ACSN 2007. – Lviv, 2007. – p. 112 – 113. 3. Lenzerini M. Data Integration: A Theoretical Perspective. / Marco Lenzerini // Proc. of the ACM Symp. on Principles of Database. Systems (PODS), 2002. – pp. 233 – 246. 4. Tierney B. Contextual Semantic Integration for Ontologies / Brendan Tierney, Mike Jackson // www.macs.hw.ac.uk/ BNCOD21/DC/Tierney.pdf, 2005. 5. White C. Data Integration: Using ETL, EAI, and EII Tools to Create an Integrated Enterprise (Report Excerpt)./ C. White // http://www.tdwi.org/ Publications/ WhatWorks/ display.aspx?id=7979, 2007. 6. Берко А.Ю. , Висоцька В.А. Моделі та методи проектування інформаційних систем електронної комерції. / Андрій Берко, Вікторія Висоцька // Автоматизовані системи управління та прилади автоматики. Всеукраїнський міжвідомчий науково-технічний збірник № 138. – Харків, 2007. – с. 55 – 66. 7. Кузнецов С.Д. Ландшафт области управления данными: аналитический обзор / С.Д. Кузнецов, М.Н. Гринев // http: // www. citforum.ru/ database/data_management_ overview/, 2008. 8. Ландэ Д.В. Основы интеграции информационных потоков: Монография / Дмитрий Ландэ – К.: Инжиниринг, 2006. – 240 с. 9. Литовский К.Ю. Слабоструктурированные данные: некоторые методы их представления и обработки запросов / К.Ю. Литовский, Г.С. Томусяк // Московская Секция ACM SIGMOD – http:// synthesis.ipi.ac.ru/ sigmod/seminar/s20000224, 2000. – с.1 – 2. 10. Смальян Р. . Теория формальных систем / Роберт Смальян. – М. : Наука, 1981. 11. Спирли Э. Корпоративные хранилища данных. Планирование, разработка, развитие / Эрик Спирли. – М. : Издательский дом "Вильямс", 2001. – 400 с.

УДК 681.325.5-181.4

В. Глухов
Національний університет “Львівська політехніка”,
кафедра електронних обчислювальних машин

ОЦІНКА АПАРАТНИХ ВИТРАТ НА РЕАЛІЗАЦІЮ БАГАТОРІВНЕВОЇ КОМП’ЮТЕРНОЇ СИСТЕМИ

Сформульовано підхід до оцінювання апаратних витрат на реалізацію багаторівневої комп’ютерної системи.

In this article multilevel computer system complexity estimation is discussed.

Вступ. Постановка проблеми

Запропоновано метод оцінювання апаратних витрат на реалізацію багаторівневих комп’ютерних систем, який ґрунтується на використанні багаторівневої моделі взаємозв’язку відкритих систем. Також пропонується метод проектування багаторівневих систем, до якого метод оцінювання входить як складова частина.

Аналіз публікацій і окреслення проблеми

Однією з складових частин гарантоздатності є конфіденційність [1, 2]. Криптографічні методи забезпечення конфіденційності ґрунтуються на використанні шифропроцесорів (ШП). У роботах [3, 4] наведено багаторівневу структуру ШП, який здійснює криптографічні перетворення відповідно до стандартів [5–8]. Для побудови ШП використовується центральний протокольний процесор і криптографічний спецпроцесор (або декілька спецпроцесорів), при цьому спецпроцесори можуть бути виконані у вигляді ядер НВІС [9]. Загалом основою для побудови багаторівневих комп’ютерних систем є багаторівнева модель взаємозв’язку відкритих систем [11]. Найбільш доступним для проектування типом НВІС є програмовані логічні інтегральні схеми (ПЛІС), як відрізняються своїми технічними характеристиками (кількість і структура логічних боків, блоків введення-виведення,

комутаторів та ін. [10]). Тому актуальним є завдання вибору ПЛІС для реалізації гарантоздатної конфіденційної системи у вигляді багаторівневої комп'ютерної системи. Велике значення для проектування засобів захисту інформації має їхня порівняльна оцінка, так само, як і оцінювання складності алгоритмів, якими реалізують ці засоби. Теорія алгоритмів і теорія складності розвинута у роботах [12–18]. Водночас питання визначення складності багаторівневих ієрархічних систем вирішене недостатньо. Також недостатньо розглянуте питання практичної реалізації теоретичних результатів на сучасному етапі розвитку, коли треба давати відповіді на питання «Чи можна цю багаторівневу систему реалізувати на ПЛІС із відомими характеристиками?» або «Які характеристики повинна мати ПЛІС для реалізації заданої багаторівневої системи?»

Мета статті

Метою роботи є створення методу оцінювання апаратних витрат на реалізацію багаторівневих комп'ютерних систем, зокрема і гарантоздатних. Метод повинен враховувати ієрархічність багаторівневих систем і ґрунтуватися на оцінці апаратних витрат на реалізацію кожного з рівнів. Також метою статті є розроблення методу проектування багаторівневих систем, до якого метод оцінювання повинен входити як складова частина.

Еталонна модель взаємозв'язку відкритих систем

Найбільш структуровано принцип побудови багаторівневих систем викладено у [11], де наведена базова семирівнева еталонна модель взаємозв'язку відкритих систем (рис. 1, 2). Основи обміну даними між сумісними рівнями ілюструє рис. 2, де фігурують протокольний та сервісний блоки даних (звідси випливає необхідність мати у складі N-рівня протокольний та спеціалізований процесор). Рис. 1 ілюструє універсальний характер моделі, оскільки не фіксує назви рівнів, а вказує їхні умовні номери (…, N+1, N, N-1, …).

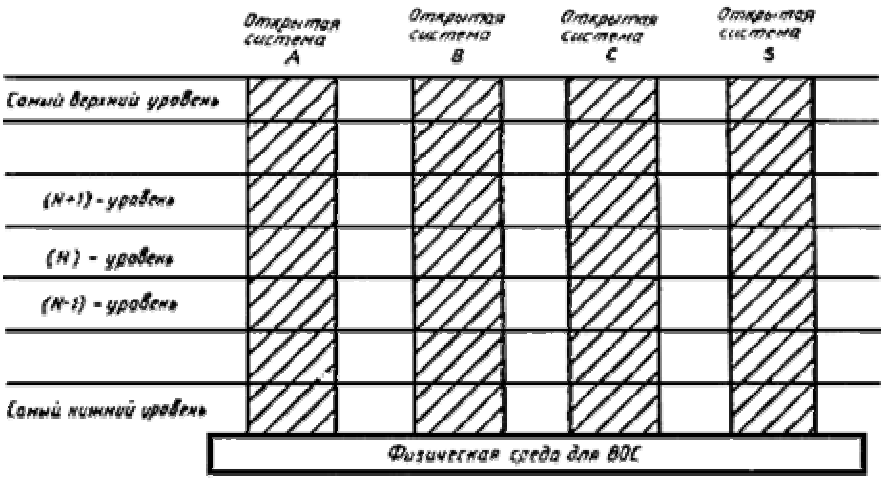

Рис. 1. Еталонна модель взаємозв'язку відкритих систем

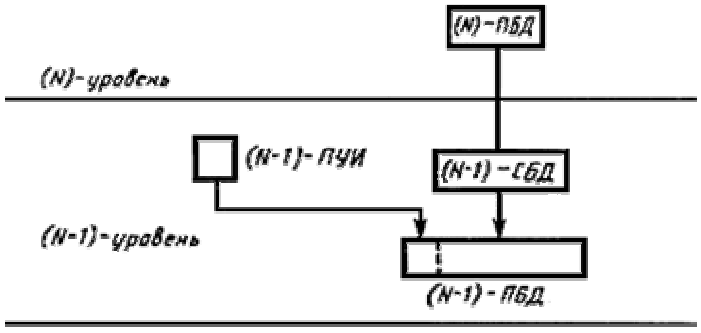

Рис. 2. Приклад перетворення блоків даних у суміжних рівнях

ПУИ – протокольна керуюча інформація, ПБД – протокольний блок даних, СБД – сервісний блок даних.

Принципи виділення семи рівнів еталонної моделі

Такі принципи [19] були використані для визначення семи рівнів еталонної моделі; вони будуть корисними і для розроблення багаторівневих систем. Не завжди можна довести, що вибрана рівнева організація є якнайкращим рішенням. Проте існують принципи, які можуть бути покладені в основу при вирішенні питання про те, де повинні проходити межі між рівнями і скільки має бути меж.

1. Не треба створювати багато рівнів – це ускладнить системотехніче завдання їх опису.

2. Проводити межу треба в тому місці, де опис послуг є найменшим і кількість операцій взаємодії через межу зведена до мінімуму.

3. Треба створювати окремі рівні для виконання таких функцій, які явно відрізняються за процесами, що реалізовують їх, або використовуваними технічними рішеннями.

4. Треба зосереджувати аналогічні функції в одному і тому рівні.

5. Проводити межі необхідно в тому місці, на яке вказує наявний досвід.

6. Слід формувати рівень функцій, що легко локалізуються. Це створює у разі потреби можливість повної перебудови рівня істотною зміною його протоколів для використання нових досягнень у галузі архітектури, апаратних і програмних засобів, не змінюючи при цьому послуги, що як отримуються від суміжних рівнів, так і надаються їм.

7. Проводити межу треба в тому місці, де в якийсь момент часу може виявитися корисним наявність відповідного стандартного інтерфейсу.

8. Треба створювати рівень тоді, коли відчувається необхідність в іншому рівні абстракції під час обробки даних, наприклад, в морфології, синтаксисі, семантиці.

9. Треба забезпечувати можливість такої зміни функцій або протоколів рівня, яке не має впливу на інші рівні.

10. Для кожного рівня треба створювати інтерфейси тільки з рівнями, які знаходяться безпосередньо вище та нижче. Схожі принципи застосовуються і до підрівнів.

11. Створювати нові підгрупи в межах вже існуючих груп функцій і утворювати з них підрівні у межах одного рівня треба у тому випадку, коли цього вимагають специфічні види послуг.

12. Де це необхідно, треба створювати два або більше підрівнів із спільною мінімальною функціональністю, щоб забезпечити інтерфейсні операції сусідніми рівнями.

13. Треба передбачати можливість обходу підрівнів.

У стандарті [11] не розглядаються переваги і недоліки внутрішніх інтерфейсів у відкритих системах. Зокрема, згадка або посилання на принцип 7 не повинні означати необхідність стандартів для таких внутрішніх інтерфейсів.

Важливо зазначити, що взаємозв’язок відкритих систем не вимагає стандартизації інтерфейсів усередині відкритих систем. Більше того, коли визначаються стандарти для таких інтерфейсів, відповідність цим стандартам не розглядається як ознака відкритості.

Структура багаторівневої гарантоздатної системи

Згідно з [20] гарантоздатність визначає міру здатності об’єкта бути працездатним і виконувати покладені на нього функції у будь-який час виконання покладеної на нього місії за умови, що на початку виконання місії об’єкт був придатний до виконання цих функцій. Система гарантоздатна, коли вона доступна, надійна, безпечна, захищена (здатна зберігати конфіденційність, забезпечувати цілісність), ремонтопридатна. Кожний об’єкт гарантоздатної (з погляду конфіденційності) системи можна представити у вигляді дворівневої (бортова ЕОМ і ШП) системи [4]. ШП є засобом протистояння діям ворожого оточення, на нього покладаються завдання криптографічного захисту та верифікації інформації.

Для побудови ШП використовується центральний процесор і криптографічний спецпроцесор (або декілька спецпроцесорів), при цьому спецпроцесори можуть бути виконані у вигляді ядер НВІС [9].

Відомі декілька методів взаємодії центрального і спеціалізованого процесорів. Центральний процесор може сприймати спецпроцесор як: 1) співпроцесор; 2) набір портів; 3) канал [21, 22].

Перший метод вимагає глибокого знання особливостей архітектури центрального процесора, його системи команд, використання її кодів для запуску співпроцесора. До того ж цей метод вимагає апаратної підтримки з боку центрального процесора. Другий і третій методи позбавлені цих недоліків.

Використання спецпроцесора як каналу підказує використання еталонної моделі взаємозв'язку відкритих систем [11] для розподілу функцій між елементами системи. Взаємодія між бортовою ЕОМ і шифропроцесором відбувається на протокольному (найвищому) рівні. Потік інформації під час її оброблення в шифропроцесорі спочатку йде з верхнього рівня на найнижчий, а потім знову підіймається на верхній (наприклад, під час шифрування відкрита інформація опускається з верхнього рівня на найнижчій, а потім зашифрована інформація підіймається з найнижчого рівня на найвищий (рис. 3).

Кожний N-рівень ШП є N-спецпроцесором, який складається з протокольного N-процесора і (N-1)-спецпроцесора. Усі спецпроцесори мають аналогічну структуру (рис. 4).

Стандарти, чинні в України, повністю визначають алгоритми роботи усіх спецпроцесорів. Міжнародні стандарти [23] є цінним і корисним доповненням до вітчизняних стандартів.

Як приклад розподілу завдань між різними рівнями наведений розподіл для ШП, що обробляє електронні підписи (табл. 1). Розподіл завдань між рівнями дає можливість визначити систему команд кожного із спецпроцесорів.

Протокольні процесори можуть бути реалізовані в складі ПЛІС [10, 26] або ззовні ПЛІС. Вони можуть бути RISC-процесорами та CISC-процесорами, 8-, 16-, 32-розрядними або можуть мати іншу розрядність [26] .

Процесори реалізуються як сукупність операційного і керуючого автоматів. На найнижчому рівні протокольні процесори можуть складатися тільки з керуючого автомату (цифрові автомати), а спеціалізовані – тільки з операційного.

Сучасні ПЛІС дають змогу реалізовувати універсальні процесори різного типу і продуктивності: hard-процесори (апаратно-реалізовані і розміщені на кристалі ПЛІС у процесі її виробництва нереконфігуровані процесори) і soft-процесори [26] (реконфігуровані, розроблені користувачем ядра [9]).

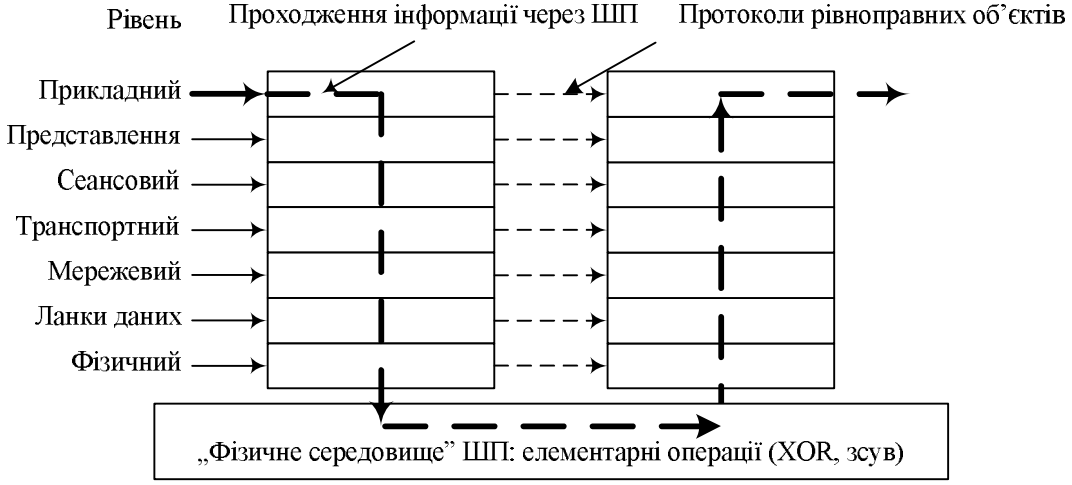

Рис. 3. Структура шифропроцесора

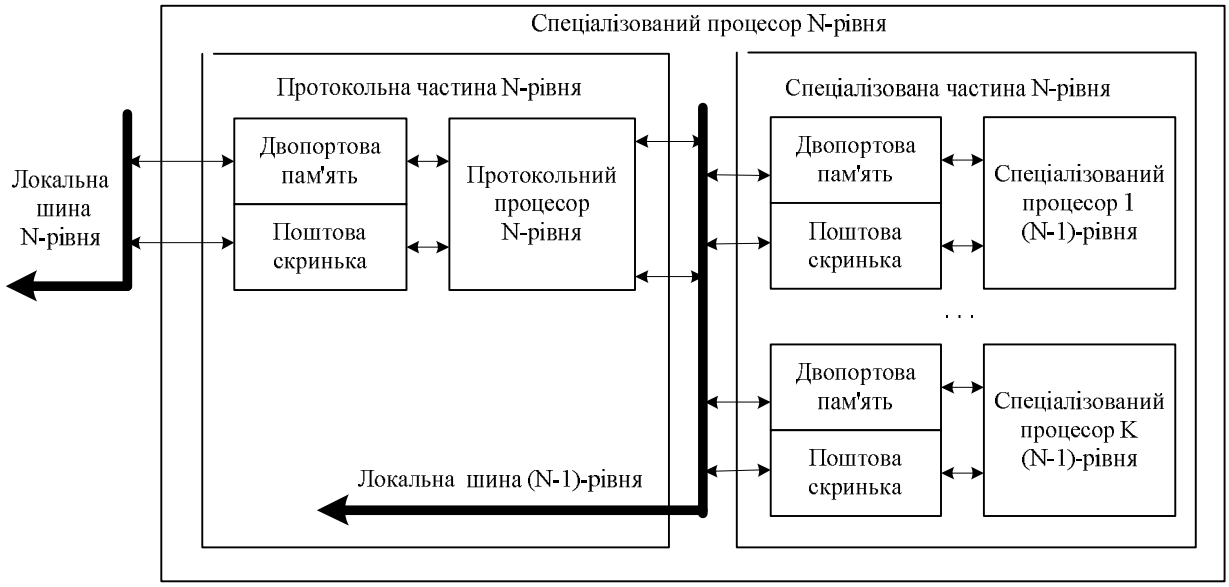

Рис. 4. Структура N-спецпроцесора (N-рівня)

Апаратні витрати на реалізацію протокольного процесора і спецпроцесора

Найважливішим показником у теорії складності, яка оперує поняттям SH-модель, є апаратна складність [18]. На практиці це веде до першочергової оцінки апаратних витрат на реалізацію проектованої системи.

Визначення відношення кількості мікросхем або інших логічних елементів (вентилів, конфігурованих логічних блоків, в SH-моделі – елементарних перетворювачів [18]) у складі протокольного і спеціалізованого процесорів дає змогу оцінити апаратні витрати на реалізацію першого, якщо відомі аналогічні витрати на реалізацію другого.

Приблизну пропорцію можна встановити, порівнюючи кількість транзисторів у відомих парах, які складаються з центральних процесорів Intel x86 [27] та математичних співпроцесорів Intel x87 [28] (табл. 2), а також з центральних процесорів [29] та відеопроцесорів [30] (табл. 3), що є «співучасниками» (одночасно використовуються в одному комп'ютері). Центральний процесор тут виступає у ролі протокольного, а математичний співпроцесор та відеопроцесор – у ролі спеціалізованого.

У парі «центральний процесор – математичний співопроцесор» відношення k кількостей транзисторів коливається у діапазоні k=0,6…3,0. У парі «центральний процесор – відеопроцесор» діапазон менший (k=0,7…1,3) і для процесорів з більшою кількістю транзисторів наближається до 1. Це значення може бути основою для оцінювання апаратних витрат на реалізацію процесорів. Тобто наближено можна прийняти, що апаратні витрати на реалізацію спецпроцесора дорівнюють апаратним витратам на реалізацію протокольного процесора (k=1).

Таблиця 1

Розподіл задач між різними рівнями шифропроцесора (рівні представлення – згідно з [23])

Рівень ШП	Рівень представлення	Тип операцій	Операції
1 (найнижчий)	Примітиви	Операції над елементами поля Галуа у нормальній формі	Піднесення до квадрата, визначення розрядів добутку
2	Примітиви	Операції над елементами поля Галуа у нормальній та поліноміальній формах	Множення, додавання, пересилання
3	Схеми	Операції над точками еліптичних кривих, криптографічні перетворення	Додавання, подвоєння, множення на число

Зрозуміло, що кількість комбінацій процесорів і відеопроцесорів може бути дуже велика.

Таблиця 3 показує співвідношення кількостей транзисторів для випадку, коли з більшим центральним процесором працює і більший відеопроцесор (під більшим тут мається на увазі «з більшою кількістю транзисторів»).

Якщо вважати, що значення k є приблизно однаковим для усіх рівнів, то можна оцінити апаратні витрати на реалізацію усієї багаторівневої системи. Якщо вваажати, що апаратні витрати на реалізацію спецпроцесора найнижчого (першого) рівня становлять одиницю, то апаратні витрати на реалізацію кожного з рівнів за наведених припущень наведені у табл. 4.

При кожному переході з верхнього рівня на нижчий апаратні витрати на реалізацію протокольного процесора доцільно зменшувати приблизно удвічі (в 1+k разів). Зменшення апаратних витрат призводить або до зменшення розрядності (32, 16, 8), або до зменшення набору функціональних вузлів, або до скорочення системи команд (перехід від процесора CISC до RISC). Таблиця 4 як ілюстрація цього підходу показує можливі типи протокольних процесорів на різних рівнях системи.

Апаратні витрати на протокольний процесор і декілька спеціалізованих процесорів

Протокольний процесор N-рівня може одночасно керувати роботою декількох (K) спецпроцесорів (N-1)-рівня. Логічно така структура утворює зірку, а з'єднання протокольного процесора зі спеціалізованими здійснюється за допомогою локальної шини (магістралі). Так утворюється зірково-магістральна структура, переваги якої доведені у [24, 25].

Спецпроцесори можуть працювати: послідовно у часі, паралельно у часі, m груп з n спецпроцесорів працюють паралельно у часі (всередині групи спецпроцесори працюють послідовно), m груп з n спецпроцесорів працюють послідовно у часі (всередині групи спецпроцесори працюють паралельно). Якщо прийняти, що апаратні витрати на реалізацію кожного (N-1)-спецпроцесора однакові і прийняти їх за 1, то апаратні витрати на реалізацію протокольного N-процесора показано в табл. 5.

Загалом загальні апаратні витати A_N на реалізацію N-рівня системи з однаковими для усіх її рівнів коефіцієнтами k, m та n дорівнюватимуть $A_N=(k+n)^N m^N=A_1^N$, де $A_1=(k+n)m$. Для усієї N-рівневої системи апаратні витрати обчислюються як сума членів геометричної прогресії і дорівнюють $A=A_1(A_1^N-1)/(A_1-1)$.

Метод проектування багаторівневих гарантоздатних комп'ютерних систем (у частині забезпечення конфіденційності)

Узагальнений метод проектування багаторівневих гарантоздатних комп'ютерних систем (у частині забезпечення конфіденційності) складається з послідовності проектних рішень:

система представляється як багаторівнева структура відповідно до еталонної моделі взаємозв'язку відкритих систем;

визначається кількість рівнів N;

кожний N-рівень шифропроцесора є N-спецпроцесором, який складається з протокольного N-процесора і (N-1)-спецпроцесора;

кожний протокольний N-процесор реалізується як універсальний процесор;

продуктивність та інтерфейси універсального процесора визначаються особливістю системи;

кожний із спецпроцесорів реалізує одне із завдань (або декілька завдань, або частину завдань) забезпечення конфіденційності;

кожний із спецпроцесорів працює відповідно до обраного стандарту;

система команд спецпроцесора визначається алгоритмом вирішення відповідного завдання;

кількість спецпроцесорів визначається потоком даних та заданим часом їхнього опрацювання;

процесори реалізуються у вигляді ядер НВІС (практично – ПЛІС), утворюючи так звану «систему на кристалі»;

визначаються апаратні витрати на реалізацію спецпроцесора рівня 1 та усього рівня 1 (A_1). Апаратні витрати А на реалізацію усієї N-рівневої системи будуть орієнтовно дорівнювати $A=A_1(A_1^N-1)/(A_1-1)$, що дозволяє обрати тип ПЛІС.

Таблиця 2

Кількість транзисторів у центральних процесорах та математичних співпроцесорах

Центральний процесор	8086	80286	Intel386™ SX	Intel386™ DX
$N_ц$ - кількість транзисторів у центральному процесорі, тис.	29	134	275	275
Співпроцесор	8087	287	387SX	387DX
N_c - кількість транзисторів у співпроцесорі, тис.	45	45	120	120
$k=N_ц/N_c$	0,6	3,0	2,3	2,3

Таблиця 3

Кількість транзисторів у центральних процесорах та відеопроцесорах

Ядро центрального процесора	Intel Pentium 4 Willamette	Intel Pentium 4 Northwood	Intel Pentium 4 Northwood	Intel Pentium 4 Prescott	AMD Athlon 64
$N_ц$ - кількість транзисторів центрального процесора, млн.	42	55	178	125	106
Відеопроцесор	nVidia GeForce FX 5200	nVidia GeForce FX 5600	nVidia GeForce FX 5900	nVidia GeForce FX 5900	ATI Radeon 9800
$N_в$ - кількість транзисторів відеопроцесора, млн.	45	75	135	135	107
$k=N_ц/N_в$	0,9	0,7	1,3	0,9	1,0

Таблиця 4

Апаратні витрати

Рівень	Апаратні витрати на реалізацію спецпроцесора	Апаратні витрати на реалізацію протокольного процесора	Апаратні витрати на реалізацію рівня	Тип протокольного процесора
1	1	k	1+k	Цифровий керуючий автомат
2	1+k	(1+k)k	$(1+k)^2$	RISC (8 розрядів)
3	$(1+k)^2$	$(1+k)^2k$	$(1+k)^3$	RISC (32 розряди)
4	$(1+k)^3$	$(1+k)^3k$	$(1+k)^4$	CISC (32 розряди)
5	$(1+k)^4$	$(1+k)^4k$	$(1+k)^5$	
6	$(1+k)^5$	$(1+k)^5k$	$(1+k)^6$	Комп'ютер на модулі
7	$(1+k)^6$	$(1+k)^6k$	$(1+k)^7$	Бортова ЕОМ

Таблиця 5

Апаратні витрати при декількох спецпроцесорах

Варіант	Принцип роботи К спецпроцесорів (K=m*n)	Кількість m спецпроцесорів, що працюють паралельно	Кількість n спецпроцесорів, що працюють послідовно	Апаратні витрати на реалізацю К спецпроцесорів (K = m*n)	Апаратні витрати на реалізацію протокольного процесора	Апаратні витрати на реалізацію рівня
1	послідовно у часі	m=1	n	m*n=n	m*k=k	(k+n)m=k+n
2	паралельно у часі	m	n=1	m*n=m	m*k	(k+n)m=(k+1)m
3	m груп паралельно, n спецпроцесорів послідовно	m	n	m*n	m*k	(k+n)m
4	n груп послідовно, m спецпроцесорів паралельно	m	n	m*n	m*k	(k+n)m

Висновки

У роботі запропонований і обґрунтований метод оцінювання апаратних витрат на реалізацію багаторівневих комп'ютерних систем, який ґрунтується на використанні багаторівневої моделі взаємозв'язку відкритих систем. Також аналізуються апаратні витрати на реалізацію кожного рівня. Цей підхід дає змогу оцінити можливість реалізації багаторівневої системи на ПЛІС конкретного типу на ранніх стадіях проектування, створити модульну ієрархічну структуру, до якої застосовуються методи паралельного і одночасного проектування, виготовлення, налагодження і тестування. Також пропонується методу проектування багаторівневих систем, до якого метод оцінювання входить як складова частина.

1. IEC 50(191):1990 International Electrotechnical Vocabulary. Chapter 191: Dependability and quality of service. 135 p. 2. IEC 60050-191-am2 (2002) Ed. 1.0 International Electrotechnical Vocabulary. Chapter 191: Dependability and quality of service. 3. Глухов В., Заіченко Н., Оліярник Б. Шифропроцесор для бортових інформаційно-керуючих систем // Наукові нотатки: Міжвузівський збірник (за напрямком «Інженерна механіка»). – Вип. 19. – Луцький державний технічний університет. – Луцьк, 2007. – С.33–43. 4. Глухов В.С., Євтушенко К.С., Заіченко Н.В., Оліярник Б.О. Криптографічні засоби спеціалізованої бортової ЕОМ для бронетехніки // Вісник Хмельницького національного університету, 2007. – №2. – С.29–33. 5. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевіряння. – К.: Державний комітет України з питань технічного регулювання та споживчої політики, 2003. 6. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. Государственный комитет СССР по стандартам. – М., 1989. 7. Межгосударственный стандарт ГОСТ 34.311-95. Информационная технология. Криптографическая защита информации. Функция хеширования. Межгосударственный совет по стандартизации, метрологии и сертификации. – Минск: Госстандарт Украины, с дополнениями, 1997. 8. Межгосударственный стандарт ГОСТ 34.310-95. Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма. Межгосударственный совет по стандартизации, метрологии и сертификации. Минск. Госстандарт Украины, с дополнениями, 1997. 9. Мельник А.О., Коркішко Т.А. Система підтримки виконання алгоритмів криптографічного захисту інформації на основі програмованого процесора та криптографічних акселераторів // Вісник Держ. ун-ту "Львівська політехніка". 2000. – № 385. – С. 77 – 80. 10. Грушвицкий Р.И., Мураев А.Х., Угрюмов Е.П. Проектирование систем на микросхемах программируемой логики. – СПб.: БХВ-Петербург, 2002. – 608 с.: ил. 11. ДСТУ ISO/IEC 7498-1:2004. Інформаційні технології. Взаємозв'язок відкритих систем. Базова еталонна модель. Ч. 1. Базова модель (ISO/IEC 7498-1:1994, IDT). 12. Черкаський М.В. Моделі неформальних алгоритмів // Вісник Нац. ун-ту "Львівська політехніка", 2000. – № 385. – С.131 – 133. 13. Черкаський М.В. SH-модель алгоритму // Вісник Нац. ун-ту "Львівська політехніка". 2001. – № 433. – С.127 – 134. 14. Мельник А.О. Черкаський М.В. Теорія алгоритмів і методи обчислень: новий курс // Вісник Нац. ун-ту "Львівська політехніка" 2001. – № 437. – С.99 – 105. 15. Черкаський М.В., Мітюков В.С. Історичний аспект складності алгоритму // Вісник Нац. ун-ту "Львівська політехніка" 2002. – № 463. – С.111–118. 16. Черкаський М.В. Еволюція тлумачення поняття "алгоритм" // Вісник Нац. ун-ту "Львівська політехніка" 2003. – № 492. – С.142 – 146. 17. Черкаський М.В., Саїд Садек Абдалла. Псевдо SH-модель // Вісник Нац. ун-ту "Львівська політехніка" 2004. – № 523. – С.145 – 150. 18. Черкаський М.В., Хусейн Халід Мурад. Універсальна SH-модель // Вісник Нац. ун-ту "Львівська політехніка" 2004. – № 523. – С.150 – 154. 19. ГОСТ 28906-91 (ИСО 7498-84, ИСО 7498-84 Доп.1-84). Системы обработки информации. Взаимосвязь открытых систем. Базовая эталонная модель. 20. Military handbook MIL-HDBK-338b. Electronic reliability design handbook. Department of defense of USA. 1 october 1998. 21. Аронов В.Б., Глухов В.С., Деревенко Я.К., Заіченко Н.В., Федуняк С.Ф. Одноплатный арифметический процессор, подключаемый к магистрали ГОСТ 26765.51-86, и средства обеспечения его серийного производства // I Научно-техн. конф. НПО "Фазотрон": Тезисы докладов. Москва, 19-21 сентября 1989 г. 22. Глухов В.С., Заіченко Н.В. Арифметический спецвычислитель с кэш-памятью команд // "Тезисы докладов 29-й научно-техн.

конф. НПО "Антей". – М., 1990. 23. IEEE Std 1363-2000 IEEE Standard Specifications for Public-Key Cryptography Sponsor Microprocessor and Microcomputer Standards Committee of the IEEE Computer Society. Approved 30 January 2000. 24. Николайчук Я.Н. Низовые вычислительные сети: Учебн. пособие. – К.: УМК БО, 1990. – 55 с. 25. Николайчук Я.М., Круцкевич Н.Д. Перспективи використання зіркової-магістральної архітектури з пам'яттю колективного доступу комп'ютерних мережах з глибоким розпаралелюванням // Вимірювальна та обчислювальна техніка в технологічних процесах: Збірник наукових праць – Хмельницький: ТУ/7 -2002. – №9. – Т. 2. – С.122–126. 26. Березко Л.О., Троценко В.В. Мультипроцесор на ПЛІС // Вісник Нац. ун-ту "Львівська політехніка" "Комп`ютерні системи та мережі". № 573. Львів, 2006. С.10-14. 27. http://www.intel.com/pressroom/kits/quickreffam.htm#i486 ©Copyright Intel Corporation. All rights reserved. Intel Corporation, 2200 Mission College Blvd., Santa Clara, CA 95052-8119, USA. Intel.com Terms of Use version 01042008. 28. http: // www. informit. com /articles/ article.aspx?p=130978&seqNum=20&rll=1 © 2008 Pearson Education, Informit. All rights reserved. 800 East 96th Street, Indianapolis, Indiana 46240. 29. Головных А., Литвинов Е. Шаг в поднебесье // Chip (Украина). 2004. – №4. – С. 36–37. 30. Взрывная волна 3D // Chip (Украина). 2004. – №1. – С.42–43.

УДК 536.5

І. Микитин
Національний університет "Львівська політехніка",
кафедра інформаційно-вимірювальних технологій

ПРОГРАМНА МОДЕЛЬ МЕТОДУ УСЕРЕДНЕННЯ ШУМОВИХ СИГНАЛІВ

Розглянуто програмну модель розрахунку середнього значення квадрата шумової напруги методом усереднення частинами.

It was considered the programmatic model of calculation of middle value of square of noise voltage by the method of integration with the parts.

Постановка проблеми

Проведені теоретичні дослідження [1] показали, що для досягнення прийнятних значень методичної похибки шумових термометрів (ШТ) час вимірювання повинен бути достатньо тривалим. Так, для отримання значення відносного середньоквадратичного відхилення методичної похибки 0.05% для частотної смуги 100кГц, опору чутливого елементу первинного перетворювача (ПП) 100 Ом та термодинамічної температури 293К потрібно проводити усереднення протягом 250 с. Час усереднення можна зменшити збільшенням опору ПП (збільшуючи тим самим рівень вимірюваного сигналу) та розширенням робочої частотної смуги ШТ. Проте така зміна параметрів ШТ приводить до підвищення вимог до аналогової та цифрової частини ШТ і, відповідно, до істотного подорожчання ШТ.

Враховуючи вищесказане, запропоновано проводити усереднення за короткі проміжки часу, а потім усереднювати отримані результати вимірювання. Наприклад, проводимо вимірювання протягом 1с. Накопичивши 250 результатів вимірювання, усереднюємо їх та отримуємо результат вимірювання термодинамічної температури з необхідною методичною похибкою.

Для перевірки цього алгоритму усереднення побудовано програмну модель, яка дає змогу проводити дослідження роботи ШТ за даним алгоритмом усереднення, вибирати оптимальний час