УДК 681.3
© *Я. В. Решетар*[1], *В. В. Хома*[1], *Я. Р. Совин*[1], *2010*

АРХІТЕКТУРНІ ОСОБЛИВОСТІ ПРОГРАМНО-АПАРАТНОЇ РЕАЛІЗАЦІЇ АЛГОРИТМІВ БЛОКОВОГО ШИФРУВАННЯ ДСТУ ГОСТ 28147:2009 НА БАЗІ ПЛІС

Стаття присвячена програмно-апаратній реалізації алгоритмів блокового шифрування ДСТУ ГОСТ 28147:2009 на базі технології ПЛІС. Акцент зроблено на архітектурних рішеннях з максимальною швидкодією та мінімальною затратою ресурсів. Наведено основні параметри VHDL–проектів дослідження та порівняльні дані реалізацій інших, відомих алгоритмів.

General aspects of GOST 28147 block cipher hardware implementation on FPGA technology are considered in this paper. Main attention was paid to architectural solutions with maximum performance and minimal area. The summary of VHDL-project implementations research comparative to other, more common algorithms, are resulted also.

1. ВСТУП

Впровадження криптографічних засобів захисту інформації поступово стає звичним явищем у багатьох галузях діяльності та побуту людини. Такий стан цілком закономірний, оскільки інформація вже давно стала одним з тих ресурсів, що визначає поступ суспільства, його життя. Водночас, розвиток інформаційних технологій сприяв виходу сфери криптографії за межі застосування виключно в інтересах держави. Сьогодні криптографічні засоби використовуються як на державному рівні, для захисту державної таємниці, документоводства так і на приватному, в діловодстві, сфері особистих інтересів громадянина.

Розвиток науки, передусім в галузі мікроелектроніки дає можливість розширювати базу для створення нових програмних та апаратних криптографічних засобів захисту інформації. Мікропроцесори, контролери, ПЛІС, смарт-карти та інші інтегральні мікросхеми, є основою більшості криптографічних пристроїв: токенів, TPM-модулів, терміналів, пристроїв персональної ідентифікації, тощо. Обмеженість обчислювальних ресурсів представлених платформ актуалізує дослідження ефективної реалізації криптографічних алгоритмів, розгляд їх архітектурних та структурних особливостей, пошук компромісу між затраченими ресурсами та продуктивністю. Особливо гостро визначена проблема стосується вітчизняної галузі, оскільки в Україні чинні власні стандарти криптографічного захисту інформації (зокрема ДСТУ ГОСТ

[1] Національний університет “Львівська політехніка”

28147:2009 визначає алгоритми блокового шифрування), дослідження реалізації яких на сучасних апаратних платформах потребує значно глибшого вивчення.

Питання, які розглядаються в статті є одними з першочергових на шляху до вирішення завдання ефективної реалізації криптографічних програмно-апаратних засобів з підвищеною стійкістю до криптоаналізу.

2. АНАЛІЗ ОСТАННІХ ДОСЛІДЖЕНЬ ТА ПОСТАНОВКА ЗАВДАННЯ

Сучасний стан справ щодо апаратної реалізації криптоалгоритмів в загальному відображений у монографіях [2] і [3]. Зокрема в них наведено порівняльні таблиці, щодо архітектур, показників продуктивності різноманітних реалізацій. Поряд з тим існують публікації вітчизняних науковців присвячені даній проблематиці [4].

Варто відзначити, що аналіз зазначених праць вказує на значний інтерес наукової громадськості в першу чергу до алгоритмів НІСТ (США), зокрема DES і AES. Крім того, заявлені показники продуктивності щодо криптографічної обробки інформації, які сягають десятків Гбіт/с, часто стосуються найсучаснішої елементної бази, вартість якої значно перевищує середньостатистичні очікування криптографічного застосування. Таким чином, мало дослідженими залишаються питання пов’язані з реалізацією алгоритмів ДСТУ ГОСТ 28147:2009 на реконфігураційних платформах, ефективним застосуванням прийнятих для алгоритмів шифрування архітектурних рішень.

З огляду на невирішені частини проблеми, у даній статті ставляться такі цілі:

– розглянути можливості ефективного застосування відомих архітектур реалізації до алгоритмів ДСТУ ГОСТ 28147:2009;

– провести синтез та дослідження шифрувального пристрою в найбільш компактній архітектурі;

– дослідити можливість максимально продуктивної реалізації на базі бюджетних платформ;

– здійснити порівняння отриманих результатів з реалізаціями інших алгоритмів.

3. СТРУКТУРА АЛГОРИТМУ БЛОКОВОГО ШИФРУВАННЯ ТА СТРАТЕГІЇ ЇЇ РЕАЛІЗАЦІЇ

Алгоритм шифрування ДСТУ ГОСТ 28147:2009 оперує 64-бітними блоками і ключем розміром 256 біт. Як і в більшості алгоритмів блокового шифрування використовується ітеративна структура, з 32-х раун-

дів [1]. Вхідними даними для кожного циклу є результат попереднього раунду, що завантажується в регістри N1 та N2 і відповідний раундовий ключ, частина основного ключа. Криптосхема раунду, що реалізує алгоритм шифрування в режимі простої заміни зображена на рис. 1.

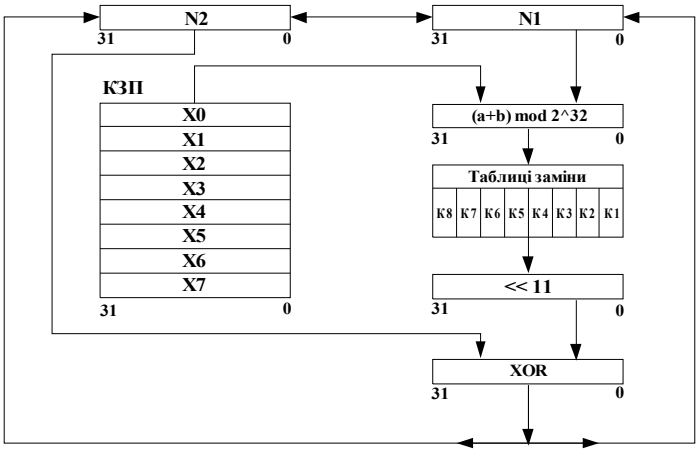

Рис. 1. Криптосхема алгоритму шифрування ДСТУ ГОСТ 28147:2009

Основними компонентами шифру є блок ключового запам'ятовуючого пристрою (КЗП), арифметичний суматор за модулем 2^{32}, 8 таблиць замін К8-К1 розміром 4х4 біти кожна, регістри циклічного зсуву та порозрядного виключного АБО (XOR). КЗП є блоком пам'яті, розбитим на 8 частин по 32 біти. Кожна частина, раундовий ключ, вибирається відповідно до номера раунду та режиму роботи (шифрування/дешифрування). В кінці кожного раунду вміст регістру N1 переноситься в регістр N2, а безпосередній результат циклу завантажується в N1. В останньому раунді вміст регістра N1 зберігається, натомість результат заноситься в N2. Результат криптографічного перетворення зчитується з регістрів N2 і N1, після чого звільняється місце для наступного блоку повідомлення.

Сучасні реконфігураційні платформи, такі як ПЛІС, містять велику кількість логічних та арифметичних вентилів, таблиць відповідності, регістрів необхідних для реалізації основних компонентів алгоритму. Паралельність обчислень на рівні ітерацій досягається завдяки застосуванню декількох стратегій побудови архітектури шифру [2, с.58].

Найпростіша *ітеративна* архітектура (рис. 2, а) реалізує ядро алгоритму шифрування, тобто комбінаційну схему раунду. Схема спрацьовує за тактовим сигналом, приймаючи сигнали з виходу на вхід. Такий підхід дозволяє заощадити ресурси ПЛІС, однак вимагає більше часу на обробку блоку даних, пропорційно кількості раундів в алгоритмі. Наявність ресурсів дає можливість продублювати схему в необхідній кількості, досягаючи повної кількості раундів, що реалізують весь алгоритм. Щоправда така архітектура *розгорнутого циклу* (loop unrolling) в багатьох випадках є малоефективною, оскільки комбінаційна схема формує значну затримку поширення сигналу (рис. 2, б).

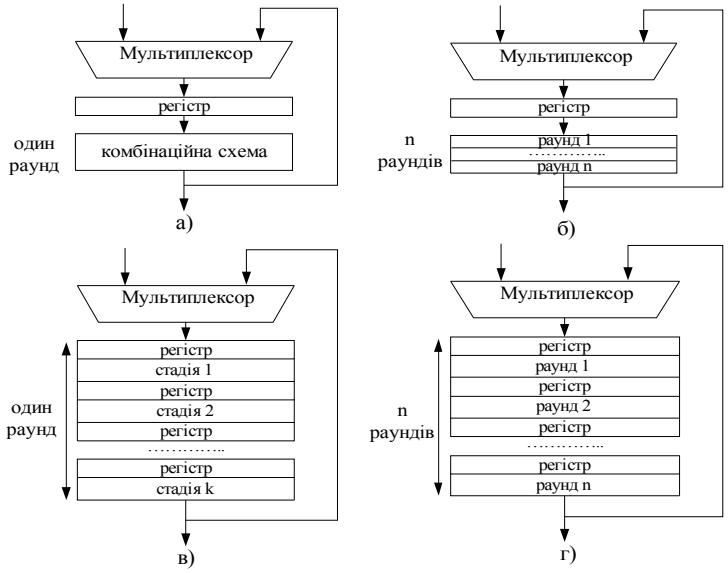

Рис. 2. Базові архітектури блокових алгоритмів шифрування:
а) ітеративна б) розгорнутого циклу
в) внутрішня конвеєрна г) зовнішня конвеєрна

В значній мірі виправити зазначений недолік дає змогу *конвеєрна* (pipeline) архітектура, шляхом розміщення проміжних регістрів між схемами, що відповідають одному раунду. Скорочення шляху поширення сигналу покращує продуктивність шифрувального пристрою загалом. Таким чином, після кожного тактового сигналу відповідні регістри зберігають проміжні (раундові) результати алгоритму шифрування, останній з яких відповідає кінцевому результату криптоперетворення.

Окрім *зовнішньої* (outer-round pipelining) конвеєризації (рис. 2, г), можливе застосування подібної архітектури і до *внутрішніх* (inner-round pipelining) компонентів схеми раунду (рис. 2, в). Виходячи з ресурсів конкретної платформи, комбінування внутрішніх і зовнішніх по відношенню до схеми проміжних станів дозволяє досягнути максимальної продуктивності [3, с.251].

4. ІТЕРАТИВНА РЕАЛІЗАЦІЯ АРХІТЕКТУРИ ШИФРАТОРА

Ітеративна структура алгоритму ДСТУ ГОСТ 28147:2009 дає можливість реалізувати шифрування за 32 такти, використовуючи ресурси комбінаційної схеми одного раунду. Як вже відзначалося, подібна архітектура дозволяє заощадити обчислювальні ресурси, в системах де не вимагається високої продуктивності. Зокрема таке рішення, є ефективним для побудови криптографічного процесора, де значна кількість загальнодоступних обчислювальних ресурсів використовується процесорним ядром, периферією, тощо.

Розглянемо детальніше основні компоненти алгоритму, з точки зору реалізації на базі ПЛІС:

Регістри N1 і N2. Відповідають внутрішній структурі ПЛІС (таблиці відповідності, тригери).

КЗП. Блок пам'яті, що зберігає один або декілька секретних ключів криптографічного перетворення. Функціонування блоку ОЗП (Distributed RAM) організовується на базі таблиць відповідності, крім того більшість сучасних ПЛІС містять спеціальні високопродуктивні блоки пам'яті BRAM (Block RAM).

Суматор за модулем 2^{32}. Найпрацемісткіший компонент алгоритму, що реалізується у вигляді комбінаційної схеми на базі загальнодоступних ресурсів ПЛІС: таблиць відповідності, мультиплексорів, вентилів арифметичної логіки.

Таблиці заміни можуть бути виражені через логічні функції на основі таблиць відповідності або як блоки ПЗП на основі Distributed/Block RAM.

Регістр циклічного зсуву не використовує жодних додаткових ресурсів, оскільки є всього лиш відповідним з'єднанням сигналів на вході і на виході.

Регістр XOR, як і інші булеві функції ефективно реалізується на базі таблиць відповідності та вентилів арифметичної логіки.

Для подальшого дослідження реалізації ітеративної архітектури алгоритму, за допомогою програмного середовища ISE WebPACK 10.1 було створено проект на мові опису апаратних компонентів VHDL. В якості базової платформи вибрано бюджетний пристрій Spartan–3A

(XC3S400A), фірми Xilinx, що поєднує помірну вартість з достатньою кількістю обчислювальних ресурсів загального призначення (400 000 еквівалентних вентилів).

Проект реалізує шифрувальний пристрій в режимі простої заміни, з наступним функціональним призначенням сигналів (рис. 3):

- шина вхідних даних (блоку) повідомлення ***plain_text***;
- шина, що визначає ключ перетворення ***key_addr***;
- тактуючий сигнал ***clk***;
- сигнал скиду ***rst***;
- сигнал, що визначає режим роботи шифратора (шифрування/дешифрування) ***cipher_mode***;
- шина вихідних даних (блоку) повідомлення ***cipher_text***.

Компоненти алгоритму (див. рис. 1) реалізовані у вигляді структурних VHDL-блоків. Поряд з тим заявлена платформа XC3S400A містить 20 блоків ОЗП, що дозволяє зекономити ресурси ПЛІС на реалізації КЗП.

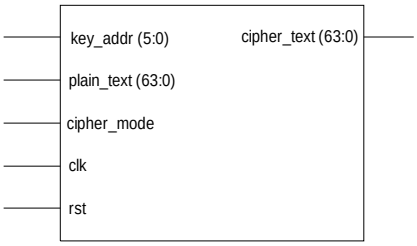

Рис. 3. Інтерфейс схеми шифратора ДСТУ ГОСТ 28147:2009

В даному проекті був використаний готовий компонент RAMB16BWE_S36, що задіює 1 блок ОЗП, сконфігурований для зберігання 512 слів довжиною 32 біти. Дана конфігурація була вибрана не випадково, оскільки в кожному раунді алгоритму використовується саме 32 бітна частина ключа перетворення [1]. Вибір відповідного раундового ключа здійснюється схемою управління, що реалізована в основній архітектурі проекту.

За кожним тактуючим імпульсом ***clk***, відбувається переприсвоєння вхідних сигналів. Однак пріоритетним є сигнал скиду ***rst***, при утриманні якого схема залишається в неробочому стані. Одразу після скидання сигналу ***rst*** відбувається переприсвоєння адреси ключа ***key_addr***, режиму роботи ***cipher_mode,*** а також обнулення лічильника ***round***.

При досягненні лічильником ***round*** значення 31, відбувається автоматичне завантаження в регістри N1 і N2 наступного блоку вхідних

даних. Сигнал ***key_addr*** визначає один з 64 ключів, які можна проініціалізувати в 18 Кбіт блокового ОЗП, а змінна ***round*** здійснює вибір відповідного раундового ключа в межах ключа перетворення.

Результати роботи реалізованого проекту можна спостерігати на діаграмі поведінкової симуляції (рис. 4). Для тесту були використані наступні секретні параметри:

Таблиця заміни K1 — (13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7);
Таблиця заміни K2 — (4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1);
Таблиця заміни K3 — (12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11);
Таблиця заміни K4 — (2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9);
Таблиця заміни K5 — (7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15);
Таблиця заміни K6 — (10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8);
Таблиця заміни K7 — (15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10);
Таблиця заміни K8 — (14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7);
Ключ 256‘h000DC9860001105D025B7224292FDE0402D11D39000CE DE401880FA80000876F.

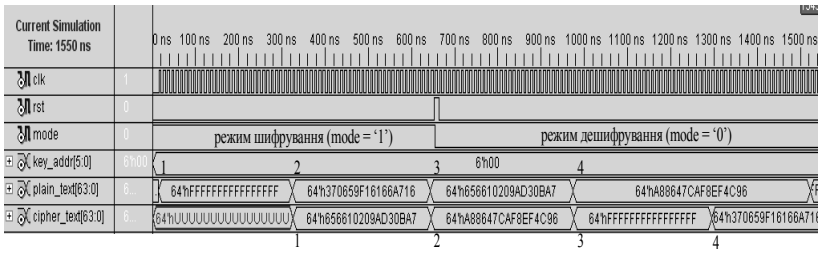

Рис. 4. Симуляція роботи шифратора ДСТУ ГОСТ 28147:2009 (режим простої заміни)

Показники продуктивності і використання ресурсів зведені в таблиці. Тривалість періоду 10 нс визначається максимальною затримкою комбінаційної схеми ядра (раунду) алгоритму. Основним компонентом, що обмежує максимальну продуктивність ітеративної архітектури є 32-ох бітний арифметичний суматор, затримка якого після програмної оптимізації складає близько 7 нс. З цієї причини неефективним виглядає застосування внутрішньої конвеєризації до даної архітектури, оскільки час проходження одного раунду таким чином, складатиме 14 нс.

Таблиця 1

Порівняльна характеристика реалізацій алгоритмів блокового шифрування

Алгоритм	К-ть раундів	Архі-тектура	Ресурси ПЛІС	Плат-форма	Період, нс (макс. част., МГц)	Прод.
ДСТУ ГОСТ 28147:2009 (проста заміна)	32	ітеративна	91 slice (2 %) 172 LUT (2 %) 1 BRAM (5 %)	Spartan-3A	9,682 (103,3)	206,6 Мбіт/с
ДСТУ ГОСТ 28147:2009 (гамування зі зворотнім зв'язком)	32	ітеративна	145 slice (4 %) 243 LUT (3 %) 1 BRAM (5 %)	Spartan-3A	9,342 (107)	214 Мбіт/с
ДСТУ ГОСТ 28147:2009 (проста заміна)	32	зовнішня конвеєрна	2307 slice (64 %) 3088 LUT (43 %) 1 BRAM (5 %)	Spartan-3A	7,715 (129,6)	8,29 Гбіт/с
DES [5]	16	конвеєрна	5036 LUT (49 %)	Virtex-2	4,219 (237)	15,1 Гбіт/с
AES–128 [6]	10	конвеєрна	17,425 slice (43 %)	Spartan-3	5,099 (196,1)	25,1 Гбіт/с

Алгоритм шифрування в режимі гамування зі зворотнім зв'язком [1], має кращу криптографічну стійкість, оскільки нівелює відповідність вихідних блоків до однакових блоків відкритого тексту. Для реалізації цього режиму в ітеративній архітектурі, можна використати попередній проект, тому що алгоритми дешифрування в обох режимах практично співпадають. Основна відмінність, що веде за собою використання додаткових ресурсів, полягає в застосуванні початкового ініціалізуючого вектора – синхропосилки та мультиплексорної схеми для визначення вхідних даних кожного раунду (див. табл.).

Варто зазначити, що реалізація суматора $\bmod 2^{32}$ виконана на базі елементів ***full adder*** [2, с.90], тобто для обчислення значення старшого біту необхідна затримка для обчислення переносу з попереднього розряду. Інші підходи до реалізації суматора можуть дещо покращити показник затримки, зокрема можна взяти до уваги особливість, що на входи таблиць замін подаються 4-ох бітні значення результату додавання.

Паралельне обчислення кожної з тетрад (без врахування переносу з молодших тетрад), в подальшому вимагає корекції результату. Для випадку відсутності переносу з молодшої тетради результат залишається вірним, якщо ж перенос мав місце можливі два варіанти. Особливість цих варіантів полягає в тому, що наявність переносу тотожна додаванню 1 до вже отриманого результату, тобто циклічному зсуву комірки таблиці заміни на 1 позицію вправо. Для випадку значення відмінного від ‘‘0b1111’’ цей перенос таким чином погашається всередині тетради, циклічним зсувом результату. У випадку результату ‘‘0b1111’’ одним лише циклічним зсувом результату перенос не погашається, тому необхідно врахувати його перехід на наступну тетраду.

На рис. 5 зображено такий блок 4-ох розрядного суматора з вузлом заміни. Зокрема він складається з 4-ох бітного напівсуматора, розширеної таблиці заміни 32x4 для врахування корекції, блоку логічних вентилів для попередження випадку ‘‘0b1111’’. Враховуючи можливість визначення випадку ‘‘0b1111’’ (сигнал D0) паралельно з визначенням результату тетради, затримка останньої таблиці заміни (ESB7) визначається послідовністю 6 пар елементів AND і OR (див. рис. 5). Для блоків першої і останньої тетради логічні вентилі для попередження випадку ‘‘0b1111’’ непотрібні, оскільки G0=‘0’, а G8 не використовується.

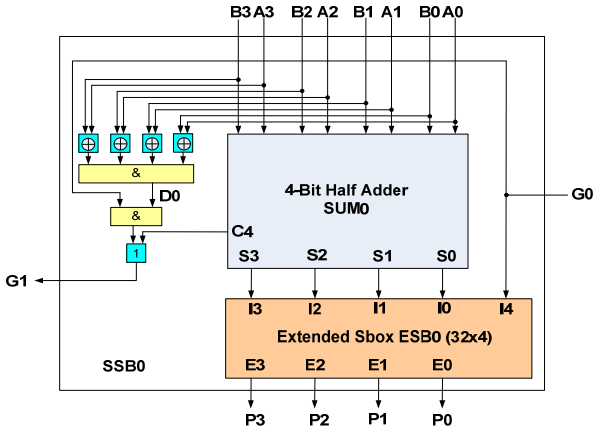

Рис. 5. Блок 4-ох розрядного суматора з вузлом заміни

Продуктивність визначених блоків в значній мірі залежить від безпосереднього розміщення і трасування їх компонентів, крім того використання обчислювальних ресурсів зростає приблизно вдвічі.

5. ЗОВНІШНЯ КОНВЕЄРНА АРХІТЕКТУРА ШИФРАТОРА

В окремих застосуваннях ставляться досить високі вимоги до швидкодії криптографічних перетворень. У цьому випадку продуктивності ітеративної архітектури може бути недостатньо. Максимальну швидкодію роботи шифрувального пристрою можна досягнути розміщенням схеми кожного раунду на одній платформі, дані в яких обробляються одночасно.

Для синтезу зовнішньої конвеєрної архітектури, були використані 32 компоненти (ядра) описані вище. Після кожного тактуючого імпульсу ***clk,*** сигнали з виходів компонентів подаються на входи наступних. Крім того, під час процедури ініціалізації ключ заноситься у 8 проміжних регістрів, дані з яких в подальшому паралельно використовуються для кожного структурного компоненту. Як видно з таблиці, такий підхід поряд з програмною оптимізацією дозволяє підвищити швидкодію до 129,6 МГц, отримуючи на виході шифрувального пристрою, приблизно кожні 7,715 нс черговий блок вихідного повідомлення.

Результати роботи шифрувального пристрою в конвеєрному режимі можна спостерігати на діаграмі (рис. 6).

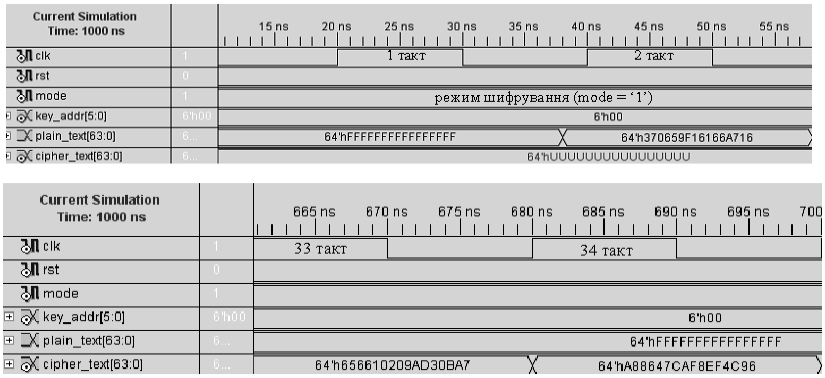

Рис. 6. Симуляція конвеєрної роботи шифратора ДСТУ ГОСТ 28147:2009 (режим простої заміни)

6. ВИСНОВКИ

Розширення сфери застосування криптографічних засобів захисту інформації вимагає ефективної реалізації алгоритмів на сучасних апаратних платформах. У даній статті розглядаються архітектурні особливості реалізації ДСТУ ГОСТ 28147:2009 на базі ПЛІС технологій.

Максимальну продуктивність 8,29 Гбіт/с було досягнуто для конвеєрної архітектури, поряд з тим навіть компактна ітеративна структура всього в 91 slice платформи Spartan-3A забезпечує швидкодію 200 Мбіт/с, достатню для більшості застосувань.

Наявність 32-х розрядного суматора в структурі алгоритмів ДСТУ ГОСТ 28147:2009 спричиняє основну затримку поширення сигналу, що збільшує період роботи та обмежує максимальну продуктивність реалізації в порівнянні з іншими алгоритмами. В окремих випадках можна досягти скорочення затримки застосувавши розширення таблиць замін, однак використання ресурсів при цьому зростає принаймні вдвічі.

У представлених в порівняльній таблиці алгоритмах DES і AES основними компонентами є прості з точки зору апаратної реалізації компоненти заміни, перестановки, суматори за модулем 2, а також менша кількість раундів, що й зумовлює їх перевагу в швидкодії. Поряд з тим, ДСТУ ГОСТ 28147:2009 відзначається простотою реалізації схеми планування раундових ключів.

Як засвідчує практика, ефективна реалізація алгоритму блокового шифрування не гарантує його захищеності. Тому перспективним залишається дослідження проблеми вразливостей через побічні канали витоку інформації, зокрема і реалізацій представлених у статті.

1. Системы обработки информации. Защита криптографическая. Алгоритмы криптографического преобразования: ДСТУ ГОСТ 28147:2009. – [Чинний від 2009-02-01]. – К.: Держспоживстандарт України, 2008. – 28 с. – (Національний стандарт України) 2. Cryptographic Algorithms on Reconfigurable Hardware / Rodriguez-Henriquez, F., Saqib, N. A., Díaz Pérez, A., Koc, C.K. – New York: Springer Science+Business Media, 2007. – 362 p. – ISBN: 978-0-387-33883-5. 3. Cryptographic Engineering / [editor Koc, C.K]. – New York: Springer Science+Business Media, 2009. – 522 p. – ISBN: 978-0-387-71816-3. 4. Баркалов А .А. Реализация алгоритма шифрования DES на базе FPGA / Баркалов А. А., Красичков А. А., Кузьменко В.О. // Обчислювальна техніка та автоматизація: наукові праці ДНТУ. – Донецьк: ДонНТУ, 2009. – № 16(147). – С. 116–120. 5. Pasham V. Trimberger S. High-Speed DES and Triple DES Encryptor-Decryptor [Електронний ресурс] / Application Note: Virtex-E Family and Virtex-II Series (XAPP270), Xilinx, 2001. – Режим доступу: http://www.xilinx.com/support/ documentation/application_notes/xapp 270.pdf. 6. Good T., Benaissa M. AES on FPGA from the Fastest to the Smallest / LNCS: Cryptographic Hardware and Embedded Systems – CHES 2005. – Berlin: Springer, 2005. – Vol. 3659. – P. 427–440.