ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

УДК 004.8:004.312.466 **doi: 10.32620/aktt.2023.6.12**

А. Є. ПЕРЕПЕЛИЦИН

Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна

ЗАБЕЗПЕЧЕННЯ УПРАВЛІННЯ ЦИФРОВИМИ ПРАВАМИ ДЛЯ СТВОРЕННЯ ШТУЧНОГО ІНТЕЛЕКТУ ЯК СЕРВІСУ НА ОСНОВІ FPGA РЕАЛІЗАЦІЇ

Предметом *вивчення в даній статті є сучасні технології FPGA для створення проєктів і надання в якості сервісів, а також засоби для забезпечення управління цифровими правами для окремих рішень.* ***Метою*** *роботи є аналіз і вдосконалення технологій для забезпечення можливості ліцензування та запобігання несанкціонованого виконання IP-ядер для реалізації систем штучно інтелекту (ШІ) в FPGA.* ***Завдання:*** *проаналізувати процес розроблення систем штучного інтелекту і надання в якості сервісів з використанням FPGA; виконати аналіз можливості реалізації механізму оренди реалізації сервісу штучного інтелекту в FPGA та ліцензування окремих його екземплярів; провести аналіз можливостей захисту коду опису реалізації механізмів оренди сервісу безпосередньо в FPGA; провести дослідження можливих джерел унікальності окремих екземплярів системи або мікросхеми FPGA; навести практичний приклад застосування механізмів забезпечення управління цифровими правами для створення сервісу штучного інтелекту в FPGA проєкті. Відповідно до поставлених завдань, були отримані наступні* ***результати****. Виконано аналіз технологічних можливостей, засобів і середовищ розроблення, мов опису та програмування для створення систем штучно інтелекту у формі сервісу з апаратною реалізацією. Проаналізовано складові частини кожного рівня реалізації сервісу ШІ на базі FPGA. Проведено аналіз можливих напрямків забезпечення управління цифровими правами для проєктів сервісів ШІ в FPGA. Проаналізовано можливість захисту коду опису апаратних рішень для можливості розповсюдження з використанням стандарту шифрування IEEE-1735. Обговорюється інтеграція стандарту шифрування із середовищами розроблення. Виконаний пошук варіантів та можливих рішень для забезпечення прив'язки проєкта до конкретного екземпляра чіпа FPGA. Виконано практичне дослідження роботи існуючого рішення від компанії Accelize для управління цифровими правами IP-ядер проєктів для FPGA.* ***Висновки:*** *Головний внесок і наукова новизна отриманих результатів полягає в наведених результатах пошуку можливих джерел відмінностей та елементів унікальності FPGA для ідентифікації окремого екземпляра системи. Запропонована обробка даних від реалізації фізично неклонованих функцій (ФНФ) в FPGA для забезпечення управління цифровими правами є принципово новою. Наведені зв'язки між рівнями реалізації ШІ як сервісу на основі FPGA показують ієрархію компонентів при побудові системи.*

Ключові слова: *DRM; FPGA; ПЛІС; FPGA як сервіс; штучний інтелект; ШІ як сервіс; хмарні сервіси; фізично неклоновані функції; середовища розроблення; Accelize; стандарт шифрування IEEE-1735.*

Вступ

Швидкий розвиток цифрових обчислень і хмарних технологій привів до нового способу споживання та обміну послугами за допомогою Інтернету. І якщо на самому початку це були досить прості типи послуг або сервісів, то зараз вони стають дедалі складнішими з вдосконаленими функціями, такими як обробка зображень та використання елементів ШІ [1].

Під час розповсюдження та розгортання хмарних сервісів необхідно гарантувати, що інтелектуальна власність достатньо захищена від нелегального доступу та використання [2].

Цього можна досягти на кількох рівнях при побудові хмарного сервісу залежно від архітектурних рівнів і залучених ресурсів [3].

Розглядаючи технології FPGA та їх широке використання для побудови на основі FPGA (FaaS) орієнтовані на сервіси ШІ і централізовані послуги, існує потреба в управлінні цифровими правами окремих екземплярів бібліотечних компонентів або IP-cores які використовуються.

Дуже багато продуктів випускає Xilinx для застосування у проєктах ШІ [4]. Існуючі карти-прискорювачі U280, U50 і U55C які позиціонуються як рішення для ШІ.

Це прискорювачі для створення проєктів ШІ, які не можуть бути ефективно реалізовані виключно програмно. До половини можливих застосувань таких прискорювачів пов'язано з ШІ.

Така можливість передбачається під час розгляду методу побудови рішень у формі FPGA as a Service (FaaS) [5].

Таке рішення дозволяє змінити парадигму програмування FPGA, а також передбачає програмування як деяку функцію. З погляду користувача це певна функція, що викликається, зокрема дистанційно, яка може бути використана і до неї здійснюється доступ.

Застосування FaaS це передовий край сучасної розробки проєктів для FPGA. Використання можливостей прискорення обчислень із використанням апаратної реалізації та проєктування для побудови систем штучного інтелекту дозволяє перейти до побудови сервісів ШІ на основі FPGA.

Загальною тенденцією є створення сервісів, які надають послуги дистанційно. Такі складні рішення ШІ на основі FPGA можуть бути реалізовані як web-сервіс, що надає обчислювальні ресурси. У зв'язку із цим стає актуальним питання ліцензування таких рішень.

З іншого боку, існує можливість продажу бібліотечних компонентів таких рішень із збереженням можливості контролю їх використання. Для цього може бути застосований механізм управління цифровими правами або DRM (Digital Rights Management), що дозволяє захищати екземпляри ШІ рішень на основі FPGA від несанкціонованого використання.

Застосування стандарт IEEE-1735 надає можливість захищати компоненти проєктів на VHDL, Verilog і System Verilog завдяки шифруванню [6].

Аналіз та застосування таких механізмів захисту апаратних реалізацій сервісів штучного інтелекту для FPGA становлять великий інтерес.

Метою даної роботи є аналіз і вдосконалення технологій для забезпечення можливості ліцензування та запобігання несанкціонованого виконання IP-ядер для реалізації систем штучно інтелекту в FPGA

Для досягнення поставленої мети, в рамках даної роботи, необхідно вирішити наступні **задачі**:

1) проаналізувати розроблення систем штучного інтелекту і надання в якості сервісів з використанням FPGA;

2) виконати аналіз можливості реалізації механізму оренди реалізації сервісу штучного інтелекту в FPGA та ліцензування окремих його екземплярів;

3) провести аналіз можливостей захисту коду опису реалізації механізмів оренди сервісу безпосередньо в FPGA;

4) провести дослідження можливих джерел унікальності окремих екземплярів системи або чіпа FPGA;

5) навести практичний приклад застосування механізмів забезпечення управління цифровими правами для створення проєкту сервісу штучного інтелекту в FPGA.

1. Аналіз розроблення сервісів штучного інтелекту з використанням FPGA

Удосконалення реалізації систем штучного інтелекту з урахуванням описаних особливостей потребує високопродуктивних та енергоефективних рішень. Виробники процесорів додають до своїх процесорів спеціальні регістри та апаратні блоки для підтримки базових обчислень, що використовуються при реалізації нейронних мереж і ще більших рішень штучного інтелекту.

Підтримка даного виду обчислень на апаратному рівні дозволяє підвищити продуктивність. Це важливо, оскільки частка завдань ШІ в загальній кількості серверних завдань невпинно збільшується.

Використання технології FPGA для апаратної реалізації систем ШІ відкриває можливість значного підвищення продуктивності [7]. Це дозволяє ефективно вирішувати поставлені завдання з використанням сервісу. Без апаратного прискорення це не надто ефективно через велику тривалість обчислень одного набору даних центральним процесором [8]. Але ж більша частина сервісів зараз базується на звичайних процесорах або графічних процесорах.

Незважаючи на це, всі ці стандартні апаратні архітектури не забезпечують необхідної динаміки зростання продуктивності обчислень, в порівнянні з FPGA. Перевагою реалізації FPGA є можливість паралельного виконання однотипних обчислень. Також можливе використання конвеєрної обробки операцій на основі набору інструкцій конкретного проєкту. Така можливість створення виділеного прискорювача для вирішення специализованих завдань сервісом дозволяє розглядати технологію FPGA як ефективне цільове обладнання для реалізації сервісів з інтенсивними обчисленнями [9].

Реалізація специфічної архітектури необхідна для побудови ефективного ШІ як сервісу. Також потрібна можливість швидкого обміну даними з програмною частиною і набором програмних рішень, які безпосередньо відповідають за взаємодію з користувачем сервісу. Це стосується всіх рішень ШІ в масштабі проєктів від локальних до хмарних.

У реалізаціях в FPGA використовується спеціалізована архітектура ШІ з урахуванням конкретного завдання, а також динамічна пам'ять для зберігання даних і обміну результатами з хост-додатком. Для підвищення продуктивності всі операції реалізуються як апаратна логіка. Це змінює набір типів даних в проекті з адаптацією їх для конкретного завдання.

Карти прискорювачі на основі FPGA це спеціальні плати, які забезпечують значне збільшення продуктивності за рахунок паралельної реалізації великої кількості операцій конкретного завдання. Швидкий зв'язок з FPGA здійснює інтерфейс PCIe.

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

Карти-прискорювачі Alveo FPGA є формою потужної плати FPGA з швидким інтерфейсом і достатньою кількістю динамічної пам'яті для рішень ШІ. Додавання таких карт прискорювачів FPGA в датацентрах дозволяє організувати сервіс FPGA [10]. Забезпечення доступу до чипу FPGA з перепрограмуванням під час виконання допомагає реалізувати FPGA як сервіс для конкретного завдання [11].

Для програміста сервісу штучного інтелекту, що працює на хост-комп'ютері, кожен прискорювач в FPGA доступний безпосередньо з хост-програми на C++ як виклик функції. Задача для обчислення може бути розділена на частини для послідовного обчислення за допомогою одного або паралельного запуску декількох потоків на підключених до хост-комп'ютера картах прискорювачів FPGA.

Кожне виконання завдання в FPGA триває не більше декількох секунд. Це обмеження фреймворку XRT для комунікації по PCIe від Xilinx для спрощення зв'язку з ядрами в FPGA через PCIe [12].

При цьому один хост-комп'ютер містить до 8 слотів з картами прискорювача FPGA [3]. Всі вони підключені до хост-комп'ютера і всі разом можуть працювати над одним і тим же завданням користувача. Вони стають однією спеціалізованою логікою, яку можна запрограмувати на виконання одного завдання в рамках всього центру обробки даних.

Цей тип прискорення може бути використаний в більшості областей, де потрібна обробка даних. Ефективність обробки даних на основі FPGA набагато вища, ніж для графічних процесорів або процесорів загального призначення. Перевага продуктивності FPGA для додатків штучного інтелекту базується на здатності створювати власну архітектуру всередині мікросхеми FPGA. Це допомагає вирішувати типові завдання набагато швидше з однаковим або меншим енергоспоживанням [13].

Розробка реалізацій апаратних прискорювачів під час побудови сервісу проєктування є важливою частиною створення таких систем. Використання автоматизованих середовищ розроблення, таких як Vitis, у поєднанні з набором фреймворків для побудови систем ШІ може значно скоротити трудовитрати під час розробки та підтримки таких проєктів.

Використання мов, які зазвичай непридатні для опису рішень на основі FPGA, надає можливість застосовувати C++, OpenCL і Python щоб зменшити складність входження в процес проєктування сервісу і залучити розробників з навичками створення систем ШІ і без досвіду FPGA до ШІ. Також можна використовувати фреймворкі TensorFlow і Caffe.

Ці описи будуть перетворені у внутрішній файл проєкту, який буде створений Vivado під час компіляції проєкту у Vitis. Після генерації він буде скомпільований звичайним способом, як проєкти RTL.

Три останні технології були додані безпосередньо в середовищі розроблення Vitis [14]. Більшість цих мов і технологій придатні для впровадження систем штучного інтелекту, оскільки більшість бібліотек для вирішення завдань ШІ написані з використанням тих же мов.

Середовище розроблення Vitis включає в себе можливість використовувати в процесі розробки оптимізатор, квантування станів елементів та компілятор системи штучного інтелекту.

Процес побудови ШІ з FPGA як сервісу спрощується завдяки використанню IP-ядер з набором моделей, що передбачають налаштування, які часто використовуються при створенні рішень ШІ [15].

Квантування є першим кроком і найважливішим процесом оптимізації проєкту ШІ, який здійснюється. шляхом переходу від представлення системи з рухомою точкою до реалізації меншої точністю на основі вимог конкретної підзадачі. Цей крок дозволяє зменшити кількість задіяних бітів у представленні вузла нейронної мережі. Цей процес можна автоматизувати за допомогою інструментів розроблення [16].

Цей процес оптимізації робить один із найбільш значних внесків у збільшення щільності паралельних вузлів нейронної мережі в реалізації FPGA. Наступним кроком оптимізації є відсікання низькопріоритетних гілок, яка передбачає відсікання деяких гілок моделі із незначним впливом на кінцевий результат. Далі виконується побудова проєкту.

Структура всіх рівнів реалізації штучного інтелекту на основі FPGA як сервісу включає карти прискорювача з мікросхемами FPGA, підключеними до головного комп'ютера через інтерфейс PCIe. Програмна частина сервісу працює на хост-комп'ютері і спілкується з картами акселератора, щоб поставити завдання в чіпи і отримати результати від ядер в FPGA (рис. 1). Усе необхідне для здійснення прискорення ШІ працює на цьому рівні.

Зв'язки між рівнями представлення реалізації ШІ як сервіс (або AI as a Service) на основі FPGA показують паралельне існування кількох екземплярів у рамках одного хост-комп'ютера.

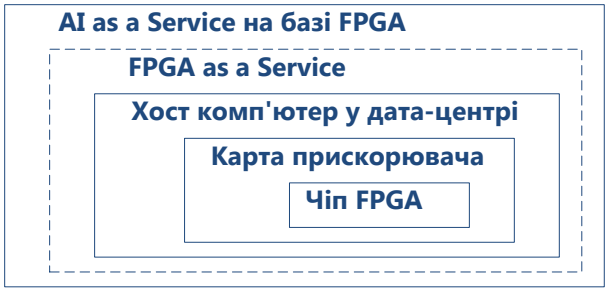

Рис. 1. Рівні представлення реалізації ШІ як сервіс на основі FPGA

2. Аналіз можливості реалізації оренди сервісу штучного інтелекту в FPGA та ліцензування окремих екземплярів

Проблематика створення таких систем ШІ на основі FPGA є дуже актуальною, тому аналіз можливих реалізації механізму оренди сервісів або бібліотечних ядер становить великий інтерес.

У зв'язку з тим, що модель роботи сервісу передбачає покупку або оренду серверного обладнання з доступом до ресурсів FPGA у вигляді окремих плат або можливість розгортання набору обладнання у конкретного користувача для здійснення його безпосередніх завдань, існує необхідність організації ліцензійного доступу до такого роду рішень і захисту цих рішень для можливості централізованого контролю їх використання.

Ліцензування є сильною альтернативою здавання сервісу в оренду. У цьому випадку користувачі можуть самостійно запускати рішення на власному обладнанні. Наприклад, якщо є багато користувачів системи і одному надається таке рішення, то він може почати перепродувати можливість доступу іншим. Тому необхідний механізм прив'язки ліцензії до кожного екземпляра такого FPGA для оренда готового рішення.

Оренда коду проєкта є альтернативою та базується на прив'язці до екземпляра, або заснована на прив'язці до іншого сервісу, до якого ядро постійно звертається. Це може бути реалізовано у вигляді бібліотечного компонента.

Це означає, що для проєкту ШІ ще на етапі створення існує можливість вбудувати механізми захисту. Коли користувачу надається IP-ядро з елементами коду, важливо виключити можливість неконтрольованого використання проєкту з використанням такого механізму захисту.

З іншого боку, клієнтський екземпляр повинен звертатися до сервера для перевірки ліцензії. Сервер має бути довіреним, щоб таке звернення не розглядалося як підозріле.

Використання популярних і довірених рішень, які підтримуються фірмою виробником і можуть бути представлені в його магазині або власних репозиторіях також є альтернативним способом забезпечення захисту рішень, пов'язаних з реалізацією програмної частини сервісу в конкретних макетних платах FPGA.

Такі рішення використовують деякі унікальні ключі екземплярів та ключі окремих сесій, які формуються та передаються ядрам, які виконуються всередині карти прискорювача FPGA для можливості активації логіки, що знаходиться у складі кожного ядра та дозволу виконання функцій такого сервісу для кожної сесії зв'язку.

3. Аналіз можливостей захисту коду опису реалізації механізмів оренди сервісу безпосередньо в FPGA

Таким чином, коли створюється комерційний сервіс, існує необхідність тимчасово комусь здати ядро в оренду. Наприклад, розроблено алгоритм стиснення зображення або детектування об'єктів на картинці або потоковому відео. Це рішення здається в оренду. Екземпляр рішення необхідно захищати від піратського використання.

IP-ядро можна здати в оренду, і коли закінчується ліцензія, можна її відкликати. У такому разі продукт для використання доступний користувачу не назавжди, а лише на період, щоб скористатися ним.

Стандарт шифрування IEEE-1735 надає можливість додавати захищені шифруванням компоненти в проєкти іншими розробниками [6]. Ці компоненти додаються до проєкту у вигляді набору файлів, які заплутані та зашифровані. Це опис RTL, що включає System Verilog, Verilog або VHDL файли.

Якщо відкрити такий файл, там буде лише набір зашифрованих символів. Середовище розроблення розшифровує цей код [17].

Існує централізоване зберігання ключів. У середовищі розроблення можна отримати RTL схемотехнічне уявлення, як побудовані ці компоненти. Існує можливість подивитися їхню структуру.

Це відкриває опосередковану можливість їхнього аналізу, зворотної розробки та відтворення. Але це не спростить процес зворотної розробки і не вкаже в явному вигляді механізм їхньої роботи, тому такий захист перешкоджає їхній компрометації.

Використання кінцевого автомата з великою кількістю станів це ще одна можливість реалізації контролю виконання ядра в FPGA. Зв'язка хост-програми та ядра всередині програми та їх обмін по послідовному каналу дозволяють використовувати підмішування одного або двох бітів у відправку та відповідь назад.

Використання регістру зсуву великої розрядності, який є частиною великого FSM з великим числом станів, дозволяє реалізувати складне кодування та управління виконанням.

Такий кінцевий автомат у разі непопадання цих бітів в очікувану послідовність просто блокує обмін і забезпечує неможливість запуску цього рішення за рахунок складної функції, яка буде працювати над послідовностями, що передаються з використанням генераторів псевдовипадкових послідовностей.

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

4. Відшукання можливих джерел унікальності окремих екземплярів системи або чіпа FPGA

Крім існуючих DRM для FPGA актуальним є пошук інших заходів захисту, які передбачають можливість використовувати унікальні ознаки або відбитки у складі кожної плати або чіпа.

Ця інформація може дозволяти автономну роботу або реалізацію захисту з можливістю передачі інформації за допомогою якогось хост рішення на централізований сервер для обробки.

В цьому випадку існує необхідність організації заходів захисту таких рішень, які можуть використовувати вбудовані засоби. Тому важливо знайти існуючі варіанти забезпечення ідентифікації екземпляра, і проаналізувати шляхи реалізації унікальності на рівні фізичних джерел відмінностей та можливості прив'язуватися до цих відмінностей. Знаходження варіантів та джерел відмінностей для забезпечення прив'язки до конкретного екземпляра мікросхеми дозволить реалізувати захист окремих IP-ядер.

Унікальний DNA код містять у собі багато FPGA мікросхем Xilinx. Виявляється, в сучасних мікросхемах може бути кілька частин і кожна має свій унікальний код [18]. Цей код можна прочитати апаратно, є спеціалізовані компоненти-порти, якими можна прочитати значення. Доступ до цих значень можливий за допомогою компонента DNA_PORT. Це унікальний номер для кожного елемента кристала у складі такої макетної плати або карти прискорювача. У деяких корпусах мікросхем міститься до трьох окремих частин із власними кодами. У складі окремих корпусів міститься кілька незалежних кремнієвих підкладок, кожна з яких містить свій унікальний номер DNA.

32-бітний код екземпляра бінарного файлу, що задається розробником і фіксується під час компіляції та генерації файлу бінарного контейнера у Vivado. Доступ до цього коду можливий із проєкту в FPGA за допомогою компонента USR_ACCESS. При цьому процес генерації одного файлу є досить трудомістким, тому такий код може бути застосований швидше для розрізнення фалів в рамках версії, а не окремих екземплярів.

Фізично неклоновані функції (ФНФ) дозволяють відрізнити один чіп від іншого завдяки максимізації або посиленню впливу дефектів виробництва як відмінності екземпляра [19]. Хоча такі FPGA функційно ідентичні, у них відрізняються параметри затримки, що впливає на швидкість поширення окремих сигналів. Якщо максимізувати цей ефект за допомогою позитивного зворотного зв'язку, це можна використовувати як параметр, щоб відрізняти один чіп від іншого.

Результати аналізу можливих елементів та джерел унікальності представлені у таблиці 1. До таких можливих джерел, на додаток до названих, належать не тільки внутрішньокристальні, а й елементи прив'язки до самої плати з чіпом FPGA. Як категорії порівняння взяті джерела унікальності, практична застосовність, зреалізація на стороні хост-додатка, реалізація на стороні FPGA і необхідність запуску такої реалізації для отримання інформації про унікальність.

Таблиця 1

Аналіз потенційних джерел унікальності для прив'язки ліцензії до конкретного екземпляра

Джерело відмінностей	Опис	Стійкість реалізації	Реалізація в хост-програми та/або в FPGA	Необхідний запуск FPGA
Особиста пошта, номер телефону	використання будь-яких даних користувача для ідентифікації	не стійка	хост-програма	ні
USB-ключ	надання ліцензії у вигляді унікального USB-ключа	стійка	хост-програма	ні
MAC адреса	використання адреси мережевого інтерфейсу в якості ключа	не стійка	хост-програма	ні
Ідентифікатор накопичувача	для ідентифікації можна взяти номер тома накопичувача	не стійка	хост-програма	ні
Системна плата	спільна конфігурація апаратних компонентів системи як ключ	не стійка	хост-програма	ні
Trusted Platform Module (TPM)	використання на платі захищеного модуля або елементу	стійка	хост-програма	ні
Номер карти прискорювача	використання унікального ідентифікатора прошивки карти	стійка	обидві	ні
DNA код	прив'язка екземпляра до ідентифікатора DNA в FPGA чіпі	стійка	обидві	необхідний
ФНФ	використання мікро дефектів кремнієвої підкладки в FPGA	не клонована	обидві	необхідний

Для ідентифікації завдань можна використовувати різні рішення. Існує можливість прив'язуватися до обладнання, наприклад, до MAC адреси мережевої плати на макетній платі, де є мережеві інтерфейси. Є можливість використовувати фізично неклоновані функції, як відбиток або ключ.

Аналіз показує, що ухвалення рішення про використання конкретного джерела унікальності або їх комбінації має бути зроблено на основі вимог та специфіки сервісу, що розробляється. Отримані результати порівняння спрощують процес прийняття рішення про вибір конкретного варіанта визначення унікальності.

5. Приклад застосування механізмів забезпечення управління цифровими правами для створення сервісу в FPGA

Для практичної реалізації контролю можливості виконання окремих екземплярів FPGA реалізації сервісу ШІ для аналізу зображень був обраний набір інструментів, запропонованих компанією Accelize [20].

Для роботи з цими рішеннями для захисту необхідний обліковий запис із набором інструментів адміністрування продуктів. За допомогою елементів меню створюється набір ключів, один з яких жорстко прописується всередині проєкту, а другий може бути використаний для можливості запуску екземпляра такого сервісу.

У цьому випадку другий ключ ідентичний до ключа користувача, який дозволяє використовувати сервіс після отримання ліцензії.

Компанія Accelize використовує свій централізований сервер для зв'язку з ядрами, що запускаються, і накладає обмеження на кількість екземплярів ядра, які можуть виконуватися одночасно для одного ключа.

Для можливості керування рішенням у FPGA необхідно додати в проєкт користувача окреме ядро kernel_drm_controller та компонент активації із зашифрованими файлами реалізації Aaccelize DRM для кожного окремого ядра в його складі.

Сам компонент активації top_drm_activator не зашифрований і здійснює взаємодію з окремим ядром kernel_drm_controller за декількома лініями спрощеного інтерфейсу AXI4 (рисунок 2).

Усередині ядра для забезпечення контролю також використовується компонент DNA_PORT для можливості прив'язки до конкретного екземпляра FPGA (рисунок 3).

Також на стороні проекту користувача в FPGA використовується додатково компонент USR_ACCESS для можливості відрізнити версію файлу бінарного контейнера (рисунок 4).

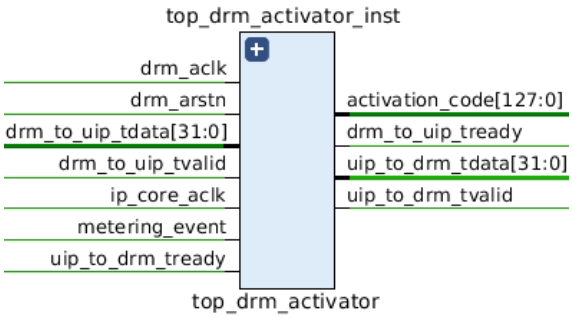

Рис. 2. Екземпляр компонента top_drm_activator

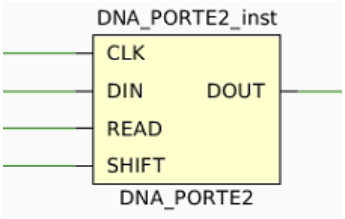

Рис. 3. Екземпляр компонента DNA_PORT

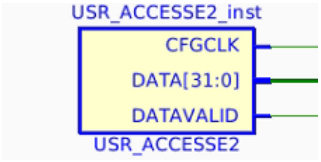

Рис. 4. Екземпляр компонента USR_ACCESS

Проєкт користувача в FPGA здійснює перевірку відповідності 128-бітного значення одержуваного ключа. Це значення може бути явно прописаним, проте його перевірку слід ускладнити і заплутати, щоб уникнути легкої компрометації такого ядра з використанням зворотної розробки.

Також екземпляр top_drm_activator дозволяє передавати користувачу сигнал metering_event від ядра для централізованого аналізу статистики використання. Кожен розробник приймає рішення, яка саме подія буде пов'язана із формуванням цього сигналу. Це може бути подія одноразового запуску ядра, або сигнал про успішне виконання довгого завдання, розбитого на окремі ітерації.

Взаємодія між хост-додатком з DRM ядром kernel_drm_controller всередині FPGA відбувається неявно та без участі користувача і підтримується для всіх типів ядер, у тому числі під керуванням XRT. Для можливості такої комунікації розробник повинен після компіляції проєкту знайти у супутніх файлах інформацію про базову адресу для ядра DRM контролера. Ця базова адреса використовується при написанні хост-додатку для кожного конкретного бінарного файлу користувача і передається в програмну частину екземпляра бібліотеки від цієї компанії.

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

Висновки

В рамках роботи проведено аналіз існуючих та перспективних напрямів забезпечення механізму прив'язки до конкретного екземпляру системи для забезпечення можливості ліцензування проєктів FPGA. Запропоновано ієрархічне представлення зв'язку окремих складових частин сервісу ШІ з апаратною реалізацією, в рамках якого показані всі рівні від мікросхеми FPGA до карти прискорювача, і до комп'ютера, де встановлений конкретний прискорювач. Карти прискорювачі на основі FPGA є спеціалізованими прискорювачами, які забезпечують значне підвищення продуктивності за рахунок паралельного виконання великої кількості завдань. Така реалізація дозволяє знизити енергоспоживання центрів обробки даних.

Оренда як готових рішень, так і оренда безпосередньо коду опису, може бути заснована на DNA, на використанні MAC адресу мережевої плати на макетної платі, або на прив'язці до якогось сервісу, до якого ядро постійно звертається.

Виявляється, в сучасних чіпах FPGA може бути кілька підкладок і кожна має свій унікальний код DNA, якій можна отримати апаратно, тому що є порти, якими можна прочитати це значення.

Для ідентифікації, в залежності від проєктів підходять різні способи, залежно від того, який тип проєкту використовується. Якщо проєкту не потрібний постійний доступ до мережі, то використовується один тип ліцензування, а якщо доступне безперервне з'єднання, то інший тип.

Крім цього, існують TPM, фізично неклоновані функції, які можна використовувати як відбиток або ключ. Цей механізм захисту може бути реалізований у вигляді бібліотечного компонента.

Розглянутий на практиці набір інструментів DRM компанії Accelize, що надає своє ядро в FPGA та набір бібліотек, що дозволяють контролювати доступ, показав, що цей інструмент дозволяє ефективно керувати можливістю використання окремих екземплярів.

При цьому, його застосування накладає обмеження на тактову частоту одного з двох доступних джерел тактування для ядра в FPGA. Така частота не повинна перевищувати 100 МГц, що є загальним недоліком при використанні таких компонентів, як DNA_PORT.

Аналіз показує, що ухвалення рішення про використання конкретного джерела унікальності або їх комбінації має бути зроблено на основі вимог та специфіки сервісу, що розробляється. Отримані результати зіставлення спрощують процес прийняття рішення про вибір конкретного варіанта визначення унікальності.

Використання існуючих DRM рішень для FPGA дозволяє здійснювати захист сервісів ШІ та надавати можливість моніторингу окремих актів використання таких рішень. Це дозволяє організовувати гнучкий процес експлуатації таких сервісів.

Серед напрямів подальших досліджень слід виділити реалізацію власних інструментів для забезпечення контролю використання з урахуванням отриманих результатів дослідження. Найбільш перспективними для реалізації можуть бути динамічна модифікація інтерфейсу ядра FPGA на основі деякого алгоритму в поєднанні з використанням генераторів псевдовипадкових послідовностей для використання спільно з обмінами даних. Усередині такого ядра може бути кінцевий автомат, який прийматиме рішення про активацію ядра на основі окремих бітів даних, що додаються до частини даних від хост-програми. Інформація про архітектуру та реалізацію кінцевого автомата може бути прихована шифруванням коду або багаторівневою обфускацією.

Література

1. Tetskyi, A. Neural networks based choice of tools for penetration testing of web applications [Text] / A. Tetskyi, V. Kharchenko, & D. Uzun // Proceedings 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018, – 2018. – P. 402-405. DOI: 10.1109/DESSERT.2018.8409167.

2. Tetskyi, A. Analysis of the Possibilities of Unauthorized Access in Content Management Systems Using Attack Trees [Text] / A. Tetskyi, V. Kharchenko, & D. Uzun // Proc. PhD Symposium at ICTERI 2018, Kyiv, Ukraine, May 14-17, 2018. – CEUR-WS, 2018. – Vol. 2122. – P. 16-25.

3. Perepelitsyn, A. Method of QoS evaluation of FPGA as a service [Text] / A. Perepelitsyn, V. Kulanov, & I. Zarizenko // Radioelectronic and Computer Systems. – 2022. – No. 4. – P. 153-160. DOI: 10.32620/reks.2022.4.12.

4. Vitis AI User Guide, Xilinx, UG1414 (v2.5) [Online]. – Available at: https://docs.xilinx.com/r/en-US/ug1414-vitis-ai. – 15.06.2022.

5. Perepelitsyn, A. Method of creation of FPGA based implementation of Artificial Intelligence as a Service [Text] / A. Perepelitsyn // Radioelectronic and Computer Systems. – 2023. – No. 3. – P. 27-36. DOI: 10.32620/reks.2023.3.03.

6. Intellectual Property. IP Encryption. AMD. [Online]. – Available at: https://www.xilinx.com/products /intellectual-property/ip-encryption.html. – 30.07.2023.

7. Kumari, B. A. S. FPGA Implementation of Neural Nets [Text] / B. A. S. Kumari, S. P. Kulkarni, & C. G. Sinchana // International Journal of Electronics and Telecommunications. – 2023. – Vol. 69, No. 3. – P. 599-604. DOI: 10.24425/ijet.2023.146513.

8. Технології реалізації штучного інтелекту як сервісу на основі апаратних прискорювачів [Текст] /

А. Є. Перепелицин, Є. В. Касапєн, Г. В. Фесенко, В. С. Харченко // Авіаційно-космічна техніка і технологія. – 2022. – № 6. – С. 57-65. DOI: 10.32620/aktt.2022.6.07.

9. Класифікація даних апаратними прискорювачами FPGA у центрах обробки даних та хмарах [Текст] / О. А. Ільяшов, К. В. Покора, В. О. Дяченко, А. А. Коваленко // Системи управління, навігації та зв'язку. – 2023. – № 2. С. 106-112. DOI: 10.26906/SUNZ.2023.2.106.

10. Alveo Product Selection Guide, Data Center Accelerator Cards, Xilinx. – Available at: https://www.xilinx.com/content/dam/xilinx/support/documents/selection-guides/alveo-product-selection-guide.pdf. – 28.02.2023.

11. Perepelitsyn, A. FPGA as a Service Solutions Development Strategy [Text] / A. Perepelitsyn, I. Zarizenko, & V. Kulanov // Proceedings 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies, DESSERT 2020. – 2020. – P. 376-380. DOI: 10.1109/DESSERT50317.2020.9125017.

12. Alveo U280 Data Center Accelerator Card User Guide, AMD, UG1314 [Online]. – Available at: https://docs.xilinx.com/r/en-US/ug1314-alveo-u280-reconfig-accel. – 28.02.2023.

13. Alveo U50 Data Center Accelerator Card Data Sheet, Xilinx, DS965 (v1.7.1) [Online]. – Available at: https://www.xilinx.com/content/dam/xilinx/support/documents/data_sheets/ds965-u50.pdf. – 19.05.2021.

14. Vitis HLS User Guide, Xilinx, UG1399 (v2021.1) [Online]. – Available at: https://docs.xilinx.com/r/en-US/ug1399-vitis-hls. – 5.08.2021.

15. AI Engine Kernel Coding Best Practices, Xilinx, UG1079 (v2022.2) [Online]. – Available at: https://docs.xilinx.com/r/en-US/ug1079-ai-engine-kernel-coding. – 19.10.2022.

16. Technological Stack for Implementation of AI as a Service based on Hardware Accelerators [Text] / A. Perepelitsyn, H. Fesenko, Y. Kasapien, & V. Kharchenko // Proceedings 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies, DESSERT 2022, – 2022. – 5 p. DOI: 10.1109/DESSERT58054.2022.10018615.

17. Vivado Design Suite User Guide. Creating and Packaging Custom IP, AMD, UG1118 (v2023.1) [Online]. – Available at: https://docs.xilinx.com/r/en-US/ug1118-vivado-creating-packaging-custom-ip. – 30.05.2023.

18. Peterson, Ed. Developing Tamper-Resistant Designs with UltraScale and UltraScale+ FPGAs, Xilinx, XAPP1098 (v1.4) [Online] / Ed. Peterson. – Available at: https://www.xilinx.com/content/dam/xilinx/support/documents/application_notes/xapp1098-tamper-resist-designs.pdf. – 05.06.2021.

19. Tan, T. H. drDRM: A PUF-Based Dynamically Reconfigurable DRM Mechanism for FPGA-Based Platform [Text] / T. H. Tan, C. Y. Ooi, & M. N. Marsono // 2018 Sixth International Symposium on Computing and Networking, CANDAR 2018, – 2018. – P. 194-200. DOI: 10.1109/CANDAR.2018.00034.

20. FPGA-Centric Software Acceleration Made Easy. [Online]. – Available at: https://www.accelize.com/blog-fpga-centric-software-acceleration-made-easy/. – 23.05.2022.

References

1. Tetskyi, A., Kharchenko, V., & Uzun, D. Neural networks based choice of tools for penetration testing of web applications. *Proceedings 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018*, 2018, pp. 402-405. DOI: 10.1109/DESSERT.2018.8409167.

2. Tetskyi, A., Kharchenko, V., & Uzun, D. Analysis of the Possibilities of Unauthorized Access in Content Management Systems Using Attack Trees. *Proc. PhD Symposium at ICTERI 2018*, Kyiv, Ukraine, May 14-17, 2018, CEUR-WS, vol. 2122, pp. 16-25.

3. Perepelitsyn, A., Kulanov, V., & Zarizenko, I. Method of QoS evaluation of FPGA as a service. *Radioelectronic and Computer Systems*, 2022, no. 4, pp. 153–160. DOI: 10.32620/reks.2022.4.12.

4. *Vitis AI User Guide, Xilinx, UG1414 (v2.5)*. Available at: https://docs.xilinx.com/r/en-US/ug1414-vitis-ai. (accessed June 15, 2022).

5. Perepelitsyn, A. Method of creation of FPGA based implementation of Artificial Intelligence as a Service. *Radioelektronni i komp'uterni sistemi – Radioelectronic and computer systems*, 2023, no. 3, pp. 27-36. DOI: 10.32620/reks.2023.3.03.

6. *Intellectual Property. IP Encryption. AMD.* Available at: https://www.xilinx.com/products/intellectual-property/ip-encryption.html. (accessed July 30, 2023).

7. Kumari, B. A. S., Kulkarni, S. P., & Sinchana, C. G. FPGA Implementation of Neural Nets. *International Journal of Electronics and Telecommunications*, 2023, vol. 69, no. 3, pp. 599-604 DOI: 10.24425/ijet.2023.146513.

8. Perepelitsyn, A., Kasapien, Y., Fesenko, H., & Kharchenko, V. Technologies for Implementing of Artificial Intelligence as a Service based on Hardware Accelerators. *Aviacijno-kosmicna tehnika i tehnologia – Aerospace technic and technology*, 2022, no. 6, pp. 57-65. DOI: 10.32620/aktt.2022.6.07.

9. Ilyashov, O., Pokora, K., Diachenko, V., & Kovalenko, A. Future of FPGA – accelerating computations in data processing centers and clouds. *Control, Navigation and Communication Systems*, 2023, no. 2, pp. 106–112. DOI: 10.26906/SUNZ.2023.2.106.

10. *Alveo Product Selection Guide, Data Center Accelerator Cards, Xilinx.* Available at: https://www.xilinx.com/content/dam/xilinx/support/documents/selection-guides/alveo-product-selection-guide.pdf. (accessed February 28, 2023).

11. Perepelitsyn, A., Zarizenko, I., & Kulanov, V. FPGA as a Service Solutions Development Strategy. *Proceedings 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies, DESSERT 2020*, 2020, pp. 376-380. DOI: 10.1109/DESSERT50317.2020.9125017.

12. *Alveo U280 Data Center Accelerator Card User Guide, Xilinx, UG1314 (v1.3).* Available at: https://docs.xilinx.com/r/en-US/ug1314-alveo-u280-reconfig-accel. (accessed February 28, 2023).

13. *Alveo U50 Data Center Accelerator Card Data Sheet, Xilinx*, DS965 (v1.7.1). Available at: https://

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

www.xilinx.com/content/dam/xilinx/support/documents/data_sheets/ds965-u50.pdf. (accessed May 19, 2021).

14. *Vitis HLS User Guide, Xilinx, UG1399 (v2021.1).* Available at: https://docs.xilinx.com/r/en-US/ug1399-vitis-hls. (accessed August 5, 2021).

15. *AI Engine Kernel Coding Best Practices, Xilinx, UG1079 (v2022.2).* Available at: https://docs.xilinx.com/r/en-US/ug1079-ai-engine-kernel-coding. (accessed October 19, 2022).

16. Perepelitsyn, A., Fesenko, H., Kasapien, Y., & Kharchenko, V. Technological Stack for Implementation of AI as a Service based on Hardware Accelerators. *Proceedings 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies, DESSERT 2022,* 2022. 5 p. DOI: 10.1109/DESSERT58054.2022.10018615.

17. *Vivado Design Suite User Guide. Creating and Packaging Custom IP, AMD, UG1118 (v2023.1).* Available at: https://docs.xilinx.com/r/en-US/ug1118-vivado-creating-packaging-custom-ip (accessed May 30, 2023).

18. Peterson, Ed. *Developing Tamper-Resistant Designs with UltraScale and UltraScale+ FPGAs, Xilinx, XAPP1098 (v1.4).* Available at: https://www.xilinx.com/content/dam/xilinx/support/documents/application_notes/xapp1098-tamper-resist-designs.pdf. (accessed June 05, 2021).

19. Tan, T. H., Ooi, C. Y., & Marsono, M. N. drDRM: A PUF-Based Dynamically Reconfigurable DRM Mechanism for FPGA-Based Platform. *2018 Sixth International Symposium on Computing and Networking, CANDAR 2018,* 2018. pp. 194-200. DOI: 10.1109/CANDAR.2018.00034.

20. *FPGA-Centric Software Acceleration Made Easy.* Available at: https://www.accelize.com/blog-fpga-centric-software-acceleration-made-easy (accessed May 23, 2022).

Надійшла до редакції 30.10.2023, розглянута на редколегії 20.11.2023

ENSURING OF DIGITAL RIGHTS MANAGEMENT OF FPGA BASED IMPLEMENTATION OF ARTIFICIAL INTELLIGENCE AS A SERVICE

Artem Perepelitsyn

The subject of study in this article is modern FPGA technologies for creation of projects and providing them as a services, as well as solutions for ensuring digital rights management of individual instances of the system. The **goal** is to analyze and improve technologies for ensuring the work of licensing mechanisms and prevention of unauthorized execution of IP-cores involved in the implementation of artificial intelligence (AI) systems in FPGA. **Task**: to analyze the process of prototyping artificial intelligence systems and providing them as a services using FPGA; to perform an analysis of the possibility of the implementation of a rental mechanism for artificial intelligence as a service based on FPGA with licensing of individual instances of the system; to analyze the possibilities of code protection for the description of the implementation of service rental mechanisms directly in the FPGA; to perform research of possible candidate elements that allow uniqueness identification of individual instances of the system or FPGA chip; and to provide practical example of the application of mechanisms for ensuring digital rights management for creation of artificial intelligence as a service project in FPGA. According to the tasks, the following **results** were obtained. The analysis of technological capabilities, tools and development environments, description and programming languages for the creation of artificial intelligence systems in the form of a service with hardware implementation is performed. The components of each level of FPGA-based AI service implementation are analyzed. An analysis of possible directions for ensuring digital rights management for AI services projects in FPGA is performed. The possibility of protecting the source code of the description of hardware solutions for the possibility of distribution using the IEEE-1735 encryption standard is analyzed. The integration of the encryption standard with the development environments is discussed. A search of options and possible solutions to ensure identification of the specific instance of the project based on FPGA chip is performed. A practical study of the use of the existing solution from Accelize company for ensuring digital rights management for IP-cores of projects for FPGA is performed. **Conclusions.** The main contribution and scientific novelty of the obtained results is in the provided results of the research of possible candidate elements that allow the detection of differences and identification of the FPGA uniqueness for the identification of the instance of the system. The proposed method of processing of data from the implementation of physical unclonable functions (PUF) in FPGA to ensure digital rights management is a principal new approach. The given relations between the levels of implementation of AI as a service based on FPGA show the hierarchy of components when building of such system.

Keywords: DRM; FPGA; PLD; FPGA as a Service; Artificial Intelligence; AI as a Service; cloud services; Physical Unclonable Functions; development environments; Accelize; IEEE 1735 Encryption Standard.

Перепелицин Артем Євгенович – канд. техн. наук, доц., доц. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Artem Perepelitsyn – PhD, Associate Professor at the Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.perepelitsyn@csn.khai.edu, ORCID: 0000-0002-5463-7889, Scopus Author ID: 56332607800.