3. Використання для побудови ДЯ і організації процесу діагностування основних теоретичних засад експертних систем, теорії моделювання систем, штучного інтелекту.

4. Шляхи підвищення надійності СОС організацією багатоверсійного програмування.

5. Аналіз впливу додаткових обсягів пам'яті на проведення діагностування, параметрів та надійності діагностичних каналів на ефективність СОС.

1. Щербаков Н.С., Достоверность работы цифровых устройств. – М.: Машиностроение, 1989. – 224 с. 2. Ваврук Є.Я. Вибір діагностичного ядра для системи опрацювання сигналів // Зб. наук. праць ІПМЕ НАНУ. – 2006. – Вип. 34. – С. 23–29. 3. Ваврук Є.Я. Організація контролю і діагностики бортових пристроїв опрацювання сигналів у режимі реального часу// Збірник матеріалів міжвузівської науково-технічної конференції науково-педагогічних працівників. – Львів: Ліга-Прес, 2006. – С. 191–192.

УДК 004.274

С.О. Власенко
Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин

ПОБУДОВА АДАПТИВНИХ ІНТЕРФЕЙСІВ КОМП'ЮТЕРНИХ СИСТЕМ НА ОСНОВІ КОНФІГУРОВАНИХ АРХІТЕКТУР

Розглянуто проблеми побудови адаптивних інтерфейсів у комп'ютерних системах та запропоновано реалізацію одного з методів на основі механізму часткової реконфігурації ПЛІС.

The problems of adaptive interfaces construction for the computer systems and it's realization on basis of partly-configured FPGA approach have been presented.

Вступ

Конфігуровані архітектури [3], або такі архітектури, що дають змогу збільшувати свої функціональні можливості під впливом наборів спеціальних керівних сигналів (сигналів конфігурування) з кожним роком все частіше використовують. Значний поштовх цьому процесу надає стрімкий розвиток технології FPGA (Field Programmable Gate Array) – програмованих логічних інтегральних Схем (ПЛІС). Сам факт появи як конфігурованих систем, так і ПЛІС зокрема, спричинили так звану "хвилю" намагань досягти з їх допомогою значного прискорення виконання різного роду обчислень аж до побудови конфігурованих суперкомп'ютерів. Дійсно, багатьом розробникам вдалося реалізувати на конфігурованих архітектурах прискорення деяких видів задач у сотні та навіть тисячі разів порівняно з їх виконанням на універсальних робочих станціях [2]. Зазвичай, найбільшого прискорення вдається досягти тоді, коли задача реалізується на великій кількості однорідних елементів, що працюють паралельно. Але, незважаючи на чудові "лабораторні" результати, на практиці не завжди вдається досягти загального збільшення швидкодії системи. Проектування архітектур паралельного оброблення даних є досить складним та, що дуже важливо, довготривалим процесом. Отже, конфігуровані комп'ютери з часом позиціонувалися як засоби, що дають змогу зменшити час виконання певного виду алгоритмів за досить значних витрат на їхнє проектування.

У цьому контексті одним з найперспективніших напрямків [1] використання конфігурованих архітектур стає їх використання як засобів комунікації та складових частин систем реального часу.

Огляд літературних джерел

У питанні побудови адаптивних інтерфейсів з можливістю самореконфігурації було випробувано досить багато підходів [1]. Ще у 1987 році Борріелло (Borriello) запропонував ввести в протокол передавання даних спеціальні позначки. Ці позначки, володіючи порівняно невеликою інформаційною надлишковістю, дають змогу транслювати один протокол в інший за одним універсальним алгоритмом. Другий підхід, запропонований Мак Мілланом (McMillan) та Пассероном (Passeron) [3], заснований на так званих "п'яти кроках" реалізації протоколу. Якщо ці п'ять етапів вдається здійснити, реалізація протоколу стає можливою також за єдиним алгоритмом. Треба зазначити, що в цих двох підходах за зовнішньою "простотою" (застосування універсального алгоритму трансляції) ховається необхідність використання складного математичного апарату для розроблення та практичної реалізації цих методів. Ще один поширений підхід [4] полягає у здійсненні апаратної реалізації основних, найпоширеніших протоколів передавання даних (у певній конкретній області) та керованому спрямуванні потоків даних на потрібну в цей момент частину інтерфейсу. Очевидною в цьому випадку є велика надлишковість (апаратна та програмна), що обмежує область використання даного підходу випадками, в яких апаратні витрати, габаритні розміри та вартість системи не є вирішальними факторами для їх реалізації.

Усі перераховані підходи мають багато спільних недоліків, наприклад такі, як значні обмеження трансляції з одного протоколу в інший (ймовірність виникнення випадків, для яких трансляція виявиться неможливою, є досить висока – біля 40 %) та відсутність механізмів для автоматичної генерації протоколів.

Постановка задачі

Аналіз проблем комунікації в комп'ютерних системах та підходів до їх вирішення дає змогу сформулювати напрямки можливих досліджень, спрямованих на покращання основних характеристик систем цього класу, зокрема, таких як час на проектування, повнота функціональності (ступінь можливості забезпечувати з'єднання з якомога більшою кількістю систем різних типів), апаратні витрати та швидкодія, які істотно збільшуються із зростанням попередньої характеристики.

У статті запропоновано шляхи побудови адаптивного інтерфейсу (АІ), який дає змогу використовувати ПЛІС для організації передачі даних за різними протоколами в реальному часі. Запропонований підхід передбачає технологію часткової реконфігурації ПЛІС фірми Xilinx для генерації та автоматичного конфігурування так званих інтерфейсних блоків [5].

Макроструктура інтерфейсного блоку як основа АІ

Інтерфейсні блоки (ІФБ) є адаптивними модулями, що можуть бути згенеровані інтерактивно з використанням так званих інтерфейсних описів (протоколів обміну з конкретною частиною комп'ютерної системи) [8]. Ключовим моментом структури ІФБ є поділ на три частини: вхідний покажчик протоколу $ПП_{вх}$, вихідний покажчик протоколу $ПП_{вих.}$ та послідовнісний покажчик ПосП (рис. 1). Послідовнісний покажчик є проміжним між $ПП_{вх}$ та $ПП_{вих}$ та призначений керувати процесом конвертації. Кожен покажчик має керівні сигнали, які дають змогу задати декілька режимів виконання операцій (залежно від специфіки трансльованого протоколу). Спеціалізований процесор (СП) здійснює внутрішній контроль за ІФБ. З цією метою він надсилає опитуючі сигнали до $ПП_{вх.}$ та $ПП_{вих}$ і аналізує їх статус. Цей процес має назву "вертикальна комунікація". Дані та квитуючі сигнали з'єднуються в горизонтальному напрямку. Цю структуру називають макроструктурою ІФБ, яка реалізується набором автоматів з кінцевим станом (АКС) [8].

Система функціонує так. Потік даних від однієї задачі (під задачею розуміємо протокол обміну певної складової частини системи, який необхідно адаптувати до іншої за допомогою нашого ІФБ) за її протоколом потрапляє у $ПП_{вх}$, що вилучає ще не оброблену інформацію з протоколу та надсилає її до ПоСП у зручній для нього формі. Ця інформація перевпорядковується відносно цього протоколу покажчиком ППвих і, нарешті, модифікована інформація надходить у вихідний протокол. Для реалізації різних протоколів кожен покажчик може генерувати різні

режими. У деяких випадках для збереження проміжних даних до цієї системи вводять пам'ять. За умов використання пам'яті можна конвертувати послідовний потік даних у паралельний.

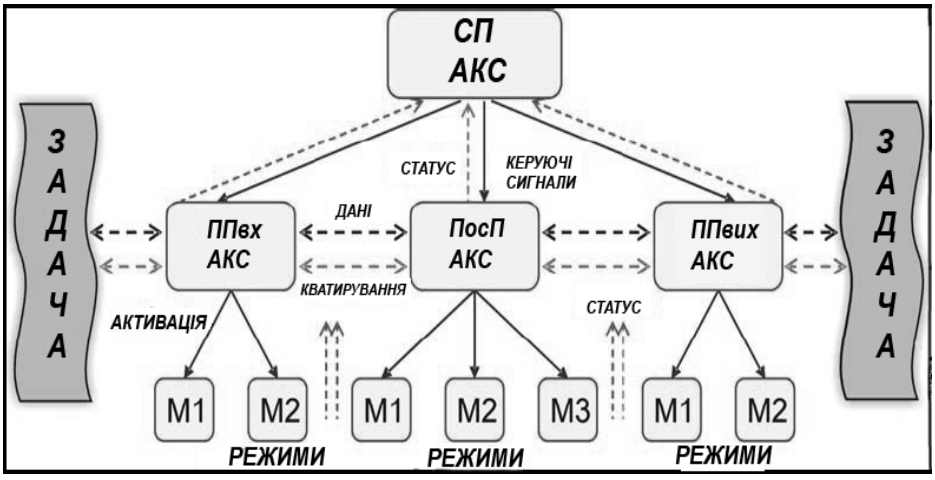

Рис. 1. Макроструктура ІФБ

Головною перевагою підходу є те, що повна реалізація інтерфейсу можлива без реалізації усіх АКС, оскільки їх можна отримати шляхом реалізації додаткових протоколів для одного АКС.

У загальному випадку, маршрутизація від АКС до АКС не завжди можлива без знання семантики даних, що передаються. Координацію усіх АКС здійснює спеціалізований процесор (СП), також реалізований як АКС. Завдяки наявності СП стає можливою конфігурація в реальному часі. Часові аспекти цього процесу реалізовані всередині СП. За допомогою СП можна виявити несправні вузли комірок ІФБ.

Паралельна адаптація інтерфейсу в реальному часі

Завдяки технології часткової реконфігурації під час використання ІФБ існує можливість паралельно змінювати конфігурацію кожного ІФБ, фактично перекомутовуючи зв'язки адаптивного інтерфейсу (розривати з'єднання між одними частинами комп'ютерної системи та з'єднувати інші залежно від поточного режиму функціонування). Реконфігурація ПЛІС здійснюється "за стовпцями" [3]. Елементарну частину ПЛІС називають секцією. Декілька секцій можуть бути об'єднані у конфігураційний блок (КБ). З технологічних причин сусідні КБ мають бути відокремлені макрокомірками (МК). У системі можуть бути присутні як конфігуровані, так і статичні блоки. Статичні блоки можуть бути представлені як окремий випадок конфігурованого блоку.

Під час заміни конфігурації виникають такі проблеми [11]:

– як сформулювати вимоги до нової конфігурації інтерфейсу?
– як визначити ту область ПЛІС, що підлягає реконфігурації?
– як вилучити зв'язки тієї задачі, що підлягає реконфігурації?
– як гарантувати коректність зв'язків нової конфігурації?
– як гарантувати проходження процесу реконфігурації без спотворень?

Після конфігурації стає необхідною адаптація інтерфейсу шляхом налаштування структури зв'язків під нову задачу. Заміна конфігурації потребує застосування концепції модульного проектування. Цей підхід дає змогу визначити для кожного модуля окрему область ПЛІС, що необхідно для реалізації часткової реконфігурації. Модульним проектуванням можна створювати окремі конфігураційні потоки для кожного модуля. Під час індексування цих потоків отримуються ідентифікатори задачі (ІФБ ІДФ), що залежать від області розміщення. Їх використовують для ідентифікації різних покажчиків протоколів ІФБ та послідовнісних покажчиків ІФБ. У випадку, коли є декілька ІФБ з ідентичними ПП або ПосП, необхідно створювати окремі бітові потоки для кожного з цих покажчиків, оскільки їх використовують в різних частинах ПЛІС. Завдяки ІФБ ІДФ у випадку заміни задачі покажчики ІФБ також будуть замінені.

Спеціалізований процесор як основна керівна ланка АІ

СП є класичним прикладом реалізації концепції plug-and-play у ПЛІС (рис. 2). Він гарантує автоматичну реконфігурацію покажчика ІФБ при заміні конфігурації. СП також сигналізує про готовність конфігурації до завантаження [7].

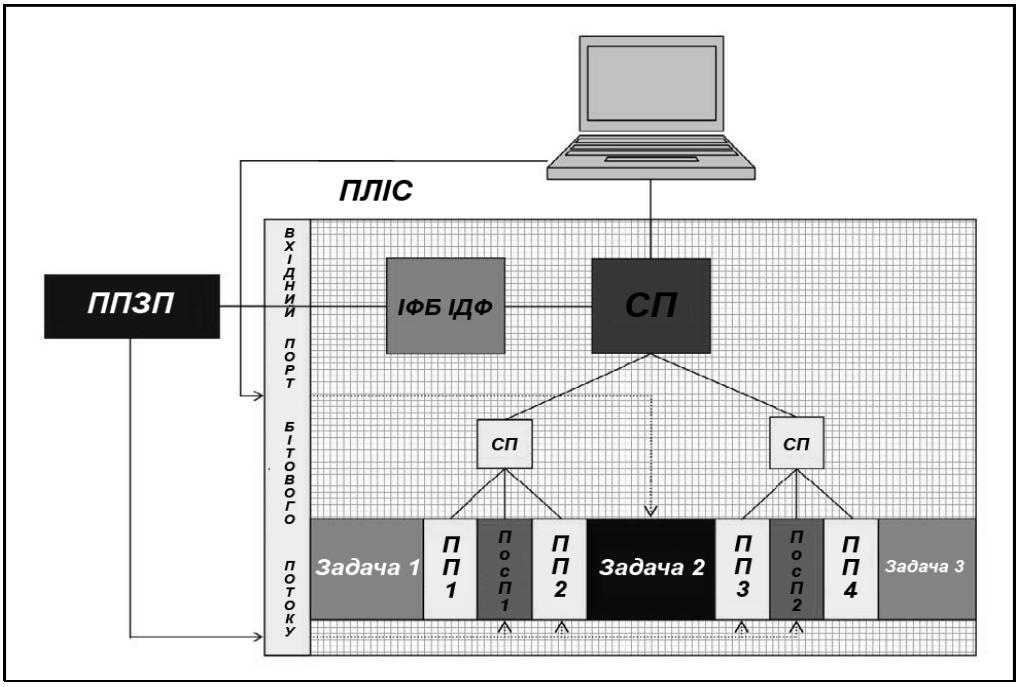

Рис. 2. Компоненти архітектури з самореконфігурацією

Архітектура з самореконфігурацією має такий принцип роботи. Персональний комп'ютер розпочинає процес заміни конфігурації, що передбачає оновлення таблиці статусу СП. Ця таблиця містить ідентифікатори (ІДФ) усіх модулів. Оновлення таблиці статусу автоматично викликає початок процесу реконфігурації. Для запобігання втрати інформації реконфігурація не дозволяється під час проведення поточної стадії обчислень. Отже, вхідний порт бітового потоку в цей час заблокований. СП з'єднується з ІФБ, які мають відношення до заміненої задачі, що викликає зупинку поточних покажчиків. ІФБ ІДФ являють собою таблицю в пам'яті, в якій зберігаються усі ІФБ ІДФ та адреси їхніх бітових потоків. Покажчики бітових потоків також зберігаються в пам'яті. СП може ініціювати завантаження покажчиків бітових потоків з пам'яті в ПЛІС. Коли цей процес завершується, СП перевіряє доступність відповідних покажчиків. У випадку, коли ІФБ ІДФ недоступний, СП видає відповідний сигнал до ПК. Якщо усі покажчики містяться в пам'яті, СП надсилає відповідні сигнали "підлеглим" СП, інформуючи, що відповідні їм покажчики будуть реконфігуровані. Після отримання цього сигналу СП вимикають відповідні покажчики, готуючи їх до реконфігурації. Після того СП завантажує з пам'яті відповідні бітові потоки. Коли завантажено нові покажчики, СП пропонує активувати їх. Після цього ПЛІС з новою задачею та новим інтерфейсом з можливістю самореконфігурації вважається повністю готовим для подальшого функціонування.

Тестування АІ

Для підтвердження коректності запропонованого підходу був реалізований інтерфейс, заснований на ІФБ (рис. 3) [10].

Інтерфейс реалізує протокол керування трьома двигунами. У певний момент часу ініціюється заміна протоколу, оскільки потрібно керувати вже п'ятьма двигунами. Заміна протоколу відбувається в повній відповідності з запропонованим підходом. При цьому послідовнісний покажчик ПосП1 замінюється на покажчик ПосП2. Для демонстрування конфігурації в реальному

часі область кожної задачі було з'єднано зі світлодіодом. При цьому під час реконфігурації світлодіоди від усіх інших ІФБ продовжували світитися, що свідчить про повне підтвердження очікуваних результатів.

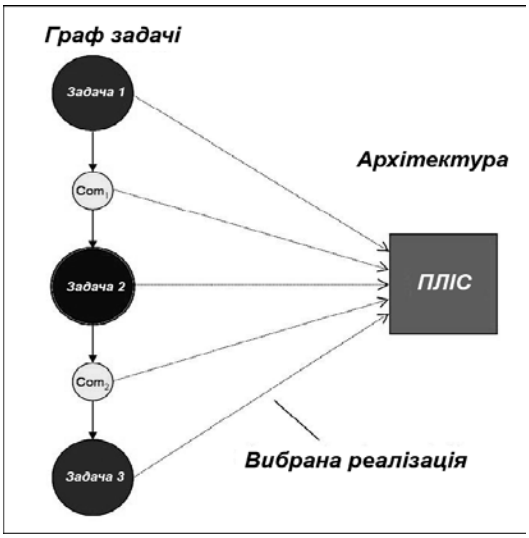

Рис. 3. Граф тестової задачі

Висновки

У результаті аналізу проблем комунікації в комп'ютерних системах та проведених досліджень було запропоновано підхід до побудови адаптивного інтерфейсу на основі інтерфейсних блоків. АІ, побудований відповідно до цього підходу, дає змогу здійснити автоматичну реконфігурацію внутрішніх зв'язків комп'ютерної системи на основі механізму часткової реконфігурації ПЛІС та за значного збільшення функціональних можливостей уникнути збільшення апаратної складності та зменшення швидкодії, що супроводжують більшість існуючих підходів. Витрати часу на проектування систем цього типу також істотно зменшуються. Оптимізація цих характеристик може бути темою подальших досліджень.

1. Bednara M., Danne K. Deppe M. Oberschelp O., Slomka F., and Teich J. Design and implementation of digital linear control systems on reconfigurable hardware. EURASIP Journal on Applied Signal Processing, to appear, 2003. 2. Bergmann N., Dawood A. Adaptive Interfacing with Reconfigurable computers // IEEE Trans. RSP'04, Sept. 2001 PDPTA), Las Vegas, Nevada, June 2003. 3. Borriell G. and Katz R.H. Synthesis and optimization of interface transducer logic. In Proceedings of 1987 IEEE International Conference on Computer Design (ICCD), 1987. 4. Danne K., Bobda C. and Kalte H.. Increasing efficiency by partial hardware reconfiguration: Case study of a multicontroller system. Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications. 5. Hardt, Wolfram and Visarius, Markus and Ih mor, Stefan. Rapid Prototyping of Real-Time Interfaces. In Field Programmable Logic (FPL) – Poster Session, Belfast, Northern Ireland, UK, October 2001. 6. Ihmor S., Hardt W. Runtime Reconfigurable Interfaces – The RTR-IFB Approach // IEEE Trans. IPDPS'04. – Apr. 2004. 7. Ihmor S., Hardt W. Self-Reconfigurationof communication interfaces // IEEE Trans. RSP'04. 8. Ihmor, Stefan and Visarius, Markus and Hardt, Wolfram. A Consistent Design Methodology for Configurable HW/SWInterfaces in Embedded Systems. In Proc. of the IFIP 17th World Computer Congress – TC10 Stream on Distributed and Parallel Embedded Systems: Design and Analysis of Distributed Embedded Systems. – Montreal, Canada, Aug. 2002. 9. Ihmor, Stefan and Visarius, Markus and Hardt, Wolfram. A Design Methodology for Application-specific Real-Time Interfaces.In Proceedings of 2002 IEEE International Conferenceon Computer Design (ICCD): VLSI in Computers& Processors // IEEE International Conference on ComputerDesign, Freiburg, Germany, Sept. 2002. 10. Passerone R., Rowson J., and Sangiovanni-Vincentelli A. Automatic synthesis of interfaces.