- Препринт. – Львов, 1990. – 43 с.
5. G. Biand E.V. Jones A Pipelined FFT Processor for Word-Sequential Data, *IEEE Transactions on Acoustics, Speech and Signal Processing*. vol.37, n.12, pp.1982-5, 1989.
6. N.R. Murthy and N.M.S. Swamy On the Real-Time Computation of DFT and DCT through Systolic Architectures, *IEEE Transactions on Signal Pricessing*, vol.42, n.4, pp.988-91, 1994.
7. Melnyk A. DSP System Based on Programmable Processor with Scalable Parametrizable Fast Orthogonal Transforms Hardware Core // *Proceedings of the XI Conference "Application of Microprocessors in Automatic Control and Measurement"*, V. 1, Warsaw, Poland, 1998, P.87-98.
8. Ерметов Ю.О. Синтез конвеєрних пристроїв для реалізації матриці корегування в алгоритмі швидкого косинусного перетворення //*Вісник ДУ "ЛП" "Комп'ютерні системи проектування. Теорія і практика"*. – 1999. – № 373. – С.128-136.
9. Мельник А.О., Ерметов Ю.О. Розробка двоканальних конвеєрних пристроїв для реалізації матриці корегування в алгоритмі швидкого косинусного перетворення //*Вісник ДУ "ЛП" "Комп'ютерна інженерія та інформаційні технології"*.– 1999. – №370. – С.28-34.
10. Мельник А.О., Ерметов Ю.О. Паралельний алгоритм обчислення матриці корегування швидкого косинусного перетворення //*Вісник ДУ "ЛП" "Комп'ютерні системи та мережі"*. – 2000. – №385. – С.100-109.
11. Мельник А.О. Спеціалізовані комп'ютерні системи реального часу. – *Львів: ДУ"Львівська політехніка"*, 1996. – 53 с.
12. Валях Е. Последовательно-параллельные вычисления. – *М.: Мир*, 1985. – 456 с.
13. Прангишвили И.В., Виленкин С.Я., Медведев И.Л. Параллельные вычислительные системы с общим управлением. – *М.: Энергоатомиздат*, – 1983. – 312 с.
14. Constant Coefficient Multipliers for the XC4000E. Application Note by Ken Chapman. -http:// www.xilinx.com

Порівняльний аналіз варіантів структурної організації процесора захисту інформації за алгоритмом DES

© В. Мельник[1)], Т. Коркішко[2)], А. Мельник[1)], Ю. Байсіг[3)], 2000
[1)]ДУ "Львівська політехніка", Кафедра ЕОМ, vamelnyk@polynet.lviv.ua, aomelnyk@polynet.lviv.ua
[2)]ТАНГ, м.Тернопіль, пл. Перемоги, 3
[3)]ВТШ ім. Г.С.Ома, м. Нюрнберг, Німеччина

Comparative analysis of the DES cryptographic processor's architecture

Abstract — **On this article the DES algorithm and comparative analysis of the DES cryptographic processor's architecture were considered. As the criteria of comparision the performance of the processor's work and equipment volume were taken.**

Key words — **DES algorithm, IP core, processor core.**

Анотація — **Розглянуто алгоритм шифрування даних DES та варіанти побудови процесора, що здійснює обробку інформації за даним алгоритмом. Проведено аналіз та порівняння варіантів побудови процесора. За критерій порівняння взято продуктивність роботи процесора та затрати обладнання на його реалізацію.**

Ключові слова — **захист інформації, алгоритм DES, режими роботи алгоритму DES, процесор, ядро процесора.**

Вступ

Одним із звдань захисту інформації є забезпечення її конфіденційності. Це завдання постає в областях, де необхідно здійснювати передавання інформації, її збереження та накопичення.

Існує кілька підходів до забезпечення конфіденційності інформації. Одним із них є використання алгоритмів блокового шифрування із симетричними ключами. На сьогодні найпоширенішим алгоритмом цього типу є DES (Data Encryption Standard), який прийнятий як стандарт шифрування інформації в США [1]. Також DES використовується як складовий алгоритм в інших стандартах, наприклад шифрування у комп'ютерних мережах типу ATM (Asynchronous Transfer Mode), протокол SSL (Secure Socket Layer), який забезпечує захист

інформації під час роботи в мережі, американська система банківських (і не тільки) стандартів ANSI.

Алгоритм DES має чотири режими роботи: ECB, CBC, CFB та OFB, кожен з яких характеризується своїми показниками криптографічної стійкості.

Для реалізації алгоритму DES використовуються програмні та апаратні комп'ютерні засоби. Програмний підхід забезпечує гнучкість зміни режимів роботи алгоритму, але має такі недоліки, як слабкий ступінь захисту ключів шифрування та невисоку продуктивність шифрування інформації. Апаратна реалізація алгоритму дозволяє досягти високої продуктивності шифрування інформації та достатнього рівня захисту ключів від зовнішнього втручання. Тому сьогодні основна увага розробників сконцентрована на цьому напрямку, що вилилось у створення ряду багаторежимних процесорів DES. Разом з тим, наявність програмованих логічних пристроїв високого ступеня інтеграції з можливістю їх перепрограмування дозволяє створювати процесори, які орієнтовані на один режим роботи. Така вузька орієнтація дозволяє оптимізувати їх структуру та отримати оптимальні показники для кожного з режимів роботи.

1. Алгоритм DES

Алгоритм шифрування даних DES був затверджений урядом США у 1977 році. Це блоковий шифр, який перетворює 64-бітовий вхідний блок даних за допомогою 56-бітового ключа у 64-бітовий вихідний блок даних [1,3]. Дані подаються на вхід паралельним 64-розрядним кодом (рис.1).

Загалом процес оброблення даних здійснюється за 16 етапів. З вхідного ключа генеруються шістнадцять 48-бітових підключів для кожного етапу окремо. Всередині кожного етапу використовуються вісім блоків підміни даних, кожен з яких відповідно до 6-бітового вхідного блоку подає на вихід 4-бітовий блок. Всі етапи є функціонально еквівалентними, на їх входи надходять 32-бітові блоки даних L_{i-1} та R_{i-1} (які формуються шляхом поділу вхідного 64-бітового блоку на дві частини) від попереднього етапу, а вихідні 32-бітові блоки L_i та R_i обчислюються відповідно до наступних формул:

для $1 \le i \le 16$

$$L_i = R_{i-1}; \quad (1)$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i), \text{ де } f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i)). \quad (2)$$

Тут E – позначення дії розширення блоку R_{i-1} з 32 до 48-бітового розміру (всі біти використовуються, деякі подвоюються); P – позначення дії переставлення розрядів у 32-бітових блоках.

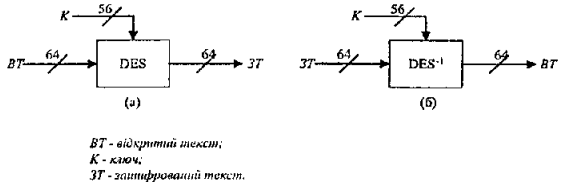

Рис.1. Інформаційний інтерфейс алгоритму DES в режимах зашифрування (а) та розшифрування (б)

Порядок переставлення є фіксованим і визначеним наперед.

Початкове переставлення IP (Initial Permutation) здійснюється перед першим етапом шифрування. На кожному етапі, крім останнього, ліва та права частини кодованого блоку даних L_i та R_i міняються місцями. Кінцеве переставлення IP^{-1} здійснюється над вихідним кодом по закінченні його оброблення в останньому етапі. Розшифрування інформації здійснюється тим же ключем, яким вона була зашифрована, але підключі до кожного з етапів алгоритму генеруються у зворотному порядку.

Алгоритм 1 дає детальну інформацію про порядок проведення оброблення даних за алгоритмом DES.

Алгоритм 1: алгоритм DES.

Вхід: відкритий текст $m_1 \ldots m_{64}$; 64-бітовий ключ $K = K_1 \ldots K_{64}$.

Вихід: 64-бітовий код $C = C_1 \ldots C_{64}$.

1. Обчислення шістнадцяти 48-бітових підключів K_i, використовуючи 64-бітовий вхідний ключ K (*для обчислення підключів використовується Алгоритм 2*).

2. $(L_0, R_0) \leftarrow IP(m_1, m_2, \ldots, m_{64})$.

3. (16 етапів) для $1 \le i \le 16$ обчислюється L_i та R_i, використовуючи формули (1) та (2), наведені вище. Обчислюється $f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$ у наступному порядку:

a) розширення $R_{i-1} \leftarrow r_1, r_2, \ldots, r_{32}$ з 32 до 48 біт (використовуючи E); $T \leftarrow E(R_{i-1})$;

b) $T' \leftarrow T \oplus K_i$. T′ визначається як вісім 6-бітових стрічок $(B_1, \ldots, B_8) = T'$;

c) $T'' \leftarrow (S_1(B_1), S_2(B_2), \ldots, S_8(B_8))$;

d) $T''' \leftarrow P(T'')$.

4. $b_1, b_2, \ldots, b_{64} \leftarrow (R_{16}, L_{16})$.

5. $C \leftarrow IP^{-1}(b_1, b_2, \ldots, b_{64})$.

Ефективний розмір вхідного ключа K становить 56 біт з 64, решта біт розглядаються як біти парності. Як правйло, це біти з номерами 8, 16,..., 64. Алгоритм 2 ілюструє порядок обчислень підключів K_i етапів DES, кожен з яких містить 48 бітів вхідного ключа K. Алгоритм використовує таблиці переставлення даних PC1 та PC2 (Permuted Choice), перша з яких перетворює вхідний 64-бітовий ключ у 56-бітовий шляхом виключення його бітів 8, 16,..., 64. Розряди

56-бітового блоку переставляються за певним наперед визначеним та постійним алгоритмом. Далі цей блок ділиться на два блоки по 28 біт, позначені в алгоритмі як C та D. На кожному етапі над обома цими блоками здійснюється операція циклічного зсуву на один чи два розряди.

Алгоритм 2: обчислення ключів шифрування.

Вхід: 64-бітовий ключ $K=K_1,...,K_{64}$ (включаючи біти парності).

Вихід: шістнадцять 48-бітових підключів K_i, $1 \le i \le 16$.

1. V_i визначається наступним способом: $V_i = 1$ для $i \in \{1,2,9,16\}$, інакше $i = 2$ (це є величина зсуву вліво 28-бітових блоків).

2. $T \leftarrow PC1(K)$; T представляється як 28-бітові змінні C_0, D_0.

3. Для $1 \le i \le 16$ підключ K_i вираховується згідно з наступними формулами: $C_i \leftarrow (C_{i-1} \hookleftarrow V_i)$, $D_i \leftarrow (D_{i-1} \hookleftarrow V_i)$, $K_i \leftarrow PC2(C_i, D_i)$ (операція "$\hookleftarrow$" означає циклічний зсув вліво).

У випадку проведення розшифрування інформації за алгоритмом DES використовується той самий ключ, що й при її зашифруванні, але обчислення підключів здійснюється у зворотному порядку.

На рис.2 подана структура, яка повністю відображає порядок оброблення інформації за стандартом DES та ілюструє виконання обчислень за наведеними вище алгоритмами. Рис.3 зображає вмістиме функції *f*, яка використовується всередині кожного етапу і є стандартною.

2. Режими роботи алгоритму DES

Блокові шифри обробляють відкритий текст блоками фіксованого розміру по n біт (здебільшого n = 64). Для шифрування блоків, що мають розмір більший вказаного, найпростішим рішенням є їх розбиття на n-бітові блоки, кожен з яких може бути кодований окремо.

Найшире у блокових шифрах використовують наступні чотири режими: ECB, CBC, CFB та OFB [2, 3]. Кожен з них має свої переваги та недоліки, які будуть описані далі, але всі вони можуть використовувати алгоритми блокового шифрування. У подальшому позначення E_K визначатиме функцію шифрування інформації блоковим шифром, який параметризується вхідним ключем K тоді, коли E_K^{-1} – функція розшифрування інформації. Вхідний відкритий текст $X=X_1...X_t$ складається з n-бітових блоків для

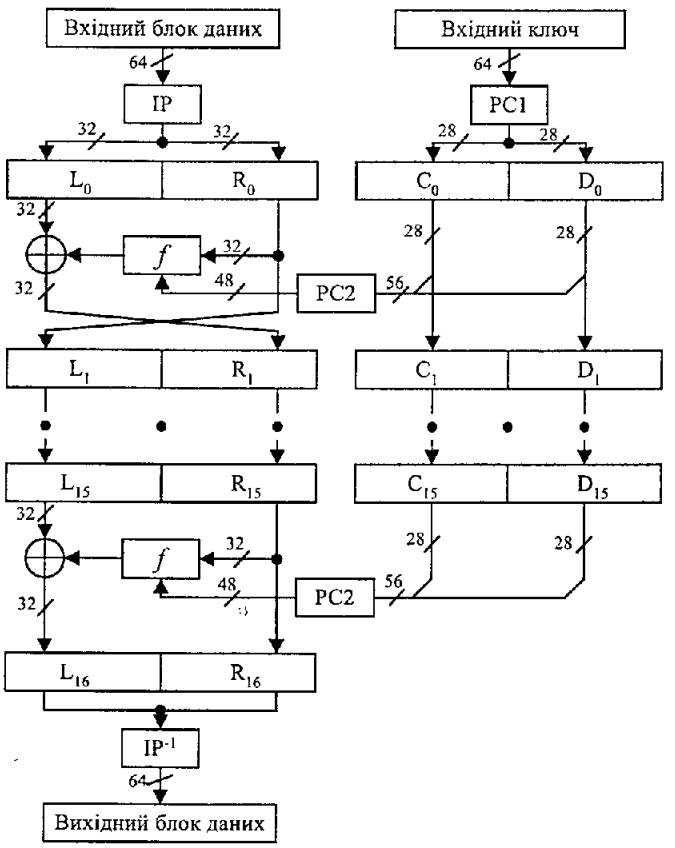

Рис.2. Обробка інформації за алгоритмом DES та генерація підключів

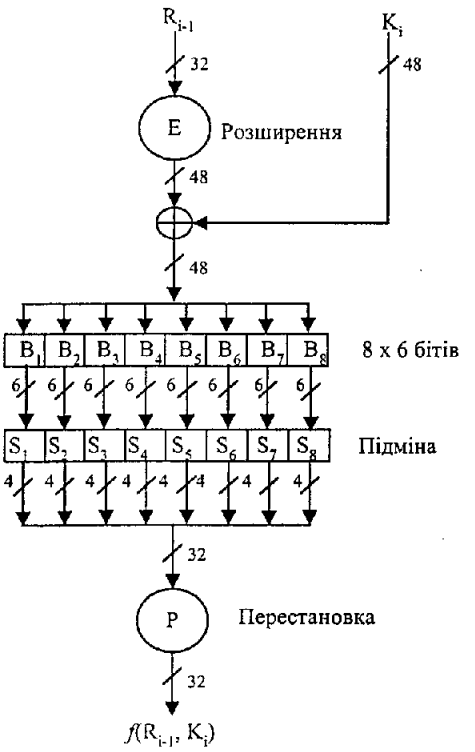

Рис.3. Функція *f*.

режимів ECB та CBC, або з r-бітових блоків для режимів CFB та OFB при фіксованому r ≤ n. Далі кожен з перелічених режимів розглядається детальніше.

1. Режим електронної кодової книги ECB (Electronic Codebook) [2] є найпростішим режимом роботи блокових шифрів. Рис.4 та Алгоритм 3 ілюструють роботу алгоритму DES у цьому режимі.

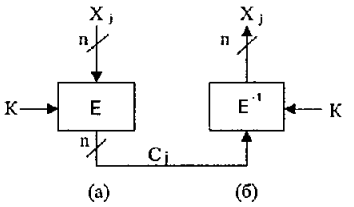

Рис.4. Робота алгоритму DES в режимі ECB зашифрування (а) та розшифрування (б)

Алгоритм 3: режим електронної кодової книги.

На вхід подаються k-бітовий ключ, n-бітові блоки відкритого тексту $X_1...X_t$. Продукуються n-бітові блоки вихідного коду $C_1...C_t$, з якого при розшифруванні формується відновлений вхідний текст.

a. Зашифрування: для $1 \le j \le t$, $C_j \leftarrow E_K(X_j)$.

b. Розшифрування: для $1 \le j \le t$, $X_j' \leftarrow E_K^{-1}(C_j)$.

2. Режим зчеплення у ланцюжок блоків зашифрованого тексту CBC (Cipher Block Chaining) [2] використовує вектор ініціалізації IV (Initialization Vector) та зворотний зв'язок. Рис.5 та Алгоритм 4 ілюструють роботу алгоритму DES у цьому режимі.

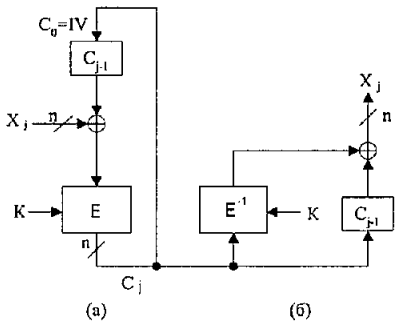

Рис.5. Робота алгоритму DES в режимі CBC зашифрування (а) та розшифрування (б)

Алгоритм 4: режим зчеплення у ланцюжок блоків зашифрованого тексту.

На вхід подаються k-бітовий ключ, n-бітовий вектор ініціалізації IV, n-бітові блоки відкритого тексту $X_1...X_t$. Продукуються n-бітові блоки вихідного коду $C_1...C_t$, з якого при розшифруванні формується відновлений вхідний текст.

a. Зашифрування: $C_0 \leftarrow IV$. Для $1 \le j \le t$, $C_j \leftarrow E_K(C_{j-1} \oplus X_j)$.

b.Розшифрування: $C_0 \leftarrow IV$. Для $1 \le j \le t$, $X_j' \leftarrow C_{j-1} \oplus E_K^{-1}(C_j)$.

3. У той час, коли режими ECB та CBC обробляють вхідний текст блоками по n біт (використовуючи n-бітовий блоковий шифр), у деяких випадках потрібно обробляти r-бітовий вхідний текст при фіксованому $r \le n$ (часто r=1 або r=8, але в будь-якому випадку $r=2^S$, де S=0...6). У такому випадку може використовуватись режим шифрування з внутрішнім зворотним зв'язком CFB (Cipher Feedback) [2]. Рис.6 та Алгоритм 5 ілюструють роботу алгоритму DES у цьому режимі.

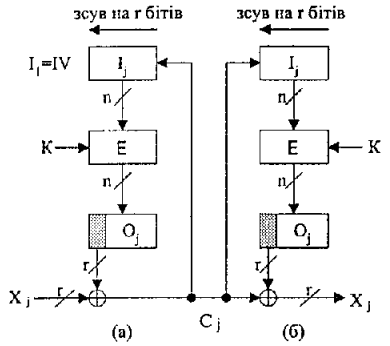

Рис.6. Робота алгоритму DES в режимі CFB зашифрування (а) та розшифрування (б).

Алгоритм 5: режим шифрування з внутрішнім зворотним зв'язком.

На вхід подаються k-бітовий ключ, n-бітовий вектор ініціалізації IV, n-бітові блоки відкритого тексту $X_1...X_U$. Продукуються r-бітові блоки вихідного коду $C_1...C_U$, з якого при розшифруванні формується відновлений вхідний текст.

a. Зашифрування: $I_1 \leftarrow IV$ (I_j – вхідне значення регістра зсуву);

для $1 \le j \le U$:

a) $O_j \leftarrow E_K(I_j)$;

b) $t_j \leftarrow$ r лівих бітів O_j;

c) $Cj \leftarrow X_j \oplus t_j$;

d) $I_{j-1} \leftarrow 2^r * I_j + C_j \bmod 2^n$.

b. Розшифрування: $I_1 \leftarrow IV$; для $1 \le j \le U$:

$X''_j \leftarrow C_j \oplus t_j$, де t_j, O_j та C_j обчислюються так само, як і у випадку зашифрування.

4. Режим шифрування зі зворотним зв'язком по виходу OFB (Output Feedback) [2] використовується поряд з попередньо розглянутим режимом CFB. Рис.7 та Алгоритми 6 та 7 ілюструють роботу алгоритму DES у цьому режимі.

У режимі OFB може використовуватись як повний зворотний зв'язок (ISO-версія), так і

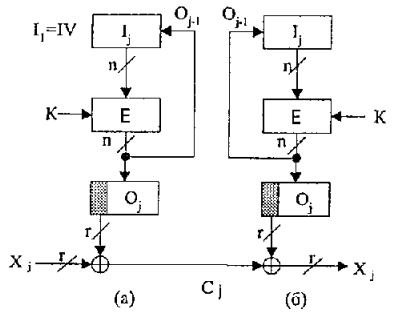

Рис.7. Робота алгоритму DES в режимі OFB зашифрування (а) та розшифрування (б)

r-бітовий зворотний зв'язок (FIPS-версія). Алгоритм 6 відображає роботу в режимі OFB з повним (n-бітовим) зворотним зв'язком, а Алгоритм 7 – OFB з r-бітовим зворотним зв'язком по виходу.

Алгоритм 6: режим шифрування з повним зворотним зв'язком по виходу (ISO-версія).

На вхід подаються k-бітовий ключ, n-бітовий вектор ініціалізації IV, n-бітові блоки відкритого тексту $X_1...X_U$. Продукуються r-бітові блоки вихідного коду $C_1...C_U$, з якого при розшифруванні формується відновлений вхідний текст.

a. Зашифрування: $I_1 \leftarrow IV$ (I_j – вхідне значення регістра зсуву);

для $1 \le j \le U$:

a) $O_j \leftarrow E_K(I_j)$;

b) $t_j \leftarrow$ r лівих бітів O_j;

c) $C_j \leftarrow X_j \oplus t_j$;

d) $I_{j+1} \leftarrow O_j$.

b. Розшифрування: $I_1 \leftarrow IV$; для $1 \le j \le U$:

$X'_j \leftarrow C_j \oplus t_j$, де t_j, O_j та C_j обчислюються так само, як і у випадку зашифрування.

Алгоритм 7: режим шифрування з r-бітовим зворотним зв'язком по виходу (FIPS-версія).

На вхід подаються k-бітовий ключ, n-бітовий вектор ініціалізації IV, n-бітові блоки відкритого тексту $X_1...X_U$. Продукуються r-бітові блоки вихідного коду $C_1...C_U$, з яких при розшифруванні формується відновлений вхідний текст.

a. Зашифрування: $I_1 \leftarrow IV$ (I_j – вхідне значення регістра зсуву);

для $1 \le j \le U$:

a) $O_j \leftarrow E_K(I_j)$;

b) $t_j \leftarrow$ r лівих бітів O_j;

c) $C_j \leftarrow X_j \oplus t_j$;

d) $I_{j+1} \leftarrow 2^r * I_j + t_j \bmod 2^n$.

b. Розшифрування: $I_1 \leftarrow IV$; для $1 \le j \le U$:

$X'_j \leftarrow C_j \oplus t_j$, де t_j, O_j та C_j обчислюються так само, як і у випадку зашифрування.

Порівнюючи характеристики кожного з режимів, варто відзначити наступне.

У режимі ECB нема залежностей між кодованими блоками даних, тому переставлення блоків коду призводить до відповідних змін у порядку надходження вихідного декодованого тексту, сама ж інформація всередині кожного блоку змін не зазнає. Ідентичні блоки вхідного тексту при зашифруванні зумовлюють появу ідентичних блоків вихідного коду, що знижує його криптографічну стійкість. Через це режим ECB не рекомендується використовувати для кодування повідомлень довжиною більшою, ніж один блок, при однаковому ключі. Недоліки режиму ECB усуваються іншими трьома режимами, в яких вводяться зворотні зв'язки, відповідно існує певна залежність між блоками оброблюваної інформації, що підвищує криптографічну стійкість коду.

3. Варіанти побудови процесора

Вище було розглянуто режими роботи блокових шифрів та алгоритм шифрування даних за американським стандартом DES. Розглянемо варіанти побудови процесора, який виконував би обробку інформації за даним алгоритмом, використовуючи можливі режими роботи блокових шифрів. На рис.8 зображено загальну структуру такого процесора [4].

Ядром процесора будемо називати операційний пристрій, який апаратно реалізує алгоритм DES і може бути використаний у розглянутих режимах роботи блокових шифрів. Комутуюча мережа призначена для конфігурування процесора для роботи у одному з режимів. Вхідний та вихідний інтерфейси забезпечують відповідно прийом вхідних даних, зберігання проміжних результатів та видачу результатів обчислень.

Розбиття структури процесора на описані вище блоки зроблено з наступних міркувань: незалежно від режиму роботи процесора алгоритм роботи його ядра буде незмінним, а решта функціональних блоків (комутуюча мережа, блок керування, вхідний та вихідний інтерфейси) будуть працювати за алгоритмом, в основі якого закладений конкретний режим роботи процесора. При оптимізації процесора під конкретний режим роботи структура ядра залишається постійною, а структура інших функціональних блоків спрощується.

Розглянемо варіанти реалізації ядра процесора. Як вже було сказано раніше, алгоритм DES складається з 16 етапів і має однорідну структуру (рис.2). Цю його особливість можна використати при побудові ядра процесора. Виділяються два варіанти такої реалізації: ітераційний та конвеєрний. Приклад ядра, реалізованого

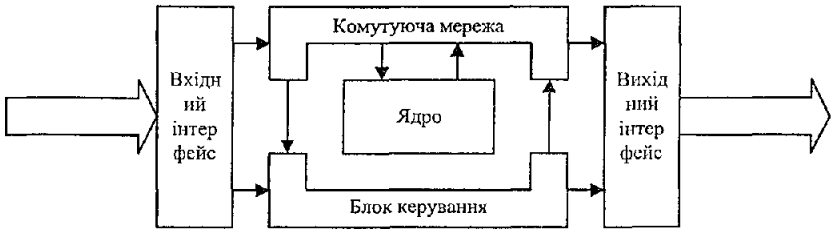

Рис.8. Загальна структура процесора захисту інформації за алгоритмом DES

ітераційно, наведений на рис.9,а, а ядра, реалізованого за конвеєрним принципом, – на рис.9,б.

На основі аналізу поданих вище структур можемо сказати, що ітераційна структура ядра процесора забезпечує можливість його реалізації при мінімальних апаратних затратах. Конвеєрна структура ядра процесора є вигідною з точки зору продуктивності.

Табл.1 ілюструє на функціональному рівні можливі варіанти побудови процесора, який виконував би оброблення інформації за алгоритмом DES. Табл.1 містить 26 варіантів побудови такого процесора. Як параметри цих процесорів тут використовуються варіанти реалізації ядра (ітераційний і конвеєрний) та функціональна орієнтація процесора (багаторежимні і орієнтовані на режим). Багаторежимні процесори обробляють інформацію у всіх вище-наведених режимах, здійснюючи як її зашифрування, так і розшифрування. Орієнтація на конкретний режим включає по три варіанти на кожен – зашифрування (E - Encryption), розшифрування (D - Decryption) та обидва напрямки (E і D). Існує можливість побудови процесора, орієнтованого на декілька режимів. При цьому повинна враховуватись подібність алгоритмів роботи процесора у певних режимах. Доцільність такої реалізації буде обгрунтовано у наступному пункті.

4. Аналіз та порівняння варіантів побудови процесора

Розглянемо основні переваги та недоліки ітераційного та конвеєрного принципів реалізації процесора, що обробляє дані за алгоритмом DES.

Якщо при побудові процесора основним завданням є його реалізація при мінімальних затратах обладнання, то прийнятною є ітераційна структура ядра процесора. Одразу ж можна сказати, що продуктивність роботи такого процесора буде невисокою, бо у кожен момент часу ним може оброблятися тільки один блок даних. Якщо ж ми маємо справу з конвеєрним ядром процесора, то продуктивність його роботи в порівнянні з продуктивністю роботи ітераційного ядра процесора буде значно вищою.

Можливі варіанти побудови комбінованої конвеєрно-ітераційної структури. Якщо взяти до уваги те, що за алгоритмом DES дані обробляються за шістнадцять ідентичних етапів, то чиста ітераційна структура повинна містити один етап, інформація по якому повинна пройти шістнадцять разів і тільки після того йти на вихід. Чистий конвеєр повинен складатися з шістнадцяти ярусів. У випадку його повного завантаження одночасно обробляються шістнадцять блоків даних, тобто продуктивність його роботи в порівнянні з продуктивністю роботи ітераційного процесора буде вищою приблизно в шістнадцять разів. Якщо реалізована ітераційна структура, яка містить кілька ярусів конвеєра, продуктивність його роботи буде вищою в порівнянні з чистою ітераційною структурою, але відповідно збільшаться апаратні затрати. Як показали дослідження, найоптимальнішим за часом обробки одного блоку даних та за ефективністю використання апаратури є варіант побудови чистого конвеєра при повному його завантаженні, тобто конвеєра з шістнадцятьма ярусами [5].

Далі наведено результати аналізу кожної з моделей та оцінено доцільність побудови процесора за розглянутими вище варіантами його побудови (табл.1).

1. Багаторежимні процесори.

- Ітераційна реалізація ядра.

При обгрунтуванні доцільності реалізації багаторежимного ітераційного процесора оглянемо можливі варіанти його побудови та оцінимо його ефективність для кожного з режимів роботи.

З рис.4 видно, що в режимі ECB продуктивність роботи процесора буде дорівнювати продуктивності роботи його ядра. Тут може бути використана як ітераційна, так і конвеєрна модель ядра процесора в залежності від конкретної потреби: висока продуктивність чи мінімальні затрати обладнання.

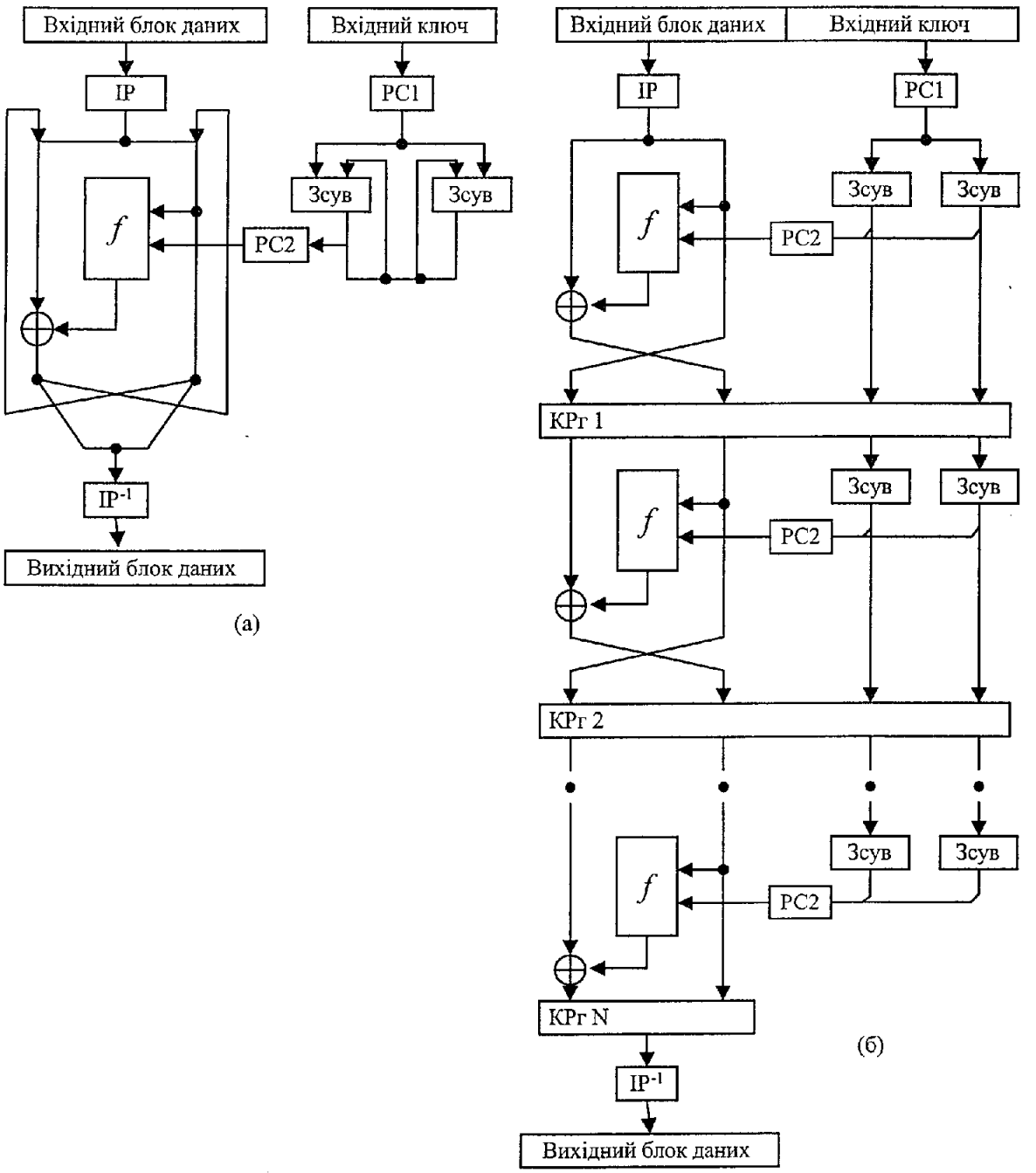

Рис.9. Структура ітераційного (а) та конвеєрного (б) ядер процесора

Таблиця 1

Можливі варіанти побудови процесора захисту інформації за алгоритмом DES

Реалізація ядра	**Функціональна орієнтація**	**№**	**ECB**		**CBC**		**CFB**		**OFB**	
			E	**D**	**E**	**D**	**E**	**D**	**E**	**D**
Ітераційна	Багаторежимні	1	✓	✓	✓	✓	✓	✓	✓	✓
	Орієнтовані на режим	2	✓	✓						
		3			✓	✓				
		4					✓	✓		
		5							✓	✓
		6	✓							
		7		✓						
		8			✓					
		9				✓				
		10					✓			
		11						✓		
		12							✓	
		13								✓
Конвеєрна	Багаторежимні	14	✓	✓	✓	✓	✓	✓	✓	✓
	Орієнтовані на режим	15	✓	✓						
		16			✓	✓				
		17					✓	✓		
		18							✓	✓
		19	✓							
		20		✓						
		21			✓					
		22				✓				
		23					✓			
		24						✓		
		25							✓	
		26								✓

На рис.5 показано роботу алгоритму DES у режимі CBC, звідки видно, що у випадку зашифрування конвеєрна модель тут не є ефективною, оскільки в ядрі в кожен момент часу може оброблятись лише один блок даних. Це зумовлено наявністю зворотного зв'язку з виходу ядра на його вхід. У випадку розшифрування зворотні зв'язки відсутні, тому з точки зору продуктивності доцільно використати конвеєр. У режимах CFB (рис.6) та OFB (рис.7) конвеєрний варіант реалізації ядра процесора не є ефективним (продуктивність його є така сама, як і в ітераційного) за наявністю зворотних зв'язків (ситуація, аналогічна до режиму CBC (зашифрування).

Отже, при побудові багаторежимного процесора ітераційна модель його ядра є більш прийнятною, ніж конвеєрна, завдяки суттєвій економії апаратних затрат при втратах продуктивності роботи в порівнянні з конвеєрною моделлю ядра тільки у режимах ECB та CBC (розшифрування).

- Конвеєрна реалізація ядра.

Зі сказаного вище можна зробити висновок, що конвеєрна модель при побудові ядра багаторежимного процесора суттєвого ефекту не дає. Хоча продуктивність підвищується для роботи у режимах ECB та CBC (розшифрування), в інших режимах вона буде аналогічною до продуктивності роботи ітераційного ядра процесора. Апаратні затрати на реалізацію при цьому значно збільшуються.

2. Орієнтовані на режим E/D процесори.

Загалом побудова орієнтованих на режим DES-процесорів звужує сферу їх застосування у порівнянні з багаторежимними, але можливість досягнення вищої продуктивності роботи методом оптимізації під конкретний режим та економія апаратних затрат компенсують цей недолік.

- Ітераційна реалізація ядра.

Під час роботи у режимі ECB з точки зору продуктивності дана модель у порівнянні з конвеєрною є неефективною. Якщо акцентувати на апаратних затратах, то така модель буде значно оптимальнішою від конвеєрної.

Дві різні ситуації спостерігаються у режимі CBC: при зашифруванні дана модель є доцільнішою до реалізації, тому що в порівнянні з конвеєрною її продуктивність не є нижчою, а за апаратними затратами має беззаперечну перевагу. При розшифруванні продуктивність не змінюється, але у випадку конвеєризації її можна значно підвищити.

Як видно з рис.6, у режимі CFB вимагається, щоб наступний блок даних був завантажений у ядро тільки після того, як закінчиться обробка поточного блоку (як у випадку зашифрування, так і при розшифруванні). Тому ітераційна структура тут є більш прийнятною.

Ситуація в режимі OFB є аналогічною до розглянутої в режимі CFB.

- Конвеєрна реалізація ядра.

Зі сказаного вище випливає, що з точки зору продуктивності конвеєрна реалізація ядра є доцільною при побудові процесорів, орієнтованих на режим ECB. В інших випадках вона ефекту не дає, за винятком процесора, орієнтованого на режим CBC, де ситуація є неоднозначною.

3. Орієнтовані на режим E та D процесори.

Така вузька спеціалізація дозволяє ще більше спростити структуру процесорів та в певних режимах добитися збільшення продуктивності їх роботи.

У режимах ECB, CFB та OFB при спеціалізації на зашифрування та розшифрування даних окремо зберігаються вищенаведені властивості (спеціалізація на E/D). Окремо розглядається варіант побудови процесорів для режиму CBC. Як було сказано вище, при зашифруванні даних (спеціалізація на E) доцільною є реалізація процесора з ітераційною структурою ядра, а при побудові процесора, орієнтованого на розшифрування (спеціалізація на D), – конвеєрна реалізація.

Обгрунтуємо доцільність побудови процесора, орієнтованого на кілька режимів роботи. Як вже було сказано, тут повинна враховуватись подібність алгоритмів роботи процесора у цих режимах. На основі проведеного вище аналізу можна сказати, що конвеєрна реалізація ядра є ефективною при побудові процесорів, що працюють у режимах ECB зашифрування та розшифрування і CBC розшифрування. В інших режимах оптимальною є реалізація ядра з ітераційною структурою, тому що тут його продуктивність не поступається продуктивності ядра, реалізованого на конвеєрі, при цьому апаратні затрати суттєво зменшуються. Враховуючи те, що складність інших блоків процесора (комутуюча мережа, блок управління, вхідний та вихідний інтерфейси) при його орієнтації на кілька режимів роботи майже не зростає, можливим є створення процесора з конвеєрною структурою ядра, який обробляє інформацію у режимах ECB та CBC (розшифрування). Якщо ядро процесора має ітераційну структуру, то можна використати його переваги під час роботи у режимах CFB, OFB та CBC (розшифрування) і створити процесор, орієнтований на ці режими чи певні їх комбінації.

Висновки

У роботі розглянуто варіанти побудови процесора, що здійснює обробку інформації за алгоритмом DES. Основна увага була сконцентрована на наступних пунктах:

- алгоритм блокового шифрування даних із симетричним ключем DES та принципи обробки інформації за цим алгоритмом;
- режими роботи блокових шифрів: ECB, CBC, CFB та OFB;
- варіанти побудови DES-процесора, який би обробляв дані у всіх чотирьох режимах (багаторежимного процесора);
- варіанти побудови орієнтованого на режим процесора, який би виконував як зашифрування даних, так і їх розшифрування;
- варіанти побудови орієнтованого на режим процесора, який би виконував тільки зашифрування даних;
- варіанти побудови орієнтованого на режим процесора, який би виконував тільки розшифрування даних.

Проведено аналіз та порівняння вищезгаданих варіантів побудови процесора. За критерії порівняння взято продуктивність роботи процесора та апаратні затрати на його реалізацію. Запропоновано два основні варіанти реалізації ядра такого процесора – конвеєрний та ітераційний. Мета першого – досягнення максимальної продуктивності, другого – зменшення апаратних затрат на реалізацію. Розглянуто можливість використання комбінованого конвеєрно-ітераційного варіанта, а також створення процесора, орієнтованого на кілька режимів роботи.

З точки зору продуктивності ефективними є багаторежимний ітераційний процесор, орієнтовані на режим ітераційні CFB, OFB та CBC (зашифрування) процесори, орієнтовані на режим конвеєрні ECB та CBC (розшифрування) процесори.

З точки зору економії обладнання ефективними є всі варіанти процесора з ітераційним принципом реалізації ядра.

Література

1. FIPS 46, "Data Encryption Standard", Federal Information Processing Standard (FIPS), Publication 46, *National Bureau of Standards, U.S. Department of Commerce*, Washington D.C.
2. FIPS 81, "Operational modes of DES", Federal Information Processing Standard (FIPS), Publication 81, *National Bureau of Standards, U.S. Department of Commerce*, Washington D.C.
3. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone "Handbook of Applied Cryptography", *CRC Press*, October 1996, 816p.
4. V.Melnyk, T Korkishko "DES Cryptographic Processor". *Report on the research project. Georg-Simon-Ohm-Fachhochschule Nuernberg*, 27 September 1999 – 28 November 1999, 178 p.
5. Korkishko T., Melnyk A. "Cryptographic processor architectures for DES algorithm."// *AFRICON'99, Cape Town, South Africa*, 1999, pp. 175-180.