DOI: 10.18137/RNU.V9187.24.02.P.132 УДК 0040054

А.П. Киреев, С.А. Шаров

СПОСОБ ПРОВЕРКИ КОРРЕКТНОСТИ ПЛАНИРОВЩИКА ЗАДАЧ ОПЕРАЦИОННОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ С ПОМОЩЬЮ СЕТЕЙ ПЕТРИ

Аннотация. В статье предложен способ проверки корректности функционирования планировщика задач операционной системы реального времени с применением модели на базе временной сети Петри. Проведен обзор различных политик планировщиков задач операционной системы реального времени. Приведен анализ применения моделей временных сетей Петри для проверки планирования задач операционной системы реального времени на ранней стадии процесса проектирования бортовой аппаратуры космического аппарата с целью определения требуемых вычислительных ресурсов при использовании различных аппаратных конфигураций. Целью исследования является разработка способа проверки корректности функционирования планировщика задач операционной системы реального времени при различных аппаратных конфигурациях вычислительных систем бортового оборудования. По результатам исследований установлено, что технические параметры вычислительной системы бортовой аппаратуры космических аппаратов влияют на стабильность функционирования планировщика операционной системы реального времени. Большое значение для малых космических аппаратов имеют технические ограничения вычислительных платформ, построенных на базе микроконтроллеров. Дальнейшее эволюционное развитие проектируемой системы возможно за счет расширения моделей для различных вычислительных систем, что обеспечит соответствие свойств планировщика задач операционной системы реального времени заданным аппаратным ограничениям.

Ключевые слова: операционная система реального времени, космический аппарат, программное обеспечение, встроенные системы, сети Петри, планировщик задач.

A.P. Kireev, S.A. Sharov

METHOD FOR CHECKING THE CORRECTNESS OF THE TASK SCHEDULER OF A REAL-TIME OPERATING SYSTEM USING PETRI NETS

Abstract. The authors of the method proposed checking the correct operation of the intelligent task scheduler for switching on a constant-time system using a model based on a temporary Petri net. A review of various policy schedulers for lighting problems in real-time systems is carried out. The article analyzes the use of temporary Petri net models for testing scheduling problems with system devices in real time at the early stage of designing on-board equipment of an external device in order to determine the required computing resources when using various hardware configurations. Based on the research results, it was found that the technical parameters of the computer system of the on-board equipment of the devices affect the stability of the constant-time system scheduler. The technical limitations of computing platforms built on the basis of microcontrollers are of great importance for small devices. Further evolutionary development of the designed system is possible by expanding the models of various computing systems, which will ensure compliance with the property of scheduler tasks that ensure time constancy given hardware limitations.

Keywords: real-time operating system, spacecraft, software, embedded network systems, Petri net, task scheduler.

Киреев Андрей Павлович
старший научный сотрудник, Военно-космическая академия имени А.Ф. Можайского, Санкт-Петербург. Сфера научных интересов: верификация программного обеспечения. Автор семи опубликованных научных работ. Author ID 796874, SPIN-код 4707-9500.
Электронный адрес: vka@mil.ru

Шаров Сергей Алексеевич
научный сотрудник, Военно-космическая академия имени А.Ф. Можайского, Санкт-Петербург. Сфера научных интересов: верификация программного обеспечения. Автор более 30 опубликованных научных работ. SPIN-код: 9260-4311, AuthorID: 1236090.
Электронный адрес: vka@mil.ru

Введение

Одной из основных систем управления современной бортовой аппаратурой (далее – БА) космических аппаратов (далее – КА) является система реального времени (далее – СРВ). В системах данного типа [1; 2] важна не только логическая корректность вычислений, но и временной интервал, в течение которого будет достигнут результат. Такая система гарантирует своевременную реакцию на внешние воздействия, что повышает устойчивость работы системы управления бортовой аппаратуры.

Операционные системы реального времени (далее – ОСРВ) позволяют решить ряд важных задач (например, задачи системы ориентирования или энергообеспечения КА) в заданные требованиями промежутки времени. При этом выполнение задачи в ограниченный промежуток времени, а также гарантированное время отклика задачи обеспечивают надежное функционирование КА. Решение поставленной цели может быть выполнено с помощью планировщика, представляющего собой программное обеспечение ОСРВ и предназначенного для управления задачами.

Применение планировщика задач позволяет системам реального времени достичь выполнения следующих качественных характеристик: справедливость, предсказуемость и баланс производительности. Для ранней проверки корректности алгоритмов планирования задач операционных систем реального времени, в зависимости от конфигурации целевой системы и временных ограничений необходимо применение современных научных методов выявления ошибок.

В данной статье представлено решение задачи проверки корректности планировщика задач операционной системы реального времени с применением временных сетей Петри – одной из формальных моделей, используемых при верификации проекта операционной системы реального времени бортовой аппаратуры малого космического аппарата.

Целью исследования является разработка способа проверки корректности функционирования планировщика задач операционной системы реального времени при различных аппаратных конфигурациях вычислительных систем бортового оборудования.

Моделирование приложения реального времени

В качестве формальной модели корректности работы планировщика задач операционной системы реального времени используется модель временной сети Петри. Исходными данными для планирования задач являются необходимые процессы, функционирующие в системе реального времени и их приоритеты [3].

Обычно работа алгоритма планировщика происходит на временном интервале (0, t), который в каждый момент времени определяет количество задач, находящихся в состоянии готовности. В случае наличия одной готовой задачи она исполняется согласно своему приоритету, либо завершается один из ее фрагментов после того, как была завершена более приоритетная задача [4; 5].

Планировать задачи в системе реального времени можно статическими и динамическими методами. В качестве политики планировщика со статическими приоритетами используется алгоритм RMS (Rate Monotonic Scheduling – планирование с монотонной скоростью). Статические приоритеты назначаются задачам в соответствии с продолжительностью цикла задания, поэтому наиболее короткая продолжительность цикла приводит к более высокому приоритету задачи.

В случае применения планировщиком алгоритма EDF (Earliest Deadline First – первоочередное планирование более раннего срока выполнения) используется динамический принцип назначения приоритетов задачам. Планировщик задач, построенный на данном алгоритме, считает, что чем ближе предельный срок выполнения задачи, тем ее приоритет выше.

При моделировании приложения реального времени представляются набором задач. Для планирования времени исполнения каждая задача обычно определяется следующими временными параметрами [6]:

C_i – время выполнения задачи в наихудшем случае (WCET – worst-case execution time);

P_i – период (продолжительность между двумя задачами);

D_i – критический срок завершения выполнения задачи;

U – коэффициент использования процессора.

Необходимое условие того, что система реального времени будет исполнимой, – коэффициент использования процессора должен быть меньше или равен 1 [6]:

• по алгоритму RMS

$$U = \sum_{i=1}^{n} \frac{C_i}{P_i} \tag{1}$$

• по алгоритму EDF

$$U = \sum_{i=1}^{n} \frac{C_i}{D_i} \tag{2}$$

В общем виде сеть Петри может быть представлена следующим выражением [7; 8]:

$$C = (P, T, I, O), \tag{3}$$

где $P = \{p_1, p_2, \ldots, p_n\}$ – конечное множество позиций; $T = \{t_1, t_2, \ldots, t_m\}$ – конечное множество переходов; $I : T \to P^{\infty}$ – входная функция (отображение из переходов во входные комплекты позиций); $O : T \to P^{\infty}$ – выходная функция (отображение из переходов в выходные комплекты позиций).

Маркировкой μ сети Петри называется функция, отображающая множество позиций P в множество неотрицательных целых чисел N [9; 10].

Активные переходы $t_j \in T$ в маркированной сети Петри с заданной маркировкой μ разрешены, если для всех $p_i \in P$ выполняется следующее условие:

$$\mu(p_i) \geq \#\left(p_i, I\left(t_j\right)\right), \tag{4}$$

где # – оператор кратности перехода; $p_i \in P$ – множество разрешённых позиций; $t_j \in T$ – множество активных переходов.

В классической модели сети Петри отсутствует понятие времени. В то же время в асинхронных системах события могут происходить в неопределенных временных интервалах, что затрудняет указание точного времени длительности таких событий или их начала и окончания. Для преодоления указанных ограничений используются расширения сети Петри – временные сети Петри с возможностью задания времени переходов и задержки маркеров в позициях сети.

В качестве модели каждой выполняемой задачи в ОСРВ можно использовать следующий кортеж переходов временной сети Петри:

$$Task_i = (c_i, r_i, s_i, e_i, d_i), \tag{5}$$

где c_i – периодически создает маркер нового экземпляра задачи; r_i – отмечает маркер задачи как подготовленный, то есть задача готова к исполнению и ожидает запуска; s_i – запускает задачу, меняет ее состояние с «ожидание» на «выполнение»; e_i – завершает выполнение задачи, освобождает общие ресурсы и отмечает ее состояние как «выполнена»; d_i – переход deadline_task происходит, когда задача все еще выполняется и создан ее новый экземпляр.

В качестве примера системы ограничений рассмотрим модель ОСРВ с двумя задачами – $Task_i$ и $Task_j$ – с более высоким приоритетом задачи $Task_i$. Тогда для проверки корректности планирования задач можно использовать следующую систему ограничений:

$$\begin{gathered}(c_i, r_i, s_i) \succ (c_j, r_j, s_j), \\ \forall \mu(p_i) \nexists \mu\left(p_{deadline_{task}}\right) \geq 1; \mu \in R(C, \mu),\end{gathered} \tag{6}$$

где $p_{deadline\_task}$ – позиция максимального срока выполнения задачи $Task_i$; $\succ$ – отношение приоритета выполняемых задач; R – множество достижимости сети Петри.

С применением математической постановки задачи разработан алгоритм проверки корректности функционирования планировщика задач операционной системы реального времени с применением временных сетей Петри.

Алгоритм проверки корректности функционирования планировщика задач ОСРВ

На основе вышеизложенного в статье представлен способ проверки корректности расписания задач планировщика операционной системы реального времени.

Планирование задач в ОСРВ производится с целью установления детерминированных гарантий как выполнения необходимых задач, так и требуемого времени их отклика. Алгоритм проверки корректности функционирования планировщика задач представлен на Рисунке 1 и предназначен для моделирования функционирования задач со статическими приоритетами в операционной системе реального времени.

При построении алгоритма планировщика задач ОСРВ используется метод RMS [4]. Статические приоритеты задач назначаются в соответствии с продолжительностью цикла задачи $Task_i$.

В исходных ограничениях моделируемых задач ОСРВ указывается их общее количество, временные ограничения, статические приоритеты. Далее, исходя из выбранной модели исполняемой задачи, формируется временная сеть Петри для заданной группы задач ОСРВ (в примере используются задачи с приоритетами 5 и 11, общее время выполнения каждой из задач составляет 2 мкс).

Для проверки корректности работы планировщика была выбрана программная среда моделирования сетей Петри TINA [11–13]. Данная среда предназначена для визуального моделирования сетей Петри (в том числе с временными ограничениями), динамического исполнения моделей сетей, проведения проверки заданных ограничений для выбранной сети Петри.

С учетом вышеизложенного разработана модель временной сети Петри для ОСРВ с заданным количеством задач и их временными характеристиками. Общий вид данной модели в программной среде представлен на Рисунке 2.

По алгоритму проверки корректности планировщика производится запуск на выполнение разработанной модели временной сети Петри. На Рисунке 2 отображен процесс функционирования системы с двумя задачами при установленных ограничениях. Программная среда выдает отчеты об ошибках в реальном масштабе времени. При необходимости производится настройка количества шагов выполнения спроектированной модели.

При этом для тестируемой модели сделано допущение выполнения задач на одном процессоре. По результатам исследования было установлено, что в случае увеличения общего времени выполнения задач до 4 мкс планировщик не сможет гарантировать их своевременного запуска.

В случае реализации ОСРВ для малых космических аппаратов дополнительно необходимо учитывать ограничения аппаратной платформы. Обычной компоновкой таких решений является реализация вычислительной системы, использующей ресурсы микропроцессорного ядра, интегрированного в один кристалл совместно с устройствами ввода/вывода.

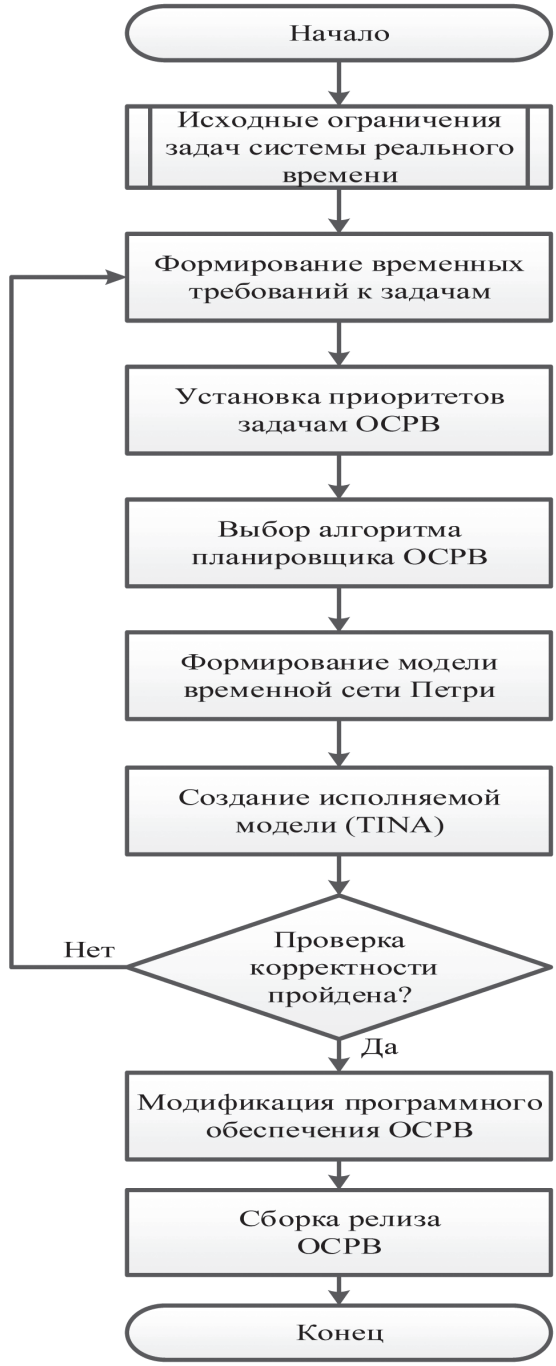

Рисунок 1. Алгоритм проверки корректности функционирования планировщика задач ОСРВ с применением сетей Петри
Источник: рисунки выполнены авторами.

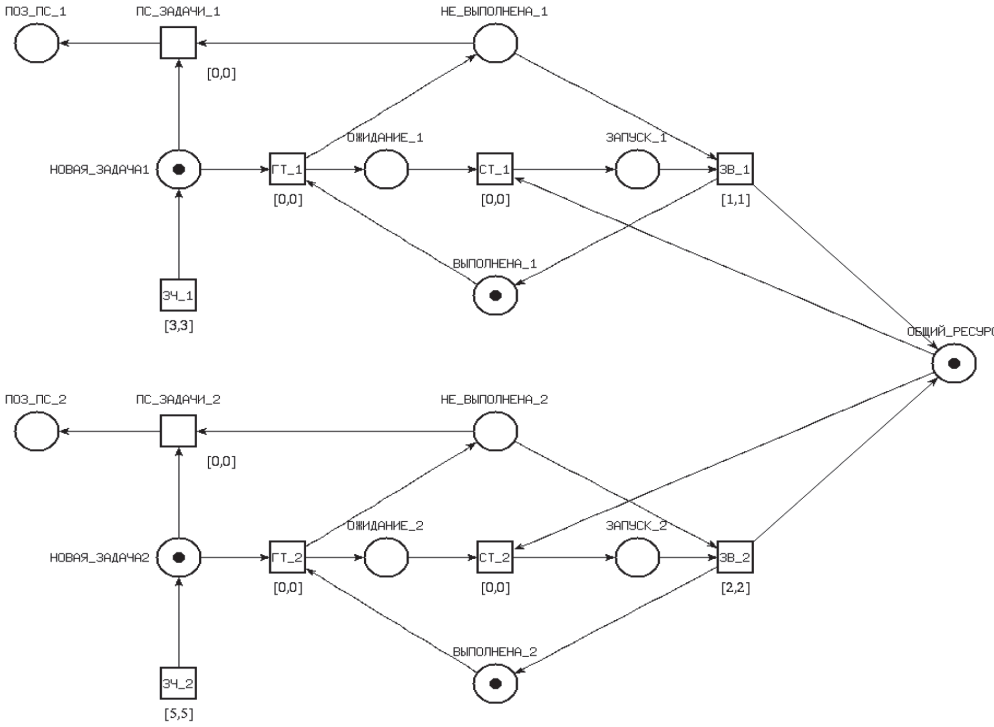

Рисунок 2. Модель планировщика ОСРВ с двумя задачами

Главной особенностью ОСРВ для встроенных решений является компактность исполнения при реализации быстрой реакции на внешние события и развитых средств межпроцессного взаимодействия.

В Таблице представлены технические параметры ОСРВ для микроконтроллерных платформ [14; 15].

Таблица

Технические параметры ОСРВ для различных платформ

Наименование и версия ОСРВ	Поддерживаемые аппаратные платформы	Макс. кол-во задач	Кол-во приори-тетов	Размер памяти, байт		Реализация алгоритмов планировщика
				Задача	Планировщик	
Zephyr OS 3.5.0	ARM, Intel x86, ARC,SPARC, RISC-V, MIPS, Xtensa	3-20	32	256-2048	512-2048	RMS,EDF, Round-Robin, Time-Slice
FreeRTOS 10.0.1	ARM, Intel x86, PIC, AVR		7	128-3072		RMS,EDF, Round-Robin
CoOS 1.1.6	ARM, Intel x86	5	64	68	168-1142	RMS,EDF, Round-Robin

Источник: [14; 15].

Указанные в Таблице аппаратные ограничения должны дополнительно учитываться при реализации бортовых вычислительных малых космических аппаратов.

Из анализа полученных результатов следует, что в случае увеличения времени выполнения задачи до 4 мкс невозможно составить корректное расписание планировщика вычислительной системы бортовой аппаратуры, что потенциально может привести к срыву выполнения задачи управления аппаратом (задачи управления ориентацией космического аппарата критично важны для его надежного функционирования). Отдельно необходимо отметить аппаратные ограничения микроконтроллерных платформ, наиболее часто используемых для построения вычислительных систем малых космических аппаратов.

Оценки планирования потоков исполнения могут основываться на временных измерениях кода прототипа или данных для аналогичного проекта (например, системы с одинаковыми реализациями ОСРВ и аппаратными ограничениями). Проводя анализ на ранней стадии процесса разработки с использованием модели временной сети Петри, можно получить качественные прогнозы функционирования планировщика задач.

Заключение

Таким образом, для проведения проверки корректности планирования задач ОСРВ, используемых при функционировании бортовой аппаратуры КА, необходимо применение разработанного алгоритма, позволяющего проводить верификацию планировщика задач ОСРВ БА КА с использованием моделей временных сетей Петри.

Применение научно-методического аппарата временных сетей Петри позволяет проводить верификацию различных задач ОСРВ, выполняемых в БА КА.

В данной статье рассмотрен способ проверки корректности функционирования планировщика задач операционной системы реального времени с применением сетей Петри. Представленный способ основан на применении моделей временных сетей Петри.

Применение средств проверки корректности функционирования планировщика задач ОСРВ на основе временных сетей Петри позволит повысить качество программного обеспечения, используемого при создании БА КА.

Результаты проведенных исследований могут быть использованы при создании новых образцов космической техники и разработки процедур верификации проектов интегрированной бортовой аппаратуры КА.

Литература

1. *Kopetz H., Steiner W.* Real-Time Systems. Design Principles for Distributed Embedded Applications. 3rd edition. Springer Cham, 2022. 406 p. DOI: https://doi.org/10.1007/978-3-031-11992-7

2. *Lee E.A., Seshia S.A.* Introduction to Embedded Systems – A Cyber-Physical Systems Approach. 2nd edition. MIT Press, 2017. 564 p. ISBN 978-0-262-53381-2.

3. *Diaz M.* (Ed.) Petri Nets Fundamental Models, Verification and Applications. ISTE Ltd and John Wiley & Sons, Inc., 2009. 581 p. ISBN 978-1-84821-079-0.

4. *Mohammadi A., Akl S.G.* Scheduling Algorithms for Real-Time Systems. Technical Report No. 2005-499. 49 p. URL: https://research.cs.queensu.ca/home/akl/techreports/scheduling.pdf (дата обращения: 01.15.2024).

5. *Третьяков А.В.* Автоматизация построения расписаний для периодических систем реального времени // Труды Института системного программирования РАН. 2012. Т. 22. С. 375–400. EDN RBTNKT. URL: https://ispranproceedings.elpub.ru/jour/article/view/1018 (дата обращения: 01.15.2024).

6. *Fotsing C., Singhoff F., Plantec A., Gaudel V., Rubini S., Li S, Tran H.N., Lemarchand L., Dissaux P., Legrand J.* Cheddar Architecture Description Language. Lab-STICC/UMR CNRS 6285, 2021. URL: https://www.academia.edu/73068943/Cheddar_Architecture_Description_Language (дата обращения: 01.15.2024).
7. *Питерсон Дж.* Теория сетей Петри и моделирование сетей / Пер. с англ. М.В. Горбатовой и др. М. : Мир, 1984. 264 с.
8. *Berard B., Cassez F., Haddad S., Lime D., Roux O.* Comparison of the Expressiveness of Timed Automata and Time Petri Nets // Pettersson P., Yi W. (Eds) Formal Modeling and Analysis of Timed Systems. FORMATS 2005. Series: Lecture Notes in Computer Science. Vol. 3829. Springer, Berlin, Heidelberg, 2005. Pp. 211–225. DOI: https://doi.org/10.1007/11603009_17
9. *Izmaylov A.A., Dworzanski L.W.* Automated Analysis of DP-systems Using Timed-Arc Petri Nets via TAPAAL Tool // Труды Института системного программирования РАН. 2020. Т. 32. Вып. 6. С. 155–166. EDN KORBVA. DOI: https://doi.org/10.15514/ISPRAS-2020-32(6)-12
10. *Berthomieu B., Le Botlan D., Dal Zilio S.* Counting Petri net markings from reductions equations // International Journal on Software Tools for Technology Transfer. 2020. No. 22. Pp. 163–181. DOI: https://doi.org/10.1007/s10009-019-00519-1
11. University of Hamburg, Germany. Complete Overview of Petri Nets Tools Database. URL: https://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/db.html (дата обращения: 15.01.2024).
12. TINA – Time petri Net Analyzer // LAAS CNRS. URL: http://projects.laas.fr/tina/ (дата обращения: 15.01.2024).
13. *Varga D., Simonac S.* A Study on Petri Net Supporting Tools for System Modeling and Analysis // Journal of Information and Organizational Sciences. 2023. Vol. 47. No. 1. DOI: https://doi.org/10.31341/jios.47.1.12
14. *Курниц А.А.* Free RTOS – операционная система для микроконтроллеров // Компоненты и технологии. 2012. № 4 (129). С. 135–144. EDN OXHGIB.
15. Supported boards. Project Documentation // Zephyr. URL: https://docs.zephyrproject.org/latest/boards/index.html (дата обращения: 28.02.2024).

References

1. Kopetz H., Steiner W. (2022) *Real-Time Systems. Design Principles for Distributed Embedded Applications*. 3rd edition. Springer Cham, 2022. 406 p. DOI: https://doi.org/10.1007/978-3-031-11992-7
2. Lee E.A., Seshia S.A. (2017) *Introduction to Embedded Systems – A Cyber-Physical Systems Approach.* 2nd edition. MIT Press, 2017. 564 p. ISBN 978-0-262-53381-2.
3. Diaz M. (Ed.) *Petri Nets Fundamental Models, Verification and Applications*. ISTE Ltd and John Wiley & Sons, Inc., 2009. 581 p. ISBN 978-1-84821-079-0.
4. Mohammadi A., Akl S.G. (2005) *Scheduling Algorithms for Real-Time Systems*. Technical Report No. 2005-499. 49 p. URL: https://research.cs.queensu.ca/home/akl/techreports/scheduling.pdf (accessed 01.15.2024).
5. Tretyakov A.V. (2012) Automation of scheduling for periodic real-time systems. *Proceedings of the Institute of System Programming of the Russian Academy of Sciences (Proceedings of ISP RAS)*. Vol. 22. Pp. 375–400. URL: https://ispranproceedings.elpub.ru/jour/article/view/1018 (accessed 01.15.2024). (In Russian).
6. Fotsing C., Singhoff F., Plantec A., Gaudel V., Rubini S., Li S, Tran H.N., Lemarchand L., Dissaux P., Legrand J. (2021) *Cheddar Architecture Description Language*. Lab-STICC/UMR CNRS 6285, 2021.

URL: https://www.academia.edu/73068943/Cheddar_Architecture_Description_Language (accessed 01.15.2024).
7. Peterson J.L. (1981) *Petri Net Theory and the Modeling of Systems*. Prentice Hall, Englewood Cliffs. (Russian edition: transl. by M.V. Gorbatova et al., Moscow : Mir Publ., 1984. 264 p.).
8. Berard B., Cassez F., Haddad S., Lime D., Roux O. (2005) Comparison of the Expressiveness of Timed Automata and Time Petri Nets. In: Pettersson P., Yi W. (Eds) *Formal Modeling and Analysis of Timed Systems*. FORMATS 2005. Series: Lecture Notes in Computer Science. Vol. 3829. Springer, Berlin, Heidelberg. Pp. 211–225. DOI: https://doi.org/10.1007/11603009_17
9. Izmaylov A.A., Dworzanski L.W. (2020) Automated Analysis of DP-systems Using Timed-Arc Petri Nets via TAPAAL Tool. *Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS)*. Vol. 32. No. 6. Pp. 155–166. DOI: https://doi.org/10.15514/ISPRAS-2020-32(6)-12
10. Berthomieu B., Le Botlan D., Dal Zilio S. (2020) Counting Petri net markings from reductions equations. *International Journal on Software Tools for Technology Transfer.* No. 22. Pp. 163–181. DOI: https://doi.org/10.1007/s10009-019-00519-1
11. University of Hamburg, Germany. *Complete Overview of Petri Nets Tools Database*. URL: https://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/db.html (accessed 01.15.2024).
12. TINA – Time petri Net Analyzer. *LAAS CNRS.* URL: http://projects.laas.fr/tina/ (accessed 01.15.2024).
13. Varga D., Simonac S. (2023) A Study on Petri Net Supporting Tools for System Modeling and Analysis. *Journal of Information and Organizational Sciences*. Vol. 47. No. 1. DOI: https://doi.org/10.31341/jios.47.1.12
14. Kurnits A.A. (2012) FreeRTOS – operating system for microcontrollers. *Components and technologies.* No. 4 (129). Pp. 135–144. (In Russian).
15. Supported boards. Project Documentation. *Zephyr*. URL: https://docs.zephyrproject.org/latest/boards/index.html (accessed 28.02.2024).