УДК 004.274
DOI https://doi.org/10.32838/2663-5941/2020.2-1/11

Голуб Т.В.
Національний університет «Запорізька політехніка»

Зеленьова І.Я.
Національний університет «Запорізька політехніка»

Грушко С.С.
Національний університет «Запорізька політехніка»

Павлішин М.А.
Національний університет «Запорізька політехніка»

ПІДСИСТЕМА АПАРАТНОГО ПРИСКОРЕННЯ КЛАСИФІКАЦІЇ ТЕКСТІВ У БАЗИСІ FPGA

Комп'ютерні засоби в галузі обробки природних мов, зокрема в задачах класифікації текстів, застосовуються в наш час досить інтенсивно. Актуальною метою досліджень у цій галузі є значне прискорення процесу класифікації текстових документів. Одним із способів вирішення подібних задач є апаратне прискорення, тобто перенесення частини обчислювальних функцій із програмної області в апаратну реалізацію.

В роботі запропоновано спосіб апаратного прискорення процесу класифікації текстових документів на основі принципів наївного методу Баєса. Досліджено процес апаратної класифікації на прикладі чотирьох вхідних масивів, які містять коди слів із різних текстів. Для визначення певної тематики використовується масив відповідних ключових слів, які попередньо були закодовані тим же способом, що і вхідні масиви.

Задачу вирішено в сучасному елементному базисі FPGA шляхом розпаралелювання обчислень в межах однієї мікросхеми. Розроблена підсистема апаратного прискорення класифікації забезпечує можливості паралельної обробки декількох текстів, а також масштабованості та перепрограмування, тобто настроювання на задану тематику. Наведено алгоритм роботи та структурну схему апаратного прискорювача, а також опис логіки функціонування основних складових блоків.

З метою тестування спроектованого пристою в елементному базисі широкого використання обрано мікросхеми FPGA фірми Altera / Intel, мову опису апаратури VHDL. Для імплементації проекту обрано пакет IDE Quartus II, відповідний до продукції фірми-виробника мікросхем. Виконано аналіз часових та апаратурних характеристик пристрою при його реалізації на мікросхемах FPGA фірми Altera / Intel в різних цінових діапазонах. Сформульовано рекомендації щодо вибору мікросхеми для ефективної реалізації прискорювача за критерієм співвідношення «ціна / якість». Визначено перспективні напрями подальшого пошуку рішень задачі апаратного прискорення класифікації текстів.

Ключові слова: *апаратне розпаралелювання обчислень, FPGA, LUT, вбудовані блоки пам'яті, класифікація текстів.*

Постановка проблеми. Комп'ютерні засоби в галузі обробки природних мов (англ. Natural Language Processing), зокрема в задачах класифікації текстів, застосовуються в наш час досить інтенсивно. Певним чином це пов'язано з тим, що із кожним роком обсяг інформації, яка зберігається на електронних носіях та жорстких дисках, значно збільшується, тому необхідні ефективні алгоритми та системи для обробки та аналізу природомовних документів.

Програмне вирішення задач класифікації зазвичай потребує багато машинного часу, оскільки потрібно виконувати цикли повного перебору великих масивів даних, розміри яких мають тенденцію і надалі зростати. Отже, актуальною задачею є значне прискорення процесу класифікації текстових документів. Одним із способів вирішення подібних задач є апаратне прискорення, тобто перенесення частини обчислювальних функцій із програмної області в апаратну реалізацію.

Аналіз останніх досліджень і публікацій. Класифікація текстів належить до групи задач комп'ютерної лінгвістики, яка включає визначення тематичної приналежності текстів, авторського

стилю, емоційного забарвлення висловлювань та багатьох інших характеристик досліджуваного тексту.

Вирішення задачі класифікації текстів – це складний багатоетапний процес аналізу змісту певного документа, а також автоматичного визначення приналежності цього документа до однієї або кількох категорій [1, 2, 4, 6] однієї тематики. В загальному плані методи побудови класифікаторів текстових документів можна поділити на дві групи: засновані на штучному інтелекті та аналітичні.

Методи класифікації, засновані на штучному інтелекті, зазвичай реалізовані на штучних нейронних мережах (ШНМ, artificial neural networks, ANN) [3, 7]. Застосування ШНМ характеризується складністю визначення вагових значень зв'язків між нейронами, великою кількістю нейронів і розрахунків.

До найпоширеніших аналітичних методів, згідно з описами в літературних джерелах, належить наївний метод Байєса (Naive Bayes, NB) [2]; метод опорних векторів (Support Vector Machine, SVM) [4]; метод k-найближчих сусідів (k-nearest neighbors (KNN) [4]; метод Роше (Rocchio) [2]; дерево рішень (decision tree) [4].

Аналітичні методи стосовно питань апаратної реалізації мають певні особливості. Так, метод опорних векторів при своїй простоті реалізації не дозволяє корегувати результати прийняття рішення, до того ж складно відслідкувати безпосередньо сам процес класифікації. Метод k-найближчих сусідів вимагає великого обсягу пам'яті для збереження базового переліку опорних документів і їх постійних перерахунків при аналізі кожного вхідного документу, що зумовлює збільшення кількості розрахунків [4, 6]. Метод Роше вимагає завантаження всього документу, призначеного для аналізу, що висуває додаткові вимоги до обсягів оперативної пам'яті. Дерево рішень характеризується великою надмірністю побудованої структури, яка вимагає недоцільних витрат оперативної пам'яті. Наївний метод Байєса є найпростішим для апаратної побудови класифікатора та забезпечує такі можливості як обробка вхідних даних у реальному масштабі часу, а також досягнення достатньої точності результату при високій швидкодії [1, 2, 6].

Для сучасної апаратної реалізації класифікаторів можуть використовуватися мікроконтролери (МК), програмований логічний контролер (ПЛК), інтегральна схема спеціального призначення (ASIC), програмовані логічні інтегральні схеми (ПЛІС) [7, 8, 10].

Особливістю апаратної реалізації класифікаторів текстових документів є необхідність його адаптації під користувацькі вимоги. Важливо враховувати необхідність зміни переліку опорних даних у відповідності з потребами користувача, а також пристрій має бути відносно недорогим за ціною. Таку можливість без втрати швидкодії може забезпечити архітектура програмованої логічної інтегральної схеми класу FPGA, тому саме цю архітектуру було обрано в цій роботі.

Постановка задачі. Метою дослідження є прискорення процесу класифікації текстів шляхом організації паралельних обчислень у мікросхемах FPGA. Для досягнення цієї мети розроблено підсистему апаратного прискорення класифікації, що забезпечує можливості паралельної обробки декількох текстів, а також масштабованості та перепрограмування, тобто настроювання на задану тематику.

Виклад основного матеріалу дослідження. Апаратна підсистема прискорення процесу класифікації текстів («акселератор») є складовою частиною апаратно-програмного комплексу (Рис. 1), запропонованого авторами для загального вирішення задачі автоматичної обробки текстових документів, із включенням усіх етапів попередньої обробки, а також етапу класифікації.

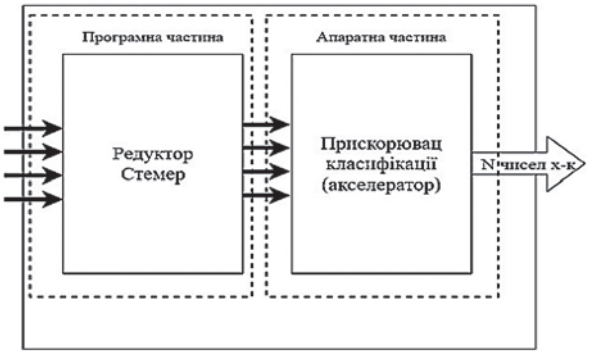

Рис. 1. Загальна структура програмно-апаратного комплексу для класифікації текстів

Розроблюваний програмно-апаратний комплекс має блочну структуру та складається з двох основних частин: програмний блок виконує первинну обробку текстових документів (кодування за ASCII кодами, видалення незначущих слів), а також функції стемінгу та редукції (скорочення простору ознак тексту, що підлягають аналізу); апаратний блок – акселератор, який виконує функції прийому та обчислення даних від програмного блоку та прискорює класифікацію шляхом паралельної обробки декількох масивів із кодами текстової інформації.

В роботі розглядається саме блок акселератора. З метою подальшого тестування та дослідження спроектованого пристрою в елементному базисі широкого використання було обрано мікросхеми

FPGA фірми Altera / Intel, мову опису апаратури VHDL, а також для імплементації проекту – пакет IDE Quartus II, відповідний до продукції обраної фірми-виробника мікросхем [11–14].

Попередньо оброблені вхідні дані завантажуються із програмного блоку до апаратного, а саме до спеціально зарезервованої внутрішньої пам'яті типу ROM або RAM у мікросхемі FPGA. В структурі комплексу на Рис. 1 програмна частина виконує попередні етапи – функції скорочення обсягу вхідної інформації та підрахунку частоти важливих слів. Частота, яка враховується в формі вагового коефіцієнту для кожного слова, представлена у вигляді чисел із комою. Відомо, що архітектура FPGA не досить пристосована для роботи із дійсними числами. Тому з метою прискорення обчислень було вирішено відмовитися від чисел із комою і використовувати формат, який ефективно підтримується архітектурою, зокрема тип *integer*. Зважаючи на те, що згідно із загальною постановкою задачі [5] сформовані редуктором числа мають лише два знаки після коми, було вирішено змістити кому на два розряди, тобто виконати множення на 10^2. Таким чином, формуються аналоги частот вже не дійсного типу, а цілого.

Для зберігання таблиці закодованого тексту в пам'яті необхідно розробити формат даних. Згідно статистики, значуща частина слова може бути визначена достатнім набором із 10-16 літер, у цьому дослідженні – із 11 літер. Це частина слова після стемінгу, тобто без суфікса та закінчення. Літери необхідно закодувати, і найпростішим варіантом є їх перетворення за таблицею ASCII. Згідно таблиці ASCII, для кожної літери є числова відповідність, яка має розмір одного байту.

Необхідно закодувати 11 літер текстового слова, тобто створити відповідний двійковий код-ключ. Отже, необхідно зарезервувати 11*8 = 88 біт для представлення коду одного слова тексту (ключа), а для коду ознаки відповідного вагового коефіцієнта достатньо 8 біт. Для числового відображення відповіді класифікатора, враховуючи формування кінцевого результата шляхом зворотнього перетворення із формату *integer* до формату *real*, виділено 32 біта, що еквівалентно типу *integer* із максимальним значенням 2^{32}. Отже, формат даних – це двійкове слово, яке має довжину 96 біт, з яких 88 біт – довжина ключа, 8 – довжина відображення вагового коефіцієнта (Рис. 2).

В цій роботі досліджено процес апаратної класифікації на прикладі чотирьох вхідних масивів, які містять коди слів із різних текстів. Для визначення певної тематики використовується масив відповідних ключових слів, які закодовані тим же способом, що і вхідні масиви.

Рис. 2. Формат даних внутрішнього масиву класифікатора

Запропонований алгоритм класифікації (Рис. 3) в узагальненому, концептуальному вигляді функціонує так: якщо у внутрішньому масиві є слово, то виконується пошук ключа цього слова у вхідному масиві (текст, оброблений програмно після проходження редуктора). Якщо знайдена відповідність ключа (KEY) із внутрішнього масиву у вхідному масиві, то необхідно перемножити значення частоти слова (VALUE) із вхідного та внутрішнього масивів і скласти всі результати множення. Якщо слово не знайдено, то беремо із внутрішнього масиву наступне слово та виконуємо ітерацію співставлення.

Після визначення формату даних і розробки загального алгоритму роботи блоку акселератора необхідно створити у VHDL-проекті внутрішні пристрої для збереження даних. Зважаючи на те, що в структурі FPGA є блоки, які функціонують як ROM або RAM [9, 10, 13], доцільно використовувати саме їх. Це дозволяє економити загальновживані ресурси (LUT) за рахунок використання спеціальних (ROM, RAM). ROM – швидкодіюча пам'ять, яка не може бути перезаписана без перепрограмування, натомість RAM має можливість завантаження нових даних до пристрою. Згідно з поставленою задачею система повинна мати можливість завантаження нових даних (нових текстів для класифікації), тому обрано RAM-комірку. IDE Quartus II значно спрощує створення таких блоків завдяки мегафункціям.

Для початкової ініціалізації вмісту файлів використовуються *.mif файли, їх вміст ініціюється у блоках пам'яті і при симуляції ці дані є доступними. Доступ до вмісту блоків пам'яті здійснюється за відповідною адресою.

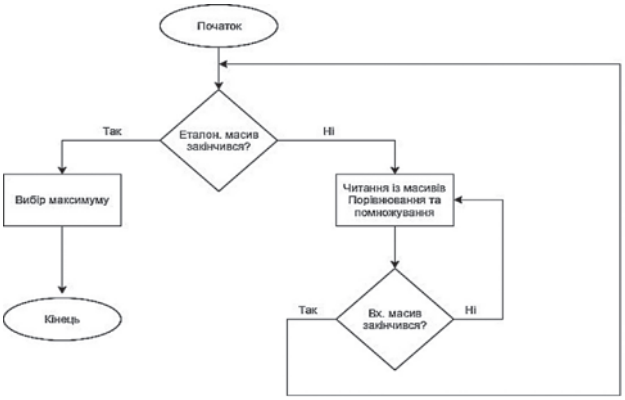

Рис. 3. Узагальнений алгоритм роботи блоку акселератора

Необхідно створити VHDL-сутність пристрою (Рис. 4) для того, щоб генерувати поточну адресу слова у внутрішньому масиві. Генератор адреси повинен мати сигнал дозволу роботи ENA, сигнал скидання R, сигналізувати про останнє слово в пам'яті isLast та генерувати адресу Addr. Логіку функціонування описано за допомогою ключового слова process. Команди в окремому процесі виконуються послідовно, але всі процеси синхронізуються сигналом CLK і виконуються паралельно, що важливо для прискорення обробки декількох масивів.

Генератор адреси та блоки пам'яті використовуються у безпосередньому зв'язку (Рис. 4). Для того, щоб слова не зчитувалися постійно, в схемі передбачено сигнал ReadENA, що дозволяє читання, та окремий сигнал WriteENA для дозволу запису.

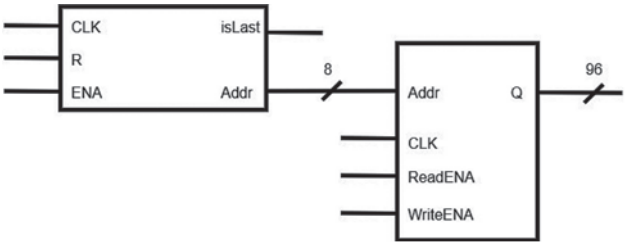

Рис. 4. Схема з'єднання генератора адреси та блоку пам'яті

Важливим функціональним блоком системи є помножувач-акумулятор (Рис. 5). У такій системі цей блок виконує функцію порівняння вхідного тексту із еталонним та перемноження коефіцієнтів у разі співпадіння. Таким чином зрозуміло, що необхідно отримувати два слова по 96 розрядів (входи A та B), порівнювати їх. Також необхідно забезпечити блок сигналом дозволу роботи для уникнення хибних результатів. Доцільно розробляти систему із передбаченням можливої масштабованості. У мові VHDL існує спеціальна конструкція *generic* для того, щоб визначати та змінювати розмірність векторів. Таким чином, система стає більш гнучкою, можна із легкістю змінювати розмірності даних.

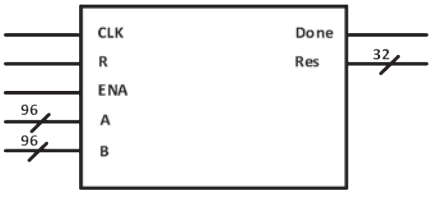

Рис. 5. Помножувач-акумулятор

Для вибору тексту, який найвірогідніше належить до певної категорії, виконується порівняння значень, отриманих від помножувача-акумулятора, з метою вибору максимального з них. Функцію порівняння виконує схема компаратора. З метою масштабованості пропонується синтезувати двопортовий компаратор і розширити його до необхідного розміру (4 входи для однієї задачі). Очевидно, що входи компаратора повинні бути розрядністю в 32 біта згідно з розрядністю виходу Res помножувача-акумулятора, мати сигнали дозволу роботи, сигнали скидання та синхроімпульсу.

Завдяки зручності мови опису апаратури VHDL робота із числами є досить тривіальною задачею. Вона зводиться до звичного програмування та використання операцій «більше», «менше». Після створення одного блоку можна проводити масштабування до чотирьох. Всі спроектовані функціональні блоки потрібно об'єднати в схему пристрою із забезпеченням алгоритмічного порядку роботи. Для цього необхідний керуючий автомат. Існують декілька типів керуючих автоматів, але в цій роботі автори використовують модель Мура, оскільки вона характеризується більш стійким функціонуванням.

Граф-схему алгоритму (ГСА) автомата Мура розроблено згідно з узагальненим алгоритмом роботи акселератора (Рис. 3). Необхідно визначитися із вхідними умовами та функціями, які буде генерувати автомат. Автори визначають умови, які необхідно аналізувати для роботи схеми акселератора:

X_1 – перевірка, чи закінчився внутрішній масив ключів;

X_2 – перевірка, чи всі масиви відпрацювали свої дані;

X_3 – перевірка, чи закінчився вхідний масив;

X_4 – вибір режиму завантаження або роботи над класифікацією.

Всі вихідні функції керуючого автомату по суті є сигналами дозволу роботи для окремих функціональних блоків акселератора. До сигналів, які генерує схема керуючого автомату, належать:

Y_1 – дозвіл читання RAM пам'яті внутрішнього масиву;

Y_2 – дозвіл читання RAM пам'яті зовнішнього масиву;

Y_3 – дозвіл роботи компаратора;

Y_2 – дозвіл роботи помножувача-акумулятора;

Y_4 – сигнал про завершення роботи;

Y_5 – дозвіл запису RAM пам'яті внутрішнього масиву.

Варто зазначити, що при проектуванні цифрових пристроїв може виникати ситуація очікування. Для того, щоб такі стани можна було опрацювати, необхідно вводити спеціальний пустий стан (який

не генерує вихідної функції). Стани автомату Мура при розмітці зручно позначати безпосередньо в операторних вершинах, оскільки вихідні сигнали залежать тільки від поточного стану (Рис. 6).

Пропонується підійти до проектування автомату комбінованим шляхом: основну частину описати за допомогою ГСА, а інші частини (дозвіл роботи та запису до RAM) позначити в коді. Це спрощує схему та код. Далі виконується синтез схеми автомата Мура за класичною методикою, для зберігання станів використовуються тригери D-типу [8–9]. Тестування схеми пристрою було виконано в пакеті ActiveHDL, що підтвердило коректність роботи. Етап тестування є обов'язковим і перехідним між розробкою та прошивкою готової схеми пристрою до реальної мікросхеми.

З метою аналізу функціонування розробленої апаратної частини експериментальне дослідження апаратного блоку, тобто акселератора, проводилося окремо від програмно реалізованого блоку [11–13]. У зв'язку з цим вхідні дані для експериментів – масиви кодів у форматі слова внутрішньої пам'яті – генеруються за допомогою консольної програми мовою C++, яка дозволяє створювати *.mif файли необхідного синтаксису для ініціації вхідних масивів у додатку Quartus II 9.2 Web Edition.

Необхідно перевірити коректність роботи паралелізму – виконати тестування послідовної і паралельної задач, порівняти час виконання. Отримані часові діаграми при реалізації на мікросхемі Cyclone III показали, що час роботи пристрою із одним тестовим масивом (662 us) приблизно дорівнює часу роботи із чотирма масивами (670 us). Це свідчить про те, що схема акселератора працює паралельно на чотирьох наборах вхідних даних. Невелика різниця в часі пояснюється витратами на пересилки даних і керуючих сигналів через збільшення загального обсягу схеми, а також різне внутрішнє розташування блоків на кристалі FPGA. Результати тестування свідчать про коректність функціонування акселератора, що є необхідною умовою для подальших досліджень.

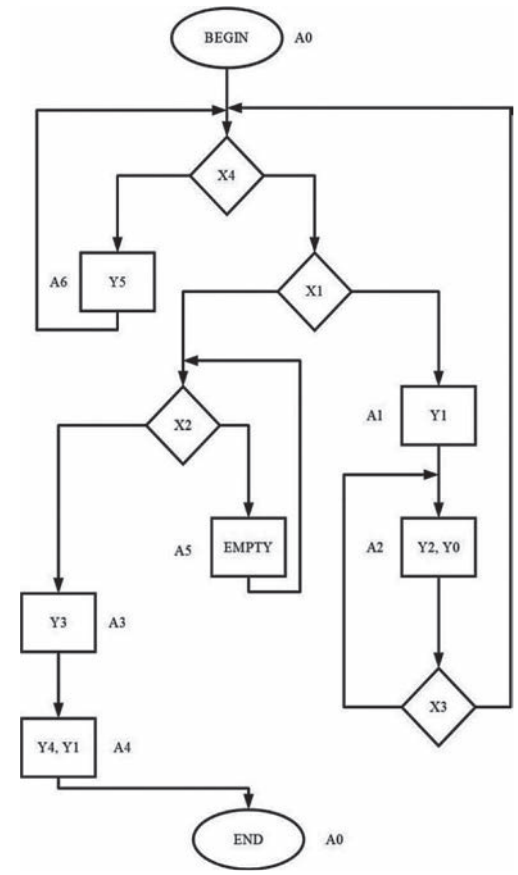

Рис. 6. Розмічена ГСА для автомату Мура

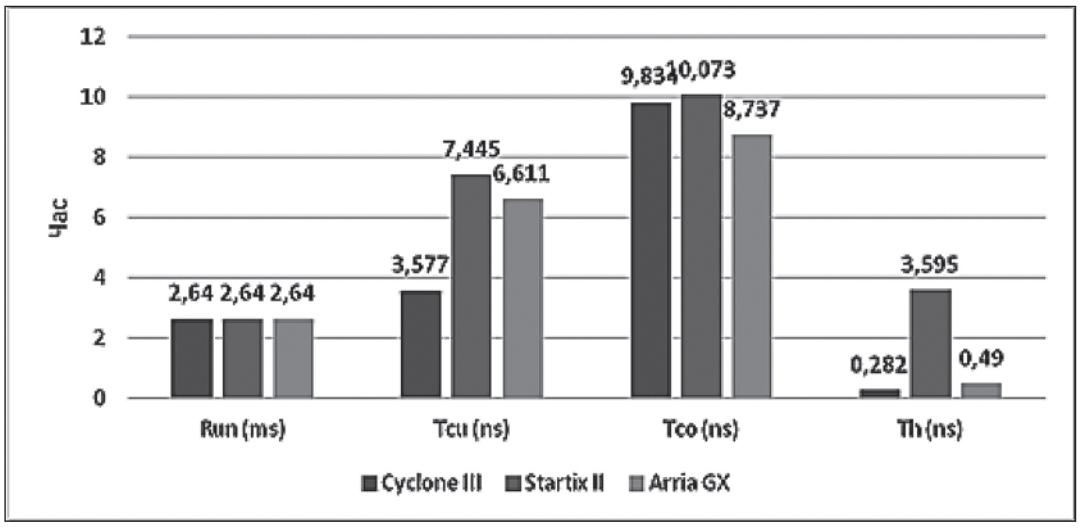

Рис. 7. Діаграма часових витрат недорогих серій мікросхем різних цінових категорій фірми Altera

За допомогою програмного продукту Quartus II 9.2 Web Edition можна проводити імплементацію в різні мікросхеми, отримувати часові характеристики роботи пристрою, апаратурні витрати, температури та на основі проведених досліджень обрати найбільш підходящу мікросхему для прошивки за критерієм «ціна-якість» [11]. Проведені тестування апаратурних та часових витрат для мікросхем фірми Altera / Intel в різних цінових категоріях. Офіційний поділ на цінові сегменти виглядав так: дешевий сегмент – Cyclone, середній – Arria, Stratix – дорогий сегмент, а також необхідно зазначити, що в кожному ціновому сегменті є дорогі, дешеві та середні мікросхеми [13]. Сегментація відповідає логічній ємності мікросхем.

Результати, представлені на Рис. 7 та в Табл. 1–2, отримані за допомогою генерування звітів про імплементацію з використанням стандартних компонентів EDA Quartus.

Аналіз отриманих результатів показав, що з практичних міркувань особливий інтерес викликають мікросхеми серії Cyclone III, тому що вони належать до недорогої лінійки, але в межах цих досліджень забезпечують кращі характеристики системи, ніж середня та дорога лінійки.

Висновки. Основна ідея паралельності в FPGA полягає у створенні незалежних блоків, що виконують незалежні функції, які потім необхідно трактувати від одного синхроімпульсу [14]. Необхідно створити чотири RAM блоки (обрана мікросхема не дозволяє більше) для зберігання даних, чотири генератори адрес, чотири помножувачі-акумулятори, але один компаратор. Над усіма цими пристроями необхідний керуючий пристрій для забезпечення взаємодії, ним є один керуючий автомат. Процес створення паралельної схеми є точно таким же, як і для послідовної.

Особливістю цього пристрою є те, що є можливість вирішувати й обернену задачу, тобто пошук для одного тексту найбільш підходящої теми з декількох можливих. Для цього необхідно внести відповідні зміни у схему пристрою, а саме керуючий автомат залишається таким же, але треба додати ще три внутрішні ключові масиви, видалити три пристрої для вхідних даних та змінити порядок підключення функціональних блоків. Необхідно зазначити, що в загальному випадку кількість вхідних масивів може бути довільною, бо це залежить від характеристик обраної мікросхеми FPGA, особливо від її логічної ємності.

Задача апаратного прискорення класифікації шляхом розпаралелювання процесів на мікросхемі FPGA є вирішеною в загальному сенсі. Перспективи подальшого розвитку досліджень в цьому напрямі полягають в апаратній реалізації на одній мікросхемі всього комплексу, включаючи функції програмного блоку. Також архітектуру прискорювача можливо вдосконалювати, наприклад шляхом застосування конвеєрів, суміщених моделей мікропрограмних автоматів, переміщенням схеми керуючого автомату до внутрішньої пам'яті [15], що потенціально може призвести до ще більшого прискорення функціонування пристрою.

Таблиця 1

Апаратурні витрати недорогих серій мікросхем різних цінових категорій фірми Altera

Family	LE	Rg	Pins	Mem (kbit)	Multipliers (шт.)	DSP (шт.)
Cyclone III	577	312	236	122.88	4	0
Startix II	295	120	236	122.88	0	16
Arria GX	298	120	236	122.88	0	16

Таблиця 2

Максимально допустимі частоти для проекту різних цінових категорій мікросхем фірми Altera (МГц)

Family	Fmax (low cost)	Fmax (middle cost)	Fmax (expensive)
Cyclone III	162.25	132.73	150.42
Arria GX	156.18	132.73	150.16
Startix II	149.01	130.5	133

Список літератури:

1. Rubin T.N., Chambers A., Smyth P., Steyvers M. Statistical topic models for multilabel document classification. *Machine Learning.* 2012. Vol. 88. No. 1–2. P. 157–208.
2. Mbaikodzi E., Dral' A.A., Sochenko I.V. The method of automatic classification of short text messages. *Information technologies and computer systems*. 2012. Vol. 3. P. 93–102.
3. Yampolsky L.S. Analytical approach to the choice of neural network topologies to solve the applied problems. *Adaptive systems of automatic control*. 2012. Vol. 20. P. 159–179.
4. Thangaraj M., Sivakami M. Text classification techniques: A literature review. *Interdisciplinary Journal of Information, Knowledge, and Management*. 2018. Vol. 13. P. 117–135.
5. Голуб Т.В., Тягунова М.Ю. Метод уменьшения размера вектора термов для классификации текстовых документов по категориям. *Проблемы региональной энергетики (Problemele energeticii regionale) (специальный выпуск)*. 2019. (SI) 2019. № 1–2(41). С. 84–94. DOI: 10.5281/zenodo.3240216.
6. Brindha S., Sukumaran S., Prabha K. A survey on classification techniques for text mining. *Proceedings of the 3rd International Conference on Advanced Computing and Communication Systems.* IEEE. Coimbatore, Indi. 2016. Available at: https://doi.org/10.1109/ICACCS.2016.7586371.
7. Соловьев В.Д. Один подход к классификации текстов на основе нейронных сетей. *Исслед. по информ.*, 2004. Выпуск 7. С. 21–28.
8. Baranov S. Logic and System Desing of Digital Systems. Tallinn, 2008. 267 p.
9. Sklyarov V., Sklyarova I., Barkalov A., Titarenko L. Synthesis and Optimization of FPGA – based Systems. Berlin, 2014. 432 p.
10. Склярова И., Скляров В. Аппаратные ускорители на базе FPGA. Конспект лекций по электротехнике. Springer Nature Switzerland. 2019. P. 566. https://doi.org/10.1007/978-3-030-20721-2_2.
11. Описание САПР Quartus II. Основные этапы проектирования СБИС ПЛ – Формирователь OFDM сигнала на плис стандарта 802.16d. URL: http://www.tnu.in.ua/study/refs/d79/file156314.html (дата звернення: 20.02.2020).
12. Іванець С., Зубань Ю., Казимир В., Литвинов В. Проектування комп'ютерних систем на основі мікросхем програмованої логіки. Суми, 2007. 312 с.
13. Altera documentation. 2018. URL: www.altera.com/support/literature/lit- index.html (дата звернення: 20.02.2020).
14. Краткий курс HDL. Часть 11. Асинхронные частоты, пересечение клоковых доменов и синхронизация URL: https://www.kit-e.ru/articles/circuit/2009_02_102.php (дата звернення: 20.02.2020).
15. Баркалов А.А., Титаренко Л.А., Зеленева И.Я., Грушко С.С. Исследование автомата с программируемой логикой в составе цифровой информационно-управляющей системы на FPGA (Research of the Finite State Machine with Programmable Logic as a Part of Digital Information and Control System based on FPGA). *Проблемы региональной энергетики (Problemele energeticii regionale) (специальный выпуск)*. 2019. № 1–2. С. 36–45.

Golub T.V., Zeleneva I.Ya., Hrushko S.S., Pavlishyn M.A. THE SUBSYSTEM OF HARDWARE ACCELERATION OF TEXT CLASSIFICATION IN THE FPGA BASIS

Computer tools in the field of natural language processing and particularly in the problems of text classification, are used quite intensively in our time. The actual goal of research in this area is to accelerate the classification process of text documents quite significantly. One way to solve such problems is hardware acceleration, that is, the transfer of part of the computational functions from the software domain to the hardware implementation. The paper proposes a method of hardware acceleration for the process of text documents classifying based on the principles of the naive Bayes method. The process of hardware classification is studied on the example of four input arrays that contain codes of words from different texts. To determine a specific topic, an array of relevant keywords is used that were previously encoded in the same way as the input arrays.

The problem is solved in the modern hardware basis of FPGA by parallelizing computations within a single chip. The developed hardware subsystem for acceleration of classification provides the possibility of parallel processing of several texts, as well as scalability and reprogramming, that is, settings for a given topic. The algorithm of operation and the structural diagram of a hardware accelerator is given, as well as a description of the functioning logic for each of the main components is given.

In order to test the designed device in the basis of widespread use, Altera / Intel FPGAs and the VHDL hardware description language were selected. To implement the project, the IDE Quartus II was selected, which corresponds to the products of the microcircuit manufacturer. The analysis of the time and hardware characteristics of the device due to its implementation on FPGAs from Altera / Intel in various price ranges is performed. Recommendations on the choice of microcircuit are formulated for the effective implementation of the accelerator according to the criterion of the "price / quality" ratio. Promising areas for further search of solutions in the field of hardware acceleration for text classification process are identified.

Key words: *hardware parallelization of calculations, FPGA, LUT, embedded memory, text classification.*