режим доступу: https://technet.microsoft.com/en-us/library/cc938606.aspx. 7. Brebner G. J. A Virtual Hardware Operating System for the Xilinx XC6200. Proc. of the International Workshop on Field-Programmable Logic, Smart Applications, New Paradigms and Compilers, 1996. 8. Brebner G. The swappable logic unit: a paradigm for virtual hardware. In K. L. Pocek and J. M. Arnold, editors, The 5th Annual IEEE Symposium on FPGAs for Custom Computing Machines (FCCM'97). – P. 77–86,

Los Alamitos, CA, Apr. 1997. IEEE Computer Society Press. 9. Steiger C., Walder H., Plazner M. Operating Systems for Reconfigurable Embedded Platforms: Online Scheduling of Real-Time Tasks // IEEE Transactions on Computers, Vol. 53, № 11. – P. 1393–1407, Noviembre 2004. 10. Göhringer D., Hübner M., Nguepi Zeutebouo E., Becker J. Operating System for Runtime Reconfigurable Multiprocessor Systems // Int. J. Reconfig. Comp. – 2011 (2011). 11. Chen Y., Hsiung P. (2005) Hardware Task Scheduling and Placement in Operating Systems for Dynamically Reconfigurable SoC // In: Yang L. T., Amamiya M., Liu Z., Guo M., Rammig F. J. (eds.) EUC 2005. LNCS, Vol. 3824. – P. 489–498. Springer, Heidelberg. 12. Morales-Villanueva A. and Gordon-Ross A. On-chip Context Save and Restore of Hardware Tasks on Partially Reconfigurable FPGAs // in Proc. of IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM). – 2013. 13. Lee T.-Y., Hu C.-C., Lai L.-W. and Tsai C.-C. Hardware Context-Switch Methodology for Dynamically Partially Reconfigurable Systems // J. Inf. Sci. Eng. – 2010. – Vol. 26, № 4. – P. 1289–1305. 14. Marconi T., Lu Y., Bertels K. and Gaydadjiev G. Online Hardware Task Scheduling and Placement Algorithm on Partially Reconfigurable Devices // In: Proceedings of International Workshop on Applied Reconfigurable Computing, Architectures, Tools and Applications (ARC 2008), March 26–28, 2008. 15. Kalte H. and Porrmann M. Context Saving and Restoring for Multitasking in Reconfigurable Systems // Proc. of the International Conference on Field Programmable Logic and Applications. – 2005. – P. 223–228. 16. Hiibner M., Schuck C., Kiihnle M., Becker J. New 2-dimensional partial dynamic reconfiguration techniques for real-time adaptive microelectronic circuits // Emerging VLSI Technologies and Architectures, 2006 // IEEE Computer Society Annual Symposium on, 2–3 March 2006. – P. 6. 17. Becker T., Luk W. Cheung, P. Y. K. Enhancing Relocatability of Partial Bitstreams for Run-Time Reconfiguration // Field-Programmable Custom Computing Machines, 2007. FCCM 2007. 15th Annual IEEE Symposium on, P. 35–44, 23–25 April 2007. 18. Ichinomiya Y., Usagawa S., Amagasaki M., Iida M., Kuga M., Sueyoshi T. Designing Flexible Reconfigurable Regions to Relocate Partial Bitstreams // Field-Programmable Custom Computing Machines (FCCM), 2012 IEEE 20th Annual International Symposium on, P. 241, April 29 2012-May 1 2012. 19. Kalte H., Lee G., Porrmann M., Ruckert U. REPLICA: A Bitstream Manipulation Filter for Module Relocation in Partial Reconfigurable Systems // Proceedings of 19th IEEE International Symposium on Parallel and Distributed Processing, 2005, 04-08 April 2005. 20. Drahonovsky T., Rozkovec M. and Novak O. Relocation of reconfigurable modules on Xilinx FPGA // Design and Diagnostics of Electronic Circuits Systems (DDECS), 2013 IEEE 16th International Symposium on, P. 175–180. 21. J. van der Veen, Fekete S., Majer M., Ahmadinia A., Bobda C., Hannig F. and Teich J. Defragmenting the module layout of a partially reconfigurable device // In Proc. 2005 Int. Conference on Engieering of Reconfigurable Systems and Algorithms. CSREA Press, 2005. 22. Koester M., Porrmann M. and Kalte H. Relocation and defragmentation for heterogeneous reconfigurable systems // In Proc. Int. Conf. Eng. Reconfig. Syst. Algorithms, 2006, P. 70–76. 23. Stephan Suijkerbuijk and Ben H. H. Juurlink, Implementing Hardware Multithreading in a VLIW Architecture, International Conference on Parallel and Distributed Computing Systems, 2005. 24. Montone A., Santambrogio M. D., Sciuto D. & Memik S. O. (2010). Placement and floorplanning in dynamically reconfigurable FPGAs. ACM Transactions on Reconfigurable Technology and Systems (TRETS), 3(4), 24. 25. Li Z., Hauck S. Configuration prefetching techniques for partial reconfigurable coprocessor with relocation and defragmentation. FPGA 2002: 187–195. 26. Partial Reconfiguration Controller v1.0. LogiCORE IP Product Guide. Vivado Design Suite. PG193 April 1, 2015. [Електронний ресурс]. – Режим доступу: http://www.xilinx.com/support/documentation/ip_documentation/prc/v1_0/pg193-partial-reconfiguration-controller.pdf. 27. Шеховцов В. А. Операційні системи. – К.: Видавнича група BHV, 2005. – 576 с. 28. Мельник А. О., Архітектура комп'ютера. – Луцьк: Волинська обласна друкарня, 2008. – 470 с. 29. Huang C. H. and Hsiung P. A. Software-controlled dynamically swappable hardware design in partially reconfigurable systems // EURASIP J. on Embedded Systems, Vol. 2008. – P. 231940, 2008. 30. Puttegowda K., Lehn D. I., Park J. H., Athanas P. and Jones M. Context switching in a run-time reconfigurable system // The J. of Supercomputing. – 2003. – Vol. 26. – P. 239–257. 31. Krasteva Y. E., de la Torre E., Riesgo T., Joly D. Virtex II FPGA Bitstream Manipulation: Application to Reconfiguration Control Systems // Field Programmable Logic and Applications, 2006. FPL'06. International Conference on. – 28–30 Aug. 2006. – P. 1–4. 32. Beckhoff C., Koch D., Torresen, J. Go Ahead: A Partial Reconfiguration Framework // Field-Programmable Custom Computing Machines (FCCM), 2012 IEEE 20th Annual International Symposium on. – April 29 2012–May 1 2012. – P. 37–44.