ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Науковий вісник НЛТУ України
Scientific Bulletin of UNFU
ISSN 1994-7836 (print)
ISSN 2519-2477 (online)
https://doi.org/10.36930/40330506
Article received 20.09.2023
Article accepted 26.10.2023
UDC 004.[05+04]
Correspondence author: M. Ya. Seneta mariana.y.seneta@lpnu.ua

І. Г. Цмоць, Ю. В. Опотяк, М. Я. Сенета, Ю. Ю. Олійник, Н. Б. Газда
Національний університет "Львівська політехніка", м. Львів, Україна

СТРУКТУРА ТА ОСОБЛИВОСТІ ОСНОВНИХ ЕТАПІВ ТЕСТУВАННЯ СПЕЦІАЛІЗОВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МОБІЛЬНОЇ РОБОТОТЕХНІЧНОЇ ПЛАТФОРМИ

Розроблено методики тестування програмно-апаратного комплексу мобільних робототехнічних платформ. Розглянуто архітектуру програмно-апаратного комплексу засобів і досліджено взаємодію компонент спеціалізованого програмного забезпечення мобільних робототехнічних платформ. Для забезпечення управління програмно-апаратним комплексом використано мікрокомп'ютерну платформу на базі SoC під керуванням ОС Linux. Проведено симуляційні тести для імітації сигналів давачів і перевірки здатності системи обробляти і виводити дані. Досліджено систему тестування продуктивності спеціалізованого програмного забезпечення мобільної робототехнічної платформи для оцінювання швидкодії, швидкості реагування та стабільності системи. Створено комплексний план тестування програмно-апаратного комплексу із дотриманням структурованого підходу. З використанням спеціалізованого обладнання (генератори вхідних послідовностей сигналів управління, генератори вхідних даних, таблиці еталонних результатів, засоби порівняння) і технологічних програмних засобів проведено спільне тестування програмних та апаратних засобів у режимі реального часу. Показано, що основними етапами тестування спеціалізованого програмного забезпечення мобільної робототехнічної платформи є: аналіз вимог до спеціалізованого програмного забезпечення; розроблення плану тестування; функціональне тестування; тестування продуктивності; перевірка на вразливість; сумісність; тестування інтерфейсу користувача; тестування на помилки; тестування реальних сценаріїв; тестування в динамічних умовах. Проведено тестування спеціалізованого програмного забезпечення з використанням модуля керування ESP32. Для організованого тестового каналу передачі даних виконано інтеграційне тестування спеціалізованого програмного забезпечення для блоку прийомо-передавача системи керування мобільною робототехнічною платформою. Проаналізовано результати тестування та розраховано середні швидкості передачі даних. Виконано тестування спеціалізованого програмного забезпечення шифрування/дешифрування на базі двох платформ – комп'ютера центру керування (ноутбук) та мікрокомп'ютера на базі SoC Allwinner H3+.

Ключові слова: мобільна робототехнічна платформа; тестування; якість програмного забезпечення; нечітке управління рухом.

Вступ / Introduction

Тестування програмного забезпечення (ПЗ) є одним із найважливіших процесів у його розробленні, оскільки перевіряє відповідність системи вимогам користувача та специфікаціям [7, 13]. Неправильний вибір, а також зміна інструментів тестування у процесі розроблення впливають на якість програмного продукту і, як наслідок, на результат дослідження загалом [2, 11, 20]. Тестування ПЗ є і одним із найбільш трудомістких процесів у розробленні ПЗ [9, 18]. Тому розробники і дослідники в цій галузі щораз більше зацікавлені у спробах його оптимізації, щоб досягти зменшення витрат [4, 19].

Створення комплексного плану тестування має вирішальне значення для будь-якого проєкту розроблення програмного чи апаратного забезпечення, не є винятком і тестування програмно-апаратного комплексу мобільних робототехнічних платформ (МРТП) [5, 8]. Комплексний план тестування МРТП заощаджує час і затрати в довготривалій перспективі внаслідок виявлення

Інформація про авторів:
Цмоць Іван Григорович, д-р техн. наук, професор, кафедра автоматизованих систем управління.
Email: ivan.tsmots@gmail.com; https://orcid.org/0000-0002-4033-8618
Опотяк Юрій Володимирович, канд. техн. наук, доцент, кафедра автоматизованих систем управління.
Email: yurii.v.opotiak@lpnu.ua; https://orcid.org/0000-0001-9889-4177
Сенета Мар'яна Ярославівна, канд. фіз.-мат. наук, доцент, кафедра автоматизованих систем управління.
Email: ariana.y.seneta@lpnu.ua; https://orcid.org/0000-0003-1249-0935
Олійник Юрій Юрійович, аспірант, кафедра автоматизованих систем управління. **Email:** yurii.oliinyk.mknm.2020@lpnu.ua
Газда Назар Богданович, аспірант, кафедра автоматизованих систем управління. **Email:** nazar.hazda@gmail.com

Цитування за ДСТУ: Цмоць І. Г., Опотяк Ю. В., Сенета М. Я., Олійник Ю. Ю., Газда Н. Б. Структура та особливості основних етапів тестування спеціалізованого програмного забезпечення мобільної робототехнічної платформи. Науковий вісник НЛТУ України. 2023, т. 33, № 5. С. 46–53.
Citation APA: Tsmots, I. G., Opotyak, Yu. V., Seneta, M. Ya., Oliynyk, Yu. Yu., & Gazda, N. B. (2023). Structure and features of the main stages of testing the specialized software of the mobile robotic platform. *Scientific Bulletin of UNFU*, 33(5), 46–53. https://doi.org/10.36930/40330506

та вирішення проблем на ранніх стадіях процесу розроблення, що дає змогу зменшити витрати на виправлення помилок до або після виходу продукту на ринок. Створення комплексного плану тестування є критичним кроком, внаслідок чого розроблювана система проходить ретельне тестування та відповідає бажаним стандартам якості, що в кінцевому підсумку забезпечує створення більш надійного та успішного продукту.

Об'єкт дослідження – процеси тестування спеціалізованого програмного забезпечення мобільної робототехнічної платформи.

Предмет дослідження – методи і засоби тестування спеціалізованого програмного забезпечення мобільної робототехнічної платформи.

Мета роботи – розробити структуру засобів і сценаріїв тестування спеціалізованого програмного забезпечення для мобільної робототехнічної платформи, щоб визначити стан якості апаратно-програмного комплексу засобів, які реалізують різні аспекти функціональності платформи.

Для досягнення зазначеної мети визначено такі *основні завдання дослідження*:

- розробити структуру спеціалізованого програмного забезпечення МРТП;
- застосувати комплексне тестування для перевірки функціональності кожного мікроконтролерного засобу;
- розробити автоматизовані модульні тести для перевірки функціональності мікроконтролерних засобів;
- розробити інтеграційні тести для забезпечення правильної спільної роботи різних компонент і підсистем;
- провести симуляційні тести для імітації сигналів давачів і перевірки здатності системи обробляти і виводити дані;
- провести тестування продуктивності та швидкодії програмно-апаратного комплексу.

Аналіз останніх досліджень та публікацій. Завдяки різноманітним перевагам мобільних робототехнічних платформ, сьогодні інтенсивно виконують дослідження зі створення для них систем автоматичного керування [1, 5, 6, 8, 10], а отже, ця проблематика є актуальною. Керування роботою МРТП відбувається за допомогою апаратно-програмного комплексу засобів, які реалізують різні аспекти функціональності платформи [1, 5, 8, 10]. Для виконання покладених завдань МРТП повинна забезпечувати реалізацію основних функцій:

- навігація (ПЗ збирання та опрацювання даних із давачів за умов перешкод і неповної інформації, ПЗ попереднього оброблення зображень та розпізнавання зображень і сцен, ПЗ моделювання навколишнього середовища, ПЗ прокладання раціональних маршрутів переміщення МРТП);
- управління переміщенням платформи (ПЗ системи нечіткого управління рухом МРТП, ПЗ безпосереднього керування приводами платформи);
- віддалене керування (ПЗ підтримки функціонування засобів зв'язку (трансіверів), ПЗ шифрування та дешифрування даних з використанням нейроподібних мереж);
- забезпечення функцій корисного навантаження, ПЗ збирання та опрацювання даних із давачів корисного навантаження, ПЗ підтримки функціонування виконавчих механізмів і маніпуляторів.

Дослідження процесу моделювання мобільної робототехнічної платформи можна знайти в різних джерелах [3, 14, 15, 17]. У сучасній науковій літературі наведено описи систем управління МРТП, розроблені у середовищі моделювання з використанням програмного забезпечення [1], проте недостатньо проаналізовано саме етапи перевірки правильності функціонування таких систем. У роботі [7] здійснено порівняльний огляд функцій відкритих і комерційних інструментів тестування, які можуть допомогти користувачам вибрати відповідний інструмент тестування ПЗ відповідно до їхніх вимог.

Можна зробити висновок, що правильно розроблений план тестування МРТП визначає критерії оцінювання успішності тестування, допомагає гарантувати правильність функціонування програмного забезпечення та переконатися, що тестування проводиться структуровано та організовано. Тому у цій роботі основну увагу зосереджено на структурі та особливостях основних етапів тестування спеціалізованого програмного забезпечення мобільної робототехнічної платформи.

Результати дослідження та їх обговорення / Research results and their discussion

Структура спеціалізованого програмного забезпечення МРТП. Для забезпечення управління МРТП за допомогою апаратно-програмного комплексу засобів використовується мікрокомп'ютерна платформа на базі SoC під керуванням ОС Linux. Для тестування та відлагодження засобів МРТП застосовано керівний мікрокомп'ютер на базі SoC Allwinner H3 під управлінням Armbian – версії ОС Linux для ARM плат відлагодження.

Взаємодію компонент спеціалізованого програмного забезпечення МРТП наведено на рис. 1. Реалізацію функцій виконують окремі програмні або апаратно-програмні компоненти. Відповідно до складності виконуваних функцій зазначені компоненти створюються на основі окремих мікрокомп'ютерів у поєднанні з апаратними засобами, реалізованими на FPGA фірми Altera (Intel) або на основі мікроконтролерів типу ESP32 та ATmega328.

Компоненти взаємодіють між собою та з керівним мікрокомп'ютером за допомогою послідовного інтерфейсу або безпровідного каналу передачі даних. Така архітектура програмно-апаратного комплексу засобів забезпечує можливість індивідуального їх відлагодження окремо та у комплексі, застосовуючи, за потреби, емулятори цих компонент.

Основні етапи тестування спеціалізованого програмного забезпечення МРТП:

1. Аналіз вимог до спеціалізованого програмного забезпечення (СПЗ).
2. Розроблення плану тестування, в якому визначаються сценарії тестування, тестові кейси, очікувані результати та критерії відповідності.
3. Функціональне тестування, яке зводиться до перевірки на правильність виконання всіх функцій програмних компонент.
4. Тестування продуктивності СПЗ.
5. Оцінювання безпеки, яке зводиться до перевірки СПЗ на вразливість та можливість злому. Це особливо важливо, якщо програмне забезпечення стосується конфіденційної інформації.
6. Сумісність – перевірка функціонування СПЗ на різних платформах, операційних системах і з іншими програмами, з якими воно взаємодіятиме.
7. Тестування інтерфейсу користувача – перевірка, що графічний інтерфейс інтуїтивно зрозумілий та зручний для користувачів.
8. Тестування на помилки – пошук і виправлення помилок у СПЗ.
9. Тестування реальних сценаріїв – виконання тестування на робочих тактових частотах у реальних умовах функціонування.

10. Тестування в динамічних умовах – оцінювання взаємодії СПЗ з рухомими об'єктами або динамічною інформацією, щоб переконатися, що воно працює стабільно за таких умов.

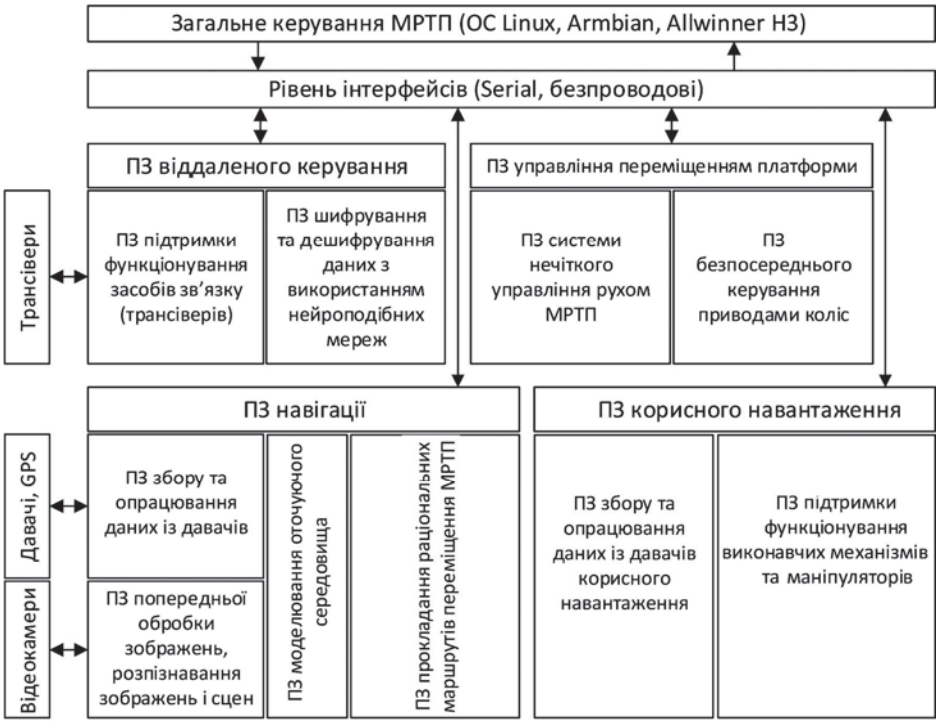

Рис. 1. Взаємодія компонент спеціалізованого ПЗ МРТП / Interaction of specialized software components of MRTP

Створення комплексного плану тестування. Щоб створити комплексний план тестування програмно-апаратного комплексу, важливо дотримуватися структурованого підходу. План тестування сформовано з урахуванням таких кроків:

- чітке окреслення обсягу тестування;
- визначення цілей тестування;
- вибір відповідних методик тестування, які будуть використані;
- визначення типів тестів, які будуть проводитися для кожного компонента системи. Приклади тестів охоплюють функціональне тестування, тестування зручності використання та тестування продуктивності;
- встановлення пріоритетів тестування та визначення, які тести будуть проведені першими. Необхідно визначити пріоритетність тестів на основі потенційного впливу окремих складових на систему та важливості для загальної функціональності системи.

Дотримуючись цих кроків і створивши комплексний план випробувань, можна переконатися, що робототехнічна система ретельно протестована та відповідає бажаним стандартам якості.

Для тестування програмно-апаратного комплексу МРТП застосовано низку методик і методологій, оскільки складність системи не дасть змоги ефективно протестувати систему повністю, не розбиваючи сам процес тестування на етапи та елементи. Зокрема, проведено модульне тестування, тестування окремих компонент, далі – процес масштабування, перевіряючи при цьому взаємодію модулів і вузлів, а також реакцію одного компонента системи на зміну стану іншого. Проведено інтеграційне тестування спеціалізованого програмного забезпечення. Нижче детальніше описано необхідні етапи для комплексного та цілісного тестування програмно-апаратного комплексу МРТП.

Розроблення модульних тестів. Щоб переконатися, що кожен мікроконтролер може правильно спілкуватися із системою Linux і що його функціональні можливості виконуються належно, потрібно розробити автоматизовані модульні тести. Це передбачає використання інфраструктури модульного тестування для створення набору тестів, які запускатимуться автоматично для перевірки функціональності мікроконтролерних засобів. Тести повинні охоплювати різні аспекти функціональності мікроконтролерних засобів, враховуючи здатність спілкуватися із системою Linux за допомогою протоколів (Serial, SPI, безпровідні), здатність отримувати та обробляти дані від давачів і пристроїв. Ці тести розроблені для виконання в автоматизованому режимі, щоб їх можна було часто запускати впродовж усього процесу розроблення, а будь-які помилки можна було швидко виявити та виправити.

Автоматизоване модульне тестування є важливою частиною процесу розроблення програмного забезпечення, оскільки дає змогу виявляти помилки на ранніх стадіях циклу. Воно гарантує, що розроблений програмний код працює належно і відповідає вимогам дизайну, що сприяє покращенню загальної якості програмного забезпечення. Макети та емулятори можуть бути корисними інструментами для тестування систем програмного забезпечення на основі мікроконтролерних засобів, що працюють на платформах ESP32, ESP8266, ATmega.

Емулятор об'єкта імітує поведінку реального об'єкта контрольованим способом, дозволяючи ізолювати та тестувати певні частини програмного забезпечення. Наприклад, можливо створити макет давача, який генерує попередньо визначені дані для імітації показань давача реального. Це може бути корисним для тестування програмного забезпечення, яке обробляє дані давачів, без використання реальних давачів, забезпечуючи однозначну інтерпретацію отримуваних результатів.

З іншого боку, емулятор забезпечує спрощену реалізацію інтерфейсу, від якого залежить ПЗ. Наприклад,

можна створити емулятор для інтерфейсу послідовного порту Linux, який відповідає на певні команди заздалегідь визначеним способом. Під час тестування комплексів програмного забезпечення на основі мікроконтролерних засобів важливо використовувати структуру модульного тестування, сумісну з цільовою платформою. Наприклад, Arduino IDE надає вбудовану структуру модульного тестування для тестування коду, що працює на платформі ATmega. Для інших платформ доступні різні фреймворки модульного тестування, такі як Unity, Ceedling або Catch2.

Використання макетів й емуляторів може бути потужною технікою для тестування систем програмного забезпечення на основі мікроконтролерних засобів. Емулюючи поведінку реальних об'єктів і їх залежностей, можна ізолювати та тестувати конкретні частини програмного забезпечення, не покладаючись на зовнішні апаратні чи програмні компоненти.

Існує кілька тестових фреймворків, які можна використовувати для перевірки коду C для мікроконтролерів, таких як ESP32, ESP8266, ARM або ATmega. Наприклад, Ceedling – інтегрована платформа тестування для коду C, яка поєднує Unity, CMock та інші інструменти тестування в єдиний пакет, що розроблений для використання з такими платформами мікроконтролерних засобів, як ARM і Atmel AVR, та забезпечує просту у використанні структуру для написання та виконання тестів. EmbUnit – платформа модульного тестування для вбудованих систем, враховуючи мікроконтролери, що підтримує низку платформ мікроконтролерів і може використовуватися для тестування коду C на таких платформах, як ARM і Atmel AVR. µCUnit – платформа для тестування коду C, розроблена спеціально для мікроконтролерних засобів, що забезпечує просту у використанні структуру для написання та виконання модульних тестів на платформах мікроконтролерів ARM, Atmel AVR і PIC. Ці фреймворки можуть допомогти писати та запускати автоматизовані тести для коду мікроконтролерних засобів, гарантуючи, що код є функціональним, надійним і працює належно.

Модульне тестування є критично важливим аспектом розроблення ПЗ, оскільки дає змогу перевіряти систему загалом, окремі компоненти програмного забезпечення або модулі окремо від решти системи. Такий підхід зменшує ймовірність дефектів або помилок у СПЗ та дає змогу виявляти та вирішувати проблеми на ранніх етапах циклу розроблення, що в кінцевому підсумку приводить до більш надійного та якіснішого програмного коду для мікроконтролерних засобів.

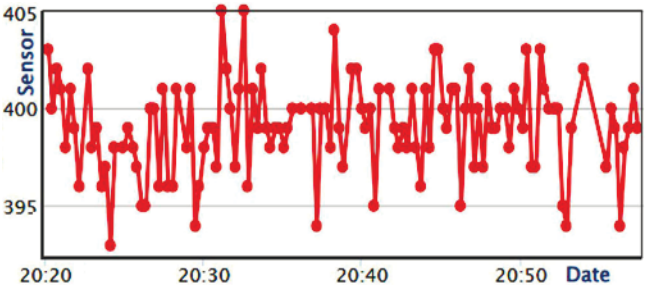

Рис. 2. Графік отриманих значень давача VL53L0X / Graph of the obtained values of the VL53L0X sensor

Тестування спеціалізованого програмного забезпечення модуля вимірювання відстані виконано за допомогою ресурсів хмарної платформи thingspeak.com, на які надсилаються тестові дані. На сайті забезпечується нагромадження даних з мікроконтролерної системи, яка тестується. Забезпечується відображення нагромаджених даних у вигляді графіка (рис. 2).

Для виконання тестування спеціалізованого ПЗ тестована мікроконтролерна система з інтегрованим давачем відстані типу VL53L0X була розміщена на відстані 400 мм від перешкоди. Під час її тестування у циклі здійснювалося зчитування та оброблення даних і їх передача на вказаний сайт. Нагромаджені дані оброблялися та аналізувалися у середовищі MatLab, яке інтегроване в обрану хмарну платформу, після чого формувалася гістограма розподілу значень отриманих даних (рис. 3).

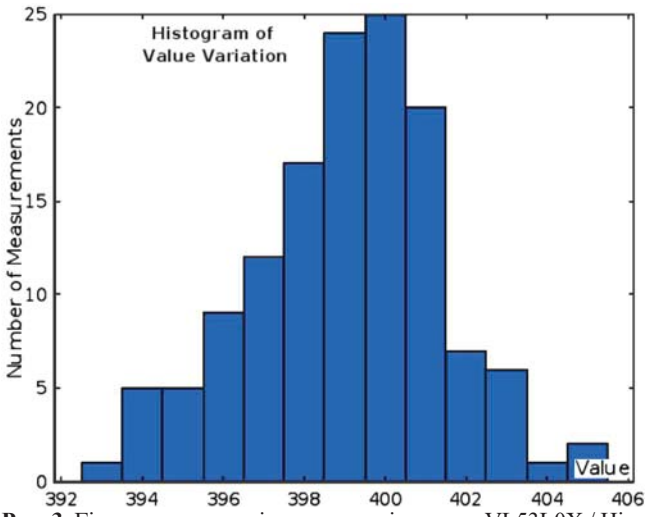

Рис. 3. Гістограма розподілу значень від давача VL53L0X / Histogram of the distribution of values from the VL53L0X sensor

Як показали результати тестування, виміряне значення знаходилося у визначених відповідними тестами межах, а відхилення становило у меншу сторону 8 одиниць, у більшу сторону – 5 одиниць, що відповідало вимогам розроблених тестів. Тестування здійснювалося упродовж 30 хв. За цей час збоїв у нагромадженні даних, пов'язаних зі втратою зв'язку, не було. Часового дрейфу вимірювань не зафіксовано. Отже, відповідно до аналізу отриманих результатів тестування було зроблено висновок про функціонування вказаного спеціалізованого ПЗ з надійністю і точністю, достатньою для функціонування у складі МРТП.

Подібно було протестовано спеціалізоване ПЗ модуля керування на базі виділеного мікроконтролера ESP32. Модуль керування забезпечує отримання відповідних команд на рух МРТП у заданому напрямку від керівного мікрокомп'ютера і формування відповідних управлінських сигналів на модуль Dual Channel H Bridge Motor Driver Module DHB-01A, який забезпечує фізичне керування двигунами МРТП. На вхід модуля керування подавалися команди, які визначають напрямок, швидкість руху та кут повороту МРТП. На їх основі мікроконтролером формувалися відповідні керівні напруги для двигунів з використанням ШІМ. Було отримано відповідні рівні напруг живлення двигунів у межах 5-15 В, що забезпечувало задане переміщення МРТП. Вимоги розроблених модульних тестів було виконано з урахуванням меж тестування.

Отже, у такий спосіб було протестовано спеціалізоване ПЗ виділеного мікроконтролера. Таке рішення забезпечило неперервність формування керівних напруг, що подаються на двигуни і їх стабільність у часі. Було

також протестовано у комплексі спеціалізованого ПЗ роботу аварійного датчика перешкоди. ПЗ мікроконтролера забезпечило отримання сигналу з аварійного датчика перешкоди і аварійну зупинку МРТП при імітації появи перешкоди незалежно від команд, які подавалися з керівного мікрокомп'ютера.

Розроблення інтеграційних тестів. Інтеграційні тести мають вирішальне значення для забезпечення правильної спільної роботи різних компонент і підсистем. Вони використовуються для перевірки належної взаємодії та зв'язку між різними компонентами програмного забезпечення, такими як мікроконтролерні засоби та система Linux у випадку робототехнічної системи. У контексті робототехнічної системи інтеграційні тести можуть допомогти перевірити точність і функціональність внутрішнього протоколу, який використовується для обміну даними між мікроконтролерними засобами та системою Linux. Ці тести також можуть допомогти виявити такі проблеми, як неправильна передача даних, втрата даних або збої зв'язку, які можуть спричинити несправність системи.

Щоб провести інтеграційні тести, потрібно створити набір тестів, який перевіряє взаємодію між тестованими компонентами. Тести перевірять, чи здатні компоненти правильно спілкуватися та чи демонструється очікувана поведінка. Існує кілька найкращих практик з проведення інтеграційних тестів, зокрема: починати з простих тестів, які відпрацьовують основні взаємодії, перш ніж переходити до складніших сценаріїв; використання макетів або емуляторів для будь-яких компонент, які ще не доступні або реалізовані не повністю; створення тестових даних, які репрезентують реальні сценарії використання; ізоляційні тести, щоб гарантувати, що несправності можна легко ідентифікувати та виправити; частий запуск інтеграційних тестів, щоб виявити проблеми якомога раніше в циклі розробки.

Для спрощення написання інтеграційних тестів для компонент програмно-апаратного комплексу можливо використати такі компоненти: Unity; Ceedling; LDRA; VectorCAST.

Інтеграційне тестування спеціалізованого ПЗ було виконано для блоку прийомо-передавача системи керування МРТП. Тестування здійснювалося для організованого тестового каналу передачі даних. На передавальному кінці застосовувалося обладнання центру керування, а саме ноутбука з під'єднаним блоком прийомо-передавача, а на прийомному – аналогічного блоку прийомо-передавача. На виході цього блоку за допомогою імітатора прийняті дані дублювалися у передавальний канал.

Табл. 1. Максимальна протестована швидкість передачі даних для різних значень Air Data Rate / Maximum tested data transfer rate for different values of Air Data Rate

Відстань, м	Air Data Rate	Si24R1, кбіт/с
100	250 кбіт/с	96-120
100	2 Мбіт/с	180-200

Тестування спеціалізованого ПЗ для трансівера Si24R1 виконувалося у умовах відкритої місцевості на відстані 100 м між передавальним і приймальним блоками без перешкод в умовах прямої видимості. Висота розміщення антени центру керування становила біля 2 м, а приймальної частини – 1 м, Тестування максимальної швидкості передачі даних у каналі зв'язку виконувалося відповідно до встановленої швидкості передавання по ефіру (Air Data Rate) для значень 250 кбіт/с та 2 Мбіт/с. Обмін вівся пакетами розміром 32 байти. Розраховувалася середня швидкість на проміжку тривалістю 10 с. Результати наведено у табл. 1.

Аналіз результатів тестування для відстані 100 м показує отриману швидкість передачі даних для трансівера Si24R1 близько 110 кбіт/с для встановленої швидкості трансивера (Air Data Rate) 0,25 Мбіт/с та 190 кбіт/с – для швидкості 2 Мбіт/с. Зменшення швидкості передавання даних пояснюють застосуванням вбудованого алгоритму повторного надсилання пакетів даних за появи збоїв. Отримана швидкість, як показало тестування, залежить від дальності та наявних завад, що призводить до збільшення кількості втрачених пакетів, необхідності їх повторного надсилання каналом, а отже, до зниження середньої швидкості передачі.

Розроблення симуляційних тестів. Симуляційні тести є важливою частиною тестування систем, які покладаються на давачі та пристрої. Використовуючи імітаційні дані давачів, симуляційні тести можуть оцінити, як система обробляє та виводить дані, допомагаючи виявити потенційні проблеми в її поведінці. Інструменти симуляційних сигналів, такі як Simulink або OpenModelica, забезпечують платформу для створення моделей системи та моделювання різних сценаріїв.

Щоб забезпечити ефективність симуляційних тестів, важливо дотримуватися передових практик: визначення критичної функціональності системи, яку необхідно протестувати за допомогою тестів моделювання; визначення реалістичних сценаріїв моделювання та наборів даних; перевірка імітаційної моделі у спосіб порівняння результатів моделювання з реальною поведінкою системи; багаторазове виконання імітаційних тестів для забезпечення їх повторюваності та узгодженості; інтеграція імітаційних тестів із загальним процесом тестування для забезпечення комплексного тестування системи.

Тестування продуктивності та швидкодії програмно-апаратного комплексу. Тестування продуктивності – це тип тестування, який проводиться для оцінювання здатності системи працювати за різних режимів робочих навантажень і рівнів стресу. У контексті робототехнічної системи тестування продуктивності може допомогти виявити вузькі місця продуктивності, проблеми зі стабільністю та складові, які потрібно вдосконалити в системі. Тестування продуктивності має вирішальне значення для розроблення вбудованих систем, оскільки продуктивність системи може безпосередньо впливати на її загальну функціональність, надійність і безпеку. Щоб провести тестування продуктивності, інструменти навантажувального тестування, такі як Apache JMeter, можна використовувати для моделювання реалістичних робочих навантажень і вимірювання здатності системи впоратися з ними.

Щоб виконати тестування продуктивності в розподіленому середовищі програмно-апаратного комплексу, важливо моделювати реалістичні сценарії використання та робочі навантаження, які точно відображають очікуване робоче середовище. Цього можна досягти за допомогою інструментів навантажувального тестування, які підтримують розподілене тестування, таких як Apache JMeter, LoadRunner і Gatling.

У налаштуваннях розподіленого тестування кілька генераторів навантаження використовуються для іміта-

ції трафіку та створення навантаження на систему. Ці генератори навантаження зазвичай встановлюються на окремих компонентах і координуються контролером, який керує виконанням тесту та збирає результати.

Контролер спілкується з генераторами навантаження через мережу, а генератори навантаження надсилають запити до системи, що тестується. Відповіді системи потім збираються та аналізуються для визначення продуктивності системи за різних умов навантаження.

Для виконання тестування продуктивності та швидкодії програмно-апаратного комплексу було виконано контроль швидкості шифрування-дешифрування залежно від заданих ключів, які визначають архітектуру застосованої нейромережі та її тип. Проведено тестування спеціалізованого ПЗ шифрування-дешифрування на основі платформи з архітектурою IA-64 (керуючий ноутбук) і мікрокомп'ютера на базі SoC Allwinner H3+ (МРТП). Тестування виконано для операцій шифрування/дешифрування тестових даних з використанням лінійної та нелінійної архітектур нейроподібної мережі. У процесі тестування архітектура нейроподібної мережі шифрування/дешифрування не змінювалася. Тестування виконувалося для архітектури з параметрами:

- Size of training vectors $n = 16$ bit;
- Number of training vectors $N = 50$;
- Size of neural networks inputs $K\_IN = 2$;
- Number of neural networks inputs $n\_IN = 8$;
- Number recalculations of K_i, M_j coefficients $N\_K\_M = 4$;
- Параметр Polynomial degree Pmax приймав значення 5 та 9.

Тестування спеціалізованого ПЗ шифрування/дешифрування було виконано на базі двох платформ – комп'ютера центру керування (ноутбук) та мікрокомп'ютера на базі SoC Allwinner H3+. Тестування виконувалося для варіантів лінійної та нелінійної (Pmax = 5, 9) нейроподібних мереж на базі ноутбука у середовищі розробки Code::Blocks ver.20.03. (табл. 2).

Табл. 2. Тривалості виконання операцій для заданих конфігурацій нейроподібної мережі (ноутбук) / Durations of execution of operations for given configurations of a neural network (laptop)

Нейромережа / Процедура	Навчання ШНМ (Training), мс	Шифрування ШНМ (EnCrypt), мс	Дешифрування ШНМ (DeCrypt), мс
Лінійна	150	12	8
Нелінійна Pmax=5	550	25	30
Нелінійна Pmax=9	1320	40	45

Результати тестування СПЗ для аналогічного випадку заданих параметрів шифрування/дешифрування за допомогою засобів мікрокомп'ютера (SoC) зведено у табл. 3. Згідно з результатами тестування, для реалізації спеціалізованого ПЗ нейромережевого шифрування/дешифрування даних для всіх конфігурацій нейромережі значень поліномів Pmax найбільш тривалою є процедура формування і навчання нейромережі (СПЗ Training).

Табл. 3. Тривалості виконання операцій для заданих конфігурацій нейроподібної мережі (SoC) / Execution times for given neural network configurations (SoC)

Нейромережа	Навчання ШНМ (СПЗ Training), мс	Шифрування ШНМ (СПЗ EnCrypt), мс	Дешифрування ШНМ (СПЗ DeCrypt), мс
Лінійна	180	15	10
Нелінійна Pmax=5	750	30	35
Нелінійна Pmax=9	1480	40	45

Тривалість виконання в обох випадках становить від 160-200 мс для лінійної нейромережі до 600 мс і більше – для нелінійної. Проте вказане СПЗ застосовується однократно за зміни ключа шифрування і надалі не впливає на СПЗ шифрування/дешифрування, оскільки там вже застосовується "навчена" мережа.

Тестування продуктивності показало, що тривалість виконання СПЗ шифрування/дешифрування для окремих блоків даних (16 біт) становить близько 10-15 мс для лінійної мережі та відповідно 25-35 мс ($P_{max} = 5$) – для нелінійної. Такі тривалості виконання СПЗ шифрування/дешифрування є прийнятними для функціонування каналу управління МРТП у режимі реального часу. Шляхом збільшення степені полінома можна досягнути більшої захищеності каналу за прийнятного збільшення тривалості виконання операцій СПЗ.

Тестування безпеки програмно-апаратного комплексу. Тестування безпеки є важливим аспектом розроблення програмного забезпечення, який зосереджується на виявленні потенційних ризиків безпеці або вразливостей у системі. У контексті МРТП тестування безпеки має вирішальне значення для забезпечення захисту системи від несанкціонованого доступу чи зловмисних атак. Одним із поширених інструментів тестування безпеки є Nessus, який є сканером вразливостей, що можна використовувати для виявлення та визначення пріоритетів проблем безпеки у системі. Тестування безпеки для програмно-апаратного комплексу МРТП має відбуватися за чітко визначеною послідовністю, щоб переконатися, що всі аспекти системи безпечні. Водночас без належного тестування програмно-апаратний роботизований комплекс може зіткнутися з різними проблемами. Наприклад, неправильна передача даних між мікроконтролерними компонентами та системою Linux може призвести до неправильного прийняття рішень системою, що призведе до неправильного руху або функціонування. Втрата даних або збої зв'язку можуть призвести до непередбачуваної поведінки системи або навіть до несправності. Невідповідне тестування продуктивності може призвести до того, що система не зможе впоратися з очікуваним навантаженням, що призведе до затримок або інших проблем. Уразливості системи безпеки можуть бути використані зловмисниками, що потенційно може завдати шкоди системі або середовищу, в якому вона працює. Звідси випливає, що відсутність ретельного тестування може призвести до того, що система стане ненадійною, непередбачуваною та потенційно небезпечною.

Отже, у дослідженні отримано такі основні результати: вдосконалено метод тестування спеціалізованого програмного забезпечення МРТП, який завдяки використанню спеціалізованого обладнання та технологічних програмних засобів забезпечує спільне тестування та відлагодження програмних та апаратних засобів для функціонування у режимі реального часу.

Обговорення результатів дослідження. Сьогодні дослідники велику увагу приділяють питанням створення систем керування для різноманітних роботизованих та автономних платформ. При цьому зараз системи керування такими платформами здебільшого створюються з використанням вбудованих систем (embedded systems), які доволі часто реалізуються на базі поєднання декількох архітектур обчислювальних засобів. Певні відмінності в організації обчислювального процесу, реалізації інтерфейсів таких різнорідних компонент, які повинні працювати у складі єдиної системи, висувають

вимоги до чіткішої організації процесу тестування таких систем, і особливо, безперешкодної їх взаємодії.

У роботі [16] розглянуто процес тестування програмного забезпечення з використанням генетичних алгоритмів, але недостатньо уваги приділено особливостям тестування саме вбудованих систем. У роботі [12] досліджено процес тестування компонент у компонентному програмному забезпеченні, проте не проаналізовано особливостей тестування для вбудованих систем управління у режимі реального часу. Системи управління МРТП, розроблені у середовищі моделювання з використанням програмного забезпечення, розглянуто у роботах [1, 10], проте недостатньо описано та проаналізовано етапи перевірки правильності функціонування таких систем. Спеціалізоване програмне забезпечення і надійність його роботи, яка визначається саме під час тестування, відіграють у цьому процесі вирішальне значення.

Отже, за результатами виконаної роботи можна сформулювати такі наукову новизну та практичну значущість результатів дослідження.

Наукова новизна отриманих результатів дослідження – вперше розроблено структуру засобів і сценаріїв тестування спеціалізованого програмного забезпечення мобільної робототехнічної платформи, що забезпечують комплексне тестування програмно-апаратних засобів для функціонування у режимі реального часу.

Практична значущість результатів дослідження – розроблену систему тестування спеціалізованого програмного забезпечення МРТП можна використати для забезпечення комплексного тестування та відлагодження широкого класу вбудованих програмно-апаратних засобів.

Висновки / Conclusions

Розроблено структуру засобів і сценаріїв тестування спеціалізованого програмного забезпечення для мобільної робототехнічної платформи, що дало змогу визначити стан якості апаратно-програмного комплексу засобів, які реалізують різні функціональності платформи. За результатами виконаного дослідження можна зробити такі основні висновки:

1. Досліджено основні етапи тестування спеціалізованого програмного забезпечення мобільних робототехнічних платформ.

2. Розроблено структуру спеціалізованого програмного забезпечення мобільних робототехнічних платформ. Із застосуванням комплексного тестування здійснено перевірку на прикладі функціональності мікроконтролерного засобу.

3. З використанням розроблених принципів і підходів розроблено модульне тестування для перевірки функціональності мікроконтролерних засобів. Розроблено інтеграційні тести для забезпечення правильної спільної роботи різних компонент і підсистем програмно-апаратного комплексу.

4. Для перевірки здатності системи обробляти і виводити дані, розроблено симуляційні тести з використанням імітаційних даних датчиків.

5. Проведено тестування продуктивності та швидкодії програмно-апаратного комплексу. Для цього було враховано контроль швидкості шифрування-дешифрування залежно від заданих ключів, які визначають архітектуру застосованої нейромережі і її тип Тестування продуктивності показало, що тривалість виконання СПЗ операцій шифрування/дешифрування є прийнятними для функціонування каналу управління МРТП у режимі реального часу. Встановлено, що шляхом збільшення степені полінома нейроподібної мережі можна досягнути більшої захищеності каналу при прийнятному збільшенні тривалості виконання операцій спеціального програмного забезпечення.

References

1. Afaq, M., Jebelli, A., & Ahmad, R. (2023). An intelligent thermal management fuzzy logic control system design and analysis using ANSYS fluent for a mobile robotic platform in extreme weather applications. J Intell Robot Syst 107, 11. https://doi.org/10.1007/s10846-022-01799-7
2. Afzal, A., Goues, C. L., Hilton, M., & Timperley, C. S. (2020). A study on challenges of testing robotic systems. 13th International Conference on Software Testing, Validation and Verification (ICST), Porto, Portugal, 96–107. https://doi.org/10.1109/ICST46399.2020.00020
3. Allagui, N. Y., Salem, F. A., & Aljuaid, A. M. (2021). Artificial fuzzy-PID gain scheduling algorithm design for motion control in differential drive mobile robotic platforms., Computational Intelligence and Neuroscience, 1–13. https://doi.org/10.1155/2021/5542888
4. Asfaw, D. (2015). Benefits of automated testing over manual testing. *International journal of innovative research in information security* (IJIRIS), 2(1), 5–13. URL: https://www.academia.edu/26811709/Benefits_of_Automated_Testing_Over_Manual_Testing
5. Bozhinoski, D., Ruscio, D., Malavolta, I., Pelliccione, P., & Crnkovic, I. (2019). Safety for mobile robotic systems: A systematic mapping study from a software engineering perspective. *Journal of Systems and Software, 151*, 150–179. https://doi.org/10.1016/j.jss.2019.02.021
6. Falsafi, M., Alipour, K., & Tarvirdizadeh, B. (2019). Fuzzy motion control for wheeled mobile robots in real-time. *Journal of Computational & Applied Research in Mechanical Engineering* (*JCARME*), *8*(2), 133–144. https://doi.org/10.22061/jcarme.2018.2204.1205
7. Gamido, H. V., & Gamido, M. V. (2019). Comparative review of the features of automated software testing tools. *International Journal of Electrical and Computer Engineering* (IJECE), *9*(5), 4473–4478. https://doi.org/10.11591/ijece.v9i5.pp4473-4478
8. García, S., Strüber, D., Brugali, D., Berger, T., & Pelliccione, P. (2020). Robotics software engineering: a perspective from the service robotics domain. Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2020). Association for Computing Machinery, New York, NY, USA, 593–604. https://doi.org/10.1145/3368089.3409743
9. Hrytsiuk, Yu. I., & Nemova, E. A. (2018). Management Features Process of Developing Software Requirements. *Scientific Bulletin of UNFU*, *28*(8), 161–169. https://doi.org/10.15421/40280832
10. Huang, H. – W., Chen, J., Chai, P. R., Ehmke, C., Rupp, P., Dadabhoy, F. Z., Feng, A., Li, S., Thomas, A. J., Silva, M., Boyer, E. W., & Traverso, G. (2022). Mobile robotic platform for contactless vital sign monitoring. Cyborg and Bionic Systems, 1–11. https://doi.org/10.34133/2022/9780497
11. ISO 9001:2015. Quality Management System – Requirements. URL: https://www.iso.org/standard/62085.html
12. Kumar, P. (2022). Issue and challenges in component testing in component based software engineering. Proceedings of the MOL2NET22 Conference on Molecular, Biomed., Comput. & Network Science and Engineering, 8th ed., MDPI, 1–5. https://doi.org/10.3390/mol2net-08-12632
13. Lavkesh, Mittal H. (2016). Comparative analysis of automated functional testing tools. *Journal of Network Communications and Emerging Technologies* (JNCET), *6*(6), 50–53. URL:

https://www.jncet.org/Manuscripts/Volume-6/Issue-6/Vol-6-issue-6-M-11.pdf
14. Nie, J., Wang, Y., Miao, Z., Jiang, Y., Zhong, H., & Lin J. (2021). Adaptive fuzzy control of mobile robots with full-state constraints and unknown longitudinal slipping. *Nonlinear Dyn, 106*, 3315–3330. https://doi.org/10.1007/s11071-021-06933-y
15. Nitulescu, M. (2007). Solution for modeling and control in mobile robots. *Journal of Control Engineering and Applied Informatics, 9*(3), 43–50.
16. Rajora, I., Sandhu, L., Bhat, M. Y., & Beniwal, R. (2023). Software testing using genetic algorithms – A review. *Journal of current research in engineering and science, 6*(1), 9, 1–13. URL: https://www.psvpec.in/jcres/2023_1/109.pdf; https://doi.org/10.5121/ijcses.2016.7203
17. Salem, F. A. (2013). Dynamic and kinematic models and control for differential drive mobile robots. *International Journal of Current Engineering and Technology, 3*(2), 253–263. URL: https://inpressco.com/wp-content/uploads/2013/03/Paper6253-2632.pdf
18. Thomas, C. M. (2014). An overview of the current state of the test-first vs. tst-last debate. Scholarly horizons: university of minnesota. *Morris Undergraduate Journal, 1*(2), 1–6. https://doi.org/10.61366/2576-2176.1015
19. Verma, V., & Malhotra, S. (2011) Applications of software testing metrics in constructing models of the software development process. *Journal of Global Research in Computer Science, 2*, 96–98. URL: https://www.rroij.com/open-access/applications-of-software-testing-metrics-in-constructing-models-of-the-software-development-process.php?aid=37488
20. Zasornova, I., Hovorushchenko, T., & Voichur, O. (2023). Study of software testing tools according to the testing levels. Computer systems and information technologies, 1, 38–46. https://doi.org/10.31891/csit-2023-1-5

I. G. Tsmots, Yu. V. Opotyak, M. Ya. Seneta, Yu. Yu. Oliynyk, N. B. Gazda

Lviv Polytechnic National University, Lviv, Ukraine

STRUCTURE AND FEATURES OF THE MAIN STAGES OF TESTING THE SPECIALIZED SOFTWARE OF THE MOBILE ROBOTIC PLATFORM

Some methods of testing the hardware and software complex of mobile robotic platforms have been developed. The architecture of the software-hardware complex of tools was considered and the interaction of the components of the specialized software of mobile robotics platforms was investigated. A microcomputer platform based on the SoC running the Linux OS is used to ensure the management of the hardware and software complex. Simulation tests were conducted to simulate sensor signals and check the system ability to process and output data. The performance testing system of the specialized software of the mobile robotics platform was studied to evaluate the speed, responsiveness, and stability of the system. A comprehensive software and hardware testing plan has been created with a structured approach. Joint testing of software and hardware tools was carried out in real time with the use of specialized equipment (generators of input sequences of control signals, generators of input data, tables of reference results, means of comparison) and technological software tools. It is shown that the main stages of testing specialized software of a mobile robotics platform are as follows: analysis of requirements for specialized software; development of a testing plan; functional testing; performance testing; vulnerability testing; compatibility; user interface testing; error testing; testing real scenarios; testing in dynamic conditions. Testing of specialized software using ESP32 control module was carried out. For the organized data transfer test channel, integration testing of specialized software for the receiver-transmitter unit of the control system of the mobile robotics platform was performed. Analysis of test results was carried out and average data transfer rates were calculated. Testing of specialized encryption/decryption software was performed on the basis of two platforms – a control center computer (laptop) and a microcomputer based on the Allwinner H3+ SoC.

Keywords: mobile robotics platform; testing; software quality; fuzzy motion control.