УДК 004.274

Клименко І. А., д.т.н.,
Сторожук О. М.

УДОСКОНАЛЕНИЙ МЕТОД АВТОМАТИЧНОГО ВІДОБРАЖЕННЯ ЗАДАЧ У РЕКОНФІГУРОВНИХ КОМП'ЮТЕРНИХ СИСТЕМАХ, КЕРОВАНИХ ПОТОКОМ ДАНИХ

Національний технічний університет України «КПІ ім. І.Сікорського»

ikliryna@gmail.com,
storozhuk.om@gmail.com

Запропоновано засоби автоматичного відображення задач на реконфігуровне обчислювальне середовище, які ґрунтуються на вдосконаленні відомого методу автоматичного розпаралелення задач на рівні макрокоманд. Удосконалено метод розподілу завдань для автоматизації керування обробкою інформації на структурному рівні РКС (реконфігуровних комп'ютерних системах), керованих потоком даних. Удосконалено апаратні засоби розподілу завдань та синхронізації, що дозволило зменшити кількість непродуктивних звернень і час звернення до загальних ресурсів, накладні витрати на організацію паралельної обробки даних та налаштування обчислювального середовища на ПЛІС (Програмовані Логічні Інтегральні Схеми), ємність використання вбудованої пам'яті ПЛІС

Ключові слова: реконфігуровні комп'ютерні системи, розпаралелювання, РКС, ПЛІС, СЛАР

Вступ

Під час процесів керування, в режимі реального часу розв'язуються траєкторні задачі, що зводяться до реалізації різних методів інтерполяції, таких як поліноміальна апроксимація, розкладання в ланцюгові дроби, таблично-алгоритмічні, ітераційні, тощо. Моделювання технічних і технологічних процесів керування зводяться до розв'язання систем диференційних та лінійних алгебричних рівнянь

В системах керування реального часу більшість задач мають дрібнозернисту структуру, за якої найвищий ступінь розпаралелення може бути досягнуто, коли кожній вершині графу відповідає окрема операція, а саме: обчислення елементарних функцій або перетворення координат, що показано в роботі [1]. При цьому максимально збільшується кількість паралельних гілок графу, що приводить до використання великої кількості паралельно працюючих обчислювальних елементів. Проте на швидкість обробки інформації негативно впливають витрати часу на обмін інформацією між паралельними гілками. У праці [2] зазначено, що розв'язання дрібнозернистих задач засобами сучасних високопродуктивних комп'ютерів забезпечує лише 1% від їх пікової продуктивності.

Зважаючи на те, що в системах керування реального часу важливішою характеристикою є не продуктивність виконання ряду обчислювальних операцій, а час обчислення певної функції, у праці [1] визначено поняття макрофункції як набір програмних або апаратних засобів для обчислення певної функції.

Актуальною парадигмою підвищення ефективності систем керування в реальному часі є пришвидшення виконання таких макрофункцій на всіх рівнях обчислювальної системи.

У працях [1, 3 – 5] показано, що ефективним підходом до пришвидшення реалізації дрібнозернистих алгоритмів є використання динамічних засобів розподілу операцій між обчислювальними вузлами, заснованих на моделі обчислень, що керуються потоком даних. Основною функціональною особливістю є те, що команди виконуються не в описаній програмою послідовності, а в міру надходження повної інформації про команду і операнди. Таким чином, визначальним

для виконання команди є доступність даних, а не заданий програмою порядок виконання команд. Ця особливість дозволяє розподіляти операції динамічно у процесі обчислення на апаратному або мікропрограмному рівні. Під час підготовки завдання немає потреби описувати процедури синхронізації процесів та обміну даними між гілками алгоритмів у прямій формі. Це створює передумови для пришвидшення процесу обробки інформації, зокрема за рахунок його реалізації на програмно-апаратному рівні обчислювального модуля реконфігуровної комп'ютерної системи (РКС) [6 – 8].

Модифікація математичної моделі формування заявок

Для автоматизації функції керування у РКС, керованих потоком даних, в роботі запропоновано засоби автоматичного відображення задач на реконфігуровне обчислювальне середовище, які ґрунтуються на вдосконаленні відомого методу автоматичного розпаралелення задач на рівні макрокоманд [3, 5]. Засоби автоматичного розпаралелення задач, розроблені для традиційних потокових обчислювальних систем, дають змогу спростити і скоротити цикл виконання макрокоманд, але не враховують апаратні і функціональні обмеження РКС. Це не дозволяє ефективно їх використовувати в обчислювальних системах, побудованих на сучасній динамічно програмовній елементній базі.

Механізм формування заявок у потокових обчислювальних системах, описаний у працях [1, 5], ґрунтується на керуванні дескрипторами даних, що надходять у систему. У працях [4, 5] запропоновано метод автоматичного динамічного розпаралелення обчислень на рівні програмних модулів, команд і операцій у паралельних системах з неоднорідним доступом до пам'яті, керованих потоком даних. Цей метод пришвидшує реалізацію алгоритмів з неявним паралелізмом у багатозадачному режимі. Але метод розроблено для реалізації спеціалізованих потокових систем та обчислювачів з гомогенним обчислювальним середовищем, реалізованим на однотипних обчислювальних блоках без будь-яких функціональних та апаратних обмежень.

Для реалізації вдосконаленого способу автоматичного відображення задач на реконфігуровне обчислювальне середовище модифікуємо математичну модель формування заявки наступним чином.

Обчислювальний процес подається за допомогою макрографу потоку даних *MDG* [9] $G_M = (N_M, D_M)$, де N_M – множина вершин; D_M – множина дуг. У вершинах графу розміщуються завдання (макрофункції) $N_i (i = \overline{1, g})$. Потік даних (дескриптор даних), який відповідає дузі D_{iv} графу, з'єднує *i*-у вершину з *v*-ю вершиною графу. Вихідні дані для кожного завдання готуються на підставі графу G_M і являють собою набір вхідних дескрипторів даних, а також бібліотеку апаратної реалізації функцій.

Далі наведено модифіковану математичну модель формування заявки з урахуванням процесу завчасної реконфігурації обчислювального середовища на ПЛІС.

Кожне завдання описується актором. Відомий формальний опис актора [1, 4, 5], що відповідає *i*-й вершині графу має вигляд:

$$U_i = \{n_i, N_i, I_i, Q_i, l_i, M_i, C_i\},$$

де n_i – ознака актора; N_i – ім'я актора; I_i – ідентифікатор завдання (команди, програми, процесу); Q_i – сумарна кількість потоків даних для завдання (вхідних дескрипторів); l_i – рівень пріоритету завдання; M_i – множина імен акторів, для яких *i*-е завдання готує дані; C_i – множина імен потоків даних, які формуються під час виконання *i*-го завдання.

Модифікований формат актора ілюструє рис. 1. У його окремому полі зберігається кількість акторів *x*, для яких актор N_i готує дані (фактично кількість елементів множини C_i), при цьому немає потреби в передаванні на апаратний рівень

пристрою формування заявок елементів множини M_i. У полі C_i актора міститься масив ідентифікаторів вихідних дескрипторів даних, які використовуються для адресації даних в пам'ять реєстрації під час формування заявок. У полі ρ_i – показник апаратного пришвидшення виконання завдання, який обчислюється за формулою [7]:

$$\rho = \frac{T_{SW}}{[T_{CONTROL} + R] + T_{HW}},$$

де T_{SW} – час обчислення задачі на процесорному ядрі; T_{HW} – час обчислення задачі на апаратурі ПЛІС; R – час реконфігурації обчислювального середовища для виконання завдання, $T_{CONTROL}$ – часова складність процесу керування відображенням завдання на реконфігуровне обчислювальне середовище. Час реконфігурації обчислювального середовища визначає непродуктивний час процесу його реконфігурації. Сума $(T_{CONTROL} + R)$ визначає накладні витрати процесу відображення обчислювальних задач на реконфігуровне обчислювальне середовище.

Дескриптор даних є множиною елементів:

$$D_{oi} = \{n_{oi}, N_i, Q, A_{oi}\},$$

де n_{oi} – ім'я (номер) потоку даних (дуги $o \to i$, що входить до i-у вершину); $A_i = \{A_{oi}\}$ – елемент адресації даних, при цьому $o \mid o = \overline{1,x}$ – індекс попередньої вершини графу.

Формат дескриптора показано на рис. 2. Виходячи з того, що всі вершини графу мають унікальні імена N_i, елемент адресації визначає місце розташування даних у пам'яті обчислювального модуля, яка доступна для кожного локального процесора. Ім'я потоку даних (дескриптора) є ознакою, що відрізняє дескриптор даних від актора.

З елементів акторів та дескрипторів відповідно до певної процедури λ формуються заявки на виконання i-го завдання, модифікована модель якої має вигляд:

$$l(U_i, D_{vi}) \to Z_i = \{I_i, \Pi_i, M_i, C_i, A_i\},$$

де $\Pi_i = 0$ – прапорець, що визначає неефективність виконання завдання апаратними засобами реконфігуровного обчислювального середовища, інакше встановлюється $\Pi_i = 1$. У модифікованій моделі заявки процес завчасної реконфігурації врахований умовою формування заявки таким чином:

$$\omega_i(Z_i) = true, \text{ якщо}$$
$$\omega_i(Z_i) = \nu_i \,\&\, \alpha_{vi} \,\&\, [j_{vi} \mid \rho_i > 1],$$

де ν_i – умова отримання дескриптора i-го завдання; α_{vi} – умова отримання елемента адресації по дузі $v \to i$; j_i – ознака виконаного налаштування ПЛІС для обчислення чергової задачі, яка формується за умови додатного значення показника пришвидшення апаратних обчислень ($\rho_i > 1$). Формат слова заявки показано на рис. 3.

Розроблення структури і принципів функціонування вдосконаленого пристрою автоматичного розподілу завдань і синхронізації на ПЛІС

Структуру відомого пристрою автоматичного розподілу завдань подано у працях [1, 4, 5]. Мнемонічну схему, що зображує структуру вдосконаленого пристрою автоматичного розподілу завдань і синхронізації (ПАРС) та принцип формування заявок з реалізацією завчасної реконфігурації обчислювального середовища, описану у праці [8], показано на рис. 4.

У склад ПАРС на відміну від відомої реалізації [1, 4, 5] уведено контролер реконфігурації та буферну пам'ять черги реконфігурації, які застосовуються для керування процесом завчасної реконфігурації обчислювального середовища на структурному рівні обчислювального модуля.

N_i	n_i	g_i	Q_i	I_i	ρ_i	x_i	C_i

Рис. 1. Формат актора

n_{oi}	N_i	Q_i	A_{oi}

Рис. 2. Формат дескриптора

I_i	M_i	C_i	A_i

Рис. 3. Формат заявки

Наявність контролера реконфігурації забезпечує розпаралелення процесу виконання макрокоманд і процесу керування реконфігурацією обчислювального середовища. Запропоновані засоби функціонують на локальному рівні обчислювального модуля і керують паралельними алгоритмами і реконфігуровними апаратними ресурсами без участі операційної системи.

Удосконалений ПАРС розроблено і досліджено на базі програмованої логічної інтегральної схеми (ПЛІС) *Cyclone II EP2C35F672C6* компанії *Altera*. Використано стандартні засоби проектування систем-на-кристалі SOPC *Builder*. Для реалізації керувального процесора та контролера реконфігурації застосовано вбудовані процесорні ядра *Nios II/s.* Модуль ПАРС реалізований як апаратна надбудова керувального процесорного ядра.

Керувальний процесор керує потоковими обчисленнями та формуванням заявок. Реалізовані на ПЛІС функціональні блоки (ФБ) з точки зору керування є пасивними функціональними елементами. Дескриптори даних для кожного завдання формуються в різні моменти часу різними ФБ на ПЛІС. Контроль за надходженням даних для виконання чергового завдання виконує ПАРС.

Завчасна реконфігурація здійснюється таким чином. Процедура λ, реалізовує автоматичну функцію керування процесом реконфігурації обчислювального середовища на підставі формальної моделі. У результаті виконання цієї функції керування в буфері черги реконфігурації (рис. 4) формується черга завчасної реконфігурації. Готовність заявки визнається формуванням ознаки готовності заявки відповідно до значення $\omega_i(Z_i)$ на керувальному виході блока готовності заявки (рис. 4).

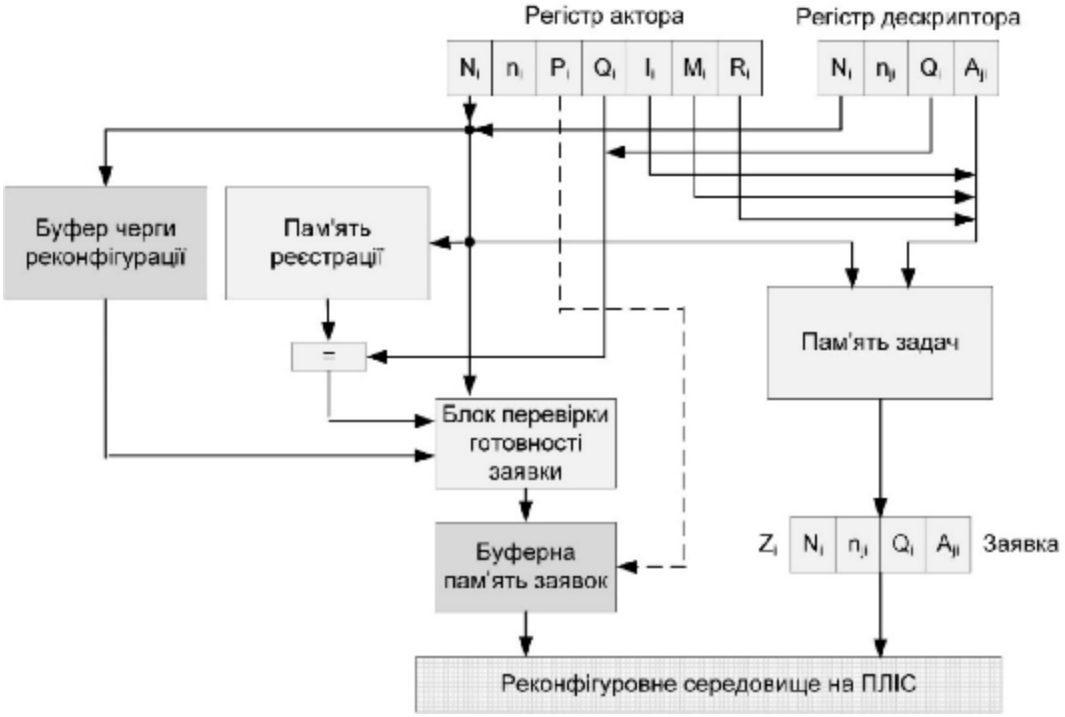

Рис. 4. Схема процесу формування заявки із завчасною реконфігурацією обчислювального середовища у вдосконаленому ПАРС

Реконфігурація обчислювального середовища здійснюється згідно з параметром «тип операції I_i» у відповідному полі актора (див. рис. 1). Під час активації процесу реконфігурації відповідно до типу виконуваної операції з бібліотеки ФБ зчитуються конфігураційні дані для налаштування обчислювального середовища на ПЛІС для виконання чергового завдання.

У процесі ініціалізації системи, коли потік акторів завантажується в пам'ять задач, відбувається одночасне завантаження черги в буфер черги реконфігурації. Для забезпечення автономного виконання процесу реконфігурації в регістри буфера черги реконфігурації одноразово завантажується лише значення поля «тип операції I_i» (рис. 1). Із надходженням кожного наступного актора на компараторі, який входить у склад інтерфейсу блока черги реконфігурації (рис. 4), перевіряється наявність у черзі реконфігурації завдання на виконання, за відсутності якого нова функція ставиться у чергу для реконфігурації обчислювального середовища. Буфер черги реконфігурації побудовано на базі асоціативної пам'яті. Для реалізації буфера черги реконфігурації використовується структура асоціативної пам'яті, де як індекс використовується поле ідентифікатора актора N_i, а як тег – поле типу операції I_i (див. рис. 1). Детально принцип функціонування такої пам'яті описано у праці [7]. Тегом для завантаження черги і виконання реконфігурації є поле типу операції I_i. У комірках пам'яті реалізовано лічильники для керування пріоритетами виконання реконфігурації, у цьому випадку за правилом «перший прийшов – перший обслугований» (*FIFO, First Input – First Output*), а також ознака налаштування відповідної апаратної задачі на поверхні ПЛІС.

З огляду на унікальність імен N_i дескрипторів для кожного завдання, умова готовності даних для виконання завдання формується на виході компаратора (див. рис. 1) таким чином:

$$\omega_i^* = \begin{cases} 1, \textit{якщо } c_i = Q_i, \\ 0, \textit{якщо } c_i \neq Q_i, \end{cases}$$

де c_i – кількість прийнятих у керувальний ОМ дескрипторів з іменем N_i.

Керувальний сигнал ω_i^* надходить на керувальний вхід кон'юнктура у складі блока перевірки готовності заявки, на другий керувальний вхід якого надходить ознака готовності налаштування обчислювального середовища відповідного завдання N_i. Якщо на кон'юнктурі формується умова:

$$\omega = \omega_i^* \& j \ ,$$

готова для виконання заявка записується в буферну пам'ять заявок. Керувальний процесор ініціює виконання чергового завдання на поверхні ПЛІС. Порядок виконання формується у черзі буферної пам'яті заявок за принципом «перший прийшов – перший обслуговуваний»,

Дані з буфера черги реконфігурації вибираються за унікальними ідентифікаторами записів асоціативної пам'яті $[N_i].[I_i]$. Формат слова даних асоціативної пам'яті буфера черги реконфігурації зображено на рис. 5.

Запис даних у пам'ять реєстрації та пам'ять задач, формування заявки на підставі готовності акторів та даних виконуються згідно з відомою технологією, яку детально описано у працях [1, 5].

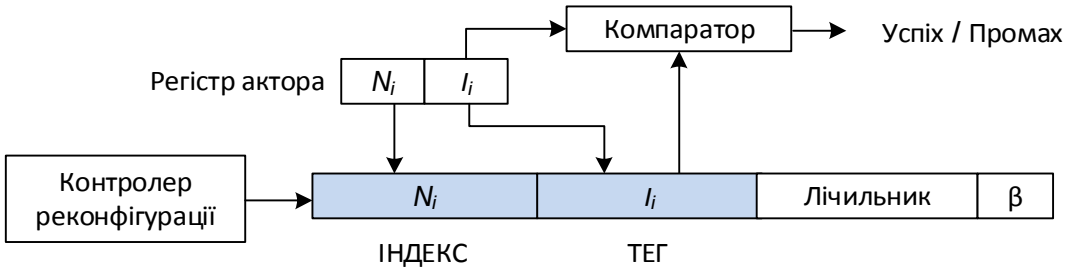

Рис. 5. Формат даних у буфері черги реконфігурації

У системі одночасно можуть розв'язуватися кілька задач у багатозадачному режимі роботи. У процесі реалізації багатопроцесорної структури обчислювального модуля будь-який процесор може виконувати функції керувального та виконавчого процесора (навіть одночасно). Керувальний процесор зчитує із зовнішньої пам'яті даних потік акторів і вхідних дескрипторів. Виконавчий процесор зчитує готову для виконання заявку і ініціює запуск завдання на ПЛІС і звернення до пам'яті даних для отримання вхідних даних для виконання завдання. Після виконання завдання виконавчий процесор повертає дескриптор даних у пам'ять реєстрації, який бере участь у подальшому формуванні наступної заявки. Один керувальний процесор у структурі обчислювального модуля, описаний у праці [6], одночасно виконує функції керувального і виконавчого процесора.

Керувальний процесор самостійно отримує необхідну інформацію з блока пам'яті заявок для виконання завдання, визначає адреси даних на підставі значень N_i і Q_i. Безпріоритетна дисципліна обслуговування заявок, описана у праці [1], дозволяє формувати всі заявки в одній черзі, що значно спрощує механізм формування заявок, сприяє зниженню витрат апаратури і спрощує вирішення конфліктів під час доступу до спільних ресурсів.

Модель реалізації обчислювального процесу на підставі вдосконалених апаратних засобів автоматичного розподілу завдань на динамічно реконфігуровне обчислювальне середовище зображено на рис. 6. Модель обчислень ґрунтується на визначенні цільової функції зменшення накладних витрат процесу відображення задач на реконфігуровне обчислювальне середовище [6],

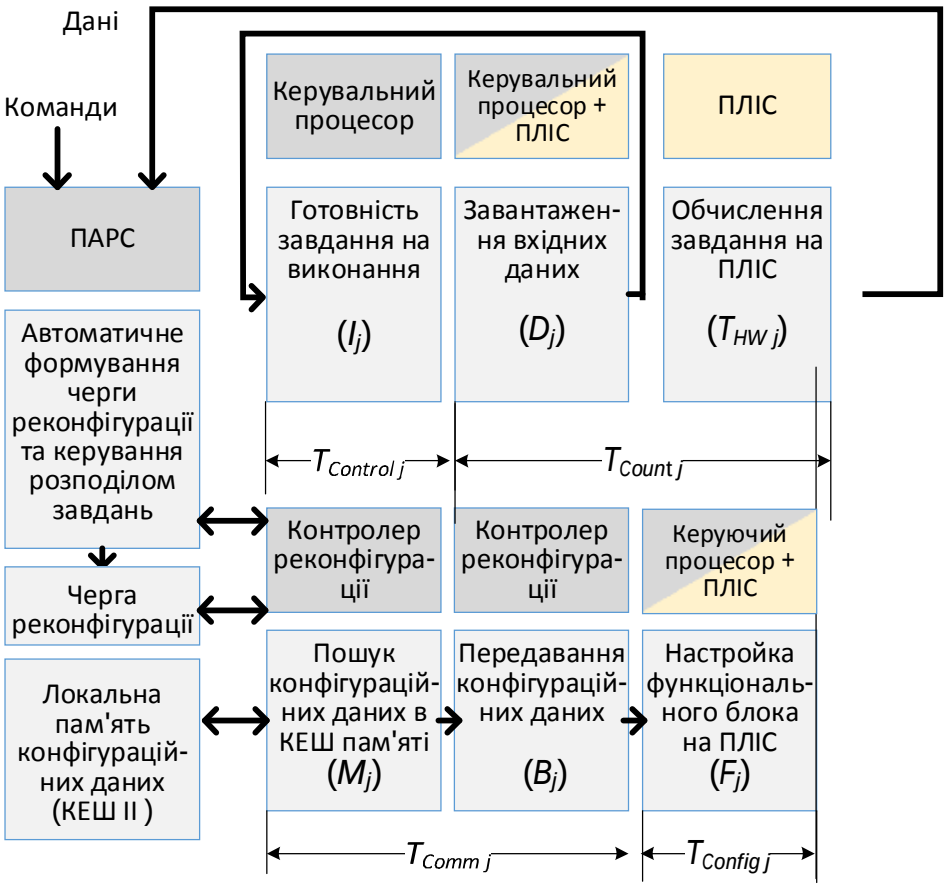

Рис. 6. Модель реалізації обчислювального процесу в системах, керованих потоком даних: КП – керувальний процесор, КП – контролер реконфігурації

яка цілком залежить від затримок, що супроводжують цей процес:

$$\min(T_{CONTROL} + \sum_j R_j) =$$

$$= \min(T_{CONTROL}) + \sum_j \min(T_{COMM\,j}) +$$

$$+ \sum_j T_{CONFIG\,j}$$

де T_{COMM} – час передавання конфігураційних даних із зовнішніх сховищ в інтерфейси ПЛІС; T_{CONFIG} – час конфігурування обчислювального середовища, при цьому $R = T_{COMM} + T_{CONFIG}$.

На відміну від традиційної моделі паралельної обробки інформації, автоматичний розподіл завдань на реконфігуровне обчислювальне середовище, що реалізовує модель потокових обчислень, дозволяє пришвидшити процес обробки інформації за рахунок ефективної реалізації прихованого паралелізму, скорочення циклу виконання завдань і зменшення непродуктивних часових витрат, у тому числі через спрощення процесу керування паралельною обробкою інформації і його апаратного пришвидшення.

Моделювання вдосконаленого пристрою автоматичного розподілу завдань та синхронізації

Імітаційна модель методу адаптивного відображення задач на реконфігуровне обчислювальне середовище, та імітаційна модель РКС, керованої потоком даних, описані у працях [7, 8]. На базі запропонованих імітаційних моделей досліджено процеси завчасної реконфігурації і повторного використання ресурсів ФБ. Програмні засоби розроблено кросплатформенною мовою *Java*. Імітаційна модель включає в себе такі блоки, реалізовані як програмні модулі, які емулюють функції відповідних апаратних функціональних елементів: *DataFlow* – головний модуль, що емулює функціонал всієї системи і об'єднує підлеглі модулі: *AppFormingService* – модуль, що відтво-

рює роботу ПАРС і містить функціонал контролера реконфігурації; *FPGA* – реконфігуровне обчислювальне середовище; *Core* – модуль, що виконує роль програмно-апаратного ядра для обчислення задач і може масштабуватися в налаштуваннях системи; *Actor, Descriptor, Task, Application, Data* – модулі, що виконують роль відповідних структур у потоковій системі і забезпечують збереження даних у пам'яті.

Досліджено серію обчислювальних алгоритмів, поданих потоковими графами, з різною кількістю однотипних задач та різним ступенем зв'язності. Графік залежності часу виконання обчислювальних алгоритмів з різною кількістю повторів однотипних задач зображено на рис. 7. На рис. 8 показано усереднену залежність часу виконання реконфігуровних обчислень для потокових алгоритмів від різних стратегій пришвидшення реконфігурації, а саме повторного використання реконфігуровних ресурсів (T_G) [6] та завчасної реконфігурації на базі автоматичного розподілу завдань (T_R) [7, 8]. Для оцінювання ефективності завчасної реконфігурації у РКС, керованих потоком даних, використано коефіцієнт пришвидшення $K = T_G / T_R$, де T_G – час виконання потокового алгоритму без будь-якого пришвидшення реконфігурації; T_R – час виконання обчислювального алгоритму із застосуванням методу адаптивного відображення задач на реконфігуровне обчислювальне середовище.

Повторне використання ресурсів ФБ апаратних задач на ПЛІС дозволяють збільшити швидкість процесу реконфігурації в середньому на 63%, що показано у праці [6].

При цьому ефективність методу адаптивного пришвидшення реконфігурації залежить від кількості однотипних задач в обчислювальному алгоритмі. Із графіка дослідження пришвидшення обчислювального процесу із застосуванням автоматичного розподілу завдань на базі моделі обчислень, керованих потоком даних, отримано середній коефіцієнт пришвидшення, що досягає двох при 20 і більше повторних однотипних завдань (рис. 8). При цьому завчасна реконфігурація дозволяє видалити майже весь непродуктивний час

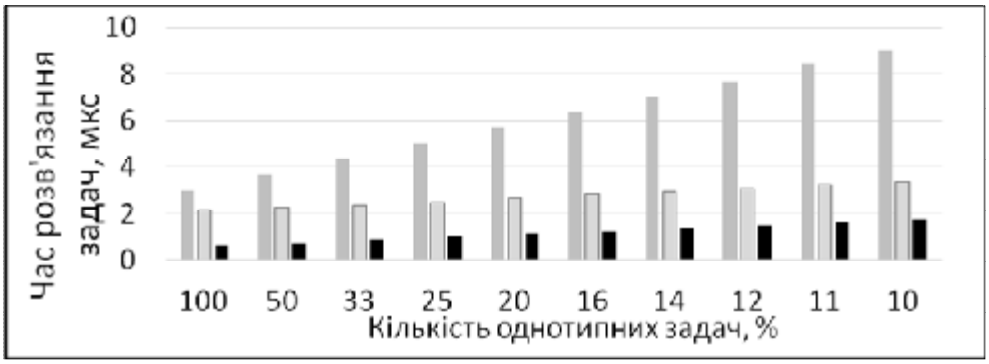

Рис. 7. Залежність комунікаційних затримок від кількості повторів однотипних функцій в обчислювальному алгоритмі: ▒ – реконфігурація з повторним використанням ресурсів ФБ; ■ – завчасна реконфігурація; □ – час виконання завдань на апаратурі ПЛІС

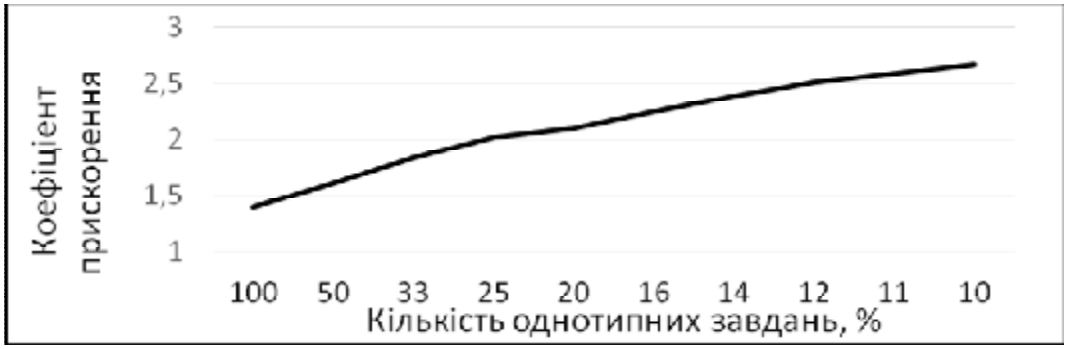

Рис. 8. Залежність коефіцієнта пришвидшення від кількості повторів однотипних задач

реконфігурації для будь-яких обчислювальних алгоритмів незалежно від кількості однотипних задач.

Висновки

Реалізація моделі обчислень, керованих потоком даних, сприяє уникненню проблем, пов'язаних із застосуванням традиційних технологій планування обчислень і розподілу обчислювальних ресурсів в РКС, що мають функціональні та апаратні обмеження.

Удосконалено метод автоматичного розподілу завдань, що дає змогу автоматизувати функцію керуванням обробкою інформації на апаратному рівні РКС, керованих потоком даних. Розроблені засоби автоматичного розподілу завдань на реконфігуровне обчислювальне середовище на ПЛІС за рахунок суміщення обчислювального процесу і процесу завчасної реконфігурації обчислювального середовища дозволяють скоротити непродуктивні витрати часу та продуктивності у процесі відображення потоку задач на реконфігуровне обчислювальне середовище.

У цілому вдосконалені засоби розподілу команд та апаратної синхронізації зменшують час звернення до загальних ресурсів, непродуктивне завантаження загального комунікаційного середовища і непродуктивні звернення до пам'яті, накладні витрати на паралельну обробку даних, ємність використовуваної вбудованої пам'яті ПЛІС. Це підвищує ефективність взаємодії між функціональними елементами ПАРС та ОМ РКС, а також розширює функціональні можливості ПАРС з урахуванням передумов для масштабування РКС.

Розроблені засоби апаратного автоматичного розподілу завдань та синхронізації процесів для РКС забезпечують автоматичне керування паралельною обробкою даних і розподілом завдань на реконфігуровне обчислювальне середовище прозоро для програмного шару обчислювальної системи, що звільняє операційну систему для розв'язання неспецифічних задач і дозволяє ефективно використовувати на рівні операційної системи традиційні технології паралельного програмування.

Моделювання методу адаптивного відображення завдань на базі розробленої апаратної імітаційної моделі ОМ РКС підтвердило теоретичні та формальні обґрунтування ефективності адаптивного відображення завдань на реконфігуровне обчислювальне середовище, що наведені в роботі, і визначило наступне:

– порівняно з технологією повторного використання обчислювальних ресурсів ФБ завчасна реконфігурація забезпечує у цілому в два рази пришвидшення обробки інформації в РКС і не залежать від частоти повторень однотипних функцій в алгоритмі;

– комплексне застосування механізмів повторного використання ресурсів ФБ і завчасної реконфігурації дозволяє видалити майже всі непродуктивні витрати під час процесу реконфігурації незалежно від частоти виконання одноти-

пних функцій протягом обчислювального процесу.

Список літератури

1. Жабин В. И. Архитектура вычислительных систем реального времени / В. И. Жабин. – К. : Век +, 2003. – 176 с.

2. *Каляев И. А*. Высокопроизводительные реконфигурируемые вычислительные системы на основе плис Virtex-6 и Virtex-7 / И. А. Каляев, А. И. Дордопуло, И. И. Левин, Е. А. Семерников // Параллельные вычислительные технологии (ПаВТ'2012), (26 – 30 марта 2012 г., Новосибирск, Россия). – Челябинск : ЮУрГУ, 2012. – С. 449 – 458.

3. *Жабин В. И.* Реализация вычислений под управлением потока дескрипторов данных в мультипроцессорных системах / В. И. Жабин // Электронное моделирование. – К. : ИПМЕ, 2003. – Т. 25, № 1. – С. 35 – 47.

4. *Жабин В. И.* Повышение эффективности параллельных вычислений в потоковых системах [Електронний ресурс] / В. И. Жабин, В. В. Жабина // Вісник Національного технічного університету України «КПІ». Інформатика, управління та обчислювальна техніка. – 2013. - Вип. 59. – С. 122-128.

5. *Жабін В. І.* Методи і засоби підвищення ефективності паралельних обчислювальних систем реального часу : автореф. дис. на здобуття наук. ступеня докт. техн. наук : спец. 05.13.13 «Обчислювальні машини, системи та мережі» / В. І. Жабін. – К. : 2006. – 36 с.

6. *Кулаков Ю. О.* Розробка методу прискорення реконфігурації в динамічно реконфігурованих обчислювальних системах / Ю. О. Кулаков, І. А. Клименко, М. В. Рудницький // Східно-Європейський журнал передових технологій. – Харків : УДА залізничного транспорту, 2015. – № 4/4 (76). – С. 25 – 30.

7. *Клименко І. А*. Засоби адаптивного відображення задач на реконфігуровну обчислювальну структуру в паралельних обчислювальних системах, що керуються потоком даних / І. А. Клименко, В. В. Ткаченко, О. М. Сторожук // Електроніка та зв'язок. – К. : НТТУ «КПІ», 2016. – Том 21, № 2 (91). – С. 71 – 77.

8. *Клименко І. А*. Спосіб автоматичногфо розпаралелювання задач із завчасною реконфігурацією на ПЛІС / І. А. Клименко, О. М. Сторожук // Тези доп. VII Міжнар. наук. конф. «Сучасні проблеми математичного моделювання, прогнозування та оптимізації (ОПТІМА 2016)», (21 – 22 квітня, 2016 р., Кам'янець-Подільський, Україна). – Кам'янець-Подільський : Кам'янець-Подільський національний університет імені І. Огієнка, 2016. – С. 93 – 94.

9. *Dümmler J.* Scalable computing with parallel tasks / J. Dümmler, T. Rauber, G. Rünger // Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS '09) (14 – 20 November 2009, Portland, Oregon, US). – ACM New York, NY, USA, 2009. – № 9 – P. 1 – 10

Статтю подано до редакції 22.05.2017