Вінницький національний технічний університет
Факультет інфокомунікацій, радіоелектроніки та наносистем
Кафедра телекомунікаційних систем та телебачення

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Методи тестування програмно-керованих інфокомунікаційних систем»

Виконав: студент 2-го курсу,
групи ТКС-20м
спеціальності 172 – Телекомунікації та радіотехніка
_____ Щербань О.А.

Керівник: к.т.н., доцент каф. ТКСТБ
_____Васильківський М.В.
«____» _____ 2021 р.

Опонент: д.т.н., професор каф. РТ
_____ Осадчук В.С.
«____» _____ 2021 р.

Допущено до захисту
Завідувач кафедри ТКСТБ
_____д.т.н., проф. Кичак В.М.
«____» _____ 2021 р.

Вінниця ВНТУ - 2021 рік

Вінницький національний технічний університет
Факультет інфокомунікацій, радіоелектроніки та наносистем
Кафедра телекомунікаційних систем та телебачення
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань - 17– Електроніка та телекомунікації
(шифр і назва)
Спеціальність - 172 – Телекомунікації та радіотехніка
(шифр і назва)
Освітньо-професійна програма - Телекомунікаційні системи та мережі

ЗАТВЕРДЖУЮ
Завідувач кафедри ТКСТБ
д.т.н., професор В.М. Кичак
“___”_____2021 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Щербаню Олегу Андрійовичу .
(прізвище, ім’я, по батькові)

1. Тема роботи Методи тестування програмно-керованих інфокомунікаційних систем
керівник роботи Васильківський Микола Володимирович, канд. техн. наук, доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “24” 09 2021 року № 277

2. Строк подання студентом роботи 01 грудня 2021 року

3. Вихідні дані до роботи тривалість пошуку помилок пам’яті в програмних ТКС при використанні стандартних засобів тестування – 60-100 год; тривалість пошуку помилок пам’яті в програмних ТКС при використанні запропонованих методів тестування – 12 год; тривалість пошуку спотворення даних в програмних ТКС при використанні стандартних засобів тестування – 65-90 год; тривалість пошуку спотворення даних в програмних ТКС при використанні запропонованих методів тестування – 17-32 год; тривалість пошуку спотворення тривалості потоків в програмних ТКС при використанні стандартних засобів тестування – 92-110 год; тривалість пошуку спотворення тривалості потоків в програмних ТКС при використанні запропонованих методів тестування – 8-14 год; підвищення ефективності виявлення всіх помилок в програмному забезпеченні ТКС – 82%; підвищення ефективності виявлення помилок, здатних призводити до відмов та збоїв в програмному забезпеченні ТКС – 5-19%.

4. Зміст текстової частини: відмовостійкі пристрої інформаційних комплексів; програмно-апаратний пристрій тестування інфокомунікаційної системи; структура комплексу та алгоритми тестування телекомунікаційних мікропроцесорних пристроїв; результати досліджень пристроїв інформаційного комплексу.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень) загальна структура імітатора несправності; архітектура системи автоматизованого тестування; структурна схема пристрою імітації несправності та аналізу реакції; алгоритм адаптивної передачі інформації; алгоритм тестування пристрою за допомогою JTAG; функціональна схема комплексу проведення випробувань; структурна схема імітаційних досліджень; структура програмно-апаратних компонентів налагодження.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Спеціальна частина	Васильківський М.В., доцент кафедри ТКСТБ		

7. Дата видачі завдання 01 вересня 2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка технічного завдання	10.09.2021р.	
2.	Відмовостійкі пристрої інформаційних комплексів	17.09.2021р.	
3.	Програмно-апаратний пристрій тестування інфокомунікаційної системи	01.10.2021р.	
4.	Структура комплексу та алгоритми тестування телекомунікаційних мікропроцесорних пристроїв	29.10.2021р.	
5.	Результати досліджень пристроїв інформаційного комплексу	19.11.2021р.	
6.	Аналіз економічної ефективності розробки	30.11.2021р.	
7.	• Охорона праці та безпека життєдіяльності	06.12.2021р.	
8.	Оформлення пояснювальної записки та графічної частини	13.12.2021р.	
9.	Нормоконтроль МКР	14.12.2021р.	
10.	Попередній захист МКР, опонування МКР	17.12.2021р.	
11.	Захист МКР ЕК	20.12.2021р.	

Студент _____ Щербань О.А.
(підпис)

Керівник роботи _____ Васильківський М.В.
(підпис)

АНОТАЦІЯ

УДК 621.391

Щербань О. А. Методи тестування програмно-керованих інфокомунікаційних систем – магістерська кваліфікаційна робота зі спеціальності 172 – Телекомунікації та радіотехніка, освітня програма – Телекомунікаційні системи та мережі – Вінниця: ВНТУ 2021 р. 160 - стор., 50 – рис., 34 – табл., 28 – бібл. – українською мовою.

Розроблено алгоритм аналізу проблемних ситуацій, що виникають при роботі автоматизованої радіостанції, інтегрованої в діючу інфраструктуру цифрових мереж з множинним доступом. Алгоритм реалізує тестування функцій радіостанції за допомогою техніки фаззінга (випадкового внесення спотворень в дані). Це дозволяє виконувати автоматичну діагностику технічного стану радіостанції та усувати системні помилки, що виникають при завантаженні призначених для користувача програм в реальне апаратне середовище пристрою (прояв властивості емерджентності).

Запропоновано модифікацію алгоритму фаззінга, що реалізує еволюційну стратегію трансформації вхідних даних перевіряється. Алгоритм розглядає набір програмних функцій, представлений у вигляді вершин графа переходів. Стратегія фаззінга передбачає на кожному етапі зміни даних (перестановка бітів і байтів, арифметичні операції), аналіз реакції програми на внесення спотворення. Модифікація алгоритму полягає в застосуванні в якості вхідних даних пристрою попередньо опрацьованих масивів найбільш часто виконуваних функцій, що дозволяє оцінити коректність роботи для основних вразливостей ПЗ та зменшити часові витрати на тестування (приблизно в три рази в порівнянні з класичними методами тестування ПЗ).

Ключові слова: мікроконтролер, програмована логічна інтегральна схема, програмне забезпечення, цифровий сигнальний процесор, спотворення вхідних даних основних програмних функцій, технічний засіб імітації несправності, телекомунікаційна система.

ANNOTATION

UDC 621.391

Shcherban O. A. Methods of testing software-controlled infocommunication systems - master's thesis in specialty 172 - Telecommunications and radio engineering, educational program - Telecommunication systems and networks - Vinnytsia: VNTU 2021 160 - pp., 50 - fig., 34 - table., 28 - bibl. - in Ukrainian.

An algorithm for analyzing problematic situations that arise during the operation of an automated radio station integrated into the existing infrastructure of digital networks with multiple access has been developed. The algorithm implements testing of radio station functions using fuzzing technique (random introduction of distortions into data). This allows you to automatically diagnose the technical condition of the radio station and eliminate system errors that occur when downloading user programs in the real hardware environment of the device (manifestation of the property of emergence).

A modification of the fuzzing algorithm that implements the evolutionary strategy of input data transformation is proposed. The algorithm considers a set of software functions represented as vertices of the transition graph. Fuzzing strategy involves at each stage of data change (permutation of bits and bytes, arithmetic operations), analysis of the program's response to distortion. Modification of the algorithm is to use as input device pre-processed arrays of the most frequently performed functions, which allows you to assess the correctness of work for major software vulnerabilities and reduce time costs for testing (approximately three times compared to classical software testing methods).

Key words: microcontroller, programmable logic integrated circuit, software, digital signal processor, distortion of input data of basic software functions, technical means of simulation of malfunction, telecommunication system.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АРМ - автоматизоване робоче місце;

АСУ - автоматизована система управління;

МК - мікроконтролер;

ОЗП - оперативний запам'ятовуючий пристрій;

ОВ - об'єкт випробувань;

ПЛІС - програмована логічна інтегральна схема;

ПЗ - програмне забезпечення;

РЕА - радіоелектронна апаратура;

САПР - система автоматизованого проектування;

НВІС - надвеликі інтегральні схеми

СУ - станція управління;

EN 50128 - європейський стандарт для забезпечення безпеки «Залізничні додатки - Системи зв'язку, сигналізації та обробки»;

ARM - сімейство готових топологій мікропроцесорних / мікроконтролерних ядер на основі удосконаленої RISC-машини;

AVR - сімейство RISC-мікроконтролерів;

DSP - цифровий сигнальний процесор;

FI - внесення несправності;

FMEA - аналіз видів і наслідків відмов;

FMECA - аналіз критичності відмов;

FMEDA -діагностичний аналіз наслідків відмов;

FPGA - програмована логічна інтегральна схема;

IEC 61508 - міжнародний стандарт «Функціональна безпека електричних / електронних / програмованих електронних систем, пов'язаних з безпекою»;

JTAG - стандарт тестової архітектура граничного сканування IEEE 1149.1;

MIPS - система команд і мікропроцесорних архітектур, розроблених компанією MIPS.

ВСТУП

Актуальність теми. В даний час одним із важливих завдань сучасної вітчизняної промисловості є розробка складних пристроїв з унікальними споживчими властивостями, здатних виконувати свої функції з високим рівнем стійкості до відмов. При цьому для апаратно-програмних комплексів з великою питомою вагою програмної частини увага приділяється вивченню методів, що ведуть до скорочення термінів розробки зразків і підвищенню їх відмовостійкості. Удосконалення методів і засобів підвищення відмовостійкості для мікропроцесорних пристроїв пов'язане з проблемою ефективності застосування відомих стандартів і методик для малосерійних пристроїв інформаційних комплексів. Крім того, для регламентованої в стандартах процедури тестування цифрових пристроїв за допомогою імітації несправності не визначено конкретних алгоритмів і необхідного комплексу технічних засобів.

Теоретичне і практичне вирішення проблеми забезпечення підвищення стійкості до відмов для проектованих мікропроцесорних пристроїв може бути досягнуто шляхом синтезу способів експериментальних досліджень по імітації несправності за допомогою спотворення вхідних даних і переднього оброблення статистичної інформації для програмного забезпечення (ПЗ), завантаженого в апаратне середовище мікропроцесорного пристрою, а також аналізу реакції на внесення несправності і визначення достатності обсягу випробувань за допомогою нечіткого логічного висновку.

Аналіз останніх досліджень. Сучасні стандарти розробки і проектування технічних програмно-апаратних пристроїв, засновані на методиці FMEA, знаходяться в системній кризі в зв'язку з постійним зростанням складності систем управління інформаційних комплексів і умов їх застосування, з одного боку, і в зв'язку з проявом властивості емерджентності цих систем, яке не враховується на етапі проектування.

Значний внесок у підвищення стійкості до відмов складних програмно-апаратних пристроїв внесли вітчизняні та зарубіжні вчені Пархоменко П.П.,

Черкесов Г.Н., Ушаков І.В., Avizienis A., Randell B. , Herrin S. , Katsuk і D. , Globl W. та ін. Питаннями проектування засобів імітації несправності займаються вчені Лучинин В.В., Arlant J., Elks C., Rannels D., Williams R., Gaisler J., Natella R. Питанням підвищення якості та надійності ПЗ присвячені роботи вчених Ліпаева В.В., Ковальова І.В., Boehm BW, Meyer J., Lyu MR, Levendel Y., Shooman ML, Tai A., Xie M, Zhou L. Дослідженням властивостей програмного забезпечення і безпеки програм присвячені роботи Berman O., Choi JG, Epstein D., She D., Pei K [1-5].

Існує список так і не вирішених в теорії проблем надійності, який був опублікований на Міжнародній конференції MMR-2000 в доповіді «Надійність: минуле, сучасне, майбутнє». На першому місці в цьому списку знаходиться проблема надійності програмного забезпечення, на третьому місці - питання надійності унікальних високо відповідальних систем. Проблеми надійності глобальних територіальних систем і телекомунікаційних мереж займає четверте і п'яте місце. До вирішення цих проблем є тільки підходи. Теорія надійності сформувалася як частина прикладної математики і розробила різні методи вирішення проблем надійності, розвинені в математичні моделі, які все ще не досягли рівня, придатного для застосування на практиці (проблема відсутності необхідних вихідних даних).

Процедура аналізу видів, причин і наслідків відмов PEA є обов'язковою складовою частиною процесу проектування і відпрацювання пристроїв, починаючи з розробки ескізного проекту до випробувань дослідних зразків.

Необхідність проведення FMEA в процесі розробки і проектування була визначена в якості додаткового етапу (по відношенню до стандарту менеджменту якості ISO 9001) і становить вимоги для постачальників в рамках корпоративної системи. На основі проведеного аналізу зроблено висновок про доцільність розробки технічних рішень для практичного застосування методики FMEA до мікропроцесорних пристроїв у вигляді розроблених алгоритмів, методик і пристроїв.

Основна ідея роботи полягає у використанні попередньо опрацьованих статистичних даних про функціонування пристрою, що перевіряється в алгоритмах імітації та виявлення помилок ПЗ і апаратних відмов.

Мета і завдання роботи. Метою даної кваліфікаційної роботи є підвищення стійкості до відмов мікропроцесорних пристроїв за допомогою технічних і програмних засобів тестування на основі імітації несправності інформаційного комплексу телекомунікаційної системи.

Задачами магістерської кваліфікаційної роботи є:

1. Аналіз проблеми стійкості до відмов програмно-апаратних комплексів і підходів до забезпечення їх працездатності в процесі проектування.

2. Розробка алгоритму аналізу проблемних ситуацій, що виникають при роботі мікропроцесорного пристрою, на основі виявлення несправності за допомогою спотворення вхідних даних.

3. Розробка алгоритму пошуку несправності на основі спотворення вхідних даних основних програмних функцій з використанням попередньо оброблених статистичних показників роботи пристрою.

4. Створення технічних засобів імітації несправності і алгоритму діагностики для мікропроцесорних пристроїв в складі інформаційного комплексу.

5. Розробка методу і критерію визначення обсягу тестових випробувань на основі статистично оброблених експериментальних даних про роботу пристрою. Створення математичної моделі для обгрунтування критерію і алгоритму прийняття рішень про достатність обсягу випробувань.

6. Реалізація і впровадження отриманих теоретичних результатів у вигляді методик, моделей, алгоритмів і програм, що використовуються при аналізі і синтезі мікропроцесорних пристроїв для інформаційного комплексу.

Об'єктом дослідження є способи дослідження стійкості до відмов мікропроцесорних пристроїв з програмним забезпеченням.

Предметом дослідження є методи, алгоритми та математичні моделі, що дозволяють забезпечити працездатність мікропроцесорного пристрою в ТКС.

Методи досліджень базуються на використанні: системного аналізу, методів теорії ймовірностей і математичній статистиці; теорії прийняття рішень, включаючи методи теорії важливості критеріїв; методах тестування і розробки програмно-апаратних систем, методики імітації несправності і техніці фазінга.

Новизна одержаних результатів:

1. Розроблено алгоритм аналізу проблемних ситуацій, що виникають при роботі автоматизованої радіостанції, інтегрованої в діючу інфраструктуру цифрових мереж з множинним доступом. Алгоритм реалізує тестування функцій радіостанції за допомогою техніки фаззінга (випадкового внесення спотворень в дані). Це дозволяє виконувати автоматичну діагностику технічного стану радіостанції та усувати системні помилки, що виникають при завантаженні призначених для користувача програм в реальне апаратне середовище пристрою (прояв властивості емерджентності).

2. Запропоновано модифікацію алгоритму фаззінга, що реалізує еволюційну стратегію трансформації вхідних даних перевіряється. Алгоритм розглядає набір програмних функцій, представлений у вигляді вершин графа переходів. Стратегія фаззінга передбачає на кожному етапі зміни даних (перестановка бітів і байтів, арифметичні операції), аналіз реакції програми на внесення спотворення. Модифікація алгоритму полягає в застосуванні в якості вхідних даних пристрою попередньо оброблених масивів найбільш часто виконуваних функцій, що дозволяє оцінити коректність роботи для основних вразливостей ПЗ та зменшити тимчасові витрати на тестування (приблизно в три рази в порівнянні з класичними методами тестування ПЗ).

3. Запропоновано пристрій імітації несправності в програмно-апаратних системах для перевірки роботи мікропроцесорного пристрою і розроблений експертний алгоритм, призначений для виявлення найбільш вразливих місць програми і заснований на аналізі ознак відмов і збоїв в регістрах ОЗП мікропроцесорного пристрою. Алгоритм дозволяє спотворити вхідні дані і виявити зміни характеристик роботи пристрою, що перевіряється, обумовлених

зміною порядку і часу виконання програмних функцій, що свідчить про наявність відмов або збоїв.

4. Розроблено метод і запропоновано критерій визначення обсягу тестових випробувань мікропроцесорних пристроїв. Створено математичну модель для обґрунтування критерію, що враховує попередньо отримані характеристики пристрою: число апаратних відмов, ймовірність помилок при виконанні програмних функцій, а також кореляцію між часом внесення помилок у вхідні дані і часом виявлення несправності. Метод, заснований на попередній статистичній обробці експериментальних даних про виконувані функції в сукупності з алгоритмом прийняття рішення на базі нечіткого логічного висновку, дозволяє оцінити достатність обсягу випробувань.

5. Запропоновано алгоритм діагностики програмно-апаратного комплексу, який дозволяє оптимізувати швидкість передачі даних шляхом пошуку найкоротшого шляху в сукупності з ранжируванням інтерфейсів прийому / передачі при наявності великого числа інтерфейсів (вузлів). Підтверджено ефективність алгоритму в умовах обмеження на час обміну інформацією між вузлами (час зменшується приблизно в два рази).

Практичне значення.

1. Методики і алгоритм аналізу проблемних ситуацій автоматизованої радіостанції, інтегрованої в діючу інфраструктуру цифрових мереж з множинним доступом, який реалізує тестування функцій радіостанції за допомогою техніки фаззінга (випадкового внесення спотворень в дані) і виявляє системні дефекти на основі обробки інформації, отриманої за допомогою інтерфейсу налагодження JTAG;

2. Програмний комплекс для автоматизації проведення випробувань мікропроцесорних пристроїв (в комплексній інфраструктурі тестування), що включає програму внесення помилок у вхідні дані, а також програму імітації системних подій (виникнення умов прийому і передачі даних) і збору інформації про реакцію пристрою на внесення спотворення з контрольно-вимірювальних приладів;

3. Імітаційна модель імітатора несправності для перевірки роботи мікропроцесорних пристроїв, що дозволяє виявити основні уразливості ПЗ пристрою шляхом порівняння параметрів його стану зі штатними значеннями, результатом якого є виявлення точок виникнення відмов і збоїв пристрою, що перевіряється.

Апробація роботи та її основні результати роботи проводилися на VIII Міжнародній науково-практичній конференції «Сучасні проблеми інфокомунікацій, радіоелектроніки та наносистем» (СПІРН-2021) у 2021 році.

1 ВІДМОВОСТІЙКІ ПРИСТРОЇ ІНФОРМАЦІЙНИХ КОМПЛЕКСІВ

У першому розділі розглядається комплекс проблем в галузі забезпечення стійкості пристроїв інформаційних комплексів до відмов, що включає проблему надійності ПЗ, проблему гармонізації вітчизняних стандартів в галузі проектування цифрових пристроїв, проблему прояву системних помилок при завантаженні призначених для користувача програм в реальне апаратне середовище пристрою.

1.1 Сучасний стан забезпечення стійкості до відмов програмно-апаратних комплексів

Сучасні стандарти розробки і проектування мікропроцесорних пристроїв засновані на наступних методиках: FMEA (Failure Mode and Effects Analysis, для аналізу видів і наслідків відмов), FMECA (Failure Mode, Effects, and Criticality Analysis, для аналізу ефектів режиму відмови і критичності), FMEDA (Failure Modes, Effects, and Diagnostic Analysis, для діагностичного аналізу ефектів і режимів відмов). Стандарти знаходяться практично в перманентній системній кризі, що відбивається в їх безперервній зміні і створенні нових стандартів [1, 7, 8, 9, 10, 11, 12, 13, 14]. Це відбувається у зв'язку з постійним зростанням складності програмно-апаратних комплексів та умов їх застосування, з одного боку, і з проявом властивості емерджентності систем, яка не враховується і не може бути враховано на етапі проектування, з іншого.

Актуальні стандарти, службовці керівництвом при створенні цифрових пристроїв, спираються на процедури визначення видів, наслідків та причин відмов з оцінкою ризику для виконання запобіжних або коригувальних дій (FMEA) [9, 10], обчислення на основі FMEA-аналізу показників безпеки - числа пріоритетності ризику або критичності відмови (FMECA) [7], знаходження за результатами FMECA-аналізу частоти (інтенсивності) відмов для оцінки надійності (FMEDA) [15]. FMEA- і FMECA-аналізи прописані в стандарті IEC

61508-7. Однак навіть на рівні стандартів існують суперечності в трактуванні процедури FMEA, який в більшості стандартів розглядається як якісний (не кількісний) метод аналізу (IEC 61508-7) [88], але в Гості Р ІСО 26262-9-2014 [11] той же FMEA вже стає кількісним методом аналізу виду і наслідків відмов і може бути використаний як кількісний метод аналізу для випадкових відмов апаратних засобів у разі наявності відомостей про кількісні значеннях інтенсивності відмов (але в стандарті не пояснюється, як саме). Так проявляється проблема гармонізації стандартів.

Стійкість до відмов визначається як здатність підсистеми продовжувати коректно виконувати необхідні функції при наявності відмов, збоїв або помилок. При цьому в ході розробки нового вироби воно існує в єдиному екземплярі у вигляді макетного зразка або прототипу і на цьому етапі життєвого циклу вироби є унікальним.

Для тестування апаратного та програмного забезпечення використовується методика «ін'єкції несправності» (fault injection) [1, 5, 9], за допомогою якої можна змоделювати будь-який вектор наслідків несправності і будь-яку кількість цих векторів, а також зробити програмне забезпечення для багаторазового відтворення сценарію векторів наслідків несправності або імітації несправності, що передбачено в міжнародних стандартах [19] IEC 61508, EN 50128 і ISO 26262 та їх вітчизняних аналогах.

Перераховані стандарти не визначають обсяги випробувань, вимоги до апаратури та технології проведення випробувань. Технологія для проведення випробувань з імітації несправності може бути створена тільки на основі апаратури імітації. Для випробування на стійкість до відмов відбувається створення системи тільки для цілей тестування, яка імітує поведінку обладнання під контролем.

Дослідженню проблеми робастності (що є наслідком прояву небажаних емерджентних властивостей при проектуванні программноаппаратних систем) присвячений ряд робіт [2, 4, 16]. Для її проявів в процесі проектування пристроїв застосовують спеціальні інструменти, в тому числі імітують несправності [3].

Одним з напрямків досліджень з допомогою спотворення вхідних даних (техніка фаззінга) має актуальність практично для всіх сучасних мікропроцесорних пристроїв з програмним забезпеченням.

За останні тридцять років складність програмного забезпечення інформаційних комплексів зросла на кілька порядків [3]. Проте, вітчизняна промисловість використовує застарілі технології для проектування цифрових систем різного ступеня складності [5]. ПЗ не виділяється в окремий «смисловий» елемент системи. Якщо підходити до програмних продуктів, що випускаються сучасними виробниками, жодна комерційна компанія не несе відповідальності за експлуатацію вразливостей і недокументованих можливостей системи.

Сучасні мікропроцесори з вбудованим програмним забезпеченням, розроблені в зарубіжних компаніях, часто містять програмно-апаратні «закладки», призначені для налагодження і тестування системи, але при цілеспрямованих діях можуть використовуватися для знищення або припинення роботи інформаційних систем.

У методології розробки відмовостійкого ПЗ дослідники виділяють три основні групи проблем [2, 3]: - відсутність єдиної методології створення отказоустойчивого ПЗ; - відсутність єдиної методології тестування отказоустойчивого ПЗ; - відсутність єдиного підходу до аналізу проблемної області.

Альтернативний погляд на цю проблему передбачає, що такої проблеми не існує в принципі. Логіка даного підходу полягає в тому, що: - програми та апаратура представляють єдиний комплекс, в якому апаратними засобами можлива компенсація окремих відмов ПЗ, і досліджувати надійність отказоустойчивого ПЗ, ігноруючи цю обставину, безглуздо;

- існує великий клас апаратурних засобів контролю за помилками отказоустойчивого ПЗ, які багато в чому компенсують відмови ПЗ; - відмовостійке ПЗ може містити тільки помилки, не виявлені на етапі налагодження, а відмова системи - наслідок комплексного впливу відмов, збоїв і

помилок, що впливають на систему, оцінити яке теоретично не представляється можливим у зв'язку з невичерпним кількістю варіантів;

- не вирішена проблема робастности ПЗ - стійкості до різних наборів вхідних даних, яка робить безглуздою створення єдиної методології створення отказоустойчивого ПЗ.

Вихід з цієї групи проблем полягає в тому, що випробування проводяться для всієї апаратури комплексно за допомогою засобів імітації несправності, виділяючи ПЗ як окремий компонент системи. Моделювання процесів пристроїв інформаційних комплексів є одним з основних завдань, що вирішуються на початковому етапі проектування системи. При розробці стратегії управління інформаційними системами виникають значні труднощі, пов'язані з тим, що наявні в цій галузі міжнародні стандарти не містять алгоритму реалізації запропонованих вимог щодо забезпечення надійності та якості ПЗ для проектованих систем [3, 5].

Актуальною проблемою є гармонізації державних стандартів з міжнародними стандартами, створення єдиної термінології для опису проблемної області, що дозволить уникати неточностей і дублювання при створенні нових цифрових пристроїв [2, 6]. Тому розробка підходів до імітації несправності при проектуванні пристроїв інформаційного комплексу буде використовувати набір технологій, який надає допустиме рішення в предметної області за допомогою застосування системного аналізу. Таким чином, для забезпечення работоспобності прототипів мікропроцесорних пристроїв вимагає застосування системного аналізу.

1.2 Аналіз методів підвищення відмовостійкості мікропроцесорних пристроїв

Сучасні мікропроцесорні пристрої використовують надзвичайно стійкі технології проектування. Промислові архітектури проходять тестування на заводах-виробниках [2, 4]. Сьогодні в світі основними елементами цифрової

техніки є інтегральні польові мікросхеми, такі як програмовані масиви логічних елементів, мікропроцесори, різні елементи пам'яті і т.п. Основні виробники великих і надвеликих мікросхем представляють фірми Altera, Xilinx, Atmel, STM32 і ін. Всі виробники зобов'язані підтверджувати якість своєї продукції, зокрема показники надійності [5, 7]. Проте, для кожного мікропроцесорного модуля, що вийшов з конвеєра, можливе використання спільно з іншими мікросхемами і програмним забезпеченням. Сучасні мікропроцесори підтримують спеціальні алгоритми з відновлення працездатності після виникнення відмов і збоїв. Проектована пристрій на базі мікропроцесора відноситься до складних обчислювальних систем, оскільки складається з окремих елементів, що взаємодіють між собою і з навколишнім середовищем (прийом і передача інформації), а також виконують спеціальні функції. В системі при завантаженні програм в цільову апаратне середовище проявляється властивість емерджентності [7, 8], що фіксується за допомогою несподіваних відмов і збоїв. Для діагностики відмов мікропроцесорні архітектури містять спеціальний блок обробки відмов, який розрізняє такі типи ситуацій: 1. Важкі відмови (Hard Fault). 2. Відмови системи управління пам'яттю (MemManage Fault). 3. Відмови програми (Usage Fault). 4. Відмови шини (Bus Fault).

Мікропроцесорне ядро містить збереження контексту викликів інструкцій, тобто при виникненні відмови регістри процесора зберігаються, після чого проводиться детальний аналіз інструкцій і викликаються адрес програми, щоб встановити причину виникнення. Проте, існують ситуації, в яких відновлення контексту виконання мікропроцесора неможливо, тому використовуються спеціальні засоби діагностики, що включають апаратні блоки контролю і верифікації сигналів системних інтерфейсів. Відмінність апаратних відмов і збоїв від програмних є для мікропроцесорних пристроїв важливим завданням, оскільки прояв наслідків програмного відмови можливо у вигляді апаратної несправності.

У зв'язку з тим, що апаратні компоненти програмно-технічних систем відносяться до класу високонадійних виробів (відповідність стандарту ISO

9000), виявлення дефектів програмного забезпечення в апаратній середовищі є найбільш актуальним. Особливо з урахуванням того, що відомі методики випробувань програмного забезпечення для виявлення несправності в складних розподілених інформаційних комплексах недостатньо ефективні і вимагають удосконалення [2, 6]. Стійкість до відмов для мікропроцесорного пристрою розглядається спільно з інтегрованими мікросхемами і програмним забезпеченням, необхідним для його функціонування в інформаційному комплексі [7] і повинна відповідати рівню вимог для інформаційного комплексу, то для них пред'являються вимоги до живучості:

$$X^* = min\{x: \Phi_{\bar{x}} = 0\}, \qquad (1.1)$$

де X^* - мінімальне підмножина, $\bar{X}$ - підмножина зруйнованих елементів, X - додаткове підмножина, $\Phi_{\bar{x}}$ умовна ймовірність відмови. Властивість безпеки для роботи мікропроцесорного пристрою забезпечується за рахунок ряду обмежень, які спрямовані в рамках теорії ризиків.

$$\min_{\psi}\{C(\psi): R(\psi) \geq R_{required}, C(\psi) \geq C_{required}\}, \qquad (1.2)$$

де Ψ є конфігурація системи, C - її вартість, R - показник надійності, а S - показник безпеки.

Практично всі сучасні технічні пристрої є цифровими. Розвиток стандартів для цифрової техніки використовує технологічну базу імітації несправності для функціональної безпеки і аналізу критичності виконання режимів [3, 13].

Представлений ряд стандартів застосуємо і до мікропроцесорних систем, які також є частиною більш складних комплексів [19].

Синтез стандартів для цифрових систем, застосований до мікропроцесорних пристроїв, дозволяє виділити необхідні вимоги до проектування. Серед них працездатність і стійкість до відмов. Сформовані в рамках стандарту ISO 26262

пропозиції по розробці включають практично всі підходи розглянутих стандартів.

Оскільки стандарт ISO 26262 розглядає функціональне обладнання для безпеки дорожніх транспортних засобів, які включають мікропроцесорні системи, то даний стандарт застосовний для випробування мікропроцесорних систем інформаційних комплексів. Особливо за умови наявності в інформаційному комплексі транспортних засобів обміну інформацією, до яких вимоги стандарту є обов'язковими.

Рисунок 1.1 – Відповідність рівнів і процедур проектування циклу функціональної безпеки

У міжнародній практиці методика FMEA трактується як якісний метод аналізу. Але в стандарті ISO 26262-9 (п. 8.2) FMEA може бути використаний як кількісний метод аналізу для випадкових відмов апаратних засобів при наявності додаткових відомостей про кількісні значеннях інтенсивності відмов. При цьому

в стандарті не пояснюється, як саме. FMEA є базовим аналізом для проведення FMECA- і FMEDA-аналізів. FMECA і FMEDA є кількісними аналізами для оцінки безпеки і надійності відповідно. FMEDA - це досить новий стандарт, і в стару редакцію IEC 61508-7 він не включений. Імітація несправності (fault injection - FI) також присутній в IEC 61508-7 (пп. B6.10 і C5.6) як самостійний якісний (не кількісний) аналіз.

FMEDA використовується для оцінки системних відмов (за якими є статистична напрацювання), а FI - для аналізу випадкових (спорадичних) відмов. FMEDA використовується в першу чергу для покупних виробів.

Електронні компоненти, до яких можна застосувати ці методи, розглядаються як окремий випадок використання імовірнісних моделей типу «чорний ящик». Для розширення технології (FI) розробляється спеціалізована апаратура, наприклад, опромінювали певні ділянки друкованої плати сильної радіацією. Підключаються імітатори використовують інтерфейси системи або висновки елементів на монтажних платах. Вбудовані імітатори є елементом самої системи і не підлягають виключенню, тобто є не видаляється частиною досліджуваної системи і можуть використовуватися за іншим призначенням, крім FI. До вбудованих відносяться і програмні методи Software Implemented Fault Injection (SWIFI) [9]. Імітація помилок може здійснюватися чисто програмним способом або за допомогою апаратного забезпечення, яке в свою чергу ділиться на стандартне і спеціальне. До стандартного відносяться внутрісхемние емулятори та засоби для налагодження (JTAG-інтерфейс) [7].

Таким чином, створення імітаторів несправності, спеціалізованих для мікропроцесорних пристроїв інформаційних комплексів, є актуальним завданням, оскільки імітація відмов і збоїв дозволяє досліджувати проектовану систему і виявити помилки програм і апаратні дефекти на етапі макетування до виходу в серійну розробку, що підвищить якість кінцевого виробу.

Імітатори несправності для мікропроцесорних систем розділені на два великі класи: імітатори безпосередньо самої несправності і їх наслідків. За способом внесення поділяють програмні (software injection) і апаратні (hardware

injection) кошти. По розташуванню по відношенню до досліджуваної системі виділяють зовнішні, вбудовані і підключаються. Зовнішні використовують режими опромінення. Імітатор для мікропроцесорної системи повинен забезпечувати наступний набір функцій [2, 9]:

- забезпечення в період випробувань всіх вхідних сигналів системи, які будуть існувати, коли система проходить перевірку; - забезпечення виходів системи шляху, які адекватно представляють контрольоване обладнання; - забезпечення можливості управління входами для забезпечення будь-яких збурень, з якими повинна справлятися тестована система.

Стандарт IEC 61508 [13] визначає у імітаторів несправності такі функції, як перевірка та верифікація. Побудова імітаторів для мікропроцесорних пристроїв на базі серійно випускаються засобів обчислювальної техніки є актуальним завданням для нових поколінь засобів автоматизації, в рамках яких вирішуються завдання:

1. Проводиться аналіз прототипів автоматизованої системи випробувань (АСД) і розробляються узагальнені моделі таких систем, засновані на імітації несправності і імітації помилок.

2. Досліджується систематизація способів імітації помилок і механізмів їх реалізації, на основі яких проводиться цілеспрямований вибір цих способів при синтезі нових засобів імітації несправності.

3. Розробляється модель програмованих засобів імітації несправності для випробувань, що дозволяє автоматизувати настройку на тип імітованої несправності і тип несправного компонента, а також володіє властивостями безпеки і сопрягаемость.

4. Розробляються моделі помилок, що дозволяють обґрунтовано вибрати тип спотворень сигналів в системі.

5. Проводиться вибір способів імітації з урахуванням розробленої моделі помилок і нові механізми їх реалізації, що враховують специфіку об'єкта дослідження.

6. Розробляється мову для опису процесів імітації наслідків несправності, що дозволяє формалізувати процес проектування імітаторів несправності.

7. Здійснюється вибір узагальненої архітектури функціонально орієнтірованного процесора для імітації наслідків несправності, на основі якої пропонується інженерна методика для синтезу імітаторів несправності.

1.3 Аналіз засобів виявлення відмов і збоїв пристроїв інформаційного комплексу

Для виявлення відмов і збоїв пристроїв інформаційного комплексу застосовуються як вбудовані засоби складових пристроїв, так і аналіз статистики комплексу. Це дозволяє виявити вихід з ладу пристроїв комплексу. Варто відзначити, що набір заходів для забезпечення стійкості до відмов буде значно відрізнятися в залежності від сфери застосування комплексу. Для космічних і атомних систем застосовуються найбільш суворі методики перевірки на відмови і збої, ніж для інформаційних комплексів [7, 12].

Щоб забезпечити перевірку на відмови і збої стосовно пристроїв, що містить ряд мікропроцесорних пристроїв, потрібно визначити інформаційний портрет контрольованої системи. Проведення аналізу складних або багатофункціональних систем здійснюється за допомогою декількох методів аналізу. На практиці використання комбінацій спадного і висхідного аналізів є досить ефективним і дозволяє забезпечити комплексний розгляд для визначення показників аналізу існуючих систем. Інформаційний портрет представляє собою результат аналізу поведінки систем в умовах імітації відмов і збоїв, де проводиться збір і обробка статистичних даних, на основі яких пропонуються оцінки параметрів комплексу [2, 3, 5].

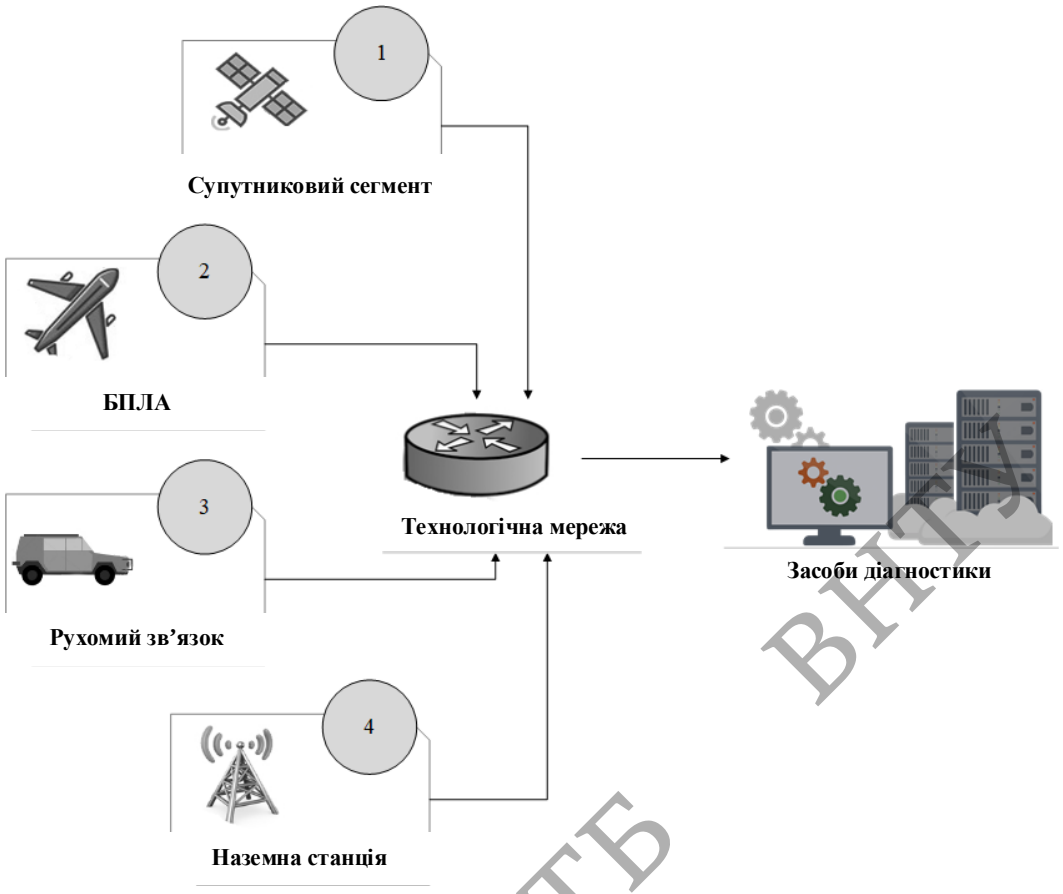


Рисунок 1.2 – Топологія програмно-апаратних компонентів мережі

Для оцінки параметрів в комплексі часто використовується статистичний аналіз за допомогою методів математичної статистики і машинного навчання.

Рисунок 1.3 – Візуалізація залежності ознак інформаційного комплексу

Статистичний аналіз умов виникнення подій використовується для встановлення кореляції між виходом пристроїв з ладу в процесі роботи [3]. В результаті моделювання встановлюється, які показники найбільш інформативні для встановлення непрацездатності конкретного вузла.

Для підтвердження апаратних відмов в інформаційному комплексі потрібне використання модулів апаратного контролю, що визначають технічні несправності компонент. Одним з ефективних підходів є порівняння сигналів на входах і виходах і параметричну регуляція модуля. При відхиленні модуля від встановленого порядку фіксується несправність, після чого інформація передається робочої станції. Застосування таких модулів актуально для основних підсистем мікроконтролера: системи управління внутрішньою пам'яттю, шин даних, введення-виведення. Для виявлення апаратних відмов потрібне використання модулів контролю, що визначають технічні несправності компонент. Відмови і збої визначаються шляхом порівняння сигналів на входах і виходах і параметричної регуляції модуля [5]. При відхиленні модуля від встановленого порядку фіксується несправність, після чого інформація передається робочої станції. Застосування таких модулів актуально для основних підсистем мікроконтролера: системи управління внутрішньою пам'яттю, шин даних, введення-виведення, апаратного таймера.

Використання імітації несправності для цілей тестування інформаційного комплексу є складним завданням і передбачає створення комплексу автоматизованих засобів, а також спеціального програмного забезпечення. Теоретично найбільшою точністю володіють експериментальні методи, які є основним способом підтвердження необхідних за технічним завданням показників працездатності із заданою точністю і достовірністю. Експериментальна оцінка показників може бути реалізована тільки після створення досліджуваної системи і є останнім і найважливішим етапом в процесі розробки інформаційних комплексів. Експериментальні методи оцінки надійності, пов'язані з організацією спеціальних випробувань, можна розділити на чотири групи [4].

1) Методи, які використовують природні фактори старіння при дослідженні систем, головний серед яких - час. При нормальних умовах експлуатації можуть бути досліджені порівняно недорогі системи часом напрацювання на відмову менше однієї тисячі годин. Для унікальних систем з часом напрацювання більш десяти тисяч годин методу першої групи абсолютно не підходять.

2) Методи, які використовують штучні фактори, що викликають старіння елементів, температура, напруга живлення, вологість, тиск, вібрація і т.п. Основна складність при випробуваннях пристрою полягає в тому, що практично неможливо підібрати параметри, при яких всі різнорідні компоненти системи прискорено старіли з близькими інтенсивностями. Внаслідок цього результати прискорених випробувань складного обладнання, яким є мікропроцесорний пристрій, завжди будуть відрізнятися від результатів природного старіння.

3) Методи, які використовують штучне введення несправності компонентів пристрою, засновані на руйнуванні структури компонентів системи або зв'язків з допомогою введення відмов і / або спотворенні сигналів в системі за допомогою генераторів перешкод - введення збоїв. Труднощі, пов'язані з введенням різних типів відмов і збоїв в реальному масштабі часу для великого числа компонентів, а також вихід з ладу обладнання системи роблять невигідним застосування даної групи методів.

4) Методи, які використовують імітацію несправності (наслідків несправності) компонентів пристрою, засновані на імітації відмов і збоїв шляхом спотворення сигналів в системі за допомогою спеціалізированнихустройств (імітаторів несправності), дозволяють точно визначити причину відмови системи, оскільки час і місце імітованої несправності відомо, а стан пристрою в момент імітації несправності може бути визначена або задано.

Таким чином з'являється можливість не тільки оцінити досягнуті показники стійкості пристрою до відмов, але і проводити спрямовану доопрацювання унікальних, дорогих систем з тим, щоб підвищити їх рівень надійності.

Процес імітації несправності полягає в тому, щоб внести несправності в реальній апаратурі, оцінити реакцію системи на внесені спотворення і зібрати статистичну інформацію про те, скільки разів система відмовить [2, 7]

$$p = \frac{\mathrm{n(t)}}{N}, \tag{1.3}$$

де p - ймовірність безвідмовної роботи, n(t) - число відмов за час t, N - загальне число відмов.

Відмови елементів системи могли не приводити до її відмови, якщо система переходила в працездатний стан за прийнятний час або приводити, якщо після відновлення працездатності досягалося тільки за рахунок діагностики і ремонту. Імітація несправності в реальній системі принципово модифікує кошти дослідження системи і покращує її якість на етапі здачі в експлуатацію.

1.4 Особливості розробки надійного програмного забезпечення

Сучасне програмне забезпечення вирішує все більше завдань, які раніше виконувалися із застосуванням апаратних засобів. Програмні компоненти використовуються практично у всіх окремих мікросхемах, які включені в мікропроцесорні модулі і підключені до них. На заміну пристроїв з поділом програм на різні апаратні модулі приходять мікропроцесори, що дозволяють виконати єдину програмну реалізацію алгоритмів і інтерфейсів, які повинні відповідати підвищеним вимогам стійкості до відмов. При збільшенні обсягу призначених для користувача програм відбулося зростання програмних помилок, що призвело до зміни і ускладнення методів і засобів налагодження, діагностики і верифікації [4, 6]. У 80-90-ті роки були спроби застосувати імовірнісні моделі для оцінювання надійності програмного забезпечення, помилок оператора, а потім і показників безпеки. Моделі надійності ПО використовуються для аналізу і пошуку дефектів програм в процесі розробки і

експлуатації. Практика розробки ПЗ передбачає пріоритет завдання забезпечення надійності над завданням її оцінки [5]. Ситуація виглядає парадоксально: абсолютно очевидно, що перш ніж забезпечувати надійність, слід навчитися її вимірювати. Але для цього потрібно мати практично прийнятну одиницю виміру надійності ПЗ і модель її розрахунку. Експериментально моделі оцінки надійності ПЗ відрізняються від обчислених значень. Проте надійність програм розглядається [5] як компонент якості програмного забезпечення. Застосування моделей розрахунку часто необгрунтовано для простих програм. Основне застосування для розрахунків на сьогоднішній день отримали моделі, які використовують структуру часу і емпіричні моделі. Моделі структури часу враховують поетапне тестування ПЗ із застосуванням оцінки за певний часовий проміжок. Програми для мікропроцесорних пристроїв відносяться до складного програмного забезпечення (критерій складності - число рядків програми) [10].

Підтвердженням практичної значущості емпіричних моделей є поява спеціальних засобів симуляції, спрямованих на практичну експлуатацію програм для цифрових електронних систем. Застосування даних моделей використовується сучасними САПР для розрахунку надійності ПЗ з використанням логіко-імовірнісних методів побудови графічних моделей безпеки, дерева подій і дерева відмов. Використання апарату математичної логіки в обчислення проводиться для великосерійних контролерів з малим об'ємом ПО, до яких застосовні формальні моделі надійності. В основному дані комплекси призначаються для великих об'єктів АСУ і промислових комплексів [6].

Тестування програм перетворилося на самостійну область дослідження, яка містить безліч методів і алгоритмів, що мають різну сферу застосування і ступінь інтеграції [6]. Більшість з них добре застосовні при моделюванні, але практичне застосування моделей залишається актуальним завданням. Це підтверджує різноманіття видів тестування програм: модульне, інтеграційне, мутаційні, регресійні, наскрізне, системне. Тестування використовує апаратні засоби для контролю, діагностики та імітації помилок. Так, процесори Intel використовують

технологію «Intel Packet Trace» [17] для налагодження програм спільно з апаратурою, мікропроцесорні системи стандартно надають доступ до регістрів, пам'яті і стека для аналізу стану програми на базі промислового інтерфейсу налагодження JTAG [7, 9].

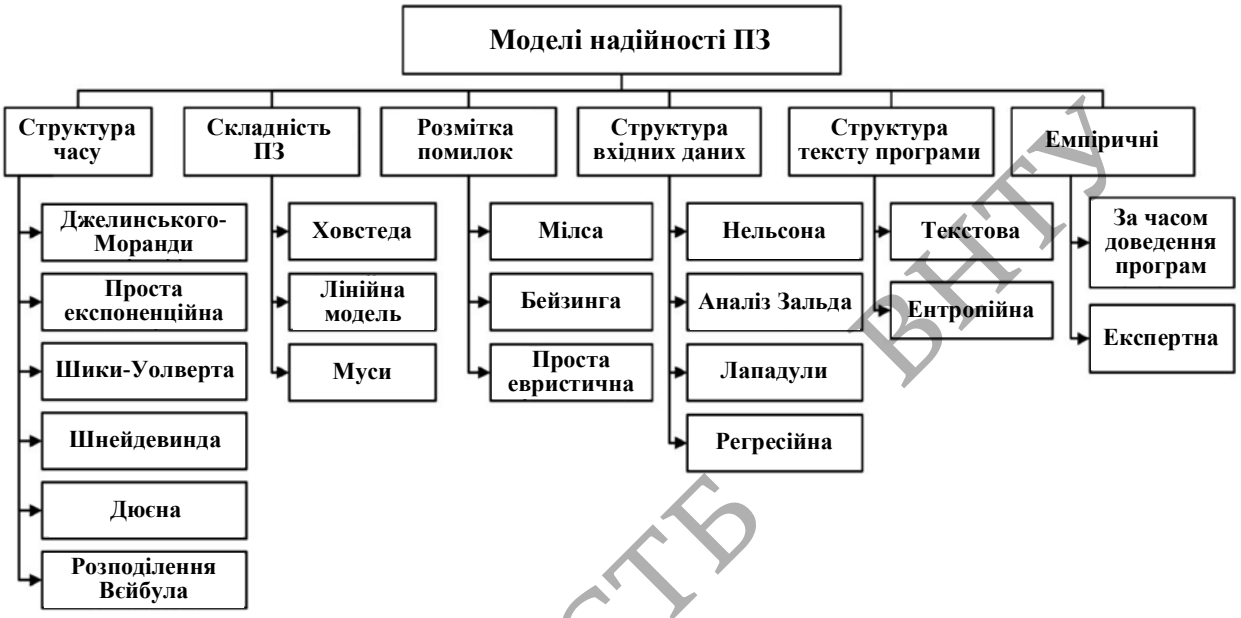


Рисунок 1.4 – Моделі надійності ПЗ

Однією з тенденцій розвитку програм можна відзначити приділення дизайну програм першочергового місця і появи спеціальної професії - архітектора програмного забезпечення, метою якого є мінімізація помилок програм за рахунок керування складністю взаємодії модулів в програмі і якісному покриттю програми тестами. Сучасні програми також використовують стандарти в галузі архітектури програмного забезпечення, що дозволяє комбінувати структурні елементи програм в блоки, які легше тестувати за рахунок застосування стилів, які представляють так звані «best practices» [37]. У сучасній розробці програм використовується ряд шаблонів архітектур програмного забезпечення. Для мікропроцесорних пристроїв частину архітектури є надлишковими або нецільовими, оскільки орієнтовані на веб-технології або серверне обладнання. Найбільш актуальне застосування концепції

«чистої архітектури», розробленої Робертом Мартіном [9]. Для спроектованої архітектури часто використовують визначення коефіцієнта надійності ПЗ [2]

$$R = \sum_{j=1}^{M} \sum_{i=1}^{N_j} PU_{ij} R_{ij} \,, \tag{1.4}$$

де M - число рівнів архітектури ПЗ, N_j - число компонентів на j-му рівні, j = 1,M, PU_{ij} - ймовірність використання компонента, R_{ij} - надійність компонента.

Розробка програм зі збільшенням обсягу кодової бази зазнає змін в плані спрощення використовуваних мов програмування. Це відбувається за допомогою конструювання нових предметно-специфіковані мов (DSL) [19, 20] і спрощення програмного коду для розробника. Проте, перехід до нових програмним мов містить ряд недоліків, які виявляються розробниками по всьому світу. Виправлення помилок мов є складним завданням, особливо з урахуванням наявності величезної кодової бази і працюючих програм. До того ж такі рішення мають проблемами, пов'язаними з точністю трансляції розроблених програм в асемблер архітектури мікропроцесорних модулів. Також можна відзначити помилки в компіляторах і інтерпретатора предметних мов [3], які спотворюють семантику програми.

Сучасні дослідники розглядають програмне забезпечення як динамічну систему, властивості якої до кінця не визначені. Тому методики і алгоритми ефективного дослідження програм і отримання практичних оцінок по експлуатації на реальних пристроях важливо для розробки якісного програмного забезпечення. Поняття якості ПЗ є багатоаспектним, дослідники не прийшли до єдиної терминологическому визначенням. Один з напрямків спрощення та систематизації розробки програмного забезпечення в частині зменшення числа програмних помилок проводиться за допомогою впровадження стандартів кодування. Існуючі стандарти JPL Institutional Coding Standard, MISRA, HICPP, Google C ++ Style Guide дозволяють зменшити число помилок за рахунок обмеження у використанні інструкцій для мови програмування [6], що дозволяє

допускати менше число помилок на стадії кодування, а не тестування. У число таких практик входить застосування при розробці програмного забезпечення контролю рівня об'єктно-орієнтованих абстракцій або рівня абстракцій предметної області. Поява технологій контролю версій (Git - система спільної розробки), автоматизованої інфраструктури віртуалізації (Docker - віртуалізація програмних компонент) спільно з методиками тестування, огляду коду, рефакторінга і статичного аналізу дозволяють вирішити більшу частину проблем при написанні коду [2, 9]. Проте розвиток засобів налагодження і діагностики для пристроїв інформаційних комплексів відбувається набагато повільніше в зв'язку з комерційними і спеціальними обмеженнями, а також використанням власних мікропроцесорних рішень для застосування в різних експлуатаційних умовах, де існують обмеження по енергоспоживанню і використовуваної пам'яті [5].

У зв'язку з чим використовуються мови 70-80-х років, такі як мова С, який має низку незаперечних переваг, але не містить модулів контролю і діагностики, як в мовах Python, C # і Java, внаслідок чого розробники за період своєї діяльності займаються написанням і налагодженням стандартних в інших мовах модулів або інтеграцією сторонніх бібліотек, які можуть містити помилки і вимагають додаткового часу на налагодження. Також запозичені рішення часто працюють не оптимальним чином, що змушує програміста пробувати нові і нові програмні бібліотеки, дотримуючись баланс між розумінням їх внутрішнього устрою і швидкістю розробки [3].

Тенденцією в області розробки програмного забезпечення є автоматизація процесів проектування на всіх етапах життєвого циклу програм. Автоматизація зачіпає процеси верифікації, тестування та контролю якості програмного коду. Сучасні сервіси спільної розробки підтримують процеси безперервної інтеграції та безперервного постачання програмного продукту (CI / CD) [7]. Проектування конструкції і випуск нових пристроїв триває більшу частину життєвого циклу проекту. У зв'язку з цим потрібно емулювати взаємодія з невиготовлення апаратними модулями. Таким чином, програміст працює з двома програмами -

для реального пристрою і для програмного емулятора. Реальне пристрій може виходити з ладу, тому необхідно підтримувати версії програмного забезпечення як для емулятора, так і для пристрою.

Процес розробки складається з наступних етапів життєвого циклу:

1. Розробка функціоналу модуля;

2. Об'єднання модулів в єдиній архітектурі;

3. Тестування функціональності компонента і системи в цілому;

4. Коригування програми в результаті апаратної коригування.

Для спільного використання версій продукту при розробці вбудованого ПЗ використовується система контролю версій. Головною проблемою розроблюваних програм є порушення алгоритмів роботи при апаратних корегування і відмовах. Чим раніше виявлена проблема, тим легше проводити коректування версії ПЗ. Для перевірки версії ПЗ і виявлення невідповідностей виконується збірка, тестування і коректування програм. Для вбудованих систем ці процедури не автоматизовані [5]. Для прискорення процесу розробки пропонується використання автоматизації проектування при виконанні наступного набору процедур: - автоматичне прибирання проекту; - автоматична перевірка проекту; - автоматичне програмування пристрою; - автоматичне тестування на базі тестової інфраструктури; - надання результатів тестів на відмови і збої системи.

Автоматизація складання проекту можлива при використанні сервера безперервної інтеграції (на базі засобів Docker). Для управління пропонується застосування систем управління репозиторіями (наприклад, Gitlab). На рисунку 1.6 наведено цикл оновлення версій програм за допомогою даного інструменту.

Для розробки ПЗ здійснюється автоматичне тестування на сервері і нагляду за виконанням Програми без участі розробника. При цьому на пошук помилок і корекцію змін йде менше часу в порівнянні з ручним пошуком, навіть з урахуванням часу проектування тестів.

Рисунок 1.5 – Основні стадії автоматичного оновлення версій програми

Розробка ПЗ при використанні даного середовища складається з наступного набору етапів:

- готується програмний образ для розробки проекту на сервері. Образ завантажується розробником на ПК, тому у всіх розробників одноманітна середовище проектування, доступна віддалено;

- всередині проекту існує два програмних продукти, які знаходяться в паралельних версіях проекту (для емулятора і для реального пристрою); - при розробці програми алгоритми перевіряються за допомогою модульних, інтеграційних і регресійних тестів. При завантаженні на сервер програма проходить набір тестів для реальної середовища і емулятора, при наявності помилок повертається на виправлення автору останніх змін.

Частина інтеграційних і навантажувальних тестів вимагає наявності у розробника спеціальної інфраструктури, що дозволяє виконати заміну об'єктів, взаємодія з якими необхідно для перевірки складних алгоритмів пристрою. В інфраструктуру входять спеціальне обладнання, яке працює в автоматичному режимі. Завдання таких пристроїв - генерація сигналів і емуляція протоколів для обміну з тестованим пристроєм. Оскільки більшість пристроїв має інтерфейсами підключення до ПК через USB або Ethernet, існує можливість програмного запуску цілої системи для імітації повноцінної взаємодії з реальним пристроєм.

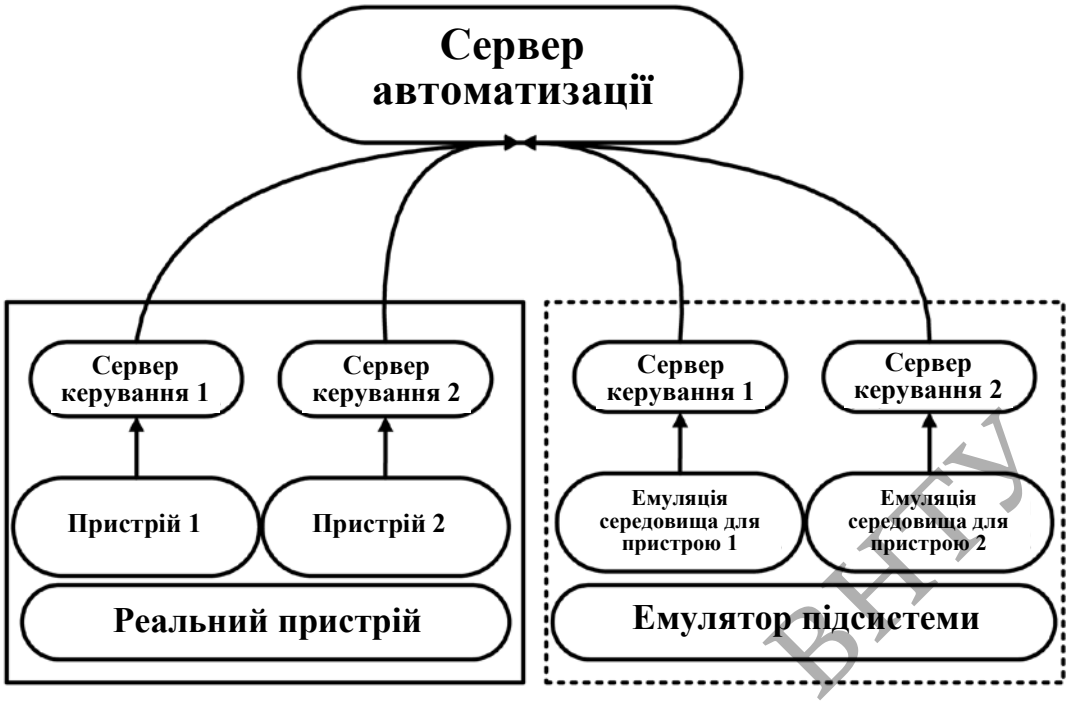


Рисунок 1.6 - Система автоматизованого тестування програм

Система тестування організовує виконання дій по запуску в автоматичному режимі. Запуск проводиться за допомогою етапів: - вибрати останню версію програми з пройденими модульними тестами, автоматично зібрати програми і за допомогою налагоджувального устаткування, що працює по інтерфейсу діагностики, завантажити версію програми в процесор вбудовується системи; - запустити підключене устаткування з сигнальними і кодовими послідовностями для інтеграційного і регресійного тестування; зберегти звіт про тестування для аналізу розробниками.

Розробка програм для сучасних пристроїв передбачає цикл роботи, якої виробляється протягом усього розробки, дослідної експлуатації і підтримки. Цикл розробки включає безпосередньо розробку архітектури, кодування ПЗ, налагодження, тестування, приймання, доопрацювання ПЗ. Дані етапи розробки варіюються в залежності від виду програмного продукту. Розробка програмного забезпечення для мікропроцесорних пристроїв має додаткові стадіями, які обумовлені наявністю апаратних засобів, які тестуються спільно з програмою. Для підвищення якості розроблюваного програмного забезпечення

використовуються гнучкі методології проектування, які націлені на мінімізацію часу існування невиявлених помилок програм.

На практиці дослідження програм добре зарекомендував підхід fault injection [7], тобто імітації відмов, який також реалізуємо за допомогою програмного забезпечення. Даний підхід використовує уявлення про те, що все критично важливі алгоритми роботи пристрою з програмним забезпеченням повинні бути максимально вивчені в обставинах, які вважаються позаштатними для функціонування пристрою. Одним з кращих варіантів організації ін'єкції несправності є техніка фаззінга [6, 7, 8], яка є різновидом мутаційного тестування. Фаззінга дозволяє тестувати програмне забезпечення в автоматичному чи напівавтоматичному режимі. Техніка полягає в передачі з додатком на вхід неправильних, несподіваних або випадкових даних. Предметом інтересу є падіння і зависання, порушення внутрішньої логіки і перевірок в коді програми, витоку пам'яті, викликані такими даними на вході. Поєднання даної техніки з використанням апаратних засобів для діагностики відмов і збоїв є актуальним для мікропроцесорних систем і вимагає розробки власних інструментів налагодження і контролю, в тому числі спеціального програмного забезпечення. В процесі фаззінга бере участь три основних компоненти: тестована програма, програма фаззінга, набір даних для проведення фаззінга. Набір даних являє собою вихідні дані у вигляді системних файлів, які будуть модифіковані в процесі роботи програми фаззінга і подані в тестируемую програму з різними видами спотворень (мутацій). Програма за допомогою інтегрованих модулів встановлює «зависання» і відмови програми та використовує «вдалі» для відмови вхідні дані як нові вхідні дані для мутації. Фаззінга включає кілька основних модулів [16]: 1. Модуль генерації даних; 2. Модуль алгоритму управління фаззінга; 3. Модуль внесення помилок; 4. Модуль діагностики помилок.

Залежно від типу системи, в якій використовується фаззінга, представлені модулі можуть бути розташовані на окремому пристрої тестування або інтегровані спільно з експлуатованої системою.

Готові рішення для фаззінга програмного забезпечення доступні для платформ загального призначення і відтворювальних пристроїв на базі мікропроцесорів. Для застосування фаззінга для вбудованих систем потрібно перепроектування та інтеграція модулів. Розглянемо класифікацію засобів організації тестування на основі фаззінга. Для проведення таких випробувань на сьогоднішній день не існує стандартів, тільки практики застосування в реальних проектах великих корпорацій, таких як Google, Facebook, Microsoft.

Застосовність фаззінга поширюється на всі види автоматизуються програмних тестів (модульні, інтеграційні, системні, графічні). Особливість застосування полягає в наданні програмами фаззінга точки входу в цільову систему або програму, що організовується на етапі опрацювання архітектури проектованого додатки [3, 8, 11, 14].

Однією з тенденцій останнього покоління програмного забезпечення є дослідження програмних засобів на потенційно вразливі місця з точки зору інформаційної безпеки, а також для обліку можливості фізичної перезапису програм або впливу на пристрої живлення, стійкості до електромагнітних атак використовуються спеціальні апаратні пристрої, що контролюють механізми безпеки функціонування програм при зміні умов функціонування апаратного середовища.

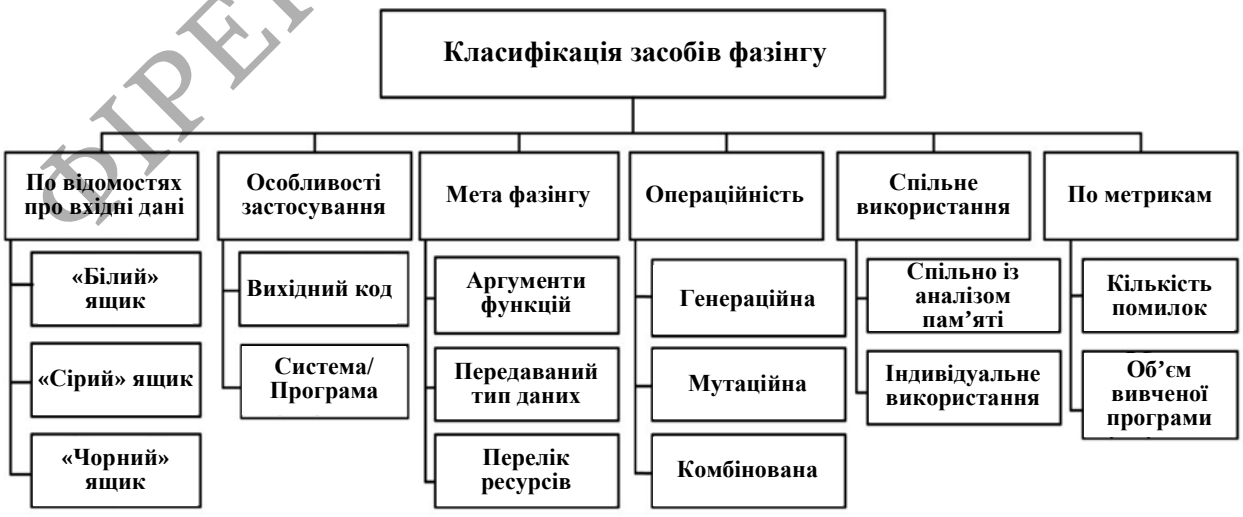


Рисунок 1.7 - Класифікація засобів фазінгу

Оскільки при розробці довірених програмних засобів мікропроцесорних пристроїв проектування здійснюється за допомогою засобів компіляції, бібліотек інтерфейсів і протоколів, які представляють собою величезну кодову базу, тому атестація і верифікація коду для зниження ризику прояви вразливостей є тенденційним завданням. Для верифікації властивостей програм і перевірки коректності функціонування використовуваних бібліотек, які застосовані при розробці програм мікропроцесорних пристроїв використовуються сучасні засоби статичного аналізу, що дозволяють визначити потенційно вразливі місця коду в процесі його проектування. Оскільки верифікація коду представляє окрему складну задачу, тому економічна доцільність проведення верифікації для мікропроцесорних пристроїв недоцільна. Тому розглядають різні інші дослідження програм, в тому числі із застосуванням моделі «чорний ящик» (таблиця 1.1) [14].

Для неперевірених бібліотек, драйверів і протоколів формується алгоритм тестування за методикою «сірого» або «чорного» ящика. Це означає, що властивості бібліотек вважаються невивченими і досліджуються в заданих станах виконання програми. Сформована діаграма переходу станів дозволяє оцінити коректність роботи тестованих модулів.

Таблиця 1.1 - Моделі тестування програми

Критерії	«Білий Ящик»	«Сірий ящик»	«Чорний ящик»
Рівень застосування	Модульне тестування	Інтеграційне тестування	Приймальне тествання
Відповідальність	Розробник	Розробник	Незалежний тестувальник/ Розробник
Знання реалізації	Обов'язково	Обов'язково для інтерфейсів	Не вимагається
Знання використання сценарію	Необхідно	Необхідно	Невідомо
Матеріали для сценаріїв тестів	Код для всієї системи	Код для інтерфейсів та окремих модулів	Специфікація

Проте, розробка програм мікропроцесорних пристроїв без урахування властивостей програми в частині базових зв'язків програмних компонентів і модулів призводить до підвищення складності семантичних і синтаксичних властивостей програми, що в свою чергу призводить до ризику появи помилок. Для зниження рівня складності програм використовується проектна архітектура, яка представляє організацію системи і описує зв'язки між компонентами цієї системи (а також зовнішнім середовищем).

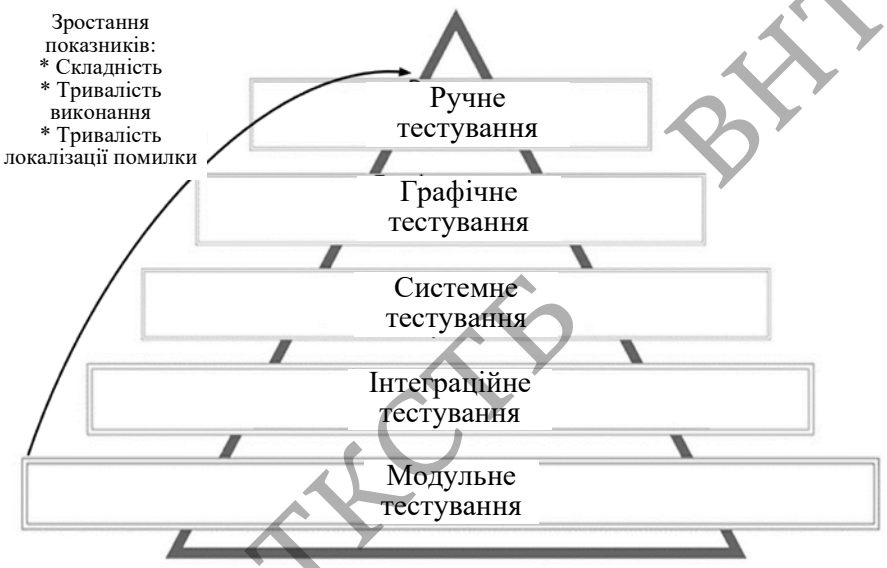


Рисунок 1.8 - Види тестування

Архітектура дозволяє визначити принципи проектування і розвитку програми [10, 17, 18]. Тестування мікропроцесорної програми представляє процес дослідження, випробування, що має на меті перевірку відповідності між реальною і очікуваною поведінкою програми на наборі тестів, розроблених з урахуванням вимог технічного завдання та алгоритмів роботи.

Для мікропроцесорних пристроїв проводяться всі типи тестування, при цьому організація інфраструктури тестування є завданням, що вимагає не тільки інженерного, а й наукового підходу. Тестування в залежності від циклу проектування програми використовує різні види.

Ресурсна вартість і час проведення тестування визначається в залежності від сфери застосування програмного продукту.

1.5 Висновки до розділу 1

В результаті аналізу робіт, присвячених питанням стійкості програмно-технічних комплексів до відмов, виявлено, що відомі методики виявлення несправності на основі регламентованих методик випробувань пристроїв недостатньо ефективні і вимагають удосконалення. При цьому існуючі галузеві стандарти вказують на необхідність здійснення імітації несправності при розробці (і в процесі макетування) мікропроцесорних пристроїв, але не визначають конкретних алгоритмів і необхідний комплекс технічних засобів для проведення випробувань.

Встановлено, що проектування унікальних і малосерійних пристроїв інформаційних комплексів, що мають складним для визначення або невідомі характеристики компонентів, пов'язане зі складнощами розрахунків традиційних показників надійності, тому потребує спеціально розробленому обладнанні для проведення експериментів для виявлення дефектів програмних і апаратних засобів.

Визначено, що відсутня гармонізація між вітчизняними та зарубіжними стандартами в галузі проектування цифрових пристроїв, включаючи аналіз видів і наслідків відмов. Це ускладнює процес проектування і встановлення характеристик працездатності пристроїв в зв'язку з необхідністю заміни якісних оцінок безпеки і надійності при роботі пристрою в реальних умовах (використовуваних у вітчизняних стандартах) на кількісні показники (використовувані зарубіжними стандартами). Виходом зі сформованої ситуації є застосування експериментальних досліджень, що включають імітацію несправності для оцінки працездатності програм і апаратури спільно.

Встановлено, що існуючі імітатори несправності для дослідження і забезпечення працездатності цифрових інформаційних систем, представлені на

комерційних ринках, є обладнанням, внутрішній устрій якого є захищається інтелектуальною власністю. У зв'язку з цим виникає необхідність створення власних коштів для імітації несправності і аналізу подальшої реакції мікропроцесорних пристроїв, що при проектуванні є єдиним варіантом для створення відмовостійких систем.

Встановлено, що в зв'язку з тим, що використовуються мікропроцесорні елементи апаратних засобів відносяться до класу високонадійних виробів, то основною проблемою забезпечення працездатності проектованих пристроїв є виявлення відмов, пов'язаних з дефектами програмного забезпечення, що їх виявляють в реальному апаратної середовищі. Для виявлення цих дефектів за доцільне розробка імітаторів несправності, які дозволяють досліджувати коректність роботи всіх програмних функцій в режимах експлуатації пристрою.

Визначено актуальність дослідження, яка обумовлена зростаючими вимогами до якості і надійності програмно-апаратних комплексів, а також відсутністю практичних рекомендацій і недостатній ефективності існуючих методів і засобів для проектування і налагодження мікроконтролерних пристроїв.

Встановлено доцільність розробки алгоритмів і технічних засобів, які забезпечують виявлення програмних і апаратних дефектів в інформаційних системах і можуть бути застосовані в різних областях вітчизняної промисловості.

2 ПРОГРАМНО-АПАРАТНИЙ ПРИСТРІЙ ТЕСТУВАННЯ ІНФОКОМУНІКАЦІЙНОЇ СИСТЕМИ

Другий розділ присвячено опису пристрою імітації відмов і збоїв.

Розглянуто склад програмного блоку для імітації несправності досліджуваного пристрою і спосіб імітації на основі спотворення вхідних даних. Розглянуто інтеграція пристрою в інформаційний комплекс і алгоритм діагностики для передачі інформації про знайдені несправності.

2.1 Вибір і обгрунтування режимів спотворення інформації для імітації відмов і збоїв мікропроцесорного пристрою

Для обгрунтування критеріїв і вибору режимів імітації несправності необхідний аналіз імітаторів несправності (таблиці 2.1 - 2.2). Імітатори несправності можливо класифікувати по ряду ознак, представлених в таблиці. Ефективність випробувань характеризується скороченням часу їх проведення, можливістю всебічних перевірок та отриманням найбільш достовірних результатів.

Таблиця 2.1 - Класифікація пристроїв імітації несправності

1. Призначення дослідження		2. Тип імітованої несправності	3. Ступін адекватності оцінки
Об'єкт	Ціль		
Покоління	FMEA Види та наслідки Відмов FMECA Ефекти режиму відмови та критичності FMEDA Диагнтика відмов Синтез та налагодження тестів комбінування	Поступові відмови	Точність оцінки (встановлено/не встановлено) Спосіб оцінки (експеримент/модель)
Сфера Використання		Постійні відмови	
Ступінь складності		Переміжні відмови	
Принцип дії		Пов'язані відмови	
Конструкція виконання		Збої	
Елементна база		Комбіновані відмови та збої	

Таблиця 2.2 - Класифікація пристроїв імітації несправності (продовження)

4.Тип впливу	5.Тип спотворення	6.Рівень імітації наслідків
Електричні сигнали	Випадкові	Транзисторні
Електромагнітний вплив		Вентильні
Лазерне випромінювання	Регульовані Аналогові/дискретні	Малі/середні/Надвеликі інтегральні схеми
Рентгенне випромінювання		Типовий елемент замін
Альфа-,бета- і гамма-випромінювання	Регульовані цифрові	Пристрій/блок/підсистема
Програмне спотворення	Комбінування	

Достовірність результатів, одержуваних в ході випробувань, є найбільш важливим критерієм при виборі засобів імітації та може бути забезпечена при адекватності процесу випробувань процесу експлуатації об'єкта випробувань (ОІ). Забезпечення адекватності можна розглядати з кількох точок зору:

- адекватність умов випробувань умов реальної роботи ОІ; - адекватність імітованих наслідків несправності реальним, що виникають в ОІ в процесі експлуатації; - адекватність станів ОІ в момент імітації несправності й станів найбільш вірогідним при реальній експлуатації.

Для класифікації способів імітації несправності представимо роботу мікропроцесорного пристрою наступним чином:

$$z = \psi(X, Y_{n,} T), \tag{2.1}$$

Вектор-шаблон результатів виконання функції формується при роботі пристрою в штатному режимі з контролем коректності виконання. Отримані значення порівнюються з реальним виконанням функцій пристрою при його тестуванні. При роботі пристрою в період виникнення відмов і збоїв функція пристрою і вектор-шаблон результатів не збігаються (за умови коректно отриманих значень вектора-шаблону). Для внесення спотворень можливо використовувати програмні і апаратні засоби. Апаратні засоби представляють технічні системи (контактні і безконтактні), які впливають на систему на фізичному рівні. Програмні засоби спотворюють значення пам'яті, регістрів в

ОЗУ і на інтерфейсах обміну з компонентами системи. Вибір аргументів або функції Ψ справного пристрою дозволяє визначити тип використовуваного імітатора несправності, який задовольняє вимогам до вартості і спрямованості пошуку в проектованій системі.

Для кожного компонента обрано варіант реалізації, який дозволяє здійснити імітацію на заданому рівні деталізації [2, 6]. Розглянемо опис конфігуратора вибору параметрів імітації несправності.

Найбільш поширеними способами імітації несправності є спотворення вхідних даних (вектор X), стану функціонування пристрою (вектор Yn) часу виконання функцій (T). Більш складним у реалізації та контролі є спотворення за допомогою виконуваної функції Ψ та (або) одного або кількох перерахованих аргументів (X, Yn, T).

Сучасні мікропроцесори, використовувані розробки цифрових пристроїв, ставляться до класу високонадійних виробів. Показники безвідмовності мікропроцесорів дуже високі (інтенсивність відмов становить одну відмову на сто мільйонів приладгодин). Для пристроїв, що не відновлюються, напрацювання на відмову T визначається наступним чином [5, 6]:

$$T = exp \int_0^\infty p(t)dt, \tag{2.2}$$

де p (t) - Можливість безвідмовної роботи; t – час роботи пристрою.

Проте обчислення напрацювання на відмову для пристроїв із власним програмним забезпеченням не передбачено. Проблема надійності ПЗ та поєднання мікропроцесорних модулів з набором програм в єдину систему збільшують ймовірність прояву якості емерджентності, що призводить до несподіваних відмов та збоїв [1, 4, 8].

Для забезпечення працездатності пристрою відхилення між вихідною функцією та функцією-шаблоном має бути мінімальним для виконання основних функцій у режимах роботи мікропроцесорного пристрою:

$$\sum_{i=1}^{M}[z_i - z_i^*] \to min, \quad (2.3)$$

де M – число порівнянь ідеальної та реальної функції під час проведення тестування. Якщо розглянути рівняння Z як рівняння від однієї змінної, інші аргументи будуть представляти набір параметрів. Для визначення цих параметрів потрібно вирішити систему рівнянь:

$$\begin{cases} \sum_{i=1}^{M}[Z_i - \psi(x_1, y_n, t)](\frac{d\psi}{dy_n})_i = 0 \\ \sum_{i=1}^{M}[Z_i - \psi(x_1, y_n, t)](\frac{d\psi}{dt})_i = 0 \end{cases}, \quad (2.4)$$

Вирішити систему рівнянь у загальній формі не можна [6] (для цього необхідно знати конкретний вид функції Ψ), тому потрібне проведення експериментальних досліджень. При прояві емерджентних властивостей під час роботи пристрою вид функції може змінюватися [4, 5], тобто. вона має складновизначуваний вигляд, що залежить від часу. Непостійність виду функції Ψ також пов'язане зі специфікою мікросхем, що застосовуються, і сприяє прояву відмов і збоїв.

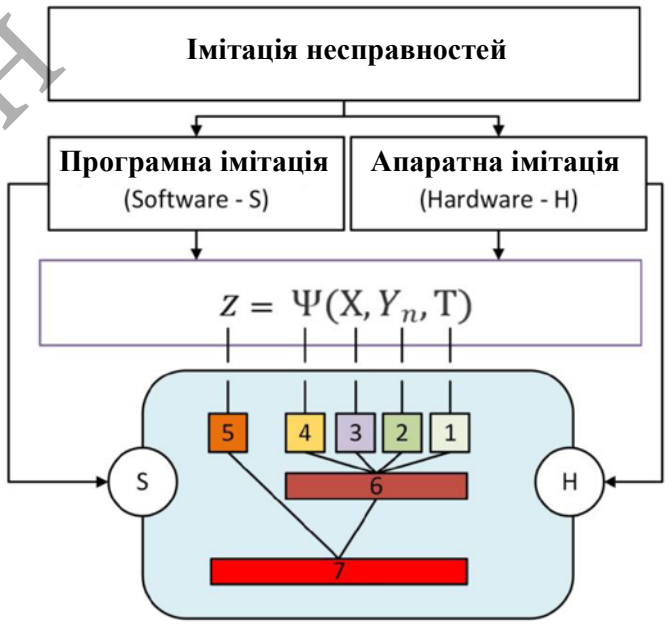


Рисунок 2.1 - Загальна структура імітатора несправності

Наприклад, дослідження станів регістрів для мікропроцесорного модуля Atmega128 показало наявність недокументованих функцій мікропроцесора [7], таких як імітація «зависання», несанкціонована передача даних, зміна часу характеристик мікросхем з виходом межі, зазначені в технічної документації.

Усі імітатори несправності можна класифікувати з допомогою запропонованих раніше способів спотворення проектованої системи тестування інформаційного комплексу [7, 8].

Оскільки експлуатація пристрою проводиться для інформаційного комплексу, потрібно враховувати особливості вибору імітатора несправності для спільного використання у складі комплексу. Для визначення причини відмови або збою пристроїв програмно-апаратного комплексу через високий рівень зв'язності може бути утруднена. На даний момент не існує єдиної системи класифікації несправності для комплексу, що застосовується для пошуку відмов та збоїв інформаційних систем [3] та визначення стійкості до відмови окремих вузлів (мікропроцесорних пристроїв). Оскільки комплекс містить безліч пристроїв і існують ймовірні події, які можуть бути причиною помилкових відмов і збоїв, то при діагностиці комплексу враховують всі відомі події [3, 4, 5], формуючи показник ентропії спільних подій:

$$G(X) = -\sum_{i=1}^{M} p_i \sum_{j=1}^{M} p_i \log_2 p_i(j), \qquad (2.5)$$

де і - стан, що залежить від попереднього вибору вузла; pi - ймовірність оптимального значення параметра; pi (j) – можливість вибору j-го за умови попереднього вибору і-го вузла.

Існуючі класифікації використовують або опис моделі програмних дефектів системи, що проектується, або спираються на введення несправності шляхом модифікації пам'яті випробуваного пристрою. Класифікація з впровадження несправності не розглядає виникнення повноцінних відмов і використовує лише інструмент модифікації пам'яті чи гібридні інструменти, які ефективні спеціалізованих архітектур пристроїв, але з адаптовані під

мікропроцесорні устрою. Таким чином, програмні та апаратні збої/відмови практично не перетинаються в системах моделювання та розглядаються на окремих рівнях, що значно знижує надійність системи та ускладнює діагностику [7]. Тому для проектування стійких до відмов та збоїв мікропроцесорних пристроїв у складі комплексу потрібно забезпечити ефективність використання імітації несправності [15, 18], за якої враховуються значні причини виникнення відмов та збоїв. Найбільш простий спосіб імітації представляє використання спотворення вхідних даних, оскільки інформаційний комплекс реалізує процедури прийому та передачі і максимально вразливий саме у процесі обміну інформацією. Прийнята класифікація типів несправності дозволяє визначити заходи, необхідних усунення кожного виду відмови, оскільки частина збоїв не вносить у систему критичних змін (рис. 2.3).

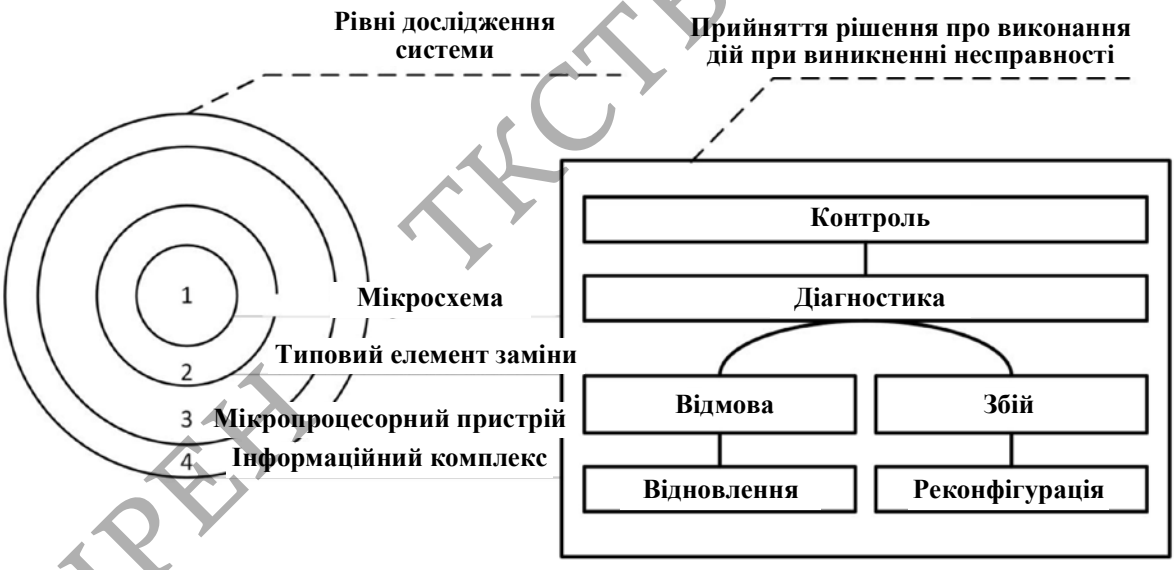


Рисунок 2.2 - Прийняття рішень для пошуку та усунення відмов та збоїв

Ієрархічний опис проектованої системи можна у вигляді кількох рівнів – лише на рівні мікросхем, типових елементів заміни, мікропроцесорного пристрою чи інформаційного комплексу. На всіх рівнях опису діють апаратно-програмні механізми контролю, діагностування, ідентифікації, реконфігурації та відновлення обчислювального процесу. Складність аналізу вже розробленого

пристрою полягає в тому, що розробник може не мати повної та достовірної інформації про всі контури контролю, діагностування, ідентифікації, відновлення та реконфігурації на всіх рівнях опису системи.

Для виправлення знайдених проблем пропонується використовувати програмні способи блокування вибраного типу несправності. Імітація використовується для отримання інформації та запобігання частині відмов і збоїв, які вимагають наявності додаткового програмного коду в залежності від типу відмови або збою, що виправляється. Під час розробки пристроїв комплексної системи види реакцій вибираються з вимог технічного завдання [70]. Для реалізації процедур контролю та діагностики несправності програмно-апаратного комплексу створено тестове середовище, що дозволяє здійснювати імітацію відмов. Відповідно до введених для мікропроцесорних пристроїв інформаційного комплексу вимог доцільно дослідити стійкість до відмов і збоїв за допомогою способу спотворення вхідних даних (вектор X), оскільки даний вид імітації сприяє виявленню помилок ПЗ та апаратних дефектів і легко інтегрується в існуючі інформаційні комплекси.

2.2 Розробка модуля ін'єкції несправності у складі програмного блоку імітації відмов та збоїв

Для аналізу засобів автоматизації випробувань на надійність використовується функціонально-структурний підхід, в основі якого лежить ідея про визначальну роль функції у діалектичному взаємозв'язку функції та структури. Вибір даного підходу пов'язаний також з якістю аналізованого матеріалу, в якому практично відсутній опис структур засобів випробувань і в той же час є опис функцій з різним ступенем деталізації. Визначення набору функцій дозволить надалі розробити нові кошти реалізації, багатофункціональні чи спеціалізовані, з урахуванням всіх вимог, що висуваються до процесу автоматизації випробувань [2, 5, 9].

Функціональна організація системи є модель системи, побудовану з урахуванням функціональних елементів і відбиває основні функціональні зв'язок між ними. Основу опису функціонально-структурної організації системи становить дерево функцій, адекватне за своєю структурою ієрархічної організації системи та розкриває багаторівневе уявлення функцій системи. Процес аналізу систем включає виявлення основних та додаткових функцій систем-прототипів, побудова узагальненого дерева функцій, виявлення базових структур та аналіз реалізації функцій у структурах.

Першою функцією, явно або неявно присутньою у всіх прототипах імітаторів несправності, є функція планування експерименту, відповідно до якої задаються типи несправності, типи компонентів, що відмовили в об'єкті випробувань і умови імітації – стан ОІ.

Друга функція надає можливість синхронізації моментів імітації несправності до певного стану пристрою – етапу, фазі роботи програми та забезпечує можливість всебічних перевірок. Наявність цієї функції пов'язана з тим, що реакція мікропроцесорного пристрою залежатиме від його стану в момент імітації несправності [3, 4].

Третя функція імітації несправності є основою автоматизації випробувань на стійкість до відмов та збоїв. Для реалізації даної функції у відомих системах використовувалися два місця підключення до ОІ: висновки мікросхем та інтерфейси.

Четвертою функцією є функція аналіз реакції пристрою на проімітовану несправність і є в описі трьох систем автоматизації випробувань на стійкість до відмов та збоїв.

П'ята функція представляє функцію збору статистики за результатами експериментів із запровадженням несправності ОІ та обчислення кількісних характеристик надійності за відомими положеннями математичної статистики. Базовий склад, що лежить в основі автоматизації випробувань, включає вісім функцій [1, 12]:

1) Ф1 - завдання типу компонента, що відмовив у пристрої;

2) Ф2 - завдання типу несправності компонента, що відмовив;

3) Ф3 - завдання умов імітації несправності стану пристрою;

4) Ф4 - виділення заданого стану пристрою (синхронізація);

5) Ф5 – імітація несправності компонента у пристрої;

6) Ф6 – аналіз реакції устрою на проімітовану несправність;

7) Ф7 – набір статистики за наслідками експериментів;

8) Ф8 - обчислення кількісних характеристик надійності пристрою.

У проаналізованих джерелах [2, 7, 8] існує принаймні два підходи до процесу імітації. Перший у тому, що імітується сама несправність, наприклад, коротке замикання виходу (входу) мікросхеми землі. Цей підхід дуже обмежений через зростання ступеня інтеграції компонентів та обмеження доступу до них. Другий підхід ґрунтується на імітації наслідків несправності – помилок, що виникають унаслідок зміни структури компонентів відмов чи збоїв. При реалізації другого підходу функція синхронізації доповнюється виділенням умов прояву несправності в місці імітації, оскільки несправність може бути потенційною (якщо компонент не бере участі у виконанні функції) і діє, коли робота несправного компонента призводить до появи помилки, яка імітується в пристрої. Умови прояву несправності фактично визначають момент переходу несправності з потенційної чинної. Реалізація другого підходу та імітації несправності призводить до необхідності визначення умов прояву несправності та розробки моделей помилок – наслідків несправності компонентів ОІ, що імітуються до функцій Ф5. Базовий склад функцій, що лежить в основі другого підходу, повинен включати: 1) Ф1 - завдання типу компонента, що відмовив у пристрої; 2) Ф2 - завдання типу несправності компонента, що відмовив; 3) Ф2.1 – визначення наслідків несправності компонента у місці імітації визначення помилки; 4) Ф2.2 – визначення умов вияву несправності у місці імітації; 5) Ф3 - завдання умов імітації несправності, тобто стану пристрою;

6) Ф4 - виділення заданого стану пристрою (синхронізація); 7) Ф5 - виділення умов прояву несправності компонента у точці імітації несправності; 8) Ф5 – імітація наслідків несправності; 9) Ф6 – аналіз реакції ОІ на проімітовану

несправність; 10) Ф7 – набір статистики за наслідками експериментів; 11) Ф8 – обчислення кількісних показників надійності устройства.

Узагальнені структури, що відображають взаємозв'язки функціональних компонентів у процесі дослідження, представлені відповідно для першого та другого варіантів імітації несправності. Функції Ф3 - Ф5 становлять основу автоматизації випробувань, реалізація Ф5 представляє основне завдання справжньої роботи, тому подальшу декомпозицію проведемо для цих функцій.

Процес виділення певного стану пристрою пов'язаний зі збором інформації про систему та порівнянням зібраної інформації з певним еталоном, інформаційним портретом у момент імітації несправності, тому функція Ф4 може бути поділена на дві: 1) Ф4.1 – збір інформації про стан апаратурних та програмних компонентів пристрою ; 2) Ф4.2 – порівняння зібраної інформації із заданим стандартом.

При реалізації функції імітації за другим варіантом можлива декомпозиція Ф5 наступні складові: 1) Ф5.1 – збір інформації про стан компонента, несправність якого імітується; 2) Ф5.2 - порівняння зібраної інформації з еталоном, що визначає перехід несправності з потенційної в діючу.

Імітація наслідків несправності Ф5 полягає у спотворенні сигналів у системі, еквівалентному наслідкам несправності (імітація помилки). Функція аналізу реакції пристрою багато в чому аналогічна функції синхронізації, різниця полягає лише в тому, що при синхронізації необхідно виділити один стан ОІ, а при аналізі реакції виділяються певні множини станів ОІ, що свідчать про його відмову або збій. У зв'язку з цим постає завдання стиснення вихідного обсягу інформації, яку пропонується вирішувати трьома способами: попереднім виділенням критеріїв відмови ОІ та реєстрацією в ОІ невеликого обсягу інформації; стисненням інформації про стан ОІ та подальшим її аналізом, комбінацією перших двох.

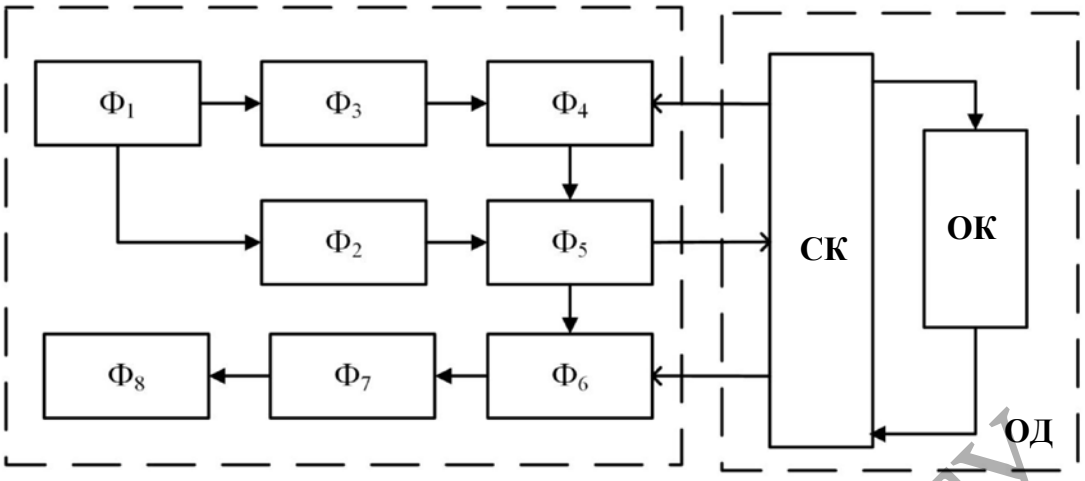


Рисунок 2.3 - Модель АСІ на основі імітації несправності

Кожен із цих способів має свої переваги та недоліки.

Декомпозиція Ф6 може бути проведена наступним чином: 1) Ф6.1 - збір інформації про стан апаратурних та програмних компонентів пристрою; 2) Ф6.2 – стиск вихідного інформаційного масиву; 3) Ф6.3 - визначення відмов та збоїв пристрою.

Питання реалізації інших функцій, що забезпечують процес автоматизації випробувань, розглядатимуться лише тією мірою, якою це необхідно для аналізу та синтезу засобів імітації несправності. Проведений аналіз дозволив виділити набір функцій, що лежить в основі автоматизації випробувань, на стійкість до відмов та збоїв та детально розглянути функції, що покладалися на засоби імітації несправності. Імітація відмов та збоїв для мікроконтролерного пристрою має специфіку, пов'язану з вибором необхідних компонентів серед усіх методів і засобів, що використовуються для проектування пристроїв з мікропроцесорними модулями, тому комплекс заходів заснований на застосуванні набору інструментів, які можна виділити в спеціалізовану архітектуру.

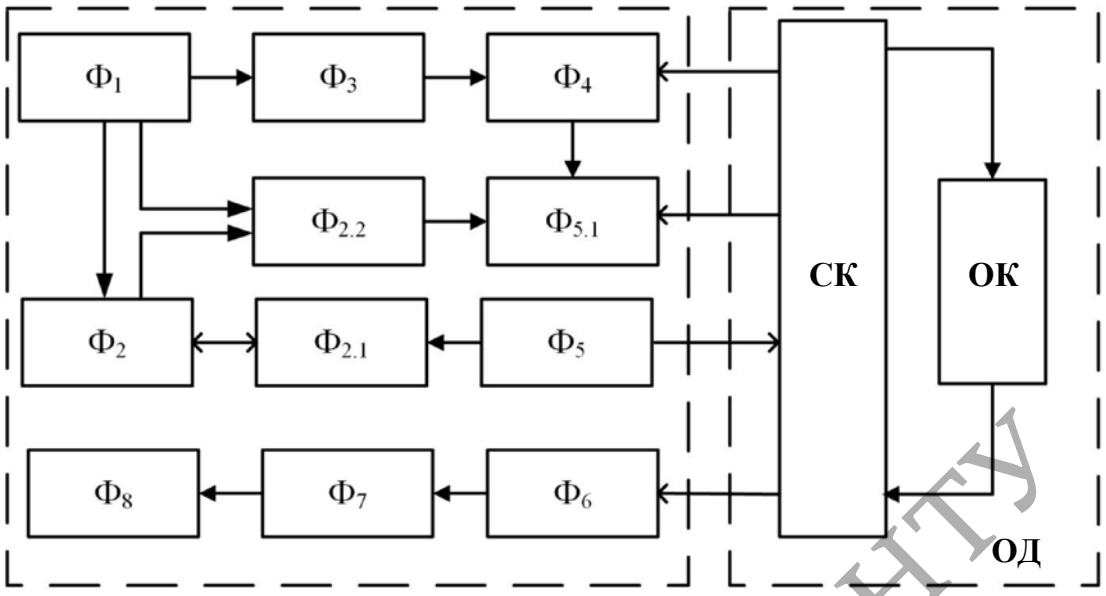


Рисунок 2.4 - Модель АСІ на основі імітації наслідків несправності

Виділимо основні рівні імітації несправності [3, 7]: − імітація на рівні інтерфейсу – вплив на шину даних мікроконтролера, обумовлений втратами та спотвореннями в отриманій інформації або впливом на процесор; − імітація на рівні процесора – виконання інструкцій процесора, які є небезпечними для надійної роботи;

− імітація на рівні програми – виконання програм, які можуть призвести до відмов процесора та інтерфейсів, логічних та апаратних відмов; − імітація на рівні введення-виведення – подання непередбачених даних алгоритмами обробки введення-виведення. Для пристрою пропонується наступний варіант архітектури автоматизованого тестування на відмови та збої, який відповідає стандартам IEC 61508 та EN 50128 [19]: архітектура складається з двох підсистем - об'єкта управління (випробуваний блок) та керуючої системи (станції управління - СУ), націленої на імітацію та підготовку результатів пошуку несправності. На рисунку 2.6 представлена функціональна схема для оцінки наслідків імітації відмов та збоїв у досліджуваній системі.

Випробуваний блок є сукупність тестованих мікропроцесорів із загальним входом і загальним виходом (їх кількість дорівнює N). Там виконується тестова

програма, яка обробляє вхідну послідовність даних і виконує ряд дій, необхідний виявлення несправності у межах тесту. Дані про результати знімаються за допомогою пристрою зовнішнього контролю та записуються до бази даних (БД).

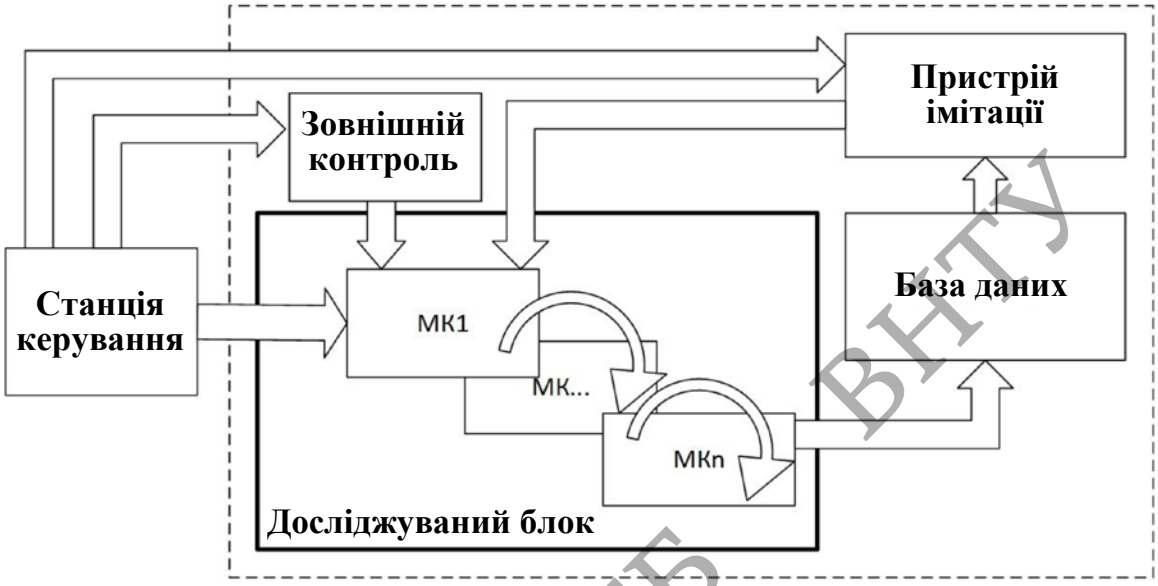


Рисунок 2.5 - Архітектура системи автоматизованого тестування

Блок пристрою імітації (ПІ) являє собою пристрій з працюючою програмою класифікації ознак сигнатур, що вибирає дані з блоку СУ і зберігає на цьому блоці інформацію про подану на вихід блоку послідовності з влучного часу.

Блок зовнішнього контролю здійснює контроль частин випробуваного блоку, список яких коригується з блоку станції імітації та пов'язаний з блоком УІ, яке за командами із СУ (реалізованим у вигляді ПК) здійснює запуск тестових послідовностей. СІ є станцією імітації, яка організує установку сигналів всередині випробуваного блоку. Окремо слід розглянути технічні засоби для імітації несправності. Як такий блок використовуються апаратні пристрої з підтримкою JTAG-інтерфейсу, які є стандартними для більшості мікроконтролерів архітектур [5, 6] DSP, ARM, AVR, MIPS. Класичне тестування системи згідно з класифікацією ГОСТа Р 51904-2002 «Програмне забезпечення

вбудованих систем» здійснюється для пошуку помилок програмного та апаратного забезпечення спільно.

Інтерфейс JTAG, стандартний для мікропроцесорних пристроїв, дозволяє отримати доступ до процесора та інтерфейсів заданого мікроконтролера. Для виконання заходів із тестування, описаних у стандарті IEC 61508, алгоритми ін'єкції можуть бути реалізовані в рамках представленої системи на станції управління та пристрої імітації.

Таблиця 2.3 - Організація класичного тестування системи

Тип ядра	Вид тесту	Спосіб організації тесту на процесорі
ARM-,MIPS-процесори	Модульний	Перевірка автомата стану контролера і окремих функцій шляхом генеації вхідних значень через віддалений інтерфейс ПК
	Інтеграційний	Імітація сигналів інтерфесів з допомогою FPGA в режимі трансляції і перехоплення сигналів системи
	Стрес-тест	Запуск процедур генерації сигналів на FPGA зі збільшенням швидкості роботи до фіксації відмови алгоритму
DSP-процесори	Модульний	Складання дерева викликів функцій та списку рішень , повне попкриття звітами викликів з передачею на ПК. Аналіз звітів на ПК за допомогою спеціальних алгортмів
	Інтеграційний	Взаємодія з САПР на ПК через загальний буфер обміну для генерації короляційної функції розраховуючих сигналів
	Стрес-тест	Імітація з допомогою побудови екосистеми на генераторі сигналів керуючого сервером непреривної інтеграції
FPGA	Модульний	Верефікація з допомогою вбудованих засобів «testBench»
	Інтеграційний	Перевірка разом з процесорами ARM, DSP Та їх стимуляції
	Стрес-тест	Управління реконфігурації алгоритмів FPGA з можливістю виводу результатів роботи на ПК

У цій роботі проведення випробувань проводиться з використанням програмного блоку імітації несправності за допомогою спеціально спроектованого пристрою імітації несправності, заснованого на використанні

засобів налагодження мікропроцесорних систем (діагностика за інтерфейсом JTAG та аналіз реакції).

Оскільки подання у стандартах технічної реалізації тестування не наведено, то розробникам доводиться використовувати власні напрацювання та технічні можливості проектованих мікроконтролерів, мікропроцесорів та мікросхем для пристроїв [4, 8].

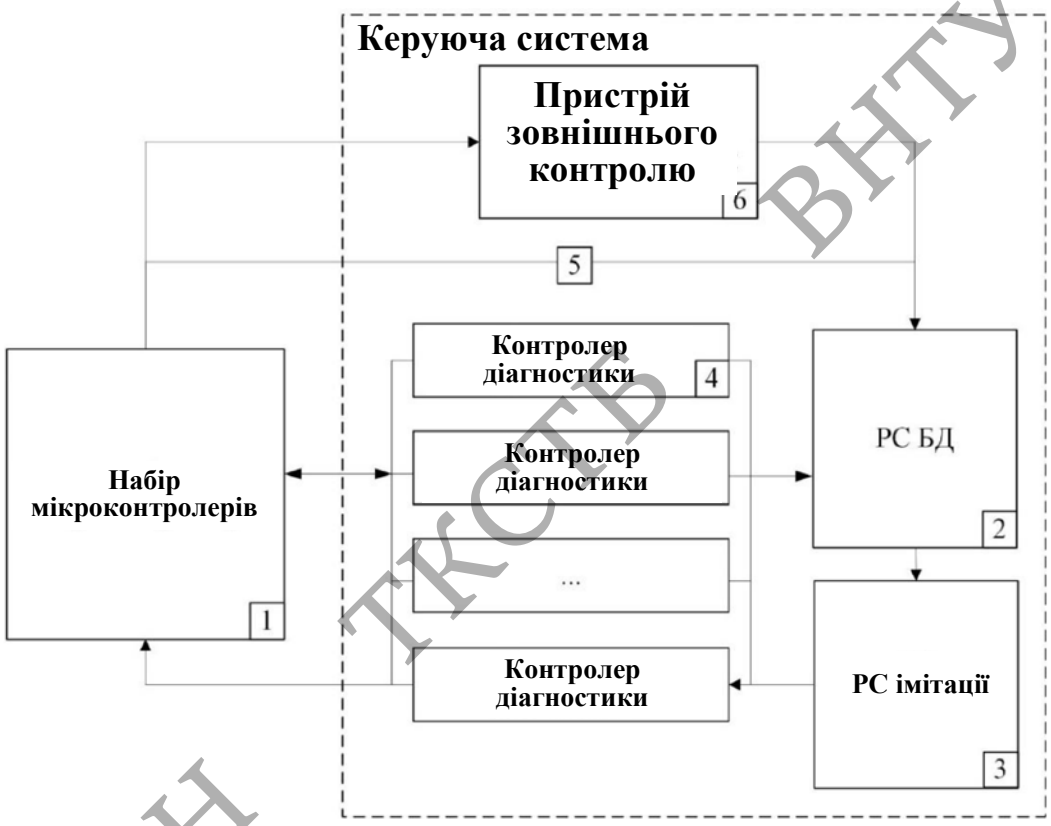


Рисунок 2.6 – Структурна схема пристрою імітації несправності та аналізу реакції

Пристрій описує процес проведення випробувань пристрою з програмним забезпеченням, що безперервно виконується, здійснюваний в автоматичному режимі. Застосування пристрою дозволяє скоротити час виявлення несправності, автоматизувати їх пошук за рахунок аналізу регістрів і ОЗП на наявність відхилень. Керуюча система здійснює експлуатацію програмно-апаратного комплексу (система, що тестується). Для моделювання станів пристроїв

використовуються гібридні моделі, в яких гібридні автомати розглядаються як особливий клас динамічних систем з безперервним часом, що викликано насамперед спільним функціонуванням безперервних та дискретних об'єктів і характерно для цифрових систем керування, в яких присутній безперервний об'єкт керування та дискретний пристрій керування (контролер). Гібридним автоматом у сучасній інтерпретації прийнято називати сукупність [2, 12]:

$$H = \{S, X, f, Start, Inv, E, G, R\}, \qquad (2.6)$$

Для дослідження графа програми використовується матриця суміжності, яка дозволяє зрозуміти густину мережі А, яка характеризує зв'язність переходів між станами програми [5, 18]

$$A = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} x_{ij}, \qquad (2.7)$$

Теорія гібридних автоматів та представлення програми у вигляді графа застосовується для організації програмного блоку ін'єкції в мікропроцесорному пристрої. Подання програми пристрою як спрямованого графа дуже часто використовується для діагностики його станів і дозволяє здійснювати точкове внесення спотворень даних [5, 7, 8]. Вершинами графа є інструкції розгалужень, які передачу управління між функціями програми.

2.3 Розробка модуля виявлення несправності у складі інформаційного комплексу

Для виявлення несправності в інформаційному комплексі використовується статистика обміну даними про функціонування в режимах штатної роботи (прийому та передачі) мікропроцесорних пристроїв. Для цього використовуються відомі алгоритми ранжирування та обробки ознак

(математичної статистики та теорії ймовірності, дерева прийняття рішень, нечіткого логічного висновку). Застосування статистичних алгоритмів дозволяє класифікувати набір параметрів їх важливості задля досягнення цілей обміну інформацією між вузлами комплексу.

Наведені результати аналізу відмов, що моделюються, і збоїв, які розділені за класами об'єктів, для яких за статистичними даними був виявлений відмова або збій. Для швидкого визначення відмови чи збою під час роботи мікропроцесорного пристрою виникає завдання розробки алгоритму швидкого пошуку шляху між вузлами, враховує вищеперелічені особливості комплексу зв'язку передачі діагностичної інформації [5, 13].

Відбір ознак формування статистики є важливим етапом отримання автоматичного виявлення ознак виникнення у мікропроцесорних пристроях відмов і збоїв.

Рисунок 2.8 - Комплекс зв'язку, що маршрутизується

У разі ознаки були сформовані з урахуванням експертної оцінки параметрів системи, які можна вилучити при маршрутизації. Як ознаки для визначення коректної роботи пристрою у складі інформаційної системи, які братимуть участь у формуванні статистики за допомогою технологічної мережі, були обрані наступні параметри [3, 4, 7, 18].

Дослідження даних для інформаційного комплексу показує, що ознаки виникають діляться приблизно на 5-7 великих груп, зібраних за відмітними ознаками. Наведемо таблицю передач (табл. 2.2) між пристроями інформаційної системи, моделювання якої виконано заданого набору векторів даних.

Таблиця 2.4 - Збір статистики в інформаційному комплексі під час моделювання

Моделювання обміну						
Результат	Передана	Прийнята	Обмін данних	№ Інт.	Погода	Адреса пам'яті
0	9	7	Середній	9	Хороша	135121239
1	6	5	Малий	3	Хороша	134258225
1	8	6	Малий	7	Погана	135153655
0	7	5	Середній	7	Хороша	134589502
1	2	8	Середній	5	Погана	135071493
1	7	1	Малий	6	Хороша	134277113
1	2	8	Малий	8	Погана	134534935
1	5	5	Середній	2	Хороша	135000506
1	9	3	Малий	4	Хороша	134890197
0	3	3	Малий	1	Хороша	134332480
…	…	…	…	…	…	…
0	6	6	Великий	1	Погана	135162910
0	6	6	Середній	4	Погана	134646037
1	1	1	Середній	1	Хороша	134382516

Потрібно мінімізувати час передачі повідомлення від вузла-ініціатора до вузла-приймача:

$$T = \sum t_{kj}\,;\; t_{kj} = t_{kj}^{pr} + t_{kj}^{tx}; k = 1, \overline{N-1}, k \neq j\,, \qquad (2.9)$$

Функція пошуку вузла-приймача дозволяє перейти до наступного проміжного вузла на графі:

$$F_{next} = j, \{j: t_{kj}(j) = \min t_{kj}\}, \quad (2.10)$$

Функція досягнення результату здійснює перевірку на досягнення вузла-приймача a_n

$$F_{end}(a_n) = 1; F_{end}(a_n) = 0; j \neq n, \quad (2.11)$$

Алгоритм призначений для використання на вузлі передачі та забезпечує просту логіку для вибору маршруту до кінцевого вузла. Так як існує безліч методів пошуку найкоротшого шляху, потрібно вибрати той, який дозволить швидше досягти заданої мети. Для обґрунтованого вибору алгоритму потрібний апарат порівняння. Теоретично складності обчислень обчислювальна складність будь-якого комп'ютерного алгоритму оцінюється з допомогою чотирьох основних показників: повнота, тимчасова складність, просторова складність, оптимальність. У нашому випадку для вирішення поставленого завдання достатньо оцінити алгоритми з тимчасової складності (оцінка часу роботи алгоритму). Класичні алгоритми знаходження оптимального шляху у випадку мають тимчасову складність, описану квадратичним рівнянням щодо вхідної кількості вершин графа для обробки.

Пізніший евристичний алгоритм пошуку на графі за найкращим збігом A star має тимчасову складність, описану логарифмічним рівнянням, але не при всіх значеннях кількості вхідних вершин. Існують стани, де він описується експоненційною залежністю [3]. Алгоритм JPS (Jump Pointer Search), розроблений в австралійському центрі інформаційних технологій, є успішною модифікацією алгоритму A star, і тимчасова складність алгоритму описується логарифмічним рівнянням значно більшої кількості вершин. Побудова алгоритму з урахуванням JPS дозволить скоротити витрати тимчасово пошуку шляху до кінцевого вузла. Оскільки цільовий стан системи залежить тільки від географічного шляху, а й від інтерфейсу, то цільова функція повинна враховувати цю обставину. У зв'язку з вищесказаним, ставиться додаткове

завдання визначити найшвидші інтерфейси вузла. Під каналом зв'язку розумітимемо канал обміну між двома сусідніми вузлами за вибраним інтерфейсом. Специфіка виділення основного інтерфейсу вузла в комплексі зв'язку полягає в тому, що канал зв'язку є динамічним, залежить від погодних умов, протоколу, що підтримується, наявності у сусіднього вузла такого ж інтерфейсу (інакше зв'язок по інтерфейсу неможлива), тому проводиться ранжування інтерфейсів між собою. Цільова функція алгоритму повинна вибирати оптимальний шлях - набір інтерфейсів та вузлів з мінімальним часом. На канал зв'язку впливають групи параметрів, виділення яких відбувається експериментально. Визначення залежності швидкості зв'язку груп параметрів каналу виконується з урахуванням обчислення умовної ентропії. Що стосується роботи комплексу зв'язку під подією розуміється поява чинника, впливає зв'язок (перешкоди у каналі зв'язку, відмова інтерфейсу, зовнішні впливу та інших.). Щоб порівнювати швидкості різних інтерфейсів, потрібно отримати шкалу оцінки. Мінімальним значенням для шкали є мінімальна швидкість інтерфейсу з меншою швидкістю та частими збоями, а максимальною – швидкість найпродуктивнішого інтерфейсу в комплексі з найкращими параметрами каналу.

Після оцінки наявних інтерфейсів комплексу потрібно додати пріоритет вибору інтерфейсу за швидкістю як обмеження цільової функції на етапі виконання алгоритму. Враховуючи можливість обриву чи неполадки інтерфейсу, перезапустити процес маршрутизації здатний кожен вузол, обраний цільового, тобто. при обриві зв'язку проміжний вузол запускає завдання пошуку найкоротшого шляху, де він є відправником. Алгоритм JPS використовується вузлом розрахунку цільової функції при обліку поставлених обмежень.

Також слід зазначити, що кількість вузлів зв'язку для отриманого алгоритму на порядок нижче, звідки слідує, що вибрані інтерфейси найкращі передачі інформації. Таке знання корисне для подальшої модифікації алгоритму. Надалі можна пересилати між сусідніми вузлами інформацію про вигідні інтерфейси під час відсутності трафіку. Під точками відсікання розуміються точки між початковою та кінцевою, переходи між якими неможливі. Алгоритм

складається з 7 етапів: На першому етапі формуються таблиці переходів між об'єктами мережі, що маршрутизується, за допомогою функції пошуку приймача для набору вузлів між початковим і кінцевим. Для цього обчислюється ентропія для подій, що впливають на зв'язок за такою формулою:

$$G(X) = -\sum_{i=1}^{N} p_i \sum_{j-1}^{M} p_j \log_2 p_i(j), \quad (2.12)$$

Також розраховується інформація про розкид значень швидкості та її відхилення:

$$D(U) = \sum_{i=1}^{N} p_i \, (U_i - (\sum_{i=1}^{N} p_i U_i))^2, \quad (2.13)$$

З отриманих характеристик проводиться ранжування точок за мінімальною вартістю функції досягнення щодо початкової точки. З другого краю етапі до роботи алгоритму формується таблиця вузлів як квадратної матриці. При меншій кількості переходів, ніж розмір матриці, відсутні вузли додаються до множини точок відсікання.

На третьому етапі для заданого проміжку часу проводять обчислення цільової функції швидкості для алгоритму MJPS за формулою та отримані значення зберігають у базі даних:

$$D(\Delta U) = \frac{1}{L}\sum_{i=1}^{L}(U_1 - \frac{1}{L}\sum_{i=1}^{L} U_1)^2, \quad (2.14)$$

На четвертому етапі моделюються додаткові точки, які відповідають альтернативним інтерфейсам для вузлів.

На п'ятому етапі виробляють визначення найкоротшого шляху алгоритму MJPS зі значеннями розрахованих швидкостей і результати заносяться до бази даних.

На шостому етапі виконується перехід від одного вузла до іншого, для якого перевіряється поява точок відсікання та при необхідності ініціюється новий розрахунок шляху.

На сьомому етапі перевіряється досягнення кінцевого вузла і при невиконанні цієї умови звіряється інформація про відмову проміжних інтерфейсів зв'язку, формується набір точок відсікання для повернення на 2-й етап. При досягненні вузла відбувається алгоритм закінчує роботу. На рисунках 2.9 та 2.10 наведено варіанти роботи алгоритму маршрутизації без відсікань, а на малюнку 3 з відсіканням точок, де (а) – алгоритм Дейкстри, (б) – алгоритм A star, (в) – алгоритм MJPS. N – кількість операції, t – час виконання у мілісекундах [5].

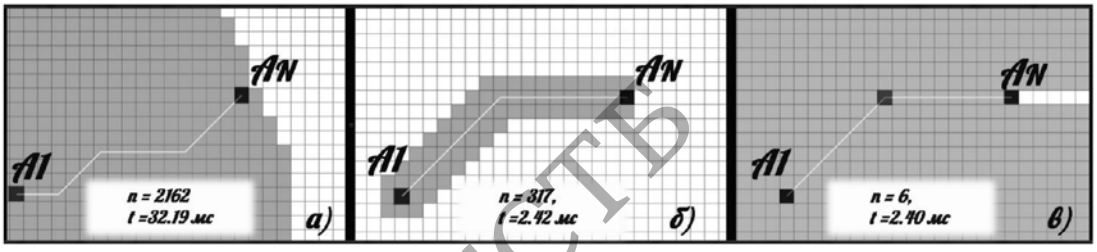


Рисунок 2.9 - Порівняння кількості операцій n та часу пошуку шляху t для заданих точок при використанні без відсікань

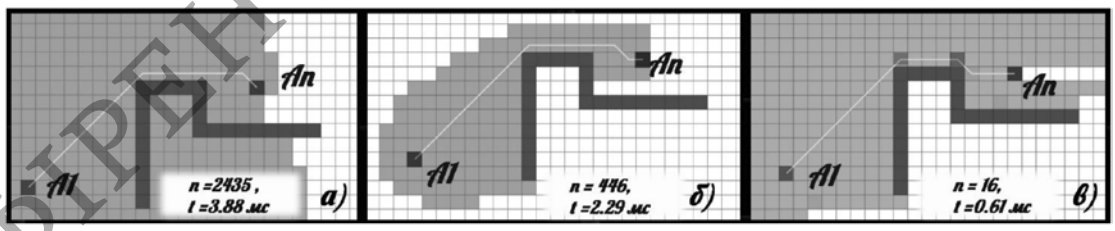


Рисунок 2.10 - Порівняння кількості операцій n та часу пошуку шляху t для заданих точок з відсіками

Видно, що запропонований алгоритм розглядає меншу кількість точок (виділені на лінії руху кольором) порівняно з іншими двома алгоритмами.

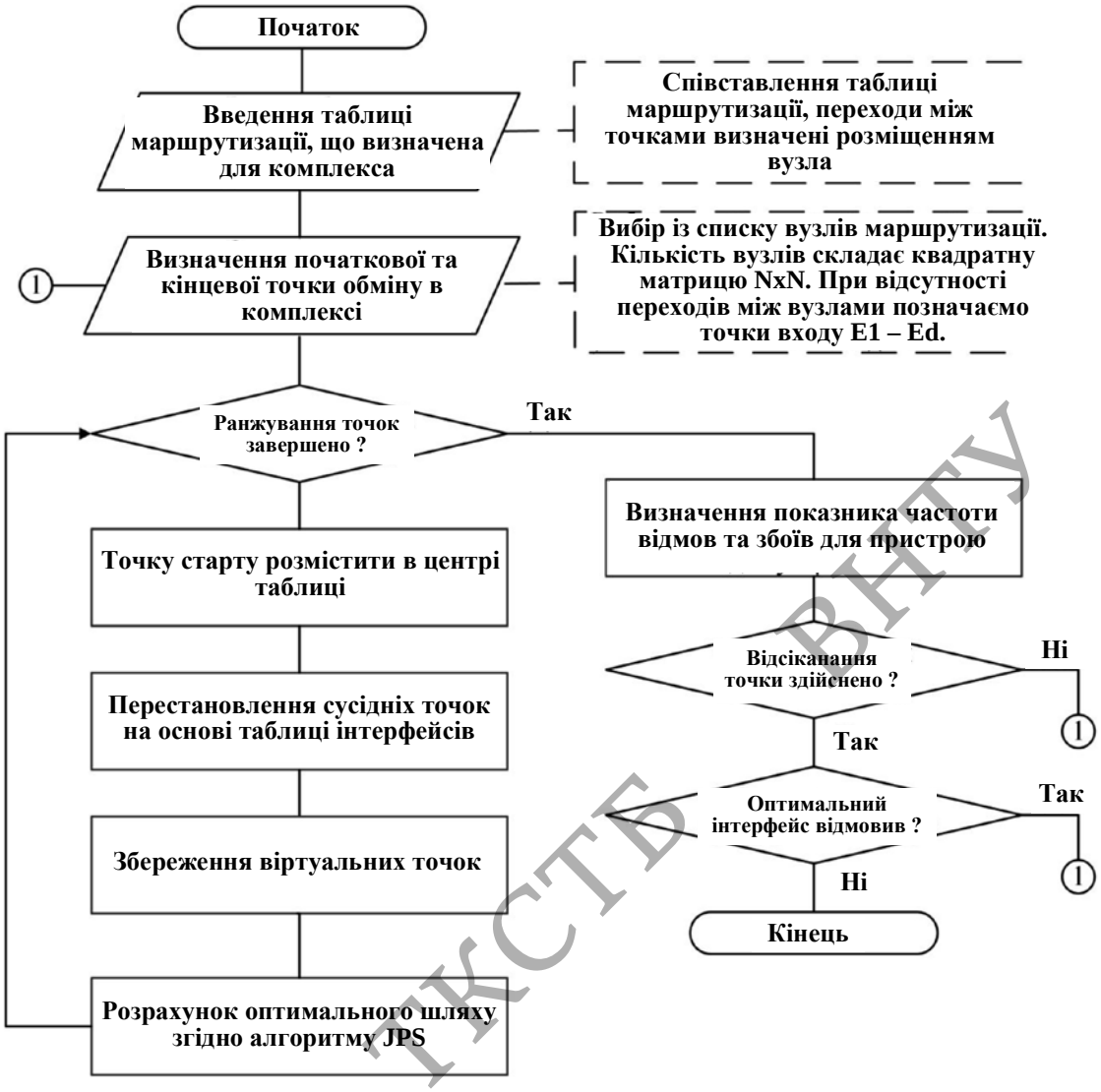


Рисунок 2.11 - Алгоритм адаптивної передачі інформації

Крім того, для алгоритмів вказані необхідна для розв'язання задачі кількість операцій n та витрат за часом на пошук шляху t для заданих точок A1 – AN. Оскільки запропонований алгоритм виконується менший час, ніж наведені алгоритми при порівняних кількостях операцій, його використання предпочтительнее. Інформаційна система складається з пристроїв, які мають набір різнорідних інтерфейсів. При застосуванні технологічної мережі формується статистика роботи комплексу за умов прийому-передачі інформації між вузлами. Технологічна мережа дозволяє виконати експертну діагностику роботи пристроїв, з урахуванням ознак встановити коректність роботи. Збір статистики здійснюється під час роботи системи та пов'язаний з алгоритмами

роботи мікропроцесорного пристрою, який накопичує набір рішень щодо адаптивного до умов середовища інформації.

2.5 Розробка модуля виявлення апаратних несправності та помилок програмного забезпечення у складі програмного блоку імітації відмов та збоїв

Робота модулів виконується разом із тестовими програмами, що змінюють значення параметрів, які контролює модуль. Модулі контролю системи дозволяють зафіксувати відмови та збої під час роботи тестової інфраструктури, але не у всіх випадках.

Для останнього модуля, що тестується, не вдалося детектувати відмови, тому для ефективності виявлення відмов і збоїв планується використання модулів виявлення спільно.

Система, що тестується представлена набором мікроконтролерів (або, як варіант, персональним комп'ютером). Керуюча система складається з п'яти компонентів. Головний компонент керуючої станції – робоча станція (РС) імітації несправності, яка проводить процедури запуску, корекції та зупинення процесу імітації несправності. Робоча станція баз даних (РС БД) варта збереження результатів (статистики) проведення випробувань. Контролери діагностики та пристрій зовнішнього контролю є набором спеціальних засобів з функціями читання та запису в пам'ять виконуваних програм за інтерфейсом JTAG.

Даний інтерфейс дозволяє здійснювати підключення до сучасних типів мікросхем та здійснювати протокол внутрішньосхемного налагодження до доступу до пам'яті та регістрів. На рисунку наведено типову схему організації JTAG інтерфейсу [9]. JTAG-інтерфейс містить через чотири обов'язкові контакти (TDI, TDO, TCK, TMS) та один додатковий контакт (TRST): TDI (Test Data In): введення даних; TDO (Test Data Out): виведення даних; TCK (Test Clock): годинник, максимальна частота якого залежить від мікросхеми; TMS (вибір

тестового режиму): контакт для керування кінцевим автоматом; TRST (Test Reset): додатковий висновок скидання кінцевого автомата.

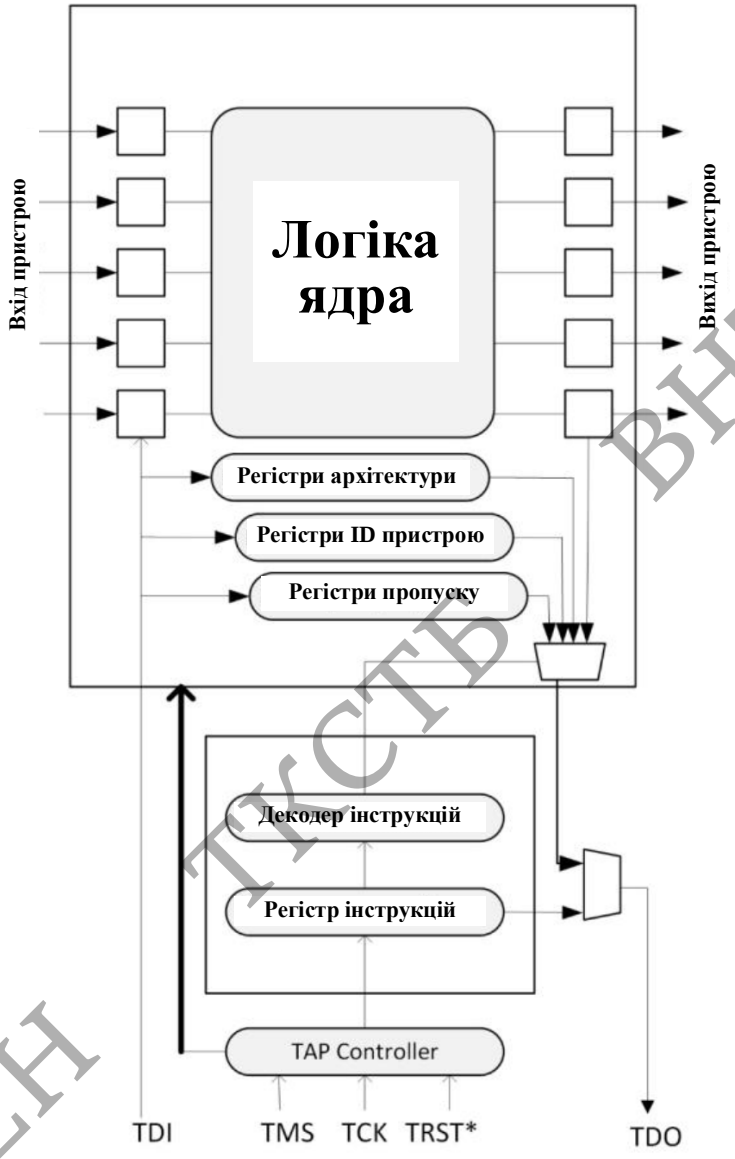


Рисунок 2.12 - Контролер діагностики на основі JTAG

Інтерфейс TAP реалізує кінцевий автомат (16 станів), який дозволяє отримати доступ до групи регістрів (IR, DR) для вимірювання мікросхеми. Управління цим кінцевим автоматом здійснюється через висновки TMS та TCK. За допомогою цього кінцевого автомата можна вибрати операцію через регістр IR (реєстр команд) і передати параметри або перевірити результат через регістр DR (реєстр даних).

РС генерує набори даних, які записуються контролером діагностики на згадку про дані набору мікроконтролерів під час виконання роботи (мікроконтролер із запущеною програмою виконує задану кількість функцій). Блоки контролю та діагностики фіксують реакцію набору мікроконтролерів після внесення змін та зберігають інформацію до бази даних, після чого РС аналізує збережені дані, коригує за даними процес імітації. Випробування зупиняються оператором РС.

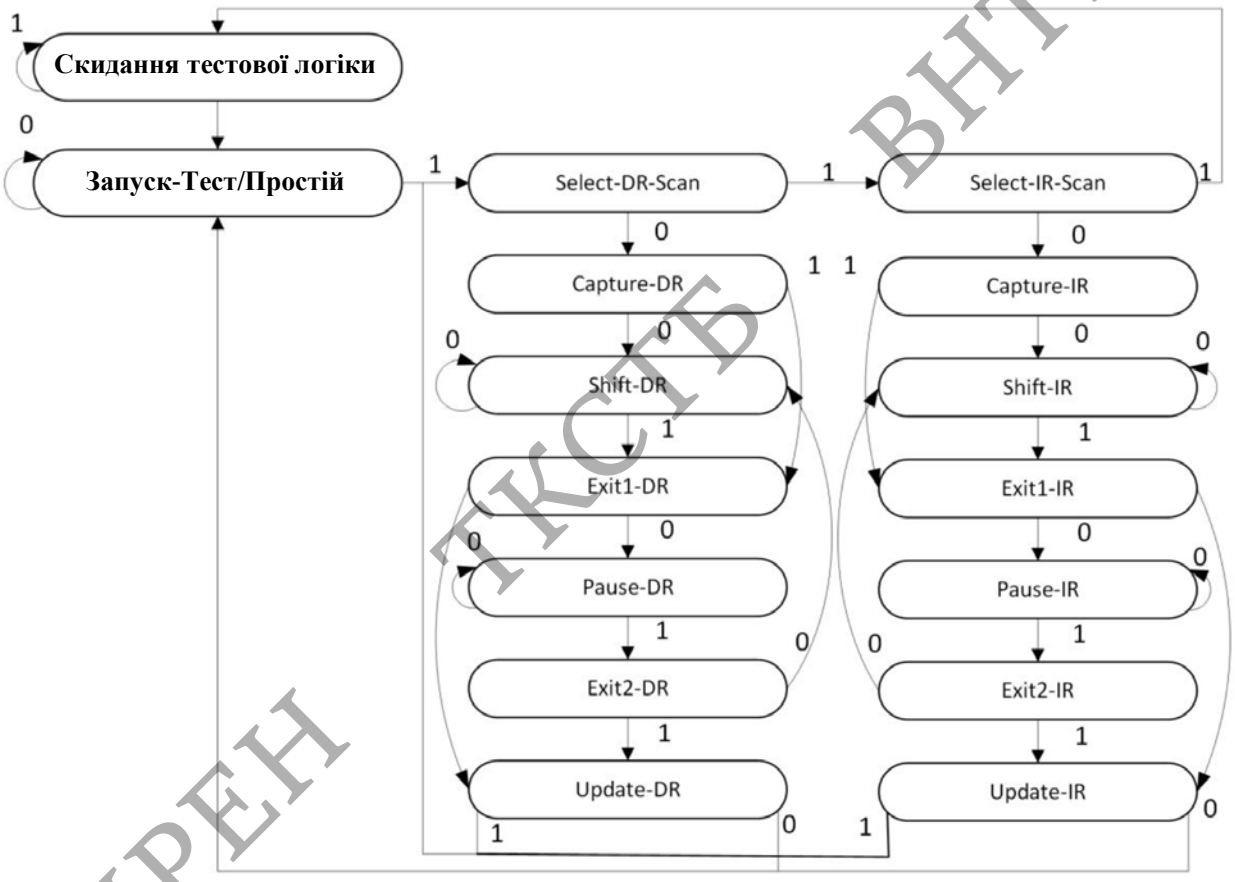


Рисунок 2.13 - Автомат станів JTAG-контролера

Програмне забезпечення, яке встановлено на РС та керує процесом імітації несправності, розглянуто нижче. При заміні мікроконтролерної системи на комп'ютер інші блоки виконані програмно, не включаючи засоби діагностики. Пристрій забезпечує автоматизацію та прискорення проведення випробувань за рахунок збору та обробки інформації про програмні функції, які виконуються в

системі, що тестується. За допомогою контролера діагностики (за інтерфейсом JTAG) збирається статистика про час, що виділяється мікроконтролером на виконання конкретних функцій програми. Таким чином, випробування щодо внесення помилок у дані проводяться спочатку на згадку, звернення до якої відбувається в програмних функціях, що займають більше мікропроцесорного часу. При імітації несправності РС імітації за допомогою засобів діагностики збирає дані та визначає ймовірність виконання програмної функції під час відмови (збою), що дозволяє при повторній роботі імітатора за наявною в базі даних визначити, яка несправність виявлена при імітації. Окремо слід розглянути технічний засіб імітації несправності, який є програмно-апаратним блоком пристрою імітації. Як блок використовується апаратні пристрої з підтримкою JTAG-інтерфейсу, які є стандартними для більшості мікроконтролерів архітектур. Інтерфейс дозволяє отримати доступ до процесора та інтерфейсів заданого мікроконтролера. Для виконання заходів із тестування, описаних у стандарті IEC 61508, алгоритми ін'єкції можуть бути реалізовані в рамках представленої системи на станції управління та пристрої імітації. Для проведення імітації несправності потрібно реалізувати впровадження (ін'єкцію) несправності програмно-апаратну систему. Блок ін'єктора несправності для пристрою має структуру, представлену рисунку 2.14.

В основному, в блоці використовується три гіпотези для формування тестових ін'єкцій несправності: White Box (ін'єкція в систему з відомими характеристиками), Gray Box (ін'єкція в частково вивчену систему), Black Box (ін'єкція в невідому систему) [14].

Ін'єктор несправності складається з алгоритму підготовки ін'єктора, що реалізується на станції управління. Цей алгоритм реалізує такі типи стратегій тестування [8, 10]: 1.Евристичний ін'єктор. Тестування на наявність несправності, що базується на знаннях про програмно-апаратні особливості системи. 2. Ін'єктор даних. Тестування на наявність несправності, що ґрунтується на зміні вхідних даних, що експлуатує проблему робастності. 3.Мутаційний ін'єктор. Тестування на наявність несправності, що ґрунтується на

псевдовипадкових змінах вихідного коду та перевірці реакції на ці зміни набору автоматичних тестів.

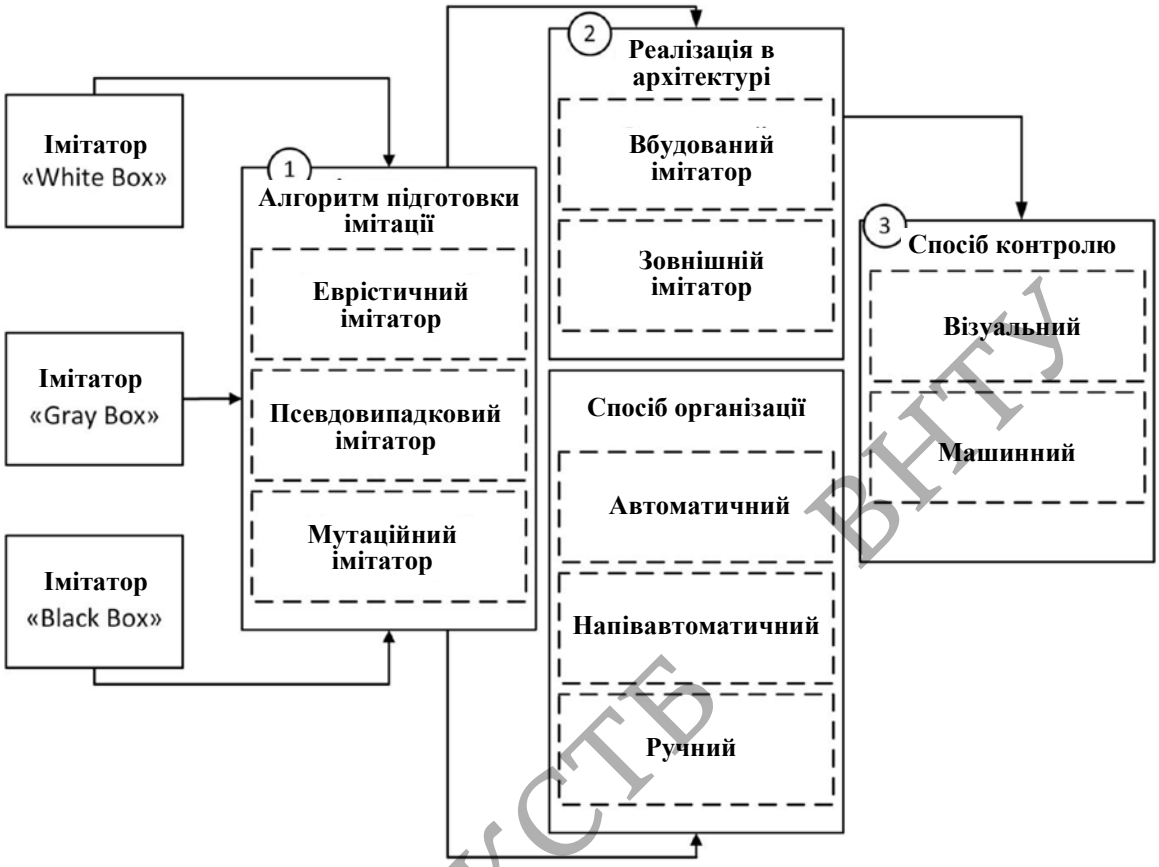


Рисунок 2.14 - Архітектура пристрою імітації (ін'єктор несправності)

Для цілей дослідження наведемо події, виявлення яких дозволить більш детально вивчити роботу мікропроцесора і класифікувати виявлені несправності для побудови в подальшому для коригування програмного та апаратного забезпечення. Ін'єкція імітованої несправності має на меті отримати відмову або збій, що фіксується за допомогою аналізу стану процесора. Ін'єкція може призвести до наступних варіантів наслідків для процесора: − відмова процесора, зупинка процесора в даному стані та повна нездатність пристрою працювати без тривалих дій щодо відновлення; − відмова процесора, зупинка процесора в даному стані та повна нездатність пристрою працювати без короткочасного перезавантаження; − періодичний неконтрольований збій процесора; −

періодичний збій процесора під час виконання функції; − одноразовий збій процесора, що визначається; − відсутність наслідків внесення змін для пристрою.

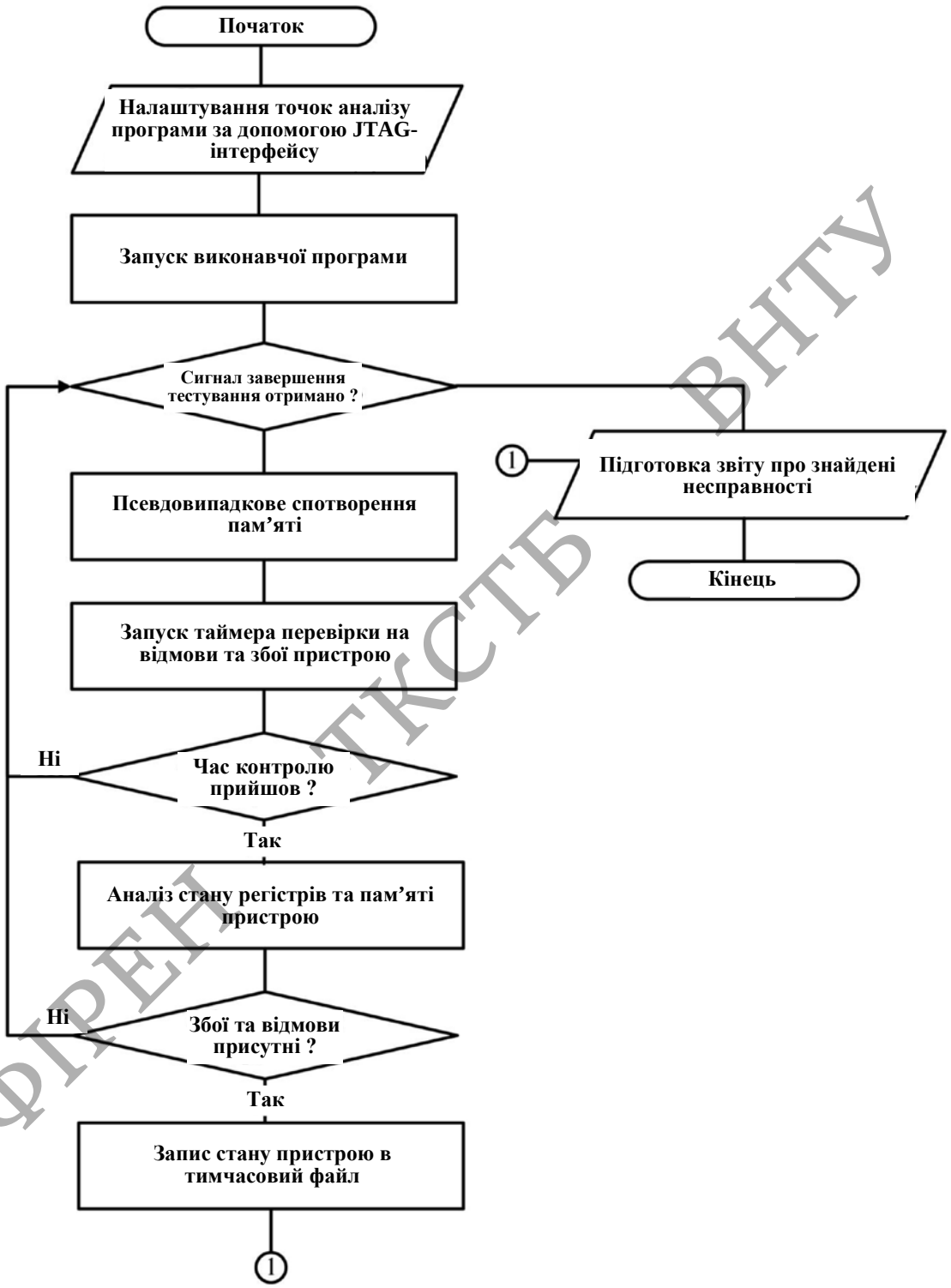


Рисунок 2.15 - Алгоритм тестування пристрою за допомогою JTAG

Аналіз реакції на імітовані несправності проводиться за допомогою засобів до модифікації яких є доступ у розробника - регістри процесора та інтерфейсів, дані, що зберігаються в ОЗП, і шаблони функцій, що виконуються, збережені до проведення ін'єкції.

2.6 Висновки до розділу 2

1. Розглянуто методи імітації несправності цифрових систем. Показано, що використання режимів спотворення вхідних даних є ефективним інструментом для тестування програмного забезпечення цільового апаратного середовища. Запропоновано структуру програмно-апаратного пристрою для імітації несправності в системах, що проектуються на базі мікропроцесорних засобів, що дозволяє виробляти ін'єкції помилок у вразливі місця ПЗ: вхідні дані та регістри пам'яті, задіяні при виконанні основних програмних функцій.

2. Досліджено методи автоматизації спотворення даних та аналізу реакції на внесені помилки в проектовану систему. Розглянуто різницю між результатами тестування на основі (1) внесення несправності у функціональні компоненти пристрою (блоки пам'яті, апаратні інтерфейси, елементи системної взаємодії) та на основі (2) імітації наслідків виникнення несправності. Показано, що використання другого методу (для якого реалізовано програмний модуль) має переваги для мікропроцесорних систем у зв'язку з відсутністю руйнівних дій та зменшенням тимчасових витрат на його здійснення.

3. Запропоновано структуру та описано функції для апаратного блоку виявлення апаратних несправності у складі програмного блоку імітації відмов та збоїв, що дозволяє класифікувати дефекти апаратних засобів.

4. Запропоновано алгоритм модуля виявлення відмов та збоїв для діагностики програмно-апаратного комплексу, який дозволяє оптимізувати швидкість передачі даних шляхом пошуку найкоротшого шляху разом із ранжуванням інтерфейсів прийому/передачі за наявності великої кількості інтерфейсів (вузлів).

3 СТРУКТУРА КОМПЛЕКСУ ТА АЛГОРИТМИ ТЕСТУВАННЯ ТЕЛЕКОМУНІКАЦІЙНИХ МІКРОПРОЦЕСОРНИХ ПРИСТРОЇВ

3.1 Структура комплексу проведення випробувань мікропроцесорних пристроїв

Розглянемо особливості існуючих аналогів у галузі імітації несправності. Відомий спосіб [6], що включає мутаційне тестування радіоелектронної апаратури та її програмного забезпечення, що керує, який використовує оцінку ймовірності виявлення імітованих моделей несправності тестованої апаратури або програми за допомогою імітації поведінки каналів введення-виведення об'єкта випробувань. Для мікропроцесорного пристрою даний варіант технічно не застосовується у зв'язку зі складністю використання для мікроконтролерів, які є власністю компаній-виробників і не можуть бути змодельовані на ПЛІС. До того ж у методу тестування відсутня чітка методика закінчення випробувань, яка залежить від моделі несправності і не дозволяє гарантувати кінцівку проведення випробувань.

При цьому типовий склад пристрою інформаційного комплексу (мікропроцесорного пристрою) включає інтерфейс JTAG, що дозволяє отримувати в режимі налагодження точну інформацію про стан пам'яті, регістрів, стека та час виконання команд на пристрої. При з'єднанні з випробуваннями електронної схеми на стійкість до відмови представляє здійснити перевірку за допомогою зовнішніх пристроїв контролю проектованої інфраструктури тестування, що полягає в імітації відмов елементів при збереженні працездатності схеми, що дозволяє в автоматичному режимі фіксувати відмови схеми за вихідними сигналами. У цій роботі пропонується розширити дані способи і застосувати на імітаторі проектованого несправності для мікроконтролера як складової частини проектованого пристрою.

Така система з програмним забезпеченням, що має режими роботи з переривання та DMA-режим, дозволить порівнювати складні режими обробки на

виході одночасно кількох результатів, що забезпечують правильний результат. Критерієм відмови системи може бути цілий набір непрямих результатів виконання функцій пристрою. Для проектованої методики необхідно здійснити використання моделі випробувань, заснованих як на знанні внутрішньої структури пристрою (модель «білого» або «сірого» ящика) та реалізації механізмів імітації, так і на основі «чорної» ящика, що дозволить дослідити складові мікросхеми, внутрішній пристрій яких не невідомо (сучасні системи включають велику кількість інтегральних мікросхем з невідомою внутрішньою структурою, що не розкривається фірмами-виробниками). Використання моделі «чорного» або «сірого» ящика при невідомій моделі НВІС дозволяє випробовувати системи з невідомою внутрішньою структурою, яка не може бути отримана від виробника.

У цьому розділі буде представлена методика, завданням якої є визначення обсягу випробувань з імітації несправності, необхідного виявлення відмов і збоїв із заданою точністю. Результат застосування такої методики полягатиме в автоматизації випробувань та скороченні часу випробувань, імітації наслідків відмов та збоїв елементів, зменшенні ймовірності відмов системи за рахунок виявлення елементів, відмова яких призводить до непрацездатності системи, визначення обсягу випробувань, необхідного для виявлення відмов та збоїв із заданою точністю та на базі спроектованого пристрою для імітації несправності у програмно-апаратних системах.

Для іскаження даних програми використовується три основні підходи. Перший полягає у генераційній схемі застосування фаззингу, тобто. існує спеціальний модуль контролю та налаштування програми фаззингу, який на основі статистики формує дані, що обробляються програмою. Фазинг [2, 6, 9] є спеціальним методом для тестування складних систем, який перевіряє стійкість роботи алгоритму до різних наборів вхідних даних. Стійкість системи до невизначених даних (властивість робастності програмного забезпечення) досліджується в умовах експлуатації програми для алгоритмів роботи пристрою. Використовуємо таке позначення: bx – ознака виникнення помилки у програмі

даних X; tthres – час здійснення фаззинга; cij - унікальний і-й збій або відмова для j-го набору даних; f – унікальна програмна функція, $\mathbb{C}$ – набір вхідних даних для мутацій. Функція Fuzz виконує спотворення даних за певними сценаріями [19]

$$Fuzz(C, t_{thres}) = \{(b_1, S_1, t_1), \dots, (b_n, S_n, t_n)\}, \quad (3.1)$$

p align="justify"> Різні комбінації вхідних даних дозволяють отримати унікальний граф виконання програми (вершинами якого є програмні функції графа, а ребра графа з'єднують вершини в порядку виклику функцій), для якого виконується перевірка роботи [7, 14, 16].

Метою проведення фаззингу є отримання максимальної кількості помилок за встановлений тимчасовий ліміт, а також максимізація обходу гілок графа програми. Як показано в роботах інших авторів, фазинг дозволяє вирішувати завдання про оптимізацію кількості знайдених уразливостей [5, 6, 14] (покриття програми) за встановлений часовий діапазон.

$$\sum_{i=1}^{N} b_x \to max, \quad (3.2)$$

Кожен окремий збій або відмова фіксується для набору даних, що викликало його виникнення.

$$C_{ij} = \begin{cases} 1, j \text{ відмова включає } i \text{ набір даних.} \\ 0, \text{в іншому випадку} \end{cases}, \quad (3.3)$$

Визначимо ознаку виявлення помилки на тестових даних, яка фіксує помилки для унікальних програмних функцій.

$$b_x = \begin{cases} 1, \text{тоді і тільки тоді коли } \exists i, j: f(c_{ij}) = x \\ 0, \text{в іншому випадку} \end{cases}, \quad (3.4)$$

При цьому слід врахувати низку обмежень для досягнення максимальної кількості помилок за встановлений час пошуку:

$$\forall i,j: c_{i,j+1} \le c_{ij},$$
$$\sum_{i.j} c_{ij} \cdot t_{ij} \le t_{thers},$$
$$\forall i,j: c_{ij} \le b_x, \text{ де } f(c_{ij}) = x, \quad (3.5)$$
$$\forall i,j: c_{ij} \le \sum_{i,j} c_{i,j},$$

Для цілей тестування використовуються спеціальні модулі збору та підготовки фазинг-тестування, які реалізовані як самостійні програмно-апаратні засоби. Такі засоби дозволяють виконати пошук даних, спрямований на виконання програмних функцій для тестування алгоритмів роботи пристрою в режимах, що найбільш цікавлять розробників. Через війну роботи алгоритму фаззинга кожної ітерації формується матриця розгалужень [10, 11], яка підсумовує кількість переходів кожної з вершин графа.

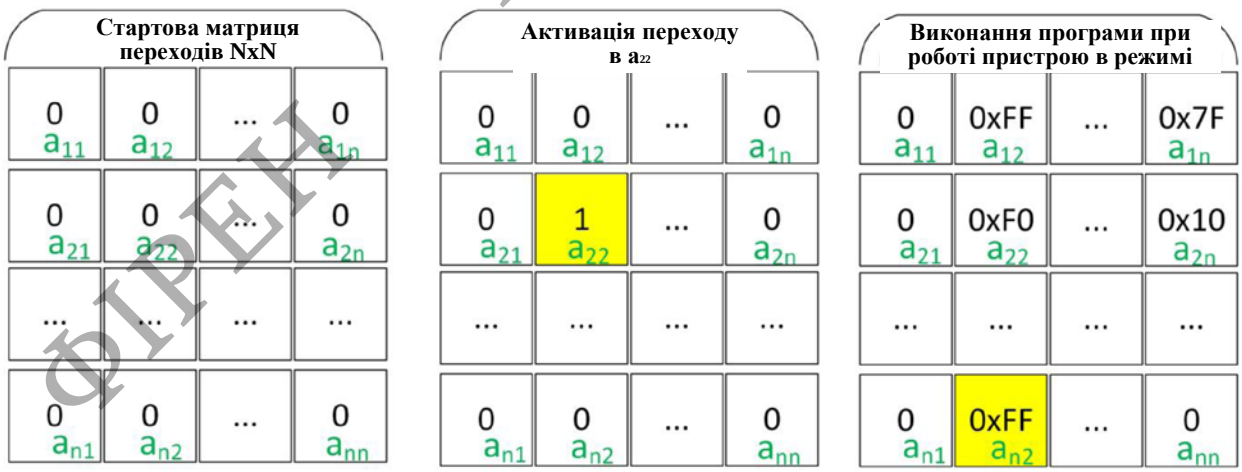


Рисунок 3.1 - Матриця розгалужень для алгоритму фазінгу

У цьому мутації вихідних даних орієнтовані дослідження відхилень реальної роботи програми від її алгоритму. Варіантом організації фаззингу є

комбінація генераційного та мутаційного спотворення даних. Вибір типу спотворення у разі проводиться на основі обумовленої розробниками стратегії тестування [8, 14]. Подано схему організації процесу фаззингу (рис. 3.2). Для цього в алгоритмі відбувається динамічний пошук та додавання нових умовних переходів (розгалужень).

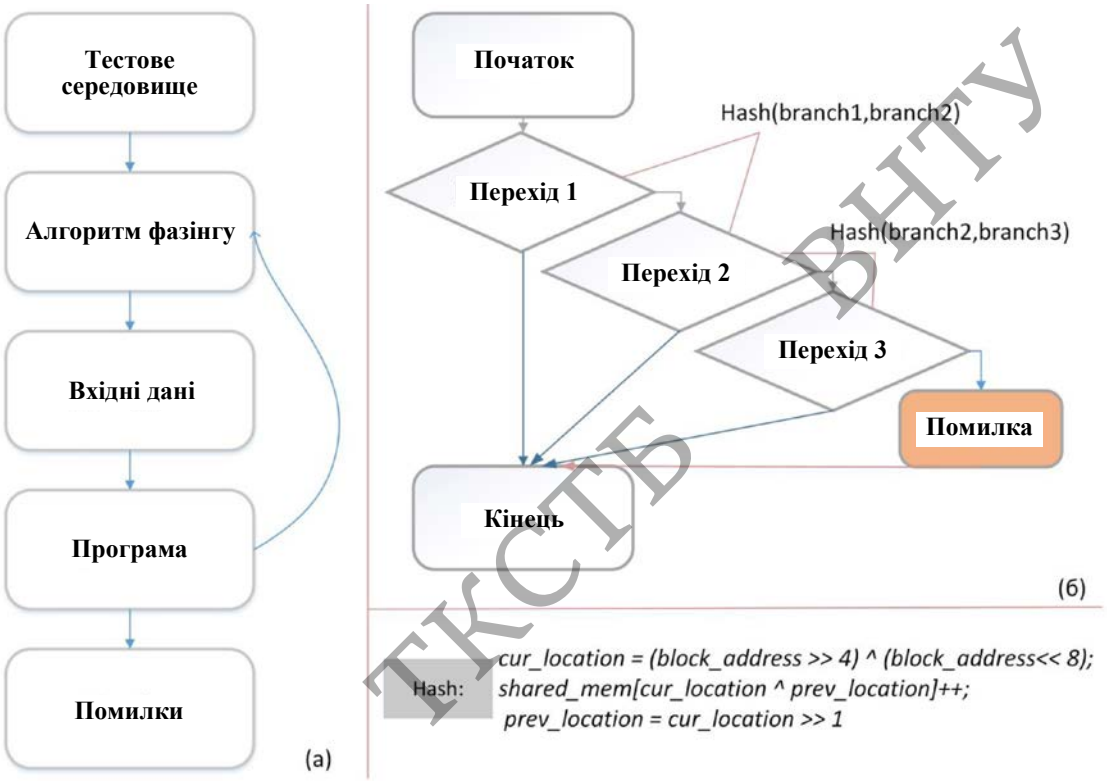


Рисунок 3.2 - Техніка фазингу: а) загальний алгоритм; б) контроль розгалужень

У цьому роботі для мікропроцесорного пристрою проводиться збір статистики роботи пристрою, і основі її аналізу формується підготовка тестових даних. Оскільки оцінка критерію визначення обсягу тестових випробувань з допомогою спотворення вхідних даних більшості програм фаззинга [12, 14, 18] виробляється вручну, нині популярні інструменти, автоматизують ці дії. Постановка задачі оптимізації обходу графа програми формулюється так [15]:

$$\max_{x \in R} \backslash\, min = \begin{cases} C_i(x) \geq 0, i \in N \\ C_i(x) = 0, i \in Q \end{cases}, \tag{3.6}$$

де F(x) – цільова функція, що містить набір всіх досягнутих розгалужень програми; x - дані для отриманого шляху; $C_i(x)$ – функції обмежень для досягнень певних вершин; R, N, Q – індекси обмежень програми, які визначаються кожної з програм.

Евристичний алгоритм фаззингу часто дозволяє управляти процесу спотворення, тому досягнення точок роботи програми, що цікавлять, залежить від вхідних даних і структури програми, що перевіряється. Досягнення функцій графа ускладнюється наявністю зациклювань і множинних повернень, що збільшує час проведення досліджень і знижує ефективність тестування. На рисунку 3.3 наведено варіант зациклювання програми фазінгу (досягнення нових вершин з певного часу не відбувається).

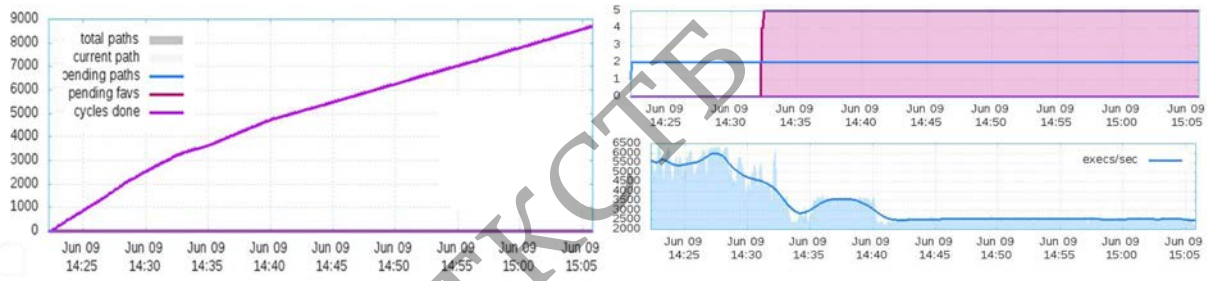


Рисунок 3.4 - Фазинг програми без використання статистики

При проектуванні систем розробникам потрібно вирішити задачу інтеграції техніки фазингу в поточну інфраструктуру тестування і з'єднати з модулями, які на відміну від фазингу дозволяють автоматично отримувати оцінку випробувань, що проводяться.

Визначення обсягу тестових випробувань та корекція роботи програми здійснюються сторонніми модулями, що активно розробляються в інженерному середовищі [14]. Тим не менш, використання модулів для мікропроцесорного пристрою вимагає наявності спільних рішень щодо фіксації стану пристрою та контролю апаратних засобів.

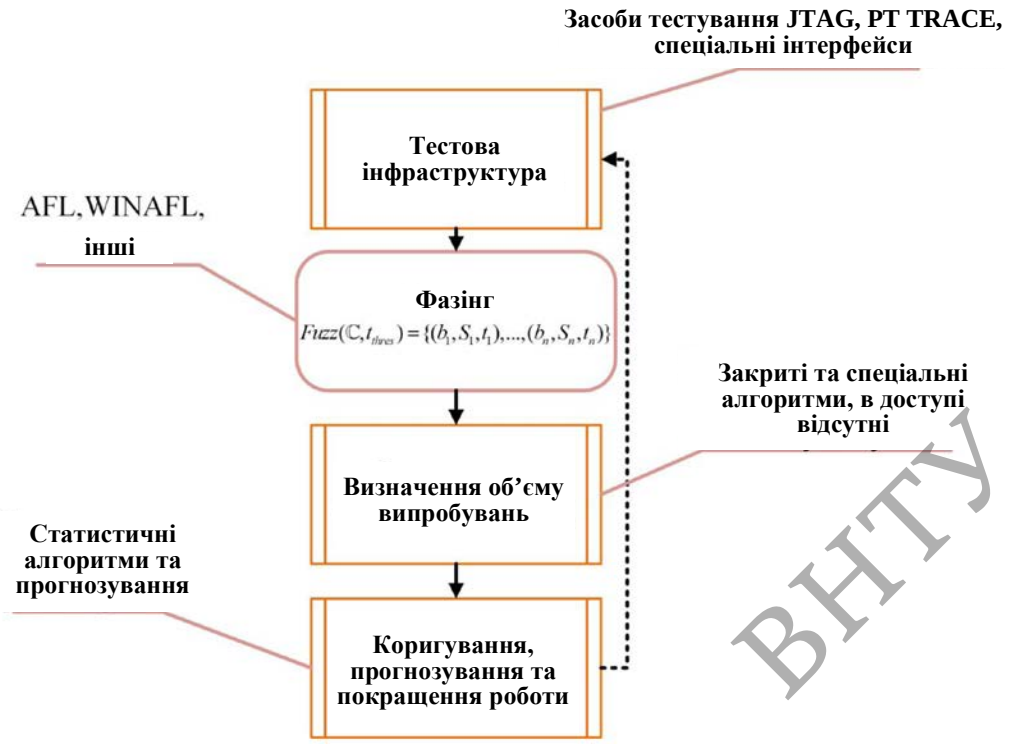


Рисунок 3.5 - Застосування фазингу у процесі випробувань

Для виконання поставлених завдань запропоновано експериментальний комплекс, який включає у своєму складі набір програмних та апаратних компонентів. Компоненти експериментального комплексу реалізують функції, описані у таблиці 3.1. Функції контролю реалізуються в програмному забезпеченні пристрою на різному рівні абстракцій (апаратні драйвера, операційна система, системні утиліти емуляції, впровадження в програму користувача). Вибір абстракції для контролю за виконанням програми пристрою проводиться за результатами експертної оцінки складності програмного забезпечення пристрою, яка формується в результаті класичного тестування. Компоненти системи є закінченими апаратними засобами, що дозволяє комбінувати їх використання у складі інформаційного комплексу.

На рисунку 3.5 представлено функціональну схему експериментального комплексу проведення випробувань, яка реалізує описані компоненти. Пристрій із засобами діагностики розглядає контроль помилок під час виконання програми

пристрою, а інфраструктура тестування забезпечує набір програмних модулів для генерації даних та аналізу реакції пристрою.

Таблиця 3.1 - Компоненти інфраструктури тестування

№	Компонент системи	Опис функції, що реалізовуються
1	Набір мікроконтролерів	Об'єкт випробування, до якого підключається пристрій імітації несправності
2	Робоча станція баз даних	Отримання та обробка даних про результати виконання функцій програмного забезпесення пристрою дя тестового набору даних
3	Пристрій імітації несправності	Верифікація інфраструктури тестування з використанням тесті системи з відомим результатом (порівняння реакції пристрою на внесення данних з необхідною реакцією). Мається на увазі , що ящо внести помилку, яка виявляє збій при класичному тестуванні, то інфраструктура тестування дозволить побачити, що помилка сталась. Якщо помилки немає, комплекс налаштований не коректно.
4	Станція імітації несправності	Аналіз виконання програми з допомогою евристичних алгоритмів моделі організації випробувань (з використанням техніки фаззінга).
5	Статистичний програмний блок: Визначення Відмов/Збоїв	Налаштування програмних засобів комплексу для проведення випробувань(інфраструктури тестування) для контролю показників програми при проведенні фаззінга.
6	Комутаційний блок JTAG/DMA	Формування модифікованих даних шляхом спотворення бітів даних для досягнення найбільшої кількості переходів в програмі.

Опишемо блоки функціональної схеми пристрою. Програмні засоби внесення даних та діагностики включають чотири блоки: - перший блок (інструментування на рівні інтерфейсів) використовує апаратний інтерфейс JTAG/SWD для отримання та внесення даних до програмного забезпечення;

- другий блок (інструментування лише на рівні ядра) реалізує спосіб інструментування лише на рівні операційної системи (системний рівень) і здійснює доступом до апаратним інтерфейсам з допомогою функції процесора, використовуючи програмні засоби процесорів Intel PT; - третій блок (інструментування лише на рівні емуляції ОС) передбачає створення копії операційної системи (емуляцію) для відстеження виконання програми з допомогою програм віртуалізації DOCKER і QEMU; - четвертий блок

(інструментування бібліотек) реалізує спосіб використання безпосередньо модифікації програми, що відстежується, за допомогою засобів компілятора та динамічних бібліотек.

Розглянемо склад і функції інфраструктури тестування пристрою, основою якої є програмні модулі тестування, реалізовані на окремому (сервісному) комп'ютері, що підключається до пристрою для внесення вхідних даних і аналізу реакції на їх внесення.

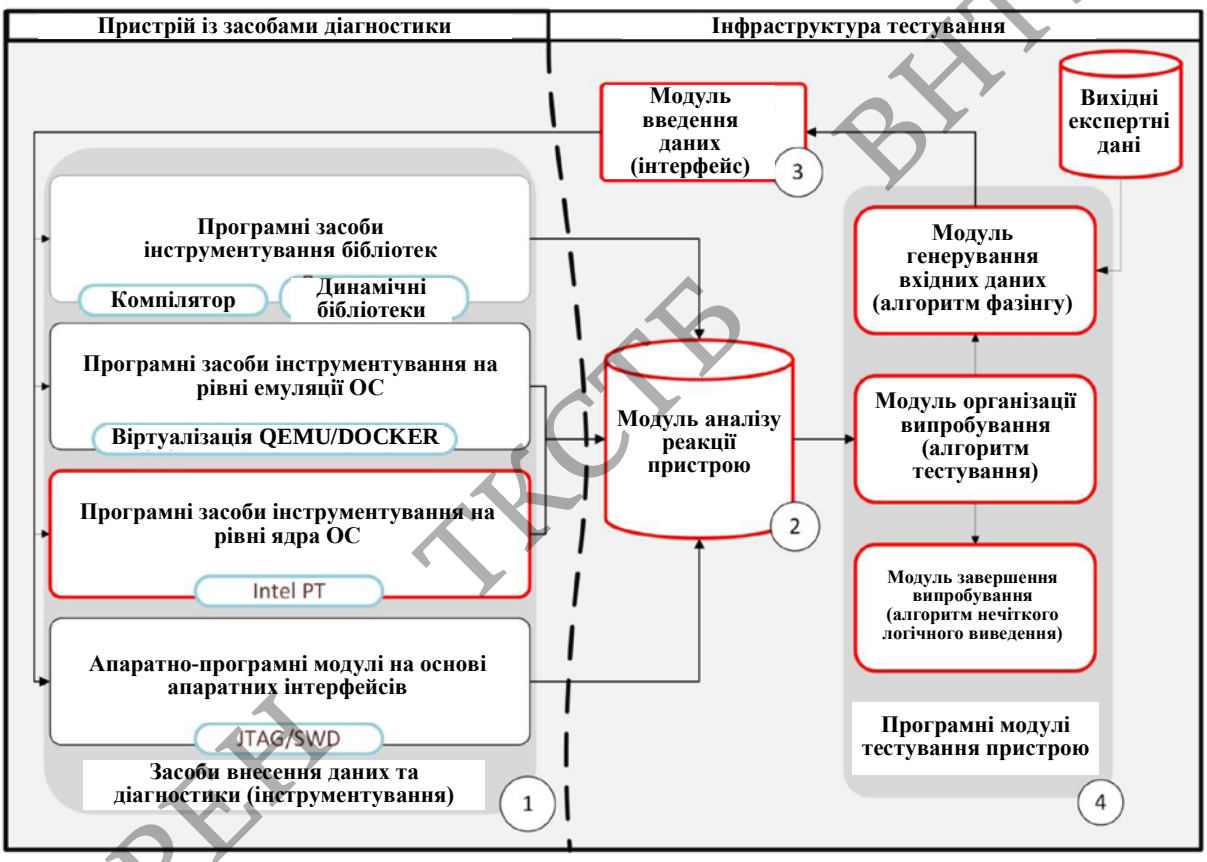


Рисунок 3.6 - Функціональна схема комплексу проведення випробувань

Модуль організації випробувань реалізує алгоритм запуску, зупинки та корекції випробувань. Для запуску випробувань він використовує модуль генерації вхідних даних, який формує спотворені дані для внесення пристрою. Модуль закінчення випробувань обробляє дані від пристрою та формує критерій закінчення випробувань. Модуль введення даних є інтерфейсом обміну для

введення та виведення інформації (від засобів діагностики) на сервісний комп'ютер. Модуль аналізу реакції пристрою реалізується на сервісному комп'ютері та приймає дані від пристрою. Модуль генерації вхідних даних за допомогою фазінгу є важливим компонентом інфраструктури тестування, який з використанням евристичного алгоритму генерації даних здійснює підготовку викривлених вхідних даних. Підготовка даних полягає у спотворенні бітів експертних даних у псевдовипадковому порядку (з нуля на одиницю) та за допомогою евристичного алгоритму фазингу, застосовуючи при цьому логічні операції XOR, AND та їх комбінації (а також роботи зі словником, перестановки та усічення), з метою задіяння більшого обсягу виконуваної програми, що з контролі проявляється як підвищення значень метрик фаззинга PATH_COV і ERR_CNT (визначаються експериментально).

Пристрій із засобами діагностики представлено з погляду інфраструктури щодо його використання. Розглянемо докладніше структуру спільної оцінки з допомогою представлених засобів (рис. 3.6) [8].

Набір тестованих мікроконтролерів 1 являє собою кілька пристроїв, що володіють JTAG-інтерфейсами стандарту IEEE 1149.1 або новіше, об'єднаних JTAG-ланцюжок за допомогою послідовного підключення даних пристроїв. Для перевірки набору мікроконтролерів виконується тестова програма, результат виконання якої відомий на будь-якій стадії виконання. При виникненні несправності система зчитує дані з блоку з мікроконтролера відмови за відсутності стандартного сигналу про виконання операцій, що видається на вихід мікроконтролера і фіксується іншими блоками системи.

Робоча станція для імітації несправності є пристрій із запущеним алгоритмом збору статистики та JTAG-програмою, яка на основі переданих команд алгоритму модифікує вміст пам'яті мікропроцесора на основі сигнатурного аналізу. Також на робочій станції запущено експертний алгоритм, який коригує зони застосування алгоритму для збільшення швидкості моделювання відмов та збоїв. Набір випробувань реалізований в алгоритмі аналізу проблемних ситуацій, що виникають під час роботи автоматизованої

радіостанції, що інтегрується в діючу інфраструктуру цифрових мереж з множинним доступом, та реалізує тестування функцій пристрою за допомогою техніки фазингу (випадкового внесення спотворень у дані).

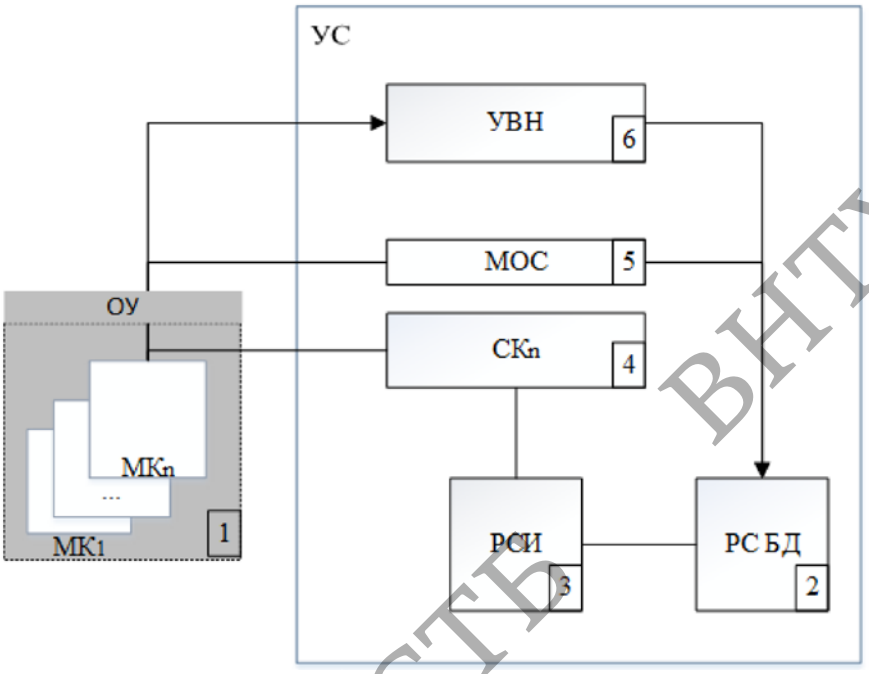


Рисунок 3.7 - Тестове середовище для імітації несправності

Алгоритм включає спотворення вхідних даних за допомогою пристрою імітації несправності (модифікований алгоритм фазінгу на основі збору статистики про функції роботи пристрою) та програмний блок, який дозволяє визначити критерій закінчення випробувань на основі математичної моделі на основі нечіткого логічного висновку (рис. 3.7 та 3.8).

Представимо набір випробувань, що реалізується в рамках виконання даного алгоритму:

1) збір та аналіз вихідних даних для мікропроцесорного пристрою;

2) формування та завантаження вихідних даних у мікропроцесорний пристрій;

3) підключення імітатора несправності та засобів синхронізації для аналізу та діагностики;

4) запуск керуючого стенду щодо випробувань;

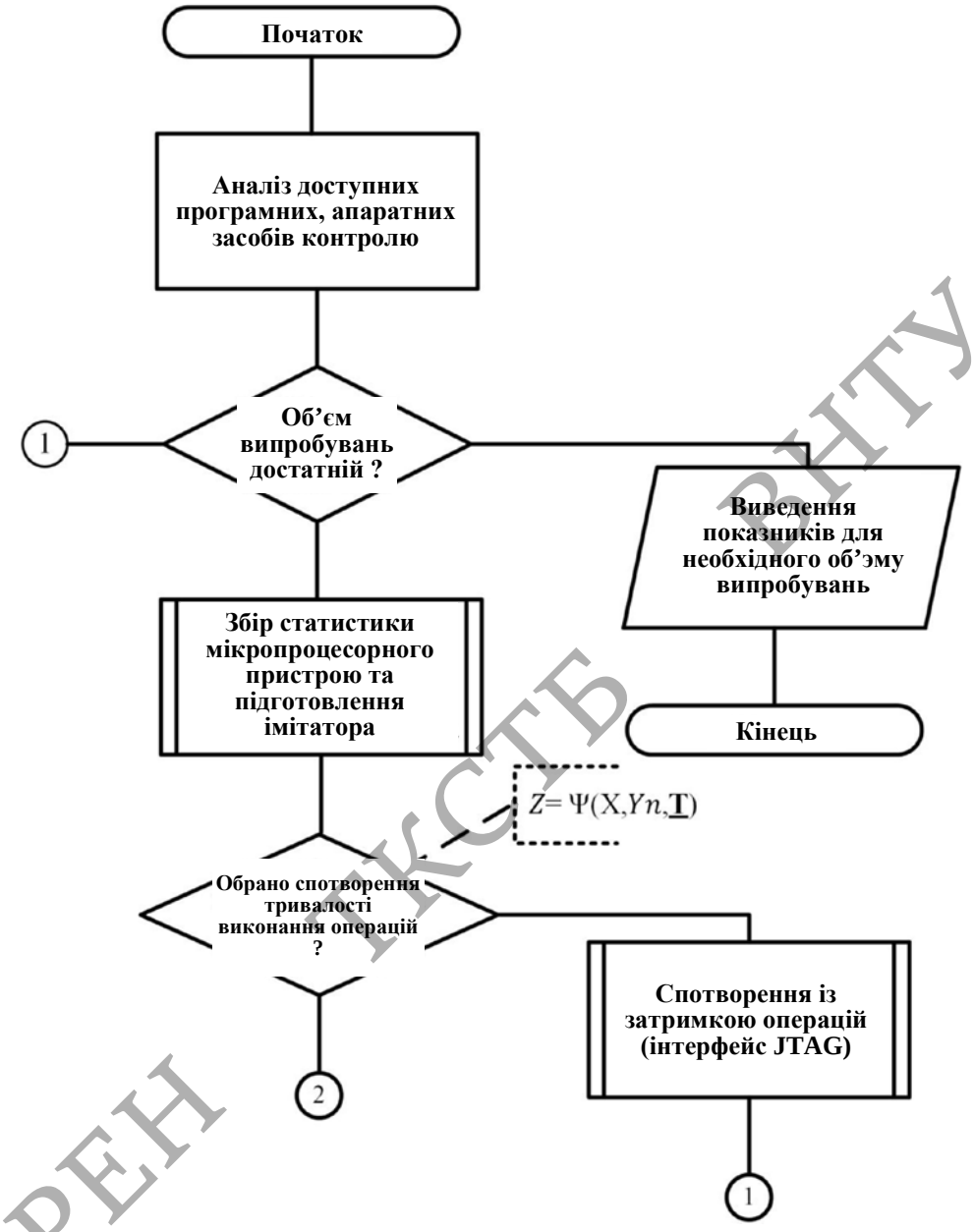


Рисунок 3.8 - Алгоритм аналізу проблемних ситуацій пристрою

5) ініціювання роботи в основних режимах устрою;

6) проведення покрокового дослідження визначення критерію закінчення випробувань;

7) отримання критерію закінчення та непрямих ознак для критерію, параметрів середнього часу напрацювання на відмову під час проведення випробувань.

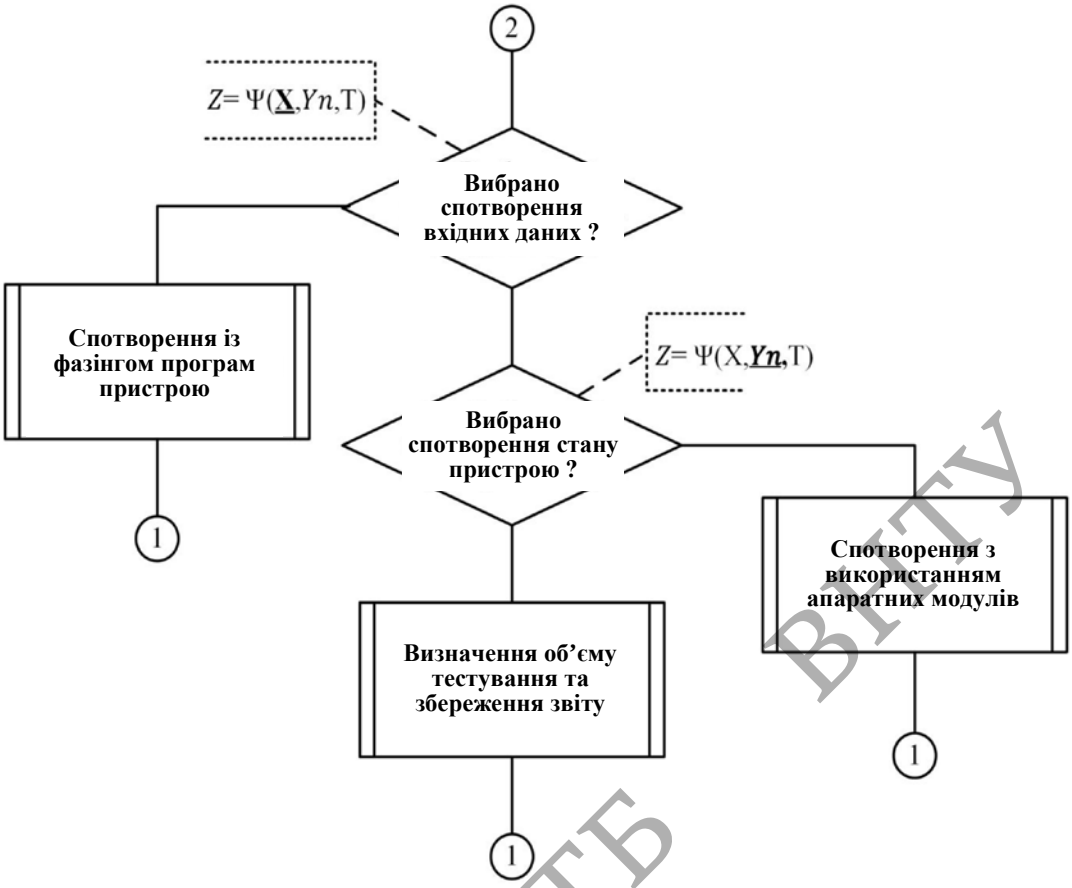


Рисунок 3.9 - Алгоритм організації випробувань (продовження)

Для здійснення даних пунктів у рамках методики провадиться ряд обчислень за допомогою засобів діагностики (на основі JTAG). Це дозволяє автоматично діагностику технічного стану для подальшого усунення системних помилок, що виявляються при завантаженні програм користувача в реальне апаратне середовище пристрою [7, 12] (прояв властивості емерджентності).

3.2 Алгоритм передобробки вихідних даних з визначенням режимів випробування пристрою, функцій і точок контролю, що перевіряються

Для випробування пристрою важливим є вивчення властивостей програми під час її експлуатації на пристрої (у складі програмно-апаратного комплексу). Динамічними показниками програми є граф розгалужень програми, статистика експлуатації асемблерних команд та програмних функцій.

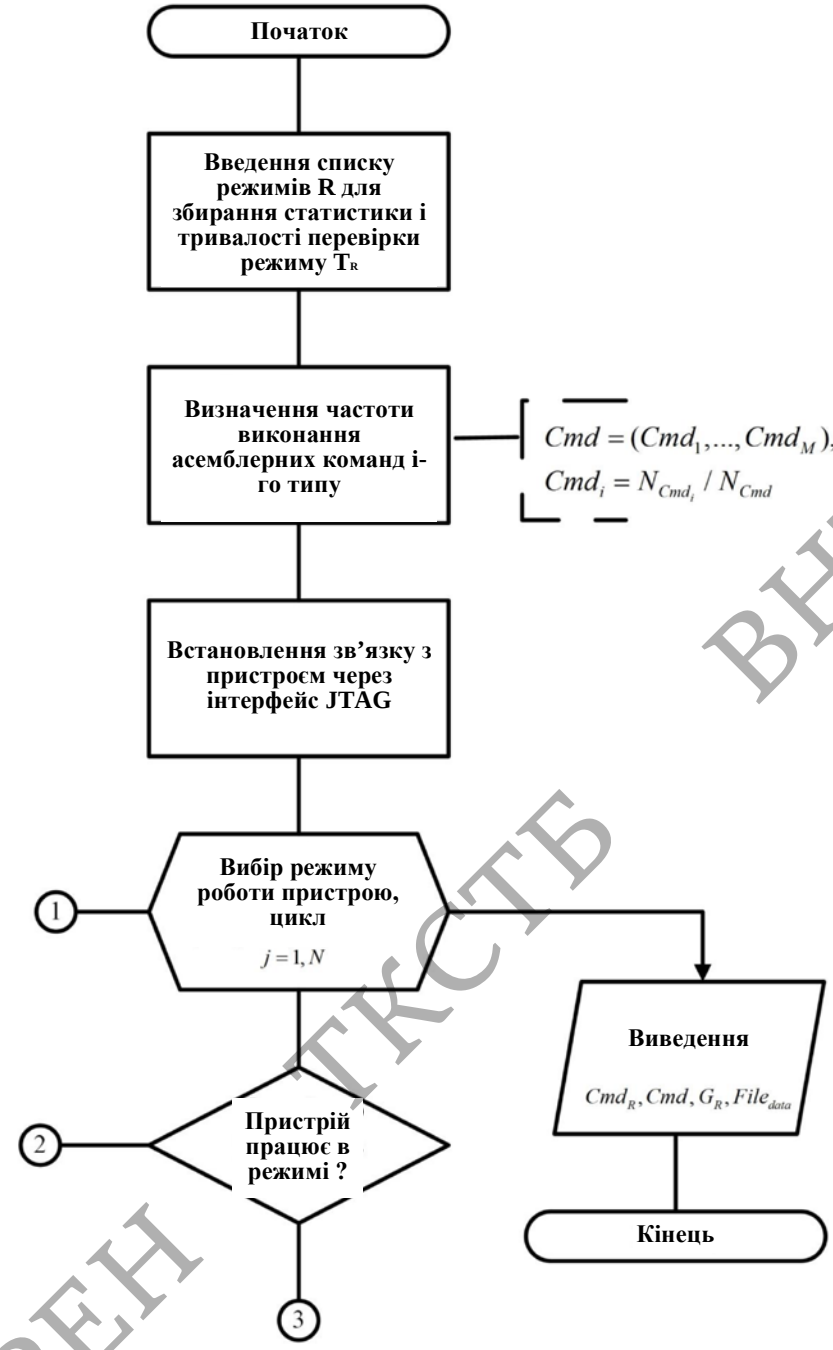


Рисунок 3.10 - Алгоритм попередньої обробки вихідних даних

Оскільки властивості програми залежить від режиму роботи пристрою, має сенс вивчення властивостей програми у межах роботи встановленого режиму.

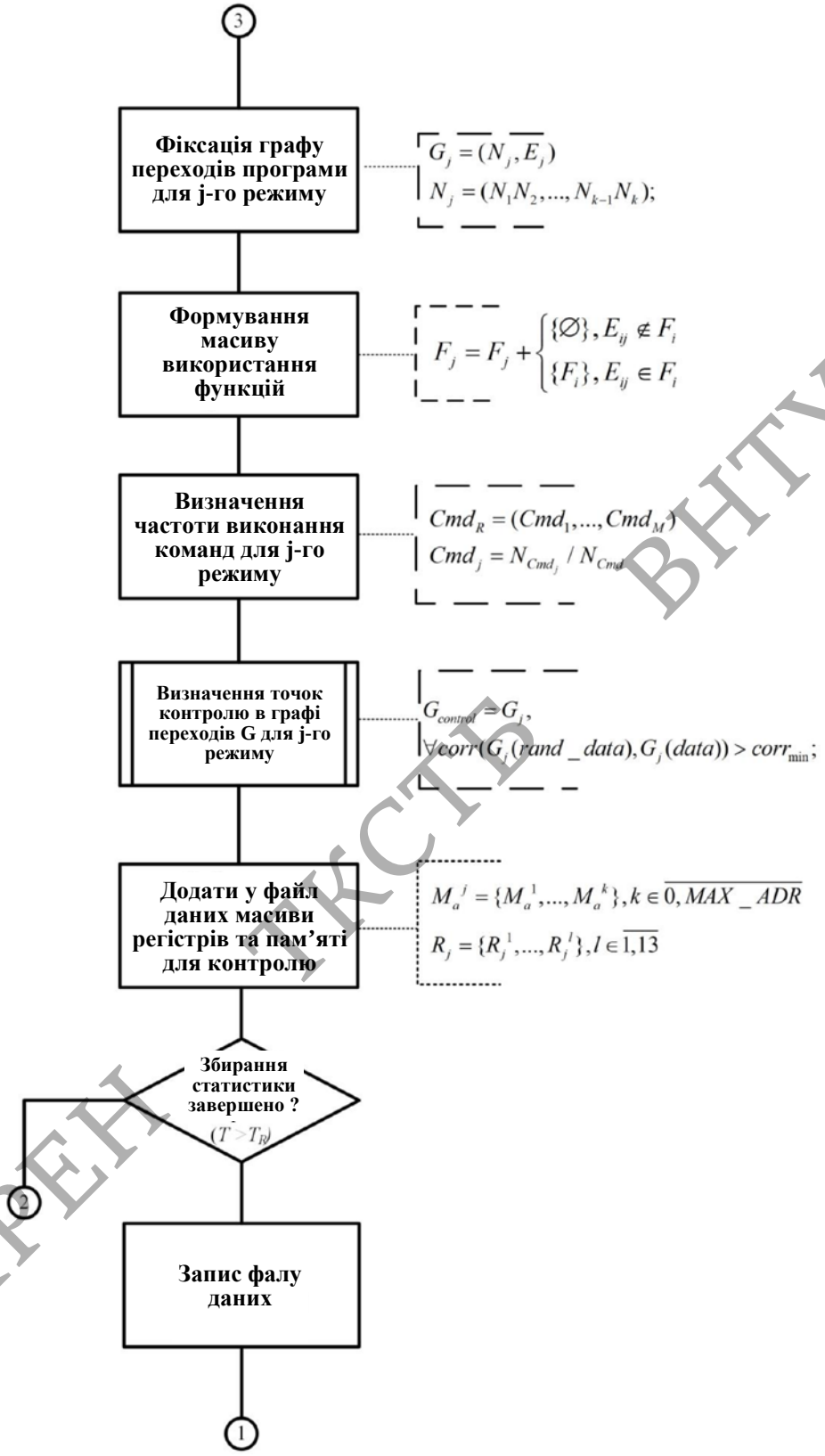


Рисунок 3.11 - Алгоритм попередньої обробки вихідних даних (продовження)

Слід зазначити, що з розробці програм часто використовують точки контролю. Під точками контролю розуміється поточний адресу пам'яті, котрій у

час виконання можна визначити вміст регістрів і даних. Тобто. у точці контролю відхилення у значеннях, що зберігаються в регістрах та пам'яті при різних вхідних даних мінімально.

Збір статистики проводиться в основних режимах роботи пристрою, для яких потрібно забезпечити стійкість до відмов та збоїв. Отримана статистика використовується як початкові дані для роботи алгоритму фаззингу, який завантажує шаблони, що дозволяють досягти вузлів графа, де розташовуються основні функції, які експлуатуються при роботі пристрою в цільових режимах мікропроцесорного пристрою. Граф розгалужень записується кожного кроку роботи алгоритму як файлового масиву. Для масивів виконується ранжування. Ранжовані дані використовуються при завантаженні в алгоритм фаззингу, що дозволяє обійти ряд обмежень для алгоритму фазингу, які важко реалізуються без модулів попереднього контролю та аналізу реакції (або аналізу параметрів лише програми без урахування характеристик проектованого пристрою).

Результатом роботи алгоритму є набір масивів даних, який вказується у вигляді файлової директорії та передається евристичному ітераційному алгоритму фазінгу.

3.3 Алгоритм програми внесення ін'єкцій відмов та збоїв у мікропроцесорний пристрій на основі техніки фаззингу

Тестування пристрою проводиться шляхом штучного внесення помилок у програмне та апаратне забезпечення пристроїв, що перевіряються. Як вихідні дані використовуються результати роботи алгоритму передобробки вихідних даних. Внесення помилок проводиться за допомогою спотворення даних, заснованого на виявлених розгалуження функції.

Алгоритм здійснює послідовне дослідження розгалужень програми, що допускають досягнення нових розгалужень (шляхів програми) та у спеціальних контрольних точках програми для порівняння з шаблоном у заданому режимі.

При цьому розгалуження належать до конкретної функції. Відмови та збої фіксуються для всієї функції загалом.

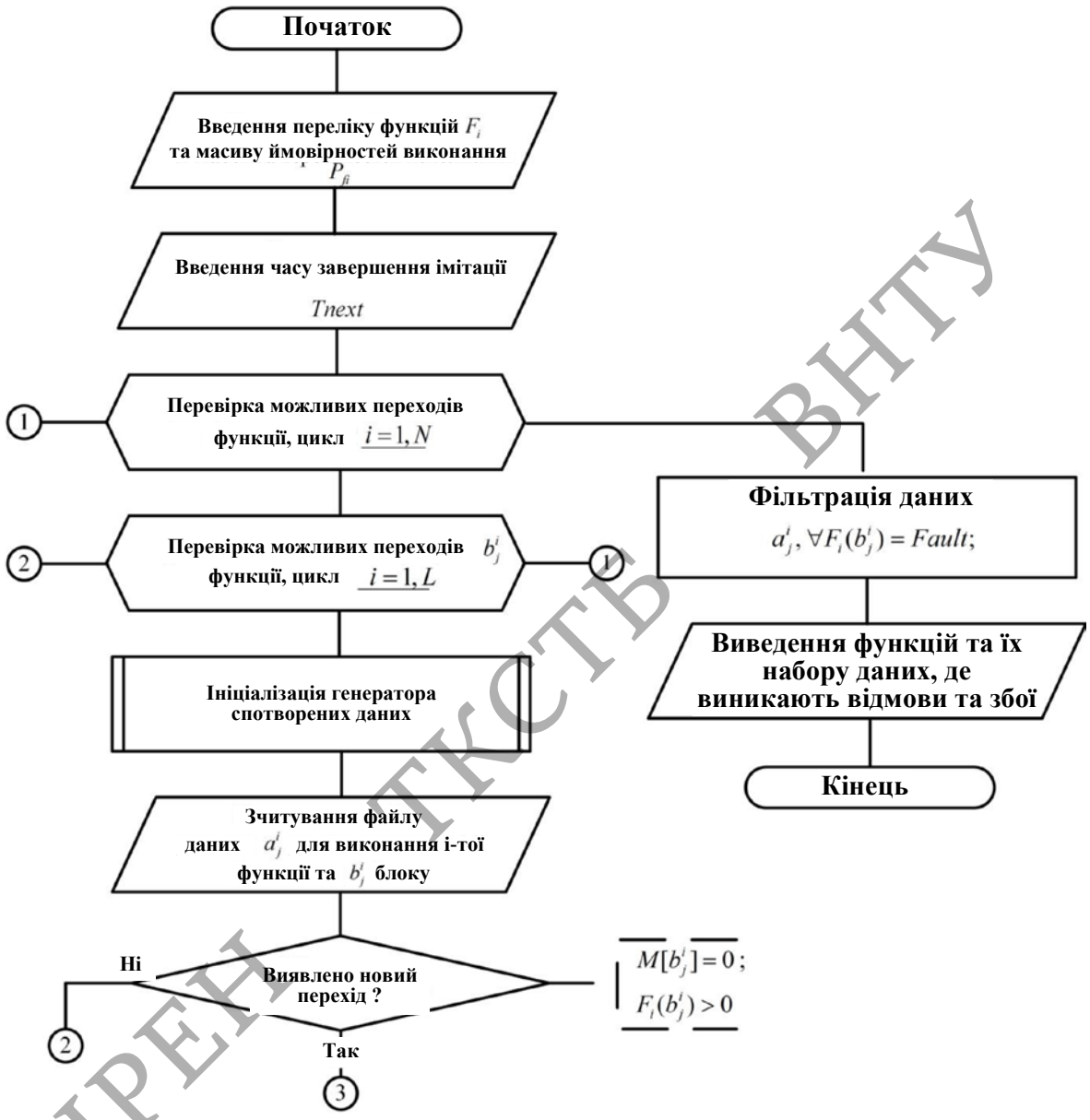


Рисунок 3.12 - Алгоритм програми внесення несправності

Алгоритм внесення випадкової ін'єкції даних та алгоритм ін'єкції даних з підготовленою гіпотезою становлять разом з алгоритмом формування статистики викликів групу алгоритмів ін'єкції несправності.

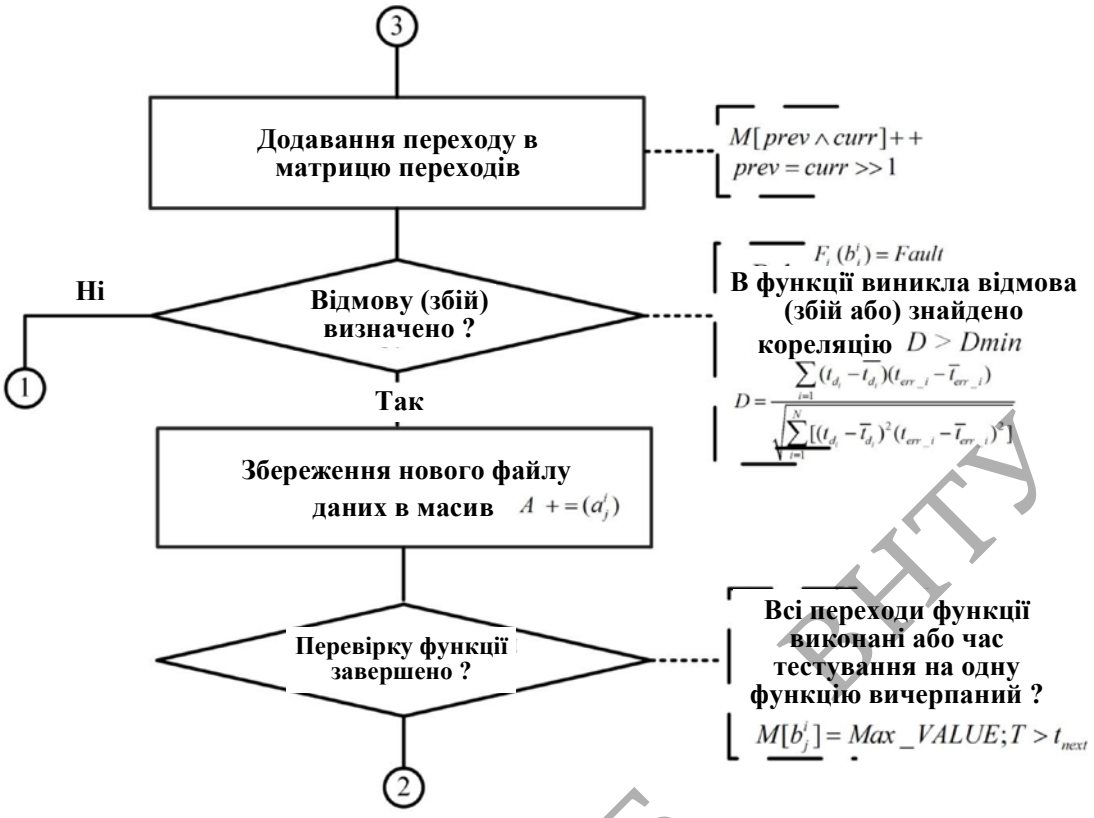


Рисунок 3.13 - Алгоритм програми внесення несправності (продовження)

Тестування програмно-апаратних пристроїв за допомогою імітації несправності регламентується стандартами IEC 61508 та ГОСТ Р МЕК 61508 і є обов'язковою вимогою при розробці пристроїв інформаційних комплексів [2, 6, 19]. Формування критерію закінчення випробувань, що дозволяють виявити дефекти складного мікропроцесорного пристрою, необхідно здійснювати на основі оцінки основних функцій, що визначаються технічним завданням. Зазвичай програма випробувань, що формується замовником проектованого пристроєм виходячи з набору функцій, що виконуються, не гарантує отримання тестового сценарію, застосовуваного на практиці. Набір випробувань може бути надлишковим або недостатнім, оскільки не враховує особливості програмно-апаратних засобів, що реалізують ці функції. Усунення надалі не виявлених дефектів тягне за собою додаткові витрати тимчасових та технічних ресурсів. Модифікація ОЗП виробляється синхронно до виконуваних алгоритмів,

використовуючи властивості та структуру функцій, що дозволяє точково використовувати фазинг.

3.4 Алгоритм визначення обсягу тестових випробувань щодо імітації несправності на основі нечіткого логічного висновку

Ключовою перевагою фазингу для системного тестування є пошук апаратурних несправності за допомогою заміни даних для інтерфейсів або апаратурних модулів у процесі випробувань, коли реакція пристрою залежить від апаратури. У роботі при розгляді питання тестування програмно-апаратних пристроїв основну увагу приділено формуванню критерію завершення випробувань, що враховує як функції системи, а й особливості програмного забезпечення та ключові параметри апаратних засобів, що характеризують стабільність роботи. Пропонується методика оцінки виконання основних функцій пристрою, що реалізується програмою випробувань на основі спільного використання техніки фаззингу (для зміни вхідних даних) та математичного апарату нечіткої логіки (для визначення моменту зупинки тестування). У перевіреному інформаційному комплексі тестуванню підлягають пристрої, які призначені для обміну інформацією користувача по мережі, а також для її обробки, зберігання та подальшого аналізу. Відхилення від шаблону обробки з певною ймовірністю визначатиме наявність помилок у програмному коді [3, 10, 11].

Після отримання статистики функцій потрібно сформувати набір ін'єкцій, який використовується в експериментальному аналізі стійкості пристрою до відмов та збоїв. Модифікація даних може бути здійснена при досягненні функцій пристроєм, які виконуються максимально інтенсивно в процесі роботи вибраного режиму пристрою. Синхронізація модифікації даних та виконуваних функцій програми здійснюється за допомогою аналізу пам'яті пристрою. Ін'єкція неправильних, спотворених даних різні часові проміжки визначає стійкість алгоритму. Як було зазначено раніше, для тестування застосовується програмна

імітація несправності з використанням техніки фаззингу, що дозволяє автоматизувати процес виявлення помилок за допомогою генерації викривлених даних на пристроях під час роботи. При недостатньому обсязі тестових випробувань ймовірність прояву невиявлених, потенційно небезпечних системи дефектів значно підвищується, тому важливо визначити критерій закінчення тестування [4, 9].

Оскільки мікропроцесорні пристрої функціонують, використовуючи складне програмне забезпечення, пропонується формувати критерій закінчення тестування на основі аналізу статистичних даних про роботу апаратних і програмних засобів цих пристроїв. Програма випробувань інформаційного комплексу використовує статистичні показники для аналізу та ухвалення рішення про закінчення або продовження випробувань. Збір статистичних даних займає тривалий час та виконується в автоматичному режимі засобами тестування та діагностики. Початкові дані формуються за експертними шаблонами. Це означає, що в якості початкових даних приймаються текстові набори, що передаються пристрою, що найчастіше застосовуються при реальній експлуатації. В результаті роботи програми формуються статистичні показники, що характеризують обсяг тестових випробувань, що відповідає необхідній точності оцінки середнього часу напрацювання на відмову (встановлену експертами). Крім того, пропонується виконати аналіз наступної реакції програмного забезпечення пристрою, використовуючи нечіткий логічний висновок для визначення моменту завершення перевірки функцій, що виконуються.

Для проведення аналізу та визначення моменту закінчення випробувань пропонується використовувати такі чотири показники [6]. Як перший показник приймається параметр, що характеризує точність оцінки середнього часу напрацювання на відмову β (визначається при виконанні програмних функцій, у відсотках):

$$\beta = P(|\tilde{P} - p| \leq \varepsilon_\beta), \tag{3.7}$$

де P – частота відмов та збоїв пристрою; p – ймовірність відмов та збоїв пристрою; εβ - допустима похибка оцінки середнього часу напрацювання на відмову; p – реальне значення ймовірності відмов та збоїв, що визначається при експлуатації пристрою; P – довірча ймовірність, яка визначається як відхилення між реальним значенням ймовірності виникнення відмов та збоїв та його оцінкою за частотою. Другим прийнятим показником є кількість апаратних відмов при виконанні програмних функцій за період проведення випробувань K. Як третє приймається показник I_β, що характеризує ймовірність виконання пристроєм некоректних дій для даного алгоритму роботи (відповідних відмов і збоїв пристрою) під час реалізації програмних функцій. Показник I_β розраховується так:

$$I_\beta \epsilon \left(\left[\tilde{P} - t_\beta \sqrt{\tilde{P}\left(1-\tilde{P}\right)K^{-1}}\right]; \left(\left[\tilde{P} + t_\beta \sqrt{\tilde{P}\left(1-\tilde{P}\right)K^{-1}}\right]\right), \qquad (3.8)$$

де K - кількість апаратних відмов пристрою; t_β – число середньоквадратичних відхилень від центру розсіювання середнього часу напрацювання на відмову; P – частота відмов та збоїв пристрою.

Показник K фіксується апаратними засобами контролю, а показник β встановлюється перед початком випробувань експертами, які враховують вимоги до стійкості до відмов та збоїв. Як показник D приймається коефіцієнт кореляції між часом внесення помилок у вхідні дані та часом настання відмов та збоїв у процесі проведення випробувань:

$$D = \frac{\sum_{i=1}^{N}(t_{di} - \overline{t_{di}})((t_{err\_i} - \overline{t_{err\_i}})}{\sqrt{\sum_{i=1}^{N}[(t_{di} - \overline{t_{di}})^2 (t_{err\_i} - \overline{t_{err\_i}})^2]}}, \qquad (3.9)$$

Засоби діагностики фіксують показники із встановленим періодом. Слід зазначити, що з недостатньому обсязі тестових випробувань ймовірність прояви невиявлених потенційно небезпечних системи дефектів значно підвищується,

тому важливо визначити критерій закінчення тестування. Існують різні критерії завершення випробувань, які забезпечують виконання вимог до надійності функціонування системи [1, 4, 7]. Алгоритми тестування програмно-апаратних пристроїв часто використовують такий показник надійності, як напрацювання на відмову, визначаючи час проведення випробувань, але не враховуючи при цьому особливості програмного забезпечення, що реалізується на пристрої. Крім того, зазвичай алгоритми тестування не беруть до уваги характер зміни роботи програмного забезпечення, викликаного зміною вхідних даних, а також технічні особливості відмов і збоїв, що виявляються. Алгоритм виконання програми випробувань ілюструє рисунок 3.14.

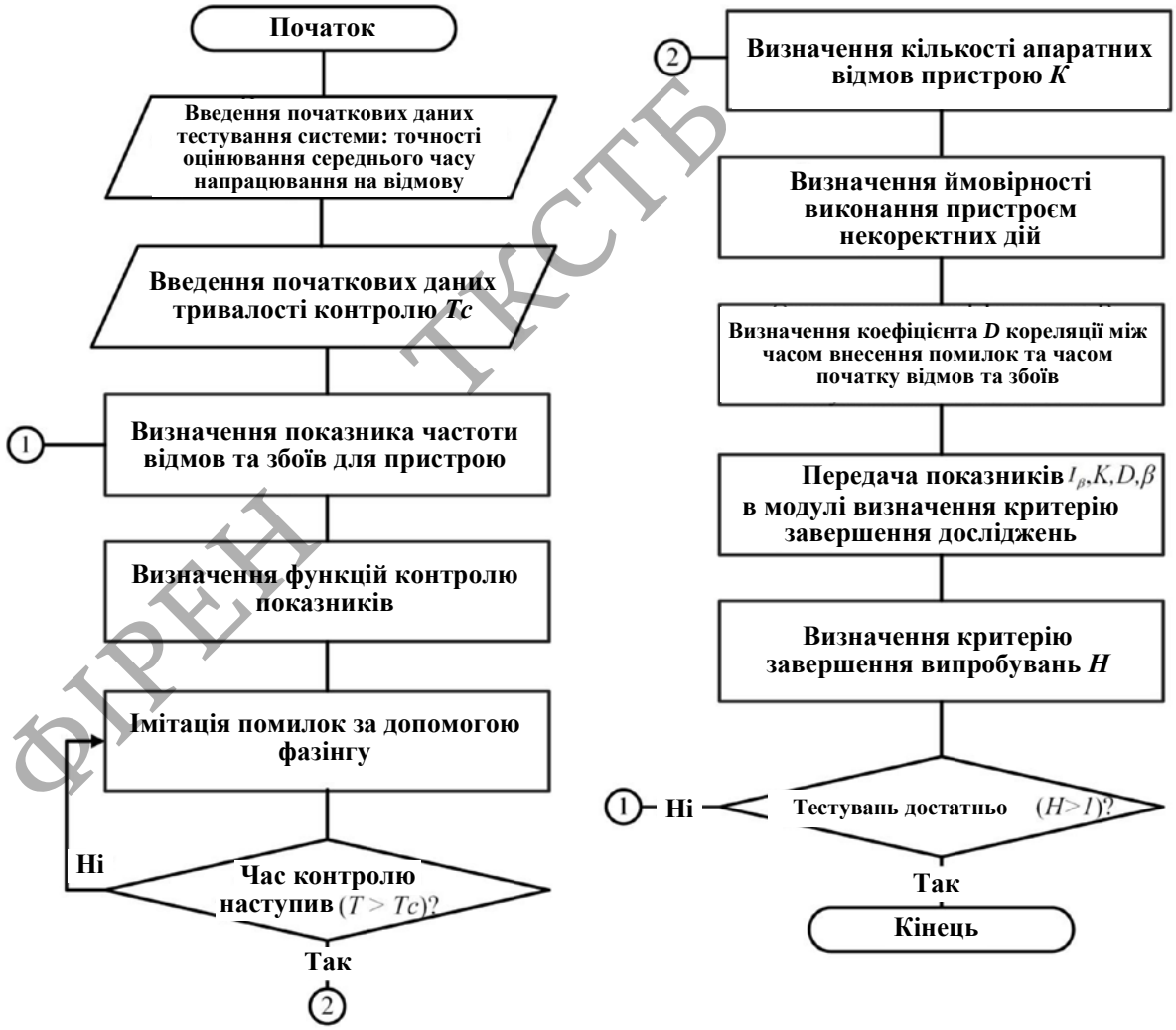


Рисунок 3.14 - Алгоритм виконання програми випробувань пристрою

Для імітації експлуатації пристрою в реальних умовах здійснюється внесення помилок модулем фазінгу, що спотворює вхідні дані пристрою під час роботи в штатних режимах.

Спотворення вхідних даних дозволяють виявити помилки при виконанні арифметичних і логічних команд, і, крім того, спотворення послідовності виконання програми, що вносяться, призводять до помилок у командах переходу, які в свою чергу ведуть до відмов і збоїв.

Сформований у результаті роботи алгоритму критерій закінчення випробувань H дозволяє визначити необхідність подальшого проведення випробувань (для обраного значення точності оцінки середнього часу напрацювання на відмову β).

Слід зазначити, що використання технології фазинг для тестування програм дозволяє здійснити автоматичний пошук помилок програмного забезпечення пристрою за допомогою спотворення вхідних даних і подальшого аналізу реакції пристрою. Для перевірки роботи програмного забезпечення пристрою за допомогою фазингу використані дві метрики: PATH_COV (кількість шляхів у програмі від загальної кількості, що приймає значення від 0% до 100%) та ERR_CNT (кількість отриманих помилок), найбільш інформативних для оцінки результатів застосування фазингу.

Критерій закінчення випробувань H формується під час роботи спеціалізованих програмних та апаратних засобів тестової інфраструктури випробувального комплексу [3, 4]:

$$H = \begin{cases} 0, \text{якщо} & \sum_{j=1}^{N} \sum_{i=1}^{M} Rule_i(I_\beta, D, K, \beta) \cdot M^{-1} \leq 1 \\ 1, \text{якщо} & \sum_{j=1}^{N} \sum_{i=1}^{M} Rule_i(I_\beta, D, K, \beta) \cdot M^{-1} < 1 \end{cases}, \quad (3.10)$$

де N – число програмних функцій; M – число продукційних правил обчислення критерію закінчення випробувань заданої функції; Rulei (Ib, D, K, b) результат обчислення критерію закінчення випробувань відповідно до і-го продукційного правила нечіткого виведення (обчислюється на основі показників).

Обсяг випробувань вважається достатнім у той момент, коли отримано перше позитивне значення критерію Н. Оскільки мікропроцесорні пристрої функціонують, використовуючи складне програмне забезпечення, пропонується формувати критерій закінчення тестування на основі аналізу статистичних даних про роботу апаратних і програмних засобів цих пристроїв.

Для визначення обсягу тестових випробувань мікропроцесорних пристроїв розроблено спеціальну програму тестування, в якій передбачено визначення моменту завершення випробувань. При цьому встановлюється, що обсяг проведених випробувань достатній на основі введеного для цього критерію закінчення випробувань. Після розрахунків та фіксації засобами діагностики показники обробляються та обчислюється значення критерію закінчення випробувань. Алгоритм обчислення з урахуванням вирішальних правил нечіткого логічного висновку [6, 8] за чотирма показниками визначає критерій закінчення випробувань. Отримане на виході даного алгоритму рішення є чисельним значенням критерію, яким визначається, випробування проведено повному обсязі або їх продовжити. Технічний комплекс, призначений для проведення випробувань, є програмно-апаратною системою, що використовується для виявлення дефектів розробленого пристрою перед проходженням приймальних випробувань. Комплекс містить інфраструктуру тестування, яка налаштовується для перевірки працездатності пристрою у необхідних режимах роботи, а також виконання штатних функцій з технічного завдання цього пристрою. Мікропроцесорний пристрій, як правило, при функціонуванні здійснює реєстрацію всіх операцій, що виконуються. Для кожного типу пристроїв склад і структура комплексу проведення випробувань відрізняється, оскільки відрізняються схемотехнічні рішення, інтерфейси і програмне забезпечення пристроїв, тому при розробці пристрою зазвичай проектується і комплекс проведення випробувань, здатний перевіряти функції даного пристрою.

Розглянемо розроблений технічний комплекс проведення тестових випробувань для влаштування прийому та передачі на базі процесорів загального

призначення за допомогою інтерфейсів граничного сканування JTAG. Інфраструктура тестування реалізує фіксацію реакції системи на імітацію помилок. По реакції системи встановлюється, чи викликані відмова або збій системи відсутністю захисту від спотворених даних у програмі чи помилка, що вводиться, повинна призводити до такої відмови (збою).

Аналіз реакції пристрою полягає в періодичному опитуванні через інтерфейс JTAG стану пристрою та передачі на комп'ютер, на якому функціонують програмні модулі тестування пристрою для розрахунку показників Iβ, β, K, D. Отримані на пристрої, що тестується, показники передаються модулю закінчення випробувань, де визначається чисельне значення критерію закінчення випробувань H. Модуль закінчення випробувань реалізований як окрема підпрограма, та його робота заснована на апараті нечіткої логіки; модуль виконаний серед MATLAB / Fuzzy Logic Toolbox [15, 16, 17].

У побудованій системі на входи нечіткого контролера надходять чотири сигнали: перший сигнал - це ймовірність виконання і-ї функції у момент настання відмови чи збою (pi). Другим вхідним сигналом є кількість фіксацій апаратних відмов та збоїв при виконанні і-ї функції (лінгвістична змінна K). Третім вхідним сигналом є встановлена точність обчислень системи (лінгвістична змінна βi). Четвертим вхідним сигналом є встановлена кореляція часу між спотворенням в і-ї функції та проявом відмов та збоїв (лінгвістична змінна D).

На виході нечіткого контролера видається значення критерію закінчення випробувань (лінгвістична змінна H), основі якого формується рішення про закінчення випробувань. Відповідно до отриманого значення імітація для мікроконтролерного пристрою або продовжується, або зупиняється з фіксацією поточних значень контрольованих показників.

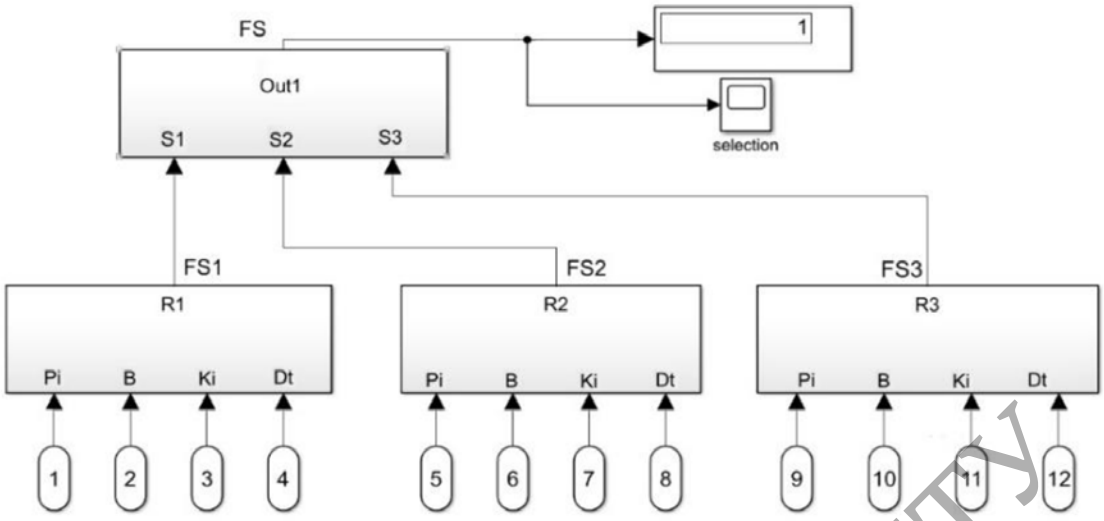


Рисунок 3.15 - Визначення критерію закінчення тестування пристрою

Модуль у процесі тестування з урахуванням показників роботи пристрою формує значення критерію закінчення випробувань. Для цього отримані в результаті імітації несправності чисельні значення показників використовують як вхідні дані для алгоритму роботи нечіткого логічного висновку. На малюнку 3.15 представлено схему обчислення критерію закінчення тестових випробувань (в модулі визначення закінчення випробувань) програмно-апаратного пристрою. Для реалізації алгоритму нечіткого логічного висновку вхідні змінні показників Iβ, β, K, D перекладаються значення нечітких лінгвістичних змінних блоками фазифікації MF1 - MF4. За виконання фазифікації число, що належить безлічі дійсних чисел, представляється як нечіткого числа (від англ. fuzzy – нечіткий) [11].

Відповідно до прийнятої логіки алгоритму, база знань нечіткого регулювання включає десять нечітких продукційних правил. При реалізації продукційних правил для отримання нечіткої множини вихідної змінної виконувались операції імплікації та агрегування. До кожного правила бази знань імплікація виконувалася з допомогою операції мінімуму.

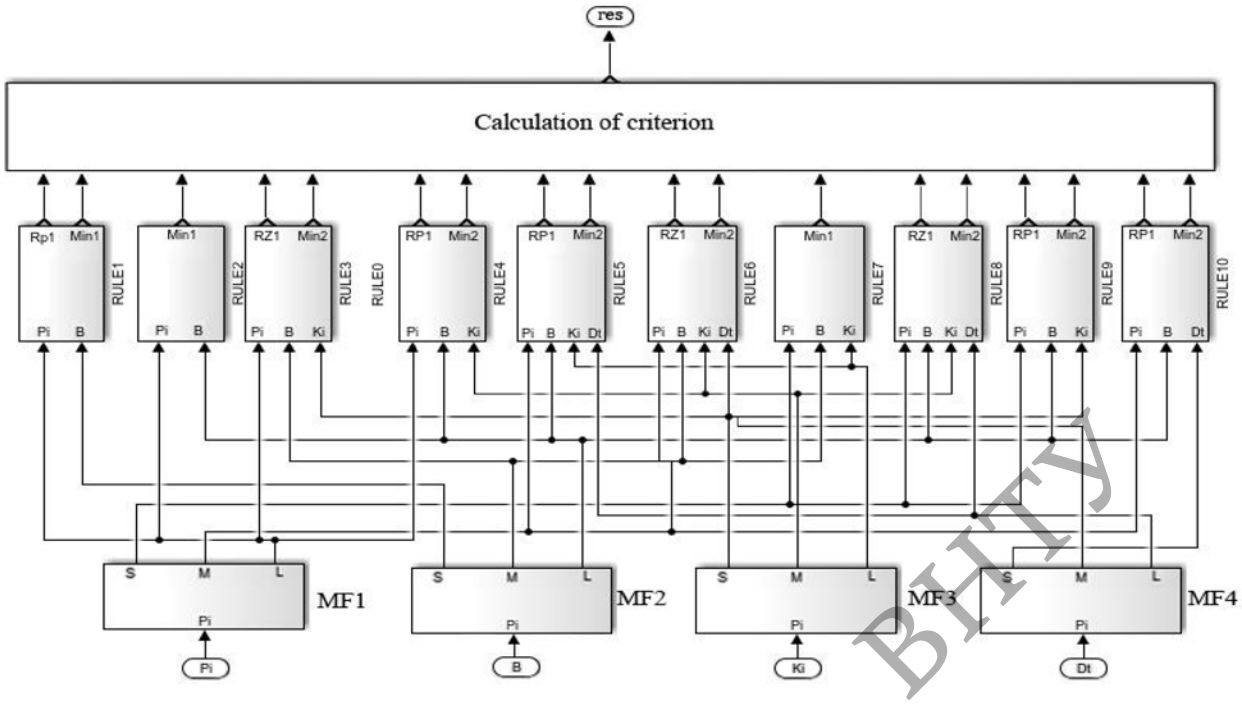


Рисунок 3.16 - Схема обчислення критерію закінчення випробувань нечіткого логічного висновку

Результат логічного висновку по всій базі знань виявляється агрегуванням нечітких множин з використанням операції максимуму. До кожного вирішального правила складається схема, реалізована з допомогою коштів Matlab (рис 3.17). Для фазифікації вхідних змінних (показників x=(Ib, β, K, D)) діапазони їхньої зміни розбиваються на лінгвістичні терми (на основі експертних оцінок). При цьому для кожного показника використовується три терми значень: S – мале, M – середнє, L – велике. Для кожного з термів будується функція приналежності μ(x) змінної x цього терму. Для завдання внутрішніх лінгвістичних термів (M – середнє) вхідних змінних використана симетрична гауссова функція приналежності (gaussmf), що формується відповідно до виразу

$$\mu(x) = e^{-(x-c)^2/(2\sigma)^2}, \qquad (3.11)$$

де параметр c визначає модальне значення функції, а σ – ширину.

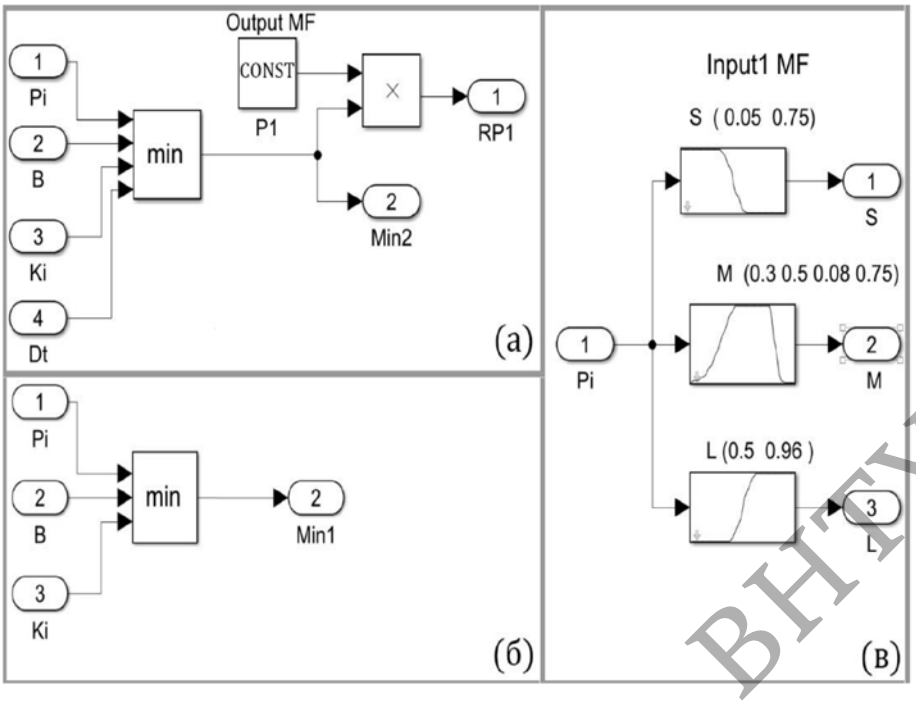


Рисунок 3.17 - Модель випробування пристрою: а) правило нечіткого висновку для позитивного рішення; б) правило нечіткого висновку для негативного рішення; в) підсистема фазифікації

Для завдання крайніх термів (S – мале та L – велике) використовуються сигмоїдні функції приладдя (sigmf), які визначаються відповідно до виразу:

$$\mu(x) = (1 + e^{-a(x-c)})^{-1}, a < c, \qquad (3.12)$$

Параметр с визначає координату точки перегину функції, а коефіцієнт а задає нахил функції у цій точці. При цьому a>0 використовується для S-подібної функції, а a<0 реалізується Z-подібна функція приналежності.

Висновок кожного правила визначається у вигляді чіткого числа критерію закінчення випробувань H, яке реалізовано як одноелементне нечітке безліч з точковою функцією приналежності. Діапазон зміни змінної H також розбивається на три терми: P_L – позитивне, P_Z – нейтральне, P_N – негативне (відповідно до належності значення критерію цим термам приймається рішення про закінчення випробувань).

Далі проводиться визначення чіткого числа критерію закінчення випробувань H шляхом виконання процедури дефазифікації (зворотного перетворення нечітких змінних на чіткі). Обчислення критерію закінчення випробувань H здійснюється шляхом визначення зваженого середнього відповідно до виразу [12, 17]:

$$H = \sum_{i=1}^{m} \mu(H) H_i (\sum_{i=1}^{m} \mu(H))^{-1}. \quad (3.13)$$

Тут Hi - значення вихідний змінної для і-го терму з одиничним значенням ступеня належності; μ(H) – ступінь приналежності до цього терму; m – число термів. Для здійснення нечіткого логічного висновку використовується сукупність нечітких правил, що реалізують на основі нечітких змінних (для показників Ib, β, K, D) обчислення критерію закінчення випробувань H за формулою (5). Правила нечіткого виведення RULE1 – RULE10, які реалізуються кожної з контрольованих програмних функцій.

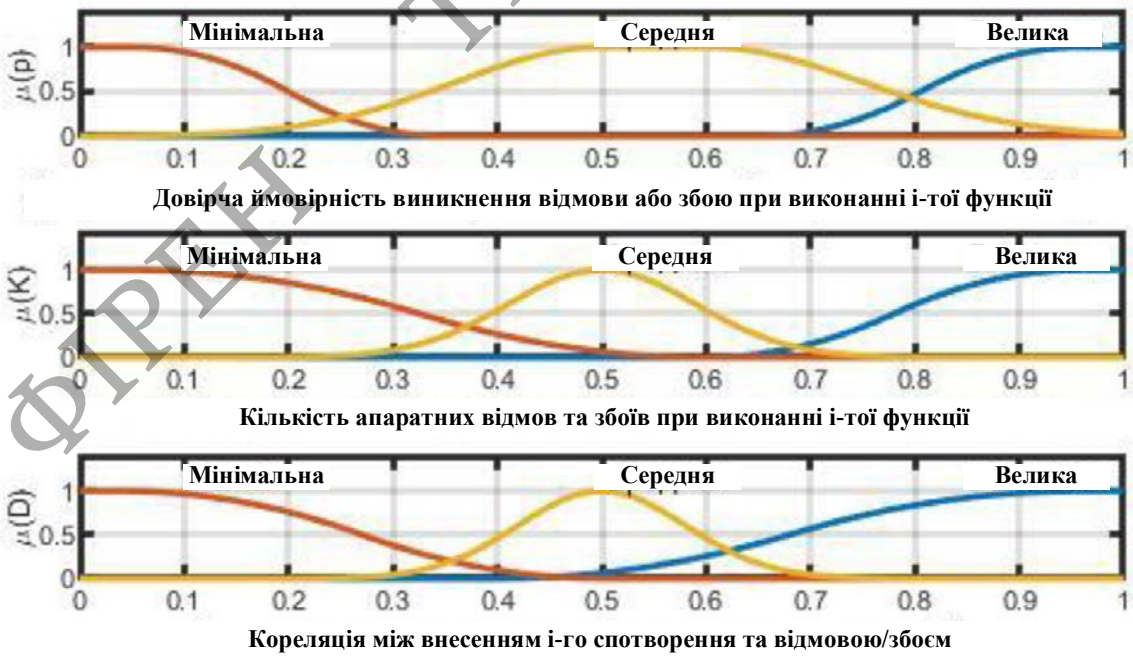


Рисунок 3.18 - Функції залежності для термів вхідних змінних, отримані в результаті експертних оцінок

Відбір програмних функцій контролю показників здійснювався в такий спосіб, щоб внесення помилок за її виконанні було найбільш значуще до роботи устройства.

Під час тестування розглядалися два режими роботи пристрою: режим прийому даних R1 та режим оновлення даних R2. Для режиму прийому даних вибрано три функції N = 3 (f1 – функція прийому, f2 – функція встановлення з'єднання, f3 – функція налаштування). Для режиму оновлення даних вибрано п'ять функцій N = 5 (f1 – функція прийому, f2 – функція встановлення з'єднання, f3 – функція налаштування, f4 – функція перевірки даних, f5 – функція оновлення даних).

Оскільки вихідні дані від пристрою передаються модуль закінчення випробувань нерівномірно за часом (у зв'язку з затримками в інтерфейсі передачі інформації), то в рамках програми випробувань передбачено, що обчислені показники передаються для формування критерію закінчення випробувань за встановлений часовий інтервал [15, 19]. Розподіл процесорного часу між програмними функціями для режимів роботи відображено у стовпці «Завантаження процесора» визначає час виконання процесором обраної функції.

Параметр «Адреса ОЗП» – розміщення функції в оперативній пам'яті пристрою, що тестується. Стовпець таблиці "Точка виклику функції" визначає назву функції, яка зустрічається у вихідному коді програми та фіксується засобами інструментування програмного коду.

3.5 Висновки до розділу 3

1. Розроблено структуру комплексу для проведення тестових випробувань, що дозволяє використовувати пристрій для імітації несправності для автоматичного пошуку помилок у програмно-апаратному забезпеченні з використанням техніки фаззингу. Показано, що стратегія фаззингу, що застосовується, передбачає на кожному етапі зміни даних (перестановка бітів і байтів, арифметичні операції) аналіз реакції програми на внесені спотворення.

Це дозволяє забезпечити повноту дослідження коректності роботи програмних функцій у реальних режимах роботи устройства.

2. Запропоновано спосіб та алгоритм передобробки вихідних даних з визначенням режимів випробування пристрою, функцій, що перевіряються, і точок контролю, які забезпечують прискорення фаззингу за рахунок підготовки даних для дослідження графа розгалужень на основі найбільш часто використовуваних функцій. Такий спосіб дозволяє здійснити ефективний контроль найбільш уразливих місць програмного забезпечення.

3. Модифіковано алгоритм внесення несправності за допомогою техніки фаззингу та розроблено алгоритм внесення ін'єкцій відмов та збоїв у мікропроцесорний пристрій. Відмінність запропонованого алгоритму від існуючих полягає у використанні інтерфейсу JTAG при аналізі реакції на імітовані несправності в процесі обходу графа розгалужень програми (за допомогою матриці розгалужень, що генерується). Основна перевага модифікованого алгоритму обумовлена використанням передоброблених статистичних показників як вхідні дані, що дозволяє підвищити повноту дослідження основних програмних функцій та прискорити виконання їх перевірки.

4. Розроблено метод та алгоритм визначення обсягу тестових випробувань на основі нечіткого логічного висновку. Запропоновано критерій визначення обсягу тестових випробувань мікропроцесорних пристроїв. Критерій, заснований на попередній статистичній обробці експериментальних даних про функції, що виконуються в сукупності з алгоритмом прийняття рішення на базі нечіткого логічного висновку, дозволяє оцінити достатність обсягу випробувань.

5.Створено математичну модель для обґрунтування критерію, що враховує попередньо отримані характеристики пристрою: кількість апаратних відмов, ймовірність помилок при виконанні програмних функцій, а також кореляцію між часом внесення помилок у вхідні дані та часом виявлення несправності.

4 РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ ПРИСТРОЇВ ІНФОРМАЦІЙНОГО КОМПЛЕКСУ

4.1 Організація досліджень інформаційного комплексу

Стенд для тестування [5, 6] складається з пристрою, АРМ оператора та пристрою імітації несправності та аналізу реакції пристрою на внесені спотворення (рис. 4.1). У режимі внесення спотворень було здійснено виявлення режимів, де виявлялися відмови та збої. Після цього було запущено процес роботи алгоритму передобробки та подальша імітація несправності. Пунктирним контуром позначена структура пристрою, що тестується. Пристрій, що тестується, підключено до автоматизованого робочого місця оператора. АРМ оператора підключається через інтерфейс сканування блоків пристрою (JTAG) до висновків модулів зчитування стану [8]. Для цілей нашого дослідження ця функція використана як можливість спотворювати дані та стан регістрів у процесі роботи пристрою.

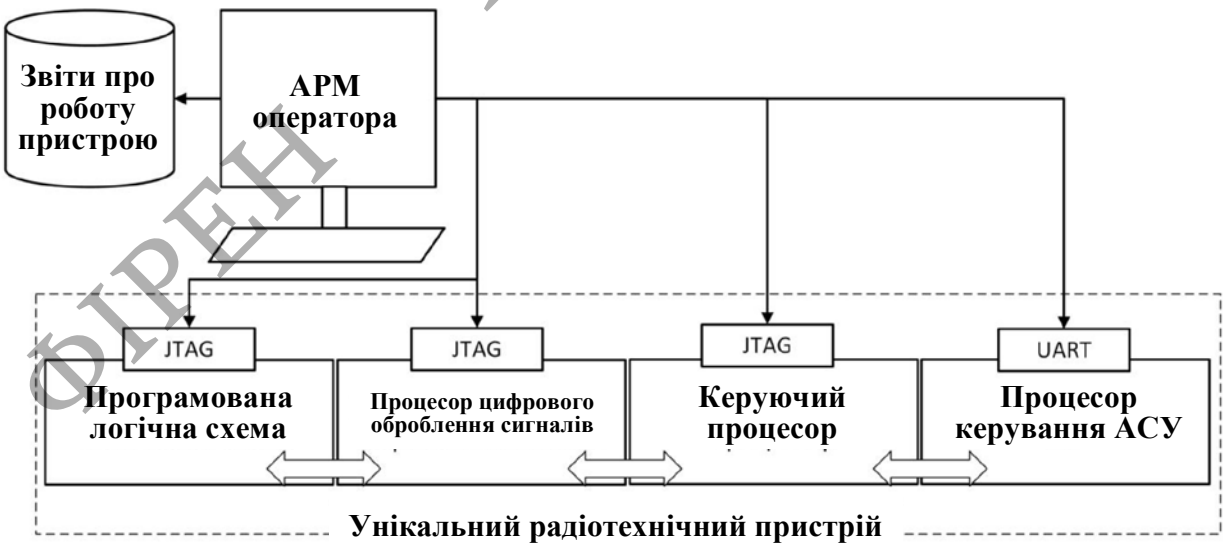


Рисунок 4.1 – Структурна схема імітаційних досліджень

Верифікація імітатора несправності проводилася за допомогою двох режимів організації випробувань: псевдовипадкового спотворення даних у процесі роботи пристрою та спотворення даних на основі отриманих відомостей про статистику використання функцій та асемблерних команд у режимах роботи.

Модулі апаратної фіксації відмов і збоїв були верифіковані на мікропроцесорних пристроях, які мають відомі апаратні проблеми. В результаті перевірки відмови та збої були верифіковані, крім модуля контролю змін, що вносяться. Використання апаратних модулів разом із класичним тестуванням нових результатів не дало.

4.2 Опис програмних засобів випробувального комплексу

Розробка програми пристрою здійснюється на робочій станції випробувального комплексу. Програма, розроблена мовою C/C++ транслюється у виконуваний файл (формат файлу – elf), який з допомогою налагодження завантажується засобами налагодження і діагностики в мікропроцесорний пристрій.

Файл вихідного коду програми може бути модифікований як у процесі виконання пристрою, так і в процесі розробки (рис. 4.2). Модифікація даних проводиться у процесі використання імітатора у секції text, data, rodata. Програмні засоби інфраструктури реалізують алгоритм управління пристроєм імітації несправності і дозволяють виконувати тестування за рахунок аналізу реакції програми у певні часові інтервали або виникнення апаратних подій. При цьому виконується зчитування доступних регістрів та оперативної пам'яті. Це дозволяє проводити комплексний аналіз стану пристрою на робочій станції.

Засоби програмування (STM32 Workbench, Eclipse та інші) містять можливість додавати контрольні точки («breakpoint») для збору шаблонної інформації про виконання програмних функцій.

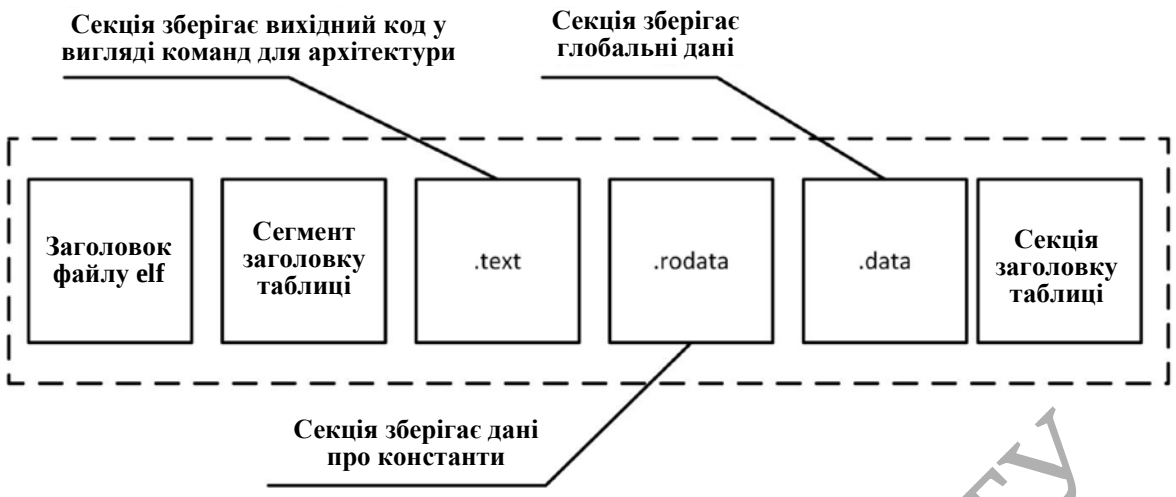


Рисунок 4.2 - Файл вихідного коду

Мікропроцесорні модулі використовують типову схему підключення (загальну для систем, що вбудовуються) з апаратною підтримкою подій завантаження поточних значень регістрів і оперативної пам'яті.

Рисунок 4.3 - Структура програмно-апаратних компонентів налагодження

Програма розробки через плагіни дозволяє призначити точку зупинки будь-яку інструкцію програмного модуля. Точці зупинки відповідає асемблерної інструкції всередині програмної функції. Пошук адреси інструкції, яка має бути модифікована, здійснюється за допомогою файлу компонування для мікропроцесорного пристрою (map-файл). Файл містить опис розміщення функцій у flash-пам'яті мікроконтролера.

Спотворення вхідних даних для досліджуваного файлу здійснюється за допомогою впровадження в регістровий файл (R01-R03) та ОЗП (D01-D03). Підготовлена послідовність даних передається за інтерфейсом JTAG і міститься

у пам'ять пристрою чи регістри. Послідовна перевірка коректності інструкції за допомогою відомих результатів роботи в точках розгалужень програми після збору інформації про статистику використання функцій у режимах прийому та передачі реалізується у вигляді алгоритму, який у кожний момент роботи перевіряє результати інструкцій та підготовляє новий вид спотворення, який залежить від поточного типу команди вивчається функції.

Це дозволяє здійснювати одразу кілька видів спотворень (рис. 4.7): − спотворення з метою порушення команди пересилання передбачає зміну даних із потів введення-виводу, модифікацію адрес; − спотворення з метою порушення команди логіки передбачає зміну умови розгалуження програми для перевірки коректності роботи алгоритму з даними;

− спотворення з метою порушення команди переходу передбачає модифікацію адреси переходу для перевірки коректності роботи програми; − спотворення з метою виклику (маскування) переривання передбачає пропуск або несвоєчасну реакцію програми пристрою на апаратні події чи події інтерфейсів;

− спотворення операцій введення-виведення передбачає спотворення даних інтерфейсів для перевірки коректного функціонування у спільних програмних модулях.

Ін'єкція відмов та збоїв здійснюється за допомогою модифікації параметрів, встановлених функцій бібліотек, які використовуються для трасування, пошуку та модифікації ділянок програми в автоматичному та напівавтоматичному режимах. Для ін'єкції потрібно встановити місце в програмі, час, вміст ін'єкції [5]. Модифікація відповідно до таблиці проводиться за допомогою трьох основних варіантів: використання синхронізації від зовнішнього імпульсу, що генерується портом вводу-виводу щодо виникнення контрольованої події, зупинка процесора, модифікація поточних даних, що виконуються, і відновлення роботи процесора; використання порту внутрішньокристального відладчика JTAG та виконання дій попереднього варіанту із синхронізацією по апаратній точці зупинки процесора; генерація відмов та збоїв без прив'язки до програмно-апаратної роботи пристрою.

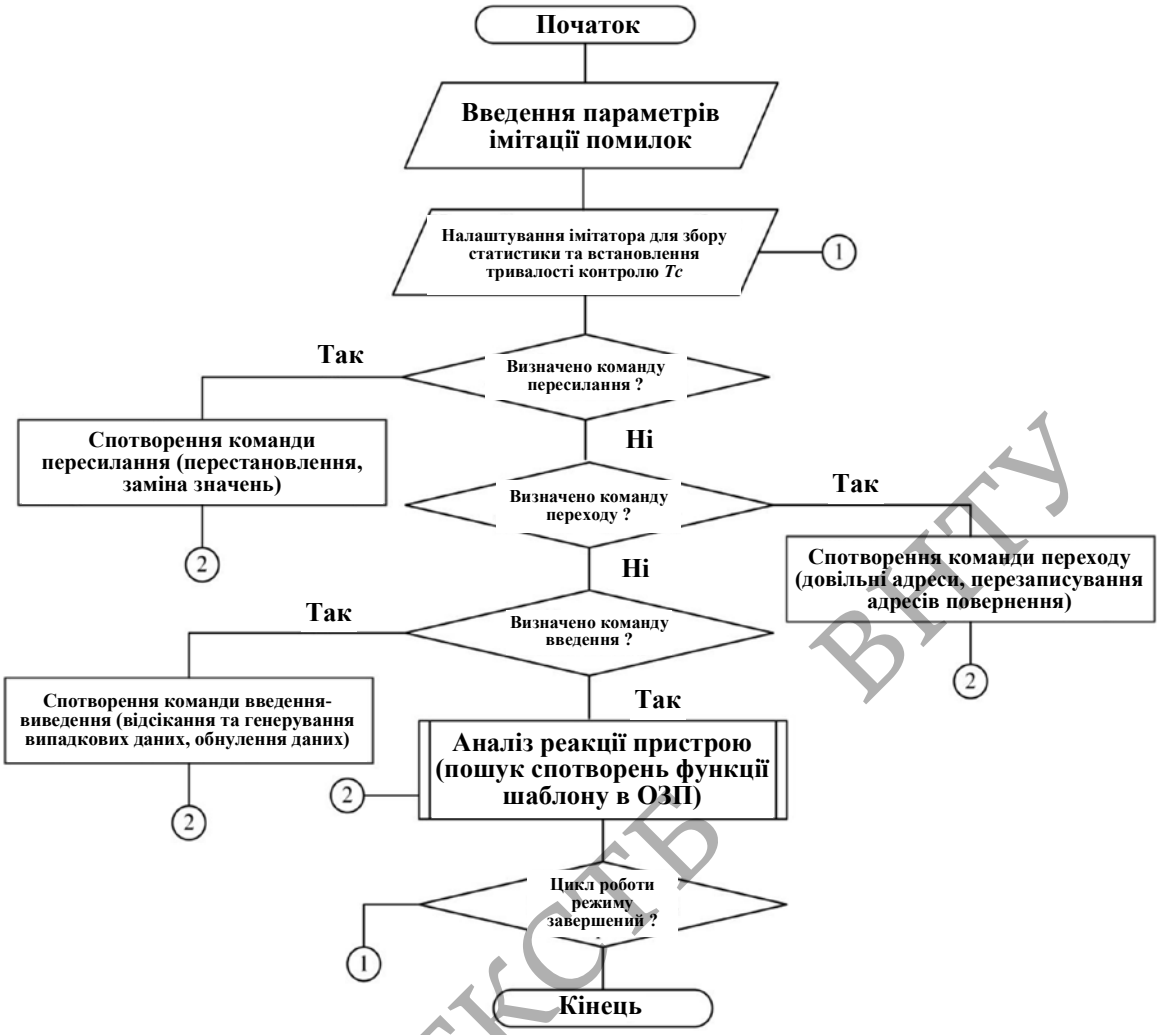


Рисунок 4.5 - Алгоритм внесення спотворень REG/DATA - типу у програму

Трасування програми було виконано за допомогою засобів відладчика за допомогою компіляторів GDB/Clang.

4.3 Результати імітації та виявлення несправності

При проведенні тестових випробувань роботи пристрою інформаційного комплексу та імітації помилок за допомогою технології фазінг визначено значення контрольованих показників пристрою.

На основі аналізу показників для режимів прийому даних R1 та оновлення даних R2 сформовані інтервали зміни показників, усередині яких забезпечуються вимоги до критерію закінчення випробувань (для обраного

значення точності оцінки середнього часу напрацювання на відмову β). На основі отриманих значень критерію закінчення випробувань H приймається рішення щодо необхідності припинення випробувань. На основі експертних оцінок, а також виходячи з умов експлуатації системи та вимог технічного завдання приймалося значення необхідного показника β=80%, решта параметрів розраховувалася за результатами роботи засобами інфраструктури тестування в рамках програми випробувань. Як можна бачити, критерій закінчення випробувань пристрою значною мірою залежить від наявності апаратних відмов (значення параметра K), що враховується при формуванні рішення про завершення випробувань. З іншого боку, для формування критерію закінчення випробувань враховується коректність роботи програмного забезпечення. Аналіз значення параметра D дозволяє виявляти зміни при виконанні алгоритму роботи (відхилення від роботи за штатним алгоритмом) під впливом спотворених даних, що вносяться в пристрій. Результати тестування програмного забезпечення пристрою ілюструє рисунок 4.8. Стовпці значень показників містять припустимі інтервали, виділені сірим кольором. Для кожної функції, що перевіряється, приймається рішення про закінчення тестування на основі величини ступеня належності критерію до безлічі значень, відповідних безлічі позитивних рішень. Загальний критерій закінчення тестування всього пристрою визначається на основі результатів, отриманих при роботі модуля закінчення випробувань для всіх функцій, і це рішення подається на дисплей. Значення критерію закінчення тестування ілюструється кольором: світлий колір – достатньо для завершення випробувань, сірий – не визначено, темний колір – недостатньо для завершення випробувань. Як бачимо, критерій H, який відповідає завершенню випробувань, відповідає об'єднанню допустимих інтервалів всім показників (сірий колір). Світло-сірим та темним кольором, відповідно, показані значення критерію H, для яких випробування мають бути продовжені.

Слід зазначити, що зі збільшенням кількості функцій зростає кількість обмежень для імітації несправності, тому час проведення випробувань значно зростає. Як показали результати дослідження, обрані показники роботи

пристрою є можливим застосовувати для тестування та інших пристроїв з цієї партії при серійному виробництві (ідентичних в частині програмно-апаратних рішень), використовуючи пропонований критерій для закінчення випробувань при серійному тестуванні.

Крім того, в результаті досліджень встановлено, що зі зростанням числа виконуваних функцій кожен із чотирьох показників окремо (у зв'язку зі складністю визначення діапазонів їх зміни для великої кількості функцій) стає недостатньо інформативним для прийняття рішення про закінчення випробувань, тому виникає необхідність вибору показників, що характеризують достатність обсягу проведених випробувань для перевірки коректного виконання функцій, можливо, на основі спільної оцінки прийнятих для дослідження показників.

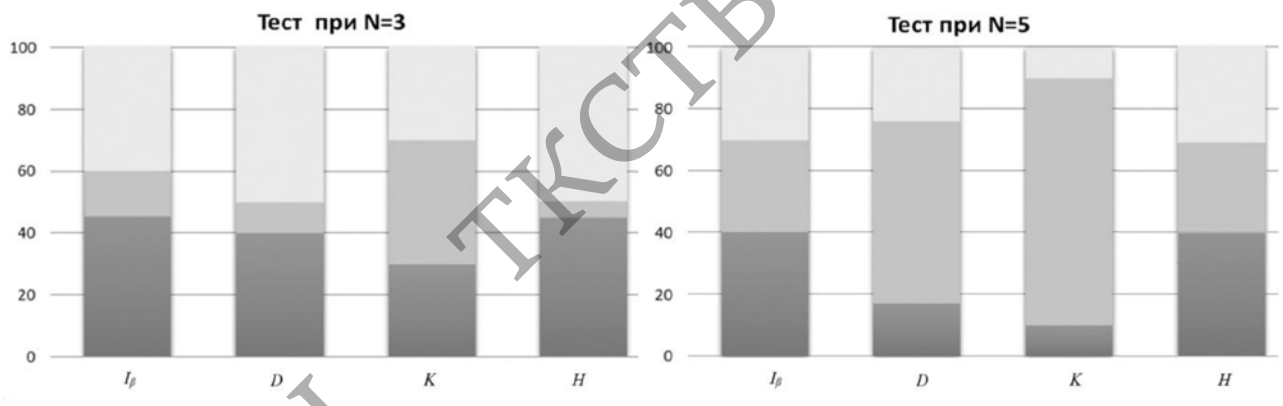


Рисунок 4.6 - Результати тестування програмного забезпечення пристрою

Аналіз переваг запропонованого підходу показав, що використання запропонованої методики не менше ніж на 5% скорочує обсяг випробувань для трьох функцій (тестування пристрою в режимі прийому даних) і не менш ніж на 10% – для п'яти функцій (тестування пристрою в режимі оновлення даних) порівняно із класичним тестуванням. Для імітації несправності випробування в процесі тестування було визначено три режими, які вимагали підвищення стійкості до відмов та збоїв внаслідок класичного тестування: режим передачі інформації в режимах роботи пристрою (Р1);

– режим обміну інформацією з іншими пристроями комплексу (Р2); – режим налаштування та діагностики виробу з робочої станції (Р3).

У зв'язку з отриманою проблемою детектування та подальшого усунення помилок програмного забезпечення проведено інтенсивну імітацію із застосуванням показників, отриманих за допомогою критерію закінчення випробувань. Наведено дані дослідження мікропроцесорного пристрою за допомогою алгоритму визначення критерію закінчення випробувань. Для режимів роботи пристрою визначено значення показників, які характеризують набір необхідного обсягу досягнення працездатності.

На рисунку 4.9 наведено результати порівняння часу пошуку несправності за допомогою стандартних засобів та під час використання випробувального стенду з імітацією несправності програмними засобами.

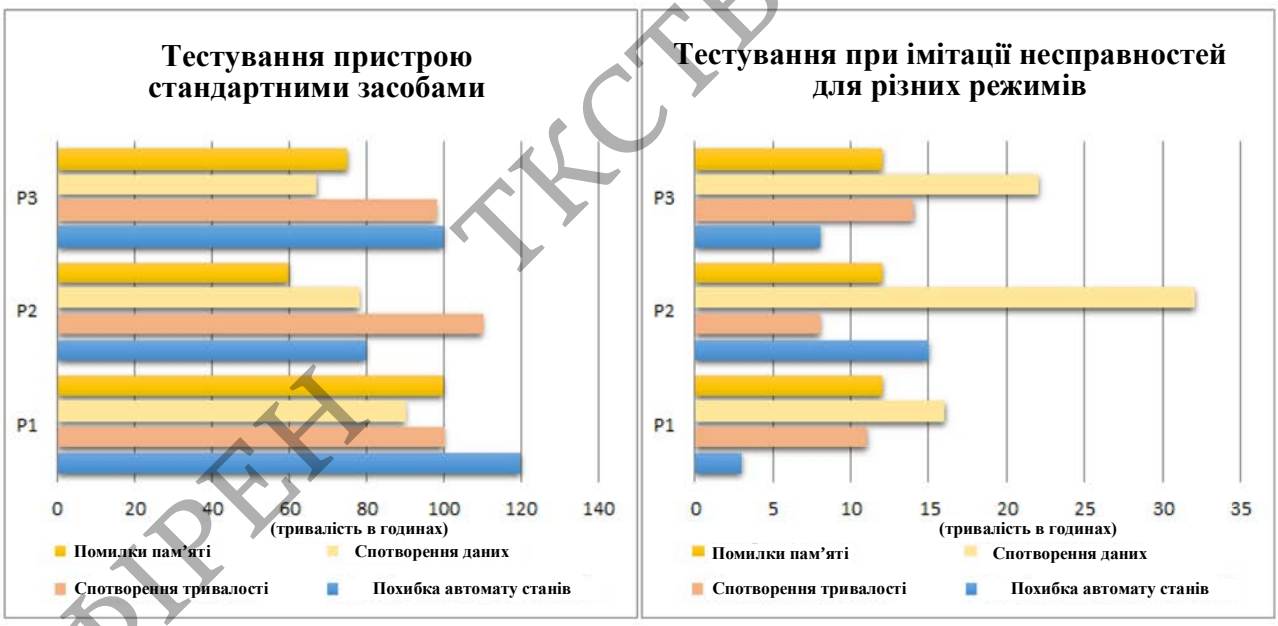


Рисунок 4.9 - Пошук помилок програмного забезпечення для режимів Р1-Р3

Слід зазначити, що для всіх режимів випробувань пошук помилок програмного забезпечення зайняв значно менше часу. Максимальний час пошуку для імітації несправності – 32 години, а для ручного пошуку – 120 годин. Результати пошуку помилок за допомогою стенду демонструють, що порівняно

з ручним пошуком кількість виявлення значно вища (на 82% від загальної кількості виявлених помилок).

4.4 Висновки до розділу 4

1. Отримано результати випробувань на розробленому макеті імітатора несправності для перевірки його функцій щодо внесення спотворень у вхідні дані та виявлення відмов та збоїв за допомогою аналізу реакції мікропроцесорного пристрою на помилки, що ін'єктуються. Встановлено, що незалежне застосування імітатора дозволяє забезпечити підвищення кількості знайдених помилок, здатних призводити до відмов та збоїв (від 5% до 19% виявлених помилок від кількості виявлених за допомогою методів класичного тестування програмного забезпечення).

2. За допомогою розроблених засобів програмного комплексу імітації системних подій та автоматизації здійснення випробувань виконані роботи з визначення основних функцій програмного забезпечення та статистики розподілу часу використання функцій у режимах роботи мікропроцесорного пристрою.

3. Встановлено, що імітація несправності із спрямованим фазингом ефективніша порівняно з використанням класичного тестування (179 випадків виявлення проявів помилок ПЗ замість 31). Запропоновані алгоритми імітації несправності та аналізу реакції також дозволили виявити помилки програмного забезпечення для прототипу швидше, ніж за допомогою застосування класичних методів тестування (32 години замість 120).

4. Отримані значення показників для позитивного критерію закінчення випробувань за допомогою методики на основі нечіткого логічного висновку, що характеризує необхідний обсяг тестових випробувань мікропроцесорного пристрою, відповідають стану працездатності мікропроцесорного пристрою.

5 ЕКОНОМІЧНА ЧАСТИНА

Виконання науково-дослідної роботи завжди передбачає отримання певних результатів і вимагає відповідних витрат. Результати виконаної роботи завжди дають нам нові знання, які в подальшому можуть бути використані для удосконалення та/або розробки (побудови) нових, більш продуктивних зразків техніки, процесів та програмного забезпечення.

Дослідження на тему «Методи тестування програмно-керованих інфокомунікаційних систем» може бути віднесено до фундаментальних і пошукових наукових досліджень і спрямоване на вирішення наукових проблем, пов’язаних з практичним застосуванням. Основою таких досліджень є науковий ефект, який виражається в отриманні наукових результатів, які збільшують обсяг знань про природу, техніку та суспільство, які розвивають теоретичну базу в тому чи іншому науковому напрямку, що дозволяє виявити нові закономірності, які можуть використовуватися на практиці.

Для цього випадку виконаємо такі етапи робіт:

1) здійснимо проведення наукового аудиту досліджень, тобто встановлення їх наукового рівня та значимості;

2) проведемо планування витрат на проведення наукових досліджень;

3) здійснимо розрахунок рівня важливості наукового дослідження та перспективності, визначимо ефективність наукових досліджень.

5.1 Оцінювання наукового ефекту

Основними ознаками наукового ефекту науково-дослідної роботи є новизна роботи, рівень її теоретичного опрацювання, перспективність, рівень розповсюдження результатів, можливість реалізації. Науковий ефект НДР на тему «Методи тестування програмно-керованих інфокомунікаційних систем» можна охарактеризувати двома показниками: ступенем наукової новизни та рівнем теоретичного опрацювання.

Значення показників ступеня новизни і рівня теоретичного опрацювання науково-дослідної роботи в балах наведені в табл. 5.1 та 5.2.

Таблиця 5.1 – Показники ступеня новизни науково-дослідної роботи виставлені експертами

Ступінь новизни	Характеристика ступеня новизни	Значення ступеня новизни, бали		
		Експерти (ПІБ, посада)		
		1	2	3
Принципово нова	Робота якісно нова за постановкою задачі і ґрунтується на застосуванні оригінальних методів дослідження. Результати дослідження відкривають новий напрям в даній галузі науки і техніки. Отримані принципово нові факти, закономірності; розроблена нова теорія. Створено принципово новий пристрій, спосіб, метод	-	-	-
Нова	Отримана нова інформація, яка суттєво зменшує невизначеність наявних значень (по-новому або вперше пояснені відомі факти, закономірності, впроваджені нові поняття, розкрита структура змісту). Проведено суттєве вдосконалення, доповнення і уточнення раніше досягнутих результатів	45	45	59
Відносно нова	Робота має елементи новизни в постановці задачі і методах дослідження. Результати дослідження систематизують і узагальнюють наявну інформацію, визначають шляхи подальших досліджень; вперше знайдено зв'язок (або знайдено новий зв'язок) між явищами. В принципі відомі положення розповсюджені на велику кількість об'єктів, в результаті чого знайдено ефективне рішення. Розроблені більш прості способи для досягнення відомих результатів. Проведена часткова раціональна модифікація (з ознаками новизни)	-	-	-
Традиційна	Робота виконана за традиційною методикою. Результати дослідження мають інформаційний характер. Підтверджені або поставлені під сумнів відомі факти та твердження, які потребують перевірки. Знайдено новий варіант рішення, який не дає суттєвих переваг в порівнянні з існуючим	-	-	-
Не нова	Отримано результат, який раніше зафіксований в інформаційному полі, та не був відомий авторам	-	-	-
Середнє значення балів експертів		49,7		

Згідно отриманого середнього значення балів експертів ступінь новизни характеризується як нова, тобто отримана нова інформація, яка суттєво зменшує невизначеність наявних знань (по-новому або вперше пояснені відомі факти, закономірності, впроваджені нові поняття, розкрита структура змісту) та проведено суттєве вдосконалення, доповнення і уточнення раніше досягнутих результатів.

Таблиця 5.2 – Показники рівня теоретичного опрацювання науково-дослідної роботи виставлені експертами

Характеристика рівня теоретичного опрацювання	Значення показника рівня теоретичного опрацювання, бали		
	Експерт (ПІБ, посада)		
	1	2	3
Відкриття закону, розробка теорії	-	-	-
Глибоке опрацювання проблеми: багатоаспектний аналіз зв'язків, взаємозалежності між фактами з наявністю пояснень, наукової систематизації з побудовою евристичної моделі або комплексного прогнозу	60	65	60
Розробка способу (алгоритму, програми), пристрою, отримання нової речовини	-	-	-
Елементарний аналіз зв'язків між фактами та наявною гіпотезою, класифікація, практичні рекомендації для окремого випадку тощо	-	-	-
Опис окремих елементарних фактів, викладення досвіду, результатів спостережень, вимірювань тощо	-	-	-
Середнє значення балів експертів	61,7		

Згідно отриманого середнього значення балів експертів рівень теоретичного опрацювання науково-дослідної роботи характеризується як глибоке опрацювання проблеми: багатоаспектний аналіз зв'язків, взаємозалежності між фактами з наявністю пояснень, наукової систематизації з побудовою евристичної моделі або комплексного прогнозу.

Показник, який характеризує рівень наукового ефекту, визначаємо за формулою [20]:

$$E_{нау} = 0,6 \cdot k_{нов} + 0,4 \cdot k_{теор}, \quad (5.1)$$

де $k_{нов}, k_{теор}$ - показники ступеня новизни та рівня теоретичного опрацювання науково-дослідної роботи, $k_{нов} = 49,7, k_{теор} = 61,7$ балів;

0,6 та 0,4 – питома вага (значимість) показників ступеня новизни та рівня теоретичного опрацювання науково-дослідної роботи.

$$E_{нау} = 0,6 \cdot k_{нов} + 0,4 \cdot k_{теор} = 0,6 \cdot 49,7 + 0,4 \cdot 61,67 = 54,47 \text{ балів.}$$

Визначення характеристики показника $E_{нау}$ проводиться на основі висновків експертів виходячи з граничних значень, які наведені в табл. 5.3.

Таблиця 5.3 – Граничні значення показника наукового ефекту

Досягнутий рівень показника	Кількість балів
Високий	70…100
Середній	50…69
Достатній	15…49
Низький (помилкові дослідження)	1…14

Відповідно до визначеного рівня наукового ефекту проведеної науково-дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем», даний рівень становить 54,47 балів і відповідає статусу - середній рівень. Тобто у даному випадку можна вести мову про потенційну фактичну ефективність науково-дослідної роботи.

5.2 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

5.2.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників. Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [20]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (5.2)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=21 дні.

$$З_о = 11350{,}00 \cdot 21 / 21 = 11350{,}00 \ грн.$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник	11350,00	540,48	21	11350,00
Інженер-розробник телекомунікаційних систем	10652,00	507,24	21	10652,00
Технік	6750,00	321,43	10	3214,29
Всього				25216,29

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Методи тестування програмно-керованих інфокомунікаційних систем» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i , \qquad (5.3)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}}, \qquad (5.4)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $Мм$=2379,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [20];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p – середнє число робочих днів в місяці, приблизно T_p = 21 дн;

$t_{зм}$ – тривалість зміни, год.

$$C_1 = 2379{,}00 \cdot 1{,}10 \cdot 1{,}5 / (21 \cdot 8) = 23{,}37 \ грн.$$

$$З_{р1} = 23{,}37 \cdot 1{,}00 = 23{,}37 \text{ грн.}$$

Таблиця 5.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
1. Встановлення допоміжного обладнання	1,00	2	1,10	23,37	23,37
2. Інсталяція програмного забезпечення	1,50	5	1,70	36,11	54,16
3. Встановлення інфокомунікаційних блоків	4,00	4	1,50	31,86	127,45
Всього					204,98

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (5.5)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 10%.

$$З_{дод} = (25216{,}29 + 204{,}98) \cdot 10 / 100\% = 2542{,}13 \text{ грн.}$$

5.2.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (5.6)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

$$Зн = (25216,29 + 204,98 + 2542,13) \cdot 22 / 100\% = 6151,95 \text{ грн.}$$

5.2.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Методи тестування програмно-керованих інфокомунікаційних систем».

Витрати на матеріали ($М$), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (5.7)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$$М_1 = 3,00 \cdot 94,20 \cdot 1,1 - 0,000 \cdot 0,00 = 310,86 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір канцелярський офісний	94,20	3,00	0,000	0,00	310,86
Папір для заміток	45,00	2,00	0,000	0,00	99,00
Начиння канцелярське	200,00	4,00	0,000	0,00	880,00
Органайзер офісний	150,00	1,00	0,000	0,00	165,00
Картридж для принтера	350,00	1,00	0,000	0,00	385,00
Всього					1839,86

5.2.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Методи тестування програмно-керованих інфокомунікаційних систем», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \qquad (5.8)$$

де $Н_j$ – кількість комплектуючих j-го виду, шт.;

$Ц_j$ – покупна ціна комплектуючих j-го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*).

$$К_в = 1 \cdot 82{,}00 \cdot 1{,}1 = 90{,}20 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.7 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Сервер	1	82,00	90,20
Блок інформаційних ресурсів	1	84,00	92,40
Сервер обслуговування викликів	1	95,00	104,50
Всього			287,10

5.2.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \ , \qquad (5.9)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, *(K_i = 1,10…1,12);*

k – кількість найменувань устаткування.

$$B_{спец} = 520{,}00 \cdot 1 \cdot 1{,}11 = 577{,}20 \text{ грн.}$$

Отримані результати зведемо до таблиці:

Таблиця 5.8 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Мультисервісна платформа	1	520,00	577,20
Обладнання IP телефонії	1	350,00	388,50
Набір інтерфейсів	1	810,00	899,10
Всього			1864,80

5.2.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.10)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, (K_i = *1,10…1,12);*

k – кількість найменувань програмних засобів.

$$B_{прг} = 7500{,}00 \cdot 1 \cdot 1{,}2 = 9000{,}00 \text{ грн.}$$

Отримані результати зведемо до таблиці:

Таблиця 5.9 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Windows 11	1	7500,00	9000,00
Microsoft Office W	1	8400,00	10080,00
Програмний пакет обробки даних	1	22500,00	27000,00
Імітатори термінальних пристроїв	3	1280,00	4608,00
Всього			50688,00

5.2.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (5.11)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$$A_{обл} = (23600{,}00 \cdot 1) / (3 \cdot 12) = 655{,}56 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.10 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Програмно-аналітичний модуль	23600,00	3	1	655,56
Мультимедійні мікропроцесорні пристрої	25340,00	5	1	422,33
Програмний комплекс розробника програмного забезпечення	10840,00	3	1	301,11
Місце оператора спеціалізоване	9650,00	5	1	160,83
Офісна оргтехніка	11780,00	4	1	245,42
Дослідницька лабораторія	186000,00	25	1	620,00
Програмне забезпечення	9200,00	2	1	383,33
Всього				2788,58

5.2.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \qquad (5.12)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e = 4{,}10$ грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$$B_e = 0,30 \cdot 120,0 \cdot 4,10 \cdot 0,95 / 0,97 = 147,60 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.11 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Серверне обладнання 1	0,30	120,0	147,60
Серверне обладнання 2	0,30	120,0	147,60
Термінальне обладнання	0,25	120,0	123,00
Програмно-аналітичний модуль	0,75	160,0	492,00
Мультимедійні мікропроцесорні пристрої	0,32	120,0	157,44
Програмний комплекс розробника програмного забезпечення	0,75	160,0	492,00
Місце оператора спеціалізоване	0,50	160,0	328,00
Офісна оргтехніка	1,20	55,0	270,60
Всього			2158,24

5.2.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов’язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з’їзди, конференції, наради, пов’язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (5.13)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 22%.

$$B_{св} = (25216,29 + 204,98) \cdot 22 / 100\% = 5592,68 \text{ грн.}$$

5.2.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (5.14)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$$B_{сп} = (25216,29 + 204,98) \cdot 30 / 100\% = 7626,38 \text{ грн.}$$

5.2.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (5.15)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$$I_в = (25216,29 + 204,98) \cdot 50 / 100\% = 12710,63 \text{ грн.}$$

5.2.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.16)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$$В_{нзв} = (25216,29 + 204,98) \cdot 100 / 100\% = 25421,26 \text{ грн.}$$

Витрати на проведення науково-дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.17)$$

$B_{заг}$= 25216,29 +204,98 +2542,13 +6151,945417 +1839,86 +287,10 +1864,80 + 50688,00 + 2788,58 +2158,24 +5592,68 +7626,38 +12710,63 +25421,26 = 145092,87 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (5.18)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,75.

ЗВ = 145092,87 / 0,75 = 193457,16 грн.

5.3 Оцінювання важливості та наукової значимості науково-дослідної роботи

Оцінювання та доведення ефективності виконання науково-дослідної роботи фундаментального чи пошукового характеру є достатньо складним процесом і часто базується на експертних оцінках, тому має вірогідний характер.

Для обґрунтування доцільності виконання науково-дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем» використовується спеціальний комплексний показник, що враховує важливість, результативність роботи, можливість впровадження її результатів у виробництво, величину витрат на роботу.

Комплексний показник K_P рівня науково-дослідної роботи може бути розрахований за формулою:

$$K_P = \frac{I^n \cdot T_C \cdot R}{B \cdot t}, \qquad (5.19)$$

де I – коефіцієнт важливості роботи. Приймемо $I = 4$;

n – коефіцієнт використання результатів роботи; $n = 0$, коли результати роботи не будуть використовуватись; $n = 1$, коли результати роботи будуть використовуватись частково; $n = 2$, коли результати роботи будуть використовуватись в дослідно-конструкторських розробках; $n = 3$, коли результати можуть використовуватись навіть без проведення дослідно-конструкторських розробок. Приймемо n=2;

T_C – коефіцієнт складності роботи. Приймемо $T_C = 2$;

R – коефіцієнт результативності роботи; якщо результати роботи плануються вище відомих, то $R = 4$; якщо результати роботи відповідають відомому рівню, то $R = 3$; якщо нижче відомих результатів, то $R = 1$. Приймемо $R = 3$;

B – вартість науково-дослідної роботи, тис. грн. Приймемо B =193457,16 грн;

t – час проведення дослідження. Приймемо t = 0,08 років, (1 міс.).

Визначення показників *I, n, T_C, R, B, t* здійснюється експертним шляхом або на основі нормативів [20].

$$K_P = \frac{I^n \cdot T_C \cdot R}{B \cdot t} = 5{,}95$$

Якщо $K_P > 1$, то науково-дослідну роботу на тему «Методи тестування програмно-керованих інфокомунікаційних систем» можна вважати ефективною з високим науковим, технічним і економічним рівнем.

5.4 Висновок до розділу 4

Витрати на проведення науково-дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем» складають 193457,16 грн. Відповідно до проведеного аналізу та розрахунків рівень наукового ефекту проведеної науково-дослідної роботи на тему «Методи тестування програмно-керованих інфокомунікаційних систем» є середній, а дослідження актуальними, рівень доцільності виконання науково-дослідної роботи $K_P > 1$, що свідчить про потенційну ефективність з високим науковим, технічним і економічним рівнем.

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

У наш час актуальним є не тільки захист людини від виробництва і навколишнього природного середовища, а й захист навколишнього природного середовища від людини та виробництва. На цю систему діють у відповідних умовах фактори надзвичайних ситуацій. Система повинна в цих умовах стійко функціонувати і забезпечувати захист людини. Система “людина – машина – середовище” гарантує досягнення таких цілей:

– отримання результату життєдіяльності, необхідного людині;

– забезпечення безпеки життєдіяльності людини;

– недопущення появи вражаючих і зменшення дії небезпечних і шкідливих факторів до допустимих значень, які не сприяють втраті працездатності й погіршенню здоров’я людини;

– зменшення небезпечної дії життєдіяльності людини на навколишнє середовище і залучення необхідних захисних мір;

– забезпечення стійкості функціонування і захисту людини при дії різних факторів надзвичайних ситуацій.

У магістерській кваліфікаційній роботі досліджуються методи тестування програмно керованих інфокомунікаційних систем. Всі дослідження і експериментальні процеси відбуваються з участю ПК.

Будь-які трудові процеси потребують заходів з охорони праці, для забезпечення безпеки працівника на робочому місці і для виключення або зменшення впливу шкідливих факторів, що можуть впливати на нього в процесі трудової діяльності.

На працівників, що проводять дослідження на ПК можуть впливати такі небезпечні та шкідливі фактори, у відповідності з прийнятою класифікацією за ГОСТ 12.0003.-74 [21].:

Фізичні: підвищена та понижена температура повітря робочої зони; підвищена та понижена рухливість повітря робочої зони; недостатня освітленість робочої зони; недостатність природного освітлення; небезпечний рівень напруги в електричному

колі, замикання якого може відбутися через тіло людини; підвищена та понижена вологість повітря; підвищений рівень електромагнітного опромінення.

Психофізіологічні: нервово – психічні перевантаження (монотонність праці, емоційні перевантаження, перенапруга аналізаторів).

6.1 Технічні рішення з безпечного виконання робіт

6.1.1 Технічні рішення з організації робочого місця під час проектування

Площа, виділена для одного робочого місця з відео терміналом або персональним комп'ютером, повинна складати не менше 6 кв. м, а обсяг – не менше 20 куб. м. При розміщенні робочих місць з відео терміналами та ПК необхідно дотримуватись таких вимог:

–робочі місця з ПК розміщуються на відстані не менше 1 м від стін зі світловими прорізами;

– відстань між бічними поверхнями відео терміналів має бути не меншою за 1,2 м; – відстань між тильною поверхнею одного ПК та екраном іншого не повинна бути меншою 2,5 м; – прохід між рядами робочих місць має бути не меншим 1 м. Вимоги цього пункту щодо відстані між бічними поверхнями ПК та відстані між тильною поверхнею одного ПК та екраном іншого враховуються також при розміщенні робочих місць з ПК в суміжних приміщеннях, з урахуванням конструктивних особливостей стін та перегородок. Усі працівники, які виконують роботи, пов'язані з експлуатацією, обслуговуванням, налагодженням та ремонтом ПК, підлягають обов'язковому медичному огляду – попередньому підчас оформлення на роботу та періодичному протягом трудової діяльності. Посадові особи та спеціалісти, інші працівники підприємств, які організовують та виконують роботи, пов'язані з експлуатацією, профілактичним обслуговуванням, налагодженням та ремонтом ПК, проходять підготовку (підвищення кваліфікації), перевірку знань з охорони праці, а також інструктажі.

6.1.2 Електробезпека виробничого приміщення

Класифікація приміщень за категоріями електробезпеки залежно від мікроклімату виробничих будівель. Допустимими визнаються умови праці у будівлях, де відносна вологість повітря не перевищує 60%, температура повітря не перевищує 35 °С, а пил та хімічно агресивне середовище – відсутні. За таких умов праці мікроклімат вважається сухим. Вологими називаються умови роботи, де відносна вологість повітря становить від 60% до 75%. Сирі – це такі умови, які характеризуються відносною вологістю повітря в будівлі більшою за 75%. Особливо сирі умови – це умови, із майже стовідсотковою відносною вологістю повітря. Гарячими умовами праці є становище, при якому температура повітря перевищує 35 °С. Запиленими визнаються умови, при яких виділяється велика кількість виробничого пилу, внаслідок чого він може залишатись на зовнішніх поверхнях або навіть проникати у середину обладнання чи апаратів. До умов праці з хімічно активним середовищем відносять умови, при яких у повітрі протягом тривалого часу залишаються гази або краплі рідин, які негативно впливають на ізолюючі властивості і струмопровідні елементи електричних інструментів.

Класифікація приміщень за рівнем електробезпеки Відповідно до ПУЕ, усі промислові приміщення (цехи, майстерні, склади) за ступенем ймовірності ураження електричним струмом можна розділити на три категорії: Будівлі з підвищеною небезпекою До цього типу будівель належать споруди, в яких наявний принаймні один із таких факторів: сирі приміщення, відносна вологість в яких протягом тривалого часу перевищує 75%; приміщення, в яких пил покриває провідники, забивається всередину машин та обладнання; приміщення зі струмопровідними підлогами (металевими, земляними, цегляними, залізобетонними); приміщення, в яких середня температура повітря зазвичай перевищує +30 °С; приміщення, в яких існує ймовірність одночасного торкання співробітника до зовнішніх елементів електричного устаткування і заземлених металевих конструкцій будівель, технологічного обладнання тощо.

Будівлі з особливо небезпечними умовами Ці будівлі характеризуються як дуже сирі приміщення зі стовідсотковою відносною вологістю. Стеля, стіни, підлога, устаткування у таких приміщеннях постійно вкриті тонким шаром крапель чи пліснявою. Слід бути вкрай обережними, оскільки при виконанні робіт з використанням електричної напруги на відкритому повітрі, всередині посудин, всередині непросохлих приміщень ймовірність ураження співробітників чи сторонніх осіб електричним струмом дуже висока. До будівель з особливо небезпечними умовами належать також споруди з хімічно активним середовищем, яке завдяки своїм властивостям завдає шкоду ізоляції та електричним матеріалам. Окрім того, будівлі, які мають одночасно дві або більше ознаки приміщень з підвищеною небезпекою так само належать до будівель із особливо небезпечними умовами.

Будівлі без факторів збільшеної небезпеки До цієї категорії належать будівлі, в яких немає жодної з ознак, властивих приміщенням з підвищеною або особливою небезпекою[22].

Для забезпечення безпеки від ураження електричним струмом працівника необхідно слідкувати, щоб приміщення було з нормальними параметрами вологості та температури повітря.

6.2 Технічні рішення з гігієни праці та виробничої санітарії

6.2.1 Мікроклімат

Стан навколишнього середовища характеризується такими метеорологічними умовами: температурою, вологістю, тиском і швидкістю руху повітря. Розглянемо почергово кожну з цих умов.Температура.Зміна температури (тепла і холоду) металевих конструкційпризводить до зміни їх стану: прискорення хімічних реакцій, деформації елементів конструкцій, зміни параметрів конструкцій, розкладання деяких органічних ізоляційних матеріалів. При відносно високих перепадах температур більшість органічних ізоляційних

матеріалів розтріскуються, відходять від основиі відриваються від стінки корпусу. Багато матеріалів, гнучких і еластичних в нормальних умовах, при низьких температурах стають крих кими і ламаються. Величина лінійної зміни розмірів радіодеталей характеризуються коефіцієнтом лінійного розширення матеріалу, який знаходиться в межах $0{,}9*10^{-6}$– $99*10^{-6}$ 1/град. Різниця в лінійних розширеннях матеріалів або деталей є причиною порушення ціліс ності конструкцій, зварних і паяних швів, порушення герметичності. Тонкі монтажні проводи з великим попереднім натягом при пониженні температури обриваються. Підвищена температура шкідливовпливає на параметри всіх елементів конструкцій, а циклічні її зміни – для багатьох вузлів є надто небезпечні.Тепловий режим конструкцій характеризується сумою температур всіх елементів, тобто тепловим полем. Для забезпечення теплового режиму радіоелектронних систем (РЕС) використовуються системи забезпечення нормального теплового режиму (СЗНТР), кожна з яких характеризується особливостями структури, інтенсивністю тепловідводу, технічними показниками.

За ступенем впливу на тепловий стан людини мікрокліматичної умови поділяють на оптимальні та допустимі. Оптимальні мікрокліматичні умови − поєднання параметрів мікроклімату, які при тривалому та систематичному впливі на людину забезпечують зберігання нормального теплового стану організму без активізації механізмів терморегуляції. Вони забезпечують відчуття теплового комфорту та створюють передумови для високого рівня працездатності [23]. Допустимі мікрокліматичні умови − поєднання параметрів мікроклімату,які при тривалому та систематичному впливі на людину можуть викликати зміни теплового стану організму, що швидко минають і нормалізуються та супроводжуються напруженням механізмів терморегуляції в межах фізіологічної адаптації. При цьому не виникає ушкоджень або порушень стану здоров'я, але можуть спостерігатися дискомфортні тепловідчуття, погіршення самопочуття та зниження працездатності[23].

Категорія робіт− розмежування робіт за важкістю на основі загальних енерговитрат організму[23].Легкі фізичні роботи (категорія I) охоплюють види діяльності, при яких витрата енергії дорівнює 105−140 Вт (90−120 ккал/год.) − категорія Іа та 141−175 Вт (121−150 ккал/год.) − категорія Іб. До категорії Іа належать роботи, що виконуються сидячи і не потребують фізичного напруження. До категорії Іб належать роботи, що виконуються сидячи, стоячи або пов'язані з ходінням та супроводжуються деяким фізичним напруженням.

Визначаємо наявну категорію робіт, як Іа.

В кабінах, на пультах та місцях керування технологічними процесами, в залах ЕОМ при виконанні робіт операторського типу повинні забезпечуватися такі оптимальні величини температури, відносної вологості та швидкості руху повітря, що зазначені в нормативному акті НПАОП 0.00-7.15-18 Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями. [24]

Оптимальні параметри мікроклімату наведені в табл. 6.1.

Таблиця 6.1 – Оптимальні параметри мікроклімату при виконанні робіт операторського типу на ПК для робіт категорії Іа

Період року	Температура повітря, ºС	Відносна вологість, %	Швидкість руху, м\с
Холодний	22-24	60-40	0,1
Теплий	23-25	60-40	0,1

Для створення сприятливих умов праці небхідно дотримуватись наведених параметрів мікроклімату робочого приміщення.

6.2.2 Склад повітря робочої зони

У сучасній техніці застосовується безліч речовин, які можуть потрапляти в повітря і становити небезпеку здоров'ю людей. Для визначення небезпечності медики досліджують вплив цих речовин на організм людини і встановлюють

безпечні для людини концентрації та дози, які можуть потрапити різними шляхами в організм людини. На промислових підприємствах повітря робочої зони може забруднюватися шкідливими речовинами, які утворюються в результаті технологічного процесу або містяться в сировині, продуктах та напівпродуктах і відходах виробництва. За токсичною дією шкідливі речовини поділяють на: кров'яні отрути, які взаємодіють з гемоглобіном крові і гальмують його здатність до приєднання кисню (оксид вуглецю, бензол, сполуки ароматичного ряду та ін.); нервові отрути, які викликають збудженість нервової системи, її виснаження, руйнування нервових тканин (наркотики, спирти, сірчаний водень, кофеїн та ін.); подразнюючі отрути, що вражають верхні дихальні шляхи і легені (аміак, сірчаний газ, паракислот, окисиазоту, ароматичні вуглеводні та ін.); ті, що пропалюють та подразнюють шкіру і слизові оболонки (сірчана та соляна кислоти, луги); печінкові отрути, дія яких супроводжується зміною та запаленням тканин печінки (спирти, дихлоретан, чотири хлористий вуглець); алергени, що змінюють реактивну спроможність організму (алкалоїди та інші речовини); канцерогени, що спричиняють утворення злоякісних пухлин (3,4-бензопірен, кам'яновугільна смола); мутагени, що впливають на генетичний апарат клітини (окисетилену, сполуки ртуті таін.). Гігієнічне нормування шкідливих речовин Залежно від ступеня токсичності, фізико-хімІчних властивостей, шляхів проникнення в організм, санітарні норми встановлюють гранично допустимі концентрації (ГДК) шкідливих речовин у повітрі робочої зони виробничих приміщень, перевищення яких не припустиме. Гранично допустимою концентрацією (ГДК) шкідливої речовини в повітрі робочої зони вважається така концентрація, вплив якої на людину вразі її щоденної регламентованої тривалості не призводить до зниження працездатності чи захворювання в період трудової діяльності та у наступний період життя, а також не справляє негативного впливу на здоров'я нащадків.

Існує багато різних способів та заходів, призначених для підтримання чистоти повітря виробничих приміщень у відповідності до вимог санітарних норм. Всі вони зводяться до конкретних заходів:

1. Запобігання проникненню шкідливих речовин у повітря робочої зони за рахунок герметизації обладнання, ущільнення з'єднань, люків та отворів, удосконалення технологічного процесу.

2. Видалення шкідливих речовин, що потрапляють у повітря робочої зони, за рахунок вентиляції, аспірації або очищення і нормалізації повітря за допомогою кондиціонерів.

3. Застосування засобів захисту людини.

6.2.3 Виробниче освітлення

Відповідно до ДБН В.2.5-28:2018 [26] Система природного освітлення відноситься до бокової. Характеристика зорових робіт – середньої точності.

Норми освітленості при штучному освітленні та КПО (для III пояса світлового клімату) при природному та сумісному освітленні зазначені у таблиці 6.2:

Таблиця 6.2 - Норми освітленості в приміщенні

Характеристика зорової роботи	Найменший розмір об'єкта розрізнювання	Розряд зорової роботи	Підрозряд зорової роботи	Контраст об'єкта розрізнення з фоном	Характеристика фона	Освітленість, лк		КПО,%			
						Штучне освітлення		Природнє освітлення		Сумісне освітлення	
						Комбіноване	Загальне	Верхнє або верхнє	Бокове	Верхнє або верхнє і бокове	Бокове
Середньої точності	Від 0,5 до 1,0	IV	б	середній	середній	200	500	4	1,5	2,4	0,9

При експлуатації штучного освітлення здійснюється контроль за рівнем напруги освітлювальної мережі, своєчасна заміна перегорілих ламп, забезпечується чистота повітря у приміщенні.

Аналіз дії світла на організм людини й основних якостей зорового сприймання дозволяє сформувати основні вимоги до виробничих приміщень, які полягають в забезпеченні достатньої освітленості робочих поверхонь, рівномірності розподілення яскравості, відсутності глибоких і різких тіней, постійної освітленості в часі.

6.2.4 Виробничий шум

Основними характеристиками звукових коливань є інтенсивність (сила), частота і форма звукової хвилі. Інтенсивність визначається енергією, що переноситься за 1 с звуковою хвилею через поверхню площею 1 м^2, яка перпендикулярна напрямку розповсюдження звукової хвилі. Одиниця вимірювання – Вт/м^2. Інтенсивність звуку можна визначити через звуковий тиск, який являє собою різницю між миттєвим значенням тиску в даній точці середовища при проходженні через неї звукових хвиль і середнім значенням тиску, яке існує в тій же точці при відсутності звуку (Па). Діапазон тисків, що сприймає вухо людини, дуже широкий (10-12Вт/м^2 – поріг больового відчуття, верхня межа), тому інтенсивність звуку виражають у логарифмічних характеристиках, використовуючи параметр, який називають рівнем звукового тиску**.**Нормативним документом, який регламентує рівні шуму для різних категорій робочих місць службових приміщень, є ДСН 3.3.6.037-99 [27].

Гранично-допустимі рівні шумів санітарними нормами встановлені для кожного класу: •для високочастотних шумів (вище 800 Гц) – 75-85 дБ;

•для середньо частотних шумів (300-800 Гц) – 85-90 дБ

•для низькочастотних шумів (до 300 Гц) – 90-100 дБ.

Одним з основних технічних заходів є зменшення при експлуатації та на стадії проектування, конструювання обладнання причин шуму і вібрації в самому джерелі утворення. Досягають цього завдяки використаннюраціональної конструкції обладнання, заміни ударної дії деталейі машин коливальною, з’єднання елементів гнучкими зв’язками, врівноважування обертових частин

механізмів, заміни металевих деталей пластмасовими, забезпеченнярізнихвласних частот коливань механізму з частотою збуджуючої сили.

6.2.5 Електромагнітні випромінювання

Під час проведення роботи із використанням ПК на розробника діє підвищений рівень електромагнітного поля.

Електромагнітні випромінювання радіочастотного діапазону, що генеруються ВДТ пов'язані перш за все з частотою формування елемента зображення, а також з інтенсивністю електронного променя, що зумовлює яскравість точок на екрані.

Допустимі рівні випромінювань моніторів ПК (за вимогами MPR II 1990:10 Шведського національного комітету з вимірювань та досліджень та нормами TCO 9295) наведені в таблиці 6.4.

Таблиця 6.4 - Допустимі рівні випромінювань моніторів ПК

Вид поля	TCO	MPR II
Змінне електричне поле		
5 Гц – 2 кГц	10 В/м	2,5 В/м
2 кГц – 400 кГц	1 В/м на відстані 0,3 м від центра екрана і 0,5 м навколо монітора	2,5 В/м на відстані 0,5 м навколо монітора
Змінне магнітне поле		
5 Гц – 2 кГц	250 нТл 200 мА/м	250 нТл 200 мА/м
2 кГц – 400 кГц	25 нТл 20 мА/м на відстані 0,3 м від центра екрана і 0,5 м навколо монітора	25 нТл 20 мА/м на відстані 0,3 м від центра екрана і 0,5 м навколо монітора

За кордоном застосовують наступний поділ на діапазони НВЧ: L(0,39-1,55 ГГц); S(1,55-5,20 ГГц); R,LS(1,7-2,6 ГГц); H(3,95-5,85 ГГц); C(5,85-8,20 ГГц); X(5,2-11 ГГц); XN(5,40-8,20 ГГц); W, XB(7,02-10,1 ГГц); Ku,Y(12,4-18 ГГц);

Для зменшення впливу електромагнітного випромінювання на працівника слід дотримуватися раціонального режиму роботи та відпочинку.

6.2.6 Психофізіологічні фактори

Психофізіологічні фактори небезпеки – чинники, обумовлені особливостями фізіології та психології людини, що можуть завдати їй шкоди за певних обставин.

Психофізіологічні фактори вибираються відповідно з Гігієнічною класифікацією праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу [28].

Психофізіологічні фактории при роботі з ПК:

- перенапруження зорових аналізаторів;
- монотонність трудового процесу;
- розумове перенапруження;
- нервово-емоційні перевантаження.

Класи умов праці за показниками напруженості праці:

Інтелектуальні навантаження:

Зміст роботи – рішення складних завдань з вибором за алгоритмом;

Сприймання інформації та їх оцінка – сприймання інформації з наступною корекцією дій та операцій;

Розподіл функцій за ступенем складності завдання - обробка, контроль, перевірка завдання.

Сенсорні навантаження:

Зосередження (%за зміну) – до 50%;

Щільність сигналів (звукові за1 год) – до 150;

Навантаження на слуховий аналізатор (%) – розбірливість слів та сигналів від 50 до 80 %;

Навантаження на голосовий апарат (протягом тижня) – від 20 до 25%.

Емоційне навантаження:

Ступінь відповідальності за результат своєї діяльності – є відповідальним за функціональну якість основної роботи; Ступінь ризику для власного життя – вірогідний;

Ступінь відповідальності за безпеку інших осіб – є відповідальним за безпеку інших.

Режим праці: Тривалість робочого дня – більше 8 год; Змінність роботи – однозмінна.

6.3 Безпека у надзвичайних ситуаціях. Дослідження безпеки роботи програмно-керованих інфокомунікаційних систем в умовах дії загрозливих чинників надзвичайних ситуацій

Система передачі даних спрямована на забезпечення послугами цифрового зв'язку для різних задач. У зв'язку з тим, що це має важливе значення для воєнної сфери, то на них можуть справляти значний вплив загрозливі чинники надзвичайні ситуації різного типу і необхідно провести дослідження безпеки роботи програмно-керованих інфокомунікаційних систем. До таких НС можна віднести: стихійні лиха (землетруси, блискавка, зливи), а особливо впливовими на РЕА мають іонізуючі випромінювання та ЕМІ. Тому при забезпеченні даних пристроїв слід забезпечити найвищий рівень захисту від тої чи іншої НС, оскільки кожна НС має свій вплив на дану систему.

Тож, в даній частині розділу необхідно виконати дослідження оцінки безпеки роботи та розробку заходів по підвищенню стійкості роботи програмно-керованих інфокомунікаційних систем безпілотного літального апарата в умовах дії іонізуючих випромінювань та електромагнітного імпульсу.

В РЕА застосовуються елементи, до складу яких входять такі матеріали: метали, неорганічні матеріали (в основному діелектрики), провідники і різноманітні органічні сполуки (діелектрики, смоли і т.д.). Серед цих матеріалів метали найбільш чутливі до впливу іонізуючих випромінювань, оскільки їм властива висока концентрація вільних носіїв.

В радіоелектронній апаратурі іонізуючі випромінювання, викликають зворотні і незворотні процеси, внаслідок яких можуть відбуватися порушення роботи електричних елементів схеми, що призводять до виходу з ладу апаратури. Так, проходячи через елементи РЕА, потік гамма-випромінень створює в них вільні носії електричних зарядів, в результаті переміщення яких виникає помилковий імпульс, який призводить до спрацьовування пристрою. При великих дозах випромінювання втрачають працездатність комплектуючі елементи систем радіоелектроніки і електроавтоматики. В результаті опромінення у транзисторах змінюється обернений струм і коефіцієнт підсилення, у конденсаторах знижуються напруги пробою та опір стікання, змінюється провідність і внутрішній нагрів; руйнується електрична ізоляція дротів з полімерних матеріалів. Неорганічні матеріали менш чутливі до впливу іонізуючих випромінювань [18].

Для інженерної практики найбільший інтерес представляє перший випадок, тобто оцінка безпеки роботи РЕА при перебуванні її в зараженій радіоактивними речовинами місцевості протягом певного часу після випадання радіоактивних речовин у даній місцевості.

ЕМІ ушкоджує напівпровідникові прилади, резистори, конденсатори. Це являє велику небезпеку для апаратури, добре захищеної від впливу інших загрозливих чинників. Тому слід пам’ятати про те, що захист апаратури від механічних ушкоджень не захищає від впливу ЕМІ. Апаратура може втратити працездатність, знаходячись у надійних захисних спорудженнях [22].

6.3.1 Дослідження безпеки роботи програмно-керованих інфокомунікаційних систем в умовах дії іонізуючих випромінювань.

За критерій безпеки роботи системи в цих умовах приймається таке граничне значення рівня ($Р_{зв}$, Р/год), при якому можуть виникнути тимчасові зміни, але пристрій буде працювати з потрібною якістю.

Приймаючи до уваги елементну базу, що використовується для реалізації програмно-керованих інфокомунікаційних систем, складається таблиця потужностей експозиційної дози опромінення для кожного елемента $P_{зв.і}$, що викликають початок зворотних змін. Отримані значення занесемо до таблиці 6.5.

Таблиця 6.5 – Потужність граничної експозиційної дози для програмно-керованих інфокомунікаційних систем

№	Елементи програмно-керованих інфокомунікаційних систем	$P_{зв.і}$, Р/с	$P_{зв.}$, Р/с
1	Процесори, інтегральні мікросхеми	10^3	10^3
2	Діоди загального призначення	10^4	
3	Транзистори загального призначення	10^4	
4	Мікросхеми	10^5	
5	Конденсатори	10^7	
6	Резистори	10^8	

Визначається елемент, який найбільшою мірою піддається впливу випромінюванням, тобто елемент із мінімальним значенням $P_{зв}$.

$$P_{зв} = 10^3 \text{ Р/с}$$

В якості критерію стійкості роботи РЕА використовується граничне значення рівня іонізуючих випромінювань:

$$P_{гр} = K_{над*} P_{зв*} K_{посл}, \quad (6.1)$$

де $P_{зв}$ - рівень радіації незворотних змін пристрою в цілому;

$K_{над}$ - коефіцієнт надійності ($K_{над}$ =0,9÷0,95);

$K_{посл}$ - коефіцієнт послаблення.

$$P_{гр} = 0,95* 10^3 *2 = 7,6 \text{ Р/с}.$$

З наведеної таблиці слідує, що мінімальні значення граничних рівнів радіації елементів, при яких в елементній базі можливі необоротні зміни мають

інтегральні мікросхеми великої ступені інтеграції та мікропроцесори – $P_{зв}=10^3$, $k_{посл}=2$.

Визначаємо допустимий час роботи пристрою:

$$t_{доп} = \left(\frac{Д_{гр} \cdot K_{осл} + 2 \cdot P_1 \cdot \sqrt{1}}{2 \cdot P_1} \right)^2, \quad (6.2)$$

$$t_{доп} = \left(\frac{10^3 \cdot 2 + 2 \cdot 7{,}6 \cdot \sqrt{1}}{2 \cdot 7{,}6} \right)^2 = 17313(год).$$

Таким чином, допустимий час роботи програмно-керованих інфокомунікаційних систем складатиме 17313 годин при максимальному рівні радіації 7,6 Р/с.

6.3.2 Дослідження безпеки роботи програмно-керованих інфокомунікаційних систем в умовах дії електромагнітного імпульсу

За критерій безпеки роботи програмно-керованих інфокомунікаційних систем в умовах дії електромагнітного імпульсу можна прийняти коефіцієнт безпеки:

$$K_б = 20 \lg \frac{U_д}{U_г} \geq 40 \text{ (дБ)},$$

де $U_д$ – допустиме коливання напруги живлення (для мікросхем 5 В);

$U_г$ – напруга наведена за рахунок електромагнітного імпульсу у вертикальних (горизонтальних) струмопровідних частинах, В.

Допустимі коливання напруги живлення:

$$U_д = U_ж + \frac{Uж}{100} * N = 5 + \frac{5}{100} = 5{,}25(B)$$

В зв’язку з тим, що окремі елементи приладу можуть мати різні значення коефіцієнтів безпеки, то стійкість роботи системи в цілому визначається мінімальним значенням коефіцієнта безпеки.

З рівняння (6.1) визначаємо:

$$U_г = \frac{U_д}{10^{\frac{40}{20}}} = \frac{5,25}{100} = 0,05(B)$$

Приймемо максимальну довжину горизонтальних струмопровідних частин $l_г$=0,58 м. Тоді горизонтальна складова напруженості електричного поля визначається за формулою:

$$E_г = U_в / l_в = 0,05/0,58 = 0,092 \text{ (В/м)}$$

Звідси вертикальна складова напруженості буде $E_в$=92 В/м.

Таким чином, робота програмно-керованих інфокомунікаційних систем на основі Ethernet технології можлива у випадку, якщо не перевищується значення вертикальної складової напруженості електричного поля 92 В/м.

6.3.3 Розробка заходів по підвищенню безпеки роботи програмно-керованих інфокомунікаційних систем в умовах надзвичайних ситуацій

З метою зменшення негативного впливу на систему передачі даних можна використати наступні методи.

Для захисту розробки, як і любих радіоелектронних пристроїв від дії іонізуючих випромінювань можна використати алюмінієві сплави, леговані елементами з високим атомним номером (лантаноїдами і рідкоземельними елементами), сплави на основі тугоплавких і рідкоземельних елементів і багатошарові матеріали. Також для боротьби з впливом іонізуючого випромінювання можна використати новітній вітчизняний метод, що полягає в

захисному покритті радіоелектронної апаратури, що розміщується на поверхнях даних елементів, які піддаються впливу іонізуючого випромінювання, відмінним тим, що захисне покриття виконане у вигляді наноструктури, яка включає сукупність атомів рідкоземельних елементів, введених в структуру армованої атомно-молекулярної металічної матриці, або утворює її захисний шар.

Найкращим для захисту від електромагнітного імпульсу є захищене металічним екраном приміщення, в якому розміщена радіоелектронна апаратура. Оскільки такий захист в ряді випадків неможливо виконати, то використовуються менш надійні засоби захисту, такі як струмопровідні сітки та плівкові покриття вікон, стільникові металеві конструкції для повітрозбірників та вентиляційних отворів і контактні пружинні прокладки, що розміщуються по периметру дверей і люків. Також для захисту кабельних вводів використовують в їх конструкції фільтри та встановлення вбудованих зенерівських діодів.

6.4 Висновки до розділу 6

В ході виконання було розглянуто вплив іонізуючого випромінювання та ЕМІ на компоненти схеми, виконано розрахунки з яких видно, що ні один з класів елементів схеми не зазнає більшого впливу за граничне значення, також розраховано термін безпечної роботи системи, який складає 17313год. Що стосується впливу електромагнітного імпульсу, то з урахуванням необхідного рівня коефіцієнта безпеки було розраховано значення напруженості електричного поля. Для підвищення безпеки роботи програмно-керованих інфокомунікаційних систем наведено основні заходи боротьби з впливом загрозливих чинників НС.

Отже основною метою даної частини розділу було дослідження безпеки роботи програмно-керованих інфокомунікаційних систем та розробка дієвих заходів по підвищенню безпеки роботи цієї системи в умовах надзвичайних ситуацій.

ВИСНОВКИ

В результаті проведених досліджень отримані нові наукові і практичні результати, спрямовані на підвищення стійкості до відмов мікропроцесорних пристроїв за рахунок виявлення помилок ПЗ, що проявляються в реальній апаратної середовищі.

1. В результаті аналізу робіт, присвячених питанням стійкості програмно-технічних комплексів до відмов, виявлено, що відомі методики виявлення несправності на основі регламентованих методик випробувань пристроїв недостатньо ефективні і вимагають удосконалення. У зв'язку з тим, що використовуються мікропроцесорні елементи апаратних засобів відносяться до класу високонадійних виробів, то основною проблемою забезпечення працездатності проектованих пристроїв є виявлення відмов, пов'язаних з дефектами програмного забезпечення, що їх виявляють в реальному апаратної середовищі, для чого потрібно імітація несправності.

2. Досліджено методи автоматизації спотворення даних і аналізу реакції на внесення помилки в проектовану систему. Розглянуто відмінності між результатами тестування на основі внесення несправності в функціональні компоненти пристрою і на основі імітації наслідків виникнення несправності. Показано, що використання другого методу має переваги для мікропроцесорних систем в зв'язку з відсутністю руйнують дій і зменшенням витрат часу на його здійснення.

3. Використання модифікованого алгоритму фаззінга, що включає предобработку вихідних даних з визначенням режимів випробування пристрою, що перевіряються функцій і точок контролю, дозволяє швидше виявити помилки за допомогою фаззінга для графа розгалужень програми мікропроцесорного пристрою і аналізу реакції пристрою за допомогою інтерфейсу JTAG на проімітувати несправності програми (з допомогою вивчення генерується матриці розгалужень).

4. Отримані значення показників для позитивного рішення про завершення випробувань на основі критерію, значення якого за допомогою методики на основі нечіткого логічного висновку, що характеризує необхідний обсяг тестових випробувань мікропроцесорного пристрою, відповідає стану працездатності мікропроцесорного пристрою.

5. Виявлено, що незалежне застосування імітатора дозволяє підвищити кількості знайдених помилок, здатних призводити до відмов (від 5% до 19% виявлених в порівнянні з класичним тестуванням), а спільне застосування імітатора з спрямованим фаззінга дозволяє забезпечити працездатність пристрою.

6. Розроблений програмно-апаратний комплекс, заснований на спільному застосуванні пропонованої методики аналізу проблемних ситуацій, модифікованого алгоритму фаззінга, а також імітатора несправності, дозволяє практично здійснити автоматизацію проведення випробувань мікропроцесорних пристроїв, і скоротити часові витрати (32 години замість 120), а також збільшити кількість виявлених дефектів ПЗ в порівнянні з класичним тестуванням на 82% від загального числа помилок (з 31 до 179 помилок).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Денисова, Л. А. Многокритериальная оптимизация на основе генетических алгоритмов при синтезе систем управления: монография / Л. А. Денисова; Ом. гос. техн. ун-т. – Омск: Изд-во ОмГТУ, 2014. – 170 с.

2. Денисова, Л. А. Модели и методы проектирования систем управления объектами с переменными параметрами: монография / Л. А. Денисова; Ом. гос. техн. ун-т. – Омск: Изд-во ОмГТУ, 2014. – 167 с.

3. Ермаков, А. Д. К синтезу адаптивных проверяющих последовательностей для недетерминированных автоматов / А. Д. Ермаков, Н. В. Евтушенко // Труды Института системного программирования РАН. – 2016. – Т. 28, вып. 3. – С. 123–144.

4. Ермаков, М. К. Проведение динамического анализа исполняемого кода формата ARM ELF на основе статического бинарного инструментирования / М. К. Ермаков // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. – 2016. – № 1 (236). – С. 108–117.

5. Легков, К. Е. Новые принципы построения автоматизированных систем управления современными инфокоммуникационными сетями специального назначения / К. Е. Легков // Наукоемкие технологии в космических исследованиях Земли. – 2015. – № 1. – С. 38–41.

6. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. – Санкт-Петербург: Питер, 2018. – 351 с.

7. Панков Д. А. Контроль и диагностика неисправностей программноаппаратного комплекса / Д. А. Панков, Л. А. Денисова // Омский научный вестник. – 2018. – № 2 (158). – С. 128–133.

8. Панков Д. А. Проектирование программно-аппаратного комплекса: определение объема тестовых испытаний микропроцессорных устройств / Д. А. Панков, Л. А. Денисова // Автоматизация в промышленности. – 2020. – № 12. – С. 23–29.

9. Панков Д. А. Разработка и исследование алгоритма маршрутизации в многофункциональном комплексе связи. / Д. А. Панков, Л. А. Денисова // Омский научный вестник. – 2017. – № 6 (156). – С. 143–145.

10. Пинкевич, В. Ю. Тестирование и отладка встраиваемых вычислительных систем на основе уровневых моделей / В. Ю. Пинкевич, А. Е. Платунов // Научнотехнический вестник информационных технологий, механики и оптики. – 2018. – Т. 18, № 5. – С. 801–808.

11. Похабов, Ю. В. Надежность в цифровых технологиях / Ю. В. Похабов // Надежность. – 2020. – Т. 20, № 2. – С. 3–11.

12. Слинкин, Д. И. Анализ современных методов тестирования и верификации проектов сверхбольших интегральных схем / Д. И. Слинкин // Программные продукты и системы. – 2017. – Т. 30, № 3. – С. 401 – 406.

13. Полесский, С. Разработка методики выявления факторов, наиболее сильно влияющих на надежность электронных устройств / С. Полесский, О. Маякова, А. Алейников // Системы управления, связи и безопасности. – 2016. – № 4. – С. 114 – 127.

14. Томилов, И. О. Фаззинг. Поиск уязвимостей в программном обеспечении без наличия исходного кода / И. О. Томилов, А. В. Трифанов // Интерэкспо ГеоСибирь. – 2017. – Т. 9, № 2. – С. 75–80.

15. Чекмарев, С. А. Технология инъектирования сбоев для тестирования сбоеустойчивости микропроцессоров, предназначенных к использованию в бортовой аппаратуре космических аппаратов / С. А. Чекмарев, В. Х. Ханов, А. С. Тимохович // Вестник СибГАУ. – 2016. – Том 17, № 3. – С. 768–781.

16. Han, H. IMF: Inferred Model-based Fuzzer / H. Han, S. Cha. – DOI:10.1145/3133956.3134103 // Computer and Communications Security: proceedings of the 2017 ACM SIGSAC conference (Dallas, Texas USA, October2017). – New York: Association for Computing Machinery, 2017. – P. 2345 – 2358.

17. Pankov D. A. Automated testing and fault diagnosis of the microcontroller system / D. A. Pankov, L. A. Denisova // IOP Conference Series: Materials Science

and Engineering. International Workshop "Advanced Technologies in Material Science, Mechanical and Automation Engineering – MIP: Engineering – 2019". Krasnoyarsk Science and Technology City Hall of the Russian Union of Scientific and Engineering Associations. – 2019. – P. 22072.

18. Pankov, D. A. Model studies of systems with diagnostics based on fault simulation / D. A. Pankov, L. A. Denisova // IOP Conference Series: Materials Science and Engineering. Krasnoyarsk Science and Technology City Hall of the Russian Union of Scientific and Engineering Associations. – 2020. – P. 12021.

19. Нікітович Д.В., Крещенко М.С., Щербань О.А. Програмно-апаратне тестування інфокомунікаційних систем // Сучасні проблеми інфокомунікацій, радіоелектроніки та наносистем (СПІРН-2021): матеріали VIII міжнародної науково-практичної конференції, м. Вінниця, 03-05 листопада 2021 р. – Вінниця, ВНТУ, 2021. – 1-4 с.

20. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

21. ГОСТ 12.0.003-74 ССБТ. Опасные и вредные производственные факторы. Классификация.

22. Правила улаштування електроустановок - [Електронний ресурс] - Режим доступу: http://www.energiy.com.ua/PUE.html

23. Санітарні норми мікроклімату виробничих приміщень. ДСН 3.3.6.042–99 [Електронний ресурс]. –Режим доступу: http//www.dnaop.com.

24. НПАОП 0.00-7.15-18 Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями. - [Електронний ресурс] - Режим доступу: http://sop.zp.ua/norm_npaop_0_00-7_15-18_01_ua.php

25. СНиП 2.04.05-91*У. Отопление, вентиляция и кондиционирование[Електронний ресурс] - Режим доступу: https://dnaop.com/html/1671/doc-%D0%A1%D0%9D%D0%B8%D0%9F_2.04.05-91_%D0%A3

26. ДБН В.2.5-28:2018 Природне і штучне освітлення - [Електронний ресурс] - Режим доступу: http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=79885

27. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та інфразвуку. - [Електронний ресурс] - Режим доступу: http://document.ua/sanitarni-normi-virobnichogo-shumu-ultrazvuku-ta-infrazvuku-nor4878.html

28. Наказ від 08.04.2014 № 248 Про затвердження Державних санітарних норм та правил Гігієнічна класифікація праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу - [Електронний ресурс] - Режим доступу: http://online.budstandart.com/ua/catalog/topiccatalogua/labor-protection/14._nakazy_ta_rozpor_183575/248+58074-detail.html

ДОДАТКИ

Додаток А

(Технічне завдання)

Додаток Б

(обов’язковий)

Загальна структура імітатора несправності

Додаток В

(обов’язковий)

Архітектура системи автоматизованого тестування

Додаток Г

(обов’язковий)

Структурна схема пристрою імітації несправності та аналізу реакції

Додаток Д

(обов’язковий)

Алгоритм адаптивної передачі інформації

Додаток Е

(обов’язковий)

Алгоритм тестування пристрою за допомогою JTAG

Додаток Є

(обов’язковий)

Функціональна схема комплексу проведення випробувань

Додаток Ж

(обов’язковий)

Структурна схема імітаційних досліджень

Додаток З

(обов’язковий)

Структура програмно-апаратних компонентів налагодження

Додаток А

(обов’язковий)

ВНТУ

ЗАТВЕРДЖУЮ

Зав.кафедри ТКСТБ ВНТУ,

докт. техн. наук, професор

В.М. Кичак

. “__” _____ 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

МЕТОДИ ТЕСТУВАННЯ ПРОГРАМНО-КЕРОВАНИХ ІНФОКОМУНІКАЦІЙНИХ СИСТЕМ

08-34.МКР.013.00.000 ТЗ

Керівник роботи

к.т.н., доц. кафедри ТКСТБ ВНТУ

Васильківський М.В.

Виконавець: ст. гр. ТКС-20м

Щербань О.А.

Вінниця-2021

1 ПІДСТАВА ДЛЯ ВИКОНАННЯ РОБОТИ

Робота проводиться на підставі наказу ректора по Вінницькому національному технічному університету від “24” 09 2021 року № 277 та індивідуального завдання на магістерську кваліфікаційну роботу.

Дата початку роботи: 01.09.2021 р.

Дата закінчення: 20.12.2021 р.

2 МЕТА І ПРИЗНАЧЕННЯ МКР

Метою даної магістерської кваліфікаційної роботи є підвищення стійкості до відмов мікропроцесорних пристроїв за допомогою технічних і програмних засобів тестування на основі імітації несправності інформаційного комплексу.

Задачами магістерської кваліфікаційної роботи є:

- розробка технічного завдання;
- аналіз проблеми стійкості до відмов програмно-апаратних комплексів і підходів до забезпечення їх працездатності в процесі проектування;
- розробка алгоритму аналізу проблемних ситуацій, що виникають при роботі мікропроцесорного пристрою, на основі виявлення несправності за допомогою спотворення вхідних даних;
- розробка алгоритму пошуку несправності на основі спотворення вхідних даних основних програмних функцій з використанням попередньо опрацьованих статистичних показників роботи пристрою.
- створення технічних засобів імітації несправності і алгоритму діагностики для мікропроцесорних пристроїв в складі інформаційного комплексу;
- розробка методу і критерію визначення обсягу тестових випробувань на основі статистично оброблених експериментальних даних про роботу

пристрою. Створення математичної моделі для обгрунтування критерію і алгоритму прийняття рішень про достатність обсягу випробувань;

- реалізація і впровадження отриманих теоретичних результатів у вигляді методики, моделей, алгоритмів і програм, що використовуються при аналізі і синтезі мікропроцесорних пристроїв для інформаційного комплексу.

Об'єктом дослідження є способи дослідження стійкості до відмов мікропроцесорних пристроїв з програмним забезпеченням.

Предметом дослідження є методи, алгоритми та математичні моделі, що дозволяють забезпечити працездатність мікропроцесорного пристрою в ТКС.

Основними завданнями роботи є:

- техніко-економічне обґрунтування доцільності даної розробки;

- відмовостійкі пристрої інформаційних комплексів;

- програмно-апаратний пристрій тестування інфокомунікаційної системи;

- структура комплексу та алгоритми тестування телекомунікаційних мікропроцесорних пристроїв;

- результати досліджень пристроїв інформаційного комплексу.;

- аналіз економічної ефективності проведеної розробки;

- дослідження питань безпеки життєдіяльності.

Методики і алгоритм аналізу проблемних ситуацій автоматизованої радіостанції, інтегрованої в діючу інфраструктуру цифрових мереж з множинним доступом, який реалізує тестування функцій радіостанції за допомогою техніки фаззінга (випадкового внесення спотворень в дані) і виявляє системні дефекти на основі обробки інформації, отриманої за допомогою інтерфейсу налагодження JTAG;

Програмний комплекс для автоматизації проведення випробувань мікропроцесорних пристроїв (в комплексній інфраструктурі тестування), що включає програму внесення помилок у вхідні дані, а також програму імітації системних подій (виникнення умов прийому і передачі даних) і збору

інформації про реакцію пристрою на внесення спотворення з контрольно-вимірювальних приладів;

Імітаційна модель імітатора несправності для перевірки роботи мікропроцесорних пристроїв, що дозволяє виявити основні уразливості ПЗ пристрою шляхом порівняння параметрів його стану зі штатними значеннями, результатом якого є виявлення точок виникнення відмов і збоїв пристрою, що перевіряється.

3 ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ МКР

Робота базується на результатах звіту із переддипломної практики “Методи тестування програмно-керованих інфокомунікаційних систем”, який виконувався у ВНТУ у 2021/2022 н.р. Під час підготовки магістерської кваліфікаційної роботи будуть використані матеріали цього звіту.

Список використаних джерел розробки:

3.1 Скляр Б. Цифровая связь. Теоретические основы и применение / Бернард Скляр ; [пер. с англ]. – М.: Изд. Дом “Вильямс”, 2003. – 1104с.

3.2 Денисова, Л. А. Многокритериальная оптимизация на основе генетических алгоритмов при синтезе систем управления: монография / Л. А. Денисова; Ом. гос. техн. ун-т. – Омск: Изд-во ОмГТУ, 2014. – 170 с.

3.3 Денисова, Л. А. Модели и методы проектирования систем управления объектами с переменными параметрами: монография / Л. А. Денисова; Ом. гос. техн. ун-т. – Омск: Изд-во ОмГТУ, 2014. – 167 с.

3.4 Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. – Санкт-Петербург: Питер, 2018. – 351 с.

3.5 Положення про кваліфікаційні роботи на другому (магістерському) рівні вищої освіти у Вінницькому національному технічному університеті / Уклад. А. О. Семенов, Л. П. Громова, Т.В. Макарова, Сердюк О.В. – Вінниця: ВНТУ, 2021 – 60 с.

3.6 Кухарчук В.В., Ігнатенко О.Г., Обертюх Р.Р. Методичні вказівки до оформлення дипломних проектів (робіт) для студентів всіх спеціальностей.- В.: ВДТУ, 2002.

3.7 Козловський В.О. Техніко-економічні обгрунтування та економічні розрахунки в дипломних проектах та роботах. Навчальний посібник. – В.: ВДТУ, 2003.

3.8 ДСТУ 3008-2015. Інформація та документація, звіти у сфері науки і техніки.- К.: ДП «УкрНДНЦ», 2016.

3.9 Разработка и оформление конструкторской документации радиоэлектронной аппаратуры. Справочник. Под ред. Э.Т.Романычевой.- М: Радио и связь, 1989.

3.10 Бортник Г.Г., Васильківський М.В. Методичні вказівки до підготовки магістерських кваліфікаційних робіт для студентів спеціальності «Телекомунікації та радіотехніка» усіх форм навчання.- Вінниця:ВНТУ, 2018.- 50 с.

4 ВИКОНАВЕЦЬ

Вінницький національний технічний університет, кафедра телекомунікаційних систем та телебачення, студент групи ТКС-20 м Щербань О.А.

5 ВИМОГИ ДО ВИКОНАННЯ МКР

Пропонується виконати підвищення стійкості до відмов мікропроцесорних пристроїв за допомогою технічних і програмних засобів тестування на основі імітації несправності інформаційного комплексу телекомунікаційної системи.

Технічні вимоги, яким повинна відповідати розробка, наступні:

- тривалість пошуку помилок пам’яті в програмних ТКС при використанні стандартних засобів тестування – 60-100 год;

- тривалість пошуку помилок пам’яті в програмних ТКС при використанні запропонованих методів тестування – 12 год;

- тривалість пошуку спотворення даних в програмних ТКС при використанні стандартних засобів тестування – 65-90 год;

- тривалість пошуку спотворення даних в програмних ТКС при використанні запропонованих методів тестування – 17-32 год;

- тривалість пошуку спотворення тривалості потоків в програмних ТКС при використанні стандартних засобів тестування – 92-110 год;

- тривалість пошуку спотворення тривалості потоків в програмних ТКС при використанні запропонованих методів тестування – 8-14 год;

- підвищення ефективності виявлення всіх помилок в програмному забезпеченні ТКС – 82%;

- підвищення ефективності виявлення помилок, здатних призводити до відмов та збоїв в програмному забезпеченні ТКС – 5-19%.

При розробці телекомунікаційних систем слід максимально використовувати стандартні та уніфіковані деталі.

6 ЕТАПИ МКР І ТЕРМІНИ ЇХ ВИКОНАННЯ

№	Назва та зміст етапу	Термін виконання		Очікувані результати	Звітна документ-тація
		початок	закінчення		
1.	Розробка технічного завдання (ТЗ)	01.09.2021р.	10.09.2021р.	Розроблене ТЗ	Додаток А
2.	Відмовостійкі пристрої інформаційних комплексів	11.09.2021р.	17.09.2021р.	Проведений аналіз	Вступ. Розділ 1.
3.	Програмно-апаратний пристрій тестування інфокомунікаційної системи	18.09.2021р.	01.10.2021р.	Проведений аналіз	Розділ 2

4.	Структура комплексу та алгоритми тестування телекомунікаційних мікропроцесорних пристроїв	02.10.2021р	29.10.2021р.	Розроблений метод	Розділ 3
5.	Результати досліджень пристроїв інформаційного комплексу	30.10.2021р.	19.11.2021р.	Розроблений алгоритм	Розділ 4
6.	Аналіз економічної ефективності	20.11.2021р.	30.11.2021р.	Економічна частина МКР	Розділ 5
7.	Охорона праці та безпека в надзвичайних ситуаціях	01.12.2021р.	06.12.2021р.	Частина ОТ та БНС	Розділ 6
8.	Оформлення пояснювальної записки (ПЗ) та графічної частини	07.12.2021р.	13.12.2021р.	Оформлена документація	ПЗ та графічна частина
9.	Нормоконтроль, попередній захист, опонування МКР	14.12. 2021р.	17.12.2021р.	Позитивні відзиви	Відгуки
10.	Захист МКР ЕК		20.12.2021р.	Позитивний захист	Протокол ЕК

7 ОЧІКУВАНІ РЕЗУЛЬТАТИ ТА ПОРЯДОК РЕАЛІЗАЦІЇ МКР

В результаті виконання роботи будуть розроблені:

- загальна структура імітатора несправності;
- архітектура системи автоматизованого тестування;
- структурна схема пристрою імітації несправності та аналізу реакції;
- алгоритм адаптивної передачі інформації;
- алгоритм тестування пристрою за допомогою JTAG;
- функціональна схема комплексу проведення випробувань;
- структурна схема імітаційних досліджень;

- структура програмно-апаратних компонентів налагодження;
- економічна частина МКР;
- розділ ОП та БНС;
- рекомендації щодо подальшого використання розробленого алгоритму.

Результати, отримані в процесі виконання даної роботи, будуть впроваджені в галузі телекомунікацій:

- Регіональний Центр експлуатації телекомунікаційної мережі України шляхом впровадження алгоритму аналізу проблемних ситуацій, що виникають при роботі автоматизованої радіостанції, інтегрованої в діючу інфраструктуру цифрових мереж з множинним доступом. Алгоритм реалізує тестування функцій радіостанції за допомогою техніки фаззінга (випадкового внесення спотворень в дані). Це дозволяє виконувати автоматичну діагностику технічного стану радіостанції та усувати системні помилки, що виникають при завантаженні призначених для користувача програм в реальне апаратне середовище пристрою (прояв властивості емерджентності);

- ПАТ “Укртелеком” шляхом впровадження алгоритму діагностики програмно-апаратного комплексу, який дозволяє оптимізувати швидкість передачі даних шляхом пошуку найкоротшого шляху в сукупності з ранжируванням інтерфейсів прийому / передачі при наявності великого числа інтерфейсів (вузлів). Підтверджено ефективність алгоритму в умовах обмеження на час обміну інформацією між вузлами (час зменшується приблизно в два рази)..

Очікуваний техніко-економічний ефект. Запропоновано пристрій імітації несправності в програмно-апаратних системах для перевірки роботи мікропроцесорного пристрою і розроблений експертний алгоритм, призначений для виявлення найбільш вразливих місць програми і заснований на аналізі ознак відмов і збоїв в регістрах ОЗП мікропроцесорного пристрою. Алгоритм дозволяє спотворити вхідні дані і виявити зміни характеристик

роботи пристрою, що перевіряється, обумовлених зміною порядку і часу виконання програмних функцій, що свідчить про наявність відмов або збоїв.

Розроблено метод і запропоновано критерій визначення обсягу тестових випробувань мікропроцесорних пристроїв. Створено математичну модель для обґрунтування критерію, що враховує попередньо отримані характеристики пристрою: число апаратних відмов, ймовірність помилок при виконанні програмних функцій, а також кореляцію між часом внесення помилок у вхідні дані і часом виявлення несправності. Метод, заснований на попередній статистичній обробці експериментальних даних про виконувані функції в сукупності з алгоритмом прийняття рішення на базі нечіткого логічного висновку, дозволяє оцінити достатність обсягу випробувань.

8 МАТЕРІАЛИ, ЯКІ ПОДАЮТЬ ПІСЛЯ ЗАКІНЧЕННЯ РОБОТИ ТА ПІД ЧАС ЕТАПІВ

За результатами виконання МКР до ЕК подаються пояснювальна записка, графічна частина МКР, відзив і рецензія.

9 ПОРЯДОК ПРИЙМАННЯ МКР ТА ЇЇ ЕТАПІВ

Поетапно результати виконання МКР розглядаються керівником роботи та обговорюються на засіданні кафедри.

Захист магістерської кваліфікаційної роботи відбувається на відкритому засіданні ЕК.

10 ВИМОГИ ДО РОЗРОБЛЮВАНОЇ ДОКУМЕНТАЦІЇ

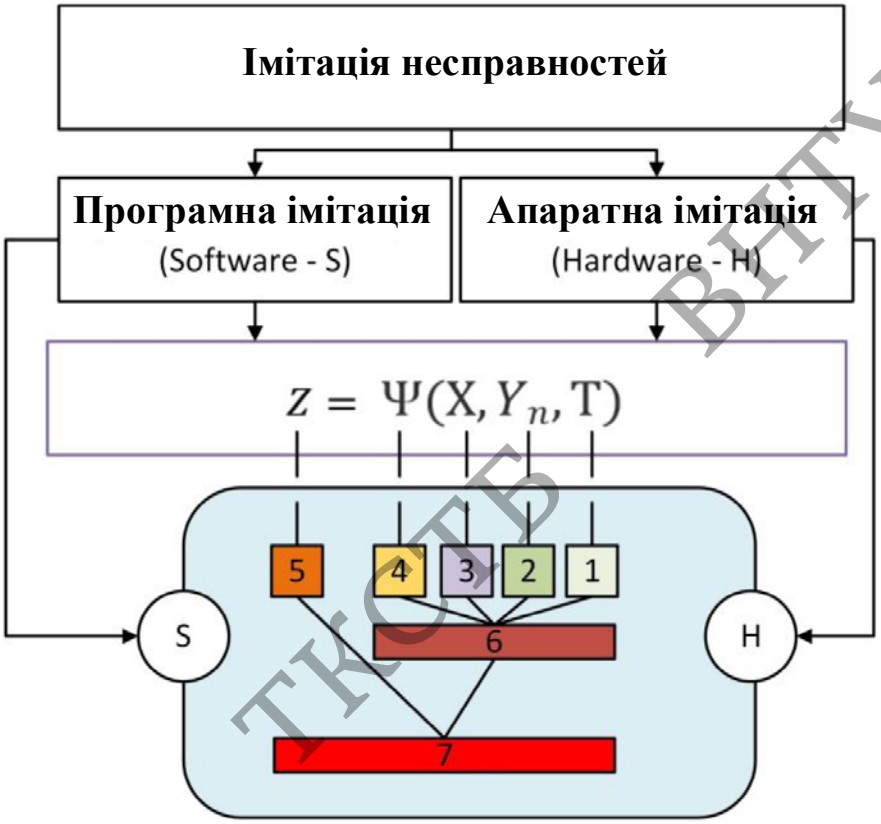
Документація, що розробляється в процесі виконання досліджень повинна містити:

- техніко-економічне обґрунтування розробки;
- загальну структуру імітатора несправності;

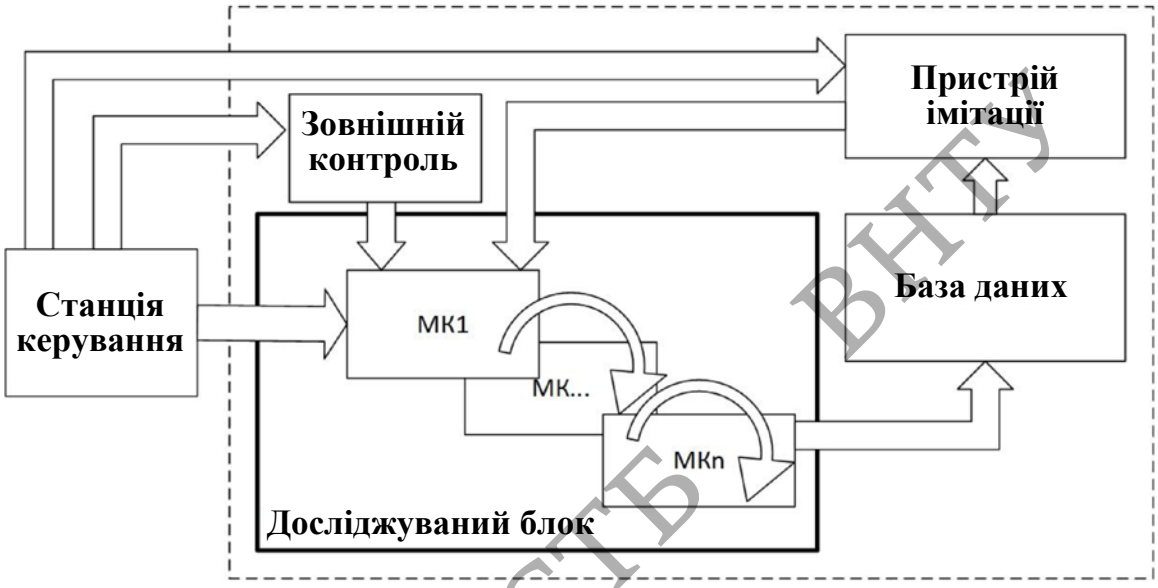
- архітектуру системи автоматизованого тестування;
- структурну схему пристрою імітації несправності та аналізу реакції;
- алгоритм адаптивної передачі інформації;
- алгоритм тестування пристрою за допомогою JTAG;
- функціональну схему комплексу проведення випробувань;
- структурну схему імітаційних досліджень;
- структуру програмно-апаратних компонентів налагодження;
- економічну частину та розділ БЖД і ЦЗ;
- рекомендації щодо подальшого використання алгоритму.

11 ВИМОГИ ЩОДО ТЕХНІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ З ОБМЕЖЕНИМ ДОСТУПОМ

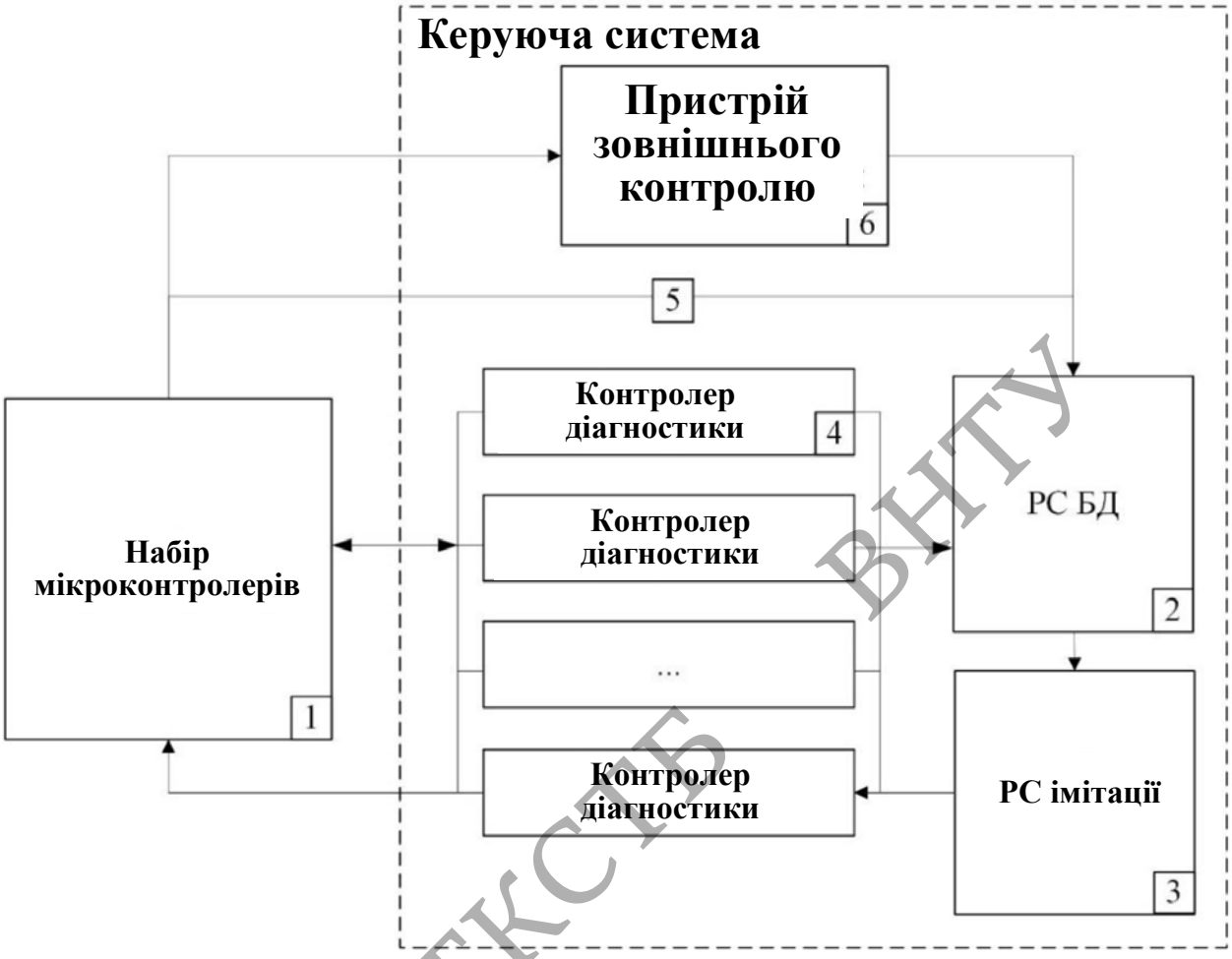
У зв’язку з тим, що інформація не є конфіденційною, заходи з її технічного захисту не передбачаються.



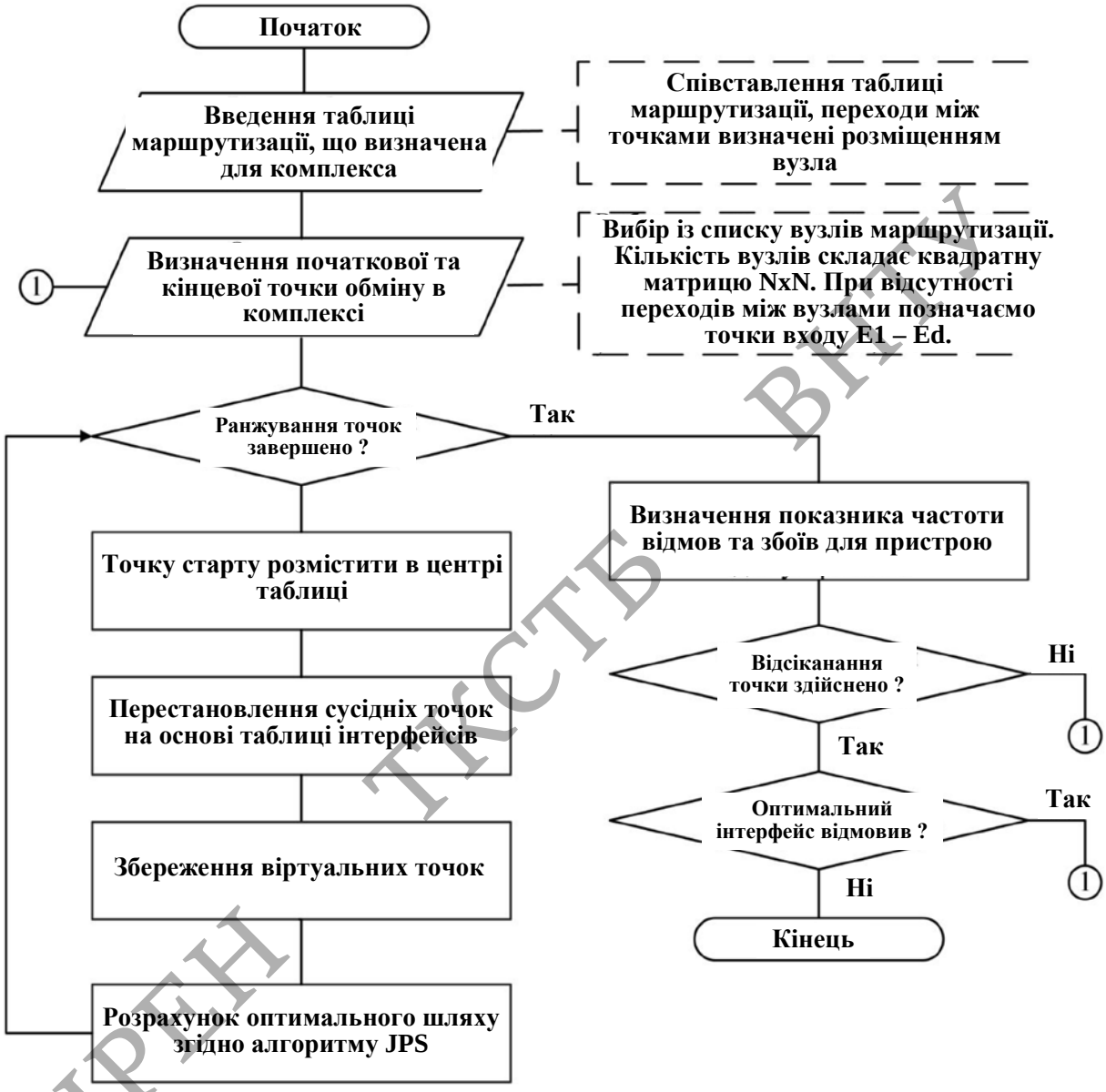
					08-34. МКР.013.00.000 Е8		
Змн.	Лист	№ докум.	Підпис	Дата			
Розроб.		Щербань О.А.			Загальна структура імітатора несправності	Літ.	Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.					
Реценз.						ВНТУ, гр. ТКС-20м	
Н. Контр.		Васильківський М.В.					
Затверд.		Кичак В.М.					



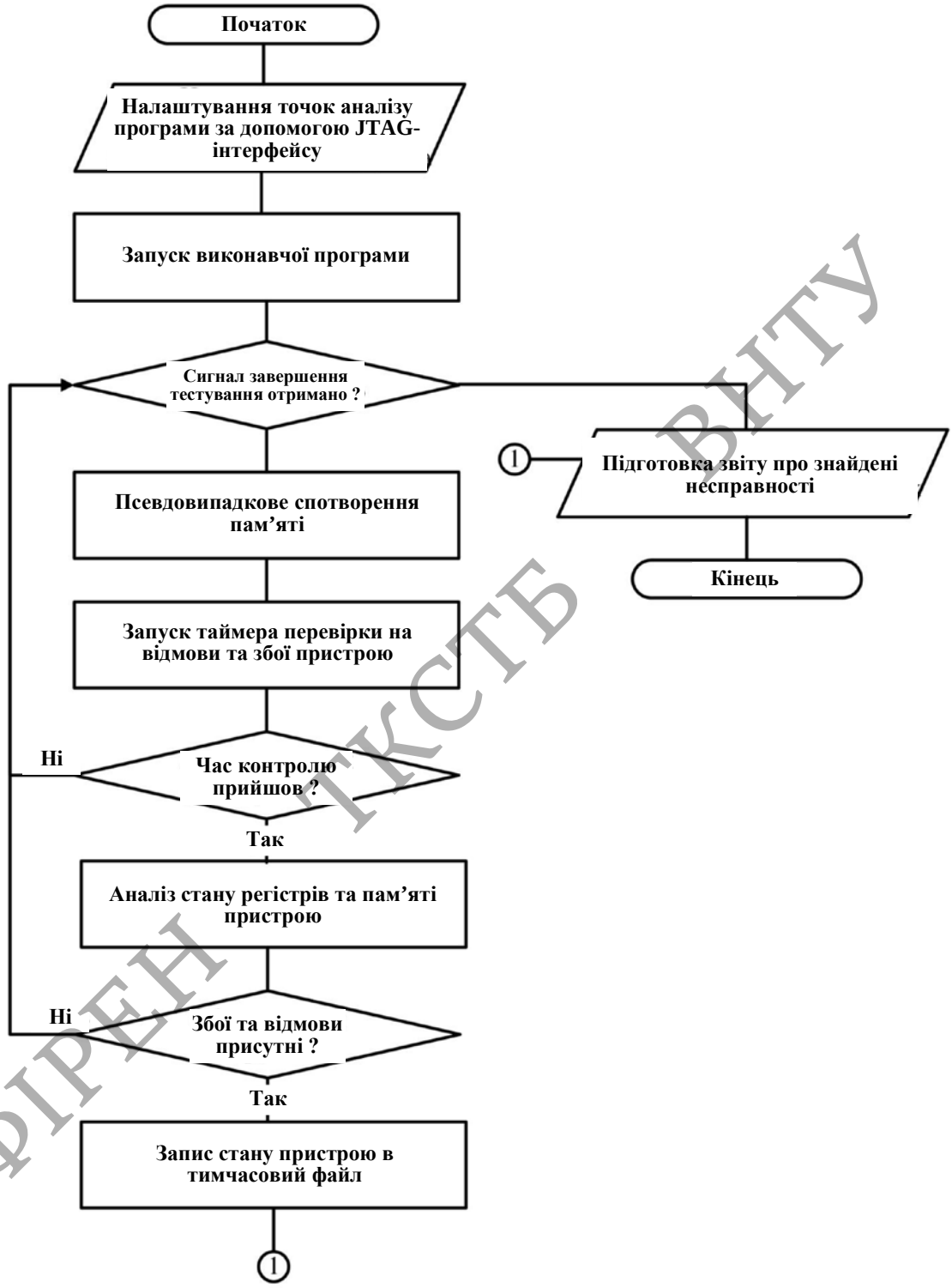
					08-34. МКР.013.00.000 Е8		
Змн.	Лист	№ докум.	Підпис	Дата			
Розроб.		Щербань О.А.			Архітектура системи автоматизованого тестування	Літ.	Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.					
Реценз.						ВНТУ, гр. ТКС-20м	
Н. Контр.		Васильківський М.В.					
Затверд.		Кичак В.М.					



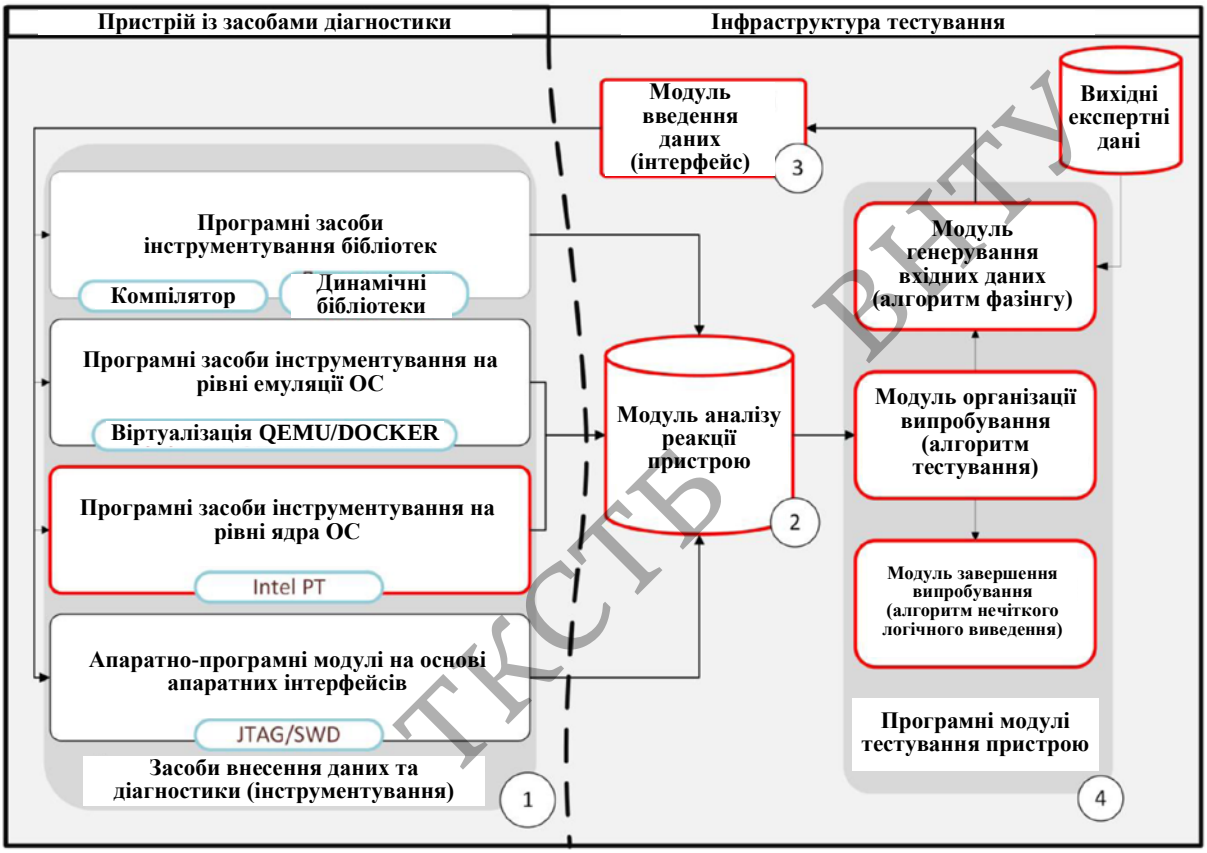
					08-34. МКР.013.00.000 Е8	
Змн.	Лист	№ докум.	Підпис	Дата		
Розроб.		Щербань О.А.			Структурна схема пристрою імітації несправності та аналізу реакції	Літ. / Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.				
Реценз.						ВНТУ, гр. ТКС-20м
Н. Контр.		Васильківський М.В.				
Затверд.		Кичак В.М.				



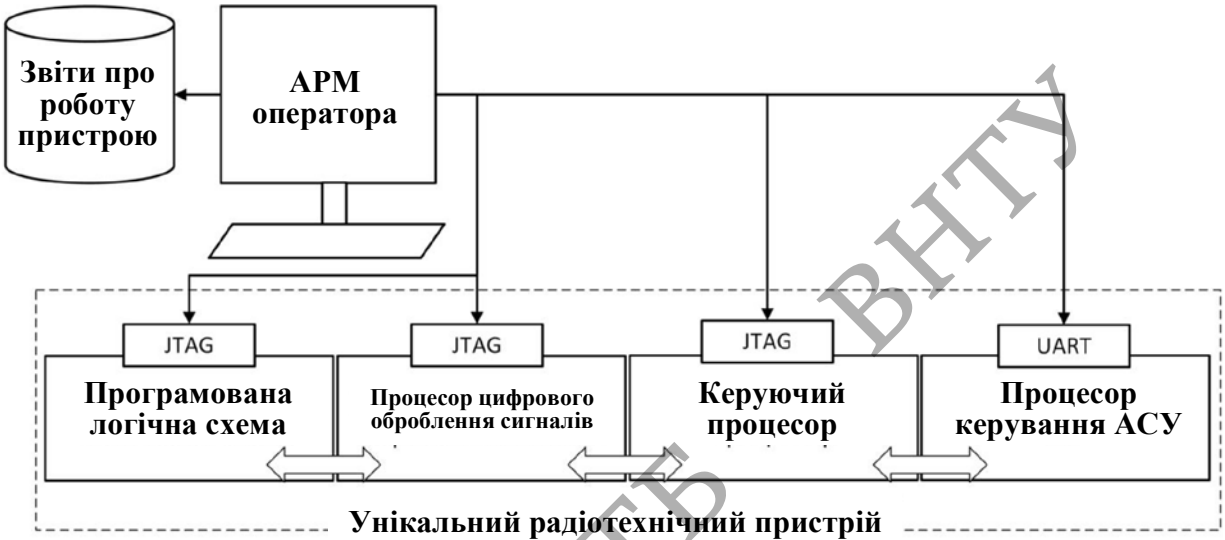
					08-34. МКР.013.00.000 Е8		
Змн.	Лист	№ докум.	Підпис	Дата			
Розроб.		Щербань О.А.			Алгоритм адаптивної передачі інформації	Літ.	Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.					
Реценз.						ВНТУ, гр. ТКС-20м	
Н. Контр.		Васильківський М.В.					
Затверд.		Кичак В.М.					



					08-34. МКР.013.00.000 Е8	
Змн.	Лист	№ докум.	Підпис	Дата		
Розроб.		Щербань О.А.			Алгоритм тестування пристрою за допомогою JTAG	Літ. / Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.				
Реценз.						ВНТУ, гр. ТКС-20м
Н. Контр.		Васильківський М.В.				
Затверд.		Кичак В.М.				



					08-34. МКР.013.00.000 Е8	
Змн.	Лист	№ докум.	Підпис	Дата		
Розроб.		Щербань О.А.			Функціональна схема комплексу проведення випробувань	Літ. / Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.				
Реценз.						ВНТУ, гр. ТКС-20м
Н. Контр.		Васильківський М.В.				
Затверд.		Кичак В.М.				



					08-34. МКР.013.00.000 Е8		
Змн.	Лист	№ докум.	Підпис	Дата			
Розроб.		Щербань О.А.			Структурна схема імітаційних досліджень	Літ.	Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.					
Реценз.						ВНТУ, гр. ТКС-20м	
Н. Контр.		Васильківський М.В.					
Затверд.		Кичак В.М.					

					08-34. МКР.013.00.000 Е8		
Змн.	Лист	№ докум.	Підпис	Дата			
Розроб.		Щербань О.А.			Структура програмно-апаратних компонентів налагодження	Літ.	Арк. 1 / Аркушів 1
Перевір.		Васильківський М.В.					
Реценз.						ВНТУ, гр. ТКС-20м	
Н. Контр.		Васильківський М.В.					
Затверд.		Кичак В.М.					