Vol. 2, No. 1, 2020

https://doi.org/10.23939/csn2020.01.068

УДК 004.382, 004.384

# ПРИНЦИПИ ПОБУДОВИ ПРОГРАМНИХ ЗАСОБІВ СИСТЕМИ НАЛАШТУВАННЯ ІНТЕГРОВАНИХ ПЛАТ

**І. І. Пастернак**

Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин



**Висвітлено функціональні можливості та зручність використання системи налаштування інтегрованих плат. Також визначено, що всі вони надають базовий функціонал для роботи апаратними продуктами і лише деякі дозволяють використовувати розширені можливості, що часто можуть бути потрібними. Наведено способи комунікації з цією програмою. Досліджено, що із використанням інтерфейса користувача проєкт є більш зрозумілим, гнучким та зручним для використання. Описано, який саме вплив мають такі кроки, як вибір мови і середовища програмування, програмних засобів, розроблення функціонального забезпечення, а також створення алгоритмів роботи тощо. Проаналізовано стан сучасних системних інтегрованих плат та визначено основний набір компонентів, що потрібен планам для коректного функціонування. Розроблено блок-схеми алгоритмів для визначення основних можливостей системи налаштування інтегрованих плат. Після діагностики інтегрованої плати з використанням різних сучасних систем виявлено такі незручності: якщо особисто не слідкувати за отриманням даних і не аналізувати їх, потім потрібно буде витратити багато часу для перечитування інформації, бо немає можливості скористатися пошуком; при тривалому з'єднання вікно не очищується, тому використання пам'яті стрімко зростає, після чого помітні затримки в роботі застосунку; неможливість використання кількох з'єднань водночас, що необхідно, якщо на плату встановлено кілька операційних систем, які працюють незалежно одна від однієї.**

**Проаналізовано вибір середовища розробки сиестеми інтегрованих плат. Визначено актуальну проблему в середовищі використання, що потребує її вирішення з використанням системи для налаштування системних плат. Встановлено той факт, що область розроблення, діагностики та налаштування апаратних засобів стрімко розвивається і постійно потребує нововведень. Обґрунтовано, що всі програмні засоби, що розробляються, повинні поєднувати високу надійність, доступну ціну, невисокі апаратні затрати та точність наданих результатів. Визначено технічний засіб: периферійний інтерфейс для обміну інформацією. Вибір зроблено, враховуючи такі вимоги до нього: дуплексність, асинхронність, надійність, ціна, доступність реалізації. Вирішено використовувати інтерфейс UART.**

**Ключові слова: плата, інтегрована плата, тестування інтегрованих плат.**

## Вступ

У повсякденному житті ми звертаємось по допомогу до технічних засобів, метою яких є спрощення цього життя. Зараз технічну індустрію використовують у найрізноманітніших сферах: у будівництві, на кухні, у транспортній та побутовій системах тощо. Розвиток технічної індустрії спричиняє постійне підвищення вимог до продуктів, що розробляються та випускаються. Тому

проводяться дослідження, що спрямовані на підвищення надійності, зменшення розмірів пристрою, збільшення функціональності. На постійні аналіз, оцінювання, дослідження та розроблення значно впливає те, що Україна прагне вийти на світовий ринок як одна з найрозвиненіших країн. Для цього потрібно забезпечувати якість, доступну ціну та конкурентноздатність продуктів. З боку апаратного забезпечення спостерігається надмірна мінімізація його розмірів та підвищення списку можливостей продукту. Для вказання логіки роботи та поведінки технічних пристроїв використовують написання коду та його встановлення. Від програмного забезпечення завжди очікують підвищення рівня керування ресурсами системи. Сучасні системи для налаштування зазвичай надають лише базову функціональність, якої буває недостатньо. Інтерфейсом користувача зазвичай є консольне вікно, і керування відбувається за допомогою консольних команд. Для початківця така програма буде не зовсім зручною, оскільки знадобиться час для ознайомлення з командами, їх вивчення та визначення способів використання.

### Аналіз останніх джерел та публікацій

Технічна індустрія постійно прогресує, комп'ютерні науки стають все більш зрілими і не перестають вражати своїми новаціями. З кожним днем найрізноманітніші комп'ютерні технології все частіше використовують у повсякденному житті. Наприклад, для таких сфер, як транспортна, військова, медична та комунікаційна рівень управління вбудованими системами постійно зростає. Система, що знаходиться безпосередньо в певному пристрої і застосовується для керування, моніторингу чи контролю ним, визначається терміном "вбудована система". Вона матиме вигляд спеціалізованої мікропроцесорної системи. Так звані вбудовані системи мають великі можливості. Навіть у побутовій електроніці майже щоденно анонсуються як новинки сучасної техніки, так і певні вдосконалення вже використовуваної. Отже, програмне забезпечення досягає все вищого і вищого рівня. Для отримання очікуваної вигоди від результатів прогресу технічної індустрії, в які вкладено величезний потенціал, від кінцевих продуктів вимагається доступна ціна і водночас висока надійність. Часто трапляється, що сучасні процеси розроблення, валідації і супроводу не мають змоги забезпечити достатньої надійності. Це є достатньо вагомою та актуальною проблемою створення системи, яка допомагатиме під час діагностування продукту.

Розроблення програмного продукту за цим напрямом допоможе дізнатися про те, з якими помилками можна зустрітися при налагодженні плати, ознайомитися з процесом передавання даних від плати для її логування та підвищити рівень досвіду розроблення програмного забезпечення та алгоритмізації.

### Постановка завдання

Описати основні принципи та рекомендації щодо системи налаштування інтегрованих плат. Зробити огляд програмного забезпечення для діагностики та налаштування інтегрованих плат.

## ОСНОВНІ РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ

### Огляд існуючих систем для тестування інтегрованих плат

Є багато програмних продуктів, що допомагають протестувати прошиту плату. Проте таких, які надають великий набір можливостей під час роботи з ними, не так багато. Одними з найрозповсюджених є Minicom, TeraTerm та Screen. Їх розглянемо нижче. Дослідивши функціональні можливості системи, до табл. 1 занесем порівняльні характеристики цих систем.

"Гарячі" кнопки в цьому випадку – підтримка кнопок з командами, які можуть надходити до плати з використанням лише одного натиску. Після аналізу табл. 1 можна зробити висновок про функціональність системи для налаштування інтегрованих плат, що розробляється. Система

налаштування інтегрованих плат повинна реалізовувати всі необхідні функції для забезпечення гнучкості в роботі систем такого типу. Також ця система має бути не дуже затратною, берегти ресурси комп'ютера.

*Таблиця 1*

**Порівняння функціональних можливостей та зручності використання наявних систем**

| Можливість | Minicom | TeraTerm | Screen |
|---|---|---|---|
| Операційна система | Ubuntu | Windows | Ubuntu |
| Передавання UART | Підтримується | Підтримується | Підтримується |
| Передавання TCP/IP | Немає | Підтримується | Немає |
| Підтримка скриптів | Підтримується | Немає | Немає |
| Дозвіл кількох з'єднань одночасно | Немає | Підтримується | Підтримується |
| Багатодокументний інтерфейс | Немає | Немає | Немає |
| Інтерфейс користувача | Консоль | Консоль з графічним меню | Консоль |
| Гарячі кнопки | Немає | Немає | Немає |
| Підтримка пошуку | Підтримується | Немає | Немає |
| Керування | Консольними командами | Консольними командами та інтерфейсом | Консольними командами |
| Робота з файлами | Тільки читання | Тільки читання | Немає |
| Редагування тексту | Підтримується | Підтримується | Немає |

**Аналіз та огляд сфери використання і систем налаштування інтегрованих плат**

Практично всі проміжні та кінцеві чи то апаратні, чи то програмні продукти потребують діагностування для забезпечення правильності роботи і гарантування надійності так само, як і програмне забезпечення системи налаштування інтегрованих плат. Вони повинні допомагати у виявленні різноманітних несправностей, зокрема неправильної поведінки продукту, що тестується. Зазвичай програмні тести та налагоджувальні програми створюють за певними встановленими потребами та умовами використання. Сформульовані вимоги до системи налаштування інтегрованих плат, що описуються з боку реалізації функцій керування, заносять у технічне завдання. Воно містить також функції виконання, яких очікується від продукту, що розробляється.

Враховуючи вимоги користувача, створюється ще функціональна специфікація. Те, наскільки пристрій відповідає поставленим вимогам, визначають за допомогою функцій, що реалізуються контролером для користувача після проектування системи. Весь набір цих функцій і є вмістом функціональної специфікації.

Оцінити кінцевий продукт, а саме системи налаштування інтегрованих плат, та перевірити, наскільки він відповідає вимогам, допомагають технічне завдання та функціональна специфікація.

Для об'ємних продуктів "ручне" тестування може забирати дуже багато часу. Ефективнішим рішенням в такому випадку є створення програмної аплікації, що надаватиме можливість діагностування та налагодження автоматизовано. Метою розроблення такого продукту є мінімізація кількості тестів, що потрібно буде написати для діагностики продукту. Для того, щоб продукт такого типу мав можливість багаторазового використання на різних проектах, потрібно виділити його основний функціонал. Щоб реалізувати поставлену вище задачу, необхідно спочатку ознайомитись з основним наповненням системи керування та визначити способи передавання даних між платою та програмною аплікацією для її діагностики.

Розроблення продукту, що зможе конкурувати на ринку з іншими, потребує немало зусиль та часу. Більша частина затрат найчастіше припадає на не написання коду, а на пошук помилок, діагностування роботи програми та налаштування розробки загалом. Також багато часу витрачається на оцінювання поведінки продукту, оскільки існує потреба перевірити функціональність всього продукту. Існує декілька способів для діагностування та налаштування розробки. Спосіб вибирають залежно від типу продукту.

**Програма з графічним інтерфейсом**

Для продукту такого типу процес налагодження – лише один з елементів системи налаштування. Наступне, що потрібно для забезпечення правильної роботи, – тестування графічного інтерфейсу користувача. Перевірка графічного інтерфейсу користувача є дуже важливим процесом, оскільки він презентує всю систему. Для такого тестування можна використовувати як автоматичні тести, так і тестування "вручну" користувачем. Коли йдеться про реалізацію поведінки апаратного продукту, зрозуміло, що вказане вище наповнення системи буде недостатнім. Можна використати середовища, що надають можливість симулювати апаратну частину. Для налаштування тоді вистачить налагоджування коду. Але коли необхідно надійне програмне забезпечення для певного фізичного пристрою, то краще проводити діагностику після того, як написаний код встановлено в апаратну частину. В такому випадку система налаштування має надавати можливість логування програми.

**Загальна структура інтегрованої плати**

Для функціонування пристрою необхідно декілька базових елементів: центральний процесор; запам'ятовувальний пристрій; пристрої введення і виведення інформації.

Всі наведені компоненти мають набори певних елементів. Якщо говорити про центральний процесор, то він може містити арифметико-логічний пристрій, регістри процесора – внутрішній запам'ятовувальний пристрій, кеш-пам'ять і керуючий пристрій. Запам'ятовувальний пристрій складається із малих запам'ятовувальних одиниць. Пристрої введення/виведення, своєю чергою, мають лінії, якими передають дані [1]. Враховуючи наведені основні елементи та зв'язки між ними, розроблено загальну структурну схему (рис. 1).

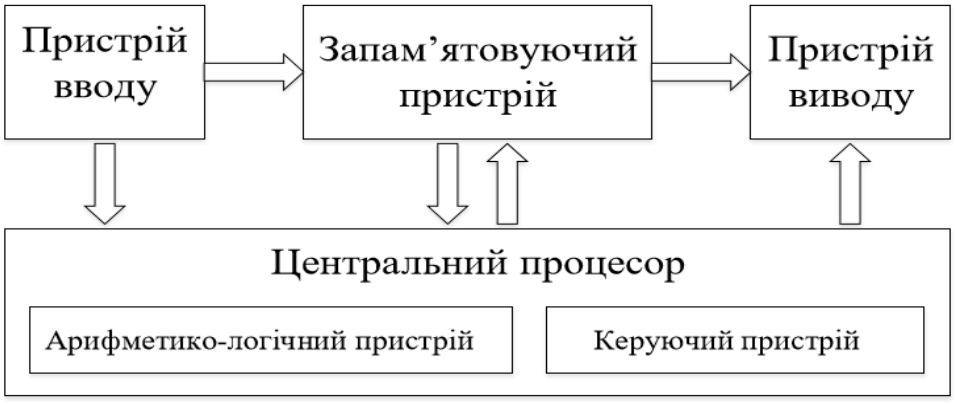


*Рис. 1. Загальна структурна схема інтегрованої плати*

Центральний процесор є основним елементом, який відповідає за роботу пристрою та контролює систему. Стан системи дає змогу вирізняти такі характеристики центральних процесорів:

- кількість ядер – впливає на швидкість роботи системи;
- розрядність – максимальна довжина інформаційного набору, що обробляється одночасно;
- тактова частота – кількість простих операцій, що виконуються за одиницю часу;
- обсяг кеш-пам'яті – використовується для зберігання даних, до яких часто звертаються.

Запам'ятовувальний пристрій призначений для зберігання даних на тривалий і короткий час. Такими даними можуть бути програми, вхідні дані та результати, зокрема проміжні.

Арифметико-логічний пристрій необхідний для перетворення даних за командами програм. Тобто там відбуваються арифметичні дії над числами, перетворення кодів та інше.

Керуючий пристрій координує роботу всіх блоків комп'ютера. Порядок роботи: вибірка команди з оперативної пам'яті, її декодування, передавання в арифметико-логічний пристрій для виконання і, нарешті, отримання та оцінювання результатів.

Блок введення/виведення відповідає за отримання та надання інформації в пристрій та з нього. Наявність цього блоку надає можливість системі взаємодіяти із зовнішнім світом. Блок містить певні пристрої. Існують такі типи: пристрій введення, пристрій виведення, пристрій введення-виведення.

**Принципи побудови систем налаштування інтегрованих плат**

Для забезпечення зручності, надійності та надання користувачу потрібного функціоналу під час роботи зі системою налаштування інтегрованих плат потрібно визначити основні принципи побудови таких систем. Їх визначають за результатами аналізу існуючих систем.

Насамперед – це забезпечення отримання інформації із системної плати. Процес має відбуватися через периферійний інтерфейс. Він має з'єднувати комп'ютер зі встановленим програмним проектом, що розробляється в цій роботі, та плату, що потребує налаштування і діагностику. Системна плата керується командами користувача. Це означає, що потрібний інтерфейс має бути дуплексним. Тоді користувач зможе отримувати логи з плати та за потреби під час прийому відправляти повідомлення.

Асинхронний прийом даних – наступна вимога до периферійного інтерфейсу, оскільки немає визначених часових термінів надходження повідомлень, відправлених платою. Вони надходять, коли дані сформовано і програмне забезпечення має відображати дані зразу ж після надходження.

Обмін файлами мережею за протоколом TCP/IP також є необхідною функцією програмного проєкту, оскільки може виникнути необхідність поділитися файлом із логами з іншими користувачами. Отже, програма повинна давати можливість вибору типу обмінника: працювати як клієнт (відправник) чи як сервер (отримувач). Результати аналізу існуючих систем для тестування плат підказали такий принцип реалізації: інтерфейс користувача, який потрібен для забезпечення зручності та прозорості використання. Інтерфейс програми має забезпечувати інтуїтивне розуміння всіх дій та функціоналу загалом. Також дизайн інтерфейсу користувача має бути побудований так, щоб найпотрібніші можливості програми були у швидкому доступі.

Система тестування повинна підтримувати роботу із файлами. Користувачу може знадобитися відкрити локальний файл для перегляду, зберегти прийняті дані на власному пристрої або оновити той, що існує. Тобто є ще один принцип реалізації програмного засобу. Базові операції редагування тексту також мають бути реалізовані і надані для використання. Наприклад, перед надсиланням файла співробітнику користувачу необхідно буде дописати власні коментарі для виділення певного фрагмента. Цей принцип реалізації надаватиме можливість виправляти та редагувати текст.

Як вже говорилось, логи – це повідомлення про стан системи, які інформують про помилки. чи попередження. Тобто, під час перегляду файла користувачу потрібно надати можливість не читати величезний файл, а зразу виділити негаразди, яких користувач шукає. Це забезпечують, функцією пошуку. Для підтвердження необхідності цього функціоналу наведемо ще одну ситуацію-приклад. Повідомляємо про стан різних компонентів плати, різних модулів і різних програмних частин. Трапляється, що необхідно дослідити поведінку тільки якогось певного модуля чи процесу. Тоді у пошуковому рядку зазначають ключові слова і проводять діагностику.

Отож, визначено основні принципи для реалізації програмного продукту, що забезпечуватимуть доцільність використання розроблюваного забезпечення.

**Аналіз засобів для реалізації системи налаштування інтегрованої плати**

Однією з найголовніших потреб, яку повинен задовольняти завершений програмний продукт, є його якість. Це характеристика, що інформує про рівень відповідності програмного забезпечення вимогам. Для того, щоб визначити якість продукту, спочатку необхідно його оцінити за критеріями якості програмного забезпечення. Існують такі критерії [2, 3]:

– надійність – підтримка визначеної працездатності;

– функціональність – здатність вирішувати поставлені завдання;

– продуктивність – забезпечення необхідної працездатності;

– зручність супроводу – відсутність проблем під час супроводу та підтримки проєкту;

– зручність використання – привабливість, легкість використання та запам’ятовування;

– переносимість – зберігання працездатності за зміни оточення.

На якість програмного продукту впливає багато різних факторів уже з самого початку його розроблення. Так впливає архітектура програмного продукту, мова та середовище програмування, правильне використання додаткових засобів для розроблення та графічне вирішення системи. Також необхідно знайти баланс між кількістю часу, що витрачається на виконання функцій програмним забезпеченням, та обсягом ресурсів комп’ютера, таких як пам’ять, завантаження системи тощо. Та насамперед необхідно дослідити сферу розроблення, ознайомитись із засобами співпраці із продуктом, що розробляється. У нас – це дослідження інтегрованих плат.

Сьогодні існує величезна кількість мов програмування. Кожна з них, безумовно має свої переваги та недоліки. Вибрати мову програмування необхідно залежно від завдання, що постає перед розробником. Наприклад, якщо потрібно розробити мобільний додаток, можна використати Java Android. У випадку потреби створення сайту або іншого інтернет-ресурсу доцільно використовувати мову програмування Java Script. Якщо завданням є програмування, наприклад, певного контролера, то ефективним вибором стануть мови C/C++. За допомогою згаданих останніми мов програмування можна вирішувати величезну кількість завдань. Їх використовують для написання драйверів, системного програмування, для написання потужних клієнтських та серверних програм і навіть для розроблення відеоігор. Зрозуміло, що наведений список далеко не повний.

Реалізовувати програмне забезпечення для тестування інтегрованих плат вирішено мовою програмування високого рівня – C++. Ця мова надає всі засоби для реалізації поставленого завдання. Вона надає простоти програмному коду завдяки підтримці об’єктно-орієнтованого програмування через класи. Ще однією причиною вибору є підтримка узагальненого програмування через шаблони і велика кількість патернів, які можна застосовувати. На відміну від мови C, мова C++ має ще такі нововведення: перевантаження операторів та імен, вбудовані функції, доповнення до стандартої бібліотеки, обробка винятків тощо [4]. Для ознайомлення із новими стандартами вибраної мови програмування використано стандарт C++11, який надає багато додаткових можливостей. Основні наведені в списку: розумні вказівники, лямбда-функції, ключові слова auto,

override і final, а також nullptr. Навіть такий короткий список основних функцій суттєво спрощує процес написання програмного коду. Кінцевий продукт повинен мати графічний інтерфейс. Він дасть змогу створити програму більш дружню для кінцевого користувача. Середовищ програмування, що дозволяють працювати із C++ і розробляти графічний інтерфейс, насправді не так багато. Але дуже відомим і не менш зручним є засіб Qt. Програмне забезпечення для тестування системних інтегрованих плат повинно забезпечувати передавання даних від плати, а також надавати можливість обміну даними через мережу. Відрізняються ці дві функції тим, що різні умови їх виконання і різні протоколи обміну даними. Але у них спільна мета – обмін даними, отже, можна використати один інтерфейс, який оцінюватиме надані йому параметри і, відштовхуючись від них, здійснюваиме передавання.

Такий задум можна реалізувати, використавши шаблон (патерн) проєктування Factory Method. Він дозволяє системі залишатись незалежною від різних типів об’єктів. Це уможливлюється використанням механізму поліморфізму – концепції об’єктно-орієнтованого програмування. Згідно згаданого патерна класи всіх кінцевих типів наслідуються від одного абстрактного базового класу. Він призначений для поліморфного використання і містить оголошення всіх функцій, що можуть застосовуватись до різних типів об’єктів [5, 6]. Для кращого розуміння Factory Method UML-діаграму класів патерна зображено на рис. 2.

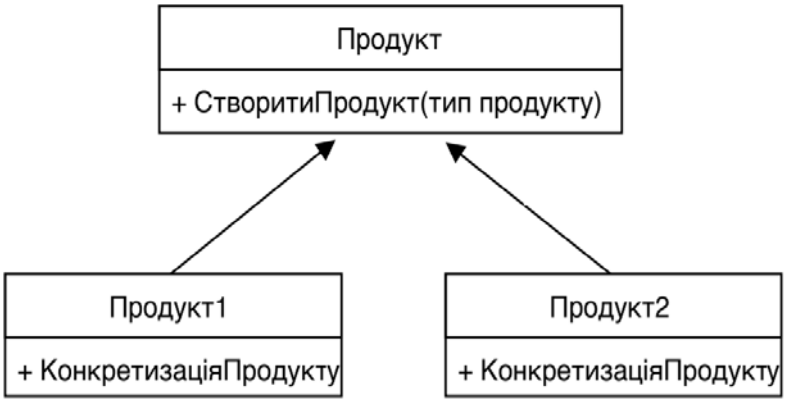


*Рис. 2. UML-діаграма класів патерна Factory Method*

Ще одним засобом, що значно спростить розроблення продукту, є збірка BOOST C++. Це зібрання великої кількості незалежних бібліотек, що розроблялися незалежними програмістами. Суттєво важливим є те, що цей набір ретельно перевірявся і тестувався на різних платформах, і виявити неправильну поведінку малоймовірно. BOOST C++ вважалися додатковим розширенням мови C++. Але багато функцій бібліотеки BOOST C++ занесено в стандарт C++11. Використання компонентів BOOST C++ значно економить час та сили програміста, особливо, якщо проект має бути кросплатформним. Компоненти бібліотеки є платформонезалежними. На рішення про використання цього допоміжного засобу також вплинуло те, що він забезпечений повною та зрозумілою документацією. Це надає можливість витрачати більше часу на розроблення архітектури продукту замість того, щоб реалізовувати прості функції.

### Вибір периферійного інтерфейсу

Зазначалось, що обмін інформацією між інтегрованою платою та комп’ютером із програмним продуктом, що розробляється, проводиться із використанням периферійного інтерфейсу.

Необхідно вибрати такий інтерфейс, що можна буде легко отримати, тобто він має бути загальнодоступним. Сьогодні день, майже всі плати і, відповідно, процесори забезпечені підтримкою послідовних портів. Дані передають однією лінією в певному напрямі по одному біту. Звісно, паралельні порти пришвидшують передавання даних. Та є випадки, коли послідовні порти мають перевагу над паралельними, наприклад, у ситуаціях, де не потрібна висока швидкість обміну даними. Тоді ми економимо апаратні ресурси, оскільки для паралельного інтерфейсу потрібно кілька ліній для одночасного передавання. Одним з найвідоміших та найдоступніших є UART, або інша назва – COM універсальний асинхронний приймач/передавач.

Функціональність цього інтерфейсу залежить від спеціального контролюючого чипа. Він розміщує на собі буфер об'ємом від 16 до 64 кілобайт. Цей буфер забезпечує кешування даних, завдяки чому немає втрат даних під час прийому і також дозволяється формування пакетів по 8 бітів, що передаються одним повідомленням. Передавання інформації забезпечують дві лінії, що скеровані в протилежні боки. Тобто, потрібно лише два піни: RX – для прийому інформації та TX – для її відправлення. Ще однією перевагою для використання послідовного інтерфейсу є те, що перехідні пристрої UART-USB є легкодоступними.

## Алгоритми роботи системи налаштування інтегрованих плат

Для реалізації програмного продукту, що буде використовуватись для діагностики прошитої плати, потрібно розробити алгоритм роботи системи загалом та алгоритми роботи її окремих компонентів. Це допоможе правильно реалізувати роботу та можливість використання функціональних переваг програми. Розроблено алгоритм основної функції системи, зображнений на рис. 3.

Алгоритм дозволив користувачу отримувати дані з плати через периферійний пристрій UART та можливість переглядати отримані дані, що постійно оновлюються. Тому для однієї сесії обміну даними створюється окремий потік.

Отже, як тільки надходить повідомлення, вікно одразу оновлюється із врахуванням нового повідомлення [7–9]. Зупиняється обмін даними вручну. Після випуску відповідної події прийом і передавання даних призупиняються, і з'єднання розривається. Дані повинні залишатися на екрані. Вони можуть знадобитися користувачу для наступних дій, наприклад, таких як зберігання у файлі, відправка їх іншому користувачу з використанням TCP/IP або пошук за цими даними.

Результатом виконання функції пошуку зазвичай є підсвітка всіх входжень, що знайдено в тексті. Оскільки лог-дані можуть бути дуже об'ємними, такий результат, як підсвітка може не задовольняти користувача, бо однаково потрібно буде перегортати велику кількість сторінок для знаходження потрібного результату-входження. Запам'ятовування номерів рядків із тексту, в якому проводиться пошук, а також і з вікна результатів виконується для спрощення пошуку всіх входжень.

Користувач переглядатиме наведений список результатів, і після вибору якогось одного з них в тексті логів відбудеться прокрутка до того місця, де є це входження з виділенням відповідного рядка. Коли тестують плату на надійність, логування системної плати може займати доволі тривалий час – іноді навіть більше доби.

Для того, щоб користувач не виконував все тестування вручну і не сидів під час усього процесу тестування за комп'ютером, пропонується реалізувати можливість завантаження скриптів. Як видно з алгоритму, все, що потрібно зробити користувачу – це написати скрипт і під час прийому даних з плати скрипт завантажити (рис. 3).

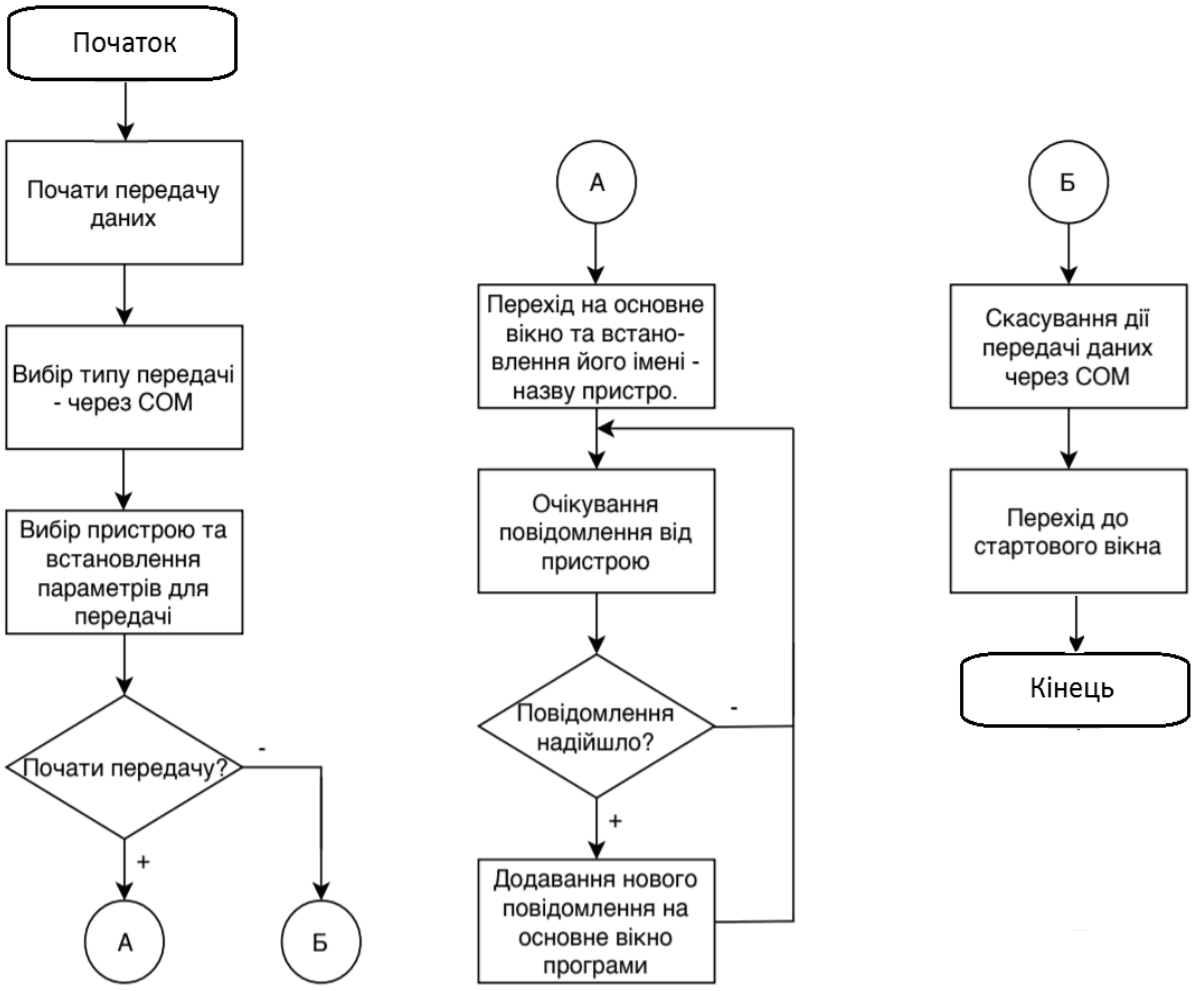


*Рис. 3. Алгоритм з'єднання та роботи з COM-пристроєм*

Поки скрипт виконується, користувач може відлучитися, а потім проаналізувати отримані логи і отримані результати. В наведеному вище алгоритмі є блок "Аналізується вибраний файл". на якому слід зупинитися окремо. Для виконання цієї частина схеми необхідно реалізувати власний аналізатор псевдокоду. Визначено, що основними командами, які можуть знадобитися користувачу, є відправка команди і затримка. Цього цілком достатньо, щоб протесувати плату. Алгоритм роботи аналізу та виконання файла-скрипта зображено на рис. 4.

Можлива ситуація, коли певний набір команд потрібно повторювати кілька разів. Саме тому пропонується введена підтримка циклічного виконання. Це означає, що певну частину команд виконуватимуть вказану користувачем кількість разів.

Реалізовано такі команди (наведено також визначення та спосіб їх правильного використання):

– send – відправка команди на плату; після ключового слова через пробіл вводити команду, відправлення якої необхідне;

– pause – затримка; після ключового слова через пробіл потрібно зазначати час затримки в секундах;

– repeat…done – ключові слова, що сигналізують про ініціалізацію циклу; вміст циклу – визначений набір команд, що потрібно виконати більше одного разу. Після ключового слова repeat через пробіл потрібно вказати ціле число, що означає кількість повторень тіла циклу.

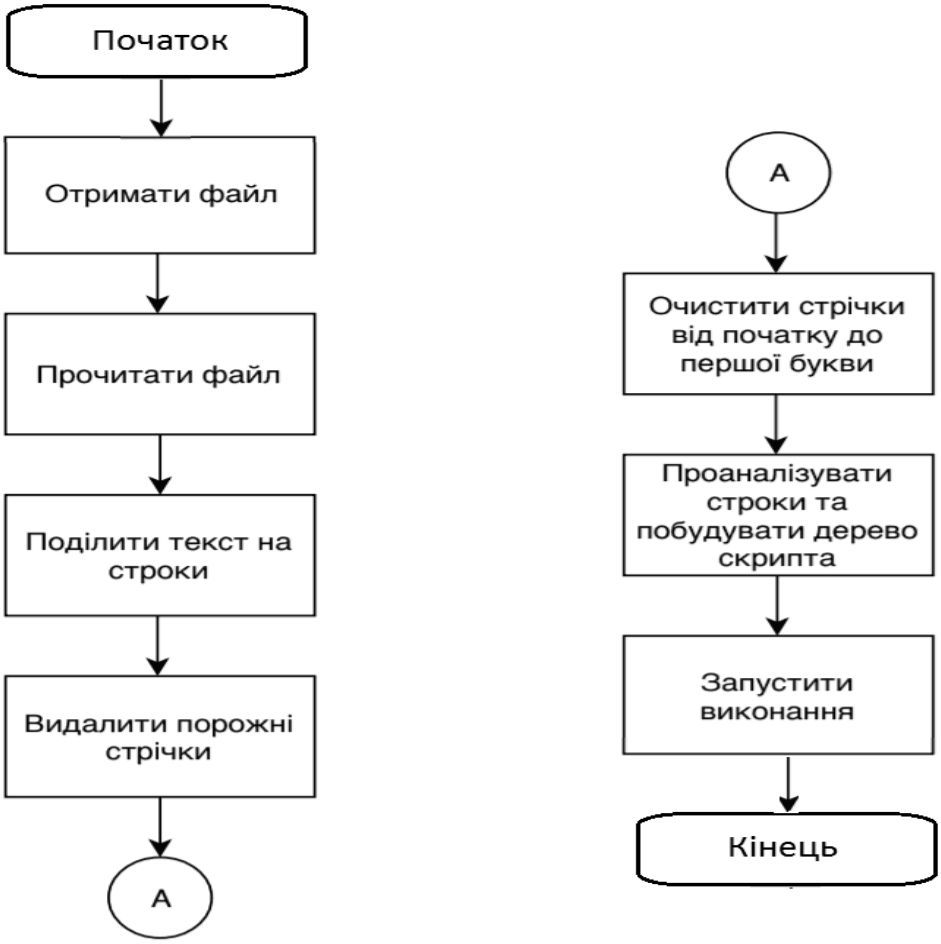


*Рис. 4. Алгоритм роботи аналізатора скриптового файлу*

Ще однією функцією, що суттєво спростить процес тестування плати – це вікно з набором команд, які можна відправляти платі. Тобто замість того, щоб постійно прописувати команду, потрібно буде просто клікати мишкою на ярлик. Для цього створено окреме вікно із двома вкладками. На першій вкладці розміщується набір кнопок-ярликів з командами. Друга вкладка призначена для створення нового ярлика.

Вкладка створення нового ярлика містить такі поля для введення: опис команди та команда. Також є кнопка «Зберегти», що додаватиме кнопку до набору на першу вкладку. Кожна кнопка з набору має своє меню, що викликається правою кнопкою миші. Меню дозволяє редагувати та видаляти кнопку.

**Висновки**

Проаналізовано сучасний стан розроблення системних плат та вбудованих систем. Також наведено та проаналізовано основні види систем налаштування інтегрованих плат, вибір кожної з них залежить від продукту, що розробляється. Вказано, які системи доцільно використовувати для консольних програмних графічних продуктів і описано систему для налаштування апаратних продуктів. Також вказано приклади існуючих систем, що надають можливість використовувати той чи інший спосіб перевірки правильності роботи та налаштування продукту.

Запропоновано та проаналізовано основні складові будь-якої системної плати. Вказано, що такими складовими є центральний процесор, пристрої введення і виведення, а ще запам’ятовувальний пристрій. Базуючись на цих основах, розроблено на описано загальну структурну

схему інтегрованої плати. З'ясовано та описано всі способи взаємодії між вказаними основними модулями. Додатково описано базу процесорів – арифметико-логічний пристрій та керуючий пристрій та визначено характеристики процесорів, за якими їх вибирають, проаналізовано вибір мови програмування. Врахувавши, що ця система для налаштування інтегрованих плат повинна співпрацювати з апаратними засобами, а також завдяки зручності використання та широкому доступному функціоналу вирішено використовувати мову програмування С++. З метою навчання цей проєкт розроблено з використанням стандарту С++11. Через великий об'єм можливостей, а також простоту використання буде застосовуватись збірка бібліотек BOOST C++. Її використання дозволить за допомогою простих інтерфейсів реалізувати багато функцій, залишаючи багато часу на проєктування та створення програмного коду.

Проаналізовано вибір середовища розроблення сиестеми інтегрованих плат. У результаті вирішено використовувати Qt Creator. Вагомою причиною використання цього засобу стала необхідність створення інтерфейсу користувача. Було описано базові властивості системи налаштування інтегрованих плат. Створення функціональних властивостей проводилось на основі аналізу існуючих систем. Додатково змодельовано ситуації із наведенням основних проблеми, що можуть виникати при тестуванні системної плати та знайдено шляхи вирішення цих проблем із використанням наведених функціональних можливостей. Розроблено блок-схеми алгоритмів для визначення основних можливостей системи налаштування інтегрованх плат. Наявність схем алгоритмів роботи програми допоможе розробнику зрозуміти принципи роботи системи та коректно написати код програми.

**Список літератури**

*1. Pelezhko D. D. Object technologies C ++ 11 / D. D. Pelezhko, V. M. Teslyuk. Lviv: Lviv Polytechnic Publishing House, 2013, 360 p.*

*2. Melnyk A. O. COMPUTER ARCHITECTURE / Anatoliy Oleksiyovych Melnyk. – Lutsk: Volyn Regional Printing House, 2008, 470 p.*

*3. Bilas O. E. Quality of software and testing / Orest Bilas. Lviv: Lviv Polytechnic Publishing House, 2011, 216 p.*

*4. Kravets P. O. Object-oriented programming / Petro Kravets. Lviv: Lviv Polytechnic Publishing House, 2012. 624 p.*

*5. Petrik M. R. Software modeling: scientific and methodical manual / MR Petrik, O. Yu. Petryk Ternopil: Ivan Pulyuy TNTU Publishing House, 2015. 200 p.*

*6. Robert J. Oberg "COM Technology Fundamentals and Programming = Understanding and Programming COM: A Practical Guide to Windows 2000 First Edition". M .: Williams, 2000. 480 p.*

*7. McGregor J., Sykes D. "Testing object-oriented software". K: Diasoft, 2002. 432 p.*

*8. Tamre L. Introduction to software testing. M.: Williams, 2003. 368 p.*

*9. Douglas Kamer, Devid L. Stevens // TCP / IP Networks, Vol. 3. Development of client / server applications. M.: Williams, 2002. 592 p.*

# PRINCIPLES OF BUILDING SYSTEMS FOR INSTALLING INTEGRATED BOARDS

**I. Pasternak**

Lviv Politechnic National University,
Computer Engineering Department



**The article examined the functionality and convenience of the integrated circuit setup system. It is also determined that they all provide basic functionality for working with hardware products, and only some allow you to use advanced features that may often be required. And the ways of communication with this program are given. It is proved that when using the interface, the project is more understandable, flexible and better to use. It describes what effect such steps as the choice of a programming language and environment, software tools, the development of functional support, as well as the creation of work algorithms and more, have. The analysis of modern integrated system boards is carried out and the main set of components is determined; plans need for proper functioning. Algorithm flowcharts have been developed to determine the main capabilities of the integrated circuit tuning system.**

**There was a diagnosis of the integrated board using various modern systems, the following inconveniences were identified: if you do not personally monitor the data and analyze it, then you will need to spend a lot of time to re-read the information, because there is no way to use the search; with a long connection, the window does not clear, so the memory usage is growing rapidly, after which there are noticeable delays in the operation of applications; the impossibility of using several connections at the same time, which is very necessary if the board has several operating systems operating independently of each other.**

**The choice of the development environment for the integrated circuit siesta is analyzed. The actual problem in the environment of use is determined, which requires its solution using the system to configure motherboards. It has been established that the field of hardware development, diagnostics and tuning is rapidly developing and constantly needs innovations. It is proved that all the software developed should combine high reliability, affordable price, low hardware costs and the accuracy of the results provided. The definition of a technical tool - a peripheral interface to ensure the exchange of information. The choice is made, taking into account such requirements for it: duplex, asynchrony, reliability, price, availability. It was decided to use the UART interface.**

**Keywords: board, integrated board, integrated board testing.**