Національний університет «Полтавська політехніка імені Юрія Кондратюка»

(повне найменування вищого навчального закладу)

Навчально-науковий інститут інформаційних технологій та робототехніки

(повна назва факультету)

Кафедра комп’ютерних та інформаційних технологій і систем

(повна назва кафедри)

**Пояснювальна записка**

**до дипломного проекту (роботи)**

|  |
| --- |
| бакалавра |

(рівень вищої освіти)

на тему

|  |
| --- |
| Автоматизація тестування application function block library компонентів |
| за допомогою LabVIEW SW та Test Stand для апаратно-програмних |
| комплексів |
|  |

Виконав: студент 4 курсу, групи 402-ТК

спеціальності

|  |
| --- |
| 123 Комп’ютерна інженерія |
| (шифр і назва спеціальності)  Головко А.В. |
| (прізвище та ініціали)  Керівник Руденко О.А. |
| (прізвище та ініціали)  Рецензент |
| (прізвище та ініціали) |

Полтава – 2021 року

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**«ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»**

**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ**

**КАФЕДРА КОМП’ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ**

**КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА**

**спеціальність 123 «Комп’ютерна інженерія»**

**на тему**

**«****Автоматизація тестування application function block library компонентів за допомогою LabVIEW SW та Test Stand для апаратно-програмних комплексів»**

**Студента групи 402-ТК Головка Анатолія Вячеславовича**

Керівник роботи

кандидат технічних наук,

доцент Руденко О.А

Завідувач кафедри

кандидат технічних наук,

доцент Головко Г.В.

Полтава – 2021

**РЕФЕРАТ**

Кваліфікаційна робота бакалавра: 55 с., 14 малюнків, 27 джерел.

**Об’єкт дослідження:** цифрова інформаційно-керуюча платформа RadICS.

**Мета дипломної роботи:** автоматизація процесу тестування, що дозволяє вилучити людський фактор та значно скоротити час тестування. Забезпечення якості, з урахуванням всіх або найбільш критичних для даного конкретного випадку складових.

**Методи:** розробка алгоритму та конфігурації вIDERPCT. Використання National Instruments (NI), а також спеціальне програмне забезпечення (Test Stand).

**Ключові слова:**Application Logic,якість та надійність програмного забезпечення,валідація, Radiy Platform Configuration Tool (RPCT), AFBL, IEC 61508 SIL 3, Instrumentation&Controlsystem (I&C), Integration testing.

**ABSTRACT**

Qualifying work of the bachelor: 55p., 14 pictures, 27sources.

**Object of study:** information and control digital platform RadICS.

**Purpose:** automation of the testing process, which allows you to remove the human factor and significantly reduce testing time. Quality assurance, taking into account all or the most critical for this particular case components.

**Methods:** algorithm development and configuration in RPCT IDE. Use of National Instruments (NI) as well as special software (TestStand).

**Keywords:** Application Logic, software quality and reliability, validation, Radiy Platform Configuration Tool (RPCT), AFBL, IEC 61508 SIL 3, Instrumentation & Control system (I&C), Integration testing.

**ЗМІСТ**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ………………6

ВСТУП………………………………………………………………………………7

[РОЗДІЛ 1 ОРГАНІЗАЦІЯ ПРОЦЕСІВ ВАЛІДАЦІЇ ТА ЇХ МЕТОДИ 8](#_Toc74665564)

[1.1 Процеси валідації та методи тестування 8](#_Toc74665566)

[1.2 Моделі розробки ПЗ 10](#_Toc74665567)

[1.3 Тестована система та конфігурація 14](#_Toc74665568)

[1.4 Тестування логіки в ручному режимі 16](#_Toc74665569)

[РОЗДІЛ 2 АВТОМАТИЗАЦІЯ ТЕСТУВАННЯ МАТЕМАТИЧНИХ ТА ЛОГІЧНИХ КОМПОНЕНТІВ, ЩО ОБРОБЛЯЮТЬСЯ В ЛОГІЧНОМУ МОДУЛІ 20](#_Toc74665570)

[2.1 Вимоги до тестування програмних модулів 20](#_Toc74665572)

[2.2 Розробка нової моделі для тестування AFBL компонентів за допомогою LABVIEW SW ТА TEST STAND для апаратно-програмних комплексів 26](#_Toc74665573)

[2.3 Устаткування конфігурації системи та прошивка логічного модуля 27](#_Toc74665574)

[РОЗДІЛ 3 ЕФЕКТИВНІСТЬ ВИКОРИСТАННЯ АВТОМАТИЗАЦІЇ ПРОЦЕСУ ВАЛІДАЦІЇ 30](#_Toc74665575)

[3.1 Ефективність використання автоматизації при тестуванні AFBL компонентів для апаратно-програмного комплексу 30](#_Toc74665577)

[3.2 Причини вимірювання ефективності. 32](#_Toc74665578)

[3.2 Вплив на ефективність автоматизації. 35](#_Toc74665579)

[РОЗДІЛ 4 ФУНКЦІОНАЛЬНА БЕЗПЕКА СИСТЕМ ЕЛЕКТРИЧНИХ, ЕЛЕКТРОННИХ, ПРОГРАМОВАНИХ ЕЛЕКТРОННИХ, ПОВ’ЯЗАНИХ З БЕЗПЕКОЮ 37](#_Toc74665580)

[4.1 Опис функціональної безпеки систем електричних, електронних, програмованих електронних, пов’язаних з безпекою 37](#_Toc74665582)

[4.2 Область застосування 40](#_Toc74665583)

[4.3 Оцінка функціональної безпеки 44](#_Toc74665584)

[ВИСНОВКИ 50](#_Toc74665585)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 51](#_Toc74665586)

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ**

**AFB** – Application Function Block.

**AFBL** – Application Function Block Library.

**AL** – Логіка додатків.

**SUT** – System Under Test.

**ПЗ** – Програмне забезпечення.

**NI** – National Instruments.

**SW** – Software.

**HW** – Hardware.

**TB** – Test Bed.

**UAL** – User Application Logic.

**FT** – Functional Testing.

**VT** – Validation Testing.

**RPCT** – Radiy Platform Configuration Tool.

**I&C** – Instrumentation&Controlsystem.

**SIL** – Safety Integrity Level.

**IEC** – The International Electrotechnical Commission.

**IDE** – Integrated development environment.

**EU** – Engineering units.

**V&V** – Verification and validation.

**I/O** – Input/Output.

**АЕС** – атомна електростанція.

**ПП** – програмний продукт.

**ЖЦ ПЗ** – життєвий цикл програмного забезпечення.

**ВСТУП**

На сьогоднішній день тестування є важливим етапом в життєвому циклі розробки ПЗ. Головна мета забезпечити якість, безпеку та надійність.

У ході розробки математичних та логічних AFB компонентів, що виконують важливу роль при створенні алгоритмів, які обробляються в сучасних комплексах для АЕС, програмують нові компоненти та удосконалюють старі.

Ручне тестування займає час на розробку алгоритму по тестуванню, задавання вхідних параметрів дискретних або аналогових сигналів. Фіксація та порівняння отриманого результату з очікуваним та етап документування.

Перевірка в ручному режимі займає близько трьох тижнів, в цей період входить:

* налаштування конфігурації;
* розробка алгоритмів компонентних схем;
* прошивка модуля [.BTS] файлом, в якому прописані конкретні сигнали I/O між AFB-блоком;
* зміна параметрів вхідного сигналу та фіксація результатів.

В час прогресу з’явилась можливість скоротити час на тестування, автоматизувавши алгоритми для перевірки компонентів.

Дипломна робота присвячена автоматизації тестування Application Function Block Library компонентів за допомогою LabVIEWSW та Test Stand для апаратно-програмних комплексів, в ній описана роль тестування в життєвому циклі розробки ПП, методи перевірки вимог, автоматизація та оптимізація тестів.

# РОЗДІЛ 1

# ОРГАНІЗАЦІЯ ПРОЦЕСІВ ВАЛІДАЦІЇ ТА ЇХ МЕТОДИ

## **Процеси валідації та методи тестування**

У роботі з програмним забезпеченням валідацією вважається специфічний процес, метою якого є доказ що під час функціонування системи досягаються результати, спочатку заплановані для її безпосереднього використання. Іншими словами, валідація – це перевірка відповідності створеної системи того очікування, яке озвучив клієнт.

Процеси верифікації та валідації (V&V) використовуються для того, щоб визначити, чи відповідають продукти розробки даної діяльності вимогам цієї діяльності та чи задовольняє виріб своє цільове використання та потреби користувачів. Вимоги до процесу життєвого циклу V&V визначені для різних рівнів цілісності. Сфера процесів V&V охоплює системи, програмне та апаратне забезпечення, а також їх інтерфейси. Цей стандарт поширюється на системи, програмне та апаратне забезпечення, що розробляються, обслуговуються або використовуються повторно (застарілі, комерційні готові [COTS], предмети, що не розробляються).

Термін програмне забезпечення також включає мікропрограму та мікрокод, а кожен термін система, програмне забезпечення та обладнання включає документацію. Процеси V&V включають аналіз, оцінку, огляд, перевірку, оцінку та випробування продукції.

Ручне тестування (manualtesting) – тестування, в якому тест-кейси виконуються людиною вручну без використання засобів автоматизації. Незважаючи на те що це звучить дуже просто, від тестувальника в ті чи інші моменти часу потрібні такі якості, як терплячість, спостережливість, креативність, вміння ставити нестандартні експерименти, а також уміння бачити і розуміти, що відбувається «всередині системи»,тобто як зовнішні впливи на додаток трансформуються в його внутрішні процеси.

Вимоги – це опис функцій чи апаратний вигляд продукту, який очікує отримати замовник на виході. Їх можна поділити на такі групи:

* вимоги до системи;
* вимоги software;
* вимоги hardware;
* вимоги кібербезпеки.

В сфері тестування можна виділити чотири методи:чорної, білої, сірої скриньки та перевірка не функціональних аспектів програм.

При використанні метода білої скриньки, розробник тест кейсу може бачити вихідний код, який стосується бібліотек ПО. Це стандартно для Unittesting, при якому частини системи перевіряються окремо. Воно забезпечує якість, працездатність та стійкість.

При використанні метода чорної скриньки, валідатор тестує продукт останньої фази розробки. Наприклад, програма для смартфону, яка в залежності від локації автоматично налаштоває погоду. Тестувальник не має програмного коду, але може використовувати принцип засіву дефектів в програмне чи апаратне забезпечення.

На етапі валідації саме вона використовує цей метод, в життєвому циклі проекту складаються три типи документів:

* план тестування де перелічені вимоги;
* специфікація з розробленими тест кейсами;
* звіт з тестування системи, чи окремого об’єкта.

Функціональне тестування – найскладніший вид та найдовший, зазвичай займає 90% часу.

Метою валідаційних тестувань є.

1. Показати замовнику, що програмне чи апаратне забезпечення відповідає заявленим вимогам.
2. Знайти ситуації, коли SW чи HW відпрацьовує помилково, з деякими дефектами або не відповідає специфікації.
3. Створити передумови для попередження виникнення помилок в програмному забезпеченні.

Кінцевою метою будь-якого процесу тестування є забезпечення якості, з урахуванням всіх, або найбільш критичних для даного конкретного випадку складових.

## **Моделі розробки ПЗ**

Для реалізації запланованих ідей, інтеграції продукту на ринок, існують моделі життєвого циклу ПЗ – моделі розробки програмного забезпечення.

Нажаль розробка не завжди відповідає життєвому циклу.

На процес перевірки вимог вибір моделі розробки ПЗ значно впливає, а саме, підбір стратегії, необхідні ресурси та розклад запланованих дій.

Моделей багато, але до основних можна віднести:

* водоспадна модель;
* V-подібна модель;
* спіральна модель;
* гнучка модель.

Водоспадна модель (waterfall, рис. 1.1) на цей час застаріла, тому що в сучасних проектах практично непридатна. Вона виконує кожну з фаз проекту одноразово, які суворо слідують одна за одною. Дуже спрощено можна сказати, що в рамках цієї моделі, будь-яка команда бачить лише попередню і наступну фазу.

У реальному житті розробці ПЗ доводиться «бачити весь проект цілком» і повертатися до попередніх фаз, щоб виправити недоробки або щось уточнити.

V-модель (рис. 1.2) є логічним подальшим розвитком водоспадних моделей. Слід зазначити, що в загальному випадку як водоспадні, так і V-подібні моделі життєвого циклу програмного забезпечення можуть містити однакові стадії. Однак принципова відмінність полягає в тому, як ця інформація використовується при реалізації проекту.

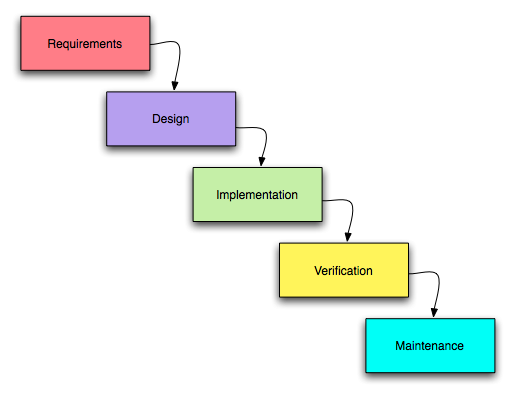


Рисунок 1.1 **–** Водоспадна модель

Дуже легко сказати, що при використанні V-подібної моделі на кожному етапі «спуску» потрібно думати, що і як буде відбуватися. Відповідний етап «на підйомі». Тестування тут проводиться на самих ранніх етапах розробки проекту, що мінімізує ризики, і крім того, виявляє і усуває безліч потенційних проблем до того, як вони виникнуть.

Спіральна модель (рис. 1.3) – це окремий випадок ітеративної поступової моделі, в якій особлива увага приділяється управлінню ризиками,зокрема організації процесу розробки проекту та контрольних пунктів.

З точки зору тестування та управління якістю, підвищена обізнаність про ризик є відчутною перевагою, коли спіральна модель використовується для розробки концептуальних проектів, де вимоги за своєю суттю є складними та нестабільними (вони можуть змінюватися кілька разів протягом проекту).

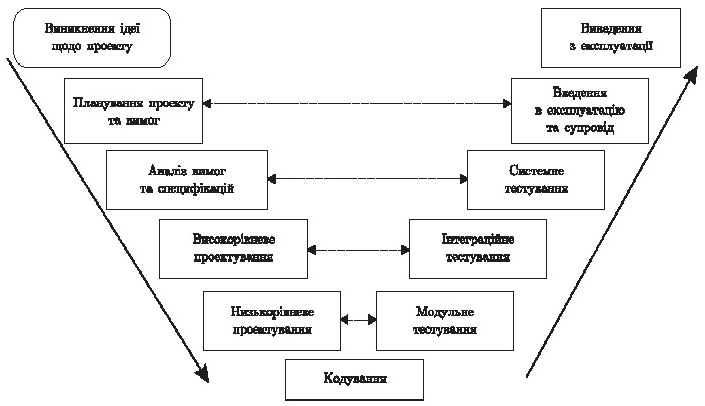


Рисунок 1.2 **–** V-подібна модель

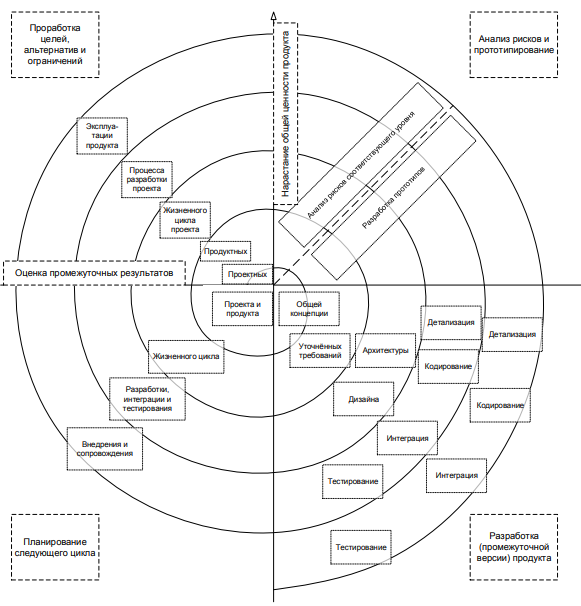


Рисунок 1.3 **–** Спіральна модель

Гнучка модель (рис. 1.4) розробки програмного забезпечення включає компіляцію різних підходів. Сюди входить ряд продуктових підходів, орієнтованих на ітераційну модель, а також динамічне формулювання та реалізація вимог за допомогою безперервної взаємодії самоорганізованих дисциплінарних команд.

Кожна ітерація – це невеликий проект. Одним із керівних принципів гнучкої моделі є особиста взаємодія між замовником та командою розробників.

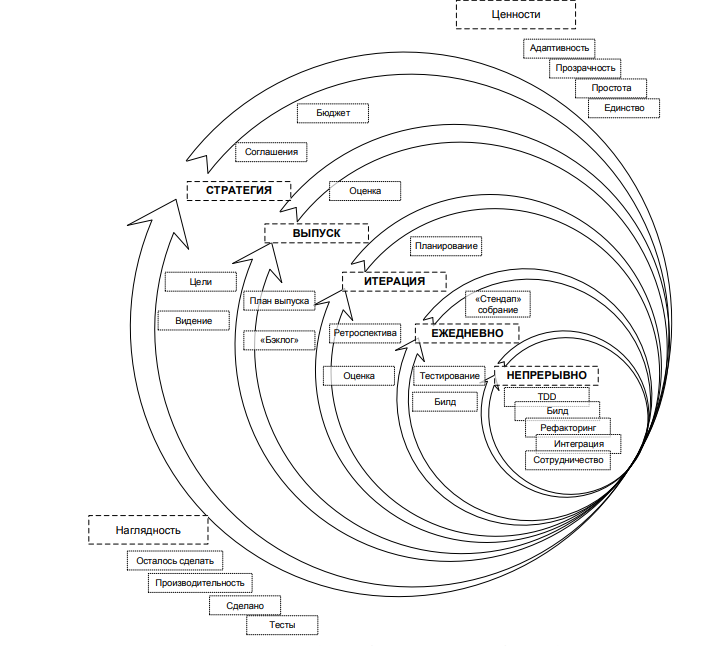


Рисунок 1.4 **–** Гнучка модель

## **Тестована система та конфігурація**

Платформа RadICS являє собою модульну платформу і включає в себе стандартизовані модулі, такі як логічний модуль, цифрові і аналогові модулі вводу / виводу (I/O), кожен з яких заснований на використанні мікросхем FPGA в якості обчислювальних механізмів. У всьому світі цифрове обладнання Radiy встановлено більш ніж в 70 основних системах безпеки та управління на діючих атомних електростанціях (АЕС).

Фундаментальна архітектура платформи RadICS складається з єдиного приладового шасі, що містить логічний модуль, а також до 14 інших модулів вводу-виводу і оптоволоконного зв’язку. Ця одноканальна конфігурація є конфігурацією цифрової платформи I & C, підтримує IEC 61508 SIL 3, призначеної для ядерної безпеки. У різних конфігураціях система на базі платформи RadICS може бути спроектована і виготовлена відповідно до вимог США класу 1E і підтримує вимоги і функції конкретних установок і систем.

На етапі інтеграційного тестування RPCT, а саме перевірки вимог по AFBL, потрібно створити в IDERPCT конфігурацію модулів.

Головним кроком є налаштування модуля “LM”, який виконує всю логіку. Логічний модуль (рис. 1.5) служить мозком для всієї платформи. Окрім виконання логіки додатків, логічний модуль взаємодіє з усіма модулями платформи, встановленими в шасі, виконує та контролює самодіагностику та контролює інші шасі системи.

Швидка і детермінована продуктивність з використанням сучасної технології FPGA має модуль LM. Час відгуку всього 5 Мілісекунд!

Цей модуль сертифікований за стандартом IEC 61508 SIL 3 для одно канальних і багатоканальних конфігурацій.

Для зв’язку з сервісами обробки інформації, діагностики та tuning(передача дискретного або аналогового сигналу I/O) потрібно вказати IP-адреси цих сервісів.

Передостаннім є розробка алгоритмів для перевірки логічних та математичних компонентів (рис. 1.6, 1.7).

Коли проект відповідає вимогам поставлених для тестування, потрібно прошити логічний модуль BTS файлом, який буде скомпільований в ході розробки конфігурації RPST.



Рисунок 1.5 – Модуль LM

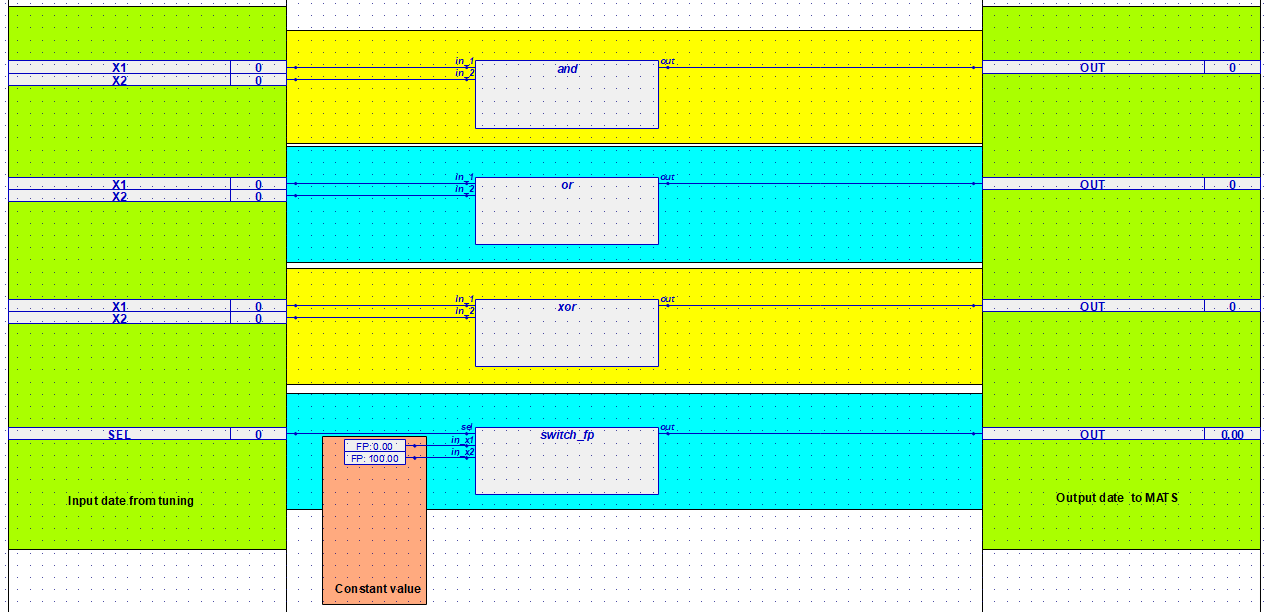


Рисунок 1.6 **–** Розробка алгоритму для перевірки AFBL blocks (AND, OR, XOR, SWITCH\_FP)

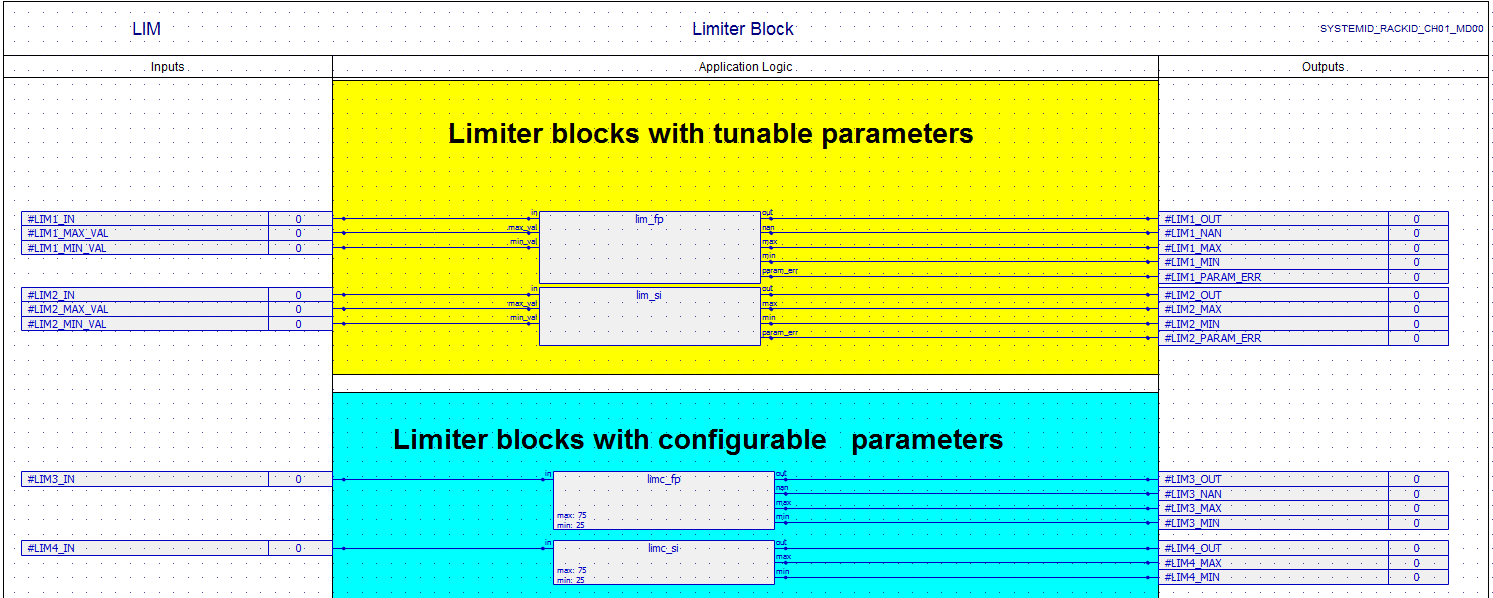


Рисунок 1.7 – Розробка алгоритму для перевірки AFBL blocks (LimiterBlock)

## **1.4 Тестування логіки в ручному режимі**

Коли платформа має коректну конфігурацію, тобто модулі в шасі знаходяться на відповідних місцях, та пройшли етап калібрування. Почати потрібно з запуску сервісів (спеціальних файлів с розширенням .bat)на робочій станції (ПК).

Усі модулі повинні бути в режимі “RUN”. Цей режим інформує про те, що помилки відсутні, конфігурація модулів вірна (модулі знаходяться на своїх місцях), та є навантаження каналів.

Режими роботи модулів:

* POWERED-OFF– цей режим інформує про відключене живлення;

Start UP – в цьому режимі FPGA отримує свою конфігурацію з EEPROM.

Потім FPGA починає свою нормальну роботу, виконання application logic та виконує тести самодіагностики. Переходячи в режим RUN (SAFE) – вся логіка працює, але виходи перекриваються функцією блокування безпеки (SOR-SafetyOverride) та підтримуються в безпечному стані. Для переходу в режим RUN потрібна дія оператора. Цей режим присутній тільки в модулях виводу.

RUN – у цьому режимі вся логіка програми працює і контролює результати. Виконуються самодіагностичні тести в режимі RUN.

TUNING – у цьому режимі сигнали типу Internal, оператор має змогу змінювати параметри, які передбачені дизайном. Для переходу в цей режим потрібно повернути ключ.

FAULTED – у цьому режимі всі виходи переведені в безпечний стан. Цей режим є результатом виявлення критичної несправності. Єдиним виходом з цього режиму є вимкнення FSC.

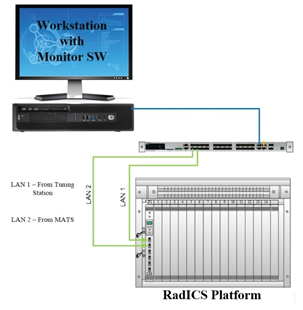


Рисунок 1.8 – SUT для ручного тестування

Сервіси для коректної роботи апаратно-програмного комплексу.

* Application date service – потрібен для перевірки та опрацювання логіки.
* Configuration service – цей сервіс перевіряє конфігурацію модулів в шасі, чи в правильній послідовності чи кількості підключені плати.
* Tuning service – дає змогу змінювати значення сигналів вхідних модулів.
* Diag sevice – надає інформацію про стан кожного модуля, та транслює дані кожного каналу.

MATS дає змогу відкрити потрібний нам алгоритм перевірки компонента(-ів), в якому розглянемо приклад тестування Limiter Block (Блок обмежувача).

Цей компонент виконує функцію обмеження в діапазоні min/max вихідного сигналу.

Принцип роботи блока вказаний на рис. 1.8.

На вході маємо таблицю з кроками для перевірки AFB-блока (табл. 1.1).

Таблиця 1.1 **–** Таблиця для проведення тестування з результатами, які очікуємо отримати

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#крок** | **Вхідні та вихідні сигнали** | | | | | | | | | |
| **Вхідний сигнал** | | | **Вихідний сигнал** | | | | | | |
| **in** | **max\_val** | **min\_val** | **out** | **nan** | **max** | **min** | **param\_err** | | |
| **FLOATING POINT LIMITER block with tunable parameters [Step 1]** | | | | | | | | | | |
| 1 | 0.00 | 0.00 | 0.00 | 0.00 | 0 | 0 | 0 | | 1 | |
| 2 | 100.00 | 75.00 | 25.00 | 75.00 | 0 | 1 | 0 | | 0 | |
| 3 | 50.00 | 75.00 | 25.00 | 50.00 | 0 | 0 | 0 | | 0 | |
| 4 | 50.00 | 25.00 | 75.00 | 0.00 | 0 | 0 | 0 | | 1 | |
| **SIGNED INTEGER LIMITER block with tunable parameters [Step 2]** | | | | | | | | | | |
| 1 | 0 | 0 | 0 | 0 | - | 0 | 0 | | | 1 |
| 2 | 0 | 75 | 25 | 25 | - | 0 | 1 | | | 0 |
| 3 | 100 | 75 | 25 | 75 | - | 1 | 0 | | | 0 |
| 4 | 50 | 75 | 25 | 50 | - | 0 | 0 | | | 0 |
| 5 | 50 | 25 | 75 | 0 | - | 0 | 0 | | | 1 |
| **FLOATING POINT LIMITER block with configurable parameters [Step 3]** | | | | | | | | | | |
| 1 | 0.00 | 75.00 | 25.00 | 25.00 | 0 | 0 | 1 | | | N/A |
| 2 | 100.00 | 75.00 | 25.00 | 75.00 | 0 | 1 | 0 | | | N/A |
| 3 | 50.00 | 75.00 | 25.00 | 50.00 | 0 | 0 | 0 | | | N/A |
| **SIGNED INTEGER LIMITER block with configurable parameters [Step 4]** | | | | | | | | | | |
| 1 | 0 | 75 | 25 | 25 | - | 0 | 1 | | | N/A |
| 2 | 100 | 75 | 25 | 75 | - | 1 | 0 | | | N/A |
| 3 | 50 | 75 | 25 | 50 | - | 0 | 0 | | | N/A |

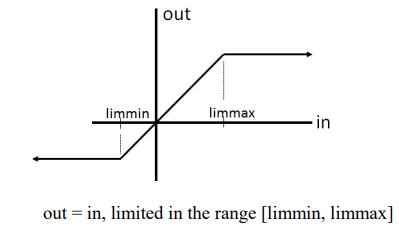


Рисунок 1.9 – Принцип роботи LimiterBlock

В якості вхідних сигналів маємо три аналогових сигнали:

1. #LIM1\_IN – значення, яке буде обчислюватись в середині діапазону [min-max];
2. #LIM1\_MAX\_VAL – сигнал який встановлює максимальну границю для вхідного сигналу «#LIM1\_IN»;
3. #LIM1\_MIN\_VAL – сигнал який встановлює границю мінімуму, якщо значення сигналу #LIM1\_IN нижче min, то результатом буде значення параметра #LIM1\_MIN\_VAL.

Результат обчислення приходить на вихідні сигнали, один аналоговий та чотири дискретних:

1. #LIM1\_OUT– видає результат обмеження вхідного сигнала;
2. #LIM1\_NAN – цей дискретний сигнал, інформує інженера про правильні задані границі обмежувача, тобто MIN<MAX, та про відсутність даних сигналу #LIM1\_IN, простими словами “нічого обмежувати”;
3. #LIM1\_MAX – цей дискретний сигнал вказує на те, що вхідний сигнал #LIM1\_IN вийшов за діапазон максимуму;
4. #LIM1\_MIN – цей дискретний сигнал вказує на те, що вхідний сигнал #LIM1\_IN вийшов за діапазон мінімуму;
5. #LIM1\_PARAM\_ERR – також важливий дискретний сигнал, який інформує про помилки в логіці, наприклад, коли прийдуть параметри MAX=40, а MIN= 100, виходячи з цього MIN>MAX, а це неможливо;

Першим кроком маємо побачити, що вхідний аналоговий сигнал у стані 0, так як MAX = MIN. Виходячи з цього дискретний сигнал з назвою “#LIM1\_PARAM\_ERR” у замкнутому стані (1) (рис. 1.10).

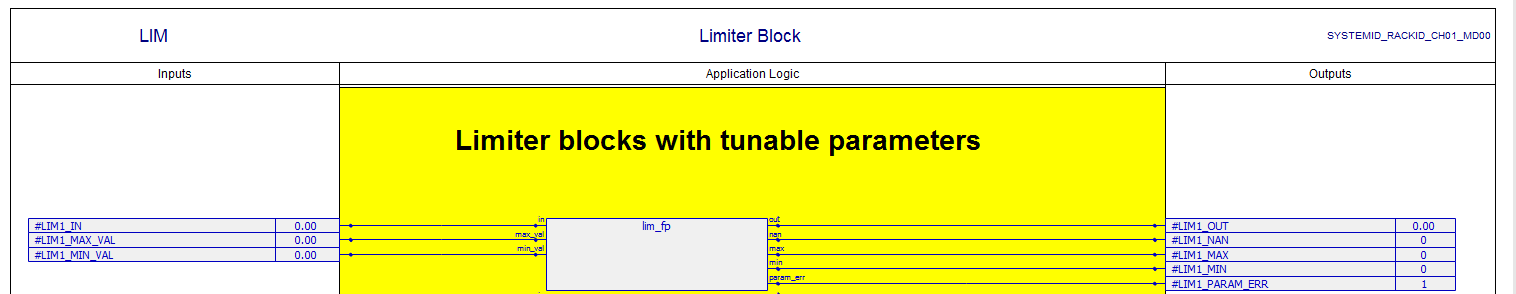


Рисунок 1.10 **–** Перший крок тестування згідно табл.1.1

Наступним кроком задано 100 інженерних одиниць на вхідний сигнал “in”, діапазон від мінімуму до максимуму дорівнює 25…75 EU, тож результат буде 75 EU, так як вхідний сигнал обмежується в діапазоні (рис. 1.11). Та з’явився прапорець #LIM1\_MAX в замкнутому стані, тому що вхідний параметр 100 EU більший ніж 75 EU верхньої границі.

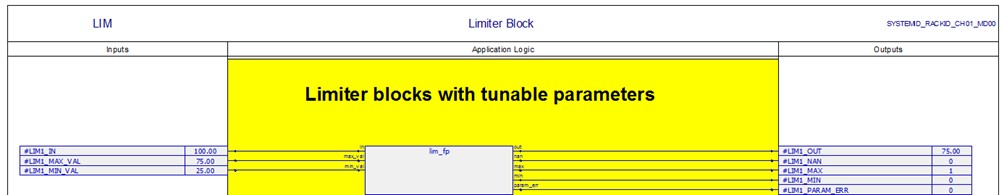


Рисунок 1.11 – Обмеження вхідного сигналу

Це був приклад перевірки двох кроків з ручного тестування AFB блока.

Проблемою є те, що таких компонентів як Lim block для перевірки близько 170 компонентів, які також окремо потрібно протестувати. Кожна перевірка може мати від 4 до 20 кроків.

Після тестування потрібно скласти звіт (Test Report), це також час (табл. 1.2).

Успіхом вважається коли актуальні результати зійшлися з очікуваними.

Таблиця 1.2 – Приклад табличного звіту

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Value** | | | | | | | | | | | | | | **Pass / Fail** |
| **Input** | | | **Output** | | | | | | | | | | |
| **Expected result** | **Actual result** | **Expected result** | **Actual result** | **Expected result** | **Actual result** | **Expected result** | **Actual result** | **Expected result** | | **Actual result** |
| **in** | **max\_val** | **min\_val** | **out** | | **nan** | | **max** | | **min** | | **param\_err** | | |
| **FLOATING POINT LIMITER block with tunable parameters [Step 1]** | | | | | | | | | | | | | | | |
| 1 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | | **Pass** |
| 2 | 100.00 | 75.00 | 25.00 | 75.00 | 75.00 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | | **Pass** |
| 3 | 50.00 | 75.00 | 25.00 | 50.00 | 50.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | **Pass** |
| 4 | 50.00 | 25.00 | 75.00 | 0.00 | 0.00 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | | **Pass** |

# РОЗДІЛ 2

# АВТОМАТИЗАЦІЯ ТЕСТУВАННЯ МАТЕМАТИЧНИХ ТА ЛОГІЧНИХ КОМПОНЕНТІВ, ЩО ОБРОБЛЯЮТЬСЯ В ЛОГІЧНОМУ МОДУЛІ

## **2.1 Вимоги до тестування програмних модулів**

Процес перевірки відповідності програмного або апаратного забезпечення всім вимогам специфікації тесту, які поставлені замовником є процес валідації. Поєднання огляду вихідного коду та модульного тестування гарантує, що програмний блок відповідає вимогам його специфікації, тобто він перевіряє блок.

При виборі відповідних методів та засобів, що відповідають вимогам цього пункту, слід враховувати наступні властивості інтерпретації властивостей у Додатку С IEC 61508-7 стандарту та неофіційні описи методів та засобів у Додатку F IEC 61508-7 для тестування програмних модулів:

* повнота тесту відповідно до специфікації програмного проекту;
* правильність тесту згідно специфікації програмного проекту (успішне виконання);
* відтворюваність;
* чітко визначена конфігурація тесту.
* перевірка правильної інтеграції програмного забезпечення є частиною процесу перевірки.

На етапі проектування та розробки слід розробляти тести на інтеграцію програмного забезпечення.

Переглядаючи інтеграцію програмного забезпечення, слід зазначити наступне:

a) розбиття програмного забезпечення на керовані, інтегровані підмножини;

b) тестові кейси та дані тестів;

c) тип контролю, який слід здійснити;

d) умови тестування, інструменти, конфігурація та використовувані програми;

e) умови, за яких експертиза вважається закінченою;

f) процедури, яких слід дотримуватися, якщо тест негативний.

Програмне забезпечення має бути перевірене на відповідність тестам інтеграції програмного забезпечення, визначеним у специфікації тесту інтеграції програмної системи. Ці тести повинні показати, що всі програмні модулі та програмні елементи / підсистеми працюють правильно разом, щоб виконувати функції, для яких вони призначені, а не виконувати будь-які ненавмисні функції.

Вищезазначене не означає тестування всіх комбінацій вхідних та вихідних даних. Може бути достатньо обстеження всіх класів еквівалентності або структурного обстеження.

Аналіз граничного значення або контрольного потоку може зменшити кількість випробувань до прийнятного рівня. Аналітичне програмне забезпечення може допомогти швидше задовольнити вимоги.

Якщо в розробці використовуються формальні методи, офіційні докази або оператори умов тестування, обсяг таких тестів може бути зменшений.

Цілісність системної безпеки, як правило, не визначається кількісно, кількісні статистичні дані (наприклад, статистичні тести, підвищення надійності) є прийнятними, якщо виконуються усі відповідні умови для статистично достовірних доказів. Для прикладу див. Додаток D до IEC 61508-7.

Результати перевірки інтеграції програмного забезпечення повинні бути задокументовані; документація повинна представляти результати аудиту та вказувати, чи були досягнуті цілі та критерії аудиту. Якщо тест не вдається, слід описати причини.

При інтеграції програмного забезпечення всі зміни повинні бути піддані аналізу впливу, який повинен визначити, на які програмні модулі впливають зміни, та визначити необхідність повторної перевірки та проектування.

Тести інтеграції повинні бути визначені на етапі проектування та розробки та використовуються для перевірки сумісності програмного та апаратного забезпечення в програмованому електронному пристрої, що стосується безпеки.

Для розробки інтеграційних тестів може знадобитися тісна співпраця з дизайнером системи E / E / PE.

Специфікація інтеграційних тестів для програмованої електроніки (апаратне та програмне забезпечення) повинна визначати:

а) поділ системи на рівні інтеграції;

б) тестові кейси та дані тестів;

в) види контролю, що здійснюється;

г) умови випробування, включаючи інструменти, утиліти та описи конфігурації;

д) умови, за яких експертиза вважається закінченою.

Інтеграційні тести для програмованої електроніки (апаратного та програмного забезпечення) повинні розрізняти операції, що виконуються розробником на своєму обладнанні, та операції, що вимагають доступу до пристроїв користувача.

У разі інтеграційних тестів для програмованої електроніки (апаратне та програмне забезпечення) слід розрізняти такі процедури:

а) включення програмного забезпечення в програмований електронний цільовий пристрій;

б) інтеграція систем E / E / PE, тобто додавання інтерфейсів, таких як датчики та виконавчі механізми;

в) повна інтеграція системи E / E / PE, що стосується безпеки.

Програмне забезпечення повинно бути інтегровано до програмної електроніки, що стосується безпеки, відповідно до встановлених тестів інтеграції для програмованої електроніки (апаратне та програмне забезпечення).

Під час тестування інтеграції програмної електроніки, пов’язаної з безпекою (апаратне та програмне забезпечення), всі зміни інтегрованої системи повинні бути піддані аналізу впливу, в ході якого слід визначити, на які програмні модулі впливають зміни та необхідність перевірки.

Тестові випадки та їх результати повинні бути задокументовані для подальшого аналізу.

Результати верифікації інтеграції програмованої електроніки (апаратне та програмне забезпечення) повинні бути задокументовані, результати верифікації повинні бути задокументовані, і має бути зазначено, чи були дотримані цілі та критерії перевірки. Якщо тест не вдається, слід описати причини невдачі.

Всі модифікації або зміни, що виникають в результаті випробування, повинні бути піддані аналізу впливу, в якому слід визначити, на які програмні елементи / модулі впливає зміна, та визначити необхідність повторної перевірки та побудови.

У цьому стандартному програмному забезпеченні (на відміну від апаратного забезпечення) не вимагається технічне обслуговування, воно завжди модифікується.

Зазвичай перевірку програмного забезпечення не можна виконувати ізольовано від апаратного та системного середовища.

Метою вимог цього підрозділу є забезпечення відповідності інтегрованої системи визначеним вимогам до програмного забезпечення системи безпеки для даного рівня цілісності безпеки.

При виборі відповідних методів та засобів, що відповідають вимогам цього підрозділу, слід враховувати наступні характеристики атестації відповідності безпеки системи:

* повнота атестації відповідності згідно специфікації програмного проекту;
* правильність підтвердження відповідності згідно специфікації програмного проекту (успішна реалізація);
* відтворюваність;
* підтвердження відповідності з точно визначеною конфігурацією.

Якщо для програмного забезпечення, пов’язаного з безпекою, відповідність вимогам, вже було встановлено при плануванні підтверджень відповідності безпеки для систем Е / Е / ПЕ, пов’язаних із безпекою, для проведення повторного підтвердження відповідності не вимагається.

Операції примирення повинні проводитися відповідно до специфікацій, розроблених для надання конкретних служб безпеки. Залежно від типу програмного забезпечення, воно відповідає за тип розробки.

Розподіл відповідальності повинен бути задокументований у плануванні безпеки.

Результати перевірки відповідності аспектів програмного забезпечення системи повинні бути задокументовані. Наступні результати повинні бути задокументовані для кожної функції безпеки, коли проводиться перевірка відповідності програмного забезпечення системи безпеки:

* хронологічний перелік процесів підтвердження відповідності, що дає змогу відновити послідовність дій (під час запису результатів тесту важливо, щоб послідовність могла бути відновлена);
* головне відновити порядок дій і не створювати перелік документів, упорядкованих за часом / датою;
* версія використовуваного плану перевірки програмного забезпечення системи;
* функції безпеки, що підлягають підтвердженню (за допомогою тестів або аналізу) із посиланнями на план перевірки аспектів програмного забезпечення системи безпеки;
* використовувані інструменти та обладнання та дані калібрування;
* результати атестаційних справ;
* розбіжності між очікуваними та фактичними результатами.

У разі розбіжностей між очікуваними та фактичними результатами проводиться аналіз і приймається рішення продовжити з підтвердженням відповідності або підготувати запит на зміну та повернутися до більш ранньої фази життєвого циклу розробки. Це рішення слід задокументувати як частину результатів оцінки відповідності аспектів програмного забезпечення системи.

Перевірка аспектів безпеки програмного забезпечення системи безпеки повинна відповідати наступним вимогам.

а) Тестування має бути основним методом перевірки програмного забезпечення. Анімація та моделювання можуть бути використані як додаткові методи.

б) Запуск програмного забезпечення повинен здійснюватися за допомогою моделювання:

* вхідних сигналів під час нормальної роботи;
* очікуваних випадків;
* небажаних станів, що вимагають втручання системи.

в) Постачальник та / або розробник (або кілька сторін, відповідальних за відповідність) повинні надати розробнику системи задокументовані результати оцінки відповідності аспектів безпеки програмного забезпечення системи та всі відповідні документи, щоб він міг відповідати вимогам IEC 61508 – 1 та IEC 61508 –2.

До результатів перевірки аспектів програмного забезпечення системи, що стосуються безпеки, застосовуються наступні вимоги:

а) тести повинні показати, що всі визначені вимоги до програмного забезпечення, пов’язаного з безпекою, правильно виконуються і що програмна система не виконує жодних ненавмисних функцій;

б) тестові випадки та їх результати повинні бути задокументовані для подальшого аналізу та незалежно оцінені відповідно до вимог рівня цілісності безпеки;

в) документовані результати підтвердження відповідності аспектів програмного забезпечення системи повинні містити або заяву про те, що програма отримала підтвердження про відповідність, або причини, чому вона не була.

## **2.2 Розробка нової моделі для тестування AFBL компонентів за допомогою LABVIEW SW ТА TEST STAND для апаратно-програмних комплексів**

Тестування програмного продукту, в якому основні етапи тестування виконуються за допомогою автоматизованих інструментів (запуск, ініціація, реалізація, обробка результатів та висновки) в англійській версії виглядає як «тестування функціональної автоматизації» – автоматизовані тести програмного забезпечення.

Прогрес в розвитку розробки AFBкомпонентів вийшов на новий рівень. На новій фазі з’явилися нові компоненти які потрібно протестувати, а також, повторно перевірити старі для того, щоб бути впевненим,що старі блоки не були змінені.

В ході інтеграційного тестування часу обмаль. Ручна перевірка займає багато часу, і виникає ще одна проблема – це після релізу нової версії потрібно знову виконувати цикл тестувань. Після обміркувань було вирішено автоматизувати систему.

Автоматизоване тестування дозволяє вилучити людський фактор при тестуванні комплексу, це значно скоротить час тестування, а також з’являється можливість складання миттєвого звіту по всім тестам.

Устаткування National Instruments (NI), що використовується в багатьох інших проектах Radics, наприклад, обладнання випробувального стенду, показало надійність і точність вимірювань. Раніше автоматичне тестування всіх проектів Radics проводилося за допомогою LabVIEW SW. Компанія NI має власне спеціальне програмне забезпечення (Test Stand), призначене для розробки автоматизованого виробничого тестування. Тому компанія провела оцінку ПО NI, щоб побачити, чи можна його використовувати для автоматичного тестування AL.

## **2.3 Устаткування конфігурації системи та прошивка логічного модуля**

На рис.2.1 показані SUT (платформа RadICS) і TB (система збору даних NI (DAS)) для автоматичного тестування RPCT AFBL. Вони були розроблені, щоб не використовувати віртуальні сигнали від станції налаштування і програмного забезпечення монітора і замінити їх реальними сигналами з використанням RadICS і NI.платформних HW модулів вводу / виводу.

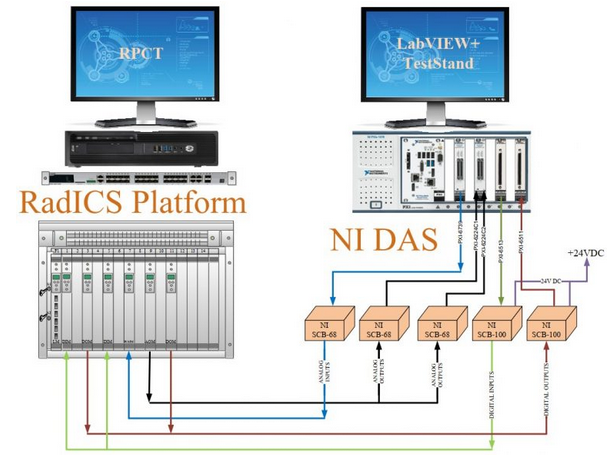
****

Рисунок 2.1–System Under Test and TB for project

Першою головною платою є «Логічний модуль», в системі RPCT прийнято вважати його нульовим. На першому та п’ятому місці в шасі підключений DIM (Discrete Input Module) – модуль дискретних входів, він має 32 канали, які поділені на дві лінії AandB;

На третьому та дев’ятому – DOM (Discrete Output Module) – модуль дискретних виходів, який також має 32 канали.

На сьомому місці підключений WAIM (Wide Analog Input Module) – це модуль вхідних аналогових сигналів, він має 32 канали, які поділені на дві лінії AandB.

Ця плата має змогу працювати в таких режимах:

* режим від 0 до 5 VDC;
* режим від –10 до 10 VDC;

На одинадцятому місці AOM (Analog Output Module) – модуль аналогових виходів. Має 32 канали, які налаштовані в режимі від –10 до 10 VDC.

Зі зворотної сторони шасі маємо спеціальні плати МЗП, які виконують функцію захисту та підключення кабелів типу MOLEX. Ось саме через ці МЗП підключається система NI. Для кожного модуля спеціальний кабель.

Кількість модулів у шасі NI і RadICS невелика (тільки по одному модулю кожного типу), і це було досягнуто за рахунок використання незалежних конфігурацій програмного забезпечення для кожного тестового прикладу. Це означає, що LM може зберігати в своїй пам’яті до 12 конфігурацій шасі з різним набором апаратного забезпечення. Test Stand дозволяв розробнику форматувати HW-модулі і канали по-різному перед кожним тестом.

Використання NI Test Stand IDE дозволило розробнику створювати тестові приклади для кожного AFBL незалежно. Очікуваний результат і вхідні дані для кожного тесту беруться з зовнішнього файлу даних CSV і можуть бути змінені інженером з тестування без практичних навичок LabVIEW (рис. 2.2). Виконання тестового прикладу повністю автоматичне і послідовне.

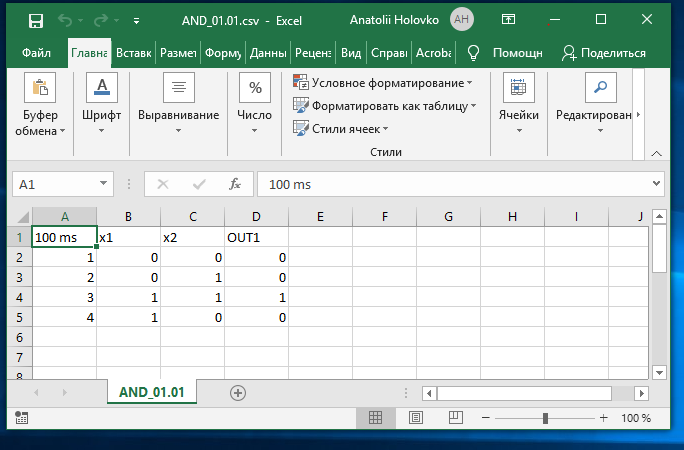
****

Рисунок2.2 – CSV файл даних

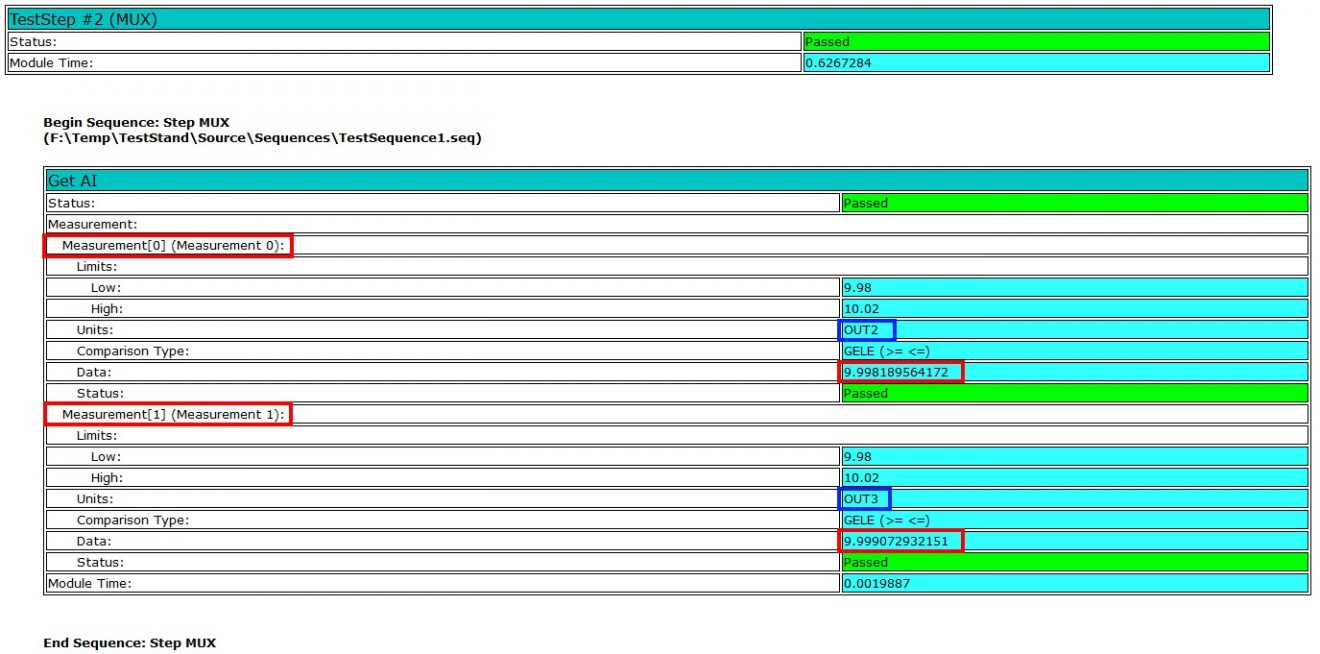


Рисунок2.3 – Автоматично згенерований звіт

Крім того, Test Stand автоматично складає зведений звіт з результатами всіх тестів. Звіт може бути представлений в HTML або в різних форматах даних, таких як XML(рис. 2.3).

Отриманий результат показує, що автоматизований процес за допомогою NI TestStand відповідає всім вимогам тестувальників і менеджерів (керівників).

# РОЗДІЛ 3

# ЕФЕКТИВНІСТЬ ВИКОРИСТАННЯ АВТОМАТИЗАЦІЇ ПРОЦЕСУ ВАЛІДАЦІЇ

## **3.1 Ефективність використання автоматизації при тестуванні AFBLкомпонентів для апаратно-програмного комплексу**

Як і всі вузько спрямовані продукти, автоматизація тестування ПЗ, має свої плюси і мінуси. Відповідно, є випадки, коли автоматичне тестування проводити можна, і варіанти, коли ручний режим більш корисний.

Розглянемо плюси автоматизації.

* Циклічність – гарантія того, що створений автотест завжди буде дотримуватися одного алгоритму перевірок, який не пропустить передбачених кроків при тестуванні.
* Швидкий результат – відпадає необхідність в часі, який потрібно людині для звірки проміжних підсумків, підтвердження безпомилковості при виконанні вимог. Дешевизна – одноразово створений софт для тестування вимагає менше зусиль для аналізу отриманих даних, в результаті, замінюючи ті ж обсяги ручного тестування без втрат в якості.
* Простір в звітності – готові результати легко обробляти, а самі звіти неважко поширити по зацікавленим особам.
* Вільні руки – людина-тестер, під час роботи програми, може виконувати іншу корисну діяльність, наприклад виконувати процедуру візуальної інспекції тільки виготовленого ПП. Дозволено проводити тестування в той час, коли навантаження на числові ресурси знижена (в неробочий час).

Розглянемо мінуси автоматизації.

* + - Циклічність – одноманітні тести не можуть зачепити інші елементи, ніж ті, для яких вони написані. Людина ж здатна помітити дрібні нестиковки на кожному рівні з тестування та зробити висновки про отримані помилки.
* Підтримка – хоч і витрати на ручне тестування більше, автоматичні тести також потрібно оновлювати і доводити, щоб функціональність перевірок відповідала рівню надійної програми (з підвищенням складності перевіряється ПЗ, виникає потреба в оновленні коду Автотест).
* Розробка – написання, а головне – налагодження і тестування Автотест вимагає багато часу. Адже, по суті, софт для тестування програмного забезпечення – не що інше, як те ж саме програмне забезпечення. Тільки функціонал дуже вузький.
* Вартість – ліцензійний примірник фреймворка для автоматизації може обійтися в пристойну суму. І хоч безкоштовні варіанти теж, як правило широко використовуються, їх функціональність часто залишає бажати багато кращого, а ліцензія повинна допомогти при виникненні проблеми.
* Дрібні помилки – автотести можуть не помічати дрібних дефектів, що не завдають шкоди функціональності коду, але псують візуальний інтерфейс і ускладнюють роботу кінцевого користувача.

Найпоширенішим методом оцінки ефективності (насамперед рентабельності) є підрахунок рентабельності інвестицій (ROI).

Він розраховується досить просто, оскільки це відношення прибутку до собівартості. Як тільки рентабельність інвестицій перевищує одиницю, то вкладені кошти повертаються та починають залучатись нові.

У випадку автоматизації прибуток означає економію на ручному тестуванні.

Крім того, виграш у цьому випадку може бути не очевидним – наприклад, результати усунення недоліків у процесі спеціального тестування інженерами, автоматизація яких звільнила час. Такий прибуток досить складно розрахувати, тому можна або зробити припущення (наприклад + 10%), або залишити його поза увагою.

Інший приклад – виключення «людського фактора» із процесу тестування системи. Це важливо, коли точність та правильність операцій є критично важливими для бізнесу. Вартість такої помилки може бути значно вищою, ніж витрати на розробку та підтримку тестування.

## **3.2 Причини вимірювання ефективності**

Вимірювання ефективності допомагає відповісти на запитання: «Чи варто застосовувати автоматизацію в проекті?», «Коли впровадження приносить нам значущий результат? Замінити трьох інженерів-ручників на одного автоматизованого інженера? тощо.

Ці розрахунки можуть допомогти сформулювати цілі (або показники) для команди автоматизації випробувань. Економія X годин на місяць ручного тестування, зменшення вартості випробувальної групи на Y умовних одиниць.

Для того, щоб генерувати дані для оцінки, необхідно визначити автоматизоване охоплення тестом. Щоб зробити його помітним, є прості правила, якими треба скористатися:

* функціональність, що охоплюється, повинна бути досить стабільною (чим стабільніша функціональність, тим менша вартість підтримки тестів);
* охоплювана функціональність повинна перевірятися регулярно і часто (чим частіше тестується функціональність, тим швидше окупається її автоматизація; нечасто і нерегулярно перевірені функціональні можливості не ефективні для автоматизації – вигідніше тестувати вручну);
* можливість тестування за допомогою автотестів.

Розрахунок ефективності автоматизації в проекті вимагає ряду вхідних даних, які легко отримати:

* вартість (у людських годинах) ручного тестування в межах вибраного покриття;
* вартість розробки автотестів (у людських годинах) у межах цього покриття;
* дані про частоту тестів, проведення оцінки.

Інтервал аналізу, як правило, становить один місяць. Однак цілком можливо адаптувати його під конкретний проект – наприклад, спринт або тижневий інтервал.

Спочатку розраховується вартість ручного тестування для кожного інтервалу. при цьому вартість буде зменшена (замінена автотестами).

Потім розраховується вартість розробки тестів.

Необхідно врахувати передбачуваний розмір команди автоматизації в проекті – це показує, як швидко можна впровадити автотести та замінити ручні тести. Потім обчислюється суму цих витрат. Фактично це загальна вартість випробувань. Для порівняння додаються витрати на ручне тестування (без автоматизації). Далі обчислюються нараховані витрати на тестування з автоматизацією та без неї.

Різниця між цими величинами показує ефективність процесу автоматизації (економія трудових витрат) протягом певного періоду часу. Очевидно, що автоматизація цих сценаріїв має значний економічний ефект, який відбувається дуже швидко і впевнено збільшується з плином часу.

Переваги автоматизованого функціонального тестування компонентів AFBL:

* значно скорочено час тестування;
* вхідні та вихідні сигнали пов’язані з фізичними каналами модулів вводу-виводу;
* показники тестів не залежать від людського фактора;
* зміни в комбінаціях тестів не вимагають знань у середовищі LabVIEW;
* можливість створення автоматичного звіту для кожного тесту.

Недоліки автоматизованого функціонального тестування компонентів AFBL:

* неможливо повністю автоматизувати тестування всіх вимог до RPCT.
* налагодження / розширення тестових кейсів вимагає залучення фахівців EQ Department;
* автоматично створені звіти поки що не можуть використовуватися в офіційних документах тестування.

Таблиця 3.1**–** Таблиця порівняння ефективності ручного з автоматизованим методом

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Функціональне тестування RPCT в ручному режимі** | | | | |
| **Тип** | | **Кількість тестів** | **Час, необхідний для виконання, людино-година (без документації)** | |
| Функціональні тести, що виконуються в ручному режимі | | 106 | 120 | |
| **Функціональне тестування RPCT з урахуванням автоматизації** | | | | |
| Функціональні тести, що виконуються в ручному режимі | 57 | | | 64 |
| Функціональні тести, що виконуються в автоматизованому режимі. | 49 | | | 0,5 |
| *t*/*T* = 120 / 64.5 = 1,86  *t* –час, витрачений на функціональне тестування з автоматизацією, *T* –час, витрачений на функціональне тестування без автоматизації. | | | | |

## **3.2 Вплив на ефективність автоматизації.**

Перший фактор,що впливає на ефективність автоматизації є частота тестового запуску.

Скажімо, тест проводиться щодня і вимагає 1 години роботи. На розробку цього тесту потрібно 4 години. Очевидно, що вже на п’ятий день авто тестів ведення є більш ефективними, ніж ручне тестування (його заміна). Що робити, якщо треба робити тести двічі на день? Потім вони набувають чинності на третій день.Варто зазначити, що тут є пастка – введення тестів на «потребу», а не на «бажання». Тим не менш, запуски повинні принести значні результати.

Другим фактором вважається «Data-Driven тестування».

Часто один і той же тест повинен виконуватися з різним набором входів або в різних середовищах. У цьому випадку автоматизація тестів розквітає у всій красі завдяки підходу, керованому даними – просто потрібно розробити тест, який приймає різні дані як вхідні дані.

Наприклад, крос-браузерне тестування або перевірка створення квитків з різними даними.

Досвід тестувальника – фактор, який має сильний вплив на витрати на розробку та суттєво знижує ефективність.

Коли люди без достатнього досвіду у створенні автотестів починають автоматизувати тестування проекту, це збільшує витрати на розробку тесту, підтримку його, аналіз запусків тощо.

Вартість інструментарію. На ринку існує велика кількість засобів платної автоматизації тестування. Багато з них значно перевершують своїх безкоштовних аналогів, дозволяючи швидше запускати більш стабільні тести, розробляти та налагоджувати їх. За збігом обставин для даного інтерфейсу автоматизації (наприклад, Adobe Air) існує майже єдиний інструмент автоматизації, вартість якого значна.

У цьому випадку вартість придбання ліцензій повинна бути пов’язана із загальною вартістю тестування.

Аналіз та підтримка. Під час оцінки ефективності часто залишаються поза увагою такі важливі речі, як аналіз та підтримка тестових запусків. Якщо є велика кількість автотестів, ці заходи можуть зайняти багато часу.

Витрати на підготовку тестових наборів. Інженери з автоматизації тестування часто отримують тестові кейси від команди ручних тестів як вихідні дані, які рідко підходять для повної автоматизації.

# РОЗДІЛ 4

# ФУНКЦІОНАЛЬНА БЕЗПЕКА СИСТЕМ ЕЛЕКТРИЧНИХ, ЕЛЕКТРОННИХ, ПРОГРАМОВАНИХ ЕЛЕКТРОННИХ, ПОВ’ЯЗАНИХ З БЕЗПЕКОЮ

## **4.1 Опис функціональної безпеки систем електричних, електронних, програмованих електронних, пов’язаних з безпекою**

Системи, що складаються з електричних і / або електронних елементів, протягом багатьох років використовуються для виконання функцій безпеки в більшості областей застосування. Комп’ютерні системи (зазвичай звані програмованими електронними системами), що застосовуються у всіх прикладних галузях для виконання функцій, не пов’язаних з безпекою, у все більш зростаючих обсягах використовуються для виконання функцій забезпечення безпеки.Для безпечного та ефективного використання технологій, заснованих на використанні комп’ютерних систем, надзвичайно важливо, щоб особи, відповідальні за прийняття рішень, мали в своєму розпорядженні керівництва з питань безпеки, які вони могли б використовувати в своїй роботі.

Цей стандарт встановлює загальний підхід до питань забезпечення безпеки для всіх стадій життєвого циклу систем, що складаються з електричних і / або електронних, і / або програмованих електронних (Е / Е / ПЕ) елементів, які використовуються для виконання функцій забезпечення безпеки. Цей уніфікований підхід був прийнятий для того, щоб розробити раціональну і послідовну політику для всіх електричних систем забезпечення безпеки. Основною метою при цьому є сприяння розробці стандартів для продукції і областей застосування на основі стандартів серії МЕК 61508.

Прикладами стандартів для продукції і областей застосування, розроблених на основі стандартів серії МЕК 61508, є [1-3].

У більшості ситуацій безпека досягається за рахунок використання декількох систем, в яких використовуються різні технології (наприклад, механічні, гідравлічні, пневматичні, електричні, електронні, програмовані електронні). Будь-яка стратегія безпеки повинна, отже, враховувати не тільки всі елементи, що входять до складу окремих систем (наприклад, датчики, вузли, виконавчі механізми), але також і всі підсистеми безпеки, що входять до складу загальної системи забезпечення безпеки. Таким чином, хоча цей стандарт присвячений в основному Е / Е / ПЕ системам, пов’язаним з безпекою, він може також надавати загальний підхід, в рамках якого розглядаються системи, пов’язані з безпекою, що базуються на інших технологіях. Визнаним фактом є існування великої різноманітності використання Е / Е / ПЕ систем в різних областях застосувань, що відрізняються різним ступенем складності, можливими небезпеками і ризиками. У кожному конкретному застосуванні необхідних заходів безпеки будуть залежати від численних факторів, які є специфічними для цього застосування. Цей стандарт, будучи базовим стандартом, дозволить формулювати такі заходи в майбутніх міжнародних стандартах для продукції і областей застосування, а також в наступних редакціях вже існуючих стандартів.

Цей стандарт:

* розглядає всі відповідні стадії життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки і програмного забезпечення системи безпеки (від початкової концепції, проектування, реалізації, експлуатації, технічного обслуговування і до зняття з експлуатації), в ході яких Е / Е / ПЕ системи використовуються для виконання функцій безпеки;
* був задуманий з урахуванням швидкого розвитку технологій; його основа є в значній мірі стійкою і повною для застосування під час майбутніх розробок;
* дозволяє розробку стандартів для продукції і областей застосування, де використовуються Е / Е / ПЕ системи, пов’язані з безпекою; розробка стандартів для продукції і областей застосування в рамках загальної структури, введеної справжнім стандартом, повинна привести до більш високого рівня узгодженості (наприклад, основних принципів, термінології і т.д.) як для окремих областей застосування, так і для їх сукупностей, що дасть переваги в плані безпеки і економіки;
* надає метод розробки специфікації вимог до системи безпеки, необхідних для досягнення заданої функціональної безпеки Е / Е / ПЕ систем, пов’язаних з безпекою;
* використовує для визначення вимог до рівнів повноти безпеки підхід, заснований на оцінці ризиків;
* вводить рівні повноти безпеки при завданні цільового рівня повноти безпеки для функцій безпеки, які повинні бути реалізовані Е / Е / ПЕ системами, пов’язаними з безпекою.

Цей стандарт не встановлює вимоги до рівня повноти безпеки для будь-якої функції безпеки і не визначає, як встановлюється рівень повноти безпеки. Однак цей стандарт ґрунтується на оцінці ризиків, на концептуальному підході і пропонує приклади методів;

* встановлює цільові заходи відмов для функцій безпеки, що реалізуються Е / Е / ПЕ системами, пов’язаними з безпекою, і пов’язує ці заходи з рівнями повноти безпеки;
* встановлює нижню межу для цільових заходів функції безпеки, що реалізується одиночній Е / Е / ПЕ системою, пов’язаної з безпекою. Для Е / Е / ПЕ систем, пов’язаних з безпекою в режимі низької інтенсивності запитів на обслуговування;
* високої інтенсивності запитів на обслуговування або в безперервному режимі: нижня межа встановлюється відповідно до середньої частотою небезпечних відмов 10 в годину.

1 Одиночна Е / Е / ПЕ система, пов’язана з безпекою, не обов’язково передбачає одноканальну архітектуру.

2 В проектах систем, пов’язаних з безпекою можна досягти більш низьких значень цільової повноти безпеки, але передбачається, що в даний час зазначені граничні значення цільової повноти безпеки можуть бути досягнуті для складних систем (наприклад, програмовані електронні системи, пов’язані з безпекою):

* встановлює вимоги щодо запобігання і управління систематичними відмовами, засновані на досвіді і висновках з практичного досвіду. З огляду на те, що ймовірність виникнення систематичних відмов в загальному випадку не може бути визначена кількісно, це дозволяє стверджувати що для функції безпеки, міри відмов пов’язаних з цією функцією, може вважатися досягнутою, якщо всі вимоги стандарту були виконані;
* вводить поняття стійкості до систематичних відмов, яка застосовується до елементу, що характеризує впевненість в тому, що повнота безпеки, що стосується систематичних відмов елемента, задовольняє вимогам заданого рівня повноти безпеки;
* застосовує широкий діапазон принципів, методів і засобів для досягнення функціональної безпеки Е / Е / ПЕ систем, пов’язаних з безпекою, але не використовує явно поняття «безпечної відмови». У той же час поняття «безпечна відмова» і «безпечний в своїй основі» можуть бути використані, але для цього необхідно забезпечити відповідні вимоги у відповідних розділах стандарту, яким ці поняття повинні задовольняти.

## **4.2 Область застосування**

Цей стандарт охоплює питання, які повинні враховуватися при використанні електричних, електронних, програмованих електронних (Е / Е / ПЕ) систем для виконання функцій безпеки.

Головною метою цього стандарту є полегшити розробку стандартів для продукції і областей застосування технічними комітетами, відповідальними за цю продукцію і область застосування.

Це дозволить повністю врахувати істотні фактори, пов’язані з виробом або областю застосування, і, таким чином, задовольнити конкретні потреби області застосування і споживачів вироби. Інша мета цього стандарту полягає в тому, щоб зробити можливою розробку Е / Е / ПЕ систем, пов’язаних з безпекою, в умовах можливого відсутності стандартів для виробів і областей застосування.

Зокрема, цей стандарт:

* застосовується до систем, пов’язаних з безпекою, коли одна або кілька таких систем включають в себе електричні, електронні, програмовані електронні елементи.Для Е / Е / ПЕ систем, пов’язаних з безпекою,що мають низьку складність, деякі вимоги, визначені в цьому стандарті, можуть виявитися необов’язковими, і стає можливим звільнення від відповідності таким вимогам (див. 4.2, а також визначення Е / Е / ПЕ систем, пов’язаних з безпекою і мають низьку складність, в МЕК 61508-4, пункт 3.4.4).Хоча людина може бути частиною системи, пов’язаної з безпекою (МЕК 61508-4, пункт 3.4.1), вимоги до людського фактору, що відносяться до проектування Е / Е / ПЕ систем, пов’язаних з безпекою, не розглядаються детально в цьому стандарті;
* є основним і застосовується до всіх Е / Е / ПЕ системам, пов’язаним з безпекою, незалежно від їх застосування;
* охоплює досягнення допустимого ризику за допомогою застосування Е / Е / ПЕ систем, пов’язаних з безпекою, але не поширюється на небезпеку, джерелом яких є саме Е / Е / ПЕ обладнання (наприклад, ураження електричним струмом);
* застосовується до всіх типів Е / Е / ПЕ систем, пов’язаних з безпекою, включаючи системи захисту і системи контролю;
* не охоплюють Е / Е / ПЕ системи, в яких:однієї Е / Е / ПЕ системи досить для досягнення допустимого ризику; необхідна повнота безпеки функцій безпеки однієї Е / Е / ПЕ системи менше задається для рівняповноти безпеки, рівної 1 (найнижчий рівень повноти безпеки в цьому стандарті);
* відноситься, головним чином, до Е / Е / ПЕ систем, пов’язаних з безпекою, відмови яких можуть впливати на безпеку людей і / або на навколишнє середовище; однак визнано, що наслідки відмови можуть також викликати серйозні економічні наслідки, і в таких випадках цей Стандарт може бути використаний для визначення будь-якої Е / Е / ПЕ системи, що використовується для захисту обладнання або продукції(МЕК 61508-4, пункт 3.1.1);
* розглядає Е / Е / ПЕ системи, пов’язані з безпекою, і інші заходи зниження ризику для того, щоб специфікації вимог безпеки Е / Е / ПЕ систем, пов’язаних з безпекою, могли бути визначені на основі систематичного аналізу ризиків;
* використовує модель життєвого циклу всієї системи безпеки як технічну основу для систематичних дій, необхідних для забезпечення функціональної безпеки Е / Е / ПЕ систем, пов’язаних з безпекою.Хоча життєвий цикл всієї системи безпеки відноситься в першу чергу до Е / Е / ПЕ систем, пов’язаних з безпекою, він може також лежати в основі аналізу будь-якої системи, пов’язаної з безпекою, незалежно від технології, на якій вона заснована (наприклад, механічної, гідравлічної або пневматичної);
* не визначає рівнів повноти безпеки для областей застосування (які повинні ґрунтуватися на докладній інформації та знаннях, що відносяться до сфери застосування). Технічні комітети, які відповідають за конкретні галузі застосування, повинні визначати, де це необхідно, рівні повноти безпеки в стандартах області застосування;
* встановлює загальні вимоги до Е / Е / ПЕ систем, пов’язаних з безпекою, де відсутні стандарти на продукцію або області застосування;
* вимагає розгляду зловмисних і непередбачених дій під час аналізу відмов та ризиків. Сфера аналізу включає в себе всі стадії життєвого циклу системи безпеки. Інші стандарти МЕК / ІСО більш глибоко розглядають дане питання, ISO / IEC / TR 19791 [4] і серію МЕК 62443 [5];
* не охоплюють запобіжні заходи, які необхідні для того, щоб запобігти пошкодженням чи іншим несприятливим впливам на функціональну безпеку Е / Е / ПЕ систем, пов’язаних з безпекою, з боку осіб, які не мають повноважень.
* не визначає вимоги до розробки, впровадження, обслуговування і / або експлуатації політики безпеки або служб безпеки, необхідних для виконання політики безпеки, які можуть знадобитися для Е / Е / ПЕ систем, пов’язаних з безпекою;
* не застосовується до медичного обладнання, що задовольняє вимогам серії МЕК 60601 [6].

Цей стандарт встановлює загальні вимоги, які застосовні до всіх частин стандарту. В інших частинах розглядаються більш конкретні питання:

* в МЕК 61508-2 і МЕК 61508-3 надані додаткові і конкретні вимоги до Е / Е / ПЕ систем, пов’язаних з безпекою (вимоги до апаратних засобів і програмного забезпечення);
* МЕК 61508-4 містить визначення термінів і скорочення, які використовуються в цьому стандарті;
* [7] містить керівництво щодо застосування МЕК 61508-1 для визначення рівнів повноти безпеки на основі використання різних методів;
* [8] містить керівництво щодо застосування МЕК 61508-2 і МЕК 61508-3;
* [9] містить огляд методів і засобів.

МЕК 61508-1 - МЕК 61508-4 являються базовими стандартами з безпеки, хоча цей статус не можна застосувати в контексті Е / Е / ПЕ систем, пов’язаних із безпекою, мають низьку складність (МЕК 61508-4, пункт 3.4.3).

В якості базових стандартів з безпеки вони призначені для використання технічними комітетами при підготовці стандартів відповідно до принципів, викладених в керівництві МЕК 104 і керівництві ISO / IEC 51, МЕК 61508-1, МЕК 61508-2, МЕК 61508-3 і МЕК 61508- 4 призначені для використання в якості самостійних стандартів.

Функція безпеки справжнього стандарту не може бути застосована до медичного обладнання, яка відповідає вимогам серії горизонтальних стандартів МЕК 60601 [6].

У коло обов’язків технічного комітету входить використання, де це можливо, основоположних стандартів з безпеки при підготовці власних стандартів. У цьому випадку вимоги, методи перевірки або умови перевірки справжнього основоположного стандарту з безпеки не застосовуватимуться, якщо на них немає конкретного посилання або вони не включені в стандарти, підготовлені цими технічними комітетами.

## **4.3 Оцінка функціональної безпеки**

Метою вимог цього розділу є визначення дій, необхідних для вивчення і винесення рішення щодо адекватності функціональної безпеки, досягнутої Е / Е / ПЕ системою (ами), пов’язаної (ими) з безпекою, або застосовуваними виробами (наприклад, елементами або підсистемами) на основі дотримання відповідних положень цього стандарту.

Для виконання однієї або більше оцінок функціональної безпеки необхідно призначити одну або більше осіб, щоб прийти до рішення про адекватність:

* функціональної безпеки, що досягається Е / Е / ПЕ системою, пов’язаною з безпекою в навколишньому середовищі, з відповідними положеннями цього стандарту;
* виконання відповідних положень цього стандарту елементами або підсистемами.

Ті, хто виконують оцінку функціональної безпеки, повинні мати доступ до всіх осіб, які виконують будь-які дії на всіх стадіях життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки або програмного забезпечення системи безпеки, а також до всієї інформації та обладнання (включаючи апаратні засоби і програмне забезпечення).

Слід визнати, що особи, які раніше брали участь в роботах на різних стадіях життєвого циклу системи безпеки, не завжди доступні, тому відповідальність повинна бути покладена на осіб, які в даний час мають відповідні функції.

Оцінку функціональної безпеки слід застосовувати до всіх стадій протягом життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки і програмного забезпечення системи безпеки, включаючи документацію, верифікацію і управління функціональної безпекою.

Особи, які здійснюють оцінку функціональної безпеки, повинні розглянути всі їх дії, а також всі результати, отримані протягом кожної стадії життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки і програмного забезпечення системи безпеки, і дати висновок про те , в якій мірі виконані цілі і вимоги цього стандарту.

Всі відповідні заяви про відповідність, надані постачальниками та іншими сторонами, відповідальними за досягнення функціональної безпеки, повинні бути включені в оцінку функціональної безпеки.

Такі заяви можуть бути зроблені для діючої системи або для вкладу дій і / або обладнання в функціональну безпеку на кожній стадії життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки і програмного забезпечення системи безпеки.

Оцінка функціональної безпеки може виконуватися після кожної стадії життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки і програмного забезпечення системи безпеки або після декількох стадій за умови виконання основної вимоги: оцінка функціональної безпеки повинна здійснюватися до виникнення виявлених небезпек.

Оцінка функціональної безпеки повинна включати в себе оцінку доказів того, що аудит функціональної безпеки було проведено (повністю або частково) відповідно до його областю застосування.

При оцінці функціональної безпеки необхідно враховувати як мінімум наступне:

* роботи, виконані з часу попередньої оцінки функціональної безпеки;
* плани або стратегію реалізації подальших оцінок функціональної безпеки для життєвих циклів всієї системи безпеки, Е / Е / ПЕ системи безпеки і програмного забезпечення системи безпеки;
* рекомендації попередніх оцінок функціональної безпеки і обсяг внесених змін.

Кожна оцінка функціональної безпеки повинна бути спланована. План повинен визначати всю інформацію, необхідну для проведення ефективної оцінки, включаючи:

* область застосування оцінки функціональної безпеки;
* залучені організації;
* необхідні ресурси;
* осіб, які здійснюють оцінку функціональної безпеки;
* рівень незалежності осіб, які виконують оцінку функціональної безпеки;
* компетентність всіх осіб, які виконують оцінку функціональної безпеки;
* вихідні матеріали при кожній оцінці функціональної безпеки;
* як оцінка функціональної безпеки співвідноситься і повинна бути інтегрована з іншими оцінками функціональної безпеки у відповідних випадках.

1 При встановленні області застосування оцінки функціональної безпеки необхідно визначити документи, які використовуються в якості вхідних матеріалів для кожної дії, пов’язаної з оцінкою функціональної безпеки, і статус цих документів.

2 План може бути сформований або відповідальними за оцінку функціональної безпеки, або відповідальними за управління функціональної безпекою, або його формування може бути розділене між ними.

Перед виконанням оцінки функціональної безпеки її план повинен бути затверджений тими, хто буде виконувати цю оцінку, і тими, хто несе відповідальність за управління функціональної безпекою.

У висновку про оцінку функціональної безпеки особи, які виконують оцінку, повинні документально оформити відповідно до планів оцінки та колом повноважень:

* виконані дії;
* отримані результати;
* висновки;
* судження про адекватність функціональної безпеки вимогам цих технічних умов;
* рекомендації, що випливають з оцінки, в т.ч. рекомендації щодо прийняття, умовного прийняття чи відхилення.

Відповідальним за будь-які дії на життєвих циклах всієї системи безпеки, Е / Е / ПЕ системи безпеки або програмного забезпечення системи безпеки, включаючи конструкторів і експертів по Е / Е / ПЕ системам, пов’язаним з безпекою, повинні бути доступні відповідні результати оцінки функціональної безпеки застосовуваних виробів.

Результати оцінки Е / Е / ПЕ системи, пов’язаної з безпекою, повинні бути доступні для інтегратора Е / Е / ПЕ системи.

Результат оцінки функціональної безпеки виробу повинен включати наступну інформацію для полегшення повторного використання результатів оцінки для більшої системи (див. МЕК 61508-2, додаток D, МЕК 61508-3, додаток D і МЕК 61508-4, пункт 3.8. 17):

a) точне визначення застосовуваного вироби, включаючи версії апаратного і програмного забезпечення.

Якщо виріб розглядається як частина більш великої системи або як сімейство обладнання, то точне визначення цієї системи або сімейства обладнання повинно бути документально оформлено;

b) умови, передбачувані в ході оцінки (наприклад, умови використання Е / Е / ПЕ системи, пов’язаної з безпекою);

c) посилання на документально оформлений доказ, на якому заснована заключна оцінка;

d) процедури, методи та інструменти, використовувані для оцінки стійкості до систематичних відмов, разом з обґрунтуванням їх ефективності;

e) процедури, методи та інструменти, використовувані для оцінки повноти безпеки апаратного забезпечення разом з обґрунтуванням використовуваного підходу і якості даних (наприклад, інтенсивність відмов або розподіл джерел даних);

f) оцінку результатів, отриманих відповідно до вимог цього стандарту і специфікації характеристик системи безпеки для застосовуваного вироби у відповідному керівництві з безпеки;

g) прийняті відхилення від вимог МЕК 61508 з відповідними роз’ясненнями і / або посиланнями на докази, що містяться в документації.

Особи, які здійснюють оцінку функціональної безпеки, повинні бути компетентними в виконуваних діях, відповідно до вимог 6.2.13-6.2.15.

1 Залежно від організаційної структури компанії і досвіду всередині компанії вимоги по незалежності осіб і підрозділів можуть бути виконані шляхом використання послуг сторонньої організації. У свою чергу компанії, які мають внутрішні структури з досвідом в оцінці ризиків і застосуванні систем, пов’язаних з безпекою, та які не залежать і відокремлені (по управлінню і використовуваних ресурсів) від тих, які несуть відповідальність за основну розробку, можуть виявитися здатними використовувати свої власні ресурси, щоб задовольнити вимогам по незалежності організації.

2 Див. Пункти 3.8.11, 3.8.12 і 3.8.13 МЕК 61508-4 для визначення термінів «незалежна особа», «незалежний підрозділ» і «незалежна організація» відповідно.

3 Особи, які здійснюють оцінку функціональної безпеки, повинні бути обережні в наданні консультацій з будь-якого питання, пов’язаного з оцінкою, оскільки це може поставити під загрозу їхню незалежність. Найчастіше прийнято давати поради з різних аспектів, що може спричинити за собою рішення про недостатність безпеки, таке як недостатність доказів, але зазвичай не прийнято давати поради або рекомендації для конкретних засобів захисту від тих чи інших проблем.

# ВИСНОВКИ

Мета автоматизації тестування математичних та логічних компонентів – зменшення часу на тестування, обмеження дій з боку інженера, який проводить тест, з головною умовою – зберегти та покращити рівень надійності.

Автоматизація коштує дорого: витрати на датчики, комп’ютери, програмне забезпечення та зусилля програміста. Але врешті-решт може виникнути чудова нова здатність, інженер звільняється від роботи по реєстрації та інтерпретації даних.

Оператору більше не потрібно координувати так багато важливих налаштувань. Підвищується якість продукції і даних.

Отже на виході маємо автоматизоване програмно апаратне середовище, яке спроможне автоматично перевірити більше 100 AFBLкомпонентів, які виконують математичні та логічні функції платформи Radics, в модулі LM. За короткий проміжок часу (не більше 10 хв.).

Завдяки цій ідеї вдалося протестувати нові версії компонентів та новий AFBLblock. Автоматично складений звіт дав змогу не витрачати час на фіксацію вхідних та вихідних даних.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. IEEE 829-2008. Standard for Software Test Documentation.
2. IEEE 1012-2012. Standard for Software Veriﬁcation and Validation.
3. IEEE 1016-2009. IEEE Standard for Information technology – Systems Design – Software Design Descriptions.
4. IEEE 1028-2008. IEEE Standard for Software Reviews and Audits.
5. IEEE 1076-2008.VHDL Language Reference Manual.
6. IEC 61508-2010. Functional safety of electrical / electronic/ programmable electronic safety-related systems – Part 3: Software requirements.
7. IEC 61508-2010. Functional safety of electrical/ electronic/ programmable electronicsafety-relatedsystems – Part 7: Over view of techniques and measure
8. ISO/ІЕС 12119. IT – Software packages – Quality requirements and testing, 1994.29 p.
9. ISO/IEC 9126-1. Software engineering – Product quality – Part 1: Quality model, 2001.26 p.
10. А. С. Алпєєв Верифікація та валідація програмованих керуючих систем АЕС.<http://www.sstc.com.ua/documents/journal/2010/3/5_3_2010_text_ru.pdf>
11. Береза А.М. Основи створення інформаційних систем: Навч. посібник. 2 видання,перероблене і доповнене – К.: КНЕУ, 2001.
12. Вовк О.Б. Аналіз та оцінювання якості програмного продукту (поняття, терміни, означення).<http://ena.lp.edu.ua:8080/bitstream/ntb/8885/1/06.pdf>
13. Георгіаді Н.Г. Моніторинг стану інформаційної системи управління економічним розвитком підприємства. <http://vlp.com.ua/files/21_30.pdf>
14. Гушко С.В., Шайкан А.В. Управлінські інформаційні системи: Навч. посібник. – Львів: Магнолія плюс, 2006. – 320 с.
15. Марченко Е. Что такое качество програмного обеспечения. http://softwaretesting.ru/library/testing/general-testing
16. Карпенко М. Ю. Технології створення програмних продуктів та інформаційних систем : навч. посібник / М. Ю. Карпенко, Н. О. Манакова, І. О. Гавриленко ; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків: ХНУМГ ім. О. М. Бекетова, 2017. – 93 с.
17. Б.М. Конорев, В.В. Сергієнко, І.Б. Туркін Доказова незалежна верифікація та прогнозування прихованих дефектів критичного програмного забезпечення на базі диверсного вимірювання інваріантів // Інженерія програмного забезпечення №1(5) 2011
18. Контролювання та регулювання економічного розвитку підприємства: проблеми, методологічні та прикладні аспекти: Монографія / О.Є.Кузьмін, С.В.Князь, Н.О.Шпак, В.А. Новицький. – Львів: В-во НУЛП, 2006. – 148 с.
19. Коцовський В.М. Супровід програмних систем: Методичний посібник для студентів спеціальності "Інженерія програмного забезпечення" / В. М. Коцовський. – Ужгород: Видавництво УжНУ "Говерла", 2016. — 52 с.
20. Орлик С. Введение в программную инженерию и управление жизненным циклом программного обеспечения. Программная инженерия. Сопровождение программного обеспечения. <http://sorlik.blogspot.com>.
21. Куліков С.С. Тестування програмного забезпечення. Базовий курс: Мінськ: 2017. – 312 с. ISBN 978-985-581-125-2.
22. Райчев І.Е. Проблеми сертифікації програмного забезпечення автоматизованих систем контролю // Вісник НАУ. -2004. -№1. -С. 23-28.
23. Райчев І.Е., Харченко О.Г. Концепція побудови сертифікаційної моделі якості програмних систем // Проблемы программирования. -2006. -№2-3. – С. 275-281.
24. Райчев И.Э., Харченко А.Г., Яцков Н.А. Исследованиеметодовтестированияпрограммныхмодулейобработкиполетнойинформации // Вісник КМУЦА. – 2000. – №1-2. – С.127-133.
25. Райчев И.Э., Харченко А.Г., Яцков Н.А. Методы создания тестових наборов данных при сертификационных испытаниях комплексов программ контроля полетов // Вісник НАУ. – 2001. – №1. – С. 126-132.
26. Ястребенецький М.О., Розен Ю.В., Виноградська С.В. та ін. Безпека атомних станцій. Система керування й захисту ядерних реакторів. Російською мовою. – К: ТОВ «Основа Принт», 2011. – С. 32-128.
27. Платформа RadICS [Електронний ресурс] – Режим доступу до ресурсу: <http://www.radiy.com/ru/>.
28. Программно-технический комплекс автоматического регулирования, разгрузки и ограниченной мощности реактора и ускоренной предупредительной защиты [Електронний ресурс] – Режим доступу до ресурсу: <http://www.radiy.com/ru/produktsiya-dlya-aes/produktsiya/sistemy-bezopasnosti-na-baze-fpga-dlya-aes/ptk-arm-rom-upz.html>.