DOI: 10.32347/2412-9933.2025.63.212-222
УДК 004.2

Юрович Іван Васильович
Аспірант кафедри системного програмування і спеціалізованих комп'ютерних систем,
https://orcid.org/0009-0005-9575-065X
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ
Зайцев Володимир Григорович
Доктор технічних наук, професор, професор кафедри системного програмування і спеціалізованих комп'ютерних систем, *https://orcid.org/0009-0009-9917-5364*
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ

МЕТОД ВИЗНАЧЕННЯ ЧАСУ ВИКОНАННЯ ПРОГРАМ У СИСТЕМАХ РЕАЛЬНОГО ЧАСУ

Анотація. *Проблема функціонування комп'ютерних систем реального часу, зокрема таких критичних до збоїв, як адаптивний круїз-контроль, полягає не лише в логічній коректності, а й у своєчасній видачі результатів, де будь-які затримки можуть мати критичні наслідки. Існуючі моделі часто спрощують систему, що моделюється, та не враховують динамічну природу появи завдань, що є невід'ємною складовою сучасних систем реального часу. У даній статті запропоновано метод визначення часу виконання завдань у системах реального часу, який враховує динамічну природу появи завдань під час роботи системи. Метод деталізує різні типи подій (асинхронні, синхронні, ізохронні) та типи завдань за характером виникнення (періодичні, спорадичні, аперіодичні), а також розглядає різні види залежностей між завданнями, включаючи умовні активації. Для реалістичної оцінки часу виконання завдань використовуються дані, отримані з досліджень аналогів до компонентів системи АКК (сенсори, контролери, актуатори). Визначені таким чином часові характеристики є основою для подальшого моделювання роботи системи, що дозволяє обрати алгоритм планувальника завдань, який задовольнятиме часові вимоги системи. Це гарантує відповідність системи її жорстким вимогам щодо затримки, забезпечуючи надійне та безпечне функціонування, а також дозволяє виявляти потенційні проблеми на ранніх етапах розробки системи.*

Ключові слова: операційна система реального часу; адаптивний круїз-контроль; час виконання завдання; сенсори; актуатори; моделювання

Постановка проблеми

Операційні системи реального часу (ОСРЧ) є спеціалізованими операційними системами, які забезпечують обробку завдань у реальному часі. Вони призначені для систем, де важлива точність і швидкість обробки даних, таких як автоматизовані системи управління, вбудовані системи, медичне обладнання, промислові контролери та інші.

ОСРЧ мають критичні вимоги до обробки завдань у реальному часі, де затримки можуть призводити до серйозних наслідків, наприклад, втрати даних, нестабільної роботи системи або навіть аварій. Ці операційні системи гарантують виконання завдань у заданих часових межах і забезпечують низьку затримку (латентність) обробки.

Розробка методів, моделей та комп'ютерних засобів для оцінки часу виконання програм у системах реального часу є важливим завданням для забезпечення коректної та надійної роботи таких систем. Створення алгоритму для операційної системи реального часу (ОСРЧ) може бути складним завданням, оскільки ОСРЧ має специфічні вимоги до обробки завдань у реальному часі. Підходи та засоби, які використовуються у цьому контексті, включають:

– *Аналіз часу виконання (Timing Analysis)*: цей підхід включає в себе аналіз програмного коду з метою визначення часових характеристик, таких як час виконання, затримки та обсяг використання ресурсів. Використовуються різні методи, включаючи символьний виконавчий аналіз, аналіз залежності даних та статичний аналіз коду.

– *Симуляція (Simulation)*: симуляційні моделі систем реального часу можуть бути використані для оцінки часових характеристик програм. Вони дозволяють відтворити роботу системи та виконання програм в умовах реального часу і аналізувати результати. Симуляційні інструменти можуть забезпечувати інформацію про час виконання, затримки та використання ресурсів.

– *Моделювання через сітку Петрі (Petri Nets)*: сітка Петрі може бути використана для моделювання систем реального часу та оцінки їхніх часових характеристик. Модель сітки Петрі дозволяє представити процеси, події та взаємодію компонентів системи, а симуляція сітки Петрі дозволяє аналізувати часові характеристики, такі як час виконання, затримки та пропускна спроможність.

– *Формальна верифікація (Formal Verification)*: формальні методи верифікації можуть бути використані для оцінки часу виконання програм у системах реального часу. Це включає в себе математичне доведення або перевірку правильності програмного коду щодо вимог щодо часу виконання та затримок.

– *Інструменти для аналізу та профілювання*: існують різні інструменти, які допомагають аналізувати час виконання програм та оцінювати їхні часові характеристики. Це можуть бути інструменти для профілювання коду, вимірювання часу виконання, аналізу затримок та ресурсного використання.

Ці методи, моделі та засоби допомагають розробникам систем реального часу оцінити та встановити часові характеристики програм, що дозволяє гарантувати вчасне та надійне виконання завдань у таких системах. Сітка Петрі має кілька переваг, які роблять її корисною у цьому контексті:

– *Моделювання паралельних процесів*: сітка Петрі надає потужність моделювання паралельних процесів, що є важливим аспектом систем реального часу. Вона дозволяє представити події, які одночасно відбуваються, та їх взаємодію, що допомагає аналізувати та оцінювати часові характеристики системи.

– *Візуалізація та аналіз*: сітка Петрі забезпечує графічну візуалізацію моделі системи реального часу, що дозволяє легко розуміти та аналізувати взаємодію компонентів та процесів. Вона допомагає ідентифікувати можливі проблеми з часовими характеристиками, такі як затримки, конфлікти ресурсів та перевантаження.

– *Симуляція та експерименти*: сітка Петрі дозволяє проводити симуляції та експерименти з моделлю системи реального часу. Це дозволяє оцінити та перевірити різні сценарії роботи системи, що допомагає прогнозувати та визначати часові характеристики, такі як час виконання завдань, затримки та пропускна спроможність.

– *Виявлення проблем та оптимізація*: сітка Петрі дозволяє виявити можливі проблеми з часовими характеристиками системи та виконати оптимізацію. Вона допомагає ідентифікувати вузькі місця, забезпечити ефективне розподілення ресурсів, вдосконалити алгоритми планування та управління, щоб гарантувати вчасну обробку завдань.

– *Формальна верифікація*: сітка Петрі може бути використана для формальної верифікації системи реального часу. Вона дозволяє перевірити відповідність системи вимогам щодо часової точності та надійності, а також виявити можливі конфлікти або некоректні поведінки.

Усі ці фактори роблять сітку Петрі потужним інструментом для оцінки та аналізу часових характеристик систем реального часу. Вона допомагає розуміти та вдосконалювати роботу системи, забезпечуючи вчасну та надійну обробку завдань. Натомість, точна оцінка часових характеристик систем реального часу є важливою з кількох причин:

– гарантування вчасної обробки: системи реального часу виконують завдання з точними часовими обмеженнями. Оцінка часових характеристик допомагає впевнитися, що система здатна виконувати завдання відповідно до вимог щодо часу відгуку, термінів виконання та інших критичних параметрів.

– виявлення проблем та оптимізація: оцінка часових характеристик дозволяє виявити потенційні проблеми, такі як затримки або перевантаження, і вжити заходів для їх виправлення. Вона допомагає виявити вузькі місця, забезпечити ефективне використання ресурсів і вдосконалити алгоритми роботи системи.

– гарантування надійності: оцінка часових характеристик допомагає забезпечити надійну роботу системи в реальному часі. Вона дозволяє виявити можливі проблеми, такі як конфлікти ресурсів або недостатній час для виконання завдань, і забезпечити належне управління ресурсами та плануванням завдань.

– валідація та сертифікація: Оцінка часових характеристик є важливим етапом у процесі валідації та сертифікації систем реального часу. Вона допомагає підтвердити, що система відповідає вимогам щодо точності та надійності виконання завдань.

Саме тому оцінка часових характеристик систем реального часу є ключовою для забезпечення коректної та надійної роботи таких систем, як автоматизовані системи управління, вбудовані системи, медичні пристрої, транспортні системи та інші, де часова точність є критично важливою.

Проте, наразі оцінка часових характеристик ОСРЧ за допомогою сітки Петрі має свої недоліки:

– Складність моделювання: створення точної моделі сітки Петрі для складної ОСРЧ може бути часо- та працезатратним завданням. Моделювання всіх аспектів системи, включаючи паралельні процеси, ресурси, затримки та інші фактори, може вимагати значних зусиль і досвіду. Крім того, підтримка та зміна моделі можуть бути складними, особливо при зміні структури або функціональності ОСРЧ.

– Розмір та складність моделей: сітки Петрі можуть стати дуже великими і складними, особливо при моделюванні великих ОСРЧ або систем з багатьма взаємодіючими компонентами. Обробка та аналіз таких моделей може стати обтяжливим та вимагати значних обчислювальних ресурсів.

– Абстракція та спрощення: моделі сітки Петрі часто потребують певного рівня абстракції та спрощення, щоб бути реалістичними та працездатними. Це може призвести до втрати деякої деталізації та точності при оцінці часових характеристик. Наприклад, певні нюанси реального середовища виконання, такі як розподілення пам'яті, операційна система та інші системні фактори, можуть бути спрощені або ігноровані в моделі.

– Залежність від вхідних даних: оцінка часових характеристик за допомогою сітки Петрі може вимагати заздалегідь відомих або оцінених значень для параметрів системи, таких як часові затримки, пропускна спроможність ресурсів тощо. Але точні значення цих параметрів можуть бути складними або непередбачуваними для великих та складних ОСРЧ.

– Обмеження формальності: хоча сітка Петрі має формальну основу, її використання для оцінки часових характеристик може потребувати додаткових припущень, евристик та наближень. Це може вплинути на точність та надійність отриманих результатів.

Метод, запропонований у даній роботі, допомагає позбутися, певною мірою, цих недоліків. Замість класичного моделювання усього процесу, що може бути доволі складним заняттям і породжує деяку кількість похибок, пропонується більш оптимізований підхід, де весь процес розбивається на повторювані підпроцеси, а їх своєю чергою можна розбити ще на повторювані підпроцеси, поки не дійдемо до декількох найменших неповторюваних алгоритмів, описати їх за допомогою сітки Петрі і врахувати в них усі додаткові фактори системи, а далі побудувати моделювання на рівні «блоків» із таких алгоритмів, за рахунок чого значно спрощується моделювання, а також суттєво зменшується кількість похибок, оскільки додаткові фактори враховані вже в самому «блоці» і за рахунок цього не є унікальними на всю систему. Цей метод можливий за рахунок самої природи ОСРЧ, оскільки в більшості випадків ОСРЧ виконує циклічно декілька алгоритмів, які мають між собою описані та легко формалізовані залежності.

Звичайно, неможливо абсолютно точно спрогнозувати, як буде себе поводити ОСРЧ без реальних досліджень з участю реальної апаратної та програмної складової, але такий спосіб моделювання дозволяє оцінити часові характеристики ОСРЧ на заданій конфігурації, і доволі точно оцінити, які конкретно параметри апаратної/програмної конфігурації системи не підійдуть або будуть критично навантаженими, а також порахувати, на якому саме місці алгоритму буде найбільше навантаження та потенційні ризики відмови системи або некоректної її роботи.

Для досягнення достовірного результату потрібно, щоб модель системи мала ідентичну поведінку та часові характеристики відповідно до системи, яка проектується. Проте, врахування всіх можливих подій та умов зовнішнього середовища під час розробки моделі є досить складною і нетривіальною задачею, адже ці дані залежать від середовища, де буде функціонувати конкретна система. Саме тому під час розробки моделі дослідники вдаються до деяких спрощень, як-от, наприклад, перехід від аналогового часу до дискретного. Також для симуляції подій зовнішнього світу можуть застосовуватися штучні події, що були згенеровані випадковим чином. Проте події, що продукуються лише випадковим чином, не завжди можуть покрити всі теоретичні ситуації, які можуть трапитися у натуральному середовищі. Тому дуже важливим кроком у дослідженні СРЧ є перевірка найгірших сценаріїв, що можуть трапитися (WCET).

У попередній роботі [1, с. 151] було запропоновано метод, що допомагає на етапі проектування системи визначити час появи завдань у системах реального часу, який у подальшому може бути використаний для моделювання роботи системи. Суть методу полягає у тому, що потрібно розбити алгоритм роботи системи на завдання, зрозуміти їх послідовність та залежності. Після розбиття системи на завдання пропонується створити блок-схему роботи алгоритму, яка буде містити у собі завдання і їх послідовність. Після цього, використовуючи схему завдань, можна наочно визначити час появи завдань. Проте, для повного моделювання СРЧ потрібно знати такі часові характеристики завдань систем реального часу, як час появи завдання, час його виконання, а також кінцевий час виконання завдання. У даній статті запропоновано метод визначення часу виконання завдань систем реального часу, що враховує динамічну природу появи завдань.

Аналіз основних досліджень і публікацій

За останній час було здійснено низку досліджень про визначення часових характеристик завдань у системах реального часу. Зазначається, що сітки Петрі можуть бути корисними для визначення відповідності системи заданим їй часовим вимогам [2, с. 49], [3, с. 79], [4, с. 45] ще на етапі проєктування системи. Нещодавні дослідження доводять [2, с. 60], що на початкових етапах розробки системи можна

визначити терміни виконання завдань та вибрати метод реалізації програмного і апаратного забезпечення, який буде відповідати заданим вимогам. У вищезгаданій роботі автори запропонували спосіб оцінки часових характеристик завдань систем реального часу, аналізуючи дані, які були отримані під час моделювання. Таким чином, при моделюванні системи відбувався розподіл процесорного часу між завданнями, згідно з обраними алгоритмами планування завдань та, використовуючи програмну модель системи, що базується на сітці Петрі. Метод гарантував визначення відповідності часових характеристик завдань для різних типів планувальників, що є потрібним для початку технічного проєктування системи реального часу. Використання запропонованих інструментальних засобів може допомогти значно зменшити терміни аналізу можливих варіантів реалізації системи реального часу, а також підвищити якість проєкту та суттєво зменшити загальні терміни і вартість усієї розробки. Проте, раніше розроблені моделі не враховували динамічну природу появи завдань, що є невід'ємною складовою новітніх систем реального часу.

Мета статті

Мета статті полягає в розробленні способу визначення часу виконання завдань комп'ютерних систем реального часу, що враховуватиме динамічну природу появи завдань під час роботи системи.

Виклад основного матеріалу

Часові параметри завдань СРЧ [1, с. 150] є ключовими для визначення відповідності системи заданим часовим вимогам, а саме для визначення термінів виконання завдань та обрання оптимального програмного і апаратного забезпечення. Такі параметри, як момент появи завдання, крайній термін виконання та час виконання завдання повинні бути відомими ще до початку моделювання або розробки системи та залежать від специфіки компонентів і навколишнього середовища СРЧ. У свою чергу, інші параметри є похідними і залежать від реалізації СРЧ, тобто її програмного і апаратного оснащення, зокрема від планувальника завдань.

Похідні параметри завдань можна визначити за допомогою методів оцінки часових характеристик завдань у системах реального часу, що виконуються шляхом аналізу даних, отриманих після моделювання розподілу процесорного часу між завданнями, згідно з обраними алгоритмами планувальника, з використанням моделі сіток Петрі для однопроцесорних [2, с. 52] та багатопроцесорних систем [4, с. 45]. Методи гарантують отримання часових характеристик завдань при обранні конкретного типу процесора і планувальника, що є необхідним для початку технічного проєктування системи реального часу. Проте, раніше розроблені моделі не враховували динамічну природу появи завдань, що є невід'ємною складовою новітніх систем реального часу.

У попередній роботі був представлений спосіб визначення часу появи завдань систем реального часу, який враховує взаємозв'язки між завданнями [1, с. 151]. Спосіб полягає в тому, щоб розбити алгоритм роботи системи на окремі завдання та детально проаналізувати кожну з них. Для цього необхідно визначити послідовність виконання завдань, а також їхні взаємозалежності. Такий підхід дозволяє чітко зрозуміти, як кожне завдання впливає на інші, які ресурси воно потребує і які часові обмеження мають бути дотримані.

Наприклад, представлений раніше алгоритм роботи адаптивного круїз-контролю [1, с. 152] має деякий набір завдань, що виконують певні функції: введення даних, обробка даних та керування складовими системи. Окрім функціонального розподілу завдань, для достовірного аналізу важливим чинником також є типи завдань за послідовністю виконання, характером виникнення та типом залежності між завданнями у системі.

Для коректної реалізації запропонованого методу такі параметри як момент появи завдання, крайній термін виконання та час виконання завдання повинні бути відомими ще до початку моделювання. Для цього потрібно розуміти природу появи завдань, порядок їхнього виконання та періодичність появи. Залежно від цих характеристик завдання поділяються на декілька типів.

Типи завдань та подій у системах реального часу

У системах реального часу завдання можуть мати різну природу появи, тип і поведінку. Від розуміння цих характеристик залежить точність оцінки часових параметрів і здатність системи дотримуватись заданих часових обмежень. У традиційних підходах до моделювання та аналізу завдань СРЧ враховуються статичні завдання з наперед відомими параметрами. Проте, в реальних умовах завдання можуть виникати динамічно, мати умовні залежності та різні джерела появи.

Типи подій

- *Асинхронні події*. Це події, які неможливо передбачити за часом [5, с. 94]. Такі події можуть ініціювати появу завдань, які не враховувались у початковому плануванні. Прикладом є зміна користувачем бажаної дистанції між авто. У системах реального часу зовнішні події вимагають негайної реакції процесора. Отримання даних по перериванню відбувається асинхронно: інтерфейсний пристрій, що

одержав нові дані, звертається до центрального процесора, посилаючи йому сигнал переривання через системну шину. Обробка переривань є критичною до часу і, як правило, вимагає заборони переривань, щоб уникнути перемикання процесів під час її виконання.

У класичних підходах до аналізу часу виконання подібні події зазвичай моделюються за найгіршим сценарієм (*worst-case*), без урахування контексту або умов, які могли б уточнити їхню ймовірність чи наслідки.

- *Синхронні події*. Це передбачувані події, які трапляються регулярно або є результатом системного таймера [5, с. 95]. Наприклад, зчитування відстані до автомобіля попереду. Їхні часові характеристики відомі заздалегідь, і їх легко планувати. У складніших сценаріях такі події можуть служити часовим маркером, що активує залежні завдання.
- *Ізохронні події*. Це проміжний тип між синхронними та асинхронними подіями. Вони відбуваються з регулярністю в межах допустимого інтервалу. Наприклад, під час екстреного інтенсивного гальмування може спрацьовувати система запобігання блокуванню коліс (ABS). Після того, як інтенсивність гальмування зменшується, ABS вимикається.

Типи завдань за характером виникнення

- *Періодичні завдання*. Це завдання, необхідність виконання яких виникає в моменти часу, що повторюються через рівні інтервали [5, с. 29]. Наприклад, оновлення показників датчика кожні 100 мс. У розглянутому алгоритмі АКК завдання T2 та T7 є періодичними, оскільки вони викликаються таймером через рівні проміжки часу для постійного моніторингу дорожньої ситуації. Кожне періодичне завдання A_i характеризується обсягом (часом) обчислення e_i та періодом T_i, що означає, що моменти запуску завдання A_i чергуються з інтервалом T_i.
- *Спорадичні завдання*. Це неперіодичні завдання з жорстким крайнім терміном, необхідність у виконанні яких виникає у випадкові моменти часу [5, с. 29]. Вони виконуються нерегулярно або у відповідь на певні події чи умови. У системах з умовним породженням завдань спорадичні завдання можуть бути результатом або тригером появи інших завдань, що створює ланцюгові залежності, не враховані в класичних моделях. Таким чином, завдання T3 (обчислення зміни швидкості) та T8 (обчислення прискорення поточного авто) є спорадичними, оскільки вони виникають лише за певних умов у випадкові моменти часу. Наприклад, T3 тільки після виконання T1 і T2 і за умови, що потрібна зміна швидкості авто.
- *Аперіодичні завдання*. Це неперіодичні завдання з м'яким крайнім терміном. Вони, як і спорадичні, виникають лише за певних умов, а не завжди перебувають у черзі виконання [5, с. 29]. До таких завдань можна віднести T5, адже зміна швидкості за допомогою двигуна відбувається, коли різниця фактичної і бажаної швидкості незначна і не потребує невідкладної корекції.

Типи завдань за послідовністю виконання

- *Послідовні завдання*. Це завдання, які виконуються одне після одного [5, с. 21]. Результат виконання попереднього завдання прямо впливає на запуск наступного. Наприклад, зчитування даних із сенсора передує їхньому аналізу. У динамічних системах така послідовність може мати умовний характер: наступне завдання з'являється тільки в разі виконання певних умов або результатів (що не враховується в класичних моделях). Наприклад, в алгоритмі адаптивного круїз-контролю завдання T1 (зчитування потрібної швидкості) і T2 (визначення поточної швидкості) є послідовними. Аналогічно, завдання T8 (обчислення прискорення поточного авто) і T9 (обчислення прискорення авто попереду) також виконуються послідовно.
- *Паралельні завдання*. Це завдання, які можуть виконуватися одночасно (на різних ядрах або в межах одного ядра з перемиканням контексту) [5, с. 90]. Між такими завданнями можуть існувати ресурсні залежності або залежності до даних. У реальному середовищі час їхнього запуску може змінюватися через появу високопріоритетних спорадичних подій. Наприклад, завдання T1 (зчитування потрібної швидкості) та T7 (визначення відстані до авто попереду) в алгоритмі адаптивного круїз-контролю є взаємонезалежними та виконуються паралельно.

Попередні моделі завдань у системах реального часу не враховують умовне динамічне породження завдань, яке виникає внаслідок обробки результатів попередніх завдань, через залежності за даними або ресурсами, а також через взаємодію з асинхронними або ізохронними подіями. Іншими словами, у зв'язку з тим, що деякі завдання мають залежності від інших завдань або ресурсів, існують також завдання, які виконуються не завжди, а отже, є деяка умова їхньої активації та додавання у чергу завдань.

Залежності між завданнями

Завдання можуть мати залежності, які визначають порядок, у якому завдання можуть починатися або завершуватися відносно одне одного.

У системах реального часу завдання можуть не лише з'являтись динамічно, але й бути взаємозалежними за умовами запуску або завершення. Такі залежності відіграють ключову роль у визначенні

часових характеристик, адже вони визначають порядок, у якому завдання можуть починатися або завершуватися відносно одне одного, що у свою чергу впливає на розклад, ресурсоємність і можливість гарантувати дотримання кінцевих термінів. Основні типи залежностей між завданнями формалізуються у вигляді логічних зв'язків, які визначають правила активації одного завдання залежно від іншого.

Типи залежностей між завданнями (рисунок):

1. Завершення до початку (ЗП). Завдання Б може початися тільки після завершення завдання А. Такий тип використовується в послідовному виконанні завдань, коли результат А є передумовою для запуску Б. Наприклад, обробка результатів можлива лише після збору даних.

2. Початок до початку (ПП). Завдання Б може початися лише після початку завдання А. Наприклад, старт аналізу даних може початися одночасно зі збором даних або з деякою затримкою. Така залежність характерна для систем з паралельною обробкою або для завдань, які виконуються конвеєрно (А збирає, Б уже попередньо аналізує).

3. Завершення до завершення (ЗЗ). Завдання Б може завершитись тільки після завершення завдання А. Наприклад, запис результатів у лог дозволено лише після завершення обробки всіх підзадач. У СРЧ використовується, коли два завдання мають бути завершені приблизно одночасно (наприклад, відео і аудіо потік мають синхронно завершитись).

4. Початок до завершення (ПЗ). Завдання Б може завершитися тільки після початку завдання А. Наприклад, старе завдання не може завершитись, поки не почнеться нове (черговий процес змінює попередній). Це доволі рідкісний випадок, але може зустрічатись у системах з чергуванням ролей або чергуванням обов'язків, де нове завдання має взяти контроль перед завершенням попереднього.

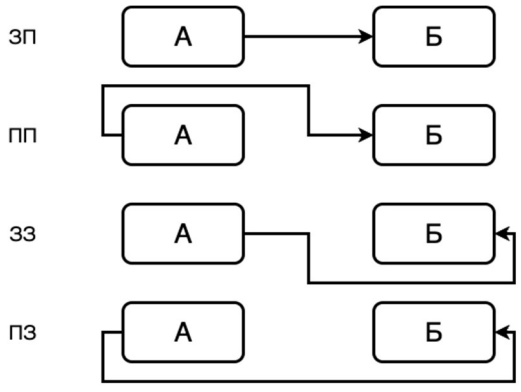

Рисунок – Типи залежностей між завданнями

Окрім вищенаведених базових типів, у реальних системах завдання можуть бути умовно активованими, тобто запуск завдання В залежить не лише від часової залежності від завдання А, а й від:

– результату виконання завдання А, наприклад, якщо завдання А завершилось з кодом 0 – запустити завдання В, інакше – завдання С.

– зовнішніх подій або тригерів, наприклад, завдання D запускається тільки при надходженні зовнішнього сигналу (спорадичні завдання);

– поточного стану системи чи наявності ресурсу, наприклад, завдання не активується, якщо ресурс зайнятий.

Тобто, у реальних умовах завдання можуть з'являтися не завжди, точніше кажучи мати умову активації, або бути результатом іншого завдання (умовне породження), а також бути залежними за результатом попередніх (*branching logic*).

Це вимагає розширення моделі залежностей, наприклад, у вигляді графа завдань з логічними умовами на ребрах. Такі підходи дозволяють точніше моделювати поведінку систем з динамічним породженням завдань і умовною логікою планування.

Визначення часу роботи завдань системи

У рамках даного дослідження продовжимо аналізувати алгоритм роботи адаптивного круїз-контролю [1, с. 152]. Круїз-контроль – це система, що підтримує сталу швидкість автомобіля без участі водія. Наприклад, на спусках система може гальмувати, а на підйомах може автоматично додавати обертів двигуна, щоб швидкість авто залишалася сталою. Адаптивний круїз-контроль виконує всі ті ж функції, що і звичайний, а також слідкує за відстанню до автомобіля, що рухається попереду.

Алгоритм роботи системи адаптивного круїз-контролю може бути представлений за допомогою таких завдань:

– Т1 – зчитування потрібної швидкості;

– Т2 – визначення поточної швидкості;

– Т3 – обчислення зміни швидкості, яка потрібна для досягнення заданих параметрів;

– Т4 – збільшення обертів двигуна;

– Т5 – зменшення обертів двигуна для незначного зменшення швидкості;

– Т6 – гальмування для швидкого зменшення швидкості;

– Т7 – визначення відстані до авто попереду;

– Т8 – обчислення прискорення поточного авто;

– Т9 – обчислення прискорення авто попереду.

Для визначення часу виконання завдань будь-якої СРЧ найбільш точним є вимірювання реальних часових характеристик усіх компонентів системи, тобто сенсорів, контролерів, актуаторів тощо. Цей спосіб надасть найточніші результати під час моделювання, адже буде безпосередньо корелювати

з наявними у системі апаратними компонентами. Проте, даний спосіб вимагає доступу до характеристик компонентів, що безпосередньо будуть використовуватися у системі, які зазвичай тримають у таємниці та не розповсюджують. Якщо часові характеристики компонентів недоступні, тоді необхідно вручну виміряти всі компоненти системи, що, очевидно, вимагає, щоб усі апаратні компоненти були наявні для проведення вимірювань. У результаті, це найбільш точний, але також і найбільш ресурсозатратний спосіб, який потребує значних коштів на придбання компонентів, їхнє підключення, налаштування та проведення вимірювання.

Проте, враховуючи, що в даній роботі метою є не вимірювання справжніх часових характеристик конкретної системи, а саме формалізація методу визначення часових характеристик завдань систем реального часу, було прийнято рішення використовувати дані, що сформовані під час останніх досліджень аналогів до компонентів системи адаптивного круїз-контролю.

Сенсори

LIDAR та радарні системи безперервно сканують навколишнє середовище для визначення транспортних засобів, що рухаються попереду. Радари, що призначені для використання на дорогах загального користування, можуть сканувати і обробляти дані зі швидкістю до 100 мс [2, с. 4].

Сенсори швидкості обертання коліс (WSS) фізично розташовані біля кожного колеса автомобіля та з'єднані з електронним блоком керування (ECU). Дані, зібрані з WSS, використовуються для обчислення швидкості руху автомобіля та для різних функцій гальмівної системи, таких як ABS (антиблокувальна система), TCS (система контролю тяги) та ESC (система електронної стабілізації). Крім того, велика кількість інших систем автомобіля також використовує швидкість транспортного засобу як вхідний параметр для виконання своїх функцій. Ці системи зазвичай функціонують на високих частотах, і дані з WSS мають надходити щонайменше кожні 10 мс (тобто з частотою 100 Гц). Наприклад, тривимірна система інерційної навігації (Dead Reckoning, DR), заснована на даних бортових сенсорів транспортного засобу, обчислює положення, швидкість у навігаційній системі координат і орієнтацію з інтервалом 0,02 секунди [3, с. 2].

Датчики відстані забезпечують вимірювання проміжку до об’єктів, з середньою затримкою в межах 20–50 мс [4, с. 3]. У роботі 2021 року, присвяченій SPAD-сенсору глибини з прямим вимірюванням часу прольоту (direct Time-of-Flight, dToF) для автомобільних LiDAR-систем, повідомляється, що розроблений LiDAR може виконувати вимірювання з роздільною здатністю 168x63 пікселі при частоті 20 кадрів на секунду, що відповідає тривалості одного сканування приблизно 50 мс [5, с. 1]. Даної частоти оновлення достатньо для коректного функціонування СРЧ АКК, зокрема – для завдань запобігання зіткненням та адаптивного контролю швидкості.

Блоки керування

Блок керування АКК є одним з ключових компонентів, що беруть участь у обробці інформації та прийнятті рішення. Після збору даних з наявних сенсорів блок керування АКК розраховує оптимальну траєкторію, прискорення та швидкість і передає вказівки іншим контролерам, таким як контролери двигуна, гальм тощо. Через обмеження, пов’язані з комерційною таємницею, реальна модель ACC-контролера невідома. Хоча джерела не вказують ізольований "час роботи" лише для ACC-контролера, вони визначають, що його діяльність є частиною критичного часового циклу, де загальна кінцева затримка обробки повідомлень системи ACC має бути менше 100 мілісекунд [6, с. 75420], щоб забезпечити безпечну та ефективну роботу в реальному часі. Тому, можна припустити, що контролер адаптивного круїз-контролю виконує алгоритми керування приблизно за 10–50 мс.

Контролер дросельної заслінки/двигуна. Експериментальні результати показують, що контролер дросельної заслінки регулює положення дроселя в межах 500 мс [7, с. 16].

Контролер гальмівної системи. Контролер гальм забезпечує активацію гальмівного механізму, що є критичним для запобігання зіткненням у системах активної безпеки. Типові електромеханічні системи керування гальмами досягають часу реакції в межах 300 мс для створення тиску або зусилля затискання [8, с. 24], що відповідає вимогам до часових характеристик систем активної безпеки в автомобілях.

Актуатори

Автомобільні актуатори дросельної заслінки. Вони керують подачею повітря в двигун, що безпосередньо впливає на потужність і швидкість автомобіля. Зазвичай демонструють час реакції в межах 300 мс, а завдяки сучасним схемам керування й оптимізації апаратної частини досягаються значення на рівні 130 мс [9, с. 1].

Актуатор гальмівної системи забезпечує фізичне спрацювання гальм протягом 200 мс, що є критично важливим для завдань екстреного гальмування та уникнення зіткнень у системах активної безпеки [10, с. 1].

Підсумок

На основі вищезазначених методологічних і концептуальних засад та для забезпечення системності й прозорості процесу, розроблено таблицю.

Таблиця – ***Опис задач адаптивного круїз контролю***

Задача	Опис	Відповідний модуль	Період виконання	Час реакції
T1	Зчитування команди водія щодо бажаної швидкості	Інтерфейс керування / користувач	Подія / на вимогу	5 мс
T2	Визначення поточної швидкості	Сенсор швидкості	500 мс	20 мс
T3	Розрахунок зміни швидкості	Блок керування ACC	Подія / на вимогу	15 мс
T4	Збільшення обертів двигуна	Контролер дросельної заслінки	Подія / на вимогу	130 мс
T5	Зменшення обертів двигуна	Контролер дросельної заслінки	Подія / на вимогу	130 мс
T6	Активація гальм для інтенсивного зниження швидкості	Контролер гальм / Актуатор гальм	Подія / на вимогу	200 мс
T7	Визначення відстані до авто попереду	LIDAR / Радар / Датчик відстані	300 мс	50 мс
T8	Обчислення прискорення поточного авто	Сенсор швидкості + ACC блок	Подія / на вимогу	35 мс
T9	Обчислення прискорення авто попереду	LIDAR + ACC блок	Подія / на вимогу	65 мс

Визначення часу роботи завдань системи

Отже, знаючи природу появи завдань, їхні залежності та часові характеристики, можна обчислити найгірший час виконання (WCE) всіх можливих варіантів подій, що можуть відбутися під час роботи системи, а також загальний час виконання повного циклу алгоритму системи.

Очевидно, час виконання алгоритму відрізнятиметься залежно від умов виконання. Для того щоб дати оцінку СРЧ, потрібно проаналізувати всі можливі варіанти виконання алгоритму роботи СРЧ та дослідити відповідність дотримання заданим часовим характеристикам [2, с. 49].

Наприклад, розглянемо спрацювання таймера визначення відстані до авто попереду як початкову подію. Тоді, за умови, що автомобіль попереду знаходиться далеко або його зовсім немає, потрібно переконатися, що прискорення даного автомобіля і того, що їде попереду, є безпечними, отже, буде виконано наступний ланцюжок завдань: T7, T8, T9. Додавши час виконання кожної з підзавдань, було визначено загальний час виконання алгоритму за вищезгаданих умов, що складає 150 мс. При цьому, якщо прискорення будь-якого одного з автомобілів буде небезпечним, тобто таким, що може призвести до зіткнення, буде також виконано завдання T3 для визначення потрібної зміни швидкості поточного автомобіля. Далі, залежно від того, наскільки стрімко зменшується відстань між автомобілями, може бути задіяно або зменшення обертів двигуна (T5), або гальмування (T6). У результаті, якщо було задіяно гальмування, то буде виконано наступний ланцюжок завдань: T7, T8, T9, T3, T6, що складає 365 мс. Інакше, якщо було зменшено оберти двигуна, то буде виконано наступний ланцюжок завдань: T7, T8, T9, T3, T5, що складає 295 мс. Якщо ж відстань до авто попереду безпечна, прискорення автомобілів у безпечних межах, але швидкість поточного авто менша, ніж задана, тоді треба обчислити потрібну зміну швидкості та збільшити оберти двигуна, тобто виконати наступну послідовність завдань: T7, T8, T9, T3, T4, що також складає 295 мс.

З іншого боку, коли водій змінює бажану швидкість, і вона відрізняється від поточної, то подальші кроки залежать від того, чи бажана швидкість більша, чи менша за поточну. Якщо бажана швидкість більша, це означає, що можна збільшити оберти двигуна, а отже, буде виконано наступну послідовність завдань: T1, T2, T3, T4, що складає 170 мс. Натомість, у випадку, коли бажана швидкість менша за поточну, це означає, що потрібно зменшити швидкість автомобіля. Якщо бажана швидкість сильно менша за поточну, для призупинки автомобіля будуть задіяні гальма, а отже, буде виконано наступну послідовність завдань: T1, T2, T3, T6, що складає 240 мс. Інакше, якщо бажана швидкість не сильно відрізняється від поточної, можна використати гальмування двигуном, тобто зменшити оберти двигуна, тому буде виконано наступну послідовність завдань: T1, T2, T3, T5, що складає 170 мс.

Для розробки моделі системи потрібно визначити час появи завдань, час їхнього виконання, а також їхній кінцевий час виконання. При цьому, варто враховувати, що сучасні системи реального часу зазвичай складаються з багатьох сенсорів, актуаторів і контролерів. Роботу цих компонентів контролює центральний процесор, який обробляє події навколишнього середовища, що надходять від різних компонентів, і відносно отриманих даних здійснює контроль та підтримку роботи системи. Надходження асинхронних подій може породити ряд взаємозалежних завдань, а також деякий набір наперед непередбачених завдань, які виникають залежно від стану системи і навколишнього середовища. Після появи нових завдань у черзі планувальник завдань повинен заново проаналізувати весь список завдань і, за потреби, перерозподілити процесорний час з урахуванням нових завдань.

Перед початком розробки моделі потрібно сформулювати часові та логічні вимоги до системи, а також початкові характеристики завдань. Для цього потрібно розбити алгоритм роботи СРЧ на завдання. Після того, як відомий список завдань, пропонується створити блок-схему роботи алгоритму системи, яка дозволить наочно дослідити залежності між завданнями та ресурсами системи. Блок-схема також допомагає визначити послідовність і комбінації завдань, які будуть виникати за певних подій. Також потрібно дослідити час виконання завдань, враховуючи апаратне забезпечення, яке буде використовуватись у фінальній системі. Далі потрібно дослідити природу появи завдань, тобто те, за яких умов завдання виникатимуть, з якою періодичністю, а також те, які завдання можуть виконуватись паралельно одне до одного. Після цього, можна визначити час появи завдань та умови їхньої появи [1, с. 151]. Дослідивши вищевказані характеристики, потрібно встановити крайні терміни виконання кожного з завдань відповідно до вимог системи. Крайні терміни можуть бути жорсткими або м'якими, залежно від вимог до поведінки системи, що впливатиме на те, як планувальник завдань буде розподіляти процесорний час між завданнями.

Після визначення початкових характеристик завдань системи, таких як час появи, час виконання і кінцевий час виконання, пропонується створити програмну модель системи. За її допомогою можна дослідити, чи система відповідає заданим їй часовим вимогам. Нещодавні дослідження [3, с. 83] доводять, що моделюючи виконання завдань з використанням сіток Петрі, можна гарантувати отримання похідних часових характеристик завдань при обранні конкретних типів процесора і планувальника. Отже, пропонується змоделювати виконання завдань системи, використовуючи різні алгоритми роботи планувальників, та дослідити, який з планувальників задовольняє часовим характеристикам системи.

У результаті буде отримано всі часові характеристики завдань залежно від обраного процесора і типу планувальника, на основі яких можна буде визначити, який з алгоритмів роботи планувальника задовольняє часові вимоги системи. Якщо декілька планувальників досягли поставлених цілей, тоді, залежно від типу системи, можна обрати той, який має найменший час відгуку, або, навпаки, той, який дозволяє використати економніший процесор для зменшення загальної вартості системи, але при цьому гарантувати справність системи у будь-яких можливих умовах використання. З іншого боку, якщо жоден з алгоритмів роботи планувальника не досягнув очікуваних часових характеристик, доцільно використати потужніший процесор або швидші апаратні компоненти, які дозволять зменшити час виконання завдань і досягти очікуваних часових вимог.

Нарешті, після аналізу моделі та переконавшись, що система задовольняє всі вимоги, розпочинаємо розробку безпосередньо кінцевої системи. Хоча розробка і аналіз моделі може здаватися додатковою витратою часу і ресурсів, моделювання є значно простішою і менш ресурсозатратною задачею, яка дозволить виявити потенційні проблеми і недоліки системи, а також переконатися, що за підібраних апаратних і програмних компонентів система буде працювати справно та передбачувано.

Висновок

Система адаптивного круїз-контролю вважається жорсткою СРЧ, де будь-яка затримка неприпустима, адже може призвести до небезпечної ситуації. Саме тому для такого типу систем необхідним є точне і детальне визначення відповідності системи заданим часовим вимогам. Ключовим завданням є визначення часових характеристик завдань, які надалі використовуються для моделювання роботи системи. Для цього необхідно знати такі параметри завдань системи реального часу, як час появи завдання, час його виконання, а також кінцевий час виконання завдання.

У роботі обґрунтовано та представлено підхід до детермінації часових характеристик виконання завдань, що адаптується до динамічної природи їхньої генерації в процесі функціонування системи.

Запропонований метод визначення вказаних часових показників базується на декомпозиції загального алгоритму функціонування системи на дискретні, чітко ідентифіковані операційні завдання. Невід'ємною складовою методу є формалізація їхньої послідовності та взаємозалежностей. Така структуризація дозволяє встановити кореляцію між фактом завершення попереднього завдання та умовою готовності до виконання наступних.

Для візуалізації цих логічних зв'язків та операційної послідовності передбачається розробка структурної блок-схеми алгоритму. Ця схема має конгруентно відображати ієрархію завдань та їхні критичні залежності, забезпечуючи транспарентність всього робочого процесу.

Ключовою особливістю методу, що враховує динамічну природу появи завдань, є те, що час появи нового завдання може бути безпосередньо визначений часом завершення попереднього, залежного від нього завдання. Наприклад, ACC-контролер постійно оновлюється даними від радарів та сенсорів, і ці оновлення мають надходити безперервно та впорядковано, що запускає відповідні завдання обробки та прийняття рішень.

Цей підхід уможливлює адекватний облік того, як зовнішні події в операційному середовищі динамічно ініціюють відповідні завдання в системі.

Визначені таким чином часові характеристики стають фундаментальною основою для подальшого моделювання роботи системи. Це є критично важливим етапом для підтвердження та вибору алгоритму планувальника завдань, який забезпечить відповідність встановленим системним вимогам.

Впровадження цього механізму дозволяє оптимізувати розподіл процесорного часу та гарантувати, що система АКК (Система Адаптивного Круїз-Контролю) задовольняє своїм жорстким вимогам щодо часу затримки. Це, у свою чергу, забезпечує ефективне та безпечне функціонування системи в цілому.

Список літератури

1. Юрович І., Зайцев В. Метод визначення часових характеристик систем реального часу. *Управління розвитком складних систем*. 2024. Вип. 59. С. 148–154. URL: https://doi.org/10.32347/2412-9933.2024.59.148-154.
2. Зайцев В. Г., Цибаєв Є. І. Оцінка часових характеристик у комп'ютерних системах реального часу з використанням сіток Петрі. *Управління розвитком складних систем*. 2023. № 54. С. 48–62.
3. Зайцев В. Г., Цибаєв Є. І. Модель оцінки часових характеристик у комп'ютерних системах реального часу з використанням сіток Петрі. *Управління розвитком складних систем*. 2019. № 40. С. 76–86. DOI: URL: dx.doi.org\10.6084/m9.figshare.11969013.
4. Зайцев В. Г., Цибаєв Є. І. Оцінка часових характеристик задач в багатопроцесорних системах реального часу з використанням сіток Петрі. *Управління розвитком складних систем*. 2020. № 42. С. 43–50.
5. Зайцев В. Г., Цибаєв Є. І. *Комп'ютерні системи реального часу*: навч. посіб. Київ, 2019.
6. Ciyun L. et al. Vehicle Detection and Tracking with Roadside LiDAR Using Improved ResNet18 and the Hungarian Algorithm. *Sensors*. 2023. Vol. 23, № 15. Art. 6757. URL: https://doi.org/10.3390/s23156757.
7. Joong-hee H., Chi-ho P. Performance evaluation on GNSS, wheel speed sensor, yaw rate sensor, and gravity sensor integrated positioning algorithm for automotive navigation system. *E3S Web of Conferences*. Republic of Korea, 2019. Vol. 84. Art. 01004. URL: https://doi.org/10.1051/e3sconf/20198401004.
8. Prasetyono A. P. et al. Multiple Sensing Method Using Moving Average Filter for Automotive Ultrasonic Sensor. *Journal of Physics: Conference Series*. 2020. Vol. 1569. Art. 032001. URL: https://doi.org/10.1088/1742-6596/1569/3/032001.
9. Kumagai O. et al. 7.3 A 189×600 Back-Illuminated Stacked SPAD Direct Time-of-Flight Depth Sensor for Automotive LiDAR Systems. *2021 IEEE International Solid-State Circuits Conference (ISSCC)*. San Francisco, USA, 2021. P. 110–112. URL: https://doi.org/10.1109/ISSCC42661.2021.9366114.
10. Almadani B., Alshammari N., Al-Roubaiey A. Adaptive Cruise Control Based on Real-Time DDS Middleware. *IEEE Access*. 2023. Vol. 11. P. 75407–75423. URL: https://doi.org/10.1109/ACCESS.2023.3296317.
11. Liu Y., Li F., Sun B. Self-Tuning Backstepping Control with Kalman-like Filter for High-Precision Control of Automotive Electronic Throttle. *Electronics*. 2023. Vol. 12, № 10. Art. 2262. URL: https://doi.org/10.3390/electronics12102262.
12. Wu T., Li J., Qin X. Braking performance oriented multi–objective optimal design of electro–mechanical brake parameters. *PLoS ONE*. 2021. Vol. 16, № 5. Art. e0251714. URL: https://doi.org/10.1371/journal.pone.0251714.
13. Kreß J. et al. Low-Cost Throttle-By-Wire-System Architecture for Two-Wheeler Vehicles. *SAE Technical Paper Series*. SAE International, 2023. URL: https://doi.org/10.4271/2023-01-0498.
14. Salgado V., Gomes D., Andrade Lima C. Modeling and Simulation of an Electromagnetic Brake-by-Wire System for Formula SAE Vehicles. *SAE Technical Paper Series*. SAE International, 2024. URL: https://doi.org/10.4271/2024-01-0495

Стаття надійшла до редколегії 06.09.2025

Yurovych Ivan
Postgraduate of the Department of System Programming and Specialized Computer Systems, Kyiv Polytechnic Institute, *https://orcid.org/0009-0005-9575-065X*
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv
Zaitsev Vladimir
DSc (Eng.), Professor, Professor of the Department of System Programming and Specialized Computer Systems, Kyiv Polytechnic Institute, *https://orcid.org/0009-0009-9917-5364*
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv

DETERMINING TASK EXECUTION TIME IN REAL-TIME SYSTEMS

Abstract. *The challenge of operating real-time computer systems-particularly those with high safety-criticality, such as adaptive cruise control-lies not only in their logical correctness but also in the timely delivery of results, where any delay can have critical consequences. Existing models often oversimplify the system being simulated and fail to consider the dynamic nature of task arrivals, which is an inherent feature of modern real-time systems. This paper proposes a novel method for determining task execution time in real-time systems that accounts for the dynamic appearance of tasks during system operation. The method classifies various types of events (asynchronous, synchronous, isochronous) and task types (periodic, sporadic, aperiodic), while also addressing different types of task dependencies, including conditional activations. For realistic estimation of task execution times, data from research on analogous ACC system components (sensors, controllers, actuators) are utilized. The resulting timing characteristics serve as a foundation for further system modeling, enabling the selection of a task scheduling algorithm that meets the system's timing requirements. This ensures compliance with stringent latency constraints, guaranteeing reliable and safe operation, and allows the early detection of potential issues during the system development process.*

Keywords: real-time operating system; adaptive cruise control; task execution time; sensors; actuators; modeling

References

1. Yurovych, I., & Zaitsev, V. (2024). Method of determining timing parameters of real time systems. *Management of Development of Complex Systems*, *59*, 148–154. DOI: URL: https://doi.org/10.32347/2412-9933.2024.59.148-154.

2. Zaitsev, V., & Tsybaev, E. (2023). Estimation of timing characteristics in real-time computer systems using petri nets. *Management of Development of Complex Systems*, *40*, 48–62.

3. Zaitsev, V., & Tsybaev, E. (2019). A model for estimating time characteristics in real-time computer systems using Petri nets. *Management of Development of Complex Systems*, *40*, 76–86.

4. Zaitsev, V., & Tsybaev, E. (2020). Evaluation of time characteristics of problems in multiprocessor real-time systems using petri networks. *Management of Development of Complex Systems*, *42*, 43–50.

5. Zaitsev, V. G., & Tsybaev, E. I. (2019). *Real-time computer systems: A textbook*. Kyiv, Ukraine: National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

6. Ciyun, L., Liu, Y., Zhang, X., Li, X., Wu, X., & Liu, X. (2023). Vehicle Detection and Tracking with Roadside LiDAR Using Improved ResNet18 and the Hungarian Algorithm. *Sensors*, *23*(15), 6757. DOI: URL: https://doi.org/10.3390/s23156757.

7. Joong-hee, H., & Chi-ho, P. (2019, May 8). Performance evaluation on GNSS, wheel speed sensor, yaw rate sensor, and gravity sensor integrated positioning algorithm for automotive navigation system. In *E3S Web of Conferences* (Vol. 84, p. 01004). Republic of Korea: EDP Sciences. DOI: URL: https://doi.org/10.1051/e3sconf/20198401004.

8. Prasetyono, A. P., Suwito, A., & Budi, S. (2020). Multiple Sensing Method Using Moving Average Filter for Automotive Ultrasonic Sensor. *Journal of Physics: Conference Series*, *1569*(3), 032001. DOI: URL: https://doi.org/10.1088/1742-6596/1569/3/032001.

9. Kumagai, O., Takamatsu, S., Tanaka, S., Tanaka, K., Oikawa, H., Kawada, Y., Suzuki, A., Arai, H., & Sakata, S. (2021). 7.3 A 189×600 Back-Illuminated Stacked SPAD Direct Time-of-Flight Depth Sensor for Automotive LiDAR Systems. In *2021 IEEE International Solid-State Circuits Conference (ISSCC)* (pp. 110–112). San Francisco, USA: IEEE. DOI: URL: https://doi.org/10.1109/ISSCC42661.2021.9366114.

10. Almadani, B., Alshammari, N., & Al-Roubaiey, A. (2023). Adaptive Cruise Control Based on Real-Time DDS Middleware. *IEEE Access*, *11*, 75407–75423. DOI: URL: https://doi.org/10.1109/ACCESS.2023.3296317.

11. Liu, Y., Li, F., & Sun, B. (2023). Self-Tuning Backstepping Control with Kalman-like Filter for High-Precision Control of Automotive Electronic Throttle. *Electronics*, *12*(10), 2262. DOI: URL: https://doi.org/10.3390/electronics12102262.

12. Wu, T., Li, J., & Qin, X. (2021). Braking performance oriented multi–objective optimal design of electro–mechanical brake parameters. *PLoS ONE*, *16*(5), e0251714. DOI: URL: https://doi.org/10.1371/journal.pone.0251714.

13. Kreß, J., Beirer, T., & Gruler, T. (2023). Low-Cost Throttle-By-Wire-System Architecture for Two-Wheeler Vehicles. *SAE Technical Paper Series*. SAE International. DOI: URL: https://doi.org/10.4271/2023-01-0498.

14. Salgado, V., Gomes, D., & Andrade Lima, C. (2024). Modeling and Simulation of an Electromagnetic Brake-by-Wire System for Formula SAE Vehicles. *SAE Technical Paper Series*. SAE International. DOI: URL: https://doi.org/10.4271/2024-01-0495.

Посилання на публікацію

APA Yurovych, I., & Zaitsev, V. (2025). Determining task execution time in real-time systems. *Management of Development of Complex Systems*, 63, 212–222, dx.doi.org\10.32347/2412-9933.2025.63.212-222.

ДСТУ Юрович І. В., Зайцев В. Г. Метод визначення часу виконання програм у системах реального часу. *Управління розвитком складних систем.* Київ, 2025. № 63. С. 212 – 222, dx.doi.org\10.32347/2412-9933.2025.63.212-222.