УДК 004.056.5:004.421.5

В. Широчин[1], доктор техн. наук; І. Васильцов[2], канд. техн. наук; Б. Карпінський[2]

[1]Національний технічний університет Україні „КПІ”.
[2]Тернопільський державний економічний університет.

МЕТОДИ ПРОТИДІЇ АТАЦІ ДИФЕРЕНЦІЙНОГО КРИПТОАНАЛІЗУ ЗБОЇВ НА ПОТОКОВІ ШИФРИ, БАЗОВАНІ НА LFSR, ТА СХЕМА ДЕТЕКТУВАННЯ ПОМИЛОК

Запропоновано методи протидії атакам на основі апаратних збоїв на потокові шифри, що використовують як базові компоненти LFSR. Проектування потокових шифрів з використанням запропонованих методів протидії є ефективнішим за традиційну техніку дублювання апаратних ресурсів. Для практичного застосування спроектована схема детектування збоїв. Додатково проаналізовано апаратні затрати на реалізацію запропонованих методів протидії атакам на основі апаратних збоїв. Запропоновано та реалізовано компоненту, що алгоритмічно є еквівалентна базовому LFSR, із вбудованими мірами протидії атакам на основі внесення апаратних збоїв ата аналізу енергоспоживання без додаткових апаратних затрат на реалізацію мір протидії для кожної з атак окремо.

V.Shyrochin; I.Vasyltsov; B.Karpinskyy

COUNTERMEASURE METHODS TO DIFFERENT FAULT ATTACK ON THE STREAM CIPHERS BASED ON THE LSFR AND ERROR DETECTION CIRCUIT

Proposed countermeasure methods to attacks based on the fault insertion on the LFSR based stream ciphers. Development of stream ciphers based on the proposed countermeasure methods is more effectively then traditional full doubling of hardware resources. For practical use developed fault detection circuit. In additional the needed hardware resources to realize proposed methods of countermeasure was analyzed. The component which analogue to basic LSFR with countermeasure methods to fault insertion attack and power attack was proposed and realized. On the component not used additional hardware resources to realize countermeasures for each separately attack.

1. Вступ

Через високі вимоги продуктивності обробки криптографічних даних, апаратні реалізації відомих криптоалгоритмів широко використовуються в сучасних системах захисту інформації. Разом з тим, розвиваються нові методи криптоаналізу таких апаратно реалізованих криптографічних алгоритмів – так звані Side-Channel Attacks (SCA – атаки побічних каналів витоку інформації). Розвинулося багато методів для їх проведення, проте найвідомішими є: аналіз енергоспоживання (Power Analysis), часовий аналіз (Timing Analysis), атака на основі внесення апаратних збоїв (Fault Insertion Attacks – FIA) або диференційна криптоатака збоїв (Differential Fault Attacks - DFA), атака на основі аналізу електромагнітного випромінювання (Electro Magnetic Emissions) [1-6].

Модель атаки даних методів передбачає безпосередній доступ криптоаналітика до аналізованого пристрою. Щодо цього, криптоаналітик може виконувати наступні дії:

- задавати будь-які дані на вхід криптографічного пристрою;
- отримувати та аналізувати дані на виході аналізованого пристрою;
- вимірювати та аналізувати часові залежності виконуваних дій криптопристроєм;
- вносити збої в роботу пристрою під час виконання ним криптографічних перетворень;
- вимірювати енергоспоживання пристрою під час його роботи;
- вимірювати електромагнітні випромінювання під час роботи пристрою.

Внаслідок аналізу інформації про особливості роботи криптографічного пристрою, зловмисник може отримати інформацію про значення секретного ключа, тобто скомпрометувати таку криптографічну систему.

Особливого інтересу серед множини атак побічних каналів набула атака на основі внесення апаратних збоїв. Вона базується на основі аналізу реакції криптопристрою на внесені збої під час виконання ним криптографічних перетворень чи під час переходу криптографічного пристрою із одного внутрішнього стану в інший. На сьогоднішній час розвинулося багато підходів до застосування даного виду атаки на криптографічні пристрої, що реалізують найрізноманітніші алгоритми криптографічного захисту.

Достатньо новим підходом є використання основних принципів атак на основі внесення апаратних збоїв FIA та DFA на потокові шифри. Деякі ідеї щодо можливостей її застосування до потокових шифрів були наведені у [6]. Також в [7] показані нові підходи до можливостей успішної реалізації даної атаки на базові потокові шифри та їх основні компоненти:

- на компоненту регістру зсуву (Linear Feedback Shift Register -LFSR), в тому числі з адаптивними зворотними зв’язками полінома, що використовується як базова компонента при побудові сучасних потокових шифрів;
- на проріджуючий генератор (shrinking generator), що є стійким до інших відомих атак за умови використання в ньому базових компонент LFSR розмірністю не менше як 64 біти [8].

Теоретичні результати, отримані в [7, 8], показують зростання ризиків практичної реалізації атак помилок на сучасні потокові шифри.

В даній статті запропоновано методи протидії атакам на основі внесення апаратних збоїв для потокових шифрів, побудованих з використанням LFSR, та реалізацію схеми детектування збоїв.

2. Алгоритм атаки на основі внесення апаратних збоїв на базову компоненту LFSR

Модель збою

Атака на основі внесення апаратних збоїв базується на дослідженні реакції криптопристрою на внесені збої в роботі під час виконання ним криптографічних процедур. На даний час розвинулося багато методів внесення збоїв в цифрові пристрої. Всі вони базуються на використанні зовнішніх дестабілізуючих факторів: електромагнітного випромінювання, температури, нестабільного енергоживлення, десинхронізації роботи пристрою, тощо. В загальному випадку внесені збої можна поділити як за об’єктом атаки, так і за типом. За об’єктом збої поділяють на:

- збої в комірки пам’яті (де, наприклад, може зберігатися ключ шифрування або підключ);
- збої, направлені на внутрішню систему керування (зміна внутрішніх станів криптопристрою);
- збої на шини даних та команд.

За типом збої можуть бути поділені на:

- встановлення в логічний стан ‘1’;
- встановлення в логічний стан ‘0’;

– інверсія бітів ('1'→'0' чи '0'→'1').

Базова структура реконфігуративного LFSR

Основною частиною потокових шифрів є генератор псевдовипадкових послідовностей. Є багато різних методів (алгоритмів) генерування псевдовипадкових послідовностей для вирішення задачі потокового шифрування, проте більшість сучасних високопродуктивних потокових шифрів базуються на використанні регістрів зсуву, а саме регістрів зсуву з лінійними зворотними зв'язками – LFSR. На рисунку 1 наведена базова структура реконфігуративного LFSR, тобто з можливістю зміни поліномів зворотних зв'язків.

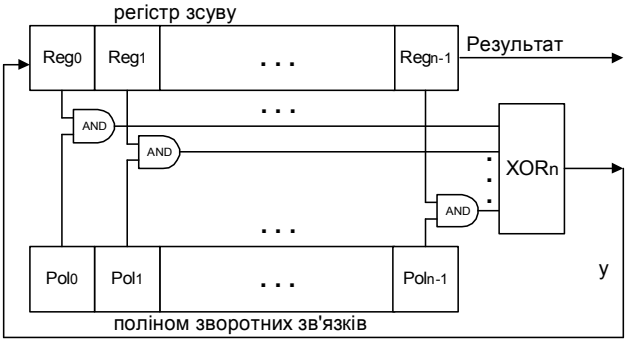


Рисунок 1 - Структура базового реконфігуративного LFSR/

З рисунка 1, Reg_0, Reg_1, …, Reg_{n-1} є комірками n- бітного регістру зсуву; Pol_0, Pol_1, …, Pol_{n-1} є комірками n- бітного полінома зворотних зв'язків; XOR_n – операція виключного АБО для n вхідних бітів; y - генерований новий псевдовипадковий біт.

Використання реконфігуративного LFSR дозволяє користувачу легко змінювати поліноми зворотних зв'язків. Більше того, така реконфігурація може бути виконана „на льоту", відповідно зміна поліномів зворотних зв'язків базової компоненти підвищує криптографічну стійкість до аналітичних атак потокових шифрів.

DFA атака на LFSR

В [7] наведено алгоритм диференційної криптоатаки збоїв на базовий LFSR. В дослідженнях модель криптоаналізу передбачала можливість внесення криптоаналітиком збоїв у визначену комірку регістру зсуву. Збій мав характер встановлення в логічний стан '1'. Для реалізації такої атаки криптоаналітик повинен володіти наступною інформацією про атакований пристрій:

1. розмірність регістру зсуву n та відповідний поліном зворотних зв'язків $\{P_0,...P_{n-1}\}$.
2. l - бітів вихідної гами y, де l - максимальна відстань між двома сусідніми зворотними зв'язками.

Зазвичай, така інформація може бути легко доступна криптоаналітику. Хід атаки полягає у виконанні наступних кроків:

1. Організовується цикл по $i = 0\text{K}\ l-1$.
2. Під час кожної ітерації циклу по i організовується цикл по $j = 0\text{K}\ m-1$, де m - кількість зворотних зв'язків полінома.
3. Кожної ітерації внутрішнього циклу проводимо наступні дії:

– якщо значення j - тої комірки регістру вже відоме, то пропустити всі операції кроку 3 для даної комірки регістру;
– "встановлюємо" (вносимо збій) значення j - тої комірки регістру (відповідна комірка регістру, значення якої йде у зворотний зв'язок - впливає на обчислення наступного біту гами);
– отримуємо $y'(i)$ - біт гами зі збоєм; $y'(i) = y(i)$;

– порівнюємо значення біту гами у нормальному режимі роботи пристрою $y(i)$ із отриманим значенням режиму роботи зі збоями $y'(i)$. Якщо відповідні значення рівні, то відповідне значення комірки регістру $x(j)$ було рівне '1', інакше $x(j)=$'0';
– проводимо перезапуск пристрою, що аналізується (LFSR).

Для успішного проведення атаки криптоаналітик повинен мати вихідну гаму, довжиною щонайменше l біт, а також провести n збоїв.

Також в [7] наведено алгоритм диференційної криптоатаки на основі внесення апаратних збоїв на проріджуючий генератор, що складається з двох базових LFSR. В загальному випадку, якщо в проріджуючому генераторі використовують LFSR розмірністю не менше як 64 біти, то такий ГПВЧ, а відповідно і побудований на його основі потоковий шифр є стійкими до відомих на даний час атак [8]. Проте, алгоритм DFA атаки з [7] показує можливість зламу конструкції за розумні затрати часу та обчислювальних ресурсів.

Саме тому, розробка методів протидії до DFA для потокових шифрів, що використовують базові LFSR, є важливою та актуальною задачею.

3. Методи протидії DFA для потокових шифрів, що базуються на LFSR

В [2, 9] наведено загальні методи детектування атак на основі внесення апаратних збоїв. В найпростішому випадку, це може бути виконано шляхом виконання двічі однієї і тієї ж операції та подальшої перевірки правильності в критично важливих частинах (або в цілому) алгоритму шифрування чи цифрового підпису. Тому звичайне дублювання обчислювальних ресурсів є одним із можливих варіантів забезпечення стійкості криптографічного пристрою до атак на основі внесення апаратних збоїв.

Разом з тим, при використанні методу дублювання апаратних ресурсів відповідно зростають вдвічі затрати обладнання, а також, в деяких випадках, це може понизити продуктивність проекту (при імплементації на програмовані логічні інтегральні схеми (ПЛІС). Саме тому метод протидії на основі повного дублювання апаратних ресурсів можна вважати неефективним шляхом вирішення даної проблеми.

В даній роботі авторами запропоновано альтернативні повному дублюванню методи протидії атакам апаратних збоїв, які можна використати для базових компонент сучасних потокових шифрів – LFSR.

Схема детектування збоїв

Перед тим, як підходити до висвітлення методів протидії атакам на основі внесення апаратних збоїв та їх аналізу, необхідно вирішити задачу перевірки результату.

Виходячи з того, що вихід одного базового LFSR рівний одному біту (як і більшості сучасних потокових шифрів), система перевірки правильності результату повинна приймати на вхід два бітові значення (теоретично однакові), а на вихід повинна подавати один із результатів. У випадку розбіжності (нерівності) вхідних даних система повинна додатково подавати сигнал попередження назовні, оскільки це означає, що відбулися неправильні обчислення результату або мало місце зловмисне спотворення даних з метою проведення атаки на основі внесення апаратних збоїв.

Виходячи з поставленої задачі, в [11] запропоновано у якості системи детектування впливу (чи помилкового обчислення результату) використовувати результат операції XOR над двома бітовими результатами еквівалентних схем, де перший результат – отриманий від пристрою потокового шифрування (генератора ПВЧ), а другий – результат, отриманий від схеми протидії атакам. Загальна схема детектування збоїв наведена на рисунку 2.

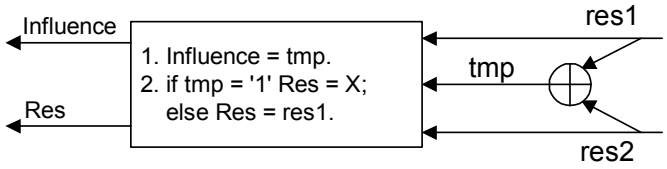


Рисунок 2 - Загальна схема детектування збоїв.

На рисунку використано наступні позначення: res1 – результат роботи першого LFSR (чи то ГПВЧ1), res2 – результат роботи другого LFSR(схеми протидії атакам), Influence – вихід, який приймає значення ‘1’ у випадку, коли значення r1 та r2 різні, Res – вихідне значення результату.

Значення res1 та res2 при нормальному режимі роботи пристрою повинні бути однакові, оскільки це є результати роботи еквівалентних систем (ГПВЧ чи то LFSR) з використанням однакових стартових умов (наприклад, початкового заповнення та полінома зворотних зв’язків). r1 та r2 можуть різнитися тільки у випадках:

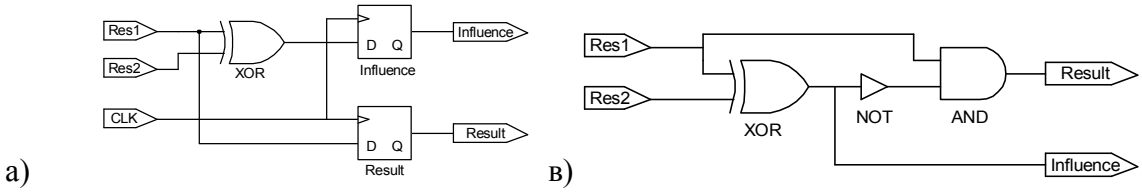
- виникнення внутрішнього збою в роботі пристрою внаслідок неякісної реалізації чи зовнішніх дестабілізуючих факторів, що малоймовірно у випадку дотримання умов використання пристрою;
- впливу зовнішніх дестабілізуючих факторів з метою проведення FIA.

Така конструкція перевірки правильності є достатньо простою та ефективною, оскільки вимагає при апаратній реалізації незначну кількість апаратних ресурсів та дозволяє детектувати появу невідповідних результатів.

Спроектовано vhdl модель даної системи, проведено функціональну симуляцію, а також проведено синтез в середовищі автоматизованого проектування Synplify Pro. Можливі логічні схеми системи детектування збоїв наведено на рисунку 3.

Випадок повного дублювання апаратних ресурсів LFSR

Як було згадано вище, дублювання передбачає виконання процедури обчислення результату двічі. Тобто можна імплементувати поряд два еквівалентні LFSR, ініціалізувати їх однаковими значеннями, а вихід перевіряти запропонованою компонентою детектування збоїв. Таку компоненту LFSR, підвищеної стійкості до FIA, назвемо *LFSRfull* – з повним дублюванням (резервуванням) обчислювальних ресурсів.



а) в)

Рисунок 3 - Логічна схема системи детектування збоїв
(де варіант а) базований на D-тригерах та варіант в) – лише на логічних елементах).

Звичайно, може використовуватися також мультидублювання, тобто трьох і більше базових компонент з подальшим мажоритарним вибором результату, проте такий підхід вважається неефективним, оскільки потребує в три і більше разів більше апаратних ресурсів для апаратної реалізації.

Загальна конструкція LFSR підвищеної стійкості FIA з повним дублюванням обчислювальних ресурсів наведена на рисунку 4, де основними складовими є:

- LFSR1 базова компонента регістру зсуву з лінійними зворотними зв’язками.
- LFSR’1 – компонента еквівалентного LFSR;
- Компонента системи перевірки результату.

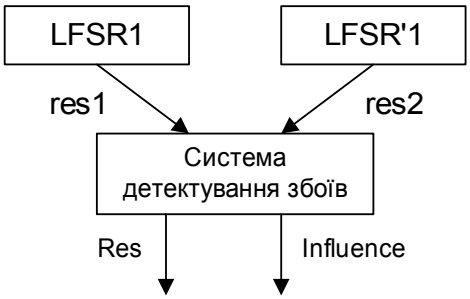


Рисунок 4 - Конструкція LFSR підвищеної стійкості до FIA на „високому" рівні.

На рисунку 4 використані ті ж позначення, що і на рисунку 2.

Як вже згадано раніше, основним недоліком такого підходу є збільшення в два рази необхідних апаратних затрат при імплементації функціонально-еквівалентного пристрою, не враховуючи незначні затрати на імплементацію системи детектування збоїв, а також можливе деяке зниження продуктивності у порівнянні з продуктивністю імплементованого базового LFSR.

Випадок часткового дублювання апаратних ресурсів LFSR.

Разом з тим, авторами запропоновано конструкцію LFSR з частковим дублюванням ресурсів, де дублюється тільки безпосередньо основна частина LFSR – регістр зсуву. Дану компоненту можна також назвати як „LFSR з частковим дублюванням". Зовнішній вигляд конструкції LFSR з частковим дублюванням наведено на рисунку 5.

Виходячи із наведених у [7] атак, на основі внесення апаратних збоїв на компоненту базового LFSR генератора, дана конструкція також буде володіти властивістю підвищеної стійкості до FIA, оскільки збої вносилися виключно в комірки регістрів зсуву. Також, виходячи з аналізу даних, наведених в [3, 9], збої успішно можуть бути реалізовані на комірки пам'яті (різного типу), тобто в тригери, з яких складається безпосередньо регістр зсуву.

Саме тому дублювання тільки компоненти регістру зсуву, в базовому LFSR-генераторі, дозволить підвищити криптографічну стійкість до FIA, оскільки ускладнить проведення збоїв. Для сигналізування про проведення атаки збоїв додатково використовується запропонована система детектування збоїв.

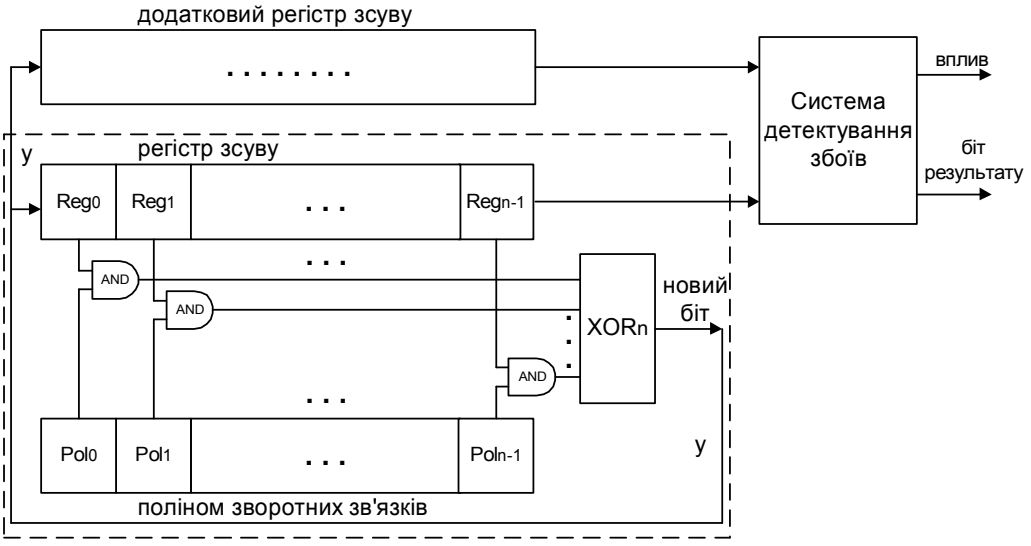


Рисунок 5 - Конструкція LFSR підвищеної стійкості до FIA з частковим дублювання ресурсів.

На рисунку 5 використані ті ж позначення, що і на рисунку 1.

Використання згортки значення регістру зсуву як метод протидії атакам на основі внесення апаратних збоїв

Можна певним чином відійти від теорії дублювання, а для перевірки функціонування пристрою використати іншу методику. Оскільки взято, апріорі, що збої будуть вноситися в комірки пам'яті, то виникає необхідність детектування впливів (збоїв) саме в регістрі зсуву базового LFSR. Це можна виконати шляхом проведення певних додаткових обчислень над даними – вмістом регістру зсуву та відповідною передачею отриманого результату другій стороні інформаційного обміну, що виконує шифрування/дешифрування повідомлення. Прикладом таких додаткових обчислень може слугувати обчислення значення „образу" або ж згортки вмісту регістру зсуву.

Нехай n - у нас розмірність LFSR. Виділимо, для прикладу, додатковий регістр розмірністю k - біт (де k кратне n) і його початкове заповнення, рівне першим k - бітам регістру зсуву. В такому випадку можна виконати операцію XOR почергово над вмістом даного регістру на всіх наступних k - біт регістру зсуву. В результаті отримається k - бітове значення, яке буде в тій чи іншій мірі „містити" відомості чи відображати значення всіх бітів регістру зсуву на певний момент часу або, іншими словами, значення згортки регістру зсуву певної розмірності. При зміні будь-якого біту основного регістру зсуву буде наявна відповідна зміна в значенні обчисленої згортки.

Звичайно, можливі варіанти, коли в регістрі зсуву відбулася множина змін і при обчисленні значення згортки ми отримаємо так звану колізію.

Значення згортки регістру зсуву, передане в чистому вигляді „назовні", може дати криптоаналітику додаткові відомості про вміст регістру зсуву, що зумовить зменшення криптографічної стійкості пристрою в цілому. Тому над значенням згортки необхідно провести деякі додаткові операції. Наприклад, можна виконати операцію XOR над значенням обчисленої згортки і деякою секретною константою. Звичайно, значення секретної константи повинно бути узгоджено, як і секретний ключ, і на боці відправника і на боці приймача.

У такому випадку значення згортки може відображати стан регістру зсуву в певний момент його роботи, що може використовуватися для перевірки правильності його функціонування або ж інформувати сторони інформаційного обміну про можливість проведення атаки на основі апаратних збоїв. Загальна структура схеми обчислення згортки для регістру зсуву наведена на рисунку 6.

Звичайно, обчислене значення згортки n-бітного регістру зсуву буде актуальне тільки для n-тактів його роботи, поки вміст регістру зсуву повністю не оновиться, а після того значення згортки необхідно обчислити знову.

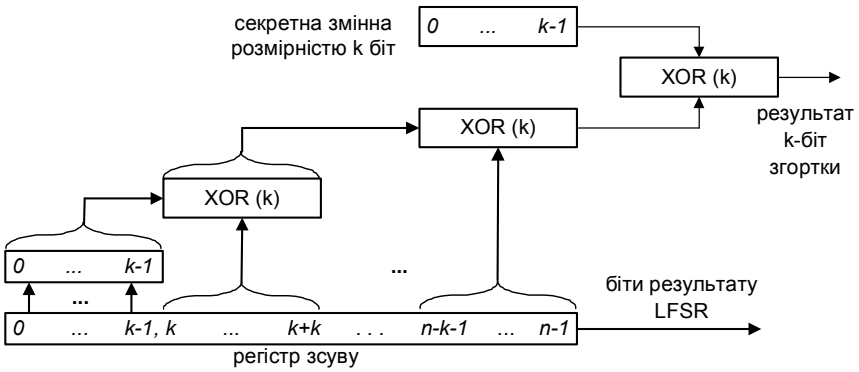


Рисунок 6 - Загальна структура схеми обчислення згортки.

4. Порівняльний аналіз апаратних реалізацій запропонованих методів протидії

Було реалізовано компоненти регістрів зсуву з лінійними зворотними зв'язками як *.vhdl описи з повним (*LFSRfull*), частковим (*LFSRpartial*) дублюванням обчислювальних ресурсів, компоненту *turning*, що обчислює значення k- бітної згортки над регістром розмірністю n та VDHL модель системи детектування збоїв, з

використанням інтегрованої системи автоматизованого проектування Active HDL. Проведено функціональну симуляцію даних компонент.

Треба зауважити, що проектовані компоненти *LFSRfull*, *LFSRpartial* володіють адаптивними зворотними зв'язками. Такі LFSR, при імплементації, займають більше апаратних ресурсів та володіють меншою продуктивністю, у порівнянні із LFSR, що мають фіксовані поліноми зворотних зв'язків. Проте з їх використанням можна отримати більше неповторюваних вихідних послідовностей на одній алгоритмічній базі, а отже, такі компоненти володіють вищою криптографічною стійкістю.

При апаратній реалізації проектованих компонент спочатку досліджено параметри імплементації пропонованої системи детектування збоїв. Виявлено, що при імплементації на мікросхему EP1K10TC100-1 серії ACEX 1K система детектування збоїв використовує тільки 2 логічні елементи ПЛІС, що містить 2 тригери та компоненту XOR. Оскільки конструкція такої компоненти є максимально простою, то компонента працює на максимально можливій для даної мікросхеми продуктивності 250 МГц.

Також, проведено експериментальні дослідження з апаратної реалізації компонент *LFSRfull* та *LFSRpartial* з різними довжинами регістрів зсуву n з використанням інтегрованої системи QUARTUS II на мікросхему EP1K10TC100-1 серії ACEX 1K. Основна мета проведених досліджень імплементації компонент з різним значенням n – дослідження параметрів імплементації та продуктивних параметрів даних компонент у порівнянні із базовою LFSR компонентою, параметри імплементації якої досліджені в [10].

Апаратна реалізація пристрою на ПЛІС характеризується такими основними параметрами: використання зайнятої площі, використання тригерів, продуктивність. З допомогою формули (1) та (2) відносно [10] можна обчислити необхідну кількість площі (базових логічних комірок мікросхеми - AR_{LFSR}) та необхідну кількість тригерів (f / f_{LFSR}) при імплементації на мікросхему EP1K10-1 серії ACEX 1K базового LFSR (відповідно з адаптивними зворотними зв'язками):

$$AR_{LFSR} = 3 \cdot n + 4; \quad (1)$$

$$f / f_{LFSR} = 2 \cdot n + 1. \quad (2)$$

Продуктивність базової LFSR компоненти для різних значень розмірності імплементованого регістру зсуву можна отримати із табличних результатів, що наведені у [10].

На рисунку 7 наведено результати продуктивності базової компоненти LFSR-генератора різної розмірності (з 8-ми до 128 біт) на мікросхемі EP1K10-1 серії ACEX1K, що отримані в ході проведення експериментів.

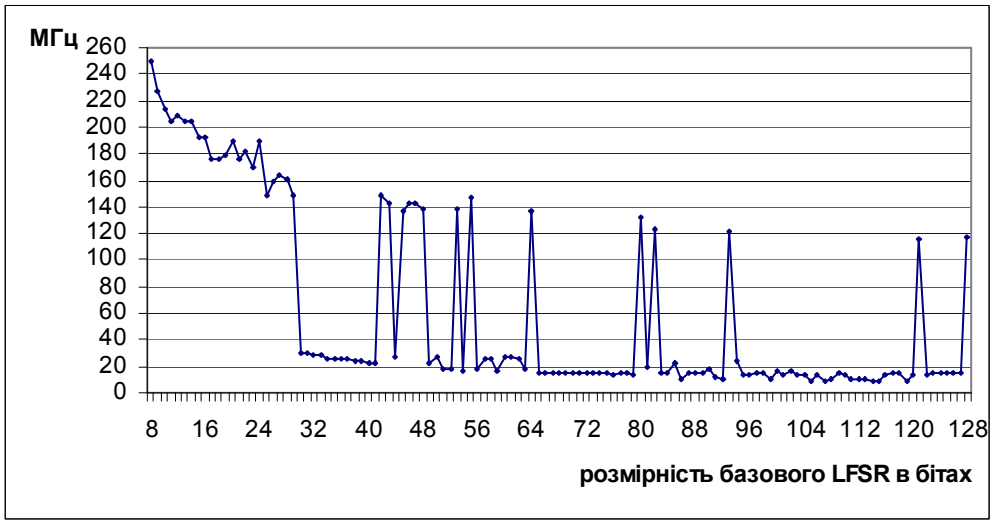


Рисунок 7 - Результати продуктивності базової компоненти LFSR- генератора різної розмірності.

Треба зауважити, що базова компонента LFSR- генератора забезпечує реконфігурацію поліному зворотних зв'язків, що, в свою чергу, підвищує лінійну складність вихідної послідовності, але також значно впливає на продуктивні характеристики та параметри імплементації ГПВЧ при його реалізації на ПЛІС.

Були проведені експериментальні дослідження з імплементації компоненти *LFSRfull* з різними значеннями розмірності регістру зсуву (від 8-ми до 128 біт) на мікросхему EP1K10-1 серії ACEX1K. В таблиці 1 наведено фрагмент узагальнених параметрів імплементації компоненти LFSRfull з регістрами зсуву від 8-ми до 17-ти біт.

Таблиця 1 - Параметри імплементації компоненти *LFSRfull* з різними значеннями розмірності регістру

Розмірність LFSR, біт	Використана площа, ЛЕ	Кількість входів/ виходів	Використаних тригерів	Частота, МГц
8	50	14	34	227,27
9	56	15	38	222,22
10	62	16	42	192,31
11	66	17	46	192,31
12	72	18	50	192,31
13	78	19	54	196,08
14	82	20	58	188,68
15	88	21	62	188,68
16	94	22	66	181,82
17	98	23	70	163,93

З допомогою формул (3) та (4) можна обчислили необхідну кількість базових логічних елементів та необхідну кількість тригерів при імплементації компоненти *LFSRfull* в залежності від розмірності регістру зсуву n відповідно.

$$AR_{LFSRfull} = 2 \cdot (3 \cdot n + 4) = 6 \cdot n + 8; \quad (3)$$

$$f / f_{LFSRfull} = 2 \cdot (2 \cdot n + 1) = 4 \cdot n + 2. \quad (4)$$

Оскільки система перевірки результату достатньо проста при реалізації апаратно, основні затрати апаратури йдуть на імплементації двох однакових LFSR-генераторів. Тому апаратні затрати компоненти *LFSRfull* у порівнянні із компонентою базового LFSR в два рази більші. На рисунку 8 наведено отримані дані продуктивності компоненти *LFSRfull* при різних значеннях розмірності регістру зсуву.

В таблиці 2 наведено фрагмент узагальнених параметрів імплементації компоненти LFSRpartial з регістрами зсуву від 8-ми до 17-ти біт.

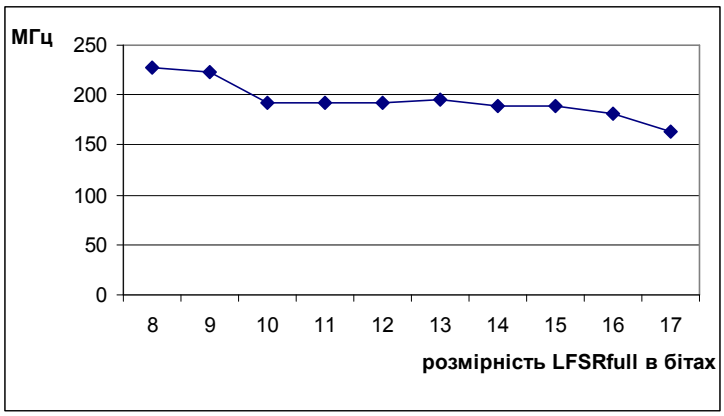


Рисунок 8 - Дані продуктивності компоненти LFSRfull при різних значеннях розмірності регістру зсуву.

Таблиця 2 - Параметри імплементації компоненти LFSRpartial з різними значеннями розмірності регістру.

Розмірність LFSR, біт	Використана площа, ЛЕ	Кількість входів/ виходів	Використаних тригерів	Частота, МГц
8	37	14	28	238,1
9	41	15	31	222,22
10	45	16	34	196,08
11	48	17	37	196,08
12	52	18	40	192,31
13	56	19	43	200
14	59	20	46	200
15	63	21	49	192,31
16	67	22	52	172,41
17	71	23	55	175,44

З допомогою формул (5) та (6) можна обчислити необхідну кількість базових логічних елементів та тригерів відповідно при імплементації компоненти *LFSRpartial* в залежності від розмірності регістру зсуву:

$$AR_{LFSRpartial} = 4 \cdot n + 5\,; \tag{5}$$

$$f / f_{LFSRpartil} = 3 \cdot n + 4\,. \tag{6}$$

На рисунку 9 наведено результати продуктивності компоненти *LFSRpartial* різної розмірності (з 8-ми до 128 біт) на мікросхемі EP1K10-1 серії ACEX1K, що отримані в ході проведення експериментів.

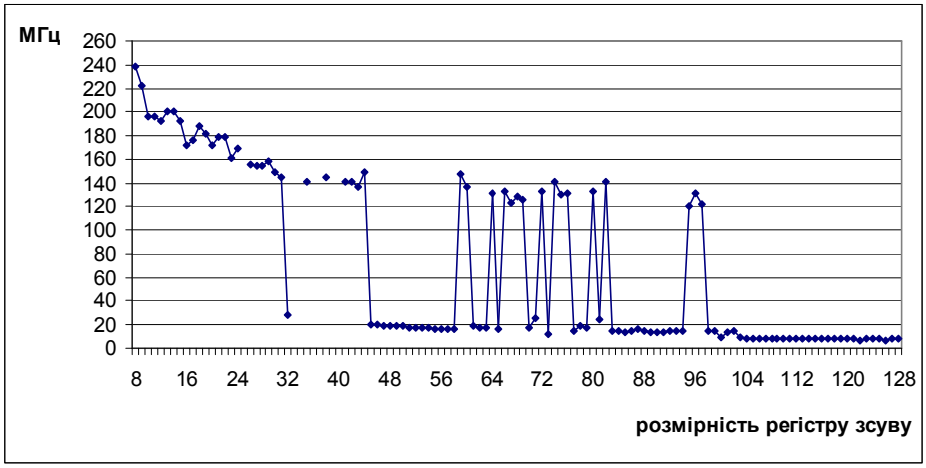


Рисунок 9 - Дані продуктивності компоненти *LFSRpartial* при різних значеннях розмірності регістру зсуву.

Звичайно, компоненту *turning* треба розглядати як з точки зору розмірності регістру зсуву - n, так і з точки зору розмірності регістру згортки - k. Звичайно, відносно загальної схеми, що наведена на рисунку 6, компонента *turning*, для початку повинна завантажити значення секретної змінної, для цього в компоненті виділено вхідний сигнал *load_key*. Для обчислення безпосередньо значення згортки над регістром даних, значення якого подаються на вхідну шину *data_reg*, виділено сигнал керування *Enable_turn.*

У таблиці 3 наведено результати імплементації даної компоненти відносно розмірності регістру зсуву та розміру регістру згортки.

Таблиця 3 - Параметри імплементації компоненти turning з різними значеннями розмірності регістру згортки та розмірності регістру зсуву LFSR

Розмірність LFSR, біт	Розмірність згортки, біт	Використана площа, ЛЕ	Кількість входів/виходів	Використаних тригерів	Частота, МГц
12	3	11	19	7	250
12	4	14	20	9	250
12	6	14	22	13	250
16	4	14	24	9	250
16	8	18	28	17	250
18	3	14	25	7	250
18	6	20	28	13	250
18	9	20	31	19	250
24	3	14	31	7	250
24	4	18	32	9	250
24	6	20	34	13	250
24	8	26	36	17	250
24	12	26	40	25	250
28	4	18	36	9	250
28	7	23	39	15	250
28	14	30	46	29	250

З аналізу отриманих результатів зрозуміло, що спроектована компонента *turning* має максимально просту для імплементації структуру, що забезпечує їй максимально можливу продуктивність (250 МГц), при імплементації на дану мікросхему. Отримані результати також дозволили вивести формули, за якими можна підрахувати необхідну кількість вхідних/вихідних сигналів (7) та тригерів (8) відповідно.

$$I/O_{turning} = n\_reg + k\_turn + 4, \tag{7}$$

де n_reg - розмірність регістру зсуву, а k_turn - розмірність згортки.

$$f/f_{LFSRturning} = 2 \cdot k\_turn + 1. \tag{8}$$

Треба зауважити, що значення згортки повинно обчислюватися щоразу, як тільки-но вміст регістру зсуву повністю обновиться. Кількість тактів на повне обновлення вмісту регістру зсуву в базового LFSR рівна його розмірності. Звичайно, це дасть деяку додаткову затримку в роботі компоненти, проте, водночас, підвищить рівень криптографічної стійкості компоненти до атак на основі апаратних збоїв.

Водночас, виходячи із аналізу міри протидії до атак на основі аналізу енергоспоживання [12], на рівні базової компоненти LFSR, та міри протидії атакам на основі внесення апаратних збоїв з використанням часткового дублювання ресурсів LFSR (рисунок 5), ми сумістимо дані види протидії в одній компоненті без внесення додаткових апаратних затрат на реалізацію протидії для кожної з атак. Ідея полягає в тому, щоб використовувати додаткових регістр і для протидії атаці на основі аналізу енергоспоживання (зберігаючи в ньому інверсні значення даних бітів основного регістру LFSR), а також використовувати його для додаткової перевірки (ускладнюючи внесення апаратних збоїв) роботи LFSR в цілому. Проте для реалізації даної ідеї необхідно змінити алгоритм функціонування компоненти детектування збоїв (рисунок 3) таким чином, щоб вона сигналізувала про збої у випадку виникнення різниці значень основного та додаткового регістрів зсуву. Загальна схема компоненти LFSR із вбудованими мірами протидії до атаки аналізу енергоспоживання та на основі внесення апаратних збоїв наведена на рисунку 10.

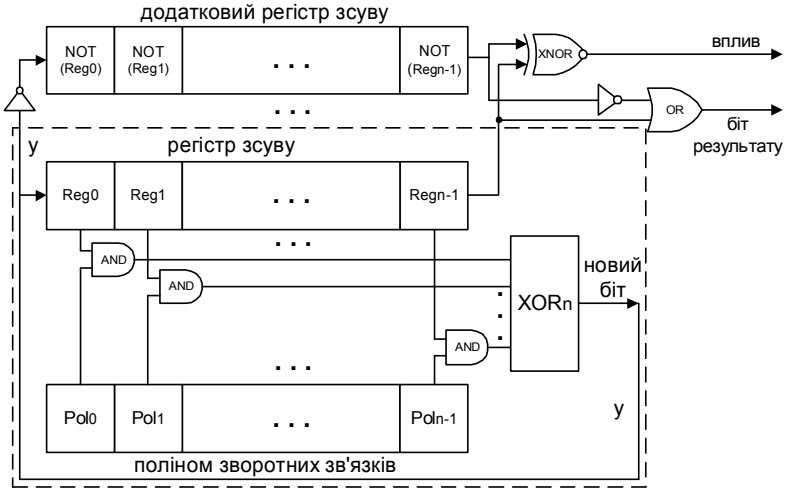


Рисунок 10 – Базовий LFSR з елементами протидії атакам на основі внесення апаратних збоїв та аналізу енергоспоживання (компонента LFSR_FP).

Така компонента (LFSR_FP) була розроблена у вигляді VHDL моделі та реалізована апаратно на мікросхему EP1K10 серії ACEX 1K. Порівняння реалізацій показало, що дана компонента володіє еквівалентними характеристиками продуктивності та реалізації, як і компонента LFSRpartial, а отже, при розрахунку апаратних затрат компоненти LFSR_FP можна використовувати формули (5) та (6).

Отримані формули дозволять розробнику спеціалізованого апаратного забезпечення підвищеної стійкості до атак на основі внесення апаратних збоїв вибрати оптимальну розмірність даної компоненти, виходячи із розмірності базового LFSR та розміру регістру згортки.

Висновки

У даній роботі показано, що традиційний підхід повного дублювання як метод протидії DFA не є ефективним вирішенням проблеми через високі затрати апаратури та втрати продуктивності. Натомість, запропоновано метод часткового дублювання ресурсів LFSR та метод обчислення значення згортки регістру зсуву. Запропоновані методи можуть використовуватися для проектування потокових шифрів, що використовують LFSR як базову компоненту підвищеної стійкості до атак на основі FIA.

Для аналізу запропонованих методів засобами автоматизованого проектування Active HDL спроектовані VHDL моделі компонент *LFSRfull* (повне дублювання ресурсів LFSR), *LFSRpartial* (часткове дублювання ресурсів LFSR), *turning* – для обчислення значення згортки розмірність k біт до n - бітного регістру та систему детектування збоїв. З допомогою пакету Quartus II проведено імплементацію їх на мікросхему EP1K10TC100-1 серії ACEX 1K.

Були проведені експериментальні дослідження імплементації проектованих компонент щодо необхідної кількості апаратних ресурсів та продуктивності в залежності від їх розмірності (розміру регістру зсуву в LFSR та розміру регістру згортки). Аналіз результатів імплементації показав, що метод часткового дублювання апаратних ресурсів LFSR дозволяє досягнути на 50% менших апаратних затрат та вищої продуктивності, ніж метод традиційного повного дублювання. Також як ефективний метод протидії даним атакам може використовуватися метод обчислення згортки від значення регістру зсуву.

Запропоновано та розроблено компоненту LFSR_FP, що алгоритмічно еквівалентна базовому LFSR. В даній компоненті реалізовано міри протидії на основі внесення апаратних збоїв та аналізу енергоспоживання. При аналізі реалізацій даної

компоненти виявлено, що вона володіє еквівалентними параметрами реалізації та продуктивності, що і компонента LFSRpartial.

Отримані результати дозволяють проектувати потокові шифри, стійкі до атак, на основі внесення апаратних збоїв з використанням базових компонент LFSR. Це також дозволяє спростити процес проектування і отримати оптимальне рішення проектування за розумний час.

Література

1. J. Kelsey, B. Schneier, D. Wagner, and C. Hall, "Side Channel Cryptanalysis of Product Ciphers," in Proceedings of ESORICS '98 , Springer- Verlag, September 1998, pp. 97- 110.
2. James Alexander Muir. Techniques of side channel cryptanalysis // University of Waterloo. Dept. of Combinatorics and Optimization, 2001.
3. Sergei P. Skorobogatov, Ross J. Anderson. Optical Fault Induction Attacks // Proceedings of the 4th International Workshop Redwood Shores, CA, USA, August pages 2-12.
4. Eli Biham , Adi Shamir, Differential Fault Analysis of Secret Key Cryptosystems // Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology, p.513-525, August 17-21, 1997.
5. D. Boneh, R. DeMillo, and R. Lipton, "On the importance of checking cryptographic protocols for faults", Proceedings of Eurocrypt, Lecture Notes in Computer Science, Springer-Verlag, LNCS 1233, pp. 37_51, 1997.
6. Jonathan J. Hoch, Adi Shamir, Fault Analysis of Stream Ciphers, Lecture Notes in Computer Science, Volume 3156, Jul 2004, Pages 240 – 253.
7. Карпінський Богдан. Атака апаратних збоїв на проріджуючий генератор псевдовипадкових чисел // Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні – 2005. – № 11. – С. 113–118.
8. Menezes A., Oorshot P., Vanstone S. Handbook of applied cryptography. – N.Y.: CRC Press Inc., 1996.– 816 p.
9. http://www.crypto.ruhr-uni-bochum.de/en_sclounge.html
10. Широчин В.П., Васильцов И.В., Карпинький Б.З. Эволюционный алгоритм определения квази-оптимальных параметров каскадного генератора псевдослучых чисел // Управляющие системы и машины, Информационные технологии. Международный научный журнал: №4 (198), июль-август 2005. - С.40-47.
11. Широчин В. П., Васильцов І. В., Карпінський Б.З., Куртяк В.І. DFA Countermeasure Method for LFSR-based Stream Ciphers and Fault Detection Circuit // Матеріали міжнародної конференції "Сучасні проблеми радіоелектроніки, телекомунікацій, комп'ютерної інженерії" TCSET`2006, 28 лютого-4 березня, 2006.- Львів-Славсько, Україна. - С.309-312.
12. Широчин В.П,, Васильцов І.В., Карпінський Б.З., Куртяк В.І.. аналіз енергоспоживання потокових шифрів побудованих на LFSR за допомогою Ваги Хемінга // Захист інформації – 2005.-№3.- С.64-73.

Одержано 22.05.2006 р.