УДК 004.382, 004.383

В.А. МЕЛЬНИК
Національний університет „Львівська політехніка”

ОСНОВИ ПОБУДОВИ ТА ОРГАНІЗАЦІЇ ФУНКЦІОНУВАННЯ РЕКОНФІГУРОВНИХ КОМП’ЮТЕРНИХ СИСТЕМ

В статті запропоновано визначення основних понять реконфігуровних комп’ютерних систем, показано базові типи їх архітектури, сформульовано спосіб виконання обчислень в цих системах, а також визначено проблемні питання їх застосування та окреслено шляхи вирішення цих питань.

Ключові слова: реконфігуровна комп’ютерна система, типи архітектури РККС, спосіб виконання обчислень в РККС.

In this paper are proposed definitions of reconfigurable computing basic terms, shown basic types of the reconfigurable computer systems architecture, formulated method of computations execution in reconfigurable computer system, and determined their application problems and identified the ways of these problems solving.

Keywords: reconfigurable computer system architecture RCCS types, the way calculations in RCCS.

Вступ

Реконфігуровні комп’ютерні системи (РККС) набули сьогодні значного поширення, що пов’язано з можливістю досягнення ними високих показників продуктивності шляхом апаратного прискорення виконання обчислювальних завдань в кристалах програмовної логіки, що входять до їх складу.

Сьогодні існує велика кількість практичних реалізацій та декілька варіантів архітектури реконфігуровних комп’ютерних систем, які відрізняються методами під’єднання кристалів програмовної логіки та характеристиками використовуваних інтерфейсів.

Аналіз досліджень та публікацій. Реконфігуровні комп’ютерні системи, їх будову та організацію функціонування на різних рівнях висвітлено в роботі [1]. В роботі [2] розглянуто питання створення високопродуктивної РККС на основі персонального комп’ютера та реконфігуровного прискорювача, побудованого на основі програмовних логічних інтегральних схем. В роботі [3] висвітлено типи архітектури РККС та методи їх проектування. В роботах [4– 26] описано приклади реалізацій РККС різної архітектури, їх елементів та допоміжних засобів.

Виділення невирішених частин. Поряд зі значним обсягом літератури, яка висвітлює різні аспекти будови, проектування, застосування РККС, описує реалізацію різноманітних прикладних алгоритмів в РККС, сьогодні відсутні літературні джерела, в яких комплексно висвітлено основні поняття РККС, підходи до її побудови та спосіб виконання нею обчислювальних завдань. Це в значній мірі ускладнює розуміння суті і теоретичних основ функціонування РККС та уповільнює розвиток напряму реконфігуровних обчислень в цілому.

Формулювання цілей. В даній статті запропоновано визначення основних понять РККС, показано базові типи їх архітектури, сформульовано спосіб виконання обчислень в РККС, а також визначено проблемні питання застосування РККС та окреслено шляхи їх вирішення.

1. Основні поняття РККС

Реконфігуровна комп’ютерна система – це комп’ютерна система, яка містить універсальний (хост) комп’ютер, реконфігуровне середовище на основі одного або більше кристалів інтегральних схем програмовної логіки для виконання обчислювально складних завдань чи їх частин, та програмне забезпечення, необхідне для зміни конфігурації реконфігуровного середовища.

Реконфігуровність комп’ютерної системи – її здатність змінювати конфігурацію (налаштовувати внутрішню структуру функціональних вузлів та зв’язків між ними) для оптимального відображення виконуваних нею обчислювальних алгоритмів на апаратному рівні з метою забезпечення максимальної продуктивності їх виконання.

Хост-комп’ютер – універсальний комп’ютер, до якого під’єднується реконфігуровне середовище при побудові РККС.

Реконфігуровне середовище – набір інтегральних схем програмовної логіки (найчастіше ПЛІС – програмовних логічних інтегральних схем), в яких синтезують спеціалізовані процесори для апаратного виконання обчислювальних завдань чи їх частин.

Реконфігуровний апаратний прискорювач – пристрій, що прискорює виконання обчислювальних завдань в універсальному комп’ютері шляхом апаратного виконання цих завдань чи їх частин, який складається з реконфігуровного середовища та допоміжного обладнання для зміни конфігурації реконфігуровного середовища і організації його взаємодії з універсальним комп’ютером через один або декілька стандартних інтерфейсів.

ПЛІС (англ. *FPGA – Field Programmable Gate Array*) – інтегральна схема, яка містить матрицю конфігуровних логічних комірок, на основі якої можна створити інтегральну схему апаратно-орієнтованого спеціалізованого процесора шляхом задіяння потрібних логічних елементів комірок та налаштування зв’язків між ними. Внаслідок двох факторів: а) закладеної універсальності логічних комірок, і б) технологічних особливостей, пов’язаних з реконфігуровністю, – спеціалізовані процесори, реалізовані на

базі ПЛІС (або іншими словами, синтезовані у ПЛІС), поступаються продуктивністю замовним інтегральним схемам, однак дозволяють досягти повної розпаралеленості виконання алгоритму. Технологія виготовлення ПЛІС дозволяє здійснювати їх конфігурування доступними засобами і необмежену кількість разів.

Конфігурування ПЛІС – завантаження файлу конфігурації до кристалу ПЛІС.

Файл конфігурації ПЛІС – бінарний файл, який отримується в результаті логічного синтезу програмної моделі спеціалізованого процесора і встановлює функції конфігуровних логічних комірок ПЛІС та налаштовує зв'язки між ними.

Програмна модель спеціалізованого процесора – модель спеціалізованого процесора мовою опису апаратних засобів, яка відображає його архітектуру на рівні міжрегістрових передач або нижчому.

Логічний синтез програмної моделі спеціалізованого процесора – трансляція програмної моделі спеціалізованого процесора з рівня міжрегістрових передач на рівень елементів конфігуровних логічних комірок цільової ПЛІС.

2. Типи архітектури РККС

Архітектуру РККС класифікують за ступенем зв'язаності універсального процесора з реконфігуровним середовищем. В роботі [3] запропоновано п'ять класів архітектури РККС, які є базовими сьогодні.

На рис. 1 показано архітектуру слабозв'язаної РККС, в якій універсальний процесор взаємодіє з реконфігуровним середовищем через інтерфейс введення / виведення як з зовнішнім по відношенню до обчислювальної системи пристроєм. Швидкість обміну інформацією тут є достатньо низькою, тому така архітектура виправдана для обчислювальних завдань, виконання яких не вимагає тісної взаємодії універсального процесора з спеціалізованим. Приклади такої архітектури РККС наведено в працях [4, 5].

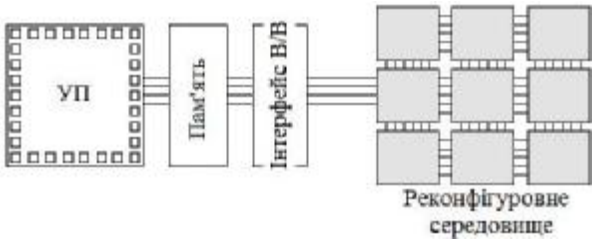
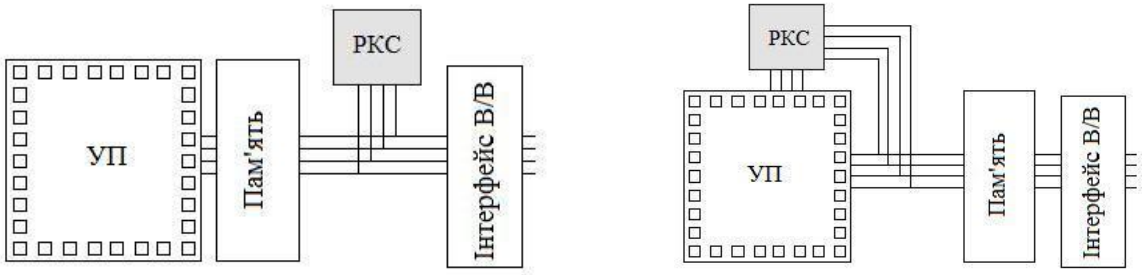


Рис. 1. Архітектура слабозв'язаної РККС

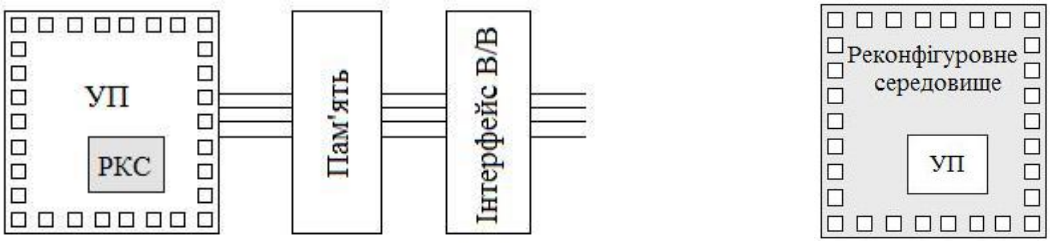
На рис. 2 показано два типи архітектури тіснозв'язаної РККС, в якій реконфігуровне середовище під'єднано до універсального процесора через системну шину, що забезпечує високу швидкість взаємодії. Спеціалізований процесор тут функціонує подібно до співпроцесора. Такі типи архітектури описано в працях [6 – 13].



а **б**

Рис. 2. Типи архітектури тіснозв'язаної РККС

На рис. 3 показано два типи архітектури інтегрованої РККС, які дозволяють досягти найвищої швидкості взаємодії універсального процесора з спеціалізованим. В першому випадку (рис. 3а) спеціалізований процесор використовується як один з операційних пристроїв універсального процесора шляхом розширення його системи команд (приклади таких реалізацій наведено в працях [14 – 17]). В другому (рис. 3б) універсальний процесор є вбудованим в реконфігуровне середовище РККС і виконує функції керування спеціалізованими процесорами, будучи при цьому «жорстко» реалізованим під час виготовлення кристалу ПЛІС (приклади реалізацій наведено в працях [18, 19]), або реалізованим як програмна модель [20 – 25].



а **б**

Рис. 3. Типи архітектури інтегрованої РККС

РККС можуть бути багатопроцесорними, тобто містити більше одного універсального процесора і реконфігуровне середовище, певним чином розподілене між ними. Як приклад, можна навести реконфігуровний суперкомп'ютер *Maxwell* [26] альянсу *FHPCA*, який побудовано на базі шасі сервера *IBM BladeCenter*, де розміщено 32 серверні плати. Кожна серверна плата включає в себе:

- процесор *Intel Xeon* з частотою 2,8 ГГц;
- 1 ГБ пам'яті;
- плату реконфігуровного прискорювача *Nallatech H*101 на основі ПЛІС *Xilinx Virtex* 4 *LX*160. Плата прискорювача містить власну пам'ять об'ємом 512 МБ;
- плату реконфігуровного прискорювача *Alpha Data ADM-XRC-4FX* на основі ПЛІС *Xilinx Virtex* 4 *FX*100. Плата прискорювача містить власну пам'ять об'ємом 1024 МБ.

3. Спосіб виконання обчислень в РККС

Розглянемо, яким способом виконуються обчислювальні завдання в РККС. Для повноти ілюстрації приймемо, що РККС містить K_{UPP} універсальних процесорів та K_{FPGA} кристалів ПЛІС, які формують її реконфігуровне середовище, причому до кожного з універсальних процесорів під'єднано один або більше кристалів ПЛІС.

Спосіб виконання обчислювального завдання T_{in} в РККС полягає в тому, що спочатку розробник здійснює конфігурування РККС, тобто налаштовує її на виконання обчислювального завдання T_{in}, виконуючи наступні дії:

1. Створює k паралельних програм P_{in_i}, $i=\overline{1...k}$, виконання обчислювального завдання T_{in}, причому значення k залежить від кількості паралельних складових в алгоритмі вирішення обчислювального завдання T_{in}, і може знаходитись в межах $1 \le k \le K_{UPP}$. Таким чином, $T_{in} = P_{in_1} \,\&\, P_{in_2} \,\&\, ... \,\&\, P_{in_k}$. Максимальної ефективності використання ресурсів універсальних процесорів РККС можна досягти за умови, коли $k = K_{UPP}$.

2. Здійснює програмно-апаратний розподіл кожної i-ї вхідної програми P_{in_i}, $i=\overline{1...k}$, шляхом виділення в ній доцільних для апаратної реалізації фрагментів F_m, $m=\overline{1...n_i}$, де n_i – кількість виділених фрагментів в програмі P_{in_i}, та їх заміни на команди роботи з реконфігуровним середовищем відповідно до архітектури РККС, і формує множину $\mathbf{P_{UPP}} = \left\{P_{UPP_i}, i=\overline{1...k}\right\}$ програм для виконання універсальними процесорами. Якщо позначити кількість програм множини $\mathbf{P_{UPP}}$, у яких виділено доцільні для апаратної реалізації фрагменти, як l, то можливі два випадки:

- $l = k$, якщо у всіх P_{in_i}, $i=\overline{1...k}$ виділено доцільні для апаратної реалізації фрагменти.
- $l < k$, якщо в деяких P_{in_i}, $i=\overline{1...k}$ не виділено доцільних для апаратної реалізації фрагментів.

3. Виконує компіляцію множини $\mathbf{P_{UPP}} = \left\{P_{UPP_i}, i=\overline{1...k}\right\}$ програм для виконання універсальними процесорами і формує множину об'єктних (виконавчих) файлів $\mathbf{obj} = \left\{obj_i, i=\overline{1...k}\right\}$.

4. Створює на мові опису апаратних засобів множину $\mathbf{PMSP} = \left\{PMSP_i, i=\overline{1...l}\right\}$ програмних моделей спеціалізованих процесорів, в якій кожен спеціалізований процесор $PMSP_i$ виконує доцільні для апаратної реалізації фрагменти F_m, $m=\overline{1...n_i}$ програми P_{in_i}, $i=\overline{1...l}$, де l – кількість програм множини $\mathbf{P_{UPP}}$, у яких виділено доцільні для апаратної реалізації фрагменти. Програмні моделі спеціалізованих процесорів найчастіше представляються мовами опису апаратних засобів на рівні міжрегістрових передач.

5. Виконує логічний синтез програмних моделей спеціалізованих процесорів $PMSP_i$ і формує множину файлів конфігурації $\mathbf{conf} = \left\{conf_q, q=\overline{1...K_{FPGA}}\right\}$ ПЛІС реконфігуровного середовища, де K_{FPGA} – кількість кристалів ПЛІС, які формують реконфігуровне середовище РККС.

6. Виконує синтез множини $\mathbf{SP} = \left\{SP_i, i=\overline{1...l}\right\}$ спеціалізованих процесорів в ПЛІС реконфігуровного середовища шляхом завантаження множини **conf** файлів конфігурації безпосередньо до ПЛІС або до пам'яті конфігурацій, під'єднаної до ПЛІС;

В результаті виконання наведених вище дій РККС є готовою до виконання обчислювального завдання T_{in}.

Далі виконання завдання ініціюється розробником і відбувається під керуванням операційної

системи, яка після отримання відповідної команди завантажує множину **obj** об'єктних файлів програм P_{UPP_i} до виконання універсальними процесорами. Кожен з універсальних процесорів UPP_i під час виконання програми P_{UPP_i} взаємодіє з спеціалізованим процесором SP_i та з іншими вузлами РККС.

Під час виконання завдання використовуються наступні основні типи операцій:

1. операції опрацювання даних з використанням стандартних команд універсальних процесорів;
2. операції опрацювання даних спеціалізованими процесорами в реконфігуровному середовищі РККС;
3. операції обміну даними між універсальними процесорами та спеціалізованими процесорами в реконфігуровному середовищі РККС;

Порядок взаємодії залежить від архітектури РККС. Для організації взаємодії універсальних процесорів зі спеціалізованими необхідно забезпечити сумісність інтерфейсів, через які ця взаємодія здійснюється. Це завдання вирішується або включенням до складу реконфігуровного середовища апаратних частин контролерів інтерфейсів, або адаптацією інтерфейсів спеціалізованих процесорів SP_i на етапі створення їх програмних моделей.

Для забезпечення виконання РККС обчислювальних завдань згідно з описаним вище способом до її складу повинні входити наступні програмні засоби:

- Для створення, відлагодження та компіляції програм – середовища програмування мовою високого рівня.
- Для розроблення та відлагодження програмних моделей спеціалізованих процесорів – середовища розробки, моделювання та верифікації проектів мовами опису апаратних засобів, наприклад, *ModelSIM* від *Mentor Graphics*, *Active-HDL* від *Aldec* та ін.
- Для виконання логічного синтезу програмних моделей спеціалізованих процесорів та конфігурування ПЛІС реконфігуровного середовища РККС – засоби проектування обчислювальних пристроїв на базі ПЛІС, на основі яких побудовано реконфігуровне середовище, які пропонуються на ринок виробниками ПЛІС, наприклад *ISE*, *Alliance*, *Foundation* компанії *Xilinx*, *Quartus* II, *Max*+II компанії *Altera* (цим компаніям належить левова частка світового ринку ПЛІС).
- Для організації взаємодії універсальних процесорів зі спеціалізованими – драйвери реконфігуровного середовища РККС та програмні частини контролерів інтерфейсів, через які ця взаємодія здійснюється.

Програмно-апаратний розподіл обчислювальних завдань виконується розробниками з допомогою засобів моделювання або з використанням профілювальників, які надають статистику виконання програми, зокрема час і частоту виконання її окремих фрагментів, що дає змогу виявити в програмі ті фрагменти, на які припадає найбільша кількість обчислень.

4. Проблемні питання застосування РККС і шляхи їх вирішення

Аналізуючи організацію функціонування РККС і спосіб виконання в ній обчислювальних завдань, можна визначити проблемні питання застосування РККС, в першу чергу наступні:

1. Оператори систем виконують програмно-апаратний розподіл вхідних завдань з допомогою засобів моделювання або на основі результатів аналізу кодів вхідних програм засобами профілювання, що вимагає значних часових затрат.
2. Часта відсутність на ринку програмних моделей спеціалізованих процесорів, необхідних для реалізації в ПЛІС реконфігуровного середовища РККС під час її підготовки до виконання завдання, що викликає необхідність розробки цих моделей від початку.
3. Розробка програмних моделей спеціалізованих процесорів на рівні міжрегістрових передач вимагає багато часу і є достатньо трудомісткою, оскільки передбачає проектування архітектури спеціалізованих процесорів, моделювання мовою опису апаратних засобів, тестування та відлагодження.
4. Надзвичайно високі вимоги до кваліфікації операторів РККС, оскільки вони повинні вміти виконувати системний аналіз, розробляти архітектуру спеціалізованих процесорів, володіти технологіями та засобами реалізації спеціалізованих процесорів в ПЛІС і мовами програмування.

Для розв`язання цих питань необхідно створювати нові технології та засоби, які дозволять автоматизувати виконання обчислювальних завдань в РККС, а саме:

1. технології та засоби автоматичного розподілу обчислювальних завдань між універсальними процесорами та реконфігуровним середовищем РККС;
2. технології та засоби високорівневого проектування програмних моделей спеціалізованих процесорів, які не вимагатимуть від операторів кваліфікації схемотехніка, а дозволять створювати спеціалізований процесор, задавши алгоритм його роботи у вигляді графа чи програми на мові високого рівня.

Наявність таких засобів дозволить скоротити час підготовки РККС до виконання завдань та понизити вимоги до кваліфікації розробників, оскільки від них не вимагатиметься вміння виконувати системний аналіз, розробляти архітектуру і програмні моделі спеціалізованих процесорів.

Подальше підвищення ефективності РККС можливе за рахунок створення єдиного проектного

потоку та інтеграції цих засобів між собою і з засобами логічного синтезу програмних моделей спеціалізованих процесорів та конфігурування ПЛІС реконфігуровного середовища, що дасть можливість автоматизувати виконання завдань в РККС і зменшити участь розробника в цьому процесі.

Висновки

1. Запропоновано визначення основних понять РККС, що спрощує розуміння принципів функціонування РККС.

2. Показано базові типи архітектури РККС, а саме: архітектуру слабозв'язаної РККС, два типи архітектури тіснозв'язаної РККС, та два типи архітектури інтегрованої РККС.

3. Сформульовано спосіб виконання обчислень в РККС та показано, які саме дії виконує оператор РККС під час виконання в ній обчислювального завдання. Показано склад необхідного програмного забезпечення РККС для забезпечення її функціонування згідно з цим способом.

4. Визначено проблемні питання застосування РККС та окреслено шляхи їх вирішення.

Література

1. Scott Hauck, André DeHon. "Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation"/ Morgan Kaufmann, 2008. – 944 p.

2. Мельник А. Використання реконфігуровних прискорювачів для підвищення продуктивності персональних комп'ютерів / А.О. Мельник, В.А. Мельник, З.Т. Сарайрех // Науковий вісник Чернівецького ун-ту. Комп'ютерні системи та компоненти. – Чернівці : Чернівецький національний університет імені Юрія Федьковича, 2010. – Т. 1. Вип.1. – С. 20– 25.

3. Todman T., G. Constantinides, S. Wilton, O. Mencer, W. Luk and P. Cheung. Reconfigurable Computing: Architectures, Design Methods, and Applications // IEE Proceedings on Computers and Digital Techniques 152 (2) pp.193– 207 (2005).

4. Cadence Design Systems Inc, Palladium Datasheet, 2004.

5. Mentor Graphics, Vstation Pro: High Performance System Verification, 2003.

6. Annapolis Microsystems, Inc., Wildfire Reference Manual, 1998.

7. Goldstein S.C., H. Schmit, M. Budiu, S. Cadambi, M. Moe and R. Taylor. PipeRench: a reconfigurable architecture and compiler. IEEE Computer, Vol. 33, No. 4, 2000, pp. 70– 77.

8. Hauser J.R. and J. Wawrzynek. Garp: a MIPS processor with a reconfigurable processor. IEEE Symposium on Field-Programmable Custom Computing Machines, IEEE Computer Society Press, 1997.

9. Laufer R., R. Taylor and H. Schmit. PCI-PipeRench and the SwordAPI: a system for stream-based reconfigurable computing. Proc. Symp. On Field-Programmable Custom Computing Machines, IEEE Computer Society Press, 1999.

10. Rupp C.R., M. Landguth, T. Garverick, E. Gomersall, H. Holt, J. Arnold and M. Gokhale. The NAPA adaptive processing architecture. IEEE Symposium on Field-Programmable Custom Computing Machines, May 1998, pp. 28– 37.

11. Singh H., M-H Lee, G. Lu, F. Kurdahi, N. Bagherzadeh and E. Chaves, MorphoSys: An integrated reconfigurable system for data-parallel and compute intensive applications., IEEE Trans. on Computers, Vol. 49, No. 5, May 2000, pp. 465– 481.

12. Vuillemin J., P. Bertin, D. Roncin, M. Shand, H. Touati and P. Boucard. Programmable Active Memories: Reconfigurable Systems come of age. IEEE Transactions on VLSI Systems, Vol. 4, No. 1, March 1996, pp. 56– 69.

13. Wittig R.D. and P. Chow. OneChip: an FPGA processor with reconfigurable logic., IEEE Symposium on FPGAs for Custom Computing Machines, 1996.

14. Marshall A., T. Stansfield, I Kostarnov, J. Vuillemin and B. Hutchings. A reconfigurable arithmetic array for multimedia applications. ACM/SIGDA International Symposium on FPGAs, Feb 1999, pp. 135– 143.

15. Mirsky E. and A. DeHon. MATRIX: a reconfigurable computing architecture with configurable instruction distribution and deployable resources. Proc. Symp. on Field-Programmable Custom Computing Machines, IEEE Computer Society Press, 1996.

16. Razdan R. and M.D. Smith. A high performance microarchitecture with hardware programmable functional units. International Symposium on Microarchitecture, 1994, pp. 172– 180.

17. Taylor M. et al. The RAW microprocessor: a computational fabric for software circuits and general purpose programs. IEEE Micro, vol 22. No. 2, March/April 2002, pp. 25.35.

18. Altera Corp., Excalibur Device Overview, May 2002.

19. Xilinx, Inc., PowerPC 405 Processor Block Reference Guide, October 2003.

20. Altera Corp. Nios II Processor Reference Handbook, May 2004.

21. Fidjeland A., W. Luk and S. Muggleton. Scalable acceleration of inductive logic programs. Proc. Int. Conf. on Field-Programmable Technology, IEEE, 2002.

22. Leong P.H.W. and K.H. Leung. A microcoded Elliptic Curve Processor using FPGA technology. IEEE Transactions on Very Large Scale Integration Systems, Vol. 10, No. 5, 2002, pp. 550– 559.

23. Seng S.P., W. Luk and P.Y.K. Cheung. Flexible instruction processors. Proc. Int. Conf. on Compilers,

Arch. and Syn. for Embedded Systens, ACM Press, 2000.

24. Seng S.P., W. Luk and P.Y.K. Cheung. Run-time adaptive flexible instruction processors. Field-Programmable Logic and Applications, LNCS 2438, Springer, 2002.

25. Xilinx, Inc., Microblaze Processor Reference Guide, June 2004.

26. Baxter Robert Maxwell – a 64 FPGA Supercomputer / Robert Baxter, Stephen Booth, Mark Bull, Geoff Cawood, James Perry, Mark Parsons, Alan Simpson, Arthur S. Trew, Andrew McCormick, Graham Smart, Ronnie Smart, Allan Cantle, Richard Chamberlain, Gildas Genest // Engineering Letters, – 2008. – Volume 16, Number 3, September – P. 426– 433.

Надійшла 10.11.2012 р.
Рецензент: д.т.н. Максимович В.Н.

УДК 621.321

С.К. ПІДЧЕНКО, А.А. ТАРАНЧУК, О.О. ЛЕВИЦЬКИЙ
Хмельницький національний університет
О.В. КАЛЬВАТИНСЬКИЙ
Центр прийому і обробки спеціальної інформації та контролю навігаційного поля

ПІДВИЩЕННЯ ТЕМПЕРАТУРНОЇ СТАБІЛЬНОСТІ КІЛ СИНХРОНІЗАЦІЇ КВАДРАТУРНИХ ДЕМОДУЛЯТОРІВ OFDM СИСТЕМ

В роботі наведено аналіз спотворень сигналу в системах з ортогональним частотним поділом каналів (OFDM), викликаних помилками синхронізації. Запропоновано структуру квадратурного демодулятора сигналів з генератором опорних сигналів на базі термокомпенсованого цифрового синтезатора частоти.

Ключові слова: квадратурна модуляція, ортогональний частотний поділ каналів, цифровий синтезатор частоти, двочастотний кварцовий генератор.

In work the analysis of signals distortions is provided in systems with orthogonal frequency division multiplexing (OFDM), due to synchronization errors. The structure of the quadrature demodulator signals with a reference signal generator based on temperature-compensated digital frequency synthesizer is offered.

Keywords: Quadrature modulation, orthogonal frequency division multiplexing, digital frequency synthesizer, dual-frequency crystal oscillator.

Вступ

На теперішній час передача даних з ортогональним частотним поділом каналів (Orthogonal Frequency Division Multiplexing, OFDM) одержала широке поширення при побудові систем цифрового зв'язку. Вона є основою багатьох стандартів передачі цифрових даних, зокрема IEEE 802.11a, g, IEEE 802.16 (бездротові локальні мережі, Wi-Fi, наземне цифрове телевізійне мовлення DVB-T/T2) та інших. Основний принцип OFDM полягає в паралельній передачі даних на множині ортогональних несучих (піднесучих) коливань. Використання ортогональних піднесучих дозволяє уникнути міжканальної інтерференції та досягнути високої спектральної ефективності, не дивлячись на те, що спектри сигналів модульованих піднесучих мають достатньо значне перекриття.

При розподілі даних між частотними каналами, утвореними піднесучими, відбувається відповідне збільшення символьного інтервалу, а також знижується чутливість системи до міжсимвольної інтерференції за умови багатопроменевого поширення сигналу. Вузька смуга частот, яку займає сигнал на кожній з піднесучих, дозволяє домогтися більш високої якості передачі за наявності частотно-селективних завмирань в каналі зв'язку, але накладає більш жорсткі вимоги до точності синхронізації за частотою. Забезпечення високої точності синхронізації є однією з ключових проблем при прийомі OFDM-сигналів. Особливості структури систем OFDM-сигналів дозволяють застосовувати різні методи синхронізації, в тому числі недоступні для систем з однією несучою. Однак дана задача не має однозначного розв'язання, незважаючи на багато існуючих досліджень в даній області [1, 2].

1. Спотворення сигналу, викликані помилками синхронізації в системах OFDM

Ефективність прийому сигналу в системах зв'язку залежить від точності синхронізації приймача і передавача у часі, за частотою і фазою (у разі когерентного прийому). Відмінності в структурі OFDM-сигналу від сигналів з однією несучою обумовлюють підвищену чутливість до частотної неузгодженості і флуктуацій фази опорних генераторів передавача і приймача, що вимагає високої точності їх синхронізації.

Представимо сигнал на вході демодулятора з квадратурною (амплітудною) модуляцією (КАМ) у наступному вигляді:

$$s_{вх}(t) = s_{КАМ}(t) = A(t)\cos\left(\omega_0 t + \phi(t)\right) = A(t)\cos\phi(t)\cos\omega_0 t - \\ -A(t)\sin\phi(t)\sin\omega_0 t = I(t)\cos\omega_0 t + Q(t)\sin\omega_0 t, \quad (1)$$

де $I(t)$ та $Q(t)$ – синфазна та квадратурна складові модулюючої функції $s(t)$.