УДК 004.42:510.6

Р. В. Кучанський[1]
Р. С. Белзецький[1]

СИСТЕМА ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ТА ПЕРЕТВОРЕННЯ ЛОГІЧНИХ СТРУКТУР МЕТОДАМИ СИМВОЛЬНОГО МОДЕЛЮВАННЯ

[1]Вінницький національний технічний університет

Анотація

У роботі розроблено та досліджено програму для автоматичного аналізу та спрощення логічних функцій. Дана програма дозволяє виконувати складні розрахунки за допомогою бібліотеки SymPy, забезпечуючи високу точність перетворення логічних виразів. Особливу увагу приділено відображенню результатів у вигляді часових діаграм та графічних формул, що відповідають вимогам ДСТУ. Цей інструмент значно полегшує проектування цифрових пристроїв та допомагає автоматично готувати технічні звіти.

Ключові слова: інтелектуальні системи, логічні функції, мінімізація, часові діаграми, аналітична оптимізація.

Abstract

In this paper develops and investigates a program for automatic analysis and simplification of logical functions. It allows performing complex calculations using the SymPy library, ensuring high accuracy of logical expression conversion. Particular attention is paid to displaying results in the form of time diagrams and graphical formulas that meet the requirements of DSTU. This tool greatly facilitates the design of digital devices and helps to automatically prepare technical reports.

Keywords: intelligent systems, logic functions, minimization, timing diagrams, analytical optimization.

Вступ

У сучасній цифровій схемотехніці проектування логічних пристроїв вимагає швидкого та безпомилкового аналізу булевих функцій. Складність сучасних систем призводить до того, що ручне спрощення логічних виразів, особливо при роботі в універсальних логічних базисах (І-НІ, АБО-НІ) [1], стає трудомістким процесом із високим ризиком виникнення помилок. Актуальною проблемою є автоматизація цих розрахунків з дотриманням вимог до оформлення технічної документації та практичних робіт.

Метою роботи є розробка програми для мінімізації та графічного моделювання логічних функцій згідно з методичними вимогами до практичної роботи. Програма призначена для перетворення структурних формул у мінімізований вигляд, а також для візуалізації часових діаграм.

Результати дослідження

Послідовність дій у розробленій програмі базується на алгоритмі, що задає чіткий порядок виконання операцій: від введення даних a_1, a_2, a_3, a_4 та їх перевірки до проведення розрахунків і побудови графічних діаграм. Цей алгоритм виступає основою для побудови архітектури програми, регламентуючи взаємодію між обчислювальними модулями та блоками візуалізації, щоб автоматизувати роботу, забезпечити ефективну автоматизацію мінімізації булевих виразів, побудови діаграм та уникнути помилок, які часто виникають при ручному спрощенні формул. Додатково система самостійно проводить перевірку коректності кожного етапу, що гарантує достовірність отриманих даних незалежно від обсягу вхідної інформації. Це дозволяє користувачеві одразу отримати готове значення та продивитися усі задіяні для цього закони спрощення. Блок схема алгоритму роботи розробленої програми представлена на рисунок 1.

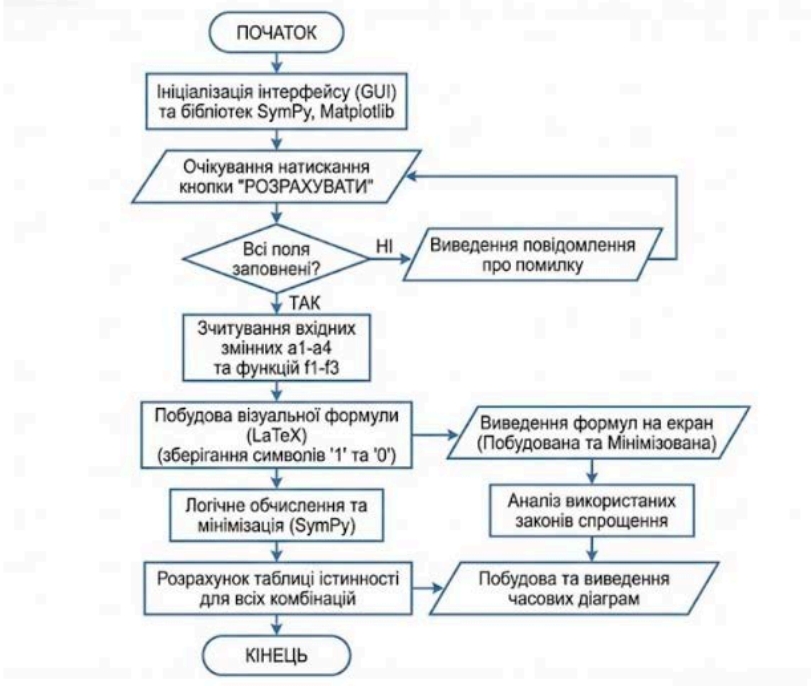

Рисунок 1 – Блок схема алгоритму роботи програми

На рисунку 2 зображена архітектура програми, яка визначає її загальну структуру. Вона написана за таким принципом, де центральний клас LogicLabClean координує взаємодію між компонентами LogicCore, LawAnalyzer та FormulaRenderer. Такий підхід дозволяє за допомогою бібліотеки SymPy [2] проводити точні обчислення та аналітичну мінімізацію виразів, у той час як модуль FormulaRenderer забезпечує рекурсивну конвертацію об’єктів у графічний формат із дотриманням вимог ДСТУ. Використання модулів GraphPlotter та LawAnalyzer у межах даної архітектури забезпечує високу ефективність автоматизації аналізу логічних структур, дозволяючи проводити комплексне моделювання сигналів та автоматично ідентифікувати використані закони алгебри логіки.

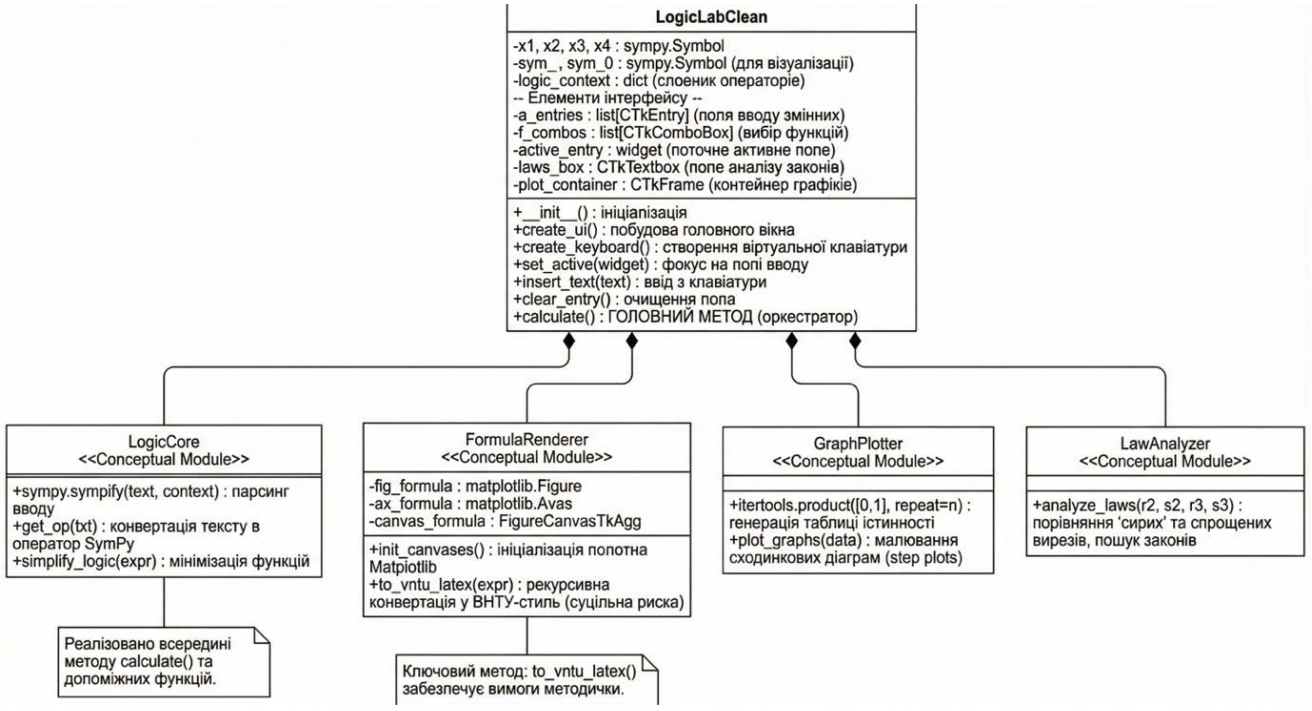

Рисунок 2 – UML-діаграма класів

На основі цих алгоритмів було створено програму на мові Python [3]. Вона забезпечує повний цикл проектування, що охоплює етапи від зчитування вхідних змінних та візуальної інтерпретації процесу підстановки констант до формування підсумкових мінімізованих виразів. Ключовою перевагою розробки є автоматична генерація графічних формул із суворим дотриманням стандартів ДСТУ, що дозволяє інтегрувати отримані результати безпосередньо у технічну документацію та навчальні звіти без необхідності додаткового графічного редагування.

Початкова форма розробленої програми представлена на рисунку 3.

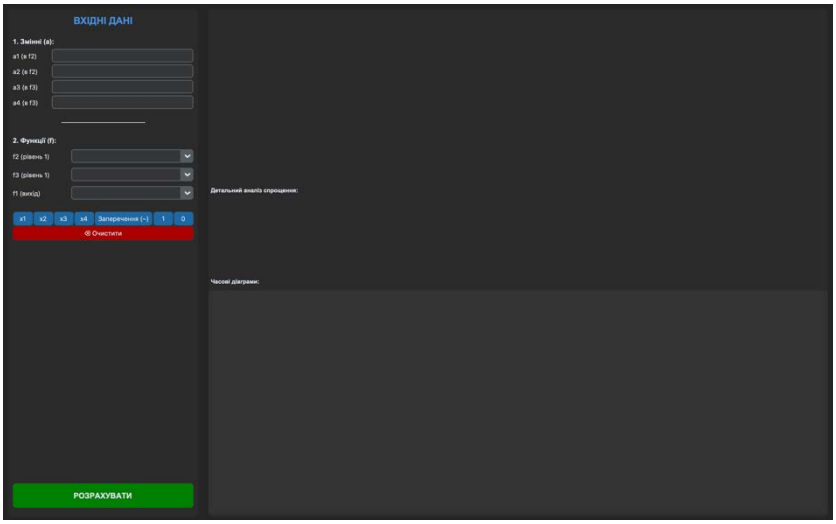

Рисунок 3 – Початкова форма програми

Окрім аналітичного спрощення, система здійснює розрахунок повних таблиць істинності та побудову ступінчастих часових діаграм для всіх можливих комбінацій сигналів, забезпечуючи наочну перевірку динаміки роботи логічного пристрою. Чітку послідовність виконання цих операцій та загальну логіку функціонування системи відображено в базовому алгоритмі (рисунок 4).

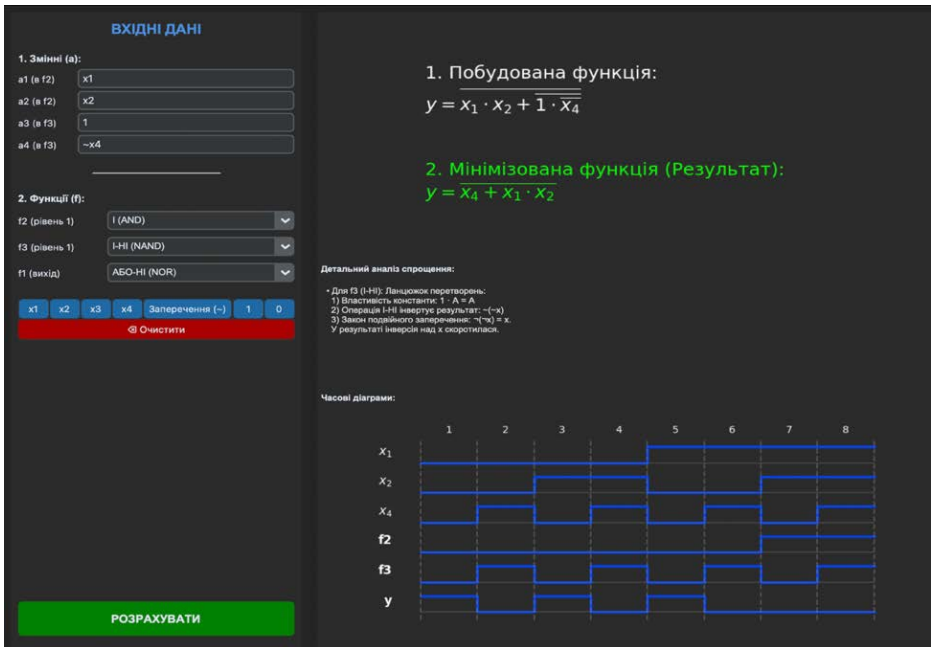

Рисунок 4 – Результат мінімізації функції

Висновки

У результаті виконання роботи було розроблено програму для мінімізації логічних функцій та проектування логічних схем. Вона автоматично обробляє складні формули та спрощує їх до мінімального вигляду, що дозволяє уникнути типових помилок для ручних розрахунків. Головна перевага програми полягає в тому, що вона подає результати у зрозумілій та професійній формі: генерує готові для звіту математичні формули з правильним оформленням згідно зі стандартами ДСТУ та будує наочні часові діаграми для перевірки сигналів. Завдяки цій програмі процес проектування цифрових пристроїв стає швидким, точним і зручним.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Rosen K. H. Discrete Mathematics and Its Applications. 8th ed. New York : McGraw-Hill Education, 2019. 1120 p. URL: https://www.youseficlass.ir/wp-content/uploads/2023/07/Discrete-Mathematics-and-Its-Applications-8th-Edition.pdf (дата звернення: 2.03.2026).
2. Meurer A., Smith C.P., Paprocki M. et al. SymPy: symbolic computing in Python. *PeerJ Computer Science*. 2017. Vol. 3. e103. URL: https://peerj.com/articles/cs-103 (дата звернення: 2.03.2026).
3. Halvorsen H. P. Python for Science and Engineering. Halvorsen. blog. 2020. 305 p. URL:https://www.halvorsen.blog/documents/programming/python/resources/Python%20for%20Science%20and%20Engineering.pdf (дата звернення: 2.03.2026).

Кучанський Роман Володимирович – студент групи 1КН-24б**,** факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, E-mail: *foxr988@gmail.com;*

Белзецький Руслан Станіславович – канд. техн. наук, доцент кафедри комп'ютерних наук, Вінницький національний технічний університет, м. Вінниця, E-mail: *belzetskyi@vntu.edu.ua*.

Kuchansky Roman V. – student of group 1KN-24b, Faculty of Intellectual Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, E-mail: *foxr988@gmail.com;*

Belzetskyi Ruslan S. – Cand. Sc. (Eng.), Assistant Professor of the Chair of Integration Education with Production, Vinnytsia National Technical University, Vinnytsia, E-mail: *belzetskyi@vntu.edu.ua*.