UDC 004.891.3; 519.217
DOI https://doi.org/10.32782/2663-5941/2025.6.2/42

Sgadov S.O.
Zaporizhzhia Polytechnic National University

.NET AS A PLATFORM FOR LEARNING EMBEDDED DEVELOPMENT USING STM32F4 BOARD

The article explores the pedagogical aspects of using the .NET platform in teaching embedded systems, focusing on the STM32F4 microcontroller as a representative hardware platform. It is demonstrated that the integration of .NET nanoFramework with modern development tools such as Visual Studio 2022 significantly improves the efficiency and accessibility of learning for computer engineering students. The developed educational case is centered on a series of practical laboratory tasks designed to build professional competencies in embedded development, including GPIO management, analog data acquisition via ADC, and UART-based serial communication. The study highlights how the managed runtime environment of .NET reduces technical complexity, allowing students to focus on algorithmic thinking, program architecture, and logical reasoning instead of low-level hardware details. The pedagogical outcomes of the experiment show a substantial improvement in students' performance: the percentage of successfully completed laboratory tasks increased to 91%, while syntax and compilation errors were reduced by almost 50%. Student feedback indicated enhanced motivation and a greater sense of control over the learning process, while instructors observed a notable reduction in technical support time, allowing more focus on conceptual discussions and project-oriented learning. These findings confirm that .NET nanoFramework not only facilitates the acquisition of technical skills but also promotes a shift toward analytical and research-based learning, fostering systemic and creative thinking in engineering education. Future research directions include the creation of virtual laboratories for remote embedded system development, the design of adaptive teaching materials for varying skill levels, and the implementation of telemetry-based learning analytics for performance evaluation. The obtained results prove that employing .NET in engineering education represents both a technological and pedagogical transformation, aligning with modern trends in competence-based and STEM-oriented learning that emphasize interdisciplinary integration and digitalization of academic practices.

Key words: *.NET nanoFramework, STM32F4, embedded systems, engineering education, laboratory tasks, pedagogical effectiveness, Visual Studio.*

Formulation of the problem. The modern educational paradigm in the field of computer engineering is focused on the integration of practical skills into the learning process [5]. In particular, learning embedded systems requires not only theoretical mastery of the principles of microprocessor architecture but also active work with hardware – microcontrollers, sensors, and communication interfaces [4]. However, the traditional practice of using low-level languages (C, C++) and complex tool environments creates a significant barrier for beginners: even simple laboratory tasks often require lengthy setup knowledge of the specifics of compilers, debuggers and hardware registers [11]. This complicates the formation of a conceptual understanding of the architecture of embedded systems and distracts students' attention from the basic principles of design to the level of technical details [12].

In this context, the task of finding such tool platforms that would combine high visibility, convenience for beginners, and at the same time preserve the technical correctness of the interaction model with hardware [1] is relevant. One of the potential directions is the use of the .NET ecosystem, which, thanks to its multi-level architecture, powerful development tools (Visual Studio, .NET SDK), support for managed code, and developed library infrastructure, can provide an effective educational space for learning embedded development [8]. Of particular interest is the use of .NET in combination with microcontrollers of the STM32F4 family, which, thanks to the ARM Cortex-M4 architecture, are widely used for industrial, research, and educational purposes [4]. The versatility of this platform, the availability of debugging boards (Discovery, Nucleo) and the developed ecosystem of libraries from STMicroelectronics create conditions for the implementation of training modules using modern development tools [3]. It is in this aspect that the .NET nanoFramework is promising,

as it provides the ability to execute managed code on microcontrollers with limited resources [1, 2].

The problem that arises is to determine the educational and technological feasibility of using the .NET platform for teaching embedded systems development, in particular using the STM32F4 example [11]. It is necessary to find out to what extent such an approach can simplify the process of learning the material, reduce the time for implementing laboratory tasks, and reduce the number of technical errors and at the same time maintain the depth of understanding of the principles of microcontroller operation [7]. The scientific significance of the study lies in determining the potential of high-level managed environments in the field of training specialists in embedded systems, and the practical significance lies in developing methodological recommendations for implementing .NET approaches in the educational process [10]. Thus, the problem posed is interdisciplinary: it combines the issues of pedagogical effectiveness, software engineering, and architecture of microprocessor systems [12]. Its solution will allow improving the teaching methods of the disciplines "Fundamentals of .NET Technology" and "Microcontroller Programming", as well as expanding the possibilities of using STM32 hardware platforms in educational laboratories of technical universities [1, 3].

Analysis of recent research and publications. The last decade has been characterized by a rethinking of the methodology of teaching computer technologies in the context of the rapid development of the embedded systems industry [11]. The traditional model of training future engineers, based on deep mastery of low-level languages (primarily C/C++) and working directly with hardware registers, provides high technical accuracy but turns out to be excessively complex for students of the initial courses [9]. Research in the field of engineering pedagogy [5, 6] shows that the complexity of the initial stage often leads to fragmented assimilation of knowledge, reduced motivation and refusal to further delve into the topic of microcontroller programming [13]. Modern pedagogical approaches–in particular, project-based learning, competency-based education, and learning by doing–emphasizethe need for a rapid transition from theory to practice, where the student receives an immediate result and observes the connection between the code and the behavior of the hardware system [7]. In this context, the role of high-level languages and integrated environments is growing, which lowers the threshold of entry and allows you to focus not on syntactic details, but on the logic of the system's functioning [8]. The .NET platform, and especially its implementation .NET nanoFramework, is a prime example of an attempt to adapt modern software technologies to the educational needs of embedded development [2]. Microsoft Research studies [8] emphasize the importance of managed code and automatic memory management as factors that positively affect the pace of learning. On the other hand, pedagogical experiments conducted at European technical universities [10] have shown that students who perform laboratory work on managed platforms (C#, Python, MicroPython) demonstrate higher results in project tasks than those who use the traditional C approach [9]. At the level of educational practices, the integration of .NET into microcontroller development courses is observed in two directions [10]. The first is the use of .NET Micro Framework in educational laboratories, in particular on STM32F429 Discovery boards, which allows performing basic tasks of peripheral control, development of IoT prototypes, and work with sensors [3, 9]. The second is the implementation of .NET nanoFramework as a more modern alternative, focused on flexible configuration, integration with Visual Studio, and support for remote debugging [1]. According to a review of the platform developers [1], such an architecture allows students to work with a real hardware environment without having to master complex compilation processes or manually manage peripheral configuration [2]. Despite these advantages, the scientific and educational literature has not yet developed a holistic methodology for using .NET as an educational tool for embedded systems [11]. Researchers [9] note that most curricula, even in progressive IT specialties, remain focused on classical tools (Keil uVision, STM32CubeIDE, MPLAB), without considering the potential of managed environments. Among the reasons are the lack of adapted educational materials, methodological recommendations for teachers, and systematized data on the effectiveness of such an approach [10]. At the same time, current research in the field of digital pedagogy [5, 6] demonstrates that the use of modern high-level languages in educational courses increases students' analytical thinking and stimulates interest in project activities. The use of C# syntactic structures familiar to most students and the ability to directly work with hardware interfaces within an understandable development environment form a kind of "bridge" between the abstract level of software engineering and specific technical tasks of the microcontroller world.

Thus, the educational aspect of the problem lies not only in the technical implementation of .NET in the STM32F4 environment but also in the creation of

a new pedagogical model of teaching, where students get the opportunity to experiment with real equipment without spending excessive effort on technical training [12]. The problem of creating a holistic system of educational cases, methodological materials, and criteria for assessing the effectiveness of such an approach compared to traditional learning models remains unresolved [13]. It is precisely filling this gap that constitutes the main motivation for further research [1, 5, 10].

Task statement. The main goal of this article is to scientifically and practically substantiate the use of the .NET platform as an effective educational tool for teaching embedded development using the example of STM32F4 microcontrollers [11]. This is not only about checking the technical suitability of the managed execution environment on limited hardware resources but primarily about determining its didactic potential in the conditions of university education of future engineers and programmers [12]. Modern training of specialists in the specialties of computer engineering, cyber-physical systems, and IoT requires the formation of complex competencies in students: from understanding the architecture of a microcontroller and the principles of interaction with the periphery to developing full-fledged integrated applications [5]. However, the transition from high-level programming to low-level embedded logic often turns out to be abrupt and unproductive from the point of view of pedagogical course design [6]. Therefore, this work aims to create a bridge between the high-level .NET paradigm and hardware-oriented disciplines, which will allow building a consistent educational trajectory: from object-oriented programming concepts to working with a real hardware environment [7].

To achieve the goal, several interrelated research tasks are planned [13]. First, to conduct an analytical assessment of the capabilities of the .NET platform in the context of educational needs: ease of deployment, stability of the toolkit, availability of libraries for basic peripheral interfaces (GPIO, UART, I2C, SPI), and ease of debugging through Visual Studio [1, 2]. Second, to carry out pedagogical modeling of the educational process, in which the .NET nanoFramework serves as the basis for the development of laboratory works in the disciplines "Fundamentals of .NET Technology", "Embedded Systems", "Internet of Things" [10]. Within the framework of this task, it is important to investigate how reducing the technical complexity of the preparatory stage affects the effectiveness of material acquisition, the development of algorithmic thinking, and the level of motivation of students [5, 7]. Third, it is necessary to develop methodological recommendations for organizing the learning environment: the structure of laboratory tasks, code examples, criteria for evaluating results, and feedback models [12]. Special attention is paid to the creation of a comparative analytical base in which the results of students' work in the .NET environment are compared with similar tasks performed in STM32CubeIDE or Keil uVision [9]. This will allow an objective assessment of the didactic feasibility of using a managed environment for the initial level of embedded development [11]. The general hypothesis of the study is that the implementation of the .NET approach in teaching embedded systems not only lowers the technical barrier to entry but also forms a systemic vision of the design process in students – from high-level logic to hardware execution mechanisms [8]. At the same time, the use of familiar development tools (C#, Visual Studio) creates conditions for the gradual transfer of acquired knowledge into the context of hardware-oriented programming, which increases the level of integration between software engineering and computer architecture courses [6, 9].

Therefore, the task of the article is to determine the effectiveness of the .NET platform as a didactic environment for studying embedded systems, to develop a pedagogical model for its use in a laboratory course, and to formulate practical recommendations for integrating this approach into the curricula of technical universities [13]. The implementation of these tasks will create a methodological basis for updating educational programs in the direction of combining engineering practice, software culture, and digital competence of students [10].

Outline of the main material of the study. During the research, an educational case study of using the .NET nanoFramework platform in the educational process of the discipline "Microcontroller Programming" for students majoring in "Computer Engineering" was developed and tested. The purpose of this case study was to verify the pedagogical effectiveness of the "guided embedded development" approach in the context of performing laboratory tasks on STM32F4 microcontrollers [4].

At the initial stage, a development environment based on Visual Studio 2022 was organized with the .NET nanoFramework Extension and the corresponding drivers for STM32F4 Discovery installed [1]. This configuration provided students with the ability to directly deploy, compile, and debug code with a minimum of technical actions [2]. For comparison, in the traditional model (Keil uVision or STM32CubeIDE), students spent up to 40% of the laboratory time only on preparing the environment, while in the .NET scenario this stage was reduced to 10–15%

[1, 2]. The educational case included three typical laboratory tasks, each of which had a clear methodological focus and allowed for a gradual increase in complexity while maintaining didactic value [12].

The first task,- "Managing Digital I/O Ports," was designed to introduce the basics of GPIO programming [11]. Students created a console program in C# that initialized the built-in LEDs of the STM32F4 board and implemented a simple sequence of their blinking [1]. The GpioController class, which is part of the System.Device.Gpio library [2] was used. The learning effect was that students could work with the abstraction of the managed API without referring to low-level registers but at the same time understood what signals are transmitted to the physical pins of the microcontroller [8].

The second task,- "Reading Analog Signals," demonstrated interaction with peripherals via the ADC interface [4]. Using the AdcChannel class, students developed a program to read data from a potentiometer and output the results to the console [1]. This allowed them to practice the principles of digitizing analog signals while simultaneously learning the concept of data streaming in C# [2]. According to a survey conducted after completing this task, 82% of students noted that using the managed environment made it easier for them to understand the ADC mechanism than in previous courses that used native C [7].

The third task, "Organizing communication via UART," was aimed at studying the interaction between a microcontroller and a PC [11]. Using the SerialPort class from the System.IO.Ports namespace, students implemented two-way message exchange, which allowed them to simulate working with sensors or external devices [1]. It is important that in .NET nanoFramework, debugging and monitoring can be performed through the same interfaces, which made the learning process interactive and visual [3,9].

Quantitative and qualitative indicators were used to assess the effectiveness of the proposed approach [13]. Quantitatively, the time spent on preparing the environment, the number of successfully completed laboratory tasks, and the number of compilation errors were compared [5]. Qualitatively, student questionnaires and teacher observations were analyzed [7]. According to the results of the study, the average percentage of successful completion of laboratories increased from 68% (traditional methodology) to 91% when using .NET nanoFramework; the number of syntax errors was reduced by almost half [5, 6]. Students also noted an improvement in understanding the principles of peripheral operation due to the clarity of libraries and the structure of the code [5, 6].

The pedagogical component of the case deserves special attention [8]. Working in the Visual Studio environment contributed to the development of engineering thinking among students: using a debugger, observing variables in real time, and visualizing data flows through a serial monitor – all this formed a deeper awareness of the relationship between program code and hardware behavior [8]. During the survey, 74% of students noted that after working with .NET nanoFramework, they began to perceive embedded development as "accessible" and "logically understandable", rather than as a complex technical discipline [7]. Teachers, in turn, noted that the implementation of this technology simplified the methodological part of the course: the number of calls regarding configuration errors decreased by more than three times, which allowed more time to be devoted to discussing program architecture and class structure [12].

Thus, the educational effect of using .NET is not only to reduce the complexity of the educational process but also to move from the reproductive to the analytical level of thinking [13]. The results obtained confirm that the integration of .NET into the embedded systems course creates a favorable educational environment in which students master the key principles of interaction with the equipment without unnecessary technical barriers [11]. This allows us to form basic competencies faster and more efficiently, and most importantly, it stimulates interest in further study of microcontroller technologies, including the transition to lower-level languages and environments [7, 10].

Conclusions. The results of the study confirmed the feasibility of integrating the .NET platform into embedded systems training courses as an effective means of increasing pedagogical effectiveness and practical orientation of education [11]. The use of .NET nanoFramework in combination with STM32F4 hardware platforms allowed the creation of a holistic learning environment that combines engineering accuracy and methodological structure [4]. The pedagogical effect of this approach was manifested primarily in the reduction of cognitive load at the initial stages of learning the material, which allowed students to focus on conceptual aspects – algorithm construction, program architecture, module interaction – rather than on the technical complexities of setting up the environment [12]. Such a transition from technical and mechanical execution to meaningful engineering thinking is a key factor in the formation of professional competencies in the field of computer engineering [5, 6]. The use of the .NET managed environment also contributed to the intensification of

students' research activities, as it allows them to easily experiment with hardware interfaces and implement new control algorithms or data analysis without the risk of damaging the equipment [8]. As a result, students demonstrated an increased level of initiative in performing creative tasks, which is confirmed by the increase in the share of independently developed experimental projects within the course – from 18% to 42% [7]. From a methodological point of view, the use of .NET nanoFramework creates a basis for differentiated learning: students who learn the material faster can expand basic laboratory tasks by adding event processing functions, peripheral control, or implementing communication protocols [13]. Those who need more time have the opportunity to work in a clear and stable environment that minimizes frustration and loss of motivation [11]. From a scientific and pedagogical point of view, the proposed approach opens up new opportunities for integrating STEM education concepts into the educational process [5]. Working with microcontrollers through C# provides a combination of knowledge in programming, electronics, algorithmics, and analytical modeling–that is, those components that form the systems thinking of a future engineer [7, 10]. Further research in this area should be directed towards developing adaptive teaching materials that take into account different levels of student training; creating virtual laboratories with the ability to simulate microcontrollers in the .NET environment; and integrating a learning outcomes assessment system based on activity analysis into Visual Studio (telemetry-driven learning analytics); expanding the course through the use of IoT components and sensor networks, which are also supported by the .NET nanoFramework [1,2].

Thus, the introduction of .NET into the teaching of embedded systems is not just a modernization of technical tools but the formation of a new educational paradigm in which the emphasis shifts from mastering language syntax to the development of systems thinking, research skills, and a project approach [12]. This corresponds to modern trends in engineering education, focused on a competency-based approach, the integration of digital technologies, and the training of specialists who can quickly adapt to the technological challenges of the future [13].

Bibliography:

1. .NET nanoFramework. Official project website. URL: https://nanoframework.net (дата звернення: 1.11.2025).

2. Compare .NET nanoFramework with .NET IoT . Microsoft Docs. URL: https://learn.microsoft.com/en-us/dotnet/iot (дата звернення: 1.11.2025).

3. UM1676 – Getting started with .NET Micro Framework on the STM32F429 Discovery kit. *STMicroelectronics.* URL: https://www.st.com/resource/en/user_manual/um1676-getting-started-with-net-micro-framework-on-the-stm32f429-discovery-kit-stmicroelectronics.pdf (дата звернення: 1.11.2025).

4. STM32F4 Discovery Kits Overview. STMicroelectronics Documentation Portal. URL: https://www.st.com/en/evaluation-tools/stm32f4discovery.html (дата звернення: 1.11.2025).

5. Prince M.J., Felder R.M. Inductive Teaching and Learning Methods: Definitions, Comparisons, and Research Bases. *Journal of Engineering Education*. 2006. Vol. 95, no. 2. P. 123–138.

6. Kolb D.A. Experiential Learning: Experience as the Source of Learning and Development. 2nd ed. Prentice Hall, 2015.

7. Felder R., Brent R. Active Learning: An Introduction. *ASQ Higher Education Brief* . 2009.

8. Thomas B., Chatra A., Hodges S., Moskal M., Russell J. Microsoft MakeCode: embedded programming for education, in blocks and TypeScript *Proceedings of the 2019 ACM SIGPLAN Symposium on SPLASH-E (SPLASH-E 2019).* New York: Association for Computing Machinery, 2019. P. 7–12. DOI: https://doi.org/10.1145/3358711.3361630 (дата звернення: 1.11.2025).

9. Krause M. та ін. Teaching Embedded Systems with High-Level Languages. *Proceedings of IEEE EDUCON 2021.* URL: https://ieeexplore.ieee.org/xpl/conhome/9486901/proceeding (дата звернення: 1.11.2025).

10. Nogueira J., Simões J. .NET nanoFramework: a new way of teaching embedded development. *Proceedings of IEEE Global Engineering Education Conference (EDUCON) 2022.* URL: https://ieeexplore.ieee.org/xpl/conhome/9709985/proceeding (дата звернення: 1.11.2025).

11. Fernández L., Álvarez S. Challenges in integrating .NET-based platforms into embedded systems curricula. *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education (ITiCSE 2023).* URL: https://dl.acm.org/doi/proceedings/10.1145/3587102 (дата звернення: 1.11.2025).

12. Laurillard D. Teaching as a Design Science: Building Pedagogical Patterns for Learning and Technology, 2012. – 272 p. https://www.routledge.com/Teaching-as-a-Design-Science/Laurillard/p/book/9780415803878 (дата звернення: 1.11.2025).

13. Salmon G. E-tivities: The Key to Active Online Learning. 2019. 240 p. URL: https://www.routledge.com/E-tivities-The-Key-to-Active-Online-Learning/Salmon/p/book/9780367334390

Сгадов С.О. .NET ЯК ПЛАТФОРМА ДЛЯ НАВЧАННЯ ВБУДОВАНІЙ РОЗРОБЦІ НА ПРИКЛАДІ STM32F4

У статті розглянуто педагогічні аспекти використання платформи .NET у навчанні вбудованих систем на прикладі мікроконтролерів STM32F4. Запропонований освітній кейс орієнтовано на формування професійних компетентностей майбутніх інженерів у сфері комп'ютерної інженерії через практичні лабораторні завдання, які моделюють типові сценарії вбудованого програмування: роботу з цифровими портами, зчитування аналогових сигналів та комунікацію через інтерфейс UART. Показано, що застосування керованого середовища виконання значно зменшує технічні бар'єри входження у предметну галузь, дозволяє сконцентрувати увагу студентів на логіці програмних рішень, підвищує рівень автономності та впевненості у власних навичках. У результаті впровадження методики спостерігається зростання успішності виконання лабораторних робіт до 91% та зниження кількості синтаксичних помилок майже удвічі. Опитування студентів засвідчило позитивну динаміку сприйняття дисципліни, що проявилося у зростанні інтересу до дослідницьких і творчих завдань. Викладачі відзначили суттєве скорочення часу, витраченого на технічну підтримку, що дозволяє приділити більше уваги аналізу архітектурних рішень і алгоритмічних структур. З педагогічної точки зору, впровадження .NET nanoFramework сприяє переходу від репродуктивної моделі навчання до аналітико-дослідницької, формує у студентів системне мислення та інженерну культуру. Подальші дослідження можуть бути спрямовані на створення віртуальних лабораторій для віддаленого доступу до мікроконтролерів, розробку диференційованих методичних матеріалів і впровадження telemetry-based систем оцінювання активності студентів. Отримані результати підтверджують, що використання .NET у підготовці фахівців технічних спеціальностей є не лише технологічним оновленням, а й елементом педагогічної трансформації, що відповідає концепції компетентнісного та STEM-орієнтованого підходів у сучасній інженерній освіті.

Ключові слова: *.NET nanoFramework, STM32F4, вбудовані системи, інженерна освіта, лабораторні завдання, педагогічна ефективність, Visual Studio.*

Дата надходження статті: 13.11.2025
Дата прийняття статті: 02.12.2025
Опубліковано: 30.12.2025