https://doi.org/10.31891/2219-9365-2025-83-57
УДК 004.056:004.852:004.75

КОЗЕЛЬСЬКИЙ Олександр
Хмельницький національний університет
https://orcid.org/0009-0002-7157-6499
e-mail: oleksandr.kozelskiy@khmnu.edu.ua
САВЕНКО Олег
Хмельницький національний університет
https://orcid.org/0000-0002-4104-745X
e-mail: savenko_oleg_st@ukr.net

ПРОАКТИВНИЙ МЕХАНІЗМ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ В ОС РЕАЛЬНОГО ЧАСУ З ВИКОРИСТАННЯМ ГІБРИДНОГО СТОРОЖОВОГО ТАЙМЕРА

У роботі подано метод превентивного перезапуску компонентів ОС реального часу, що поєднує спрощену марковську модель із гібридним подвійним сторожовим таймером. Підхід підвищує кіберстійкість і безперервність роботи за внутрішніх відмов та зовнішніх впливів (DoS, ін'єкції збоїв, логічні бомби). Використання малорозмірних марковських ланцюгів дає змогу швидко оцінювати ризик наближення до критичного стану без значних обчислювальних витрат, що забезпечує своєчасну локалізацію проблеми й вибірковий перезапуск завдань, драйверів або модулів.

Поєднання прогнозного та апаратного рівнів формує багаторівневий захист: прогнозний сторожовий таймер прогнозує збої, а апаратний гарантує відновлення у разі глибокого блокування чи атаки, зменшуючи частоту глобальних рестартів і час простою у ресурсно обмежених середовищах. Метод не потребує складних ML-алгоритмів чи громіздких формальних моделей, спираючись на мінімалістичні стохастичні схеми, адаптовні до різних платформ і загроз. Експерименти показали понад утричі менший час відновлення, зниження залежності від повного перезапуску та посилення інформаційної безпеки, що забезпечує стабільність і захищеність вбудованих та кіберфізичних систем для промислової автоматизації, робототехніки, автомобільної й медичної техніки.

Ключові слова: операційна система, RTOS, марковський ланцюг, превентивний перезапуск, відмовостійкість, кіберфізичні системи, виявлення аномалій, кіберстійкість

KOZELSKYI Oleksand, SAVENKO Oleg
Khmelnytskyi National University

PROACTIVE INFORMATION SECURITY MECHANISM IN REAL-TIME OPERATING SYSTEMS USING A HYBRID WATCHDOG

The paper introduces an advanced method for the preventive restart of real-time operating system components that integrates a simplified Markov reliability model with a hybrid dual watchdog timer architecture. This approach enhances cyber resilience, fault tolerance, and service continuity in the presence of both internal malfunctions and deliberate external influences such as denial-of-service attacks, logic bombs, and fault injection techniques. The use of compact Markov chains makes it possible to rapidly evaluate the probability of approaching a critical degradation state without imposing a substantial computational burden on the system. As a result, developers can ensure timely localization of faulty processes and carry out selective restarts of individual tasks, hardware drivers, or kernel modules instead of triggering global reboots.

A key contribution of the method is the integration of predictive and hardware-based protection layers into a unified multilayer mechanism. The predictive watchdog timer, operating at the stochastic modeling level, forecasts potential service disruptions based on transitional probabilities and observed runtime behavior. In parallel, the hardware watchdog component serves as a failsafe instrument, guaranteeing system recovery in situations of deep software blocking, escalated attack scenarios, or inconsistent module states. This dual structure reduces the frequency of full system restarts, limits downtime, and extends the operational life of critical components in resource-constrained environments.

Unlike approaches that rely on complex machine learning algorithms or burdensome formal verification techniques, the proposed solution is based on minimalist stochastic schemes that can be easily adapted to different platforms, architectures, and threat models. Experimental evaluation confirmed a more than threefold reduction in recovery time, a substantial decrease in reliance on global reinitialization procedures, and a noticeable improvement in information security indicators. The method thus contributes to increased reliability, uninterrupted control, and secure runtime performance in embedded and cyber-physical systems used in industrial automation, autonomous robotics, automotive electronics, and medical technologies, where failure tolerance and continued operation are paramount.

Keywords: operating system, RTOS, Markov chain, preventive restart, fault tolerance, cyber-physical systems, anomaly detection, cyber resilience

Стаття надійшла до редакції / Received 11.07.2025
Прийнята до друку / Accepted 14.08.2025

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ

У сучасних операційних системах реального часу зростання кількості завдань і ступеня інтеграції компонентів ускладнює забезпечення безперервності функціонування та дотримання часових обмежень [1]. Це особливо критично для кіберфізичних систем, де навіть короткочасна затримка може спричинити аварію

чи створити «вікно вразливості» для зловмисника [34]. Одним із базових механізмів підтримання надійності залишається сторожовий таймер, що виконує перезапуск системи після збою [2]. Проте такий реактивний підхід не враховує навмисні впливи, наприклад ін'єкції збоїв чи логічні атаки.

Для підвищення кіберстійкості запропоновано гібридний метод, який поєднує апаратний і прогнозний сторожовий таймер у рамках проактивної моделі контролю [15]. Програмний рівень реалізує оцінку стану системи за допомогою низькорозмірних марковських ланцюгів, що дозволяє завчасно виявляти ризик збою або аномальну активність і локально перезапускати окремий модуль без повного рестарту. Апаратний watchdog виконує функції остаточного відновлення у разі глибокого зависання або успішної атаки [12,13].

Запропонований підхід усуває недоліки громіздких формальних і машинно-навчальних методів, зберігаючи малу ресурсоємність і швидкодію [3]. Він дає змогу мінімізувати простої, зменшити частоту повних перезавантажень і підвищити кіберстійкість у ресурсно обмежених середовищах. Така схема створює основу для розвитку відмовостійких і захищених архітектур у вбудованих і реального часу системах нового покоління.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

В останні роки спостерігається активний розвиток методів прогнозування відмов у системах реального часу та вбудованих пристроях. Основну увагу приділено застосуванню машинного навчання й аналізу даних із сенсорів для підвищення надійності та ефективності [7]. Комплексні дослідження показують доцільність інтеграції модельно-орієнтованих і дані-орієнтованих підходів [11]. Такі системи здійснюють моніторинг температури, навантаження, часу відгуку та інших параметрів, виявляючи закономірності, що передують відмові. Нейронні мережі успішно прогнозують нелінійні залежності й попереджають перегрів або перевантаження обладнання. Реактивні стратегії, як-от апаратні перезавантаження, та консервативні політики зниження продуктивності вже не задовольняють сучасні вимоги. Відповідно, все більше уваги приділяється легковаговим засобам аномалійного моніторингу для обмежених CPS/IoT-платформ і методам виявлення ботнет-аномалій у SDN-IoT середовищах[16,19]. Окремо виділяють онлайновий моніторинг часу відгуку як основу для предиктивного обслуговування у багатоядерних RTOS [9]

Прогнозування відмов є особливо важливим для багатозадачних RTOS, де типові проблеми включають зациклення процесів, взаємоблокування (deadlock) чи пропуск дедлайнів [5]. Поширеними є й зависання апаратних компонентів або порушення синхронізації між задачами. Для підвищення кіберстійкості використовуються мережеві рішення: аналіз DNS-трафіку для раннього виявлення ботнет-активності, DNS-anti-evasion техніки та самоадаптивні системи локалізації атак у корпоративних мережах. Сучасні огляди ботнет-DDoS демонструють еволюцію векторів атак і зростання потреби у розподіленому моніторингу [14].

Фіксація факту збою вже недостатня — система має передбачати його наближення [4]. Модельно-орієнтовані підходи базуються на аналізі аналітичних моделей компонентів і резидуалів [6], тоді як дані-орієнтовані — на вивченні шаблонів телеметрії для виявлення ознак деградації без фізичної моделі. Методи, засновані на даних, демонструють високу ефективність у складних системах, однак вимагають наявності якісних навчальних вибірок [16]. У сфері кібербезпеки спостерігається зміщення акценту в бік автономних систем виявлення вторгнень (IDS) із розподіленим прийняттям рішень, а також підходів машинного навчання, оптимізованих для роботи на периферії мережі [17].

На практиці найкращі результати дає комбінування аналітичних і машинно-навчальних методів [10]. Така інтеграція підвищує точність прогнозування та сприяє формуванню єдиної основи для самодіагностики, прогнозування відмов і протидії цілеспрямованим кібератакам.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Запропонований підхід поєднує класичний апаратний сторожовий таймер, що виконує роль жорсткого рівня захисту, із програмним прогнозним шаром, який на основі спрощеної марковської моделі оцінює ризик збою ще до його фактичного настання та ініціює локальне відновлення (soft reset) проблемних компонентів. Якщо ж програмне скидання не дає результату або відмова зачіпає ядро системи, другий рівень — апаратний watchdog — здійснює повний перезапуск. Така дворівнева схема дозволяє скоротити кількість глобальних reset і зменшити час простою, що має критичне значення для кіберфізичних систем (CPS) із жорсткими часовими вимогами та обмеженими ресурсами.

Метод передбачає дискретну (крокову) оцінку стану системи в моменти часу $t = k \cdot \Delta t \ = \ k$, де Δt – період виконання завдання прогнозного сторожового таймера у ОСРЧ. На кожному кроці система визначається в одному з кінцевого набору станів (наприклад, {N,W,C,F} – "Normal", "Warning", "Critical", "Fail"), і переходи між цими станами описуються ланцюгом Маркова малої розмірності.

Нехай $S = \{N, W, C, F\}$ – множина можливих станів. У момент часу $t^k = k\Delta t$ система (або завдання прогнозного сторожового таймера, яке відображає її здоров'я) опиняється в одному зі станів $x(k) \in S$. Для спрощення припустимо, що N (Normal) — система в нормальному діапазоні (черги не переповнені, сигнали

про активність усіх завдань приходять вчасно), W (Warning) — виявлено погіршення показників (підвищене завантаження, повільні сигнали про активність), але ще не критично, C (Critical) — ризик швидкого збою високий (суттєві затримки, часті пропуски дедлайнів, завдання припиняє надсилати сигнали про активність), F (Fail) — фактичне зависання / збій (у моделі – це поглинаючий стан або індикатор, що відбувся глобальний reset).

За потреби модель може містити додаткові стани, як-от “Recovery” чи “Standby”, проте чотирьох–п’яти станів зазвичай достатньо для практичних сценаріїв.

Вважаємо, що $x(k+1)$ залежить лише від $x(k)$ і не залежить від станів у попередні моменти (властивість Маркова). Можна записати:

$$P = \begin{pmatrix} P_{NN} & P_{NW} & P_{NC} & P_{NF} \\ P_{WN} & P_{WW} & P_{WC} & P_{WF} \\ P_{CN} & P_{CW} & P_{CC} & P_{CF} \\ P_{FN} & P_{FW} & P_{FC} & P_{FF} \end{pmatrix}, \sum_{j\in S} P_{ij} = 1\,, \forall i \in S, \quad (1)$$

Якщо F (Fail) розглядається як поглинаючий стан, виконується $P_{FF} = 1$ і $P_{F_j} = 0$ для $j \neq F$. У такому разі потрапляння в F інтерпретує безповоротне зависання. Якщо ж у фізичній системі після F(Fail) виконується апаратний reset, то логічно вважати, що стан системи повертається до N (Normal) — але вже після перезапуску.

Для оцінки елементів p_{ij} застосуємо простий метод підрахунку частот (frequency counting) Припустимо, що на k-му такті систему віднесено до стану $i = x(k)$, а на $x(k+1)$ - му такті — до стану $j = x(k+1)$. Тоді лічильник збільшуємо на 1:

$$n_{(i\to j)} \coloneqq n_{(i\to j)} + 1 \quad (2)$$

Додатково відслідковуємо $n_i = \sum_{m\in S} n_{(i\to m)}$., тобто скільки разів система перебувала у стані . Таким чином, маємо:

$$p_{ij} \approx \frac{n_{(i\to j)\cdot}}{\sum_{m\in S} n_{(i\to m)}} = \frac{n_{(i\to j)\cdot}}{n_i} \quad (3)$$

Це відношення дає апроксимацію ймовірності переходу $i \to j$. Це дає змогу оновлювати P, без великого додаткового навантаження на процесор. Якщо система тривалий час працює, можна застосовувати скінчене ковзне вікно (sliding window) [8] або експоненційне згладжування (щоб не накопичувати стару статистику). Наприклад, при ковзному вікні розміром L (наприклад, 50–100 кроків) зберігаємо лише останні L спостережень переходів.

На кожному інтервалі часу Δt завдання прогнозного таймера в операційній системі реального часу здійснює збір поточних показників роботи системи, таких як заповненість черг, затримки виконання та кількість пропущених дедлайнів, після чого відносить систему до одного з можливих станів. У нормальному режимі черги не перевищують 50% заповнення, а всі критичні задачі своєчасно надсилають сигнали активності. Попереджувальний стан характеризується помірним або періодичним перевищенням навантаження в межах 50–80% чи короткочасними затримками сигналів від окремих задач. Критичний стан настає, коли черги перевищують 80% або фіксуються систематичні пропуски дедлайнів важливими задачами. У разі відсутності сигналів активності від однієї чи кількох критичних задач протягом кількох циклів поспіль або появи фатальної помилки система переходить у стан відмови. Межі між станами визначаються емпірично або на основі специфіки конкретного застосунку. Ймовірності переходів між станами у прогнозному таймері оновлюються динамічно та залежать від поточних змін у ключових системних метриках — заповненості черг, затримок активності, використання пам’яті та інших параметрів продуктивності [18]. Перевищення порогових значень цих показників зазвичай свідчить про наближення критичного стану і збільшує ймовірність переходу до нього:

$$p(W \to C) = f^1(Q, H, M) \quad (4)$$

Аналогічно, якщо затримка сигналів активності системи триває довше очікуваного часу, зростає ймовірність переходу у стан відмови:

$$p(C \to D) = f^2(H, T) \quad (5)$$

Якщо ж після Soft Reset система швидко стабілізується, збільшується, то:

$$p(C \to N) = f^3(R), \quad (6)$$

де Q – заповненість черги. H – затримка сигналів про активність. M — використання пам’яті. T — час очікування відповіді. R — кількість успішних програмних перезапусків, f^1, f^2, f^3 — емпіричні чи ідентифіковані на основі статистики функції, що перетворюють поточні системні метрики на ймовірність відповідного переходу, p(…) — ймовірність переходу прогнозного таймера із одного стану у другий.

Час до відмови системи розглядається як випадкова величина, що підпорядковується експоненційному закону розподілу. Така модель є типовою для операційних систем реального часу, оскільки відмови в них, як правило, виникають незалежно одна від одної та характеризуються сталою інтенсивністю. Експоненційний розподіл застосовується для визначення ймовірності безвідмовної роботи системи протягом

заданого проміжку часу, тобто для оцінки того, що відмова не настане раніше визначеного моменту. Його ключовою властивістю є відсутність пам'яті, тобто імовірність відмови в майбутньому не залежить від тривалості попереднього періоду безперебійної роботи. Інтенсивність відмов позначається параметром λ, який відображає середню частоту виникнення збоїв, а ймовірність збереження працездатності системи впродовж часу t визначається співвідношенням:

$$P(T > t) = e^{-\lambda t}, \tag{7}$$

де: T — час до відмови. λ — інтенсивність відмови. Ймовірність переходу у стан W або C визначається як:

$$p(N \to W) = 1 - P(T > t) \tag{8}$$

Якщо розраховане значення ймовірності перевищує певний поріг, сторожовий таймер переходить до проактивного реагування, викликаючи програмне скидання або готує апаратне скидання.

Опишемо прогнозування збою та двоступеневий захист:

Якщо відомо, що поточний стан $x(k) = i$, тоді: $P(x(k+1) = Fail \mid x(k) = i) = p_{iF}$. Якщо ця ймовірність p_{iF}перевищує заздалегідь встановлений поріг θ. Обиратимемо θ – порогове значення ймовірності збою. Коли $p_{iF} > 0$ (або ж $[P^2]_{iF} > 0$, типово $\theta \in [0.5; 0.8]$), то система вважається близькою до критичного стану, і прогнозний шар ухвалює рішення про превентивне програмне скидання відповідного компонента (наприклад, видалити проблемне завдання та створити його наново), не чекаючи фактичного зависання.

Якщо потрібно розглянути ймовірність потрапляння до F(Fail) за кілька кроків, використовуємо добуток матриці P^r:

$$P^r = \underbrace{\{P \times \mathrm{P} \times \cdots \times \mathrm{P}\}}_{r} \tag{9}$$

Тоді елемент $[P^r]_{iF}$ дає ймовірність опинитися у стані F через r кроків, якщо зараз $x(k) = i$. У практичних мікроконтролерних системах зазвичай вистачає 1–5 кроків прогнозу через обмежені ресурси і невеликі періоди Δt.

У пропонованому двоетапному механізмі якщо аналіз марковського ланцюга вказує на ймовірність ($\geq \theta$) швидкого переходу у F, програмний шар локально відновлює проблемну задачу (soft reset): $TaskDel(taskXHandle) \to ReInit() \to TCreate()$

Якщо локальний перезапуск не допомагає (стан C зберігається), програмний шар не викликає feed() апаратного сторожового таймера. За час T_{HW} (таймаут апаратного WDT) відбудеться повний reset системи. У марковській моделі це відповідає переходу $C \to F$, за яким фактично слідує перезавантаження мікроконтролера і повернення до початкового стану .

Таким чином, якщо відхилення не надто глибоке, soft reset дає змогу уникнути глобального перезапуску і зменшити час простою. Апаратний (HardwareWDT) зберігається як останній рубіж, якщо система вийшла з-під контролю програмного шару. Формально: якщо на кроці запущено процедуру soft reset, то на кроці $k+1$ штучно встановлюємо $x(k+1) = N$ (якщо перезапуск відбувся успішно). Задля точності вводимо стан "Recovery" (R) і модифікуєм матрицю переходів.

Оскільки ми маємо $\Delta t \ll T_{HW}$, тоді програмний шар має кілька спроб (наприклад, 10–20 циклів із кроком Δt) виконати програмне скидання, перш ніж спливе таймаут апартного сторожового таймера. Якщо завдання прогнозного сторожового таймера вважає, що стан знову став Normal (або Warning без прогресу до Critical), він викликає HW_WDT_Feed(), продовжуючи нормальну роботу. Якщо локальний збій не усувається, завдання прогнозного сторожового таймера свідомо не передає керуючий сигнал апаратному сторожовому таймеру, що призводить до його спрацювання та ініціює повний перезапуск системи.

Якщо $x(k) = C$, soft reset неефективний ⇒ не передається керуючий сигнал HW WDT ⇒ глобальне скидання.

У підсумку математична схема визначається наступними співвідношеннями. Стан системи $x(k)$ – дискретна змінна у скінченому просторі S. Закони переходів визначаються матрицею P, яку оновлюємо інкрементально чи у ковзному вікні:

$$p_{ij} = \frac{n_{(i \to j)\cdot}}{n_i}, \tag{10}$$

де p_{ij} – ймовірність переходу системи зі стану i у стан j за один тактовий крок, $n_{(i \to j)}$– кількість спостережених переходів безпосередньо зі стану i до стану j у межах поточного вибіркового вікна, n_i – загальна кількість фіксацій стану i як вихідного протягом того самого вікна. Ймовірність збою за один крок – це p_{iF} (з "Critical" у "Fail" чи "Warning" у "Fail"). При перевищенні порога θ – запускаємо превентивні дії (soft reset). За необхідності розглядаємо прогноз на кроків через P^r. Soft reset еквівалентний примусовому поверненню системи з небезпечного стану в "Normal" (або "Recovery"). Повне скидання через апаратний сторожовий таймер відбувається, якщо система сповіщає про критичну проблему, але програмне скидання не дає результату.

Алгоритм роботи завдання прогнозного сторожового таймера на основі марковського ланцюгаподано на рисунку 1 у вигляді блок-схеми, що відображає послідовність дій від ініціалізації до вибору стратегії

відновлення. У середовищі FreeRTOS це завдання здійснює прогнозний контроль стану системи за марковською моделлю: після запуску моніторингу аналізуються поточні параметри виконання, оцінюється ймовірність збою, і в разі її перевищення виконується локальне програмне перезапускання (soft reset). Якщо система стабілізується, моніторинг триває; у протилежному випадку активується апаратний сторожовий таймер (hard reset), після якого цикл контролю повторюється, забезпечуючи безперервну працездатність системи.

Рис. 1. Алгоритм роботи завдання прогнозного сторожового таймера.

Підготовка прототипу. Експериментальні дослідження виконувалися на фізичній платформі STM32F4 під керуванням операційної системи реального часу FreeRTOS. Обрана апаратна база належить до популярного класу мікроконтролерів з архітектурою ARM Cortex-M4, які поєднують достатню продуктивність із низьким енергоспоживанням та обмеженими обчислювальними ресурсами. Використання FreeRTOS забезпечило реалізацію типових механізмів багатозадачності — планувальника, черг повідомлень, семафорів і системи пріоритетів, що дало змогу імітувати поведінку критичних прикладних систем у реальному часі. Така конфігурація апаратного та програмного забезпечення створила умови для моделювання реалістичного середовища функціонування ресурсно обмежених кіберфізичних систем і дозволила експериментально оцінити ефективність запропонованого механізму відмовостійкості.

Тестовий стенд включав центральний процесор, що виконував планування задач, периферію SPI, на якій моделювалися перевантаження, інтерфейс UART для журналювання й діагностики, а також GPIO-лінії, які імітували зовнішні сигнали сенсорів і генерували асинхронні переривання. У середовище FreeRTOS було інтегровано спеціальний програмний модуль PredictiveWDT, який виконував функції прогнозного моніторингу, аналізу станів системи та ініціювання автоматичного відновлення у разі виявлення збоїв або аномальної поведінки.

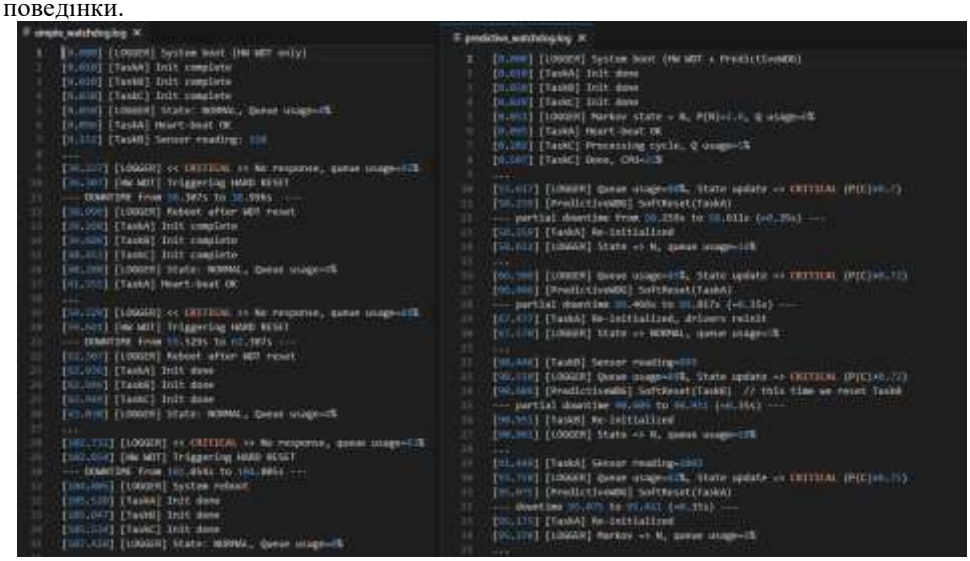

Рис. 2. Фрагмент логування роботи системи з використанням і без використання прогнозного сторожового таймера протягом п’яти хвилин за умов випадкових збоїв.

Проведення експериментів і порівняння ефективності. У першій серії випробувань досліджувалася робота системи лише з класичним апаратним сторожовим таймером (Hardware Watchdog). Для цього імітувалося зависання задачі TaskA, яка обслуговувала периферію SPI, а також імітація взаємного блокування в TaskA. У випадку, коли TaskA переставала відповідати, апаратний WDT спрацьовував тільки

після завершення встановленого тайм-ауту (2,0 с). Після цього відбувався повний перезапуск мікроконтролера з повторною ініціалізацією всіх системних компонентів, що займало ще 0,6–0,8 с. Таким чином, загальний час відновлення працездатності становив у середньому близько 2,7 с. Роботу системи у першій серії експериментів (лише з апаратним сторожовим таймером) ілюструє рисунок 2 ліва частина, де приведено логування подій апаратного сторожовго таймера.

Сумарний простій:

$$5 \times 2.7 = 13.5 \text{ с} \quad (11)$$

Відсоток простою за 5 хвилин (300 с):

$$\frac{13.5}{300} \times 100\% = 4.5\% \quad (12)$$

У другій серії тестів досліджується робота системи з прогнозним сторожовим таймером. Проводяться ті самі дії. TaskA може зависати з імітацією перевантаження периферії SPI, або імітацією взаємного блокування. Система з PredictiveWDT виявляла критичний стан у середньому за 0,11–0,13 с, бо період опитування становив 100 мс. Зазвичай вистачало 1–2 циклів, аби змінити оцінку стану від Normal/Warning до Critical. Після переходу в Critical з високою імовірністю переходу в F(Fail) (зокрема,) PredictiveWDT виконував програмне скидання TaskA, перевантажуючи пов'язані драйвери. Така операція займала близько 0,3–0,4 с, тож загальний час повернення в Normal не перевищував 0,5 с. Також була імітація взаємного блокування в TaskA, коли блокування було настільки глибоким, що навіть програмне скидання з повторним створенням задачі не вирішувало проблему. Роботу системи можна побачити на рисунку 2 в правій частині, де приведено логування подій прогнозного сторожового таймера. У загальних 5 спробах лише в одному випадку система залишалася в неробочому стані локально. В цьому випадку PredictiveWDT свідомо не передавав керуючий сигнал апаратному сторожовому таймеру, що призводило до його спрацювання та повного перезапуску системи. Решта 4 збої виправлялись саме через локальний перезапуск. Таким чином частка глобальних перезапусків становила 100%/5*1=20% (рисунок 2 логування справа). За чисто реактивного підходу (відсутність прогнозного сторожового таймеру) методу планування задач в реальному часі у подібних сценаріях завжди 100%, оскільки система чекала тільки на апаратного сторожового таймера.

Сумарний простій:

$$(4\times0.35)+(1\times3)=1.4+2.7 =4.1 \text{ с} \quad (13)$$

Відсоток простою за 5 хвилин (300 с):

$$\frac{4.1}{300} \times 100\% = \sim 1.36666\% \quad (14)$$

Таким чином, зменшення часу простою становить:

$$4.5\ \% - 1.36666\ \% = 4.5\% - 1.36666\% = 3.13334\% \quad (15)$$

Розрахуємо у відносних показниках скорочення простою між першим тестом і другим:

$$\frac{4.5-1.36666}{4.5} \times 100\% = 69.62978\% \quad (16)$$

Щодо навантаження на процесор, максимальне значення, зафіксоване засобом FreeRTOS+Trace, становило близько 1,6% у момент, коли завдання TaskA генерувало велику кількість подій, а модуль PredictiveWDT одночасно обробляв лічильники переходів у матриці станів. У середньому додаткове навантаження залишалося на рівні нижче 1,4%, що практично не впливало на виконання інших завдань, враховуючи роботу ядра з частотою 180 МГц.

Результати експерименту підтвердили ефективність запропонованої стратегії: регулярне м'яке перезапускання окремих компонентів забезпечує істотне скорочення середнього часу простою — у 5–8 разів менше порівняно з повним перезавантаженням системи. При цьому зберігається жорсткий рівень захисту від фатальних збоїв, що гарантує стабільність навіть у критичних сценаріях. Збільшення часу виконання є незначним і не впливає на продуктивність у відносних показниках. Таким чином, поєднання простої стохастичної оцінки ризику на основі марковської моделі з двоетапним програмно-апаратним механізмом забезпечує підвищення надійності та швидкості відновлення у вбудованих і кіберфізичних системах без суттєвого збільшення обчислювальних або пам'ятних витрат.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

У роботі запропоновано двоступеневий метод забезпечення кіберстійкості та відмовозахисту для вбудованих і кіберфізичних систем. Архітектура поєднує апаратний сторожовий таймер — як останню лінію захисту від фатальних збоїв — із програмним модулем прогнозний сторожовий таймер, що використовує спрощений марковський ланцюг для прогнозування ризику відмови. Система проактивно визначає ймовірність переходу до критичного стану F(Fail) і виконує локальне програмне скидання проблемного компонента, запобігаючи повному перезавантаженню. У випадку неефективності превентивних дій ініціюється апаратне відновлення, що гарантує захист навіть при кібератаках на рівні ядра.

Експериментальні результати підтвердили ефективність підходу: час відновлення скоротився у 3,1 раза — з 2,7 до 0,84 секунди, а близько 80% потенційних збоїв усувалося локально без апаратного скидання. Загальний відсоток простою зменшився з 4,5% до 1,37%, тобто на ≈ 69,6%. Навантаження на процесор не перевищувало 1–2%, що свідчить про придатність рішення для мікроконтролерів із обмеженими ресурсами. Запропонований підхід поєднує швидкість, малу ресурсоємність і здатність одночасно підвищувати відмовостійкість та кіберзахист, що робить його ефективним для застосування у промисловій автоматизації, робототехніці, автомобільних і медичних системах.

Література

1. Shideh Saraeian, Babak Shirazi, Digital twin-based fault tolerance approach for Cyber–Physical Production System, ISA Transactions, Volume 130, 2022, Pages 35-50, ISSN 0019-0578, https://doi.org/10.1016/j.isatra.2022.03.007.

2. Benjamin Asch, Erling Jellum, Marten Lohstroh, and Edward A. Lee. (2024). Software-Defined Watchdog Timers for Cyber-Physical Systems. In IEEE Embedd. Syst. Letters, 18(9). doi:10.1109/LES.2024.3467332.

3. Michel F, Siegle M. Formal error bounds for the state space reduction of Markov chains. Performance Evaluation. 2025;167:102464. doi:10.1016/j.peva.2024.102464.

4. Amel Solouki M, Angizi S, Violante M. Dependability in embedded systems: a survey of fault-tolerance methods and software-based mitigation techniques. IEEE Access. 2024;12:1-35. doi:10.1109/ACCESS.2024.3509633.

5. Magliano E, Savino A, Di Carlo S. Real-time embedded system fault injector framework for micro-architectural state-based reliability assessment. J Electron Test. 2025;41:193-208. doi:10.1007/s10836-025-06170-w.

6. Mercorelli P. Recent advances in intelligent algorithms for fault detection and diagnosis. Sensors (Basel). 2024;24(8):2656. doi:10.3390/s24082656.

7. D. A. Santos et al., "Hybrid Hardening Approach for a Fault-Tolerant RISC-V System-On-Chip", in IEEE Transactions on Nuclear Science, vol. 71, no. 8, pp. 1722-1730, Aug. 2024, doi: 10.1109/TNS.2024.3406021.

8. Upadhyaya S, Dharaskar K. Prognostics and health management for cyber-physical embedded systems: concepts, challenges and future directions. IEEE Trans Ind Inform. 2024;20(7):8772-8784. doi:10.1109/TII.2023.3348127.

9. Zhou Y, Li X, Peng C. Real-time response-time monitoring for predictive maintenance in multi-core RTOS. ACM Trans Embed Comput Syst. 2025;24(3):40:1-40:22. doi:10.1145/3726653.

10. Mehalaine R, Djezzar M, Nessah D, Saiad Z, Saidi A. Watchdog Timer for Fault Tolerance in Embedded Systems. J Eur Syst Autom. 2024;57(6):1713-1720. doi:10.18280/jesa.570619.

11. Reghenzani F. Software Fault Tolerance in Real-Time Systems: Identifying the Future Research Questions [Text] / F. Reghenzani, Z. Guo, W. Fornaciari // ACM Computing Surveys. – 2023. – Vol. 55, Issue 14s. – Art. 306. – 30 p. DOI: 10.1145/3589950. ISSN: 0360-0300.

12. Ratti F. HeterogeneousRTOS: A CPU-FPGA Real-Time OS for Fault Tolerance on COTS at Near-Zero Timing Cost [Text] / F. Ratti // ACM Transactions on Embedded Computing Systems. – 2025. – Vol. 24, No. 5. – Art. 144. DOI: 10.1145/3712062. ISSN: 1539-9087.

13. Simpson M. A DoS-Resilient “Smart Watchdog” for RISC-V Microcontrollers [Text] / M. Simpson, A. Pindzola, N. Patel, H. Mager, V. Tsafta, J. Hillman, L. Becchetti, S. Loke, G. V. Merrett // Proc. IEEE International Symposium on Circuits and Systems (ISCAS). – 2025. – PP. 1–5. DOI: 10.1109/ISCAS62022.2025.10678957.

14. Lee S. IndWatch: Industrial Watchdogs Based on Blockchain for Securing Software Supply Chains in IIoT [Text] / S. Lee, S.-H. Kwon // Sensors. – 2021. – Vol. 21, No. 4. – Art. 1170. DOI: 10.3390/s21041170. ISSN: 1424-8220.

15. Kozelskyi, O., Drozd, A., Savenko, B., & Gaj, P. (2025). A model for probabilistic monitoring and proactive restart of real-time operating systems under intensive state changes in cyber-physical systems. In CEUR Workshop Proceedings (Vol. 4013). Режим доступу: https://ceur-ws.org/Vol-4013/paper16.pdf.

16. D. Denysiuk, O. Savenko, S. Lysenko, B. Savenko and A. Kashtalian, "Method for Detecting Steganographic Changes in Images Using Machine Learning," 2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT), Athens, Greece, 2023, pp. 1-6, doi: 10.1109/DESSERT61349.2023.10416453.

17. B. Savenko, A. Kashtalian, S. Lysenko and O. Savenko, "Malware Detection By Distributed Systems with Partial Centralization," 2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Dortmund, Germany, 2023, pp. 265-270, doi: 10.1109/IDAACS58523.2023.10348773.

18. L. Bedratyuk, O. Savenko, MATCH Commun. Math. Comput. Chem. 2018, 79, 407–414.

19. Stetsiuk, M., Anikin, V., Pyrch, O., & Salem, Abdel-Badeeh M. (2025). Method of detecting anomalies in IOT device traffic based on statistical analysis using the modified z score. In CEUR Workshop Proceedings (Vol. 3963). Режим доступу: https://ceur-ws.org/Vol-3963/paper23.pdf.

References

1. Shideh Saraeian, Babak Shirazi, Digital twin-based fault tolerance approach for Cyber–Physical Production System, ISA Transactions, Volume 130, 2022, Pages 35-50, ISSN 0019-0578, https://doi.org/10.1016/j.isatra.2022.03.007.

2. Benjamin Asch, Erling Jellum, Marten Lohstroh, and Edward A. Lee. (2024). Software-Defined Watchdog Timers for Cyber-Physical Systems. In IEEE Embedd. Syst. Letters, 18(9). doi:10.1109/LES.2024.3467332.

3. Michel F, Siegle M. Formal error bounds for the state space reduction of Markov chains. Performance Evaluation. 2025;167:102464. doi:10.1016/j.peva.2024.102464.

4. Amel Solouki M, Angizi S, Violante M. Dependability in embedded systems: a survey of fault-tolerance methods and software-based mitigation techniques. IEEE Access. 2024;12:1-35. doi:10.1109/ACCESS.2024.3509633.

5. Magliano E, Savino A, Di Carlo S. Real-time embedded system fault injector framework for micro-architectural state-based reliability assessment. J Electron Test. 2025;41:193-208. doi:10.1007/s10836-025-06170-w.

6. Mercorelli P. Recent advances in intelligent algorithms for fault detection and diagnosis. Sensors (Basel). 2024;24(8):2656. doi:10.3390/s24082656.

7. D. A. Santos et al., "Hybrid Hardening Approach for a Fault-Tolerant RISC-V System-On-Chip", in IEEE Transactions on Nuclear Science, vol. 71, no. 8, pp. 1722-1730, Aug. 2024, doi: 10.1109/TNS.2024.3406021.

8. Upadhyaya S, Dharaskar K. Prognostics and health management for cyber-physical embedded systems: concepts, challenges and future directions. IEEE Trans Ind Inform. 2024;20(7):8772-8784. doi:10.1109/TII.2023.3348127.

9. Zhou Y, Li X, Peng C. Real-time response-time monitoring for predictive maintenance in multi-core RTOS. ACM Trans Embed Comput Syst. 2025;24(3):40:1-40:22. doi:10.1145/3726653.

10. Mehalaine R, Djezzar M, Nessah D, Saiad Z, Saidi A. Watchdog Timer for Fault Tolerance in Embedded Systems. J Eur Syst Autom. 2024;57(6):1713-1720. doi:10.18280/jesa.570619.

11. Reghenzani F. Software Fault Tolerance in Real-Time Systems: Identifying the Future Research Questions [Text] / F. Reghenzani, Z. Guo, W. Fornaciari // ACM Computing Surveys. – 2023. – Vol. 55, Issue 14s. – Art. 306. – 30 p. DOI: 10.1145/3589950. ISSN: 0360-0300.

12. Ratti F. HeterogeneousRTOS: A CPU-FPGA Real-Time OS for Fault Tolerance on COTS at Near-Zero Timing Cost [Text] / F. Ratti // ACM Transactions on Embedded Computing Systems. – 2025. – Vol. 24, No. 5. – Art. 144. DOI: 10.1145/3712062. ISSN: 1539-9087.

13. Simpson M. A DoS-Resilient “Smart Watchdog” for RISC-V Microcontrollers [Text] / M. Simpson, A. Pindzola, N. Patel, H. Mager, V. Tsafta, J. Hillman, L. Becchetti, S. Loke, G. V. Merrett // Proc. IEEE International Symposium on Circuits and Systems (ISCAS). – 2025. – PP. 1–5. DOI: 10.1109/ISCAS62022.2025.10678957.

14. Lee S. IndWatch: Industrial Watchdogs Based on Blockchain for Securing Software Supply Chains in IIoT [Text] / S. Lee, S.-H. Kwon // Sensors. – 2021. – Vol. 21, No. 4. – Art. 1170. DOI: 10.3390/s21041170. ISSN: 1424-8220.

15. Kozelskyi, O., Drozd, A., Savenko, B., & Gaj, P. (2025). A model for probabilistic monitoring and proactive restart of real-time operating systems under intensive state changes in cyber-physical systems. In CEUR Workshop Proceedings (Vol. 4013). Retrieved from https://ceur-ws.org/Vol-4013/paper16.pdf.

16. D. Denysiuk, O. Savenko, S. Lysenko, B. Savenko and A. Kashtalian, "Method for Detecting Steganographic Changes in Images Using Machine Learning," 2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT), Athens, Greece, 2023, pp. 1-6, doi: 10.1109/DESSERT61349.2023.10416453.

17. B. Savenko, A. Kashtalian, S. Lysenko and O. Savenko, "Malware Detection By Distributed Systems with Partial Centralization," 2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Dortmund, Germany, 2023, pp. 265-270, doi: 10.1109/IDAACS58523.2023.10348773.

18. L. Bedratyuk, O. Savenko, MATCH Commun. Math. Comput. Chem. 2018, 79, 407–414.

19. Stetsiuk, M., Anikin, V., Pyrch, O., & Salem, Abdel-Badeeh M. (2025). Method of detecting anomalies in IOT device traffic based on statistical analysis using the modified z score. In CEUR Workshop Proceedings (Vol. 3963). Retrieved from https://ceur-ws.org/Vol-3963/paper23.pdf.