УДК 004.274:004.056

С.Я. Гільгурт, Київ
О.Г. Кіслов, Київ
В.М. Попова, Київ
І.М. Лях, Ужгород

ПРИСКОРЕНЕ ОБЧИСЛЕННЯ ХАРАКТЕРИСТИК РЕКОНФІГУРОВНИХ СХЕМ РОЗПІЗНАВАННЯ НА БАЗІ АСОЦІАТИВНОЇ ПАМ'ЯТІ ТА ЦИФРОВИХ КОМПАРАТОРІВ

Abstract. The combination methods that have recently been proposed to increase the efficiency of reconfigurable security tools require techniques to speed up the calculation of technical parameters used as optimization criteria. In this paper we propose an approach build on so called "calculator-functions", aimed at quickly calculating the amount of resources required to synthesize a device required. An example of constructing the calculator-function for original CAM-based scheme which is one of the most common methods for creating reconfigurable pattern matching circuits is considered.

Вступ

У зв’язку зі сталим зростом об’єму мережевого трафіку, кількості та витонченості атак програмні рішення останнім часом не впорються з задачею розпізнавання сигнатур для таких технічних засобів захисту інформації, як мережеві системи виявлення вторгнень, антивірусні сканери, фільтри протидії мережевим хробакам, тощо. Вирішити проблему дозволяють реконфігуровні апаратні рішення на базі програмованих логічних інтегральних схем – ПЛІС, які поєднують в собі продуктивність спецпроцесорів і гнучкість програмного забезпечення [1 – 3]. Але їх ефективне застосування потребує додаткових досліджень. Одним з напрямів є підхід, що базується на поєднанні в одному модулі різних принципів побудови апаратних схем розпізнавання таким чином, щоб якнайкраще використати переваги кожного з них [4]. Алгоритми, які реалізують такі методи комбінування, повинні швидко обчислювати часові та ресурсні характеристики блоків та модулів розпізнавання. В згаданій вище роботі з метою прискорення загального процесу оптимізації запропоновано створювати для кожного компонента, що реалізує певний підхід, так звану функцію-калькулятор.

Метою даної роботи є розробка та дослідження принципів створення функцій-калькуляторів для одного з найпоширеніших підходів до побудови реконфігуровних сигнатурних засобів захисту, який заснований на використанні асоціативної пам'яті (АП), відповідний англомовний термін – Content-Addressable Memory (CAM), на базі цифрових компараторів (ЦК) [5].

Синтез модуля розпізнавання сигнатурної системи захисту

Найважливішим компонентом, характеристики якого безпосередньо впливають на властивості сигнатурної системи захисту інформації в цілому, є модуль розпізнавання (МР) [6].

Суть методів комбінування полягає в поділі набору патернів (що мають відшуковуватися системою у вхідному потоці даних) між кількома блоками розпізнавання різної природи БР$_i$ таким чином, щоб досягти найкращих характеристик МР і, як наслідок, всієї системи захисту. Найвища ефективність при цьому досягається за рахунок охоплення обох процесів – поділу патернів та підбору параметрів БР$_i$ – загальною процедурою оптимізації. Змінними параметрами при цьому є кількість підгруп патернів (для кожного блоку – окрема підгрупа) та певна комбінація їх розподілу між підгрупами, а також варіанти реалізації кожного з блоків БР$_i$, які обираються з бібліотеки готових компонентів [4]. Критерії оптимізації можуть використовуватися різні в залежності от потреб користувачів. Метою оптимізації є мінімізація або максимізація певної цільової функції, в якості якої виступає чисельне значення певного технічного параметру: об’єм споживаних ресурсів, швидкодія, продуктивність, тощо [7]. Алгоритм, що реалізує комбінований метод, варіює змінні параметри, обчислюючи на кожному кроці обрані характеристики – критерії оптимізації кожного з БР$_i$ та МР в цілому.

Технічні параметри модулю розпізнавання

Розглянемо технічні параметри, які найчастіше використовуються в якості цільових функцій процедури оптимізації.

Ресурсні параметри

Будь-яка цифрова схема в загальному випадку складається з декількох компонентів. Відповідно ресурси R, потрібні для її синтезу, складаються з ресурсів R_i, потрібних для створення кожного з компонентів

$$R = \sum_{i=1}^{n} R_i ,$$

де n – кількість компонентів у складі схеми

Оцінити обсяг ресурсів, потрібних для синтезу даної обчислюваної структури, що створюється на базі реконфігуровного обчислювача [8], можна в деяких умовних одиницях. Фізичним сенсом такої одиниці може бути мінімальний структурний елемент ПЛІС, наприклад, пошукова таблиця LUT. Тоді об’єм ресурсів, що споживає i-й блок розпізнавання БР$_i$, може бути обчислений в умовних еквівалентних пошукових таблицях LUT:

$$R_i = L_i + \alpha F_i + \beta B_i + \gamma M_i, \quad (1)$$

де L_i – об'єм ресурсів логіки ПЛІС, які потрібні для синтезу i-го компоненту (власне кількість пошукових таблиць LUT);

F_i – об'єм ресурсів розподіленої пам'яті ПЛІС (кількість тригерів),

B_i – об'єм ресурсів блокової пам'яті ПЛІС (Мбітів),

M_i – об'єм ресурсів зовнішньої пам'яті – бортової пам'яті реконфігуровного обчислювача (Мб),

α, β, γ – коефіцієнти приведення ресурсів різного типу до логічного ресурсу (пошукових таблиць LUT).

Швидкісні параметри

Кожен компонент обробляє вхідні дані з певною швидкодією, головним з параметрів якої є час затримки розповсюдження сигналу T_i. В разі паралельного з'єднання компонентів час швидкодія схеми в цілому визначається найповільнішим з них:

$$T = \max(T_i).$$

У випадку послідовного з'єднання компонентів час затримки цифрової схеми в цілому визначається як сума затримок кожного з них:

$$T = \sum_{i=1}^{n} T_i .$$

Теоретично, ресурсні та часові характеристики реконфігуровних пристроїв, які потрібні для обчислення цільової функції, можна знайти шляхом синтезу їх цифрових схем за допомогою інструментальних засобів створення конфігурацій для ПЛІС, наприклад, программного пакету Xilinx WebPack ISE [9]. Але цей процес потребує забагато часу [10], що унеможливлює його використання на практиці.

Тому було запропоновано техніку прискореного обчислення характеристик блоків розпізнавання, суть якої полягає у створенні для кожного i-го бібліотечного компонента (з яких в процесі виконання процедури оптимізації складається МР) так званої функції-калькулятора θ_i. Така функція, маючи на вході задану підгрупу патернів P_i та окремі параметри задіяної ПЛІС, повинна обчислювати та видавати на виході в якості результату чисельну оцінку об'єму ресурсів R_i, потрібних для створення компонента, що розпізнаватиме цю підгрупу патернів, та чисельну оцінку часової затримки T_i, яку він матиме після синтезу:

$$\theta_i = \{ R_i, T_i \}.$$

Оскільки важливу роль в реалізації методів оптимізації відіграють властивості набору патернів, розглянемо допоміжні положення, які спрощуватиме процес формалізації та оперування ними.

Формалізований опис набору патернів

Положимо, що $P = \{\sigma, m_{\min}, m_{\max}, \delta, \mu, \nu\}$ – це множина патернів (фіксованих послідовностей символів, поданих у певному кодуванні), яка характеризується потужністю σ, довжиною найкоротшого $m_{\min}$ та найдовшого $m_{\max}$ патернів, функцією розподілу довжин δ, а також першою та другою функціями самоподоби μ та ν.

Для забезпечення можливості здійснення розрахунків, пов'язаних с множиною патернів, запропоновано наступну техніку впорядкування патернів у наборі.

Упорядкуємо всі патерни в наборі за зростанням їх довжин, як подано на рис. 1. Тут $p_1, p_2, p_3, \ldots p_k, \ldots p_\sigma$ – патерни, що входять до множини P. Складені з квадратів стовпчики зображають складені з символів рядки.

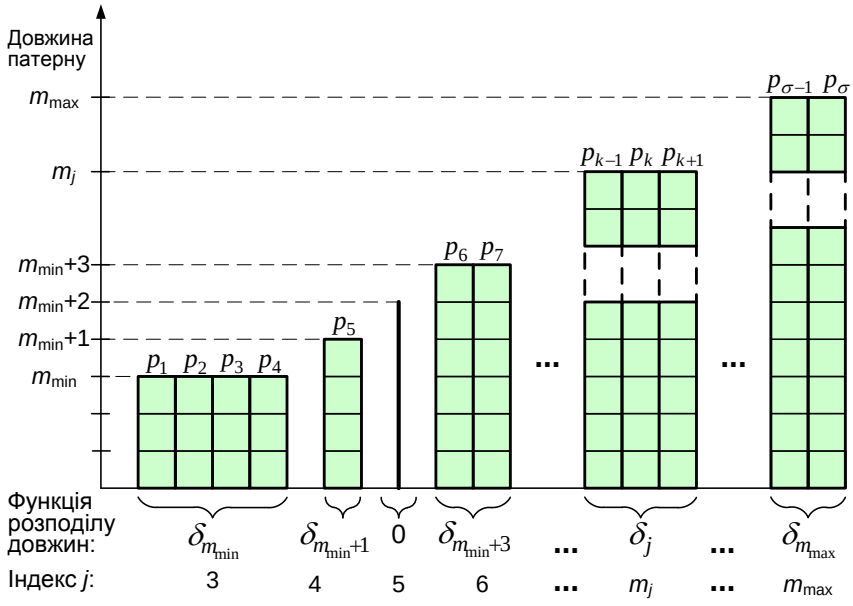

Рис. 1. Впорядкування патернів пакетами з однакової довжини

Назвемо *пакетами* сукупності патернів однакової довжини, не звертаючи уваги, як само упорядковані патерни всередині пакету. Введемо індекс j таким, що співпадає з довжиною патернів в пакеті $j = m_{\min}, m_{\min}+1, m_{\min}+2, \ldots m_j, \ldots m_{\max}$, де $m_{\min}$ – довжина найкоротшого патерну; $m_{\max}$ – довжина найдовшого патерну, притому кожне наступне його значення обов'язково на одиницю більше за попереднє, тобто в нумерації немає пропусків. Тоді довжина кожного патерну в наборі співпадатиме з індексом його пакету: $m_j = j$. Але зворотне твердження хибне – для деяких індексів j розмір відповідного пакету може бути нульовим, якщо патерни відповідної довжини відсутні.

Визначимо функцію розподілу довжин δ (як залежність від індексу j) рівною кількості патернів однакової довжини у відповідному пакеті: $\delta(j) = \delta_j$. В прикладі на рис. 1 $j = 3, 4, 5, \ldots m_{\max}$; $\delta(3) = 4$, $\delta(4) = 1$, $\delta(5) = 0$, $\delta(6) = 2$, $\delta(m_{\max}) = 2$.

За допомогою функції розподілу довжин можливо обчислити кількість символів в наборі патернів Ω. Ця величина дорівнює сумі символів у кожному пакеті, яка в свою чергу дорівнює добутку довжини рядків на їх кількість в пакеті:

$$\Omega = \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot m_j = \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot j . \qquad (2)$$

Техніка створення функцій-калькуляторів блоків розпізнавання, побудованих з використанням асоціативної пам'яті

Один з основних підходів до побудови реконфігуровних сигнатурних засобів захисту заснований на використанні асоціативної пам'яті [5, 11 – 15]. Оскільки при реалізації пристроїв такого пам'яті цього класу реконфігуровними засобами найчастіше використовують цифрові компаратори, в даному дослідженні поняття АП та ЦК використовуватимуться я синоніми.

В зв'язку с прозорістю та регулярністю структур на ЦК функція-калькулятор для блоку розпізнавання, що використовує АП, може бути заснована на прямому підрахунку потрібних ресурсів та часових затримок. В деяких випадках результати такого підрахунку можуть уточнюватися внаслідок виявлення та використання певних технічних особливостей реалізації задіяного підходу. Отже запропонована техніка створення функції-калькуляторів складається з етапу прямого підрахунку та, в загальному випадку, декількох етапів уточнення отриманих співвідношень.

Оскільки схеми розпізнавання на АП не потребують блокової або зовнішньої пам'яті, вираз (1) для знаходження ресурсних витрат схеми, побудованої за таким підходом спрощується до вигляду:

$$R^*_{\text{CAM}} = L_{\text{CAM}} + \alpha F_{\text{CAM}}, \qquad (3)$$

де L_{CAM} і F_{CAM} – відповідно кількість LUT і тригерів в схемі.

В зв'язку с передбаченими подальшими уточненнями значення оцінки споживаних схемою ресурсів розглядатиме величину R^*_{CAM} як таку, що отримана в першому наближенні.

Незважаючи на наявність великої кількості модифікацій базових підходів до побудови реконфігуровних засобів захисту та технік підвищення їх ефективності [5] для ознайомлення з принципами побудови функції-калькуляторів доцільно обмежитися розгляданням лише базової схеми АП на ЦК, що отримала назву BSCAM, від якої походять більшість модифікацій.

Базова схема BSCAM. Ресурсна складова

На рис. 2 наведена спрощена схема, що пояснює принцип функціонування базової схеми розпізнавання BSCAM. Вона містить конвеєр, складений з 8-розрядних регістрів RG_i, компаратори CMP_1 … CMP_3, кожен з яких виконує функцію порівняння з якимось певним символом, та логічний елемент "І", що об'єднує їх виходи. Набір компараторів відповідає одному з патернів, які підлягають розпізнаванню. На вхід конвеєру подається послідовність символів, що аналізується. В разі збігу фрагменту цієї послідовності з патерном "ABC" на виході Match з'являється активний сигнал.

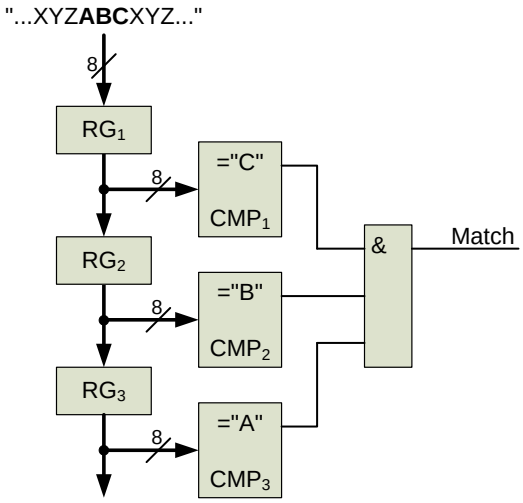

Рис. 2. Базова схема безпосереднього розпізнавання компараторами патерну "ABC"

Схема на рис. 2 виглядає досить простою. Щоб зрозуміти труднощі її практичного застосування реконфігуровними засобами розглянемо наступний факт. Виходи регістра RG_1 подаються на компаратори, що відповідають останньому символу кожного з патернів словнику сигнатур, виходи регістра RG_2 – на компаратори передостаннього символу кожного з патернів і т.д. Оскільки словник сигнатур сучасних засобів захисту може містити велику кількість патернів (десятки та сотні тисяч), здатність навантаження (fan-out) виходів цих регістрів, синтезованих на штатних компонентах ПЛІС, виявляється недостатньою. А ще довжина цифрових ліній, що зв’язують велику кількість логічних елементів, розподілених по площині кристалу ПЛІС, призводить до затримок розповсюдження сигналу, внаслідок чого знижується максимально можлива частота функціонування всієї цифрової схеми.

Для вирішення проблеми створюють конвеєр з декількох ступенів (рис. 3), на кожному з котрих вихідні сигнали розгалужуються на множину входів D-тригерів наступного ступеню (див. Fig. 4 в [14]).

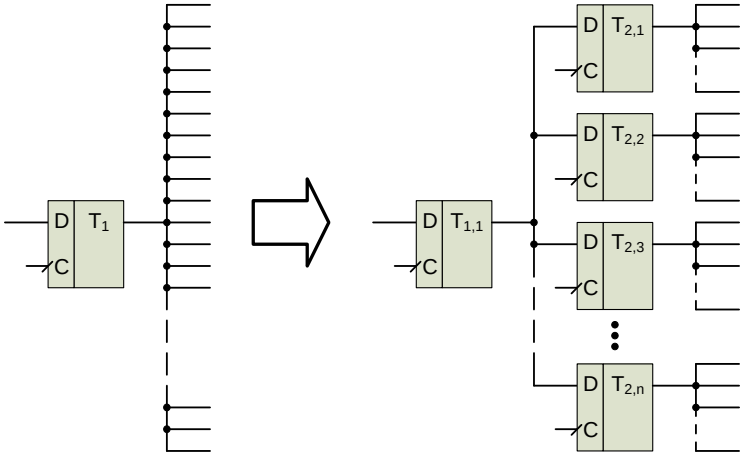

Рис. 3. Вирішення проблеми навантаження виходів

Таке рішення дозволяє розподілити виходи регістрів RG_i на довільну кількість компараторів без зниження тактової частоти.

Інша проблема пов'язана з надвеликою кількістю входів логічних елементів "І" схеми на компараторах, оскільки довжина патернів може сягати десятків символів, а пошукові таблиці LUT сучасних ПЛІС, на яких реалізуються ці елементи, мають зазвичай 4, 6 або 8 входів. Дана складність також вирішується шляхом конвеєризації (див. рис. 4, Fig. 2(a) в [11] або Fig. 5 в [14]).

Використовуючи формулу (3), знайдемо кількість обчислювальних ресурсів, що потребує базова схема розпізнавання на цифрових компараторах

BSCAM (рис. 2).

Пошукові таблиці LUT в схемі BSCAM використовуються, по-перше, для синтезу компараторів CMP, по-друге – конвеєра для створення багатовходової схеми "I" (рис. 4), тому кількість пошукових таблиць LUT, що потребує схема BSCAM, складається з двох відповідних доданків:

$$L_{\text{BSCAM}} = L_{\text{CMP}} + L_{\&}. \tag{4}$$

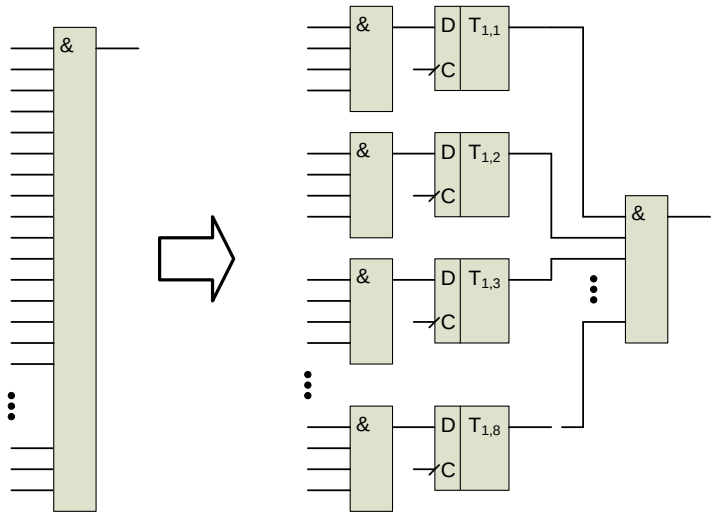

Рис. 4. Вирішення проблеми входів логічних елементів "I"

Особливістю схеми BSCAM є той факт, що вона потребує стільки компараторів, скільки символів сумарно міститься у всіх патернах, що мають розпізнаватися, тобто Ω.

Один цифровий компаратор CMP, що розпізнає символ у байтовому кодуванні, потребує дві пошукові таблиці LUT на 4 входи чи на 6 входів, або одну 8-входову LUT. Введемо функцію-кваліфікатор

$$\Lambda(z) = \begin{cases} 1, & z \geq 8 \\ 2, & z < 8 \end{cases}.$$

Тоді кількість LUT, що потрібна для створення всіх компараторів схеми BSCAM, дорівнюватиме

$$L_{\text{CMP}} = \Lambda(x) \cdot \Omega, \tag{5}$$

де x – кількість входів пошукової таблиці LUT для заданої ПЛІС.

Нескладно перевірити, що кількість LUT, потрібних для об'єднання j входів конвеєрною схемою "I" згідно рис. 4 з урахуванням того факту, що на x-входовій LUT можна синтезувати логічну схему "I" не більш, чим на x входів, для розпізнавання патерну довжиною j потрібно таблиць LUT у кількості

$$L_j = L(j) = \left\lceil \frac{j-1}{x-1} \right\rceil. \tag{6}$$

Кількість LUT для складання всіх конвеєрів всіх патернів підраховується аналогічно (2):

$$L_{\&} = \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot L_j = \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left\lceil \frac{j-1}{x-1} \right\rceil. \tag{7}$$

Підставляючи (5) та (7) в (4) та враховуючи (2), а також той факт, що кількість входів x під час розрахунків функції-калькулятору є константою, отримуємо загальну кількість LUT для схеми BSCAM:

$$L_{\text{BSCAM}} = \Lambda(x) \cdot \Omega + \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left\lceil \frac{j-1}{x-1} \right\rceil = \Lambda(x) \cdot \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot j + \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left\lceil \frac{j-1}{x-1} \right\rceil =$$

$$= \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left(\Lambda(x) \cdot j + \left\lceil \frac{j-1}{x-1} \right\rceil \right). \tag{8}$$

Кількість тригерів, потрібних для створення схеми BSCAM, складається з кількості тригерів F_{RG}, потрібної для побудови вхідного конвеєру (рис. 2), кількості тригерів F_{fan} в конвеєрі розгалуження для підвищення здатності навантаження (fan-out) виходів регістрів вхідного конвеєра (рис. 3) та кількості тригерів $F_{\&}$ в конвеєрі для об'єднання по "I" виходів всіх компараторів для всіх патернів (рис. 4). Тому кількість тригерів, що потребує схема BSCAM, складається з трьох відповідних доданків:

$$F_{\text{BSCAM}} = F_{\text{RG}} + F_{\text{fan}} + F_{\&}. \tag{9}$$

Довжина вхідного конвеєру, по якому просувається потік символів, що розпізнаються, дорівнює найдовшому патерну з підгрупи $m_{\max}$, а ширина – одному байту, тобто восьми бітам. Тому для його побудови потрібно тригерів:

$$F_{\mathrm{RG}} = 8\,m_{\max}\,. \qquad (10)$$

Підрахунок числа F_{fan} виявляється дещо складнішим порівняно з попередніми розрахунками. Навантаження на вихідні ланцюги регістрів вхідного конвеєру нерівномірне. Якщо з його перших $m_{\min}$ ступенів сигнали використовуються для розпізнавання всіх патернів підгрупи, то з останнього – тільки для розпізнавання пакету з найдовших патернів, як можна бачити на рис. 5.

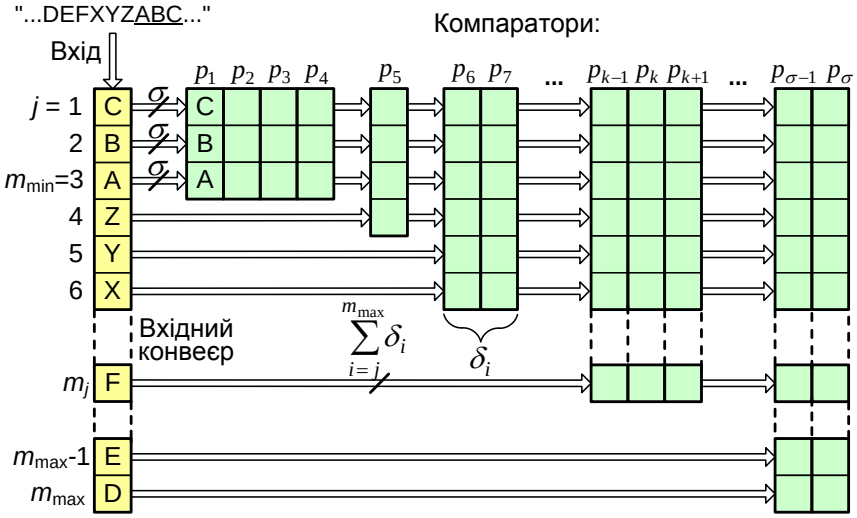

Рис. 5. Розподіл навантаження на вхідний регістр у схемі BSCAM

Тут складені з квадратів стовпчики зображають складені з компараторів схеми розпізнавання патернів $p_1, p_2, p_3, \ldots p_k, \ldots p_\sigma$, що входять до множини P.

Сигнали з виходів кожного з перших $m_{\min}$ каскадів конвеєра подаються на всі σ патернів підгрупи, тобто мають навантаження:

$$O_1 = \sum_{i=m_{\min}}^{m_{\max}} \delta_i = \sigma\,.$$

Вихід кожного j-го з решти каскадів конвеєра, починаючи з $i = m_{\min}+1$, подається на патерни, що входять у пакети з номерами від j по $m_{\max}$, тобто повинен розгалужуватися на кількість входів

$$O_j = \sum_{i=j}^{m_{\max}} \delta_i\,.$$

Кількість тригерів, потрібних для розгалуження одного виходу на O входів конвеєром може бути обчислена аналогічною (6) за формулою:

$$F(O)=\left\lceil \frac{O-1}{y-1} \right\rceil, \tag{11}$$

де y – здатність навантаження виходів тригерів для заданої ПЛІС.

Тоді загальна кількість тригерів, потрібних для розвантаження всіх виходів вхідного конвеєру – як перших $m_{\min}$, так і решти, з урахуванням (11):

$$F_{\text{fan}} = m_{\min}\left\lceil \frac{O_1-1}{y-1} \right\rceil + \sum_{j=m_{\min}+1}^{m_{\max}} \left\lceil \frac{O_j-1}{y-1} \right\rceil =$$

$$= m_{\min}\left\lceil \frac{\sigma-1}{y-1} \right\rceil + \sum_{j=m_{\min}+1}^{m_{\max}} \left\lceil \frac{\sum_{i=j}^{m_{\max}} \delta_i - 1}{y-1} \right\rceil. \tag{12}$$

Кількість тригерів в конвеєрі для об'єднання по "I" для кожного патерну відрізняється на одиницю від кількості LUT $L_{\&}$ у цьому ж конвеєрі згідно (7), тому що після останнього каскаду схеми "I" тригер не потрібен:

$$F_{\&} = L_{\&} - 1 = \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left\lceil \frac{j-1}{x-1} \right\rceil - 1. \tag{13}$$

Отже, згідно (9), враховуючи (10), (12) та (13), отримуємо загальну кількість тригерів у схемі BSCAM:

$$F_{\text{BSCAM}} = 8m_{\max} + m_{\min}\left\lceil \frac{\sigma-1}{y-1} \right\rceil + \sum_{j=m_{\min}+1}^{m_{\max}} \left\lceil \frac{\sum_{i=j}^{m_{\max}} \delta_i - 1}{y-1} \right\rceil + \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left\lceil \frac{j-1}{x-1} \right\rceil - 1. \tag{14}$$

Нарешті, загальна кількість обчислювальних ресурсів у базовій схемі BSCAM розпізнавання на базі ЦК згідно виразів (3), (8) та (14):

$$R^*_{\text{BSCAM}} = L_{\text{BSCAM}} + \alpha F_{\text{BSCAM}} = \sum_{j=m_{\min}}^{m_{\max}} \delta_j \cdot \left(\Lambda(x) \cdot j + \left\lceil \frac{j-1}{x-1} \right\rceil \right) +$$

$$+\alpha\left(8m_{\max}+m_{\min}\left\lceil\frac{\sigma-1}{y-1}\right\rceil+\sum_{i=m_{\min}+1}^{m_{\max}}\left\lceil\frac{\sum_{i=j}^{m_{\max}}\delta_i-1}{y-1}\right\rceil+\sum_{j=m_{\min}}^{m_{\max}}\delta_j\cdot\left\lceil\frac{j-1}{x-1}\right\rceil-1\right). \quad (15)$$

Отже, формула (15) в першому наближенні дає вихідний результат функції-калькулятора (її ресурсної частини) для схеми BSCAM, яка може бути використана в якості одного з блоків розпізнавання БР$_j$ при побудові МР методами комбінування.

Ця формула може бути уточнена, згідно наступних міркувань. Оскільки кожна логічна комірка ПЛІС одночасно володіє обчислювальними ресурсами двох типів – логічними та розподіленої пам'яті – тому що містить як пошукові таблиці LUT, так і тригері, окремі підрахунки ресурсів кожного типу призводять до не зовсім точних результатів.

У випадку BSCAM таке уточнення полягає в наступному. Якщо роздивитися конвеєрну схему багатовходового логічного елементу "I" (рис. 2), можна зрозуміти, що тригери та пошукові таблиці LUT, що мають бути задіяні при його синтезі, можуть бути використані сумісно з однієї логічної комірки ПЛІС, тобто врахована кількість тригерів може бути зменшена на величину $F_{\&}\approx L_{\&}$, тобто доданок $F_{\&}$ можна вилучити з виразу (9). В результаті, отримуємо питому залежність у другому наближенні:

$$R^{**}_{\text{BSCAM}}=\sum_{j=m_{\min}}^{m_{\max}}\delta_j\left(\Lambda(x)\cdot j+\left\lceil\frac{j-1}{x-1}\right\rceil\right)+$$

$$+\alpha\left(8m_{\max}+m_{\min}\left\lceil\frac{\sigma-1}{y-1}\right\rceil+\sum_{j=m_{\min}+1}^{m_{\max}}\left\lceil\frac{\sum_{i=j}^{m_{\max}}\delta_i-1}{y-1}\right\rceil\right). \quad (16)$$

Проаналізуємо здобутий результат.

Вираз (16) є ресурсною складовою θ_R функції-калькулятора, яка надає оцінку ресурсів, потрібних для створення реконфігуровними засобами схеми BSCAM, призначеної для розпізнавання множини патернів P.

В загальному випадку така функція $\theta_R=\theta_R(P,A)$ залежить від властивостей множини патернів

$$P = \{\sigma, m_{\min}, m_{\max}, \delta, \mu, \nu\},$$

які є вхідними змінними в процеси оптимізації, та параметрів реконфігуровного прискорювача

$$A = \{x, y, s, b, \alpha, \beta, \gamma\},$$

що є константами. Тут:

σ – потужність множини патернів;

$m_{\min}$ – довжина найкоротшого, а $m_{\max}$ – найдовшого з патернів;

δ – функція розподілу довжин патернів;

μ і ν – відповідно перша та друга функції самоподоби;

x – кількість входів пошукової таблиці LUT для заданої ПЛІС;

y – здатність навантаження виходів тригерів для заданої ПЛІС;

s – розмір схеми затримки, побудованої на базі LUT;

b розмір (у бітах) блокової пам'яті ПЛІС;

α, β, γ – відповідно коефіцієнти приведення ресурсів розподіленої пам'яті ПЛІС, блокової пам'яті ПЛІС та зовнішньої пам'яті прискорювача до логічного ресурсу (пошукових таблиць LUT).

В нашому випадку (16) функція-калькулятор містить тільки змінні $\sigma, m_{\min}, m_{\max}, \delta$ та константи x, y, α. Зауважимо, що відсутність в даному виразі функцій самоподоби μ та ν свідчить про те, що базова схема BSCAM не використовує надмірність, яка притаманна словнику сигнатур, що є її недоліком, який вирішується чисельними модифікаціями базової схеми [5].

Висновки

В проведеному дослідженні розглянуто техніку прискореного обчислення технічних параметрів реконфігуровної схеми розпізнавання, які дозволяють виконувати процедуру пошуку оптимального рішення за прийнятний час. Наведений приклад побудови так званої функції-калькулятора, а саме – її ресурсної складової для базової схеми BSCAM пошуку патернів, побудованої у вигляді асоціативної пам'яті на базі цифрових компараторів. Отримана залежність вихідного значення оцінки потрібних ресурсів ПЛІС від вхідних змінних, якими є параметри множини патернів, що мають розпізнаватися, та низці констант, що є параметрами реконфігуровного прискорювача, який використовується. Отримана залежність дозволяє не тільки прискорити процедуру оптимізації, але також використовувати її для попередньої оцінки технічних рішень.

1. Реконфигурируемые вычислительные системы: Основы и приложения / А.В. Палагин, В.Н. Опанасенко. – К.: «Просвіта», 2006. – 280 с.

2. *Paxson V., Asanovic K., Dharmapurikar S., Lockwood J., etc.* Rethinking Hardware

Support for Network Analysis and Intrusion Prevention // USENIX First Workshop on Hot Topics in Security (HotSec), Vancouver, B.C., July 31, 2006.
3. *H. Chen, Y. Chen, D.H. Summerville* A Survey on the Application of FPGAs for Network Infrastructure Security // IEEE Communications Surveys and Tutorials. – 2011. – P. 541–561.
4. *Гільгурт С.Я.* Методи побудови оптимальних схем розпізнавання для реконфігуровних засобів інформаційної безпеки // Безпека інформації. – 2019. – Т. 25, № 2. – С.74-81.
5. *Гільгурт С.Я.* Побудова асоціативної пам'яті на цифрових компараторах реконфігуровними засобами для вирішення задач інформаційної безпеки // Электронное моделирование. – 2019. – Т. 41, № 3. – С.59-80.
6. *Евдокимов В.Ф., Давиденко А.Н., Гильгурт С.Я.* Централизованный синтез вычислительных структур реконфигурируемых ускорителей задач информационной безопасности // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2018. – Вип. 82. – С.3-11.
7. *Hilgurt S.* Method for Constructing Reconfigurable Multi-Pattern Matching Modules for Information Security Systems // Захист інформації і безпека інформаційних систем: Матеріали VII Міжнар. наук.-техн. конф, м. Львів, 30 – 31 травня 2019. – Львів: Видавництво Львівської політехніки, 2019. – С.134-135.
8. *Гильгурт С.Я.* Реконфигурируемые вычислители. Аналитический обзор // Электронное моделирование. – 2013. – Т.35, № 4. – С.49-72.
9. Xilinx [Електронний ресурс]. – Режим доступу: https://www.xilinx.com. – Загл. з екрану. – (Дата звернення: 15.11.2019).
10. *Евдокимов В.Ф., Давиденко А.Н., Гильгурт С.Я.* Организация централизованной генерации файлов конфигураций для аппаратных ускорителей задач информационной безопасности // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2017. – Вип. 81. – С.3-11.
11. *Sourdis I., Pnevmatikatos D.N.* Fast, large-scale string match for a 10Gbps FPGA-based network Intrusion Detection System // Field-Programmable Logic and Applications, Proceedings. – 2003. – Vol. 2778. – p. 880-889.
12. *Sourdis I., Pnevmatikatos D.N.* Pre-decoded CAMs for efficient and high-speed NIDS pattern matching // 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, Proceedings. – 2004. – p. 258-267.
13. *Cho Y.H., Mangione-Smith W.H.* Deep packet filter with dedicated logic and read only memories // 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, Proceedings. – 2004. – p. 125-134.
14. *Huang J., Yang Z.K., Du X., Liu W.* FPGA based high speed and low area cost pattern matching // IEEE Region 10 Conference (TENCON 2005); TENCON IEEE Region 10 Conference Proceedings. Melbourne, Australia: IEEE, 2005, p. 2693-2697.
15. *Sourdis I., Pnevmatikatos D.N., Vassiliadis S.* Scalable multigigabit pattern matching for packet inspection // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. – 2008. – Feb. – Vol. 16, No. 2. – p. 156-166.

http://doi.org/10.5281/zenodo.3860720

Поступила 2.09.2019р.