УДК 681.3

Т.А. Коркішко*, А.О. Мельник
Національний університет "Львівська політехніка",
кафедра ЕОМ
*Тернопільська академія народного господарства

ВИМОГИ ДО ПРОДУКТИВНОСТІ ПРОЦЕСІВ ШИФРУВАННЯ СИМЕТРИЧНИМИ БЛОКОВИМИ АЛГОРИТМАМИ

Проаналізовано особливості структурної організації симетричних блокових алгоритмів шифрування (СБАШ). Здійснено оцінку складності виконання симетричних блокових алгоритмів шифрування у типових областях застосування. Сформульовані вимоги та запропоновано вираз для оцінки продуктивності багатоканального процесора СБАШ. Оцінено складність широко вживаних алгоритмів СБАШ та визначено області доцільного використання одноканальних процесорів СБАШ. Запропоновано створювати багатоканальні процесори СБАШ, вільних від недоліків, притаманних одноканальним процесорам.

The analysis of the particularities of the symmetric block encryption algorithm (SBEA) structures is considered. The estimation of the complexity of these algorithms is provided for the typical usage areas. The requirements are formulated and an expression is proposed for the performance estimation of the multichannel SBEA processor. The complexity of the wide-used SBEA is estimated as well as the appropriate usage fields for the single-channel processors are defined. The creation of the multichannel SBEA processors is proposed, such the processors have no disadvantages that have single-channel processors.

I. Вступ

Процесори шифрування даних симетричними блоковими алгоритмами шифрування (СБАШ) широко використовуються для забезпечення конфіденційності даних у системах передачі, обробки та зберігання [1]. До найчастіше вживаних СБАШ належать алгоритми, що передбачають обробку блоків даних розміром 64 – 256 біт із використанням ключа шифрування розміром 56 – 256 біт [2]. У більшості випадків обробка блоків даних повинна проводитися в реальному масштабі часу, тобто у міру їх надходження [3]. При цьому частота надходження вхідних даних та видачі результатів може лежати у широких межах – від десятків кГц до сотень МГц. Отже, процесори шифрування СБАШ повинні забезпечувати неперервний прийом, обробку даних в темпі їх надходження та видачу результуючих масивів даних. У даній роботі оцінюються вимоги до продуктивності процесорів шифрування СБАШ для багатоканальної обробки даних у реальному масштабі часу.

II. Структурна організація СБАШ

Аналіз СБАШ [2] та базових способів їх побудови показав, що їх структурну організацію на високому рівні можна подати у вигляді поєднання двох основних компонент: процедури обробки даних та процедури обчислення розпису ключа, а порядок

обробки блоку даних та ключа залежить від структури алгоритму та від типу виконуваної операції – зашифрування чи розшифрування. Подальший аналіз цих компонент дозволив подати СБАШ у вигляді сукупності чотирьох процедур: обробки даних для операції зашифрування, обробки даних для операції розшифрування, обчислення розпису ключа для зашифрування, обчислення розпису ключа для розшифрування, що, в свою чергу, складаються з дрібніших елементів: для процедури обробки даних це етапи початкової модифікації даних, основної обробки даних, кінцевої модифікації даних; для процедури обчислення розпису ключа – етапи початкової модифікації ключа, основної обробки ключа, кінцевої модифікації підключів.

Аналіз особливостей структурної організації процедур формування наборів підключів СБАШ дозволив зробити висновок про можливість їх поділу на пряму, ітеративну та комбіновану процедури.

Наведемо коротко характеристики кожного варіанта виконання процедури обчислення розпису ключа.

Пряма процедура формування набору підключів передбачає використання лише елементів ключа та додаткової інформації, наприклад спеціально вибраних констант, які залежать від номера генерованого набору підключів.

Ітеративна процедура передбачає використання ключа шифрування лише один раз для формування першого проміжного набору підключів (який, однак, також може використовуватися у процедурі обробки даних СБАШ) та подальшого ітераційного обчислення наступних наборів підключів, де проміжний набір використовується як початковий вхідний ключ. При обчисленнях може використовуватись додаткова інформація, аналогічно до прямої процедури.

Комбінована процедура передбачає комбінування підходів прямої та ітераційної процедур: біжучий набір підключів обраховується із використанням як ключа шифрування, так і попереднього набору підключів (разом із додатковою інформацією).

Необхідно зазначити, що функції етапів обробки для обчислення наборів підключів можуть різнитися між собою не тільки для операцій розшифрування та зашифрування, а й для різних наборів підключів.

Процес обробки даних СБАШ подамо як послідовність таких етапів: (i) налаштування структури та параметрів процедури обробки даних згідно з операцією шифрування та ключа шифрування; (ii) обчислення модифікованого блоку даних з використанням набору підключів для етапу початкової модифікації даних; (iii) обчислення останнього проміжного блоку даних з використанням наборів підключів для етапу основної обробки даних; (iv) обчислення вихідного блоку даних з використанням набору підключів для етапу кінцевої модифікації даних. Етап (i) виконується лише один раз при зміні ключа шифрування чи типу операції шифрування. Наступні етапи виконуються кожного разу при отриманні на вхід процедури обробки даних нового блоку даних.

Аналогічно до процедури обчислення розпису ключа, функції етапів обробки даних можуть різнитися між собою не тільки для операцій розшифрування та зашифрування, а й для різних номерів раундів. Тому додатковим параметром кожної функції є номер раунду, для якого вона використовується.

Оскільки до узагальненої структури СБАШ входять дві структурно самостійні та функціонально пов'язані процедури – процедура обробки даних та процедура обробки ключа, то виділимо два способи їх виконання відносно одна однієї: одноразове та паралельне.

У першому випадку при обробці блоку даних обчислені набори підключів зберігаються у деякій пам'яті та використовуються в міру необхідності. Процес обробки наступних блоків даних передбачає використання вже готових наборів підключів. При зміні операції чи ключа шифрування ініціюється процедура обчислення розпису ключа та проводиться формування та збереження усіх наборів підключів із нового ключа. Якщо процедура обробки даних використовує додаткові модифікації блоку даних перед та після виконання етапу основної обробки даних, то необхідні для цього дані також обчислюються при виконанні процедури обчислення розпису ключа.

У другому випадку необхідний набір підключів формується паралельно із виконанням процедури обробки даних. Певний обчислений набір підключів використовується на відповідному етапі виконання процедури обробки даних. Зміна ключа чи типу операції шифрування приводить до необхідності ініціації налаштування структур та параметрів процедур обчислення розпису ключа та обробки даних.

Вибір конкретного способу залежить від структури процедури обчислення розпису ключа, структури процедури обробки даних, організації обчислень з обробки даних та області застосування процесора шифрування СБАШ.

III. Складність СБАШ

Складність виконання СБАШ визначається як кількість операцій, необхідних для обробки одного блоку вхідних даних при заданому ключі та типі операції шифрування. Такий підхід узгоджується з визначенням часової складності, запропонованим у [4]: часова складність визначається кількістю елементів схеми, розташованих вздовж критичного шляху розповсюдження сигналу. Для випадку оцінки складності СБАШ під часовою складністю будемо розуміти кількість операцій, необхідних для зашифрування одного блоку даних, а одиницею складності СБАШ буде елементарна операція із перетворення даних СБАШ [5]. Вираз для оцінки складності процедури обчислення розпису ключа має вигляд:

$$K_{KEY} = K_{KEY}^{INIT} + K_{KEY}^{F} + \sum_{i=1}^{Nr} \sum_{j=1}^{N_{Sk}^{i}} K_{RK}^{ij} + K_{KEY}^{L}, \qquad (1)$$

де K_{KEY}^{INIT} – складність налаштування процедури обчислення розпису ключа при зміні ключа чи типу операції шифрування, K_{KEY}^{F}, K_{KEY}^{L} – складність обчислення наборів підключів для початкової та кінцевої модифікації даних відповідно, K_{RK}^{ij} – складність обчислення і-го підключа і-го набору, $i = 1,\ldots, Nr$, $j = 1,\ldots, N_{Sk}^{i}$, N_{Sk}^{i} – кількість підключів у і-му наборі. Для оцінки складності процедури обробки даних застосовується вираз:

$$K_D = K_D^{INIT} + K_D^{F} + \sum_{i=1}^{Nr} K_D^{i} + K_D^{L}, \qquad (2)$$

де K_D^{INIT} – складність налаштування процедури обробки даних при зміні ключа чи типу операції шифрування, K_D^F, K_D^L – складність виконання етапів початкової та кінцевої модифікації даних відповідно, K_D^i – складність і-го раунду основного етапу обробки даних, $i = 1,\ldots, Nr$.

Формули (1) та (2) відображають відповідно складність обчислення розпису ключів та складність обробки даних при початковому завданні ключа та виконанні операції шифрування над одним блоком даних. Тому складність СБАШ K_A при обробці одного блоку та операції шифрування m дорівнює сумі складностей виконання процедури обчислення розпису ключа та процедури обробки даних:

$$K_A^m = K_{KEY}^m + K_D^m, \tag{3}$$

де K_{KEY}^m – складність виконання процедури обчислення розпису ключа K_{KEY} для операції шифрування m, K_D^m – складність виконання процедури обробки даних K_D для операції шифрування m, $m = \{e, d\}$, e – операція зашифрування, d – операція розшифрування. При розгляді структур складових процедур було зазначено, що функції обробки даних та ключа можуть різнитися при виконанні різних типів операції шифрування, тому складність СБАШ буде різною при виконанні цих операцій, тобто $K_A^d \neq K_A^e$.

Вираз (3) оцінює складність виконання СБАШ при обробці одного блоку даних із використанням одного ключа шифрування. Вирази для оцінки складності виконання СБАШ у типових застосуваннях з обробки даних із різними розмірами масивів даних та кількістю ключів, що використовуються для їх обробки та різних способів виконання складових процедур СБАШ наведені у табл. 1.

Таблиця 1

Спосіб виконання складових СБАШ*)	Розмір масиву ключів	Розмір масиву даних	Вираз для оцінки складності СБАШ (операцій на один блок)
ООРК	1	N_D	$K_A = K_{KEY} + K_D^{INIT} + N_D(K_D - K_D^{INIT})$ (4)
ООРК	K_L	N_L	$K_A = K_L(K_{KEY} + K_D^{INIT}) + N_L(K_D - K_D^{INIT})$ (5)
ПОРК	1	N_D	$K_A = K_{KEY}^{INIT} + K_D^{INIT} + N_D(K_{KEY} + K_D - K_{KEY}^{INIT} - K_D^{INIT})$ (6)
ПОРК	K_L	N_L	$K_A = K_L(K_{KEY}^{INIT} + K_D^{INIT}) + N_L(K_{KEY} + K_D - K_{KEY}^{INIT} - K_D^{INIT})$ (7)

*) ООРК – одноразове обчислення розпису ключа, ПОРК – паралельне обчислення розпису ключа

IV. Оцінка необхідної продуктивності процесорів шифрування СБАШ

Для оцінки необхідної продуктивності процесорів шифрування СБАШ скористаємося методикою, запропонованою в [6], згідно з якою вимоги до продуктивності P оцінюються із врахуванням кількості каналів надходження даних K, складності алгоритму R, частоти надходження даних f та розміру масиву даних, що обробляється N:

$$P = \frac{R \cdot K \cdot f}{N}, \tag{8}$$

де $R = R(N)$, а P вимірюється в операціях за секунду.

Розглянемо застосування виразу (8) для випадку процесорів шифрування СБАШ. При цьому зауважимо, що СБАШ передбачає обробку блоків даних фіксованого розміру n із використанням ключа шифрування розміром k. Наведені вирази для оцінки складності СБАШ враховують обробку ключа. Тому розмір масиву даних, який обробляється, становить один блок, або n бітів. Прийнявши, що кількість каналів даних становить H, максимальна частота надходження вхідних даних каналів f_D, складність СБАШ при виконанні операції m шифрування K_A^m, отримаємо вираз для оцінки продуктивності процесорів шифрування СБАШ у блоках за секунду:

$$P^m = \frac{K_A^m \cdot H \cdot f_D}{n}, \tag{9}$$

де P^m – продуктивність обробки пари блок-ключ СБАШ при виконанні операції m.

Оскільки, з одного боку, процесори шифрування можуть бути орієнтовані на виконання операцій як зашифрування, так і розшифрування, а з іншого – складність СБАШ може бути різною при різних типах операцій шифрування, тобто $K_A^d \neq K_A^e$, то для забезпечення необхідної продуктивності обробки даних при різних типах операцій шифрування для виразу (9) необхідно обирати найбільше значення складності СБАШ. Тобто, вираз (9) можна записати як:

$$P^m = \frac{MAX(K_A^e, K_A^d) \cdot H \cdot f_D}{n}, \tag{10}$$

де функція $MAX(K_A^e, K_A^d)$ набуває найбільшого із значень своїх аргументів.

Вираз (10) визначає вимогу до продуктивності процесорів шифрування для обробки однієї пари блок-ключ. Якщо ж необхідно оцінити продуктивність процесорів шифрування при обробці масиву блоків із використанням одного ключа, то при одноразовому виконанні процедури обчислення розпису ключа складність СБАШ K_A^m необхідно обчислювати згідно з виразом (4), а при використанні паралельного виконання складових процедур – згідно з виразом (5). Аналогічно при оцінці продуктивності процесорів шифрування для обробки масиву даних із використанням масиву ключів для одноразового виконання процедури обчислення розпису ключа необхідно користуватися виразом (6), а при паралельному – виразом (7).

Розглянемо приклад оцінки вимог до продуктивності процесора шифрування на прикладі СБАШ ГОСТ28147-89 [7] при обробці масиву даних розміром N_D із використанням одного ключа. Для цього спочатку проаналізуємо складність виконання алгоритму.

СБАШ ГОСТ 28147-89 призначений для обробки блоків даних розміром 64 бітів із використанням 256-бітового ключа шифрування. Процедура обчислення розпису ключа полягає у виборі 32-бітових підключів для кожного раунду процедури обробки даних із 32 раундів.

Розглянемо складові елементи формул (1) та (2) обчислення складності алгоритму при зашифруванні даних. Згідно із введеною класифікацією процедур обчислення розпису ключа, процедура обчислення розпису ключа алгоритму ГОСТ28147-89 належить до класу прямих процедур. Приймемо, що зміна ключа шифрування не супроводжується зміною таблиці замін.

Складність налаштування процедури обчислення розпису ключа при зміні ключа чи типу операції шифрування $K_{KEY}^{INIT} = 0$. Складність обчислення наборів підключів для початкової модифікації даних та кінцевої модифікації становить відповідно $K_{KEY}^{F} = 0$ та $K_{KEY}^{L} = 0$, оскільки ці набори не використовуються. Оскільки процедура формування наборів підключів не передбачає виконання обчислювальних операцій над ключем шифрування, то складність обчислення і-го підключа j-го набору K_{RK}^{ij} виразимо як кількість операцій формування адреси для вибірки з пам'яті ключа шифрування підключа, тому $K_{RK}^{ij} = 1$. Кількість раундів $Nr = 32$, кількість підключів у кожному наборі є однаковою та становить $N_{Sk}^{i} = 1$. Для процедури обробки даних отримаємо такі значення складових формули (2): складність виконання етапів початкової та кінцевої модифікації даних K_D^F, K_D^L становлять відповідно $K_D^F = 0$, $K_D^L = 0$, оскільки ці етапи відсутні; складність і-го раунду основного етапу обробки даних K_D^i є однаковою для усіх раундів. Алгоритм для обробки даних одним раундом використовує такий перелік операцій: одну операцію додавання за модулем 2^{32}, одну операцію побітового додавання за модулем 2 над 32-розрядними блоками, вісім операцій замін за таблицею чотирьох бітових векторів на чотирибітові значення, одну операцію циклічного зсуву 32-бітового блоку на 11 розрядів вліво, одну операцію обміну частин блоку даних. Отже, загальна кількість операцій для одного раунду становитиме $K_D^i = 12$.

Обчислення складності складових процедур алгоритму за формулами (1) та (2) дають такі результати: $K_{KEY} = 32$, $K_D = 384$. Оскільки необхідно оцінити складність виконання алгоритму з обробки масиву даних із використанням ключа, то скористаємося виразом (4). Підставивши в (4) отримані значення K_{KEY} (вираз (1)) та K_D (вираз (2)), отримаємо такий вираз для оцінки складності алгоритму: $K_A = 384N_D + 32$ операцій.

Результати оцінки складності широко вживаних СБАШ [1] за запропонованою вище методикою наведені у табл. 2.

Таблиця 2

Алгоритм	Розмір блоку, біт	Розмір ключа, біт	Складність виконання, операцій
ГОСТ28147-89	64	256	$K_A = 384N_D + 32$
DES	64	56	$K_A = 692N_D$
TrippleDES	64	112	$K_A = 2076N_D$
IDEA	64	128	$K_A = 216N_D + 92$
RC5-32/12/16	64	128	$K_A = 74N_D + 144$
Rijndael (AES)	128	128	$K_A = 688N_D + 168$
RC6-64/20/16	128	128	$K_A = 264N_D + 552$
Twofish	128	128	$K_A = 312N_D + 14040$

Отже, кількість операцій для обробки одного блоку даних за допомогою СБАШ залежить від типу алгоритму, розміру блоку даних та розміру ключа шифрування і лежить в межах декількох сотень – десятків тисяч операцій. Як було зазначено вище, СБАШ можуть бути реалізованими за допомогою широкого спектру універсальних та спеціалізованих засобів, таких як універсальні програмовані процесори (УПП), процесори обробки сигналів (ПОС), програмовані логічні інтегральні схеми (ПЛІС) та замовлені інтегральні схеми (ЗІС) [8, 9, 10] (табл. 3). Для ПЛІС та ЗІС подано два значення продуктивності роботи в Мбіт/с – при мінімальному (мін) та максимальному (макс) використанні обладнання.

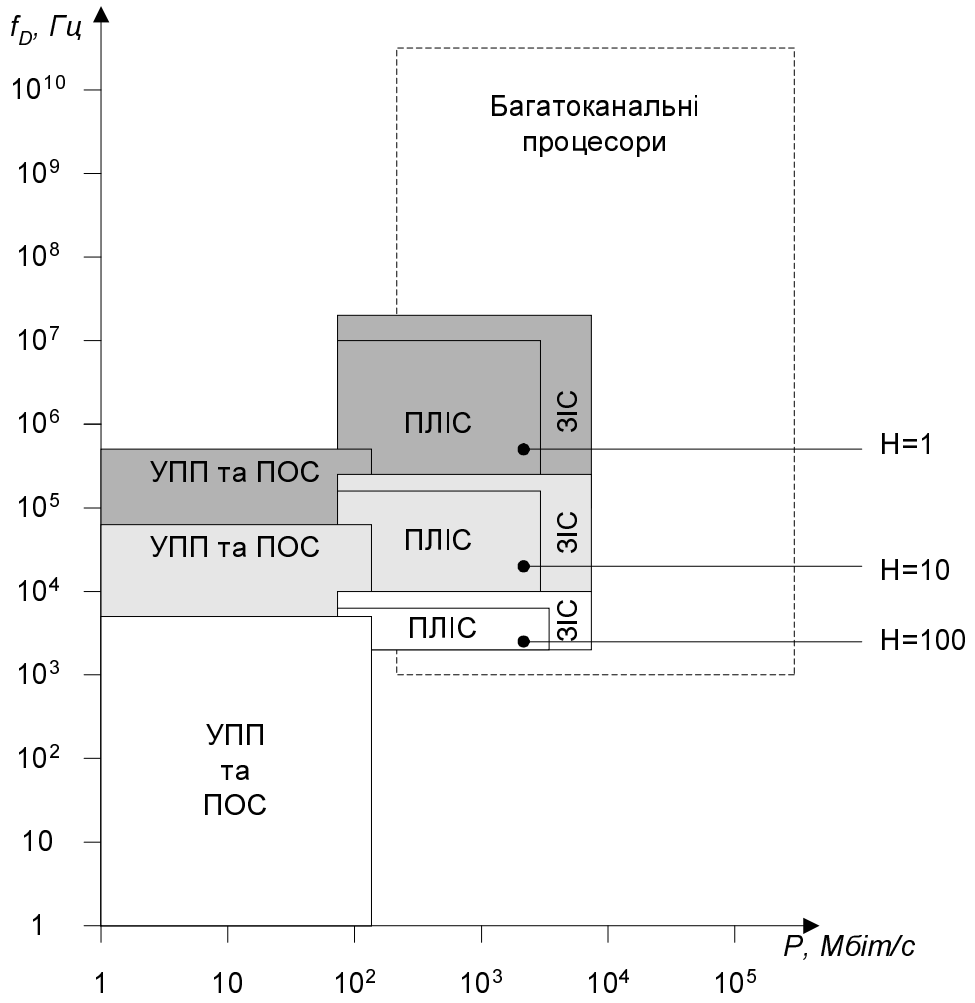
Таблиця 3

СБАШ	Універсальні програмовані процесори											Програмовані		Замовлені	
	Pentium, МГц			Pentium II, МГц					SGI R12000, МГц	Ultra SPARC II, МГц	TMS 320C6x, МГц	Логічні інтегральні схеми		Інтегральні схеми	
	90	166	266	200	200	200	266	450	300	2*366	200	мін	макс	мін	макс
MARS				26,5	38,1	69,4		188	65,67	35,49	91,4	101,8		55	2200
RC6				30,29	36,4	112,8		258	86,27	25,3	128	112	2400	102	2100
Rijndael	42,4			31,64	41,6	70,5		243	66,1	54,67	112,3	353	1940	443	5450
Serpent				7,46	12,7	26,8			43,14	39,98	33,2	146	4860	202	8100
Twofish				24,31	20,1	95		204	60,32	47,06	148,9	173	1590	103	2300
GOST	32,8											200		16	
RC5	31,8	58,5	91,2				171,2								
DES	16,9	19,2	29,6				39,2					400		1000	
3DES	6,2	7,2	11,2				14,4					150		100	
IDEA	9,75	20,8	33,6				32,8							245	

* пусті клітинки – відсутність у авторів інформації.

Скориставшись виразом (10), табл. 2 і 3, побудуємо діаграму областей доцільного використання одноканальних засобів шифрування даних як залежність частоти надходження вхідних даних від продуктивності цих засобів при H=1, H=10 та H=100 (рис. 1).

Тобто, збільшення кількості оброблюваних каналів обробки даних на одноканальних засобах виконання СБАШ суттєво зменшує частотний діапазон надходження даних із каналів. В той же час сучасні задачі захисту даних та напрямки розвитку великих інтегральних схем захисту інформації [3] вимагають постійного збільшення частоти надходження даних та кількості оброблюваних каналів. Оскільки існуючі одноканальні засоби виконання СБАШ не можуть задовольнити постійно зростаючі вимоги до швидкості обробки даних, зокрема багатоканальної обробки, то виникає задача побудови спеціалізованих багатоканальних процесорів СБАШ, які позбавлені цих недоліків (штрихова область на рисунку). Багатоканальний процесор СБАШ повинен забезпечувати неперервний прийом, обробку даних у міру їх надходження та видачу результуючих масивів даних для багатоканальної обробки даних у реальному масштабі часу.



Області доцільного використання засобів СБАШ

V. Висновки

У роботі проведено аналіз структурної організації СБАШ. Для основних складових СБАШ: процедури обчислення розпису ключа та процедури обробки даних, запропоновано аналітичні вирази обчислення їх складності, що дало змогу провести оцінку складності виконання СБАШ у типових умовах застосування. Проведено аналіз функціональних залежностей складності СБАШ від розмірів масивів даних та ключів, кількості операцій для виконання кожної структурної складової СБАШ, що дозволило запропонувати відповідні аналітичні вирази оцінки складності.

Базуючись на отриманих виразах оцінки складності СБАШ, запропоновано вираз для розрахунку необхідної продуктивності процесора шифрування симетричними блоковими алгоритмами у типових умовах використання при багатоканальній обробці даних. Проведено оцінку складності широко - вживаних СБАШ, що дало змогу визначити області доцільного використання існуючих процесорів СБАШ при обробці даних як з одного, так і з декількох каналів. Запропоновано створювати багатоканальні процесори СБАШ, які би були позбавлені недоліків, притаманних одноканальним процесорам при обробці даних із декількох каналів у реальному масштабі часу.

1. Schneier B, Applied Cryptography, // John Wiley & Sons. – 1996. 2. Мельник А., Коркішко Т. Стан та напрямки розвитку надвеликих інтегрованих схем захисту інформації // Зб. праць II

наук.-техн. конф. «Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні». 24 – 26 квітня 2000. – К., 2000. – С. 275 – 281. 3. Черкаський М., Складність апаратно-програмних комп'ютерних засобів // Матеріали міжнар. наук.-техн. конф. "Сучасні проблеми в комп'ютерних науках в Україні" (CCU'2000). – Славське, 2000. – С. 58 – 67. 4. Мельник А.О. Спеціалізовані комп'ютерні системи реального часу. – Львів, 1996. – 53 С. 5. ГОСТ 28147-89. Система обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. 6. Weeks B., Bean M., Rozylowicz T., Ficke C. Hardware performance simulations of Round 2 Advanced Encryption Standard algorithms. // Proceedings of 3rd AES conference. – New York, Apr. 2000. P. 286 – 304. 7. Dandalis A., Prasanna V. K., P. Rolim J. D., A Comparative Study of Performance of AES Final Candidates Using FPGAs, in Workshop on Cryptographic Hardware and Embedded Systems – CHES '00 (C. Koc and C. Paar, eds.), (Worcester, Massachusetts, USA), Springer-Verlag, – Aug 2000. 8. Bassham L., Efficiency Testing of ANSI C Implementations of Round1 Candidate Algorithms For The Advanced Encryption Standard, NIST AES report.

УДК 681.317

В.Т. Кремінь
Національний університет "Львівська політехніка",
кафедра ЕОМ

КВАЗІПАРАЛЕЛЬНИЙ АЛГОРИТМ МІНІМІЗАЦІЇ БАГАТОЕКСТРЕМАЛЬНИХ ФУНКЦІЙ

Розроблено квазіпаралельний алгоритм оптимізації багатоекстремальних функцій із застосуванням паралельної оптимізації функції за допомогою кількох методів. Наведено результати порівняльного тестування розробленого алгоритму із існуючими.

New quasi-parallel optimization method for minimization multiextremum functions is proposed. The method is suitable for model parameter estimation using experimentally measured data. The test examples show better probability of convergence and possibility to obtain lower cost function values in comparison with other existing methods.

Вступ

Розв'язання багатьох технічних задач часто вимагає оптимізації функцій багатьох змінних. Можна навести ряд прикладів таких задач: вибір параметрів систем автоматичного керування, визначення параметрів моделей об'єктів різних типів за певними виміряними характеристиками цих моделей, синтез частотних фільтрів із заданими характеристиками тощо. Типовий приклад застосування методів оптимізації – визначення параметрів моделей компонентів електронних схем для підсистем автоматизованого проектування, наприклад SPICE. На основі певних виміряних характеристик компонентів необхідно визначити параметри моделі компонента.