DOI: https://doi.org/10.36910/6775-2524-0560-2024-55-05
УДК 004.021:621.391.8
Виннишин Олег Ярославович, спеціаліст
http://orcid.org/0009-0006-7113-1501
Київський національний університет імені Тараса Шевченка, м. Київ, Україна

КОМПЛЕКСНА МЕТОДИКА ОПТИМІЗАЦІЇ ПРОГРАМНОГО КОДУ VHDL ПРИ ПРОЕКТУВАННІ ПРОГРАМОВАНОЇ ІНТЕГРАЛЬНОЇ СХЕМИ

Виннишин О.Я. Комплексна методика оптимізації програмного коду VHDL при проектуванні програмованої інтегральної схеми. Проведено аналіз підходів, що використовуються при оптимізації інформаційної системи на основі програмованої інтегральної схеми, що описується програмними алгоритмами, розробленими з застосуванням засобів мови опису апаратури VHDL. Основні принципи оптимізації визначались через представлення логіки проекту та функціонального моделювання програмованої інтегральної схеми у відповідності до декларованого функціоналу та умов експлуатації. Перехід від високорівневого до низькорівневого представлення при проектуванні логічних блоків програмованої інтегральної схеми, що проводиться на етапі компіляції, дозволяє визначити кількість логічних елементів, а отже завдяки функціональному моделюванню оцінюється архітектура схеми та формуються методичні рекомендації. Застосування комплексної методики оптимізації на основі проведеного аналізу надає можливість збільшити швидкодію виконання поставленої задачі з застосування розробленої програмованої інтегральної схеми, уникнути ризики надмірного генерування логіки, спростити арифметичні операції та логічні вирази, видалити блоки програмного коду, що не виконуються внаслідок модифікації проекту, застосувати засоби паралелізація процесів, а також виконати структурування блоків програмного коду з метою зменшення ризику появи помилок при його компіляції. Наведений у рамках дослідження інструментарій включав у себе введення у програмний код додаткових змінних і констант, зменшення об'єму програмного коду у тілі циклу, видалення зайвих блоків програмного коду в умовних конструкціях, структурування арифметичних операцій та логічних виразів, а також належний поділ блоків програмного коду. Окрема задача полягає у оптимізації блоку арифметико-логічного пристрою програмованої інтегральної схеми, що надає можливість зменшити ресурсоємність, скоротити кількість логічних вентилів, зменшити ризик виникнення помилок за умови виникнення затримки при подачі сигналів операційного коду, а також налагодити спільне використання ресурсів для окремих складових логічних блоків. Розроблена методика, таким чином, стала основою для формування цілісного набору підходів по формуванню логічної схеми через оптимізацію програмного коду VHDL у відповідності до вимог, поставлених при проектуванні програмованих інтегральних схем
Ключові слова: програмована інтегральна схема, мова опису апаратури, функціональна модель, алгоритми оптимізації, арифметико-логічний пристрій, паралелізація процесів.

Vynnyshyn O. A Comprehensive Methodology for Optimizing Vhdl Program Code when Designing a Programmable Integrated Circuit. An analysis of approaches used in optimizing information systems based on field-programmable gate arrays described by software algorithms developed using VHDL is conducted. The fundamental principles of optimization were determined through the representation of the project logic and functional modeling of the FPGA in accordance with the declared functionality and operating conditions. The transition from high-level to low-level representation in the design of logical blocks of the FPGA, carried out at the compilation stage, allows for the determination of the number of logic elements. Consequently, functional modeling assesses the architecture of the circuit and forms methodological recommendations. The application of a comprehensive optimization methodology based on the conducted analysis enables the increase of task execution speed using the developed FPGA, avoids the risks of excessive logic generation, simplifies arithmetic operations and logical expressions, removes blocks of code that are not executed due to project modifications, applies process parallelization tools, and structures code blocks to reduce the risk of errors during compilation. The tools introduced within the study included the addition of variables and constants to the code, reduction of code volume within loop bodies, removal of unnecessary code blocks in conditional constructions, structuring of arithmetic operations and logical expressions, as well as appropriate partitioning of code blocks. A separate task lies in the optimization of the arithmetic logic unit block of the FPGA, which allows for the reduction of resource consumption, the reduction of the number of logic gates, the mitigation of error risks due to signal delay in opcode delivery, and the establishment of resource sharing among individual components of logical blocks. The developed methodology, thus, serves as the basis for forming a comprehensive set of approaches for generating a logical circuit through VHDL code optimization in accordance with the requirements set during FPGA design.
Keywords: FPGA, hardware description language, functional model, optimization algorithms, arithmetic logic unit, process parallelization.

Постановка проблеми. На сьогоднішній день високою актуальністю характеризується завдання організації комп'ютерних систем, що відповідає вимогам по інформаційній стійкості та рівню захищеності від зовнішніх загроз [1-4]. Зазначається, найбільш ефективно відповідне завдання може бути вирішено через проектування інформаційної системи на основі програмованої логічної інтегральної схеми (Programmable Logic Device; PLD). Необхідно зазначити, що впровадження PLD при організації масштабованих комп'ютерних систем характеризується адаптивністю до зміни набору задач завдяки перепрограмування базових функцій та відповідне конфігурування інформаційної системи [1, 2]. Це надає можливість побудувати гнучку систему

захисту, що базується на зміні алгоритмів обробки потокових даних та, зокрема, впровадженні криптографічних процедур [3, 4]. Водночас, стійкість системи і оптимізація розподілу навантаження на обчислювальний ресурс забезпечується шляхом паралелізації процесів при роботі у режимі реального часу [5, 6], впровадженні протоколів спільного використання логічних блоків [7, 8], а також інтеграції логічних блоків з іншими компонентами PLD [9, 10]. У якості окремого класу PLD можна виділити програмовані логічні інтегральні схеми, архітектура яких визначається як набір програмованих користувачем вентильних матриць (Field-Programmable Gate Arrays; FPGA). Можна вказати, що FPGA характеризуються високою складністю структури, що відповідає широкому набору функціональних можливостей як окремого логічного пристрою, так і загальної інформаційної системи, що організується на її основі [11, 12]. Це включає у себе високі показники адаптивності, пов'язані з програмованістю повного набору логічних функцій засобами FPGA [11], а також ефективні підходи по паралелізації процесів передачі і обробки потокових даних, щ оптимізує інформаційну систему за цільовими показниками швидкодії і навантаження на обчислювальний ресурс при роботі у режимі реального часу [12].

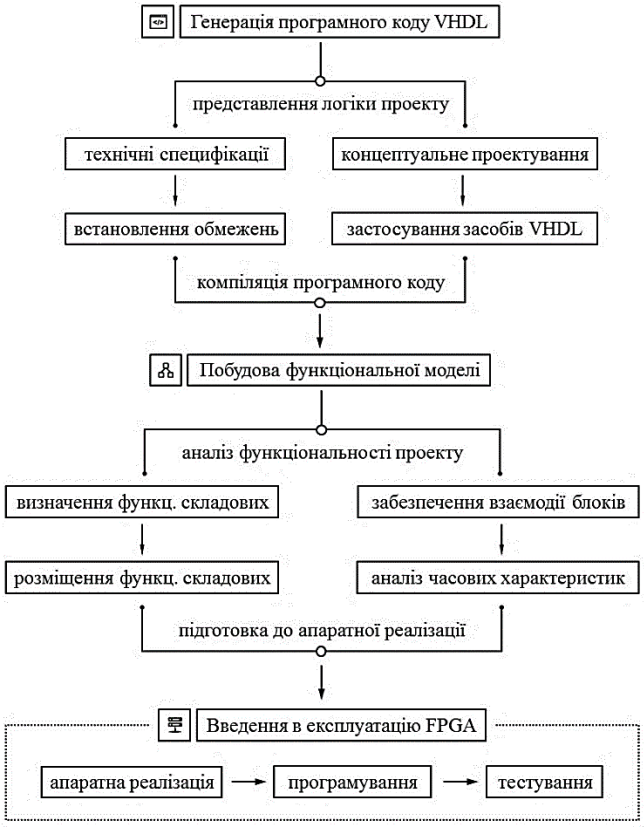

Рис. 1. Базова схема проектування FPGA

Як показав проведений аналіз, основні переваги PLD та, в особливості, FPGA, як окремого і найбільш функціонального класу PLD полягають у програмованості відповідних логічних пристроїв. Формалізація зазначеного процесу полягає у визначенні інструментарію мови опису апаратури (Hardware Description Language; HDL) та, зокрема, високошвидкісної мови опису апаратури, як то VHDL (Very-High-Speed Integrate Circuit Hardware Description Language), як спеціалізованої мови програмування дизайну FPGA, синтаксис якої буде розглянуто у даному дослідженні [13-15]. Впровадження інструментарію VHDL при проектуванні FPGA надає широкі можливості для проведення автоматичного аналізу та імітації робочих режимів експлуатації відповідно до експлуатаційних вимог. VHDL характеризується модульністю структури, що надає можливість реорганізації і спільного використання окремих блоків програмного коду, належний опис на основі VHDL забезпечує паралельне виконання набору завдань, а математичні моделі, що формуються засобами VHDL реалізують ефективне прогнозування результатів роботи у відповідності до заданих сценаріїв. Також важливою перевагою VHDL є стандартизація

програмного коду, що забезпечує можливість його портування без внесення змін та сумісність з інструментарієм, що використовується при організації широкого набору інформаційних систем.

Зазначені переваги формування PLD на основі наборів програмованих користувачем вентильних матриць з застосуванням інструментарію мови опису апаратури вказують на високу актуальність задачі побудови комплексної методики оптимізації програмного коду VHDL при проектуванні FPGA.

Аналіз останніх досліджень і публікацій. Найбільш актуальним підходом при проектуванні архітектури FPGA є формування комплексної методики, що базується на визначенні технічних специфікацій і функціональних можливостей відповідного логічного пристрою [14, 15]. При цьому комплексна методика проектування FPGA включає у себе виконання наступних етапів (рис. 1):

- представлення логіки проекту побудови FPGA на рівні генерації програмного коду VHDL;
- функціональне моделювання FPGA з визначенням продуктивності виконання операцій для різних режимів роботи;
- побудова FPGA на апаратному рівні, програмування логічного пристрою та тестування.

Таким чином, перехід від високорівневого до низькорівневого представлення при проектуванні логічних блоків FPGA, що проводиться на етапі компіляції, дозволяє визначити кількість логічних елементів, а завдяки функціональному моделюванню оцінюється архітектура схеми та формуються методичні рекомендації. На етапі функціонального моделювання оцінюється ефективність алгоритмів обробки та захисту потокових даних [3, 4, 15], визначаються підходи по паралелізації процесів, що виконуються логічними блоками FPGA [5, 6, 15], формуються протоколи спільного використання та інтеграції логічних блоків [7-10, 15]. При цьому невирішеною частиною загального дослідження є задача побудови комплексної методики оптимізації архітектури FPGA, що надає можливість сформувати набір підходів по проектуванню схеми з застосуванням засобів VHDL.

Формулювання цілей статті. У відповідності до проведеного аналізу метою дослідження є визначення підходів по оптимізації програмного коду VHDL при проектуванні FPGA та формування на основі зазначених підходів цілісної методології проектування стійкої і масштабованої інформаційної системи.

Виклад основного матеріалу. У якості початкового прикладу оптимізації архітектури FPGA через співставлення високорівневого та низькорівневого представлення проекту пропонується розглянути реалізацію на основі програмованої інтегральної логічної схеми завдання розрахунку суми двох з величин з набору у залежності від контрольного сигналу. Формалізуємо завдання наступним чином: на основі набору поточних даних $\{x_1; x_2; x_3\}$, що поступають на вхід логічного пристрою у відповідності до контрольного двійкового значення x необхідно розрахувати вихідне значення, як $y = x_1 + x_2$ при $x = 1$ або $y = x_2 + x_3$ при $x = 0$. Засобами VHDL відповідна задача вирішується через створення проекту «P1» з чотирма вхідними значеннями «X_IN: in std_logic», «X1_IN: in std_logic», «X2_IN: in std_logic» і «X3_IN: in std_logic» та одним вихідним значення «Y_OUT: out std_logic», а також застосування умовного оператора «if-then-else» у тілі проекту. Слід зазначити, що з метою спрощення програмного коду умовний оператор співставляє двійкову величину X_IN лише зі значенням «1», альтернативне значення вважається за «0» і для нього використовується команда «else».

На рис. 2 наведено співставлення лістингу відповідного програмного коду з функціональною моделлю, що генерується в результаті його компіляції. Аналіз функціональної схеми надає можливість відзначити надмірне генерування логічних елементів (один мультиплексор і два суматори), що не відповідає простоті як поставленої задачі, так і генерованого програмного коду. Процедура оптимізації при цьому має базуватись на модифікації високорівневого представлення FPGA.

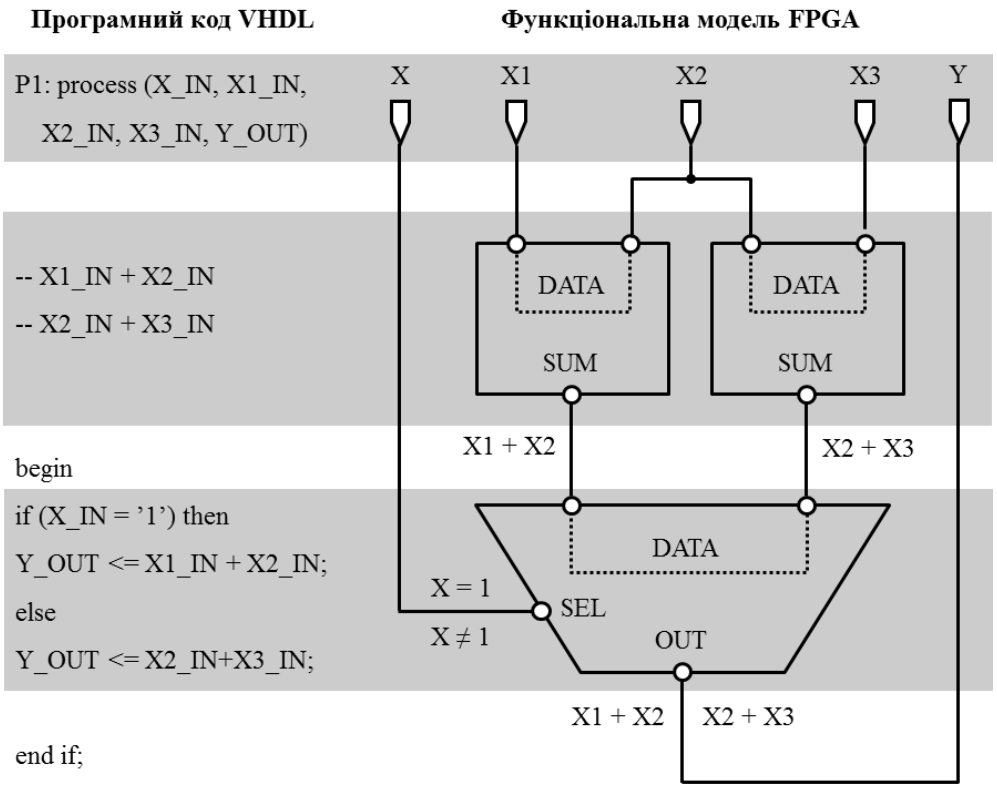

Рис. 2. Базова функціональна модель FPGA з суматороми і мультиплексором

Розглянемо модифікований проект «P1M», у рамках якого умовний оператор в залежності від контрольного значення X_IN обирає величину X1_IN або X3_IN, яка вже надалі поза тіла умовного оператору додається до величини X2_IN. Очевидно, що результат вирішення поставленого завдання при виконанні програмного коду «PM» і «P1M» не відрізняється, але компіляція проекту «P1M» надає можливість отримати функціональну схему, яка містить лише один суматор (рис. 3), що дозволяє скоротити кількість логічних елементів у півтори рази.

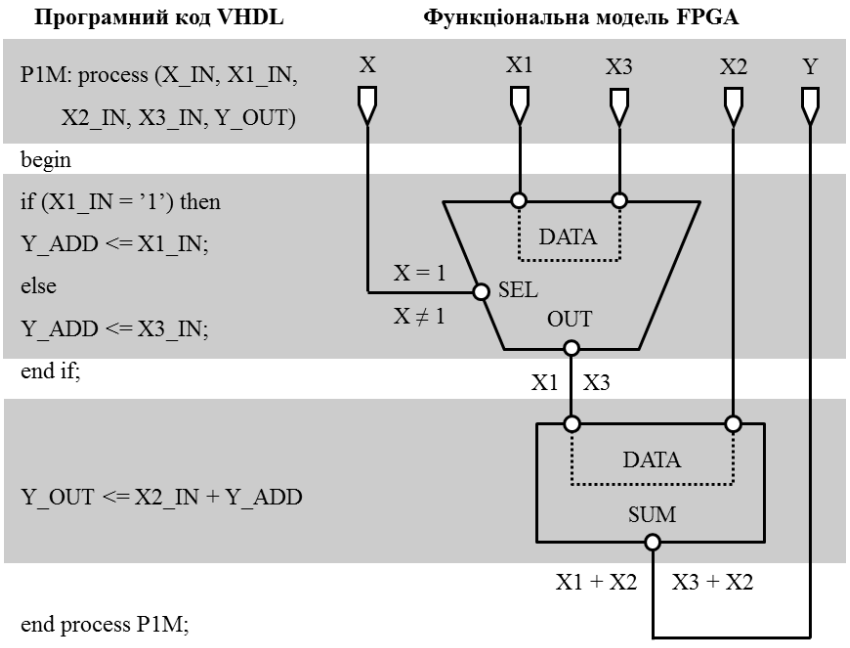

Рис. 3. Оптимізована функціональна модель FPGA з суматором і мультиплексором

Для складних архітектур FPGA відповідний підхід надає можливість суттєво зменшити загальну площу логічного пристрою, енергоспоживання та ризик виникнення помилок при його експлуатації.

Іншим прикладом, що пропонується розглянути у дослідженні, є оптимізація арифметико-логічного пристрою (Arithmetic Logic Unit; ALU) через паралелізацію процесів на рівні модифікації арифметичних операцій у програмному коді. Розглянемо задачу побудови функціонального блоку FPGA, що виконує задачу розрахунку вихідної величини на основі набору вхідних величин $\{x_1; x_2; x_3; x_4\}$ як $y = x_1 + x_2 - x_3 - x_4$. Задачу можна розглядати як тривіальну, що засобами VHDL вирішується через створення проекту «P2» з чотирма вхідними значеннями «X1_IN: in std_logic», «X2_IN: in std_logic», «X3_IN: in std_logic» і «X4_IN: in std_logic» та одним вихідним значення «Y_OUT: out std_logic», а також застосуванням блоків суматорів і субтракторів, причому виконання арифметичних операцій описується як «Y_OUT <= X1_IN + X2_IN – X3_IN – X4_IN».

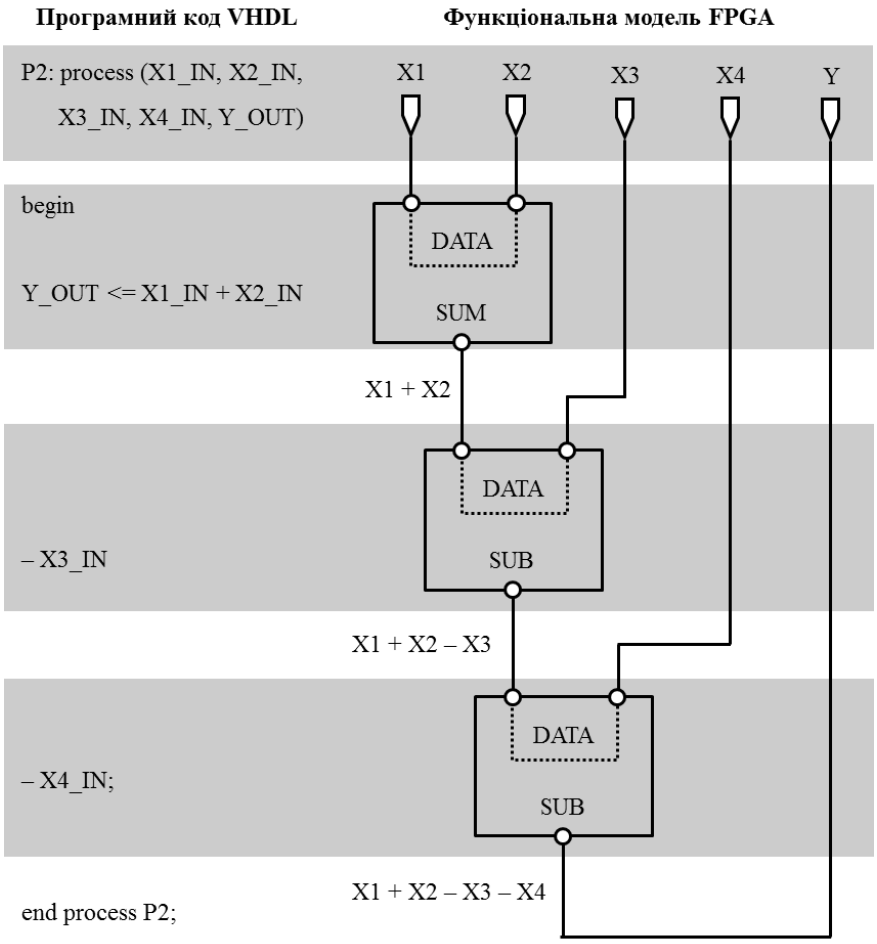

Рис. 4. Базова функціональна модель блоку ALU

Тим не менш, компіляція програмного коду надає можливість сформувати функціональну модель (рис. 4), яка свідчить про те, що відповідна арифметична операція виконується у три етапи, у той час як паралелізація надала б можливість суттєво збільшити швидкодію виконання операцій блоком ALU. Засобами VHDL можна спростити вираз як «Y_OUT <= X1_IN + X2_IN – (X3_IN + X4_IN)». Компіляція програмного коду відповідного проекту «P2M» вказує на реорганізацію функціональної схеми таким чином, що два з трьох процесів виконуються паралельно, що збільшує швидкодію пристрою у півтори рази.

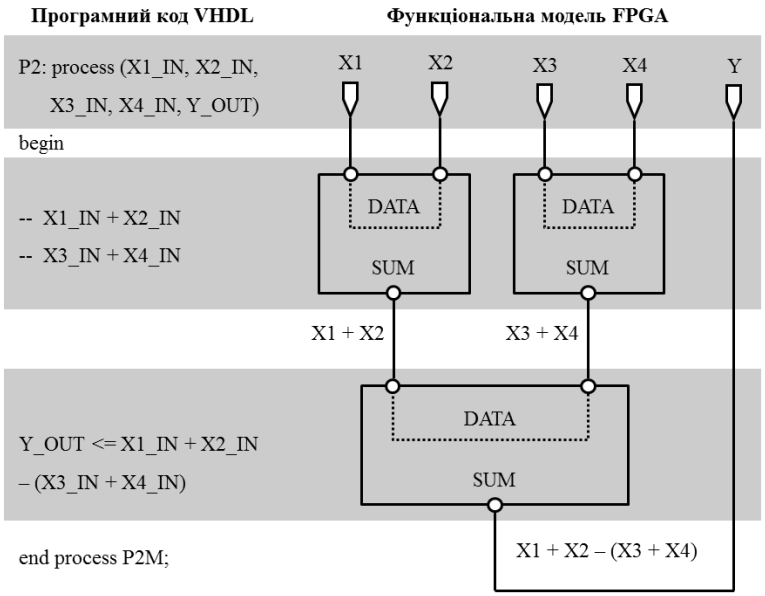

Рис. 5. Модифікована функціональна модель блоку ALU

Отже, можна зазначити, що співвіднесення низькорівневого і високорівневого представлення при проектуванні архітектури FPGA надає можливість суттєво збільшити продуктивність роботи логічного пристрою за цільовими показниками представленими на кількісному рівні, як то загальна площа схеми, кількість логічних елементів, швидкодія виконання операцій, а також ризик виникнення помилок. Застосування комплексної методики оптимізації програмного коду VHDL при проектуванні FPGA можливе через введення додаткових змінних та констант з метою спрощення операцій і виразів, виключення надмірного генерування логіки через зменшення об'єму програмного коду у тілі циклу та видалення блоків коду в умовних конструкціях, що не використовується, а також структурування блоків програмного коду.

Висновки. Проведений аналізу надав можливість визначити набір підходів по оптимізації програмного коду VHDL при проектуванні FPGA та формування на основі зазначених підходів цілісної методології проектування стійкої і масштабованої інформаційної системи. У дослідженні розглянуто ряд підходів по спрощенню архітектури програмованої логічної інтегральної схеми та збільшенню швидкодії обробки потокових даних та виконання операцій зазначених у специфікаціях FPGA.

Список використаних джерел

1. Alsabbagh, W., & Langendörfer, P. (2023). Security of programmable logic controllers and related systems: Today and Tomorrow. *IEEE Open Journal of the Industrial Electronics Society, 4*, 659–693. https://doi.org/10.1109/ojies.2023.3335976.
2. Chizek, M. (2019). Programmable logic device (PLD) safety design approach. *Journal of System Safety, 55*(1), 32–41. https://doi.org/10.56094/jss.v55i1.54.
3. Zhou, Y. (2016). Security analysis of a mutual-authentication cryptographic protocol based on Strand Space Model theory. *Revista Tecnica De La Facultad De Ingenieria Universidad Del Zulia*. https://doi.org/10.21311/001.39.3.8.
4. Almeida, J. B., Barbosa, M., Barthe, G., & Dupressoir, F. (2016). Verifiable side-channel security of cryptographic implementations: Constant-time Mee-CBC. *Fast Software Encryption,* 163–184. https://doi.org/10.1007/978-3-662-52993-5_9.
5. Timchenko, L. I., Petrovski, M. S., Kokriatskaia, N. I., & Denysova, A. E. (2014). Laser beam images classification methods with the use of parallel-hierarchical network running on PLD. *2014 X International Symposium on Telecommunications (BIHTEL).* https://doi.org/10.1109/bihtel.2014.6987644.
6. Kelly, M. S., Mayes, K., & Walker, J. F. (2017). Characterising a CPU fault attack model via run-time data analysis. *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST).* https://doi.org/10.1109/hst.2017.7951802.
7. Babu, H. Md. (2022). Generic Complex Programmable Logic Device Board. *VLSI Circuits and Embedded Systems*, 299–310. https://doi.org/10.1201/9781003269182-24.
8. Kalkhof, T., & Koch, A. (2022). Direct device-to-device physical page migrations in Multi-FPGA shared virtual memory systems. *2022 32nd International Conference on Field-Programmable Logic and Applications (FPL).* https://doi.org/10.1109/fpl57034.2022.00043.

9. Communication Networks, S. and. (2023). Retracted: Hierarchical network security measurement and optimal proactive defense in cloud computing environments. *Security and Communication Networks, 2023*, 1–1. https://doi.org/10.1155/2023/9808414.
10. Li, R., Decocq, B., Fang, Y., Zeng, Z., & Barros, A. (2022). A petri net-based model to study the impact of traffic changes on 5G network resilience. *Book of Extended Abstracts for the 32nd European Safety and Reliability Conference.* https://doi.org/10.3850/978-981-18-5183-4_s28-02-493.
11. Sunkavilli, S., & Yu, Q. (2022). Security threats and countermeasure deployment using partial reconfiguration in FPGA CAD Tools. *2022 IEEE International Symposium on Hardware Oriented Security and Trust (HOST).* https://doi.org/10.1109/host54066.2022.9839731.
12. Kumar Saha, S., & Bobda, C. (2020). FPGA accelerated embedded system security through hardware isolation. 2020 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). https://doi.org/10.1109/asianhost51057.2020.9358258.
13. Sunkavilli, S., Zhang, Z., & Yu, Q. (2023). FPGA security: Security threats from untrusted FPGA CAD toolchain. *Frontiers of Quality Electronic Design (QED)*, 551–573. https://doi.org/10.1007/978-3-031-16344-9_14.
14. Ekonde Sone, M. (2020). A new cross-layer FPGA-based security scheme for wireless networks. *Computer and Network Security*. https://doi.org/10.5772/intechopen.82390.
15. Taraate, V. (2017). Erratum to: Pld Based Design with VHDL. *PLD Based Design with VHDL*. https://doi.org/10.1007/978-981-10-3296-7_12

References

1. Alsabbagh, W., & Langendörfer, P. (2023). Security of programmable logic controllers and related systems: Today and Tomorrow. *IEEE Open Journal of the Industrial Electronics Society*, *4*, 659–693. https://doi.org/10.1109/ojies.2023.3335976.
2. Chizek, M. (2019). Programmable logic device (PLD) safety design approach. *Journal of System Safety*, *55*(1), 32–41. https://doi.org/10.56094/jss.v55i1.54.
3. Zhou, Y. (2016). Security analysis of a mutual-authentication cryptographic protocol based on Strand Space Model theory. *Revista Tecnica De La Facultad De Ingenieria Universidad Del Zulia*. https://doi.org/10.21311/001.39.3.8.
4. Almeida, J. B., Barbosa, M., Barthe, G., & Dupressoir, F. (2016). Verifiable side-channel security of cryptographic implementations: Constant-time Mee-CBC. *Fast Software Encryption*, 163–184. https://doi.org/10.1007/978-3-662-52993-5_9.
5. Timchenko, L. I., Petrovski, M. S., Kokriatskaia, N. I., & Denysova, A. E. (2014). Laser beam images classification methods with the use of parallel-hierarchical network running on PLD. *2014 X International Symposium on Telecommunications (BIHTEL).* https://doi.org/10.1109/bihtel.2014.6987644.
6. Kelly, M. S., Mayes, K., & Walker, J. F. (2017). Characterising a CPU fault attack model via run-time data analysis. *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST).* https://doi.org/10.1109/hst.2017.7951802.
7. Babu, H. Md. (2022). Generic Complex Programmable Logic Device Board. *VLSI Circuits and Embedded Systems*, 299–310. https://doi.org/10.1201/9781003269182-24.
8. Kalkhof, T., & Koch, A. (2022). Direct device-to-device physical page migrations in Multi-FPGA shared virtual memory systems. *2022 32nd International Conference on Field-Programmable Logic and Applications (FPL).* https://doi.org/10.1109/fpl57034.2022.00043.
9. Communication Networks, S. and. (2023). Retracted: Hierarchical network security measurement and optimal proactive defense in cloud computing environments. *Security and Communication Networks, 2023*, 1–1. https://doi.org/10.1155/2023/9808414.
10. Li, R., Decocq, B., Fang, Y., Zeng, Z., & Barros, A. (2022). A petri net-based model to study the impact of traffic changes on 5G network resilience. *Book of Extended Abstracts for the 32nd European Safety and Reliability Conference.* https://doi.org/10.3850/978-981-18-5183-4_s28-02-493.
11. Sunkavilli, S., & Yu, Q. (2022). Security threats and countermeasure deployment using partial reconfiguration in FPGA CAD Tools. *2022 IEEE International Symposium on Hardware Oriented Security and Trust (HOST).* https://doi.org/10.1109/host54066.2022.9839731.
12. Kumar Saha, S., & Bobda, C. (2020). FPGA accelerated embedded system security through hardware isolation. 2020 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). https://doi.org/10.1109/asianhost51057.2020.9358258.
13. Sunkavilli, S., Zhang, Z., & Yu, Q. (2023). FPGA security: Security threats from untrusted FPGA CAD toolchain. *Frontiers of Quality Electronic Design (QED)*, 551–573. https://doi.org/10.1007/978-3-031-16344-9_14.
14. Ekonde Sone, M. (2020). A new cross-layer FPGA-based security scheme for wireless networks. *Computer and Network Security*. https://doi.org/10.5772/intechopen.82390.
15. Taraate, V. (2017). Erratum to: Pld Based Design with VHDL. *PLD Based Design with VHDL*. https://doi.org/10.1007/978-981-10-3296-7_12