УДК 004.31 DOI: 10.20535/2523-4455.2019.24.1.167007

Апаратний прискорювач операції доказу виконаної роботи в криптовалюті IOTA

Сачов[f] С. О.
e-mail sergiosachov@gmail.com
Короткий[s] Є. В., к.т.н. доц, ORCID 0000-0001-8302-4873
e-mail e.korotkiy@kpi.ua
Кафедра конструювання електронно-обчислювальної апаратури keoa.kpi.ua
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського» kpi.ua
Київ, Україна

***Анотація*—Запропоновано структуру апаратного прискорювача операції доказу виконаної роботи (Proof-of-work, PoW) в криптовалюті IOTA. Описано запропоновану структуру з застосуванням мови Verilog. Розроблений апаратний прискорювач синтезовано в базисі програмованої логіки типу FPGA та інтегровано в систему-на-кристалі для FPGA Cyclone V з вбудованим ARM процесором. Створено Linux-драйвер для використання прискорювача з програм користувача. Проведено оцінку апаратурних витрат та продуктивності розрахунків запропонованого прискорювача у порівнянні з програмною реалізацією та існуючим аналогом.**

Бібл. 14, рис. 6.

Ключові слова — геш-функція; Verilog; обчислювач; програмована логіка; система-на-кристалі; криптовалюта; інтернет речей; апаратний акселератор.

I. Вступ

Одним із ключових викликів для інтернету речей (Internet-of-Things, IoT) є питання безпеки. Не менш важлива наявність ефективних механізмів оплати даних від сенсорів. IoT прилади генерують значні потоки даних, які можна і необхідно монетизувати.

Відповідно до прогнозу компанії Ericsson, у 2021 році до мережі Internet буде підключено порядку 28 мільярдів IoT та мобільних пристроїв [1], що будуть активно обмінюватися даними, у т. ч. виконувати платежі.

Прикладами подібних застосунків є автоматичний продаж даних від сенсорів безпілотних автомобілів про затори, аварії, погодні умови чи стан доріг, платежі за пріоритет руху у заторах, продаж мобільними пристроями даних про користувача, наприклад, даних про маршрути покупців у торгівельних центрах тощо.

Відомо, що платіжна система VISA у моменти пікового навантаження може обробляти порядку 2000 транзакцій за секунду [2]. Станом на 2018 рік подібні транзакції виконуються переважно людьми, однак значне розповсюдження IoT пристроїв та міжмашинних платежів у недалекому майбутньому може створити значне навантаження на існуючі платіжні системи, обумовлюючи збільшення комісій та тривалості транзакцій.

Для вирішення проблеми міжмашинної взаємодії було розроблено криптовалюту IOTA, що реалізує захищені аутентифіковані канали передачі даних між IoT пристроями та мікроплатежі без комісій [3-5].

Класичні криптовалюти, на зразок Bitcoin та Ethereum, використовують блокчейн для збереження інформації про платежі. Блокчейн є послідовністю блоків, що зберігають транзакції платежів. Наприклад, програмне забезпечення вузлів мережі Bitcoin кожні 10 хвилин (в середньому) формує новий блок з емітованими користувачами платіжними транзакціями. Подібний підхід обмежує масштабованість платіжної системи і знижує її продуктивність. Як наслідок, Bitcoin та Ethereum дозволяють опрацьовувати лише 7 та 15 транзакцій за секунду [6, 7] відповідно. У порівнянні з платіжною системою VISA це дуже незначні цифри.

В основі ж криптовалюти IOTA лежить так званий Tangle, що має структуру орієнтованого ациклічного графа [3]. Використання подібної структури дозволяє додавати транзакції до Tangle паралельно (різні транзакції додаються одночасно до різних вершин графу), а не послідовно блок за блоком, як у класичному блокчейні, що збільшує масштабованість і продуктивність системи. Криптовалюта IOTA реалізована таким чином, що Tangle має властивості розподіленості та незмінності, що дозволяє зберігати в ньому інформацію про платіжні транзакції.

Щоб додати транзакцію до Tangle, необхідно перевірити та підтвердити дві інші транзакції, а також провести обчислювально складну операцію доказу виконаної роботи Proof-of-Work (PoW) з метою захисту IOTA мережі від спам-атак та атак Сівілли [7].

Для виконання операції PoW необхідно ітеративно розраховувати геш-функцію Curl від IOTA транзакції та змінювати поле Nonce до одержання необхідної кількості послідовних нулів у старших розрядах результату геш-функції.

З метою підвищення рівня криптографічної захищеності розробники IOTA запропонували нову геш-функцію Curl в трійковій системі числення, застосування якої збільшує кількість можливих комбінацій і робить алгоритм більш стійким до атак методом прямого перебору. Для підписування вхідних транзакцій в криптовалюті IOTA застосовують одноразові цифрові підписи Вінтерніца (Winternitz One-time Signature) [8], стійкі до атак з використанням квантових комп'ютерів.

На жаль, програмні реалізації Curl функціонують надзвичайно повільно. Операція PoW з застосуванням програмної реалізації Curl в IoT пристроях може тривати до 50 хвилин, що катастрофічно сповільнює відправлення IOTA транзакцій. Як наслідок, актуальною стає задача розробки апаратного прискорювача операції PoW криптовалюти IOTA для зменшення тривалості генерації платіжних транзакцій на IoT платформах.

II. Аналіз відомих напрацювань

Основні положення криптовалюти IOTA викладені в [3-5].

Вихідний код програмного забезпечення IOTA знаходиться у відкритому доступі [9]. Наявні програмні реалізації бібліотек IOTA мовами C, C++, JS, Python, Rust, Go, Java.

До квітня 2018 року була відсутня інформація про апаратні реалізації геш-функції CURL та про апаратні прискорювачі операції PoW для криптовалюти IOTA.

У квітні 2018 року у відкритому доступі з'явився модуль апаратного прискорювача операції PoW криптовалюти IOTA, описаний мовою VHDL [10]. Зазначене рішення не може бути використане для розрахунку геш-функції Curl від довільного блоку даних, а для його функціонування необхідний окремий мікрокомп'ютер Raspberry Pi.

В [11] дослідники з MIT виявили можливість за певних умов отримувати колізії в геш-функції Curl. Для унеможливлення експлуатації цієї вразливості у хакерських атаках розробники IOTA прийняли рішення використовувати адаптовану геш-функцію Keccak для підписування IOTA транзакцій та генерації адрес. Однак геш-функція Curl досі застосовується в операції PoW.

III. Постановка задачі

Метою пропонованої роботи є розробка та оцінка характеристик апаратного прискорювача операції PoW криптовалюти IOTA для вбудованих систем на основі мікросхем програмованої логіки Intel FPGA Cyclone V з вбудованим апаратним ARM процесором.

Для досягнення мети необхідно вирішити наступні задачі:

- опрацювати теоретичні основи криптовалюти IOTA та геш-функції Curl;
- розробити апаратну реалізацію геш-функції Curl;
- створити апаратний прискорювач операції PoW криптовалюти IOTA;
- інтегрувати розроблений прискорювач в систему-на-кристалі (СнК) для FPGA Cyclone V з вбудованим ARM процесором та забезпечити обмін даними з PoW прискорювачем з використанням прямого доступу до пам'яті (DMA);
- розробити Linux-драйвер для використання апаратного прискорювача PoW з програм користувача;
- виконати оцінку продуктивності роботи та апаратурних витрат запропонованого апаратного прискорювача операції PoW.

IV. Теоретичні основи криптовалюти IOTA

Криптовалюта IOTA призначена для виконання платежів та захищеного і розподіленого обміну повідомленнями між пристроями інтернету речей.

Відсутність комісії за проведення платіжних транзакцій дозволяє здійснювати мікроплатежі величиною у долі центу, що відкриває можливість застосування нових бізнес-моделей, приклади яких наведені у вступі.

Кожна транзакція містить поле повідомлення. Бібліотека Masked Authenticated Messaging дозволяє шифрувати вміст поля повідомлень та створювати захищені канали зв'язку з використанням аутентифікації, що відповідають шаблону проектування "публікація-підписка".

Для реалізації подібної платіжної системи з можливістю захищеного обміну даними необхідно забезпечити незмінність сформованих транзакцій в розподіленій мережі вузлів зберігання та обробки.

Транзакції криптовалюти IOTA є вершинами орієнтованого ациклічного графу під назвою Tangle, джерелом якого є генезис-транзакція, сформована під час першого запуску мережі вузлів IOTA. Листами графу є непідтверджені (не перевірені) транзакції. Кожна вершина в Tangle, окрім джерела, вказує на транзакції, перевірені під час її формування. Частина графу Tangle зображена на рис. 1.

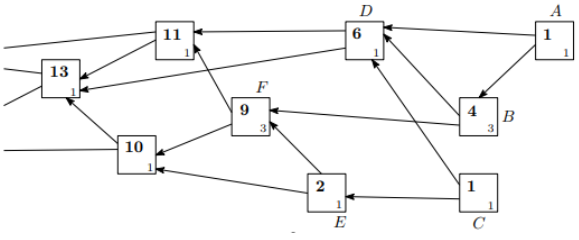

Рис. 1. Частина орієнтованого ациклічного графу Tangle.

Копії графу Tangle зберігаються та синхронізуються в розподіленій peer-to-peer мережі вузлів криптовалюти IOTA, які називають нодами. IOTA-ноди виконують задачу додавання нових транзакцій до Tangle. Для додавання нової транзакції необхідно обрати і перевірити на коректність вмісту дві непідтверджені транзакції, що знаходяться у листах графу. Механізм перевірки розглянемо далі. Існує можливість сформувати і підписати транзакції на пристрої користувача та відправити їх до IOTA-ноди для додавання до Tangle.

Введемо необхідні визначення. Всі транзакції в Tangle містяться у множині T. Для транзакцій $x, y \in T$:

1) $y \mapsto x$ означає, що y прямо підтверджує x. В такому випадку x та y на графі Tangle з'єднані дугою, що направлена від y до x;

2) $y \nrightarrow x$ означає, що y непрямо підтверджує x. В такому випадку $\exists z \in T : y \mapsto z \wedge z \rightsquigarrow x$;

3) $y \rightsquigarrow x$ означає, що y прямо або непрямо підтверджує x.

Для листів графу $x \in T$ справедливо $\nexists y \in T : y \rightsquigarrow x$.

Кожна вершина $x \in T$ має власну вагу $w(x)$, що є мірою обчислень, виконаних при формуванні транзакції x. В поточній реалізації IOTA $\forall x \in T,\ w(x) = 1$. В майбутніх версіях програмного забезпечення IOTA $\forall x \in T,\ w(x) = 3^n,\ n \in N$.

Кумулятивна вага H_x для вершини $x \in T$ розраховується за формулою:

$$H_x = w(x) + \sum_{y \in A} w(y) = 1 + |A|, \qquad A = \{z \in T \mid z \rightsquigarrow x\} \tag{1}$$

На рис. 1 власна вага зображена в правому нижньому куті вершини графу, а кумулятивна вага знаходиться в лівому верхньому куті.

Для підвищення криптографічної захищеності IOTA використовує збалансовану трійкову систему числення, де розряди чисел приймають значення -1, 0, 1. Розряд в такій системі називають тріт. Три тріта формують трайт, що може приймати $3^3 = 27$ значень. Для представлення трайтів в IOTA використовують 26 літер A-Z і цифру 9.

Транзакція IOTA включає наступні поля:

Signature Message Fragment (2187 трайтів) — для транзакції з негативним балансом містить першу частину цифрового підпису. Для транзакції з нульовим або позитивним балансом може містити повідомлення. Для транзакції з нульовим балансом, що має таку ж адресу, як інша транзакція з негативним балансом, містить другу частину цифрового підпису;

Address (81 трайт) — адреса транзакції;

Value (27 трайтів) — кількість криптовалюти IOTA, що передається у транзакції. Може приймати нульове значення для передачі повідомлень в полі *Signature Message Fragment* без витрат;

Obsolete Tag (27 трайтів) — використовується для розрахунку коректного значення поля Bundle Hash;

Time Stamp (9 трайтів) — час початку формування транзакції в форматі Unix Timestamp;

Current Index (9 трайтів) — індекс транзакції в колекції Bundle;

Last Index (9 трайтів) — загальна кількість транзакцій в колекції Bundle;

Bundle Hash (81 трайт) — геш-функція Keccak від колекції транзакцій Bundle;

Trunk Hash (81 трайт) та Branch Hash (81 трайт) — геш-функції Curl двох непідтверджених транзакцій, обраних при формуванні даної транзакції;

Tag (27 трайтів) — ідентифікатор, що може бути визначений користувачем;

Attachment Time Stamp (9 трайтів) — час завершення операції PoW в форматі Unix Timestamp;

Attachment Time Stamp Lower Bound (9 трайтів) — зарезервовано для використання в майбутньому;

Attachment Time Stamp Upper Bound (9 трайтів) — зарезервовано для використання в майбутньому;

Nonce (27 трайтів) — поле, що змінюється під час операції PoW.

Таким чином, розмір IOTA транзакції складає 2673 трайтів.

Розглянемо формування деяких полів IOTA транзакції.

Баланс користувача в IOTA визначається унікальним ідентифікатором Seed, що має розмір 81 трайт ($27^{81} \simeq 8.7 \cdot 10^{115}$комбінацій) і є одночасно логіном та паролем. Адреса в IOTA є односторонньою функцією, що реалізована з використанням геш-функції Keccak від Seed і номеру адреси. Користувач зазвичай має кілька адрес.

Транзакції IOTA, що містять адреси, з яких пересилається криптовалюта, підписуються за допомогою одноразових цифрових підписів Вінтерніца. Таємний ключ є односторонньою функцією, що реалізована з використанням геш-функції Keccak від Seed і номеру адреси. Відкритий ключ є значенням адреси. Кошти з певної адреси можна переказати, лише знаючи Seed, на базі якого згенерована адреса. Внаслідок використання одноразових цифрових підписів адресу IOTA можна використати у якості джерела коштів лише один раз.

Для переказу криптовалюти IOTA з однієї адреси на іншу необхідно створити колекцію транзакцій, що має назву Bundle. Сума балансів (полів Value) транзакцій всередині Bundle повинна дорівнювати нулю. Кожна транзакція всередині Bundle має індекс. Індексу 0 відповідає транзакція з адресою отримувача коштів, на яку пересилається криптовалюта IOTA

у кількості, що визначається додатнім значенням поля Value. Далі йдуть пари транзакцій, що описують адреси, з яких перераховуються кошти. Загальний баланс таких адрес не може бути меншим, ніж поле Value транзакції з індексом 0. Перша транзакція кожної пари містить у полі Value від'ємне значення пов'язаного з адресою балансу, а поле Signature Message Fragment містить першу частину цифрового підпису. Друга транзакція кожної пари містить поле Value = 0 і другу частину цифрового підпису. У випадку, якщо загальний баланс транзакцій-джерел коштів перевищує значення поля Value транзакції отримувача коштів, до Bundle додають ще одну транзакцію, що містить нову адресу відправника коштів, на яку буде перераховано залишок (поле Value такої транзакції приймає додатне значення).

Для розрахунку геш-функції від колекції транзакцій Bundle розраховують геш-функцію Keccak від полів Address, Value, Obsolete Tag, Time Stamp, Index та Last Index всіх транзакцій, що входять до Bundle.

Якщо транзакція не використовується для переказу коштів і містить лише повідомлення, Bundle включає одну транзакцію з адресою отримувача повідомлення і Value=0.

Останнім кроком формування Bundle є виконання операції PoW для кожної транзакції всередині Bundle з метою захисту IOTA мережі від спам-атак та атак Сівілли. Операція PoW полягає у знаходженні такого значення поля Nonce, що результат геш-функції Curl від транзакції містить необхідну кількість нульових останніх трітів. Необхідна кількість нульових останніх трітів визначається параметром Minimum Weight Magnitude (MWM) і станом на початок 2019 року дорівнює 14. Необхідне значення Nonce знаходять шляхом перебору, підставляючи в поле Nonce нове значення, розраховуючи геш-функцію Curl і перевіряючи MWM останніх трітів результату на рівність нулю. Операція PoW є найбільш обчислювально складним етапом під час формування транзакції, оскільки програмні реалізації Curl мають низьку продуктивність. На початку 2018 року, на момент старту наших досліджень, не існувало доступних апаратних реалізацій операції PoW для IOTA.

Розглянемо механізм знаходження двох непідтверджених вершин Tangle, що обираються для перевірки під час формування IOTA транзакції. На певній глибині графу Tangle обирається транзакція, що є початком для двох процесів зваженого випадкового блукання за алгоритмом Markov Chain Monte Carlo (MCMC). Транзакції на кожному кроці випадкового блукання перевіряються на цілісність та коректність (нульовий сумарний баланс транзакцій в Bundle, наявність необхідних коштів на обраній адресі-джерелі, відповідність геш-функцій вмісту транзакції, коректність результату PoW). У випадку знаходження транзакції, що не пройшла перевірку, MCMC алгоритм повертається на крок назад і продовжує пошук за іншим напрямом. Транзакція, що є вихідною вершиною для початку алгоритму MCMC в поточній реалізації IOTA, обирається наступним чином. Кожні дві хвилини спеціальні вузли мережі IOTA, що мають назву Координатор, формують транзакції з нульовим вмістом і унікальним порядковим номером, що підписані цифровим підписом Координатора. Такі транзакції називаються Milestones і задають напрямок зростання Tangle. Кожна наступна Milestone прямо або непрямо підтверджує попередню. В майбутньому, при досягненні необхідно великої кількості вузлів в мережі, розробники IOTA планують відмовитися від Координатора. Наразі початком алгоритму MCMC обирають одну із створених Координатором Milestones, номер якої є результатом різниці номеру останньої Milestone і параметру Depth, що приймає значення від 3 до 15. Після відмови від Координатора алгоритм вибору початкового вузла для MCMC буде змінено відповідно до правил, описаних в [3]. Під час випадкового блукання за алгоритмом MCMC імовірність переходу від транзакції x до транзакції y, якщо $y \mapsto x$, розраховується за формулою:

$$P_{xy} = \frac{e^{-\alpha\left(H_x - H_y\right)}}{\sum_{z:z \rightsquigarrow x} e^{-\alpha\left(H_x - H_z\right)}},$$

де H_i — кумулятивна вага транзакції $i \in T$; α — параметр, що визначає ступінь випадковості переходу від x до y і станом на початок 2019 року приймає значення 0.001.

Якщо зловмисник спробує додати до Tangle транзакцію з некоректним вмістом, інші вузли IOTA мережі не будуть підтверджувати зкомпрометовану транзакцію. Таким чином забезпечується незмінність вмісту Tangle.

Станом на січень 2019 року транзакція вважається успішно прийнятою в Tangle, якщо її прямо або непрямо підтверджує Milestone-транзакція, зформована Координатором. Після відмови від Координатора ступінь успішного прийняття транзакції буде визначатися ступенем її підтвердження множиною листів Tangle, обраних за алгоритмом MCMC.

V. Апаратна реалізація геш-функції Curl

Геш-функція Curl побудована на основі sponge-конструкції, що вперше запропонована в геш-функції Keccak [12]. Розрахунок геш-функції з застосуванням sponge-конструкції складається з двох етапів:

1) Відображення вхідного блоку даних у змінну стану State за допомогою операції Absorb.
2) Одержання результату геш-функції із змінної State за допомогою операції Squeeze.

Операції Absorb і Squeeze використовують функцію Transform.

Блок-схеми алгоритмів для реалізації Transform, Absorb і Squeeze геш-функції Curl наведені на рис. 2 та рис. 3. Функція Transform виконує 81 раунд перестановок трітів у змінній State. Розмір змінної State складає 729 трітів. Під час операції Absorb нові дані записуються у молодші 243 тріти змінної State. Під час операції Squeeze нові тріти для формування

результату Curl зчитуються з молодших 243 трітів змінної State. Функція Transform виконує перестановки над всіма трітами змінної State. Значення таблиці індексів (INDEX) і таблиці істинності (TRUTH_TABLE) наведені у вихідному коді реалізації геш-функції Curl мовою C [13].

Аналізуючи програмну реалізацію Curl з [13], можна помітити, що кожен тріт нового значення State залежить від двох трітів попереднього значення State за певною закономірністю:

new_state[0] ← state[0], state[364]

new_state[1] ← state[364], state[728]

new_state[2] ← state[728], state[363]

new_state[3] ← state[363], state[727]

new_state[4] ← state[727], state[362] … і т. д.

З урахуванням зазначеного спостереження запропоновано структуру апаратного обчислювача функції Transform, що наведена на рис. 4. Для представлення трітів в двійковому базисі програмованої логіки використано двобітні числа із забороненою комбінацією 2'b10. Для представлення State використано синхронний по фронту регістр. Оскільки State містить 729 трітів, а для представлення кожного тріта необхідно два біта, для реалізації регістру State знадобилося 1458 тригерів.

Для реалізації операцій Absorb і Squeeze необхідно завантажувати/зчитувати змінну State під керуванням зовнішнього скінченного автомату або мікропроцесора, після чого запускати на виконання операцію Transform, повторюючи ітерації до опрацювання усього вхідного повідомлення у випадку операції Absorb, або до одержання результату геш-функції необхідного розміру у випадку операції Squeeze.

Вихідний код мовою Verilog для реалізації запропонованої структури апаратного обчислювача геш-функції Curl знаходиться у відкритому доступі [14]. Запропонована реалізація в кожному раунді вираховує за один такт нове значення State. Виконання операції Transform потребує 81 такту.

VI. Реалізація апаратного прискорювача операції PoW

На основі апаратної реалізації геш-функції Curl, описаної у попередньому розділі, було створено апаратний прискорювач операції PoW (рис. 5) та СнК на базі ARM процесора з інтегрованим апаратним прискорювачем операції PoW для криптовалюти IOTA (рис. 6).

Принцип функціонування апаратного прискорювача операції PoW наступний. Апаратний прискорювач включає $N \geq 2$ обчислювачів операції PoW, де N є параметром, який можна перевизначати. Один із обчислювачів є головним (Master PoW Calc Unit), а інші обчислювачі є допоміжними (Slave PoW Calc Units).

Для трітів транзакції IOTA, за виключенням поля Nonce, що міститься в кінці транзакції, головний обчислювач виконує операцію Absorb геш-функції Curl. Логіка роботи блоку Next State Logic визначає реалізацію операції Transform, що описана у попередньому розділі (рис. 4). Після завершення операції Absorb проміжний результат, що міститься в регістрі State, записується в регістр Midstate. Далі всі обчислювачі ітеративно завантажують значення регістрів State з регістру Midstate, заповнюють поле Nonce випадковими значеннями, що створюються блоками New Nonce Gen, виконують операцію Transform, по завершенню якої перевіряють на рівність нулю останні MWM трітів результату розрахунку Curl, що є умовою завершення операції PoW.

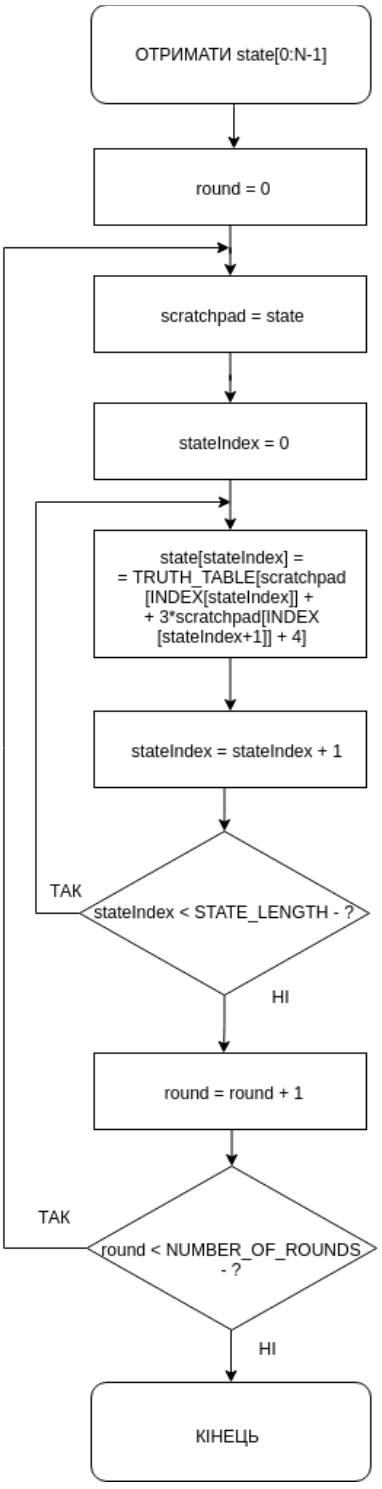

Рис. 2. Блок-схема алгоритму для реалізації операції Transform.

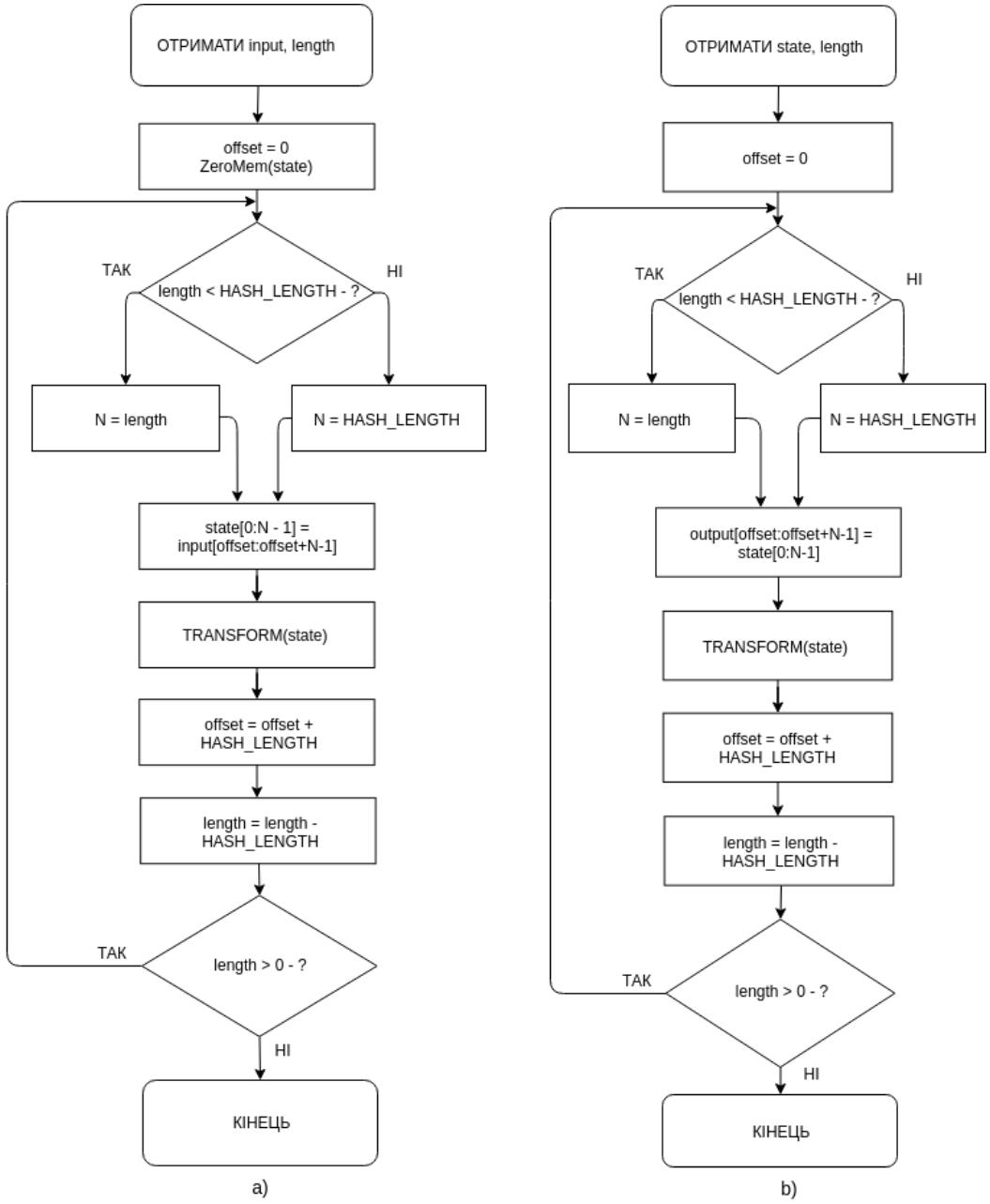

Рис. 3. Блок-схеми алгоритмів для реалізації операцій a) Absorb, b) Squeeze.

Якщо умова завершення PoW не досягнута у жодному з обчислювачів, відбувається перехід до наступної ітерації, в іншому випадку операція PoW завершується і знайдене значення поля Nonce можна зчитати із відповідного обчислювача з використанням мультиплексора Sel Nonce. Перевірка умови завершення операції PoW виконується блоком Check State Logic. Логіка керування в апаратному прискорювачі PoW реалізована з використанням скінченного автомату Control FSM.

В програмній реалізації операції PoW для формування нового значення поля Nonce використано операцію інкременту в трійковій системі числення. Такий підхід є небажаним у апаратній реалізації, оскільки блок трійкового інкременту потребує значних апаратурних витрат. У пропонованій реалізації для створення нового значення Nonce в блоці New Nonce Gen використано генератор псевдовипадкової послідовності на базі регістру зсуву з лінійним зворотним зв'язком. Якщо певні тріти згенерованого Nonce приймають заборонене значення 2’b10, їх вміст замінюється дозволеним значенням 2’b00.

Для використання в програмах користувача апаратний прискорювач операції PoW криптовалюти IOTA інтегровано в СнК на базі ARM процесора. Структуру системи наведено на рис. 6. Систему реалізовано на базі мікросхеми Cyclone V компанії Intel FPGA, що містить апаратний ARM процесор, блок програмованої логіки, сполучений з ARM 256-бітними з’єднаннями, та апаратний контролер DDR3 пам’яті.

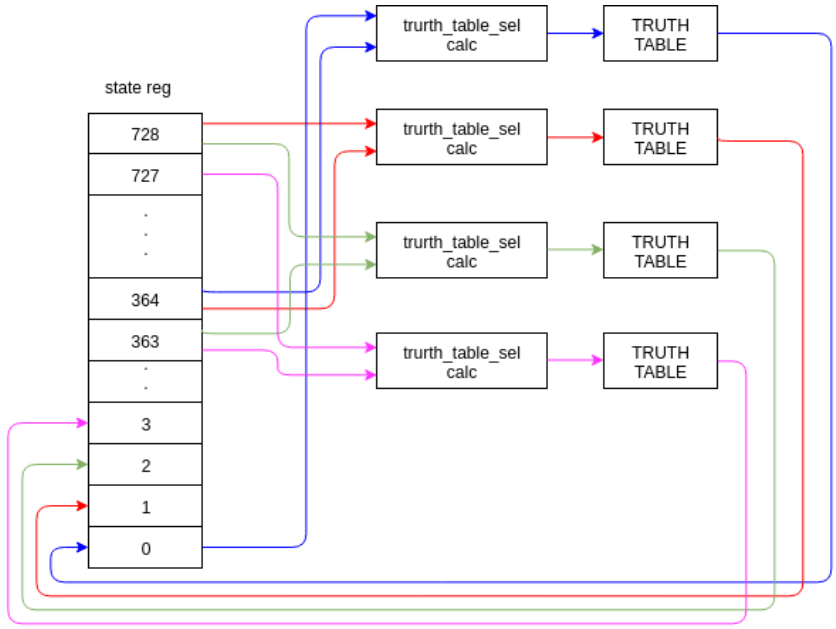

Рис. 4. Структурна схема апаратного обчислювача функції Transform, що є основою геш-функції Curl.

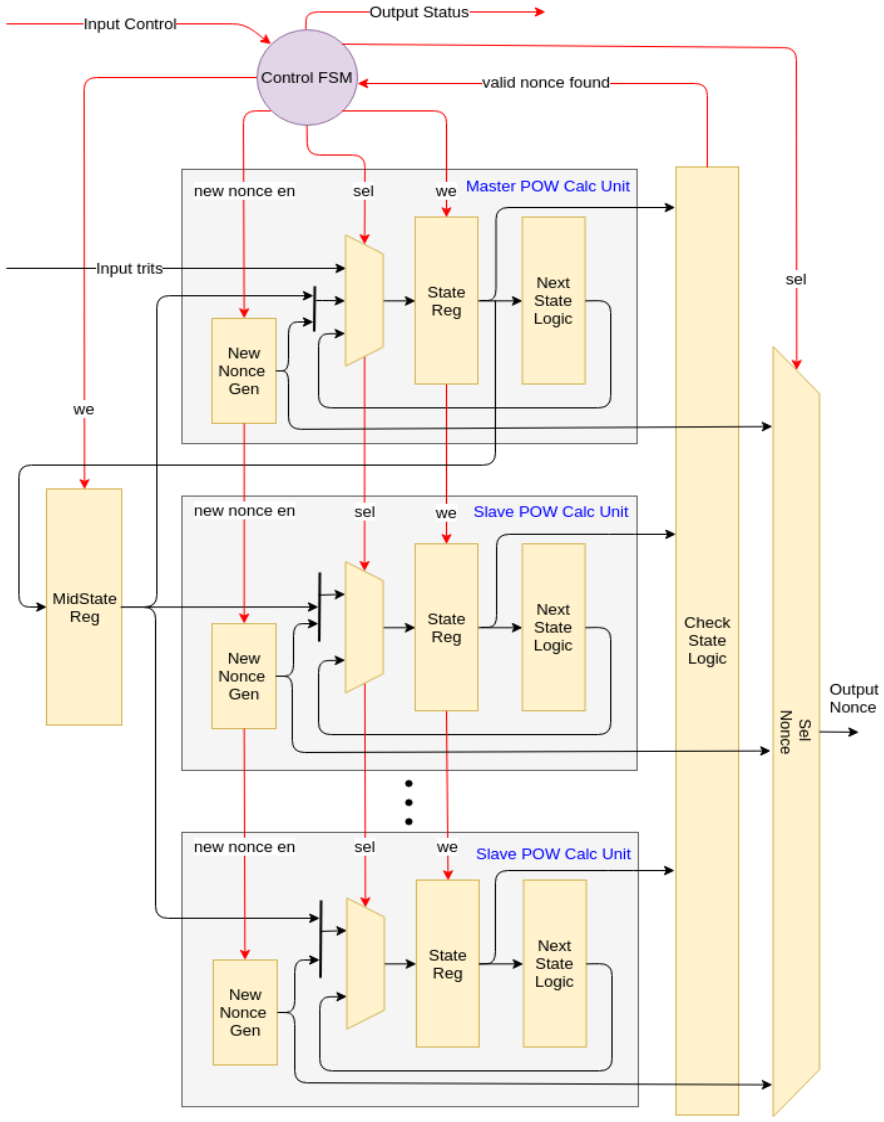

Рис. 5. Структура апаратного прискорювача операції PoW для криптовалюти IOTA.

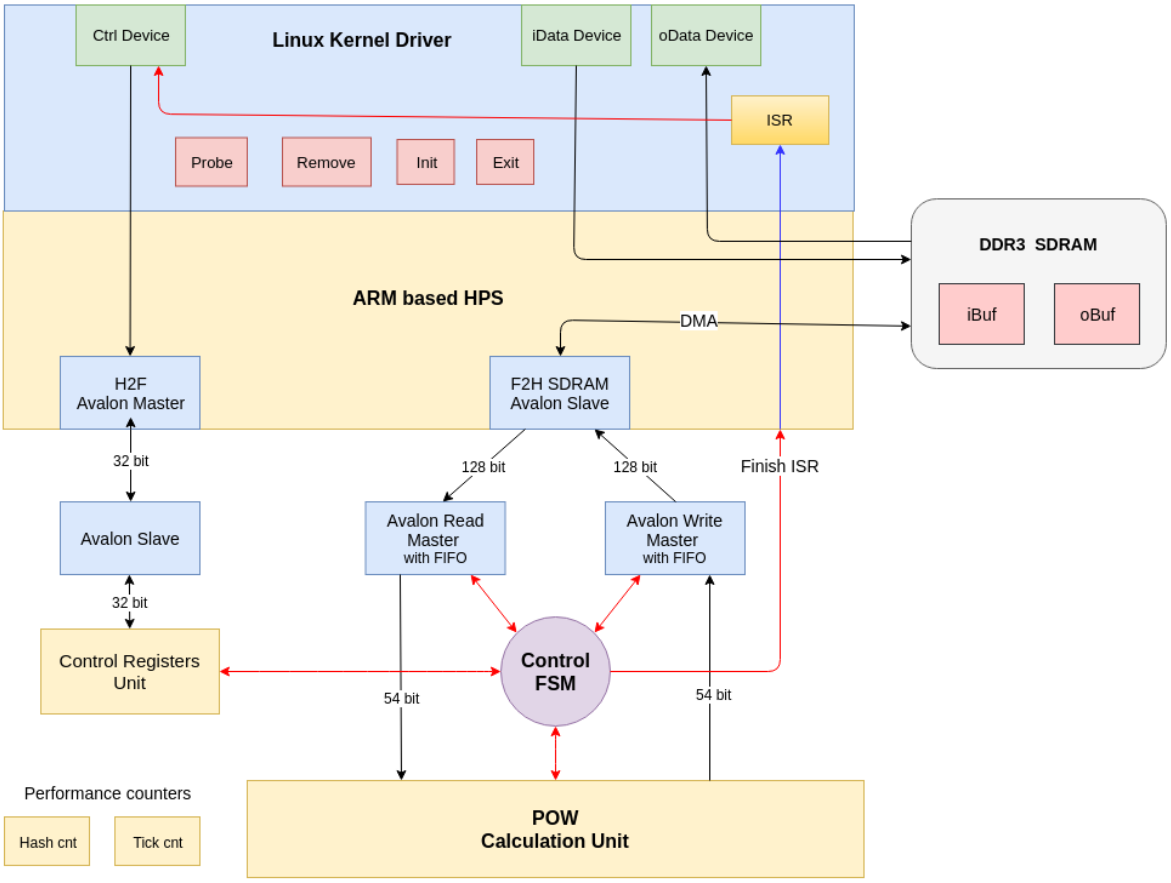

Рис. 6. Структура системи-на-кристалі на базі ARM процесора з інтегрованим апаратним прискорювачем операції PoW.

У пропонованій СнК апаратний прискорювач операції PoW підключено до вбудованого ARM процесора за допомогою двох інтерфейсів шини Avalon: ведучого (master) та веденого (slave).

З використанням веденого Avalon інтерфейсу можна записувати/зчитувати 32-бітні регістри керування з програм, що виконуються ARM процесором.

Головний регістр керування має номер (адресу) 0. Перший біт цього регістру використовується для запуску операції PoW. В регістр з номером 1 драйвер записує адресу вхідного буферу в оперативній пам'яті, в якому міститься IOTA транзакція. Розмір вхідного буферу складає 8019 тритів (2673 трайтів). В регістр з номером 2 драйвер записує адресу вихідного буферу в оперативній пам'яті, куди буде збережено поле Nonce, що є результатом операції PoW. Розмір вихідного буферу складає 81 тріт (27 трайтів). В програмах користувача з репозиторію [9] тріт представлено, як знакове 8-бітне число. Отже, розмір вхідного і вихідного буферів в бітах розраховують, як добуток значення в тритах на 8. Регістр з номером 3 містить значення MWM, що використовується в операції PoW. Регістр з номером 4 містить кількість геш-функцій Curl, які довелося обчислити для знаходження необхідного поля Nonce. Регістри з номерами 5-6 містять 64-розрядне число, рівне кількості тактів, що знадобилися для знаходження Nonce і визначають тривалість PoW.

З використанням ведучого Avalon інтерфейсу здійснюється прямий доступ до буферів в DDR3 оперативній пам'яті. Такий підхід дає змогу зчитувати IOTA транзакцію із вхідного буферу в пам'яті і записувати результат PoW (поле Nonce) у вихідний буфер без участі центрального процесору. Адреси вхідного і вихідного буферів зберігаються в регістрах керування.

Функціонуванням частини системи, що міститься в програмованій логіці, керує скінченний автомат Control FSM з стійкими станами IDLE, LOAD, TRANSFORM, POW, STORE. Після початку функціонування пристрою та після завершення операції PoW керуючий автомат знаходиться у стані очікування IDLE. Після встановлення першого біту головного керуючого регістру автомат переходить в стан LOAD, з застосуванням прямого доступу до пам'яті завантажує частину IOTA транзакції з вхідного буферу, після чого переходить в стан TRANSFORM. Стани LOAD і TRANSFORM змінюють один одного до завершення операції Absorb від транзакції IOTA за виключенням поля Nonce. Далі автомат переходить в стан PoW і запускає розрахунок операції PoW таким чином, як це було описано вище. По завершенні операції PoW автомат переходить у стан STORE і записує знайдене поле Nonce у вихідний буфер. Після того, як результат операції PoW збережений до оперативної пам'яті, керуючий автомат формує імпульс на лінії переривання процесора ARM, інформуючи

драйвер пристрою про завершення розрахунків і переходить в стан IDLE.

Було створено Linux-драйвер, з використанням якого програми користувача мають можливість записувати і зчитувати дані з вхідного та вихідного буферів в ядрі ОС Linux, записувати регістри керування прискорювача та очікувати на переривання, що сигналізує про завершення операції PoW.

Вихідний код апаратного прискорювача операції PoW та СнК мовою Verilog, а також вихідний код Linux-драйверу мовою С знаходиться у відкритому доступі [14].

VII. Оцінка продуктивності обчислень та апаратурних витрат

Для оцінки продуктивності обчислень і апаратурних витрат запропоновану СнК апаратного прискорювача операції PoW було синтезовано для FPGA Cyclone V 5CSEBA6U23I7 з вбудованим ARM. Кількість обчислювачів операції PoW визначено рівною 11. Тактова частота 100 МГц.

Апаратурні витрати склали 1200 комірок типу ALM та 2400 синхронних по фронту D-тригерів для одного апаратного обчислювача операції PoW (3% ресурсів обраної FPGA мікросхеми). Апаратурні витрати прискорювача, що містить 11 обчислювачів, склали відповідно 30% ресурсів обраної FPGA мікросхеми. За необхідності кількість обчислювачів можна перевизначити за допомогою параметру *N*. Максимальна тактова частота обчислювача функції Transform — 256 МГц. Максимальна тактова частота СнК прискорювача PoW склала 130-140 МГц, в залежності від кількості обчислювачів PoW. Продуктивність одного PoW обчислювача складає 1 204 819 гешів на секунду для тактової частоти 100 МГц.

Для вимірювання продуктивності програмної реалізації PoW криптовалюти IOTA використано реалізацію CCURL мовою С, створену розробниками IOTA [9], [13] та зкомпільовану для дистрибутиву Linux Debian, запущеного на ARM процесорі налагоджувальної плати DE10-Nano. Тривалість виконання програмної реалізації PoW визначалася з використанням утиліти time ОС Linux.

Для вимірювання продуктивності апаратного прискорювача операції PoW криптовалюти IOTA написано користувацьку програму для ОС Linux, що випадковим чином генерує задану кількість IOTA транзакцій, за допомогою драйверу запускає роботу апаратного акселератора PoW і зчитує результат. Далі з використанням еталонної програмної реалізації бібліотеки CCURL розраховується геш-функція від IOTA транзакції зі знайденим полем Nonce і виконується перевірка на рівність нулю MWM останніх трітів результату. Апаратний прискорювач рахує кількість тактів системної частоти та кількість обчислених геш-функцій Curl, що знадобилися для одержання результату операції PoW. Після зчитування зазначених даних з регістрів прискорювача розраховується тривалість операції PoW та продуктивність обчислень у вигляді кількості геш-функцій, розрахованих за секунду.

Тривалість виконання програмної реалізації операції PoW криптовалюти IOTA для різних транзакцій і MWM = 15 склала від 10 хв до 50 хв (15 хв в середньому).

Тривалість виконання операції PoW з використанням запропонованого апаратного прискорювача для різних транзакцій і MWM = 15 склала від 0.1 до 4 сек (0.8 сек в середньому). Загальний гешрейт прискорювача, що містить 11 апаратних обчислювачів операції PoW, склав 13 253 012 гешів на секунду.

Застосування запропонованого апаратного прискорювача операції PoW для криптовалюти IOTA дозволило пришвидшити операцію PoW в 1000 разів (в середньому) у порівнянні з програмною реалізацією.

Вихідний код запропонованого апаратного прискорювача було оприлюднено у травні 2018 року. В цей же час з'явилася ще одна реалізація апаратного прискорювача операції PoW криптовалюти IOTA, описана мовою VHDL, з аналогічними апаратурними витратами і продуктивністю [10]. Для функціонування згаданого аналога необхідний зовнішній мікрокомп'ютер Raspberry Pi, а обмін даними з прискорювачем реалізований через послідовний інтерфейс SPI, на відміну від швидкісного прямого доступу до пам’яті, застосованого авторами пропонованої роботи.

Висновки

В роботі запропоновано структуру апаратного обчислювача геш-функції Curl, в якій один раунд операції Transform розраховується за один такт, а розрахунок геш-функції потребує 81 такту.

На основі створеного обчислювача геш-функції Curl запропоновано структуру апаратного обчислювача операції PoW криптовалюти IOTA.

На основі запропонованих структур апаратних обчислювачів створено реалізацію апаратного прискорювача операції PoW криптовалюти IOTA мовою Verilog, де кількість обчислювачів PoW можна визначити за допомогою параметру *N*.

Розроблений апаратний прискорювач інтегровано в СнК на базі ARM процесора. Створено робочий прототип на базі налагоджувальної плати DE10-Nano. Прототип включає 11 обчислювачів PoW і функціонує на частоті 100 МГц. Для використання прискорювача в програмах користувача створено драйвер для ОС Linux.

Апаратурні витрати склали 1200 ALM комірок та 2400 D-тригерів для одного апаратного обчислювача операції PoW (3% ресурсів обраної FPGA мікросхеми). Загальні апаратурні витрати прискорювача розраховуються, як добуток витрат PoW обчислювача на кількість обчислювачів.

Застосування запропонованого апаратного прискорювача дозволило пришвидшити операцію PoW криптовалюти IOTA в 1000 разів (в середньому) у порівнянні з програмною реалізацією.

ВНЕСОК АВТОРІВ

Короткий Є.В.: розробка апаратного прискорювача операції PoW для криптовалюти IOTA, створення драйверу.

Сачов С.О.: оцінка продуктивності обчислень та апаратурних витрат запропонованого рішення.

ПЕРЕЛІК ПОСИЛАНЬ

[1] "Ericsson mobility report," 2015. [Online]. Available: https://www.ericsson.com/assets/local/news/2016/03/ericsson-mobility-report-nov-2015.pdf.

[2] "VISA processing," 2018. [Online]. Available: https://www.visaeurope.com/enabling-payments/processing.

[3] S. Popov, "The Tangle," 2018. [Online]. Available: https://iota.org/IOTA_Whitepaper.pdf.

[4] Q. Bramas, "The Stability and the Security of the Tangle," Apr. 2018, **URL:** https://hal.archives-ouvertes.fr/hal-01716111.

[5] N. B. Biradar, "IOTA-Next Generation Block chain," *Int. J. Eng. Comput. Sci.*, vol. 7, no. 04, pp. 23823–23826, Apr. 2018, **DOI:** 10.18535/ijecs/v7i4.05.

[6] L. M. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2018, pp. 1545–1550, **DOI:** 10.23919/MIPRO.2018.8400278.

[7] M. Scherer, "Performance and Scalability of Blockchain Networks and Smart Contracts," 2017, **URL:** http://www.diva-portal.org/smash/get/diva2:1111497/FULLTEXT01.pdf.

[8] R. C. Merkle, "A Digital Signature Based on a Conventional Encryption Function," in Advances in Cryptology — CRYPTO '87, Springer, Berlin, Heidelberg, 1988, pp. 369–378, **URL:** http://link.springer.com/10.1007/3-540-48184-2_32.

[9] "Source code for IOTA solutions" [Online]. Available: https://github.com/iotaledger.

[10] T. Pototschnig, "Source code for IOTA VHDL PoW." [Online]. Available: https://github.com/shufps/iota_vhdl_pow.

[11] E. Heilman, N. Narula, T. Dryja, and M. Virza, "IOTA Vulnerability Report: Cryptanalysis of the Curl Hash Function Enabling Practical Signature Forgery Attacks on the IOTA Cryptocurrency," **URL:** https://github.com/mit-dci/tangled-curl/blob/master/vuln-iota.md.

[12] G. Bertoni, M. Peeters, D. Joan, and G. Van Assche, "On the security of the keyed sponge construction," in *Proceedings of the Symmetric Key Encryption Workshop*, 2011, pp. 1–15.

[13] P. D. Handy, "Source code for CURL hash function." [Online]. Available: https://github.com/iotaledger/ccurl/blob/master/src/lib/curl.c.

[14] I. Korotkyi and S. Sachov, "Source code for the proposed IOTA PoW hardware accelerator," 2018. [Online]. Available:https://github.com/LampaLab/iota_fpga/tree/master/pow_accel_soc.

Надійшла до редакції 14 січня 2019 р.

УДК 004.31

Аппаратный ускоритель операции доказательства выполненной работы в криптовалюте IOTA

Сачов[f] С. А.
e-mail sergiosachov@gmail.com
Короткий[s] Е. В., к.т.н. доц., ORCID 0000-0001-8302-4873
e-mail e.korotkiy@kpi.ua
Кафедра конструирования электронно-вычислительной аппаратуры keoa.kpi.ua
Национальный технический университет Украины
«Киевский политехнический институт имени Игоря Сикорского» kpi.ua
Киев, Украина

***Аннотация*—Предложена структура аппаратного ускорителя операции доказательства выполненной работы (Proof-of-work, PoW) в криптовалюте IOTA. Предложенная структура реализована с применением языка Verilog. Разработанный аппаратный ускоритель синтезирован в базисе программируемой логики типа FPGA и интегрирован в систему-на-кристалле для FPGA Cyclone V со встроенным ARM процессором. Создан Linux-драйвер для применения ускорителя в пользовательских программах. Выполнена оценка аппаратурных затрат и производительности вычислений предложенного ускорителя по сравнению с программной реализацией и существующим аналогом.**

Библ. 14, рис. 6.

Ключевые слова — хеш-функция; Verilog; вычислитель; программируемая логика; система-на-кристалле; криптовалюта; интернет вещей; аппаратный акселератор.

UDC 004.31

Hardware Accelerator for Proof-Of-Work Operation in IOTA Cryptocurrency

S. O. Sachov[f]
e-mail sergiosachov@gmail.com
I. V. Korotkyi[s], PhD Assoc.Prof., ORCID 0000-0001-8302-4873
e-mail e.korotkiy@kpi.ua
Department of design of electronic digital equipment keoa.kpi.ua
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" kpi.ua
Kyiv, Ukraine

***Abstract*—The authors proposed hardware accelerator for the proof-of-work (PoW) operation in IOTA cryptocurrency. IOTA allows making secure and authenticated quantum resistant channels between IoT devices for communication and micropayments with no fees. Hardware acceleration reduces transaction time and increases throughput. In the article authors consider a basic theory of IOTA operations, generating and signing transaction with Winternitz one-time signatures. The authors describe operation principle of a new ternary hash function Curl. Winternitz one-time signatures and ternary hash function make IOTA quantum resistant. The core of IOTA is called Tangle and unlike Blockchain has Directed Acyclic Graph structure. There are no miners in the Tangle and IoT devices themselves maintain network operation, which leads to unlimited scalability and absence of fees. To add new transaction to the Tangle, IoT devices need to perform PoW operation for spam and Sybil attack protection — iteratively calculate Curl hash function for the IOTA transaction and change nonce field of the transaction until obtained result doesn't satisfy given criteria, which is some amount of consecutive zero ternary values at the end of transaction hash. The software implementation of Curl hash function is very slow, the PoW operation on embedded devices can last up to 50 minutes, so the hardware acceleration of PoW operation is relevant task. In the proposed work authors created hardware accelerator for IOTA PoW operation. The structure and operation principle of accelerator is described. The proof-of-concept implementations was launched on DE10-nano board, based on Intel programmable logic chip. The proposed PoW hardware accelerator has parameterizable structure. It is possible manually set the number of PoW computing units by changing parameter value. In such parameterizable system one PoW computing unit is master and all remaining PoW units are slaves. Master PoW unit absorbs IOTA transaction, except nonce part, to midstate register utilizing sponge-like approach. Then all POW computing units (master and slaves) preload own state registers from midstate, randomly change personal nonces and start iterative search of valid nonce. When one of PoW computing units finds a valid nonce, PoW operation ends, nonce stored to destination buffer in SDRAM and interrupt is generated for ARM CPU. Final implementation of IOTA PoW hardware accelerator for DE10-nano board contains 11 PoW computing units, delivers 13.2 MH/s hash rate and gives x1000 speedup, compared to software implementation from IOTA developers, for only 30% of 5CSEBA6U23I7 programmable logic chip resources at 100 MHz clock frequency. The average PoW computation time in such implementation is 0.8 second.**

Ref. 14, img. 6.

Keywords - hash function; Verilog; computing unit; programming logic; system-on-chip; cryptocurrency; Internet of Things; hardware accelerator.