МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.

«____»_____ 20__ р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Тестове програмне забезпечення пульта перевірки комп'ютерних блоків

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям підготовки 6.050102 – "Комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	Щербакова М.Є. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Критська Я.О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Рижков А.А. (ініціали, прізвище)
Група:		КІ-14з

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050102 – "Комп'ютерна інженерія"
(шифр і назва)
Спеціальність
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри
І.С. Скарга-Бандурова
«_____» _____20____р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Рижкова Антона Анатолійовича
(прізвище, ім'я, по батькові)

1. Тема роботи Тестове програмне забезпечення пульта перевірки комп'ютерних блоків

керівник проекту (роботи) Щербакова М.Є., к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від 14 05 2018 р. № 117/48

2. Термін подання студентом роботи 11.06.2018

3. Вихідні дані до роботи Матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз предметної області та постановка задачі; вибір засобів для розробки; розробка тестового програмного забезпечення пульта перевірки комп'ютерних блоків; охорона праці та безпека в надзвичайних ситуаціях.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці та безпека в надзвичайних ситуаціях	ст. викл. Критська Я.О.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Ознайомлення з предметною галуззю	03.05.18 – 6.05.18	
2	Аналіз існуючих аналогів	7.05.18 – 8.05.18	
3	Вибір засобів для розробки	9.05.18 – 15.05.18	
4	Розробка системи	16.05.18 – 01.06.18	
5	Розробка розділу «Охорона праці та	02.06.18 – 06.06.18	
	безпека в надзвичайних ситуаціях»		
6	Оформлення пояснювальної записки	07.06.18 – 11.06.18	

Студент _____ Рижков А.А.
(підпис) (прізвище та ініціали)

Керівник _____ Щербакова М.Є.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту (роботи) бакалавра: 104 с., 12 рис., 6 табл., 18 бібліографічних джерел посилань, 3 додатки.

Об’єкт розробки: програмне забезпечення для тестування пульту перевірки комп’ютерних блоків.

Мета роботи: розробка тестового програмного забезпечення пульта перевірки комп’ютерних блоків.

В проекті виконано:

1. Проведено аналіз аспектів та методів розробки програмного забезпечення для тестування пульту перевірки комп’ютерних блоків.
2. Спроектовано та розроблено тестове програмне забезпечення пульта перевірки комп’ютерних блоків.
3. Зроблено аналіз умов праці та приведено рекомендації щодо організації робочого місця.

Отримані наступні результати: придатне до використання програмне забезпечення для тестування пульту перевірки комп’ютерних блоків.

Практичне значення, галузь застосування роботи: програмне забезпечення може бути використане у галузі продажу та реклами.

Ключові слова: ПУЛЬТ ПЕРЕВІРКИ ПП-78, ДІАГНОСТИКА, ТЕСТУВАННЯ, МОВА ПРОГРАМУВАННЯ C++, WINPCAP, QtCreator

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А,. м. Сєвєродонецьк, 93400.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

АСУ ТП – автоматизована система управління технологічними процесами.

ВДТ – відео дисплейний термінал.

ЕОМ – електронно-обчислювальна машина.

КСТ – контролер сигнальної точки КСТ-2 ИТКЯ.468332.229.

МСв – модуль зв’язку МСв-5 ИТКЯ.467142.023.

ПЕОМ – персональна електронно-обчислювальна машина.

ПЗ – програмне забезпечення.

ПК – персональний комп’ютер.

ПП – пульт перевірки ПП-78 ИТКЯ.442261.047.

ПТК – програмно-технічний комплекс.

СКУ – система контролю і управління.

ВСТУП

За останні роки в мікроелектроніці швидкий розвиток отримав напрямок, пов'язаний з випуском мікроконтролерів, які призначені для "інтелектуалізації" устаткування різноманітного призначення. Використання мікроконтролерів в системах управління забезпечує досягнення винятково високих показників вартості, надійності, споживаній потужності, габаритних розмірів.

Сучасний рівень розвитку й удосконалення електронної техніки дозволяє застосовувати системи інтелектуального моніторингу та тестування пристроїв, не долучаючи безпосередньо до процесу тестування електронного устаткування людини.

У даному дипломному проекті розробляється програмне забезпечення для тестування пульту перевірки комп’ютерних блоків.

Пульт перевірки застосовується під час перевірки та ремонту пристроїв диспетчерської централізації, з метою визначення їх справності та відповідності вимогам.

Сьогодні існує велика потреба підприємств у комп'ютерних програмах, що реалізують автоматичну перевірку устаткування. Такі програмні засоби дозволяють значно заощадити на часі перевірки, зменшити долю пропущених несправностей при перевірці безпосередньо майстром з ремонту, знизити залежність результатів перевірки від людського фактору.

Процес тестування має на меті продемонструвати розробникам і замовникам, що пристрій відповідає вимогам, а також, у разі їх наявності, виявити ситуації, в яких поведінка тестованого пристрою є неправильною, небажаною або не відповідає специфікації.

Програма повинна забезпечувати реалізацію наступних завдань:

1) виконання тесту модемів на наявність несправностей;

2) виконання тесту наладки.

3) перевірка повідомлень, що вводяться, на коректність та їх коригування у разі необхідності;

4) формування короткого і повного протоколів за результатами тестування.

Метою виконання дипломного проекту є розробка тестового програмного забезпечення пульта перевірки комп’ютерних блоків, а також вивчення питань, що стосуються охорони праці та навколишнього середовища.

1 АНАЛІЗ ВИМОГ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис пульта перевірки ПП-78

Пульт перевірки ПП-78 (далі – ПП) застосовується при перевірці та ремонті пристроїв диспетчерської централізації, з метою визначення їх справності та відповідності вимогам.

ПП виготовляється в якості допоміжного виробу для систем диспетчерської централізації.

ПП призначений для контролю якості контролерів сигнальних точок КСТ-2 і модулів зв’язку МСВ-5, а в подальшому і інших виробів при серійному виробництві.

ПП забезпечує виконання наступних основних функцій:

– обмін повідомленнями з персональною електронною обчислювальною машиною (далі – ПЕОМ) по 100 Мбітчерез канал зв’язку Ethernet (далі – Ethernet);

– формування повідомлень з інформацією, прийнятою від персональної електронної обчислювальної машини, з наступною передачею через модулятор-демодулятор МДМ-20 467766.007 (далі – МДМ-20) або модулятор-демодулятор МДМ-20/1 467766.007-01 (далі – МДМ-20/1). Номер МДМ-20 (МДМ-20/1) визначається в повідомленні від ПЕОМ;

– формування повідомлень з інформацією, прийнятою від МДМ-20 і МДМ-20/1, з подальшою передачею в ПЕОМ;

– комутацію 26 каналів, згідно командам від ПЕОМ по Ethernet;

– управління вісьмома виходами для формування адреси;

– підключення підвищувального трансформатора;

– відключення живлення пристрою, що перевіряється;

– перемикання схеми для вимірювання струму або напруги;

– управління подачею напруги на канали 13 і 14 щодо каналу 24;

– формування напруг 3,3 і 5 В.

1.2 Основи проведення тестування

Діагностичні програми, застосовувані в ПК, можна розділити на чотири рівні:

– діагностичні програми BIOS – POST (Power – OnSelfTest – процедура самоперевірки при включенні);

– діагностичні програми операційних систем;

– діагностичні програми фірм-виробників обладнання;

– діагностичні програми загального призначення.

Тестування – це одна з технік контролю якості, що включає в себе дії з планування робіт (TestManagement), проектування тестів (TestDesign), виконання тестування (TestExecution) та аналізу отриманих результатів (TestAnalysis).

Процес тестування має дві різні цілі:

– продемонструвати розробникам і замовникам, що пристрій відповідає вимогам;

– виявити ситуації, в яких поведінка тестованого пристрою є неправильною, небажаною або не відповідає специфікації.

В даний час промисловістю випускаються інтегральні схеми, складні за своєю структурою і функціональним призначенням. У зв’язку з цим виникає проблема контролю результатів роботи придатних інтегральних схем і виявлення причин виникаючих поламок.

Витрати на тестування складних інтегральних схем із залученням контрольно-вимірювальної апаратури можуть у багато разів перевищувати вартість самої інтегральної схеми через тривалість процесу тестування і складність його реалізації.

При програмному тестуванні створених моделей необхідно враховувати можливості використовуваної обчислювальної техніки.

Не тільки інструменти допомагають ремонтувати ПК. Щоб перевірити працездатність комп’ютера, його окремих модулів, точно визначити робочі параметри, існує спеціальне програмне забезпечення, яке є зручним діагностичним інструментом. З його допомогою можна протестувати і визначити характеристики системи, окремих пристроїв і прийняти рішення про доцільність заміни.

Програми можна розділити на ті, які тестують ПК або інший електронно-цифровий пристрій повністю, дозволяючи визначити конфігурацію пристрою, виводять специфікацію, перевіряючи всі компоненти тестованого пристрою; і спеціальні програми, призначені для перевірки лише окремих частин необхідного електронно-цифрового обладнання.

Діагностичне програмне забезпечення – програмне забезпечення, призначене для виявлення несправностей тестованого електронного пристрою.

Більшість тестових програм можна запускати в пакетному режимі, що дозволяє без втручання оператора виконати цілу серію тестів. Можна умовно розділити тестові програми на програми автоматизованої діагностики та програми для проведення ручного тестування. Програми автоматизованої діагностики найбільш ефективні в тому випадку, якщо вам необхідно виявити можливі дефекти або виконати однакову послідовність тестів для декількох пристроїв одночасно.

Програми для тестування обладнання аналізують конфігурацію тестованих електронно-цифрових пристроїв і видають детальну інформацію про результати проведеного дослідження (у вікні виводу програми, або ж в окремо створеному файлі протоколу).

Розрізняють два типи мікродіагностування: вбудоване і те, що завантажується. У разі вбудованогомікродіагностуваннязастосовується діагностичні прошивки, які розміщуються в постійній пам’яті ЕОМ. При

мікродіагностуванні, що завантажується, діагностичні прошивкирозміщуються на зовнішньому носії даних. При зберіганні в постійній пам’яті мікродіагностуванняявляє собою звичайну прошивку, що використовує стандартний набір мікрооперацій. Однак внаслідок обмеженого обсягу постійної пам’яті на обсяг мікродіагностування, що застосовується, накладаються досить жорсткі обмеження, в результаті чого доводиться використовувати різні способи стиснення інформації. Для цієї мети іноді використовують спеціально створені тестові набори. Це дозволяє зменшити необхідний для тестових констант обсяг мікропрограмної пам’яті [3].

Як правило, застосуванні мікропрограми, що зберігається в постійній пам’яті для транспортування результатів перевірки до місця з еталонів використовуються стандартні мікрооперації, а для порівняння – такі схеми, як суматор, схеми контролю або аналізу умов. Як прошивки, для аналізу використовується також вбудоване діагностування стану схем контролю ЕОМ.

У більшості випадків для перевірки правильності функціонування обладнання на вхід пристрою подається певна інформаційна послідовність. За отриманими результатами на виході, при їх порівнянні з еталонними результатами, і робиться висновок про справність або про несправності тестованого устаткування. Еталонні значення визначаються в процесі розробки пристрою і прописуються в супроводжувальній електронно-цифровий пристрій технічній документації.

1.3 Загальна характеристика мови програмування С++

С++ в даний час вважається однією з основних мов, що використовуються для розробки комерційних програмних продуктів. В останні роки ця ситуація почала трохи змінюватися на користь такої мови програмування, як Java, але вже зараз багато програмістів, які кинули С++ заради Java, вирішили за краще повернутися до використання мови С++.

C++ є мовою програмування загального призначення. Природна для неї область застосування – системне програмування, що розуміється в широкому сенсі цього слова. Крім того, C++ успішно використовується в багатьох областях створення додатків, що далеко виходять за зазначені рамки. Реалізації C++ тепер є на всіх машинах, починаючи з самих скромних мікрокомп'ютерів – до найбільших супер-ЕОМ, і практично для всіх операційних систем.

Безумовно C++ багато чим зобов'язаний мові C, яка зберігається як його підмножина. Збережено і всі властиві C, засоби низького рівня, призначені для вирішення найбільш нагальних завдань системного програмування [13].

Основне призначення C++ – спростити і зробити більш приємним процес програмування для окремого програміста.

У проекті C++ особлива увага приділяється структуруванню програми. Невелику програму (скажімо, не більше 1000 рядків) можна змусити працювати, порушуючи всі правила хорошого стилю програмування. Однак, діючи так, людина вже не зможе впоратися з великою програмою. Якщо у програми в 10 000 рядків погана структура, то можна виявити, що нові помилки з'являються в ній так само швидко, як виправляються старі [16]. C++ створювалася з метою, щоб велику програму можна було структурувати таким чином, щоб одній людині не довелося працювати з текстом в 25000 рядків.

Існують, звичайно, програми ще більшого розміру. Однак ті з них, які дійсно використовуються, зазвичай можна розбити на кілька практично незалежних частин, кожна з яких має значно менший розмір. Природно, труднощі написання та супроводу програми визначаються не тільки числом рядків тексту, а й складністю предметної області [5].

Об'єктно-орієнтоване програмування – це метод програмування, спосіб написання «хороших» програм для безлічі завдань.

Мова C++ характеризується введенням гнучких і ефективних засобів, призначених для побудови нових типів. Програміст структурує свою задачу, визначивши нові типи, які точно відповідають поняттям предметної області

завдання. Такий метод побудови програми називають абстракцією даних. Інформація про типи міститься в деяких об'єктах типів, визначених користувачем. Програмування з використанням таких об'єктів називають об'єктно-орієнтованим. Якщо цей метод застосовується правильно, то програми стають коротшими і зрозумілішими, а супровід їх спрощується.

Ключовим поняттям С++ є клас – визначений користувачем тип. Класи забезпечують приховування даних, їх ініціалізацію, неявне перетворення типів користувача, динамічне завдання типів, контрольоване користувачем управління пам'яттю і засоби для перевантаження операцій. С++ містить удосконалення, прямо з класами не зв'язані: символічні константи, функції-підстановки, стандартні значення параметрів функцій, перевантаження імен функцій, операції керування вільною пам'яттю і контрольний тип [7].

Об'єктно-орієнтоване програмування найкращим чином надає технологію управління елементами будь-якої складності, створюючи умови для багаторазового використання програмних компонентів та об'єднання даних з методами їх обробки.

Суть об'єктно-орієнтованого програмування полягає у використанні концепції «об'єктів», тобто, швидше, образів, ніж даних.

Керівна ідея цього підходу полягає в прагненні зв'язати дані з методами, що обробляють ці дані, в єдине ціле – об'єкт. Об'єкти мають характеристики і можливості.

Фактично об'єктно-орієнтоване програмування можна розглядати як модульне програмування нового рівня, коли замість багато в чому випадкового, механічного об'єднання процедур і даних, акцент робиться на їх смисловий зв'язок.

Об'єктна модель здатна однаково добре описати як елементи управління графічного інтерфейсу (типу кнопок і розкривних списків), так і реальні об'єкти. Таким чином, завдання об'єктно-орієнтованого програмування полягає в тому, щоб правильно уявити ці об'єкти на мові програмування.

У мові С ++ повністю підтримуються принципи об’єктно-орієнтованого програмування, включаючи три кити, на яких воно складається: інкапсуляцію, успадкування і поліморфізм.

Інкапсуляція – поєднання структур даних з функціями (методами), призначеними для маніпулювання цими даними. Інкапсуляція досягається шляхом введення класу нового механізму структурування та типізації даних [6].

Спадкування – створення нових, похідних класів, які успадковують дані і функції від одного або декількох раніше визначених базових класів. При цьому можливе перевизначення або додавання нових даних і методів. В результаті створюється ієрархія класів.

Поліморфізм – привласнення методом єдиного імені або ідентифікатора в рамках ієрархії класів таким чином, щоб будь-який клас в ієрархії мав можливість по-своєму виконувати пов’язані із цим методом дії.

Мова С++ проектувалася як «краща С», що підтримує абстракцію даних і об’єктно-орієнтоване програмування. При цьому вонаповинна бути придатною для більшості основних завдань системного програмування.

Основні труднощі для мови, яка створювалася в розрахунку на методи приховування даних, абстракції даних і об’єктно-орієнтованого програмування, в тому, що для того, щоб бути мовою загального призначення, вона повинна:

– йти на традиційних машинах;
– співіснувати з традиційними операційними системами і мовами;
– змагатися з традиційними мовами програмування в ефективності виконання програми;
– бути придатною у всіх основних областях створюваних додатків.

Мова проектувалася в розрахунку на сучасні методи трансляції, які забезпечують перевірку узгодженості програми, її ефективність і компактність подання. Основним засобом боротьби зі складністю програм є, насамперед, строгий контроль типів і інкапсуляція. Особливо це стосується великих програм,

створюваних багатьма людьми. Користувач може не виступати одним з творців таких програм, і може взагалі не бути програмістом. Оскільки ніяку справжню програму не можна написати без підтримки бібліотек, створюваних іншими програмістами, останнє зауваження можна віднести практично до всіх програм [8].

C++ проектувався для підтримки того принципу, що всяка програма є моделлю деяких існуючих в реальності понять, а клас є конкретним поданням поняття, взятого з області програми. Тому класи пронизують всю програму на C++, і накладаються жорсткі вимоги на гнучкість поняття класу, компактність об'єктів класу та ефективність їх використання.

Мова C ++ стала потужним і стрімким ривком у розвитку програмування. C ++ і донині займає панівне становище серед мов програмування в світі. Величезна безліч професійних програмістів використовують саме її при розробці різного роду проектів. Очевидно, ця мова буде зберігати своє солідне становище ще не один рік, при цьому як і раніше розвиваючись і вдосконалюючись.

1.4 Інтегроване середовище розробки QtCreator

Інтегроване середовище розробки (IntegratedDevelopmentEnvironment – IDE) – це набір інструментів, об'єднаних в одному додатку. Його використання істотно полегшує роботу розробника. QtCreator входить в пакет Qt SDK. Мета створення інтегрованого середовища розробки QtCreator – заповнити цю нішу і надати платформо незалежне середовище розробки, орієнтоване на Qt-проекти на C ++ і також QML. Тепер на будь якій підтримуваній платформі ви можете використовувати одну й ту ж інтегровану середу розробки.

Середа QtCreator не має свого власного компілятора, компоновщика і відладчика, тому використовуються доступні на платформі засоби. У Windows це MinGW (англ. Minimalist GNU for Windows), який міститься в пакеті Qt SDK.

Майстер проектів – це засіб для автоматичного створення мінімальних стартових проектів для ваших програм. Створюваний проект складається з ресурсів, вихідного тексту і заголовку.

Текстовий редактор – це, власне, засіб, за допомогою якого створюється програмний код. Редактор сам підсвічує синтаксис, автоматично доповнює код і має ряд інших переваг, що роблять програмування швидше і зручніше.

За допомогою вбудованої в середу програми QtDesigner можна створювати або змінювати форми, не покидаючи інтегроване середовище розробки.

QtCreator надає готові шаблони проектів, за допомогою яких можна легко почати створення програми. Шаблони – це «стрижень-скелет», що містить ресурси, вихідний текст і заголовки.

Середа QtCreator в режимі редагування володіє наступними основними зонами:

– головне меню;
– кнопки активації вікон виводу з поясненнями праворуч від їх номерів;
– смуга зміни режимів (редагування, наладка, допомогу і т.д.) і секція компілювання і запуску;
– вікно оглядача проектів;
– вікно редактора (основна робоча область, у якій можна редагувати файли).

QtCreator – це інтегроване середовище розробки, спеціально розраховане на специфіку бібліотеки Qt. Він надає розробнику весь необхідний інструментарій для створення програм. Його використання полегшує роботу і дозволяє редагувати і компілювати програми без необхідності застосування інших додатків. Типові дії, які можна виконати в середовищі розробки QtCreator:

– створити базовий виконуваний додаток за допомогою майстра проектів без написання коду;
– редагувати вихідний код ваших програм;
– додавати і викликати нові файли;

– створювати графічну оболонку для вашоїрозробкиінтерактивно, за допомогою вбудованої програми QtDesigner;

– компілювати і компонувати програми;

– наладка готової програми.

Для наладки програм QtCreator надає інтерактивний відладчик. Існують помилки синтаксису, компонування, часу виконання та логічні помилки. Відладчик використовується для виявлення місця та причин виникнення двох останніх типів помилок. Для цього відладчик дозволяє встановлювати контрольні точки, надає команди трасування StepOver, StepInto, StepOut і вікна спостереження за змінними і ланцюжками викликів.

1.5 Загальна характеристика WinPcap

WinPcap – це популярний інструмент, що працює в середовищі Microsoft Windows, та дозволяє додаткам захоплювати і передавати мережеві пакети в обхід стека протоколів. Має такі додаткові функції, як фільтрацію пакетів на рівні ядра, движок статистики мережі і підтримку видаленого захоплення пакетів.

WinPcap складається з драйвера, який розширює операційну систему, забезпечуючи їй підтримку низькорівневого доступу до мережі, і бібліотеки, яка безпосередньо використовується для низькорівневого доступу. Ця бібліотека також містить у собі Windows-версію широко відомого libpcap, API для ОС Unix.

Завдяки своєму набору функцій, WinPcap є движком для захоплення і фільтрації пакетів, на якому ґрунтуються багато комерційних, а також безкоштовні мережеві інструменти, включаючи аналізатори протоколів, мережеві монітори, системи виявлення мережевого вторгнення, генератори трафіку і утиліти тестування мережі. Інструмент містить в собі всю необхідну документацію, а також різні керівництва.

В якості ключових особливостей та функцій програми можна виділити:

– висока ефективність: WinPcap реалізує всі можливості оптимізації, описані в літературі (наприклад фільтрація і буферизація ядра, часткова копія пакету та інше). Утиліта значно перевершує за швидкодією подібні програми;

– популярність: WinPcap використовується як мережевий інтерфейс для багатьох цілей: аналізатор протоколів, мережевий моніторинг, інструмент для виявлення мережевої активності, сніфер і т.д .;

– протестований і налагоджений: висока популярність значно допомогла у тестування програми. Утиліта практично не має помилок і багів;

– зручний і простий: WinPcap поширюється як невеликий виконуваний файл, який можна запустити на будь-якій операційній системі. Запустіть виконуваний файл і програма почне виконувати свої функції;

– мультиплатформеність: підтримує всі операційні системи сімейства Windows;

– документація: в програму включено детальне керівництво користувача з безліччю гіперпосилань. Також, в документацію включений підручник, який крок за кроком проведе через всі налаштування утиліти.

1.6 Технічне завдання на розробку

1.6.1 Вимоги до ПЗ для ПП-78

Програмне забезпечення спільно з апаратурою ПП-78 повинно забезпечувати:

– обмін повідомленнями з персональним комп’ютером (далі – ПК) по 100 MbitEthernet

– формування повідомлень з інформацією, прийнятою від ПК, з подальшою передачею через один з модемів (номер модему визначається в повідомленні від ПК);

– формування повідомлень з інформацією, прийнятоє від модемів, з

подальшою передачею в ПК;

– управління станом 10 незалежних каналів, 13 каналів з об’єднаним «загальним проводом», восьми каналів з об’єднаним «загальним проводом» для формування адреси КСТ згідно командам від ПК по Ethernet.

1.6.2 Вимоги до тестів для ПП-78

Для тестування модулів зв’язку (МСВ) і контролерів сигнальних точок (КСТ) повинні бути розроблені окремі тести з окремими технічними вимогами.

Для тестування самого ПП повинен бути розроблений тест ПП. Схема перевірки ПП наведена в додатку [Додаток А].

Тест ПП, що виконується на ПК, має складатися з двох перевірок: перевірка модемів, загальна перевірка ПП. При виконанні тесту повинні формуватися два протоколи: короткий і докладний.

У короткому протоколі повинні бути умови проведення тесту (температура, тиск, вологість та ін.), назва модуля, що перевіряється, версія ПЗ модуля, версія ПЗ пульта перевірки, час і дата початку проведення тестування, час та дата закінчення проведення тестування, виконувані перевірки, кількість виконаних циклів перевірок, кількість циклів з помилками, результат перевірки (придатний або не придатний), ПІБ перевіряючого.

У докладному протоколі до вмісту короткого протоколу повинні бути додані кожен етап виконання перевірки на кожному циклі тестування і повідомлення про помилки.

Алгоритм перевірки модемів може бути наступним:

1) надіслати в ПП повідомлення довжиною 30 байтів, що містить у першому байті F1h, а в інших порядковий номер байта в повідомленні, в перший модем і вивести повідомлення в протокол – «Відправка повідомлення в модем 1»;
2) очікувати повідомлення згідно з таблицею 5 від модему 2 протягом 100 ms:

– Якщо повідомлення не прийшло, в протокол виконання тесту записати – «Помилка. Немає відповіді від ПП-78 «та закінчити виконання перевірки;

– Якщо прийшло повідомлення з помилками – в протокол вивести помилки з повідомлення, починаючи повідомлення зі слова «Помилка.», І закінчити виконання перевірки;

– Якщо прийшло повідомлення з даними, надісланими на кроці 1, в протокол виконання тесту записати – «Модем 2 прийняв повідомлення»;

– Якщо прийшло повідомлення з даними, що не збігаються з надісланими на кроці 1, в протокол виконання тесту записати – «Помилка. Модем 2 прийняв помилкове повідомлення» і закінчити виконання перевірки;

3) надіслати в ПП повідомлення довжиною 30 байтів, що містить у першому байті F2h, а в інших порядковий номер байта в повідомленні, у другій модем і вивести повідомлення в протокол – «Відправка повідомлення в модем 2»;

4) очікувати повідомлення від модему 1 протягом 100 ms:

– Якщо повідомлення не прийшло, в протокол виконання тесту записати – «Помилка. Немає відповіді від ПП-78 «та закінчити виконання перевірки;

– Якщо прийшло повідомлення з помилками – в протокол вивести помилки з повідомлення, починаючи повідомлення зі слова «Помилка.», І закінчити виконання перевірки;

– Якщо прийшло повідомлення з даними, надісланими на кроці 1, в протокол виконання тесту записати – «Модем 1 прийняв повідомлення»;

– Якщо прийшло повідомлення з даними, що не збігаються з надісланими на кроці 1, в протокол виконання тесту записати – «Помилка. Модем 1 прийняв помилкове повідомлення «і закінчити виконання перевірки;

Режим «Наладка» повинен забезпечувати можливість передачі повідомлення користувача (наприклад, з файлу або такого, що вводиться у вікні в шістнадцятковому вигляді) через будь-який з модемів в покроковому і циклічному режимі. Повинна бути можливість зміни часу циклу від мінімально можливого до 5 s.

Режим «Наладка» повинен забезпечувати відображення прийнятих і переданих через модеми повідомлень на моніторі ПК в шістнадцятковому вигляді у вікні з прокруткою. Кожне повідомлення повинно нумеруватися.

Повинна бути можливість управління всіма дискретними виходами ПП.

Стан виходів ПП і помилки від ПП повинні відображати на моніторі ПК.

2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ТЕСТОВОГО ПЗ

ПП призначений для контролю якості контролерів сигнальних точок КСТ і модулів зв'язку, а в подальшому і інших виробів при серійному виробництві.

За ступенем апробованості розроблене програмне забезпечення відноситься до вперше розробляємого.

ПО спільно з апаратурою ПП не має небезпечних відмов і, відповідно, вимоги до функціональної безпеки не пред'являються.

Поле «Контрольна сума Ethernet-кадру» складається з чотирьох байтів і містить контрольну суму кадру (FCS, FrameCheckSequence), обчислену за алгоритмом CRC-32. При обчисленні CRC-32 використовується утворюючий поліном:

x^32+x^26+x^23+x^22+x^16+x^12+x^11+x^10+x^8+x^7+x^5+x^4+x^2+x+1.

При отриманні кадру контролер Ethernet виконує обчислення контрольної суми для цього кадру і порівняння отриманого по інтерфейсу значення з обчисленим значенням контрольної суми і, таким чином, визначає, чи не викривленийотриманий кадр (контролер Ethernet ПП налаштований на режим «викидання» перекручених кадрів).

MAC-адреса контролера Ethernet ПП: 0x02, 0x00, 0x00, 0x00, 0x03, 0xFF;

Структура поля «Заголовок повідомлення» приведена в таблиці 2.1.

Таблиця 2.1 - Структура поля «Заголовок повідомлення»

	Байти Ethernet кадру	Найменування поля	Кількість байт	Значення байт поля (hex)						
				0 (мол.байт)	1	2	3	4	5	6
Заголовок повідомлення (18 байтів)	22 - 23	Довжина Ethernetкадру, (К-21 до 1490 байт)	2							
	24 - 30	Ідентифікатор Ethernet-кадру	7	crc_h	Номер повідомлення		crc_i		0	
	31	Адресаодержувача повідомлення	1	x						
	32	Керуючий байт повідомлення	1	0x89						
	33	Адреса джерела повідомлення	1	y						
	34 - 35	Довжина даних (1-1450)	2							
	36 - 39	Ідентифікатор повідомлення	4	Kup• 10	Kui• 10	0	0			

Заголовок повідомлення включає такі поля:

– довжина інформаційної частини Ethernet-кадру (байти 22-23). Значення поля не більш, ніж 1468 байтів (18 байт заголовка і не більше 1450 байт даних);

– ідентифікатор Ethernet-кадру, який включає наступні елементи:

1) crc_h (байт 24 згідно таблиці 2.3) – контрольнасума заголовка повідомлення (15 байтів заголовка, від 25 до 39 включно згідно з таблицею 2.3). Контрольна сума заголовка повідомлення включає і байти контрольної суми даних повідомлення (crc_i). Тому при видачі повідомлення спочатку формується контрольна сума даних crc_i, а потім - контрольна сума заголовка повідомлення. При прийомі повідомлення спочатку виконується підрахунок і перевірка контрольної суми заголовка повідомлення, а потім при позитивних результатах перевірки контрольної суми заголовка повідомлення - підрахунок і перевірка контрольної суми даних повідомлення;

2) номер повідомлення (байти 26-27 згідно з таблицею 2.3) збільшується на одиницю для кожного чергового повідомлення (0, 1, 2, ..., 65535, 0, 1, 2, ...).

Використовується для контролю фактів втрат повідомлень;

3) crc_i (байти 27-28 згідно з таблицею 2.3) - контрольна сума даних повідомлення (області «Дані повідомлення» відповідно до таблиці 1). Байти 29, 30 приймають значення 0x00, 0x00;

– адреса одержувача повідомлення x (байт 31 в таблиці 2.3), який дорівнює у разі повідомлення в ПП 0xFF, а у разі повідомлення від ПП - адресою джерела повідомлення у, у прийнятому ПП повідомленні;

– керуючий байт повідомлення (байт 32 в таблиці 2.3) дорівнює 0x89;

– адреса джерела повідомлення у (байт 33 в таблиці 2.3), це адреса абонента мережі Ethernet НУ першого рівня, в повідомленнях від ПП дорівнює 0xFF;

– «Довжина даних повідомлення» (байти 34-35 в таблиці 2.3), в байтах, від 1 до 1450;

– «Ідентифікатор повідомлення» (байти 36-39 в таблиці 2.3). Це резервне поле, яке може виявитися корисним для використання в інструментальних програмах імітації обмінів ПК - ПП.

У повідомленнях до ПП від ПК:

– байт 36 приймає значення 0x80 (kup = 0x8);

– байт 37 приймає значення 0x80 (kui = 0x8) у повідомленнях, сформованих ПК;

– байти 38, 39 приймають значення 0x00, 0x00.

У повідомленняхвід ПП до ПК:

– байт 36 приймає значення, отримане в байті 37 повідомлення від ПК;

– байт 37 приймає значення 0x80 (kui = 0x8);

– байти 38, 39 приймають значення 0x00, 0x00.

Контрольна сума заголовка crc_h обчислюється згідно з алгоритмом CRC8 (поліном x ^ 8 + x ^ 5 + x ^ 4 + 1). При підрахунку контрольної суми заголовка враховуються 15 байтів заголовка (байти 25 - 39 згідно з таблицею 2.3). Алгоритм обчислення контрольної суми наступний:

– ініціалізується початкове значення crc_h, crc_h = 0xFF;

– crc_h послідовно перераховується з урахуванням значень байтів заголовка (15 байтів, починаючи з поля «номер повідомлення") за такою формулою

$$crc\_h = Tab8CRC [crc\_h \wedge *buf\_h ++], \quad (1)$$

де buf_h – байт заголовка, що враховується під час підрахунку контрольної суми.

Таблицю для підрахунку контрольної суми CRC8 наведено у лістингу програми [Додаток Б, рядки 3-36].

Якщо виникла помилка контрольної суми заголовка повідомлення crc_h, то повідомлення вважається помилковим і ігнорується.

Контрольна сума інформаційної частини повідомлення crc_i обчислюється комбінованим методом з використанням алгоритму CRC8 (поліном x ^ 8 + x ^ 5 + x ^ 4 + 1) і алгоритму CRC16 (поліном x ^ 16 + x ^ 12 + x ^ 5 + 1). При підрахунку контрольної суми інформаційної частини повідомлення враховуються байти 40 - K відповідно до таблиці 2.1.

Суть алгоритму полягає в наступному:

– інформаційна частина повідомлення розбивається на фрагменти довжиною по 15 байт, останній фрагмент може мати довжину менше 15 байт;

– для кожного фрагмента підраховується контрольна сума згідно з алгоритмом CRC8 (поліном x ^ 8 + x ^ 5 + x ^ 4 + 1);

– контрольна сума інформаційної частини повідомлення обчислюється згідно з алгоритмом CRC16 методом підсумовування CRC8-контрольних сум фрагментів повідомлення.

Таблицю для підрахунку контрольної суми CRC16 наведено у лістингу програми [Додаток Б, рядки 37-70].

Якщо виникла помилка контрольної суми інформаційної частини повідомлення crc_i, то повідомлення вважається помилковим і ігнорується.

ПП аналізує повідомлення від ПК і формує при виявленні помилок повідомлення згідно таблиці 2.2. При виявленні помилок в повідомленні ПП не приймає інформацію повідомлення до виконання.

Таблиця 2.2 – Форматповідомлення від ПП в ПК при виявленні помилок

Байти Ethernet-кадру	Призначення бітів							
	7	6	5	4	3	2	1	0
40	Addr = 255							
41	0							
42	TypeMes = 4							
43	0							
44	ER_CRCI	ER_L	ER_KUI_2	ER_KUI_2	ER_KUI_1	ER_CTRLB	ER_KUP	ER_CRCH
45	0							
46	0					ER_TYPEMES	ER_RES	ER_ADDR
47	0							
48	0					ER_LUPR	ER_OVER_M2	ER_OVER_M1
49...64	0							

Примітка

1 Addr – адреса ПП, дорівнює 0xFF.

2 TypeMes - тип повідомлення. Для повідомлення з помилками TypeMes = 4.

4 ER_CRCH - ознака помилки контрольної суми заголовка crc_h (байт 24 таблиці 3). 1 - є помилка, 0 - ні.

5 ER_KUP - ознака помилки ідентифікатора повідомлення (36 байт таблиці 3). 1 - є помилка, 0 - ні.

6 ER_CTRLB - ознака помилки керуючого байта повідомлення (байт 32 в таблиці 3). 1 - є помилка, 0 - ні.

7 ER_KUI_1 - ознака помилки молодшої частини ідентифікатора повідомлення (37 байт таблиці 3). 1 - є помилка, 0 - ні.

8 ER_KUI_2 - ознака помилки старшої частини ідентифікатора повідомлення (37 байт таблиці 3). 1 - є помилка, 0 - ні.

9 ER_L - помилка довжини повідомлення (байти 22 і 23 таблиці 3). 1 - є помилка, 0 - ні.

10 ER_CRCI - помилка контрольної суми даних повідомлення crc_i (байти 27-28 згідно з таблицею 3). 1 - є помилка, 0 - ні.

11 ER_ADDR - помилка адреси ПП (40 байт повідомлення). 1 - є помилка, 0 - ні.

12 ER_RES - помилка адреси ПП (41 байт повідомлення). 1 - є помилка, 0 - ні.

13 ER_TYPEMES - помилка типу повідомлення (42 байт повідомлення). 1 - є помилка, 0 - ні.

14 ER_OVER_M1 - переповнення черги повідомлень в модем 1. 1 - є помилка, 0 - ні.

15 ER_OVER_M2 - переповнення черги повідомлень в модем 1. 1 - є помилка, 0 - ні.

16 ER_LUPR - помилка довжини повідомлення з TypeMes = 3.. 1 - є помилка, 0 - ні.

Для тестування розроблені окремі тести з окремими технічними вимогами.Тест ПП, що виконується на ПК, складається з двох перевірок: перевірка ключів, перевірка модемів. При виконанні тесту формуються два протоколи: короткий і докладний.

Режим «Наладка» забезпечує можливість передачі користувальницького повідомлення (з файлу або вводиться у вікні в шістнадцятковому вигляді) через будь-який з модемів в покроковому і циклічному режимі. Забезпечена можливість зміни часу циклу від мінімально можливого до 5 s.

1.7 Бібліотека Qt мови C++

Сьогодні практично неможливо уявити собі додаток, що не володіє інтерфейсом користувача. Поняття Software і GUI (GraphicalUserInterface) нерозривно пов’язані один з одним. Бібліотеки для створення інтерфейсу користувача застосовуються в багатьох операційних системах (ОС), починаючи з Motif для ОС UNIX і закінчуючи широко відомою MFC (Microsoft FoundationClasses) від Microsoft для ОС Windows.

Хоча Windows API (ApplicationProgrammingInterface, інтерфейс програмування додатків) володіє всім необхідним для створення графічного інтерфейсу користувача, використання цих доступних «інструментів» вимагає великих витрат часу і практичного досвіду. Навіть бібліотека MFC, покликана полегшити процес написання програм для ОС Windows, не дає тієї простоти і легкості в процесі створення програм, як хотілося б. Тому і сьогодні розробники, як і раніше, витрачають масу часу на реалізацію інтерфейсу користувача. Але найбільший недолік, пов’язаний із застосуванням цих бібліотек – це платформо залежність.

Платформонезалежна реалізація програм – це майбутнє програмної індустрії. З кожним днем вона буде набувати все більш зростаюче значення.

Виграш же від реалізації платформонезалежних додатків очевидний: значно скорочується час розробки, тому щопри цьому немає необхідності писати код двічі, і що не менш важливо, відпадає необхідність знати специфіку кожної з платформ, для якої пишеться програма [9].

Qt – відмінне рішення для програмістів, які пишуть на мові C++, змушених зараз виконувати подвійну роботу з реалізації своїх додатків для ОС Windows, Linux і Mac OS X. Вибір на користь Qt позбавить вас від цих проблем. Qt надає підтримку великого числа операційних систем: Microsoft Windows (Windows XP / Vista / Windows 7), Mac OS X, Linux, Solaris, AIX, Irix, NetBSD, OpenBSD, HP-UX, FreeBSD та інші клони UNIX з X11. Більш того, завдяки вбудованості пакету QtEmbedded – всі можливості Qt доступні також і в інтегрованих системах (EmbeddedSystems). Qt використовує інтерфейс API низького рівня, що дозволяє додаткам працювати настільки ж ефективно, як і додатки, розроблені спеціально для конкретної платформи. Є також як офіційні, так і неофіційні портування бібліотеки Qt для мобільних платформ MeeGo, WebOS, iPhone і Android.

Незважаючи на те, що платформо незалежність є однією з найбільш привабливих можливостей бібліотеки, багато розробників використовують Qt також для створення додатків, що працюють тільки на одній платформі. Роблять вони це з тих міркувань, що їм подобається інструментарій та ідейний підхід самої бібліотеки. Подібний підхід надає їм додаткову гнучкість. Зважаючи на те, що вимоги до програмного продукту з плином часу постійно піддаються змінам, при появі необхідності пристосувати продукт для якої-небудь ще платформи це не складе великої складності.

Використання в розробці різних компіляторів C ++ ще більше підвищує правильність коду ваших програм, оскільки попереджувальні повідомлення ви будете отримувати від різних компіляторів.

Для прискорення та спрощення створення користувацьких інтерфейсів Qt надає програму QtDesigner, що дозволяє робити це в інтерактивному режимі.

На сьогоднішній день Qt – це продукт, який широко використовується розробниками усього світу. Компаній, що використовують цю бібліотеку, більше 4000. З числа деяких активних користувачів Qt можна назвати такі відомі компанії, як: Adobe, AT & T, Cannon, HP, Bosch, Boeing, IBM, Motorola, NASA, NEC, Pioneer, Sharp, Siemens, Sony і Xerox та ін.

Ось кілька найяскравіших прикладів використання даної бібліотеки:

– редактор тривимірної графіки AutodeskMaya (www.autodesk.com) (рис. 2.1);

Рисунок 2.1 - Редактор тривимірної графіки AutodeskMaya

– інтернет-пейджерSkype (www.skype.com) компанії Microsoft, призначений для голосового зв’язку VoIP (VoiceOver IP), дзвінків на звичайні телефони і проведення відео конференцій через Інтернет (рис. 2.2);

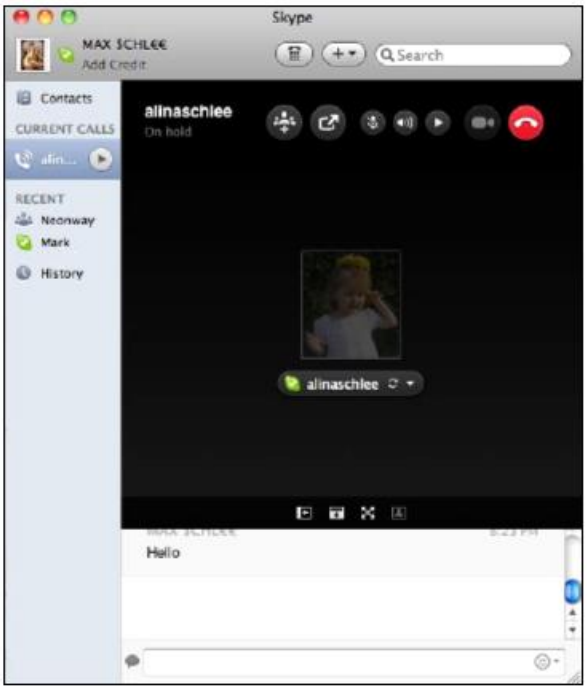

Рисунок 2.2 - Інтернет-пейджерSkype

– програма AdobePhotoshopAlbum (www.adobe.com) для обробки растрових зображень (рис. 2.3);

Рисунок 2.3 - Програма AdobePhotoshopAlbum

– вільний програвач VLC mediaplayer (www.videolan.org/vlc/), починаючи з версії 0.9;

– мережева карта світу GoogleEarth (earth.google.com), яка дозволяє розглядати цікавідля нас ділянки поверхні нашої планети з висоти до 200 м (рис.

2.4);

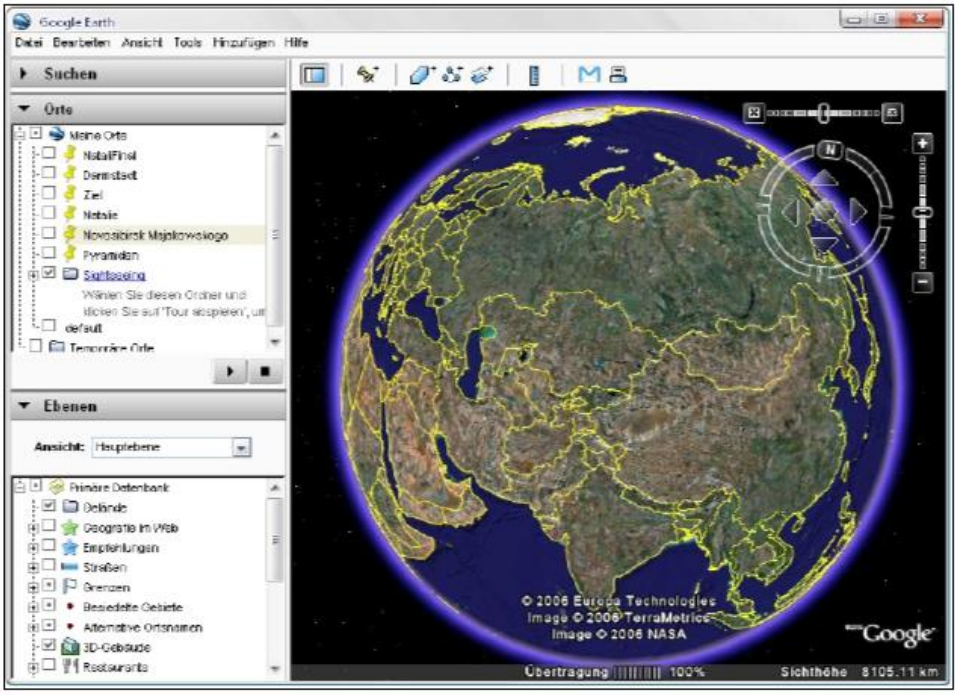

Рисунок 2.4 - Мережева карта світу GoogleEarth

– програма для віртуалізації операційних систем VirtualBox (www.virtualbox.org) від SunMicrosystems (рис. 2.5);

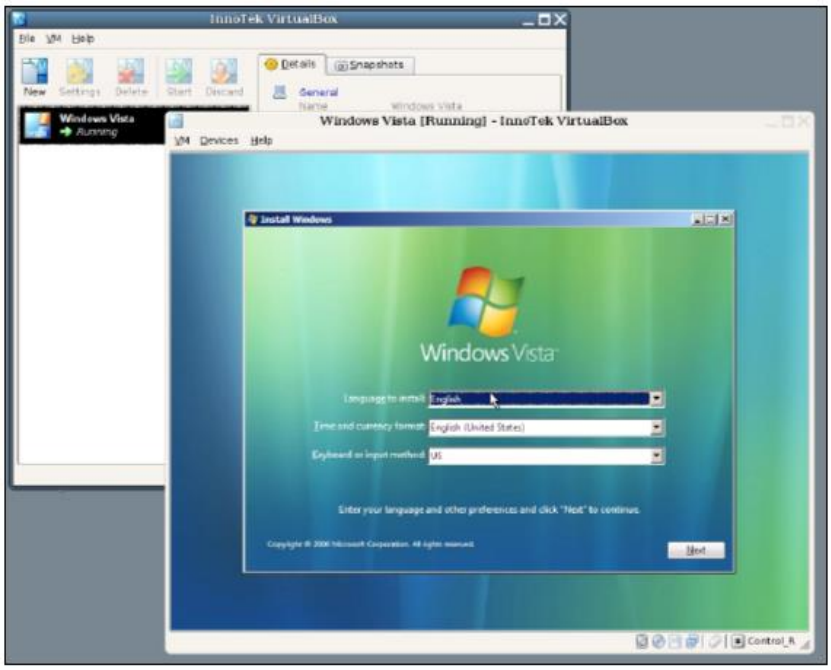

Рисунок 2.5 - Програма для віртуалізації операційних систем VirtualBox

Багато хто звик вважати, що Qt – це засіб для створення тільки інтерфейсу користувача. Це неправильно, Qt – це повний інструментарій для програмування. Цей інструментарій складається з окремих модулів і надає:

– підтримку двовимірної і тривимірної графіки (фактично, будучи стандартом для платформо незалежного програмування на OpenGL);
– можливість інтернаціоналізації, яка дозволяє значно розширити ринок збуту ваших програм;
– використання формату XML (eXtensibleMarkupLanguage);
– STL-сумісну бібліотеку контейнерів;
– підтримку стандартних протоколів введення-виведення;
– класи для роботи з мережею;
– підтримку програмування баз даних, включаючи підтримку Oracle, Microsoft SQL Server, IBM DB2, MySQL, SQLite, Sybase, PostgreSQL.

Qt – повністю об’єктно-орієнтована бібліотека. Нова концепція ведення між об’єктових комунікацій, іменована «сигнали і слоти», повністю замінює колишню, не цілком надійну модель зворотних викликів. Також є можливість обробки подій, наприклад натискання клавіш клавіатури, переміщення миші і т.д.

1.7.1 Протокол обміну даними між ПК та ПП

Обмін даними між ПК і ПП організовується у вигляді блоків даних. Динаміка обміну, а також структура повідомлень, визначаються повідомленнями, переданими з ПК в ПП і з зовнішнього пристрою в модеми.

Швидкість прийому / передачі по Ethernet повинна бути 100 Mbit / s.

Повідомлення по мережі Ethernet передаються Ethernet-кадрами. Загальна структура Ethernet-кадру для каналу зв’язку ПК – ПП повинна мати вигляд, наведений у таблиці 2.3.

Таблиця 2.3 – Загальна структура Ethernet-кадру для каналу зв’язку ПК – ПП

Найменування поля	**Байти**	**Призначення**
Заголовок Ethernet-кадру	0 – 21	Заголовок Ethernet-кадру (байти 0-7 формуються і обробляються апаратно контролерами Ethernet ПП і ПК, байти 8-21 формуються і обробляються програмними засобами доставки повідомлення по мережі Ethernet ПП, ПК)
Інформаційна частина Ethernet-кадру (повідомлення)	22 – 39	Заголовок повідомлення
	40 – K, де 40≤K<1490	Дані повідомлення
Контрольна сума Ethernet-кадру	(K+1) – (K+4)	Контрольна сума (формується і обробляється апаратно Ethernet-контролерами ПП і ПК)

Службові поля «Тема Ethernet-кадру», «Тема повідомлення», «Дані повідомлення» і «Контрольна сума Ethernet-кадру» призначені для апаратної й програмної ідентифікації та контролю прийнятого повідомлення:

– Поле «Тема Ethernet-кадру» формується відповідно до стандарту IEEE 802.3 CSMA / CD (ETHERNET). Поле містить преамбулу і початковий обмежувач кадру (байти 0-7), які формуються і обробляються апаратно контролерами Ethernet ПП і ПК. MAC-адреси (фізичні адреси) приймача і джерела і тип протоколу (байти 8-21) формуються і обробляються програмними засобами доставки повідомлення по мережі Ethernet ПП, ПК;

Заголовок Ethernet-кадру (байти 0-21) згідно зі стандартом 802.3 включає п’ять полів загальною довжиною 22 байта:

– поле преамбули кадру складається з семи синхронізуючих байтів 101010102. При манчестерському кодуванні ця комбінація представляється у фізичному середовищі періодичним хвильовим сигналом з частотою 5 MHz;

– початковий обмежувач кадру (SFD, Start-of-frame-delimiter) складається з одного байта 101010112. Поява цієї комбінації бітів є вказівкою на те, що наступний байт – це перший байт MAC-адреси одержувача;

– MAC-адресу одержувача, довжина шість байт;

– MAC-адресу джерела, довжина шість байт;

– тип протоколу, довжина два байта. Повинен бути рівний 0x1307.

MAC-адреси абонентів Ethernet-мережі повинні бути унікальні в межах даної мережі.

MAC-адресу контролера Ethernet ПП: 0x02, 0x00, 0x00, 0x00, 0x03, 0xFF;

– Поле «Заголовок повідомлення» формується і обробляється програмними засобами доставки повідомлення по мережі Ethernet ПП, ПК. Поле містить довжину інформаційної частини Ethernet-кадру, ідентифікатор кадру, логічні адреси приймача і джерела, керуючий байт повідомлення, довжину області даних та ідентифікатор повідомлення;

– поле «Дані повідомлення» призначене для передачі даних з ПК в ПП і назад;

– поле «Контрольна сума Ethernet-кадру» призначене для виявлення помилок при передачі повідомлення, формується і контролюється апаратно (контролерами Ethernet ПП і ПК). Поле складається з чотирьох байтів і містить контрольну суму кадру (FCS, FrameCheckSequence), обчислену за алгоритмом CRC-32.

ЧерезEthernet від персонального комп’ютера мікроконтролер (МК) ПП приймає дані управління виходами, інформацію для передачі через модеми в пристрій. У повідомленнях в персональний комп’ютер ПП передає дані від пристрою, що перевіряється і стан своїх виходів. Обмін даними між ПК і ПП організований у вигляді блоків даних.

В одному прийнятому ПП повідомленні по Ethernet для передачі інформації через один з модемів містяться дані для формування тільки одного повідомлення мережі перегону.

В одному прийнятому ПК повідомленні по Ethernet з інформацією, прийнятої одним з модемів ПП, містяться дані тільки одного повідомлення мережі.

Структура поля «Тема Ethernet-кадру» приведена в таблиці 2.4.

Таблиця 2.4 – Структура поля «Тема Ethernet-кадру»

	Байти	Найменування поля	Кількість байтів	Значення байтів поля (hex)						
				0 (молодшийбайт)	1	2	3	4	5	6
Заголовок Ethernet-кадру	0 – 6	Преамбула Кадру	7	10101010_2	-“-	-“-	-“-	-“-	-“-	-“-
	7	SFD	1	10101011_2						
	8 – 13	MAC-адреса одержувача	6	p0	p1	p2	p3	p4	p5	
	14 – 19	MAC-адреса джерела	6	g0	g1	g2	g3	g4	g5	
	20 – 21	Тип протоколу	2	07	13					

1.8 Структура програми тестування ПП-78

Під час запуску програми з’являється головне вікно програми «Диспетчер тестів».

Головне вікно програми містить наступні елементи:

1) рядок меню:

а) вибір Ethernet-адаптера (з переліком наявних адаптерів);

б) вибір протоколів для перегляду (коротких та детальних);

в) виклик довідки (перегляд інформації про програму, схеми підключення).

2) список для вибору тестів:

а) тест працездатності пристрою;

б) тест наладки;

в) кнопка для керування вибором тестів.

3) область відображення повідомлень: кількостізнайдених мережевих адаптерів, стан виконання тестування.

4) засоби для встановлення кількості циклівтестування.

5) Елементикерування програмою (кнопки «Виконати» та «Вихід»).

Зовнішній вигляд головного вікна приведено на рис.2.6.

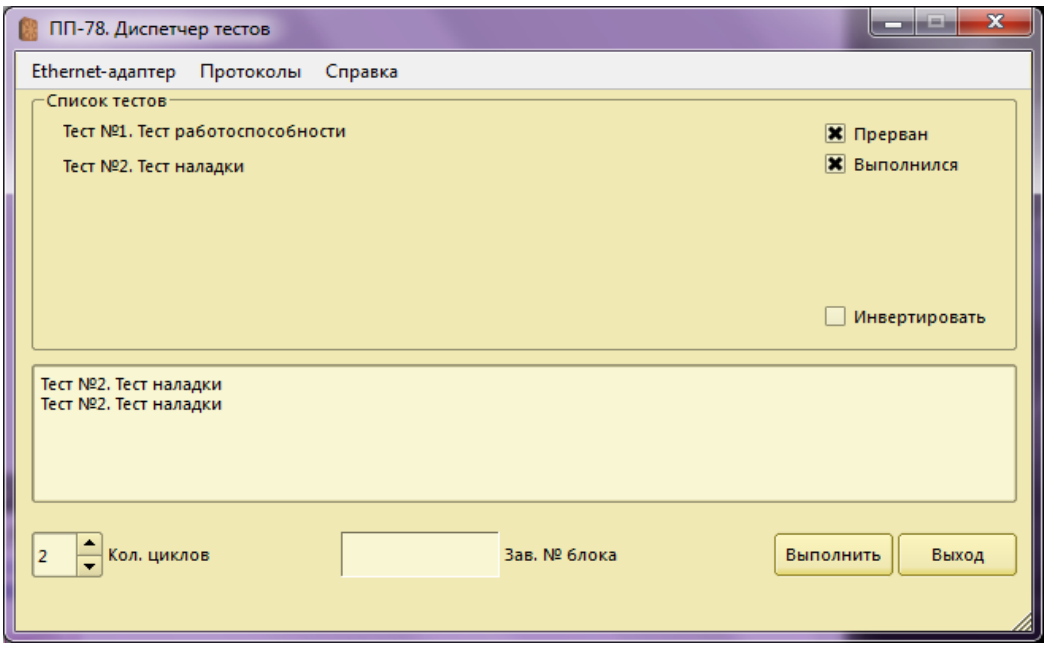

Рисунок 2.6 - Вікно диспетчеру тестів ПП-78

На початку виконання будь-якого з обраних тестів користувачеві на екран виводиться вікно зі схемою, котру необхідно зібрати для проведення тестування.

Зображення вікна приведено на рис.2.7.

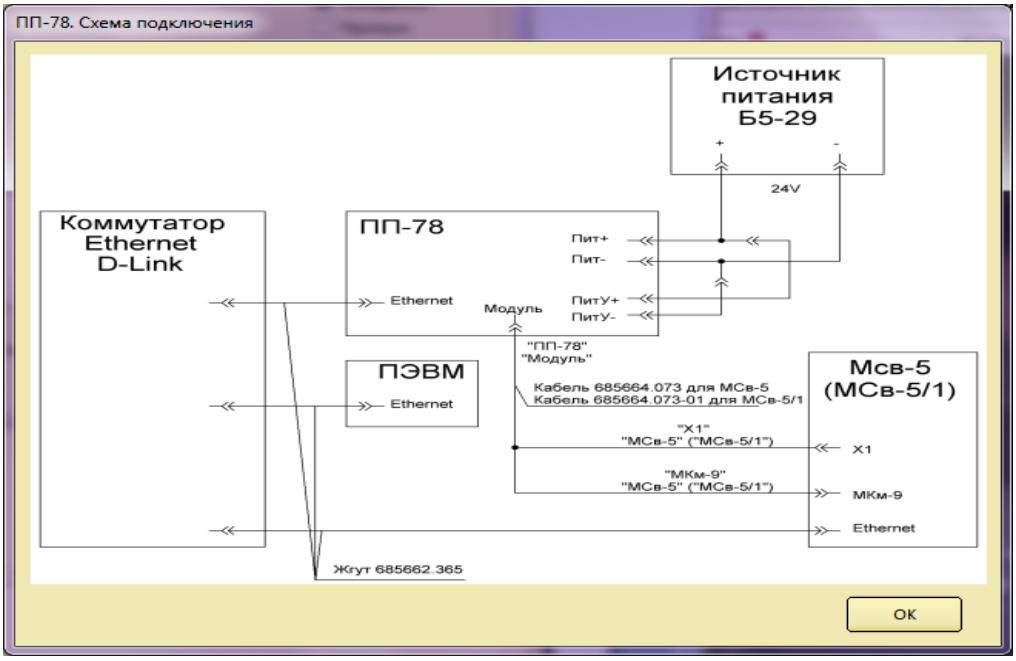

Рисунок 2.7 - Вікно зі схемою підключення

В головному вікні програми, поруч з назвою кожного тесту відображується поточний стан виконання тесту. Це може бути один із наступних станів:

– пропуск;

– очікування;

– виконання;

– виконаний;

– помилка;

– перерваний.

Тести виконано у вигляді динамічних бібліотек, що підключаються у міру потреби.

Після вибору необхідного тесту (тестів) починається виконання тестування, результати тестування заносяться у файл протоколу (короткого та детального) формату *.txt. Ім’я документу із протоколом дається наступним чином: «protocol PP-78 [дата] [час]». Таким чином, ім’я файлу протоколу завжди є унікальним.

Структура програми складається файлів, наведених у табл.2.5.

Таблиця 2.5. – Структура програми тестування

№ з/п	**Назва файлу**	**Опис файлу**
1	**2**	**3**
1	*defines.h*	Заголовний файл. Містить опис основних змінних.
2	*errors.h*	Заголовний файл. Містить опис ресурсів для перевірки та визначення помилок.
3	*message.h*	Заголовний файл. Містить опис ресурсів для відправлення та захвату повідомлень.
4	*protocol.h*	Заголовний файл. Містить опис ресурсів для складання протоколів тестування.
5	*frames.cpp*	Файл вихідного коду. Містить реалізацію введення повідомлення для режиму «Наладка».
Підпроект Debug_test		
6	*debug_test.pro*	Файл проекту підпрограми debug_test

Табл. 2.5. Продовження

1	2	3
7	*debug.h*	Заголовний файл. Містить опис ресурсів для режиму «Наладка».
8	*schema.h*	Заголовний файл. Містить опис ресурсів для виведення схеми підключення пристрою.
9	*debug.cpp*	Файл вихідного коду. Містить реалізацію режиму «Наладка».
10	*main.cpp*	Файл вихідного коду. Основна функція підпрограми.
11	*schema.cpp*	Файл вихідного коду. Містить реалізацію виведення схеми підключення пристрою.
Підпроект Diagnostic		
12	*diagnostic.pro*	Файл проекту підпрограми diagnostic
13	*about.h*	Заголовний файл. Містить опис ресурсів для виведення інформації про програму.
14	*diagnostic.h*	Заголовний файл. Містить опис ресурсів для реалізації тестування.
15	*map.h*	Заголовний файл. Містить опис ресурсів для схеми підключення.
16	*about.cpp*	Файл вихідного коду. Містить реалізацію виведення інформації про програму.
17	*diagnostic.cpp*	Файл вихідного коду. Містить реалізацію функцій тестування.
18	*main.cpp*	Файл вихідного коду. Основна функція підпрограми.
19	*map.cpp*	Файл вихідного коду.
Підпроект test_1_PP-78		
20	*test_1_PP-78.pro*	Файл проекту підпрограми test_1_PP-78
21	*test_1.h*	Заголовний файл. Містить опис ресурсів для виконання тесту модемів.
22	*main.cpp*	Файл вихідного коду. Основна функція підпрограми
23	*test_1.cpp*	Файл вихідного коду. Містить реалізацію тестування модемів.

Лістинг перелічених файлів наведено у додатках [Додаток Б].

1.9 Розроблення тесту працездатності модемів

Основні задачі тесту працездатності:

– відправити перевірочне повідомлення до модему;

– зчитати відправлене повідомлення;

– порівняти отримане повідомлення з еталонним;

– за отриманими результатами зробити висновки на сформувати протокол перевірки;

– при наявності помилок проаналізувати їх, занести до детального протоколу тестування.

Загальний алгоритм реалізації перевірки модемів можна представити наступним чином (рис.2.8).

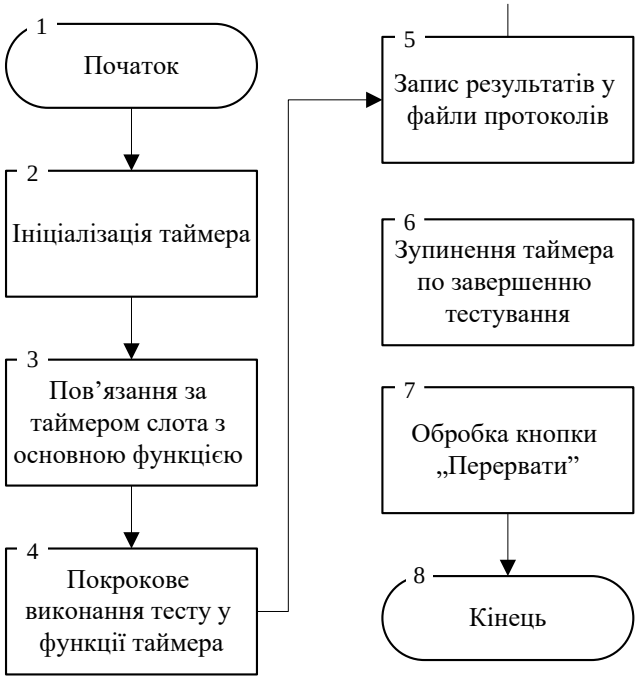

Рисунок 2.8 - Блок-схема алгоритму реалізації тесту працездатності

Загальна схема виконання тесту перевірки працездатності виглядає наступним чином (рис.2.9).

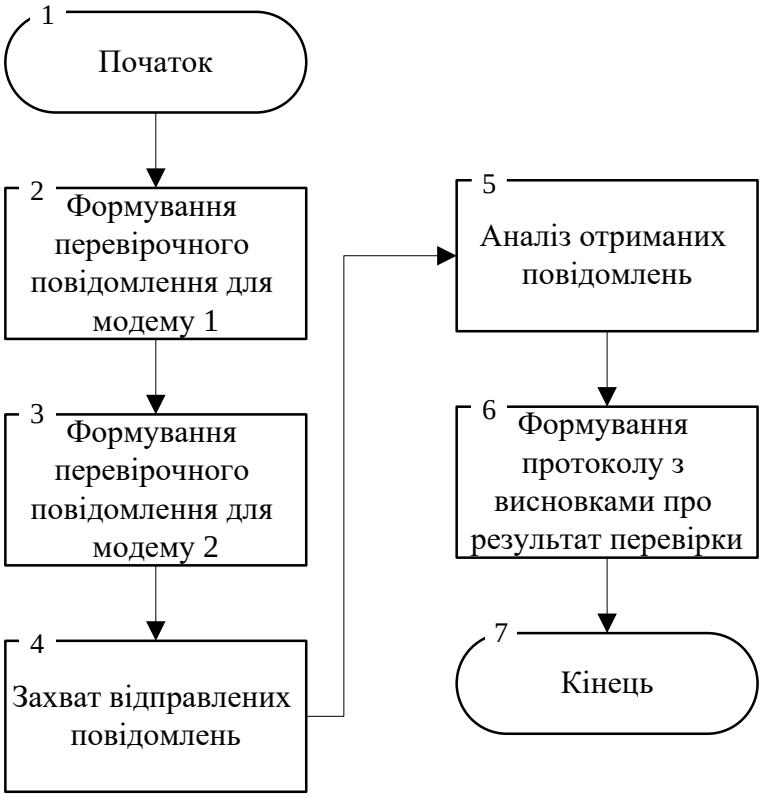

Рисунок 2.9 - Блок-схема тесту перевірки працездатності модемів

Вікно з відображенням прогресу виконання тесту перевірки модемів зображено на рис. 2.10.

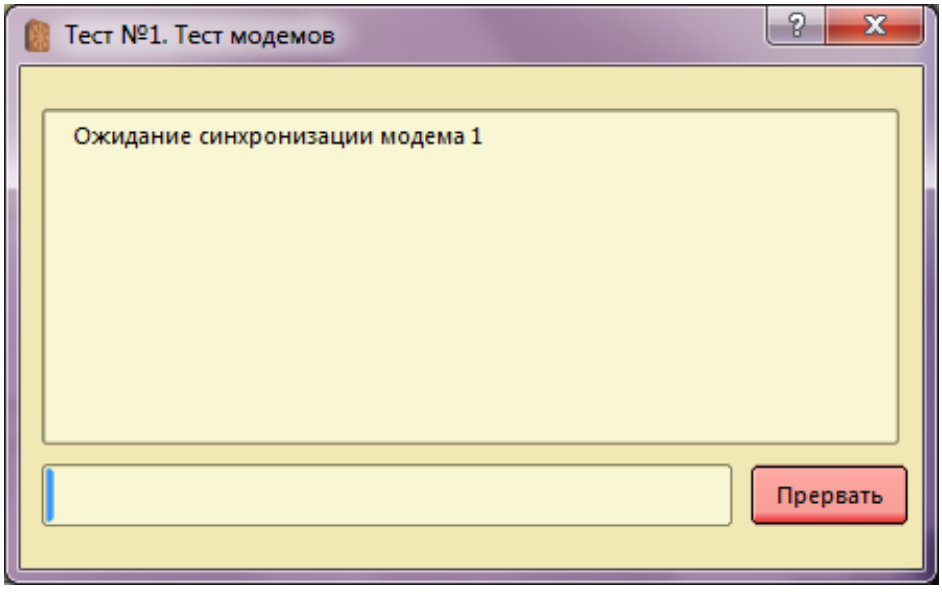

Рисунок 2.10 - Вікно тесту модемів

1.10 Розроблення тесту наладки

Тест наладки дає змогу передавати повідомлення користувача, яке може бути сформовано одним із наступних шляхів:

– зчитуванням з заздалегідь підготованого файлу;

– безпосереднім введенням повідомлення через діалогове вікно.

Вікно тесту наладки зображено на рис. 2.11.

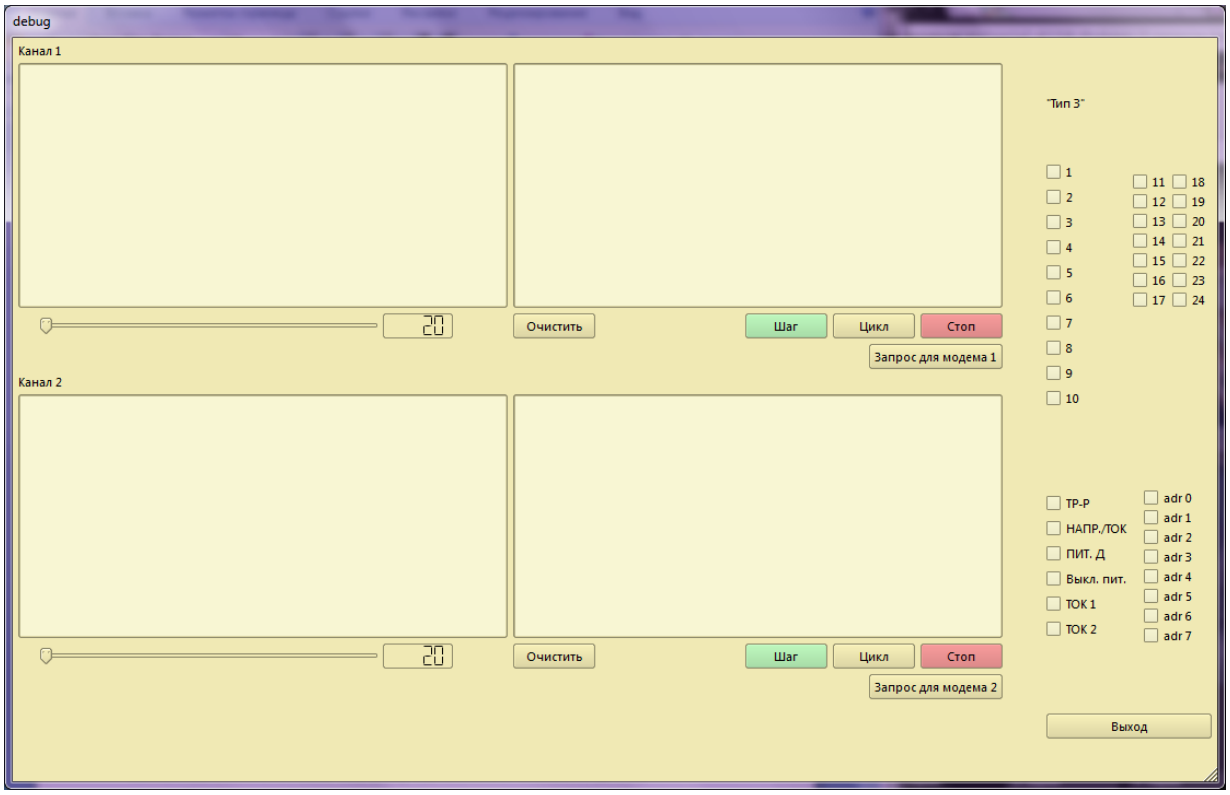

Рисунок 2.11 - Вікно тесту наладки

Загальна схема виконання тесту наладки приведена на рис.2.12.

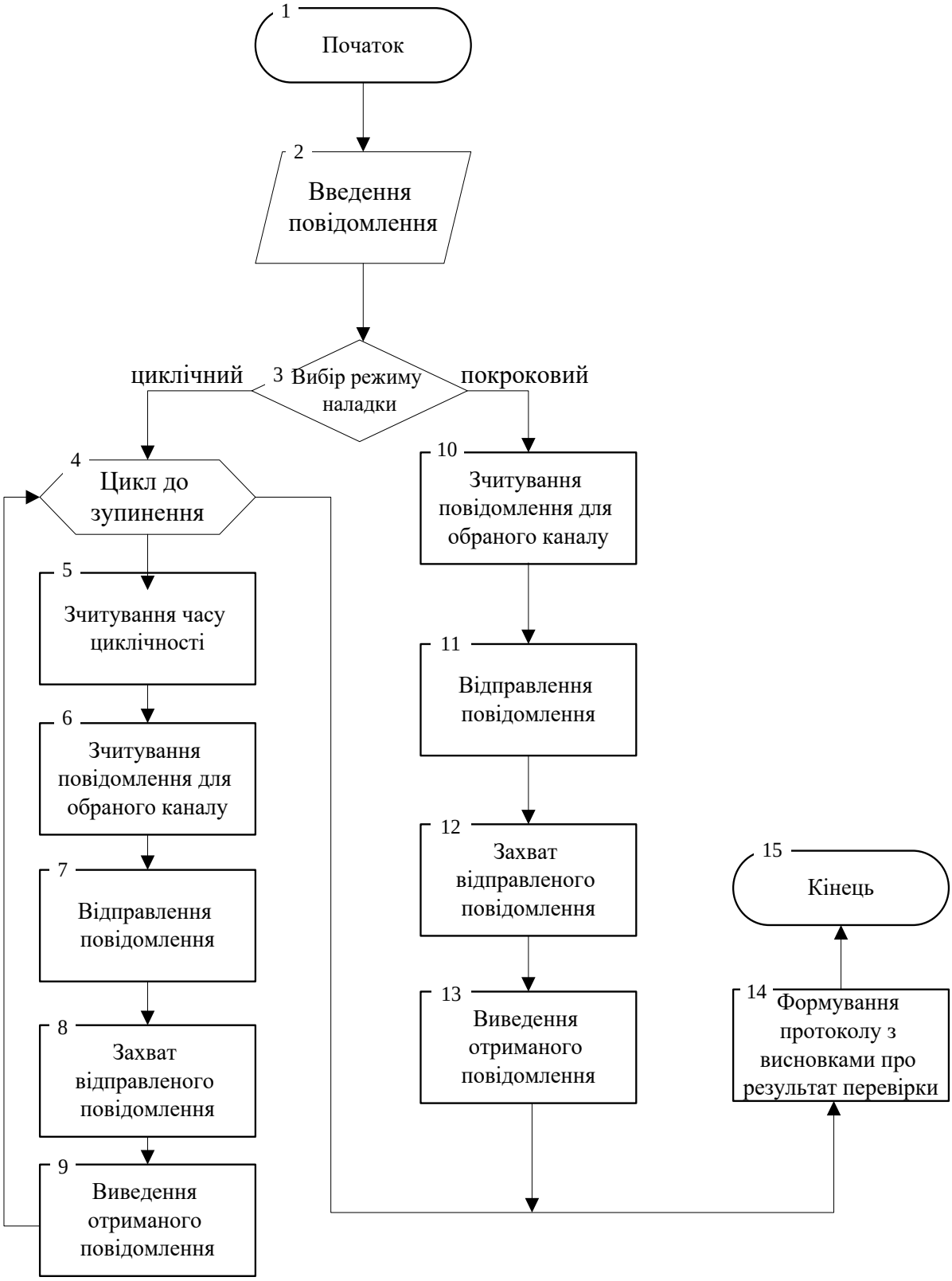

Рисунок 2.12 - Блок-схема тесту наладки

Форма вікна тесту наладки містить:

– поля для відображення отриманих / відправлених повідомлень для 2 каналів передачі;

– кнопки для керування процесом проведення тесту для кожного з каналів;

– бігунок для встановлення часу циклу передачі повідомлень;

– блок елементів керування для перевірки працездатності ключів - виконання тесту «Типу 3»;

– клавішу для виходу з тесту наладки.

Повідомлення формується у шістнадцятковому вигляді. Сформоване повідомлення передається через будь-який з модемів у покроковому і циклічному режимах. Також у програмі забезпечена можливість зміни часу циклу від мінімально можливого (20 мс) до 5 с.

1.11 Тестування розробленого програмного забезпечення

Тестування розробленого додатку проводилося на різних комп'ютерах.

Варто відмітити, що дана програма була розроблена виключно для використання у СНВО «Імпульс». А отже, з урахуванням особливостей налаштування програми, налаштувань фільтр повідомлень, що отримуються, для правильного функціонування програму необхідно використовувати на комп'ютерах, які приєднані до комп'ютерної мережі підприємства СНВО «Імпульс».

Текст короткого протоколу тестування працездатності модемів:

Протокол
Тип блока: ПП-78
Дата и время создания протокола: 12.05.2018 18:31:41
Заводской номер: _____
Условия проверки:
температура окружающего воздуха, град.С - _____
относительная влажность воздуха,% - _____
атмосферное давление,kPa - _____
прочие воздействия - _____

Список тестов:

Тест №1. Тест работоспособности
Тест №2. Тест наладки

Результат:
версия ПО пульта проверки ПП-78 -1
количество пройденных циклов 1 из 1

блок годен
Проверка ПП-78 завершена 12.05.2018 18:32:11
Проверяющий _____ _____ _____
(Ф.И.О.) подпись дата

Текст детального протоколу тестування працездатності модемів:

Протокол
Тип блока: ПП-78
Дата и время создания протокола: 12.05.2018 19:31:32
Заводской номер: _____
Условияпроверки:
температура окружающего воздуха, град.С - _____
относительная влажность воздуха,% - _____
атмосферное давление,kPa - _____
прочие воздействия - _____

Список тестов:
Тест №1. Тест работоспособности
Ожидание синхронизации модема 1
Модем 1 не синхронизирован
Тест прерван. 12.05.2018 19.32.02.954

Результат:
версия ПО пульта проверки ПП-78 -1
количество пройденных циклов 1 из 1

блок не годен
Проверка ПП-78 прервана 12.05.2018 19:32:03

Проверяющий _____ _____ _____
(Ф.И.О.) подпись дата

Налаштування фільтру WinPcардозволяє приймати пакети даних, які відправляються для тестування даного пристрою. Виконується перевірка джерела генерації повідомлень за mac-адресою. В результаті роботи програмного забезпечення, відправлені пакети можна відстежити, наприклад засобами програми WireShark.

В результаті проведеного тестування розроблене програмне забезпечення довело свою працездатність. Отримані результати проведеного тестування пристрою ПП-78 показали результат, що відповідає дійсності.

3 НОРМИ І ВИМОГИ ОХОРОНИ ПРАЦІ НА РОБОЧОМУ МІСЦІ

3.1 Загальні положення

Зростання ефективності праці та збереження здоров’я працівників під час професійної діяльності залишаються актуальною проблемою і одним із шляхів її вирішення є підвищення, як соціальної відповідальності держави, так і особистої відповідальності працівників за безпеку праці.

Основним завданням розділу "Охорона праці" є розробка технічних, санітарно-гігієнічних і організаційних заходів, спрямованих на усунення причин виробничого травматизму, професійної захворюваності, підвищення продуктивності праці.

До роботи з персональними ЕОМ і зовнішніми пристроями ПК допускаються особи, які пройшли спеціальне навчання, медичний огляд, перевірку знань вимог даної інструкції і питань пожежної безпеки, а також інструктажі в установленому на підприємстві порядку.

Заборонено доступ до роботи осіб, які не пройшли навчання, інструктаж і перевірку знань з охорони праці, медичний огляд (ст. 17, ст. 18 Закону України «Про охорону праці»)

Оператори і користувачі ПК зобов'язані:

– знати пристрої і правила безпечної експлуатації комп'ютерної техніки;

– проходити в установленому порядку періодичні медичні огляди;

– виконувати роботу відповідно до завдання безпосереднього керівника;

– не допускати в робочу зону сторонніх осіб;

– при роботі з Інтернетом користуватися тільки ресурсами мережі, необхідними для виконання поставленого виробничого завдання;

– відкривати вкладення в листи від невідомих користувачів

– не робити завантаження файлів, не перевіривши їх на наявність вірусів;

– знати і виконувати вимоги Закону України "про авторські та суміжні права";

– знати місця розташування первинних засобів пожежогасіння та вміти ними користуватися;

– виконувати правила внутрішнього трудового розпорядку, які регулюються Законом про охорону праці;

– дотримуватися правил особистої гігієни.

3.2 Аналіз стану умов праці

Для створення системи автоматизації збору інформації достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп’ютером.

Оформлення дипломного проекту з розробки системи автоматизації збору інформації за фізичним навантаженням відноситься до категорії легкі роботи (Іа), її виконують сидячи з періодичним ходінням. Щодо характеру організування виконання дипломної роботи, то він підпадає під нав’язаний режим, оскільки певні розділи роботи необхідно виконати у встановлені конкретні терміни. За ступенем нервово-психічної напруги виконання роботи можна віднести до ІІ – ІІІ ступеня і кваліфікувати як помірно напружений – напружений за умови успішного виконання поставлених завдань.

Під час виконання робіт використовують ПК та периферійні пристрої (лазерні та струменеві), що призводить до навантаження на окремі системи організму. Такі перекоси у напруженні різних систем організму, що трапляються під час роботи з ПК, зокрема, значна напруженість зорового аналізатора і довготривале малорухоме положення перед екраном, не тільки не зменшують загального напруження, а навпаки, призводять до його посилення і появи стресових реакцій.

Найбільшому ризику виникнення різноманітних порушень піддаються:

органи зору, м’язово-скелетна система, нервово-психічна діяльність, репродуктивна функція у жінок.

Тобто наявні психофізіологічні небезпечні та шкідливі фактори:

- фізичного перевантаження:
    1) статичного;
    2) динамічного;
- нервово-психічного перевантаження:
    1) розумового перенапруження;
    2) монотонності праці;
    3) перенапруження аналізаторів;
    4) емоційних перевантажень.

Рекомендовано застосування екранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів захисту, а також інші профілактичні заходи [5]. Роботу за дипломним проектом визнано, таку, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви тривалістю 15 хвилин через кожну годину роботи.

3.3 Аналіз потенційних небезпечних і шкідливих виробничих факторів при роботі з персональним комп'ютером

Основними характеристиками персонального комп'ютера є наступні:

- робоча напруга U = + 220 + -5%;
- робочий струм I = 2А;
- споживана потужність P =-350 Вт.

Роботу користувача розробленої підсистеми слід віднести до категорії Іа (легкі фізичні роботи). До даної категорії відносяться всі види діяльності, які виконуються сидячи, з періодом ходінням, і не потребують фізичного напруження

[6].

Згідно з [3] при експлуатації даного програмного продукту існують такі небезпечні і шкідливі виробничі фактори:

- фізичні:
    1) підвищений рівень напруги електричної мережі;
    2) підвищена або знижена рухомість та вологість повітря;
    3) підвищений рівень статичної електрики;
    4) відсутність або нестача природного світла;
    5) підвищений рівень електромагнітного випромінювання;
- психофізіологічні:
- фізичні перевантаження:
    1) статичні;
    2) динамічні;
- нервово-психічні перевантаження:
    1) розумове перенапруження;
    2) монотонність праці;
    3) емоційні перевантаження.

3.4 Заходи з охорони праці

3.4.1 Організація робочого місця з ПК

Площа на одне робоче місце з ВДТ або ПК не менше 6,0 кв. м, об'єм - не менше 20,0 куб. м. Приміщення має природне і штучне освітлення. При розміщенні робочих місць включена можливість прямих відблисків від джерел природного та штучного освітлення. Робочі місця розташовувані так, щоб природне світло падало з лівого боку.

Екран відеомонітора розташован на оптимальній відстані від очей користувача, близько 600 мм. Клавіатура розташована на поверхні столу на відстані 100-300 мм від краю, зверненого до користувача.

Забороняється палити в приміщенні, в якому розташовані ПК і її зовнішні пристрої.

3.4.2 Електробезпека

Основним небезпечним фактором при роботі з ЕОМ є небезпека ураження людини електричним струмом, яка посилюється тим, що органи чуття людини не можуть на відстані виявити наявність електричної напруги на обладнанні.

Тяжкість ураження людини електричним струмом залежить від цілого ряду чинників:

- значення сили струму;
- електричного опору тіла людини і тривалості протікання струму в тілі;
- типу і частоти струму;
- індивідуальних властивостей людини і навколишнього середовища.

Приміщення для ЕОМ відноситься до приміщень без підвищеної небезпеки, тобто в приміщення, в яких відсутні умови, що створюють підвищену або особливу небезпеку.

Електробезпека забезпечується:

- відповідною конструкцією електроустановок;
- застосуванням технічних способів і засобів захисту;
- організаційними і технічними заходами.

Основними технічними способами і засобами захисту від ураження електричним струмом, що використовуються окремо або в поєднанні один з одним, є:

- захисне заземлення;
- занулення;
- мала напруга;
- захисне відключення;

– ізоляція струмоведучих частин;

– попереджувальна сигналізація, блокування, знаки безпеки;

3.4.3 Розрахунок захисного заземлення

Основними технічними способами і засобами захисту від ураження електричним струмом, що передбачаються в даному дипломному проекті, є:

– захисне заземлення,

– занулення,

– захисне відключення,

– ізоляція струмоведучих частин.

Завдання захисного заземлення - усунення небезпеки ураження струмом у випадку дотику до корпусу та інших струмоведучих металевих частин електроустановок, які опинилися під напругою.

Розрахунок заземлюючого контуру виконується виходячи з умови:

$$R_{\text{ЗУ}} = \frac{R_{\text{З}} \cdot R_{\text{П}}}{R_{\text{П}} \cdot n \cdot \eta_{\text{З}} + R_{\text{З}} \cdot \eta_{\text{П}}} \leq 4 \text{ Ом}, \quad (3.1)$$

де $R_З$ - опір заземлювача (стержня, труби, куточка і т.д.), Ом;

$R_П$ - Опір лінії, що з'єднує заземлювачі, Ом;

n - кількість заземлювачів;

$\eta_З$ і $\eta_П$ - Коефіцієнти екранування відповідно заземлювача і з'єднує смуги ($\eta_З$= 0,2 ÷ 0,9; $\eta_П$= 0,1 ÷ 0,7).

Опір заземлювача розраховується за формулою 4.2

$$R_{\text{З}} = \frac{\rho}{2 \cdot \pi \cdot l}\left(In\frac{2 \cdot l}{d} + \frac{1}{2} \cdot In\frac{4 \cdot t + l}{4 \cdot t - l}\right) \quad (3.2)$$

де ρ - питомий опір ґрунту (взяти з довідкової літератури);

l - довжина заземлювача (для труб 2-3 м, для стрижнів до 10 м), м;

d - діаметр заземлювача (для стрижнів 0,01-0,03 м, для труб 0,03-0,05 м);

t - відстань від середини забитого в ґрунт заземлювача до рівня землі, м.

Розрахуємо опір заземлювача:

$$R_з = \frac{60}{2 \cdot \pi \cdot 3}\left(\ln\frac{2 \cdot 3}{0.03} + \frac{1}{2} \cdot \ln\frac{4 \cdot 1 + 3}{4 \cdot 1 - 3}\right) = 19{,}96 \qquad (3.3)$$

Опір лінії, що з'єднує заземлювачі розраховується за формулою 4.4

$$R_п = \frac{\rho}{2 \cdot \pi \cdot l} \cdot \ln\frac{2 \cdot L^2}{b \cdot t}, \qquad (3.4)$$

де L - довжина лінії, що з'єднує заземлювачі (при контурному заземленні вона приблизно дорівнює периметру виробничої будівлі), м;

b - ширина смуги (0,03 - при прокладанні всередині будівлі і 0,05 - при прокладанні поза будівлею), м;

t - глибина заземлення від рівня землі (0,5 м.).

Розрахуємо опір лінії, що з'єднує заземлювачі

$$R_П = \frac{60}{2 \cdot \pi \cdot 3} \cdot In\frac{2 \cdot 50^2}{0.03 \cdot 0.5} = 14.37, \qquad (3.5)$$

Необхідна кількість заземлювачів, розраховується за формулою 4.6

$$n = \frac{2 \cdot R_з}{4 \cdot \eta_з}, \qquad (3.6)$$

де 4 - допустимий загальний опір;

2 - коефіцієнт сезонності.

Розрахуємо необхідну кількість заземлювачів,

$$n = \frac{2 \cdot 19,9}{4 \cdot 0.5} = 19,9 \approx 20 \quad , \qquad (3.7)$$

Округлимо результат в більшу сторону і отримуємо необхідну кількість заземлювачів - 20. Маючи всі необхідні дані розрахуємо опір заземлюючого контуру.

$$R_{зу} = \frac{19,96 \cdot 14,37}{14,37 \cdot 20 \cdot 0,5 + 19,96 \cdot 0,4} = 1,89 \leq 4 \quad Ом \quad , \qquad (3.8)$$

Опір заземлюючого контуру 1,89 Ом, що відповідає умові Rзу < 4Ом.

3.5 Заходи, що забезпечують виробничу санітарію та гігієну праці

3.5.1 Мікроклімат

Трудова діяльність людини завжди протікає в певних метеорологічних умовах, які визначаються поєднанням температури повітря, швидкості його руху і відносної вологості, тиском і тепловим випромінюванням від нагрітих поверхонь. Оскільки експлуатація проектованого програмного засобу відбувається в приміщенні, то ці показники в сукупності (за винятком тиску) називаються мікрокліматом виробничого приміщення. На даний час основним нормативним документом, щодо нормалізації мікроклімату є [1].

Важкість праці характеризує сукупну дію всіх елементів, складових умови праці, на працездатність людини, його здоров'я, життєдіяльність і відновлення робочої сили. У такому представлені поняття тяжкості праці однаково застосовні як до розумової, так і до фізичної праці. Згідно [1] тяжкість роботи персоналу, який обслуговує ЕОМ, відноситься до легкої категорії 1б (роботи, виконувані сидячи, не вимагаючи систематичного фізичного напруження і перенесення важких предметів). Загальні санітарно-гігієнічні вимоги до повітря робочої зони. Оптимальні норми мікроклімату в робочій зоні, що забезпечуються для робіт

легкої категорії 1б приведені в таблиці 3.1.

Таблиця 3.1 - Оптимальні норми мікроклімату

Період року	Температура, ºС	Відносна вологість,%	Швидкість вітру, м / с
Холодний	21 - 23	60 - 40	0,1
Теплий	22 - 24	60 - 40	0,2

3.5.2 Освітлення

Світло є природною умовою існування людини . Воно впливає на стан вищих психічних функцій і фізіологічні процеси в організмі. Гарне освітлення діє тонізуюче, створює гарний настрій, поліпшує протікання основних процесів вищої нервової діяльності.

Збільшення освітленості сприяє поліпшенню працездатності навіть в тих випадках, коли процес праці практично не залежить від зорового сприйняття. При поганому освітленні людина швидко втомлюється, працює менш продуктивно, виникає потенційна небезпека помилкових дій і нещасних випадків.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла у світильниках загального освітлення, оскільки люмінесцентні лампи мають високу потужність (80 Вт), тривалий термін служби (до 10000 годин), спектральний склад випромінюваного світла, близький до сонячного. При експлуатації ЕОМ виконується зорова робота IV в розряд точності (середня точність). При цьому нормована освітленість на робочому місці (Ен) дорівнює 200 лк. Джерелом природного освітлення є сонячне світло. У приміщенні, де розташовані ЕОМ передбачається природне освітлення, рівень якого відповідає стандартам [4].

Регулярно повинен проводитися контроль освітленості, який підтверджує,

що рівень освітленості задовільний і для даного приміщення в світлий час доби достатньо природного освітлення.

3.6 Рекомендації щодо пожежної безпеки

Виникнення пожежі можливо, якщо на об'єкті є горючі речовини, окислювач і джерела запалювання. Для оцінки пожежної небезпеки слід проаналізувати ймовірність взаємодії цих трьох чинників.

Горючими матеріалами в приміщенні, де розташовані ЕОМ, є:

– поліамід - матеріал корпусу мікросхем, горюча речовина, температура самозаймання 420 ° С;

– пластикат кабельний - матеріал ізоляції кабелів, горючий матеріал, показник горючості більше 2.1;

– деревина - будівельний і оздоблювальний матеріал, з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, температура запалювання 255 ° С, температура самозаймання 399 ° С.

Згідно [7] таке приміщення належить до категорії "В" (пожежонебезпечної).

Причинами можливого загоряння і пожежі можуть бути:

– несправність електроустановки;

– конструктивні недоліки обладнання;

– коротке замикання в електричних мережах;

– запалювання горючих матеріалів, що знаходяться в безпосередній близькості від електроустановки.

Продуктами згоряння, що виділяються під час пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор та ін.

При горінні пластмас, крім звичних продуктів згоряння, виділяються різні продукти термічного розкладання: хлорангідридні кислоти; формальдегіди; хлористий водень; фосген; синильна кислота; аміак; фенол; ацетон; стирол [2].

Для захисту персоналу від впливу небезпечних і шкідливих факторів пожежі проектом передбачається застосування промислового протигаза фільтруючого з коробкою марки В (жовтий).

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється розвиненою системою електроживлення ЕОМ. Небезпека загорання в ЕОМ пов'язана з великою кількістю щільно розташованих на платі і блоках електронних вузлів і схем, електричних і комутаційних кабелів, резисторів, конденсаторів, напівпровідникових діодів і транзисторів.

Пожежна безпека при застосуванні ЕОМ забезпечується:

– системою запобігання пожежі:

– системою протипожежного захисту:

– організаційно-технічними заходами.

Запобігти утворенню горючого середовища не надається технічно можливим. Тому проектом передбачаються способи і засоби запобігання утворенню (або внесення) в горюче середовище джерел запалювання, таких як:

– застосування електроустаткування, відповідної пожежонебезпечної і вибухонебезпечної зонами відповідно до ПУЕ [8];

– застосування в конструкції швидкодіючих засобів захисного відключення можливих джерел запалювання;

– виключення можливості появи іскрового розряду в займистою середовищі з енергією, яка дорівнює і вище мінімальної енергії запалювання.

Висновок до розділу 3

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в кваліфікаційній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника.

Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Була наведені розміри приміщення та значення температури, вологості і рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

ВИСНОВКИ

В дипломному проекті було розроблено програмне забезпечення тестування пульта перевірки комп’ютерних блоків.

Програмне забезпечення було розроблене засобами високорівневої мови програмування C++ з використанням засобів бібліотеки Qt.

Пульт перевірки застосовується під час перевірки та ремонту пристроїв диспетчерської централізації, з метою визначення їх справності та відповідності вимогам.

Процес тестування має на меті продемонструвати розробникам і замовникам, що пристрій відповідає вимогам, а також виявити ситуації, в яких поведінка тестованого пристрою є неправильною, небажаною або не відповідає специфікації.

Розроблене програмне забезпечення для тестування пульту перевірки комп’ютерних блоків було створене у відповідності до поставленого завдання, відповідає всім вимогам технічного завдання та стандартам підприємства.

В розділі «Охорона праці» в результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в кваліфікаційній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника.

Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Була наведені розміри приміщення та значення температури, вологості і рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ГОСТ 12.1.005-88 "ССБТ Загальні санітарно-гігієнічні вимоги до повітря робочої зони ".

2. ГОСТ 12.1.044-89 ССБТ. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения.

3. ГОСТ 12.0.003-74 “ССБТ. Опасные и вредные производственные факторы. Классификация”

4. ДБН В.2.5-28:2015 Природне і штучне освітлення.

5. ДСанПіН 3.3.2-007-98 Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

6. НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації ЕОМ.

7. НАПБ Б.03.002-2007 Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою.

8. Правила улаштування електроустановок (ПУЕ). Видання 3-тє, перероб. і доп. – К.: Мінпаливенерго України, 2010. – 736 с.

9. Qt 4.8. Профессиональное программирование на C++. — СПб.: БХВ-Петербург, 2012. – 912 с.

10. WinPcap Documentation 4.1.2. Authors: The WinPcap Team. URL: - http://www.winpcap.org

11. Елисеев В.В., Ларгин В.А., Пивоваров Г.Ю. Программно-технические комплексы АСУ ТП: Учебное пособие. – К: Издательско-полиграфический центр «Киевский университет». 2003. – 429 с.

12. Институт Открытия Способностей OpenAbilitiesInstitute. URL: - www.oai.ru

13. Инструкция по охране труда для работников отдела № 6 ЧАО «СНПО «Импульс». – ЧАО «СНПО «Импульс», 2012.

14. Иуду К.А. Надежность, контроль и диагностика вычислительных машин и систем. М. - Высшая школа, 2003, 252 с.

15. Козлов М. Развитие Бизнеса / Ру 2.0. URL: – devbiz.narod.ru.

16. Программное обеспечение Пульта проверки ПП-78 0229767.00984-01 ТТ. Технические требования к ПО. – ЧАО «СНПО «Импульс», 2014.

17. Пульт проверки ПП-78. Инструкция по проверке и приемке. – ЧАО «СНПО «Импульс», 2014.

18. Северодонецкое НПО «Импульс». URL: – www.imp.lg.ua.

ДОДАТОК А

Схема перевірки ПП

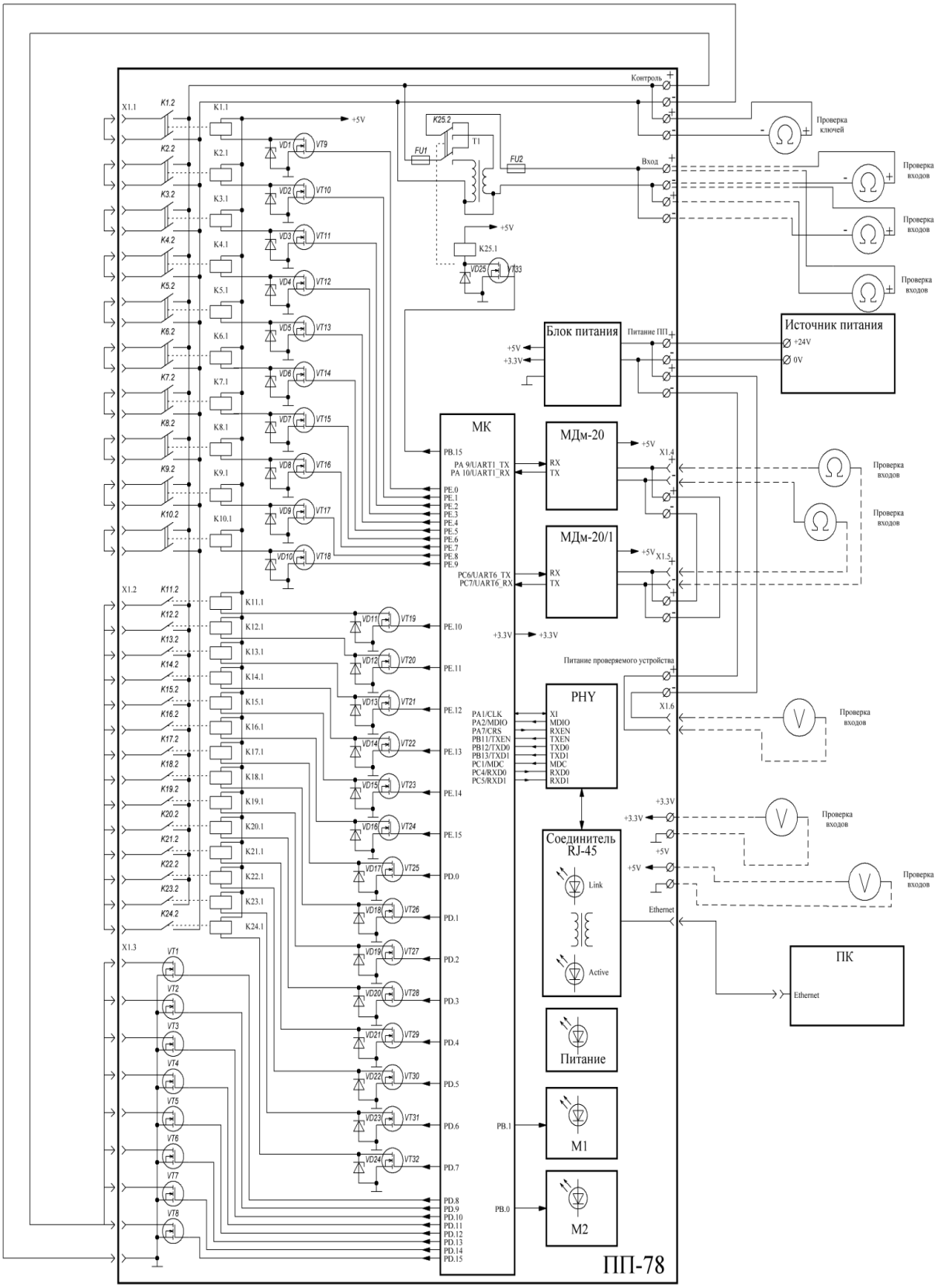

Рис. А.1 – Схема перевірки ПП-78

ДОДАТОК Б

Приклади лістингів файлів підпректів Debug_test, Diagnostic та test_1_PP-78

frames.cpp

```
1  #include "protocol.h"  #include <QtGlobal>
2  uint16_t ALT_CI_SWAPH_INST(unsigned long);
3  const unsigned char Crc8Table[256] = {
4      0x00, 0x31, 0x62, 0x53, 0xC4, 0xF5, 0xA6, 0x97,
5      0xB9, 0x88, 0xDB, 0xEA, 0x7D, 0x4C, 0x1F, 0x2E,
6      0x43, 0x72, 0x21, 0x10, 0x87, 0xB6, 0xE5, 0xD4,
7      0xFA, 0xCB, 0x98, 0xA9, 0x3E, 0x0F, 0x5C, 0x6D,
8      0x86, 0xB7, 0xE4, 0xD5, 0x42, 0x73, 0x20, 0x11,
9      0x3F, 0x0E, 0x5D, 0x6C, 0xFB, 0xCA, 0x99, 0xA8,
10     0xC5, 0xF4, 0xA7, 0x96, 0x01, 0x30, 0x63, 0x52,
11     0x7C, 0x4D, 0x1E, 0x2F, 0xB8, 0x89, 0xDA, 0xEB,
12     0x3D, 0x0C, 0x5F, 0x6E, 0xF9, 0xC8, 0x9B, 0xAA,
13     0x84, 0xB5, 0xE6, 0xD7, 0x40, 0x71, 0x22, 0x13,
14     0x7E, 0x4F, 0x1C, 0x2D, 0xBA, 0x8B, 0xD8, 0xE9,
15     0xC7, 0xF6, 0xA5, 0x94, 0x03, 0x32, 0x61, 0x50,
16     0xBB, 0x8A, 0xD9, 0xE8, 0x7F, 0x4E, 0x1D, 0x2C,
17     0x02, 0x33, 0x60, 0x51, 0xC6, 0xF7, 0xA4, 0x95,
18     0xF8, 0xC9, 0x9A, 0xAB, 0x3C, 0x0D, 0x5E, 0x6F,
19     0x41, 0x70, 0x23, 0x12, 0x85, 0xB4, 0xE7, 0xD6,
20     0x7A, 0x4B, 0x18, 0x29, 0xBE, 0x8F, 0xDC, 0xED,
21     0xC3, 0xF2, 0xA1, 0x90, 0x07, 0x36, 0x65, 0x54,
22     0x39, 0x08, 0x5B, 0x6A, 0xFD, 0xCC, 0x9F, 0xAE,
23     0x80, 0xB1, 0xE2, 0xD3, 0x44, 0x75, 0x26, 0x17,
24     0xFC, 0xCD, 0x9E, 0xAF, 0x38, 0x09, 0x5A, 0x6B,
25     0x45, 0x74, 0x27, 0x16, 0x81, 0xB0, 0xE3, 0xD2,
26     0xBF, 0x8E, 0xDD, 0xEC, 0x7B, 0x4A, 0x19, 0x28,
27     0x06, 0x37, 0x64, 0x55, 0xC2, 0xF3, 0xA0, 0x91,
28     0x47, 0x76, 0x25, 0x14, 0x83, 0xB2, 0xE1, 0xD0,
29     0xFE, 0xCF, 0x9C, 0xAD, 0x3A, 0x0B, 0x58, 0x69,
30     0x04, 0x35, 0x66, 0x57, 0xC0, 0xF1, 0xA2, 0x93,
31     0xBD, 0x8C, 0xDF, 0xEE, 0x79, 0x48, 0x1B, 0x2A,
32     0xC1, 0xF0, 0xA3, 0x92, 0x05, 0x34, 0x67, 0x56,
```

```
    0x78, 0x49, 0x1A, 0x2B, 0xBC, 0x8D, 0xDE, 0xEF,
    0x82, 0xB3, 0xE0, 0xD1, 0x46, 0x77, 0x24, 0x15,
    0x3B, 0x0A, 0x59, 0x68, 0xFF, 0xCE, 0x9D, 0xAC
};
unsigned short CrcTable[256] = {
    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
    0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
    0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
    0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
    0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
    0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
    0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
    0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc,
    0x48c4, 0x58e5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
    0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
    0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12,
    0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
    0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03, 0x0c60, 0x1c41,
    0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49,
    0x7e97, 0x6eb6, 0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
    0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a, 0x9f59, 0x8f78,
    0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f,
    0x1080, 0x00a1, 0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
    0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c, 0xe37f, 0xf35e,
    0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256,
    0xb5ea, 0xa5cb, 0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
    0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
    0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c,
    0x26d3, 0x36f2, 0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
    0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9, 0xb98a, 0xa9ab,
    0x5844, 0x4865, 0x7806, 0x6827, 0x18c0, 0x08e1, 0x3882, 0x28a3,
    0xcb7d, 0xdb5c, 0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
    0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0, 0x2ab3, 0x3a92,
    0xfd2e, 0xed0f, 0xdd6c, 0xcd4d, 0xbdaa, 0xad8b, 0x9de8, 0x8dc9,
    0x7c26, 0x6c07, 0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
    0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba, 0x8fd9, 0x9ff8,
    0x6e17, 0x7e36, 0x4e55, 0x5e74, 0x2e93, 0x3eb2, 0x0ed1, 0x1ef0
  };
uint8_t checkCRC_h(uint8_t *buf, unsigned long size)
```

```
{  unsigned long crc = 0xFFFFFFFF;
  do {       crc = Crc8Table[(uint8_t)(*buf++ ^ crc)];      // Add new 8bit value to CRC8
  } while(--size);
  return (uint8_t)crc;       // Read CRC8 result
}
uint16_t calckCRC_i(uint8_t *buf, unsigned long size)
{  unsigned long i;
  uint16_t CurCrc = 0xffff;
  uint8_t CurCrc8 = 0xff;
  for(i = 0; i < size;) {
       CurCrc8 = Crc8Table[CurCrc8 ^ *(buf + i++)];
       if((i %15) == 0) {
         CurCrc = (CurCrc << 8) ^ CrcTable[((CurCrc >> 8) & 255) ^ CurCrc8];
         CurCrc8 = 0xff;       }  }
  if(i % 15) CurCrc = (CurCrc << 8) ^ CrcTable[((CurCrc >> 8) & 255) ^ CurCrc8];
  return CurCrc;
}
uint16_t Crc16(uint8_t *pData, uint16_t len)
{  uint16_t crc = 0;
  do { crc = (crc << 8) ^ CrcTable[(uint8_t)((crc >> 8) ^ *pData++)];
  } while(--len);
  return crc;
}
uint16_t ALT_CI_SWAPH_INST(unsigned long tmp)
{  __asm__("bswapl %0":"=r"(tmp):"r"(tmp));
    tmp >>= 16;
    return (uint16_t)tmp;}
```

debug.cpp

```
#include "debug.h" #include "ui_debug.h" #include "schema.h" #include "ui_schema.h"
QTextCodec *vCodec = QTextCodec::codecForName("Windows-1251");
extern uint8_t checkCRC_h(uint8_t*, unsigned long);
extern uint16_t calckCRC_i(uint8_t*, unsigned long);
extern uint16_t ALT_CI_SWAPH_INST(unsigned long);
debug::debug(QWidget *parent) :
```

```
    QMainWindow(parent),
ui(new Ui::debug)
{  QTextCodec *codec = QTextCodec::codecForName("Windows-1251");
    setWindowFlags (Qt::Window /*| Qt::FramelessWindowHint */|
Qt::WindowTitleHint);
    setWindowTitle(NAME_TEST_2);
ui->setupUi(this);
    QApplication::setStyle(new QPlastiqueStyle);
ui->mainToolBar->hide();
ui->menuBar->hide();
ui->lcdInt_1->display(MIN_TIMEOUT);
ui->lcdInt_2->display(MIN_TIMEOUT);
    for (int i = 0; i < 2; i++){
channel[i].pTim = new QTimer(this);
channel[i].pMess = (__request*) &channel[i].bodyMess;
channel[i].intervalTim =  MIN_TIMEOUT;
channel[i].cntRxPacks = DEFAULT_VALUE;
channel[i].cntTxPacks = DEFAULT_VALUE;
channel[i].sizePack = DEFAULT_VALUE;
channel[i].lastOpenFile = "";
channel[i].buffer =  new QString();
channel[i].singleShot = false;
    }
param.result = TEST_COMPLETE;
pServ = (__request*) &bodyServPack;
pTim_2 = new QTimer(this);
    connect(ui->checkAddr_0,          SIGNAL(clicked()),          this,
SLOT(sendServiceMess()));
    …
    connect(channel[1].pTim,          SIGNAL(timeout()),          this,
SLOT(sendFrames_ch2()));
    connect(pTim_2, SIGNAL(timeout()), this, SLOT(filterFrames()));
pTim_2->setInterval(READ_TIMEOUT);
pTim_2->start();
}
debug::~debug()
{   delete ui;
}
void debug::sendFrames_ch1()
```

```
{channel[0].sizePack = genFrame(channel[0].pMess, TYPE_MODEM_1,
channel[0].buffer,
channel[0].cntTxPacks);
   if (pcap_sendpacket(param.fp,                    // Adapter
channel[0].bodyMess,                               // buffer with the packet
channel[0].sizePack + 14)                          // size
   != 0) ui->textBrowser->append(QString("FAIL"));
   if (channel[0].singleShot) on_pushStop_1_clicked();
}
void debug::sendFrames_ch2()
{channel[1].sizePack = genFrame(channel[1].pMess, TYPE_MODEM_2,
channel[1].buffer,
channel[1].cntTxPacks);
   if (pcap_sendpacket(param.fp,                    // Adapter
channel[1].bodyMess,                               // buffer with the packet
channel[1].sizePack + 14)                          // size
   != 0) ui->textBrowser_3->append(QString("FAIL"));
   if (channel[1].singleShot) on_pushStop_2_clicked();
}
void debug::filterFrames()
{   struct pcap_pkthdr *pkt_header;
   const u_char *pkt_data;
   if (pcap_next_ex(param.fp, &pkt_header, &pkt_data) == 1) {
      __frame *rdPack = (__frame *) pkt_data;
      qDebug()<<"TypeMess = "<<rdPack->header.EtherType;
      switch(rdPack->bd.TypeMes){
         case 1:{ if (rdPack->header.src_mac[0] == MAC_PP_78_0 &&
                  rdPack->header.src_mac[1] == MAC_PP_78_1 &&
                  rdPack->header.src_mac[2] == MAC_PP_78_2 &&
                  rdPack->header.src_mac[3] == MAC_PP_78_3 &&
                  rdPack->header.src_mac[4] == MAC_PP_78_4 &&
                  rdPack->header.src_mac[5] == MAC_PP_78_5)
               displayRxFrame(rdPack,                ui->textBrowser_2,
channel[0].cntRxPacks++);
            else          displayRxFrame(rdPack,          ui->textBrowser,
channel[0].cntTxPacks++);
            break;
         }
         case 2: {  if (rdPack->header.src_mac[0] == MAC_PP_78_0 &&
```

```
                    rdPack->header.src_mac[1] == MAC_PP_78_1 &&
                    rdPack->header.src_mac[2] == MAC_PP_78_2 &&
                    rdPack->header.src_mac[3] == MAC_PP_78_3 &&
                    rdPack->header.src_mac[4] == MAC_PP_78_4 &&
                    rdPack->header.src_mac[5] == MAC_PP_78_5)
                displayRxFrame(rdPack, ui->textBrowser_4,
channel[1].cntRxPacks++);
            else displayRxFrame(rdPack, ui->textBrowser_3,
channel[1].cntTxPacks++);
            break;  }
         default: break;
}   }   }
unsigned int debug::genDataMessage(QString* src, __mess_type_m* dt)
{ unsigned int size = 0;
  QString s = QString::QString(*src);
  int i =0;
  for(i = 0; i <s.length()-1; i++)   if (s[i] == '\n' ) s[i] = ' ';   i = 0;
  int j = 0, k = 0;
  while(i < s.length()-1) {       if (s[i] ==  ' ') i++;       else
      { bool ok;
        j = s.indexOf(" ", i);
        if (j == -1) j = s.length();
        QString numStr = s.mid(i, j-i);
        unsigned int num = numStr.toInt(&ok, 16);
        if (ok) { dt->DataAns[k++] = (uint8_t) num; size++;}
        i = j;
  }  }     return size;
}
int debug::genFrame(__request* pkt, uint8_t type, QString* buf, uint32_t
numMess)
{  unsigned int size = 0;
   __PP_head* body = (__PP_head*) &pkt->req;
   body->addr = MAC_PP_78_5;
   body->res = 0;
   body->TypeMes = type;
   size+= 3;
   switch (type) {
      case 3: { __req_type_3 *mc = (__req_type_3 *)(&pkt->req.data);
         mc->ReadWrite = FLAG_WRITE;
```

```
225             mc->K1_K8 =         (type3.bKey_1<< 0) |
226                             (type3.bKey_2<<      1)     |     (type3.bKey_3<<      2)     |
227     (type3.bKey_4<< 3) |
228                             (type3.bKey_5<<      4)     |     (type3.bKey_6<<      5)     |
229     (type3.bKey_7<< 6) ;
230             …
231             mc->stateAdr   =              (type3.bAdr0<<   0)   |   (type3.bAdr1<<   1)   |
232     (type3.bAdr2<< 2) |
233                             (type3.bAdr3<< 3) | (type3.bAdr4<< 4) | (type3.bAdr7<< 7)
234     ;
235             for (unsigned int i = 0; i < sizeof(mc->reserve); i++) mc->reserve[i] = 0;
236             size = SIZE_SERVICE_TYPE;
237             break;   }
238           default:{  __mess_type_m *mr = ( __mess_type_m *)(&pkt->req.data);
239             mr->Lans = (uint8_t) genDataMessage(buf, mr);
240             size += mr->Lans + 1;                    // += LAsk + Data
241             break;
242       }   }
243       __eth_head* head = (__eth_head*) &pkt->header;
244       for (int i = 0; i < 6; i++) head->src_mac[i] = param.mac[i];
245       head->dest_mac[0] = MAC_PP_78_0;
246       …
247       head->dest_mac[5] = MAC_PP_78_5;
248       head->EtherType = ETH_TYPE_0713;
249       head->length = size + 18;
250       head->id_cadr.num_mess = (uint16_t) ALT_CI_SWAPH_INST(numMess);
251       head->id_cadr.reserve = DEFAULT_VALUE;
252       head->dest_addr = head->dest_mac[5];
253       head->control = CTRL_BYTE;
254       head->src_addr = head->src_mac[5];
255       head->inf_length = size;
256       head->id_message.kui = KUP_ID;
257       head->id_message.kup = KUP_ID;
258       head->id_message.reserve = DEFAULT_VALUE;
259       head->id_cadr.crc_i = calckCRC_i((uint8_t *) &body->addr, size);
260       head->id_cadr.crc_h  =  checkCRC_h((uint8_t  *)&head->id_cadr.num_mess,
261     15);
262       return size + 18;
263     }
```

```
void debug::displayRxFrame(__frame *fm, QTextBrowser * log, uint32_t
counter)
{  QString text;
   uint8_t* p;
   switch (fm->bd.TypeMes) {
      case TYPE_MODEM_1: {__mess_type_m* ma_1=(__mess_type_m*)
(&fm->bd.data);
         p = (uint8_t*) ma_1->DataAns;
         text += getFormFrame(p, ma_1->Lans, counter);
         break;
      }
      case TYPE_MODEM_2: {
         __mess_type_m* ma_2 = (__mess_type_m*)(&fm->bd.data);
         p = (uint8_t*) ma_2->DataAns;
         text += getFormFrame(p, ma_2->Lans, counter);
         break;      }
      default: break;   }
   log->append(text);
}
void debug::sendServiceMess()
{  updateStates();
sizePack = genFrame(pServ, TYPE_SERVICE, NULL, DEFAULT_VALUE);
   if (pcap_sendpacket(param.fp,                    // Adapter
bodyServPack,                                 // buffer with the packet
sizePack + 14)                                // size
   == 0);
}
QString debug::getFormFrame(uint8_t *src, uint32_t cnt, uint32_t counter)
{  QTextCodec *codec = QTextCodec::codecForName("Windows-1251");
   QString result = "";
   uint16_t shift = 0x0000; uint32_t i = 0;
   result.append(STR_NUM_MESS.arg(counter));
   result.append("\n");
   return result;
}
void debug::updateStates()
{type3.bAdr0 = (ui->checkAddr_0->isChecked()) ? 1 : 0;
…
type3.bKey_30 = (ui->checkPow_off->isChecked()) ? 1 : 0;
```

```
303 type3.rKey_30 = 0;
304 }
305 /*****************************/
306 /*          BUTTONS          */
307 /*****************************/
308 void debug::on_pushStep_1_clicked()
309 {  if(ui->pushStep_1->isChecked()){
310 channel[0].singleShot = true;
311 channel[0].pTim->setInterval(channel[0].intervalTim);
312 channel[0].pTim->start();
313 ui->pushCycle_1->setDisabled(true);
314 ui->pushStep_1->setDisabled(true);
315 }  }
316 void debug::on_pushCycle_1_clicked()
317 {  if(ui->pushCycle_1->isChecked()) {
318 channel[0].singleShot = false;
319 channel[0].pTim->setInterval(channel[0].intervalTim);
320 channel[0].pTim->start();
321 ui->pushCycle_1->setDisabled(true);
322 ui->pushStep_1->setDisabled(true);
323 }  }
324 void debug::on_pushStop_1_clicked()
325 {channel[0].pTim->stop();
326 ui->pushCycle_1->setChecked(false);
327 ui->pushStep_1->setChecked(false);
328 ui->pushCycle_1->setEnabled(true);
329 ui->pushStep_1->setEnabled(true);
330 }
331 void debug::on_pushStep_2_clicked()
332 {    if(ui->pushStep_2->isChecked()){
333 channel[1].singleShot = true;
334 channel[1].pTim->setInterval(channel[1].intervalTim);
335 channel[1].pTim->start();
336 ui->pushCycle_2->setDisabled(true);
337 ui->pushStep_2->setDisabled(true);
338 }  }
339 void debug::on_pushCycle_2_clicked()
340 {  if(ui->pushCycle_2->isChecked()) {
341 channel[1].singleShot = false;
```

```
342 channel[1].pTim->setInterval(channel[1].intervalTim);
343 channel[1].pTim->start();
344 ui->pushCycle_2->setDisabled(true);
345 ui->pushStep_2->setDisabled(true);
346 }  }
347 void debug::on_pushStop_2_clicked()
348 {channel[1].pTim->stop();
349 ui->pushCycle_2->setChecked(false);
350 ui->pushStep_2->setChecked(false);
351 ui->pushCycle_2->setEnabled(true);
352 ui->pushStep_2->setEnabled(true);
353 }
354 void debug::on_pushClear_1_clicked()
355 {ui->textBrowser->clear();
356 ui->textBrowser_2->clear();
357 }
358 void debug::on_pushClear_2_clicked()
359 {ui->textBrowser_3->clear();
360 ui->textBrowser_4->clear();
361 }
362 void debug::on_sliderInt_1_valueChanged(int value)
363 {ui->lcdInt_1->display(value);
364 channel[0].intervalTim = value;
365 }
366 void debug::on_sliderInt_2_valueChanged(int value)
367 {ui->lcdInt_2->display(value);
368 channel[1].intervalTim = value;
369 }
370 void debug::on_pushMessage_clicked()
371 {sh = new schema(0, channel[0].buffer);
372 sh->exec(); //show modal form "SETUP"
373 }
374 void debug::on_pushMessage_2_clicked()
375 {sh = new schema(0, channel[1].buffer);
376 sh->exec(); //show modal form "SETUP"
377 }
378 void debug::on_pushQuit_debug_clicked()
379 {  for(int i = 0; i <2; i++) channel[i].pTim->stop();
380 pTim_2->stop();
```

```
    close();
}
```

diagnostic.h

```
#ifndef DIAGNOSTIC_H #define DIAGNOSTIC_H #include
<QtCore/QTextStream>
#include <QMainWindow> #include <QMenu> #include <QtGui>
#include <about.h>
#include <pcap.h> #include <protocol.h> #include <QMenuBar> #include
<QtCore/QFile>
#include <QtNetwork> #include <QTextCodec> #include <map.h> #include
<message.h>
enum __statusTest {_waiting,      //ожидание
                   _omission,     //пропуск
                   _running,      //выполнение
                   _complete,     //выполнился
                   _interrupt,    //прерван
                   _error         //ошибка
          };
namespace Ui {class diagnostic;
}
class diagnostic : public QMainWindow
{  Q_OBJECT
public:
    explicit diagnostic(QWidget *parent = 0);
    ~diagnostic();
    private:
    Ui::diagnostic *ui;
    //<! members
    QMenuBar *menuBar;
    …
    QAction *aboutMeAction;
    pcap_if_t *alldevs;
    QDateTime dateTime;
    static int const MAX_ETH_CARDS = 5;
    uint8_t numEthCards,
          numActiveCard,
          numTests,
```

```
417             numCycles,
418             actCycle;
419         int     verPP_78;
420         bool    isError,
421             isInterrupt;
422         //<! structures
423         struct __attributeTest
424         {       __statusTest statusTest;
425           bool permit;                  // get value from check boxes
426         }attributeTest[10];
427         struct PCAP_address
428         {       QString hardwareAddress_str_QT;
429           unsigned char  hardwareAddress_char_QT[6];
430           QString iface_name_QT;
431           QString description_QT;
432           QString name;
433           QString description;
434          } PCAP_addr[10];
435         struct __ethDevs {
436           QString         name,
437                         description,
438                         strMACAddr;
439           uint8_t         macAddress[6];
440           bool            valid;
441           pcap_if_t       *ptr;
442         } ethDevs[MAX_ETH_CARDS];
443         //<! methods()
444         void createDirs();
445         void createMenu();
446         void createEthActions();
447         void openEthernetCard();
448         void getMAC(QString src, uint8_t mas[]);
449         void on_mac_address();
450         void writeLog(QString q);
451         void writeLog(char* q);
452         unsigned char lit2num(unsigned char  lit);
453         uint8_t findTest();
454         void printNameTest(const char*, int);
```

```
455     void changeAttributeTest(QCheckBox* check, uint8_t index, __statusTest
456 temp);
457     void initGUI();
458     void headProt(QFile* src);
459     void fooProt(QFile* src);
460     //<! slots()
461     private slots:
462         void on_buttonPerform_clicked();
463         …
464         void on_checkTest_6_clicked(bool checked);
465 };
466 #endif // DIAGNOSTIC_H
```

diagnostic.cpp

```
467 #include "diagnostic.h"        #include "ui_diagnostic.h"
468 QTextCodec *codec = QTextCodec::codecForName("Windows-1251");
469 extern "C" { pcap_t* pcap_open (const char * , int ,int, int, struct pcap_rmtauth
470 *, char *);}
471 extern  pcap_t* fp;
472 pcap_if_t *pd[5];  //      Maximum      ethernet      adapters      in      system
473 MAX_ETH_CARDS = 5
474 pcap_if_t *d;
475 char errbuf[PCAP_ERRBUF_SIZE];
476 diagnostic::diagnostic(QWidget *parent) :
477     QMainWindow(parent),
478 ui(new Ui::diagnostic)
479 {ui->setupUi(this);
480     for (int i = 0; i < 5; i++) ethDevs[i].valid = false;
481     for (int i = 0; i < 10; i++) {
482 attributeTest[i].permit = false;
483 attributeTest[i].statusTest = _omission;
484     }
485     initGUI();
486     createDirs();
487 }
488 diagnostic::~diagnostic()
489 {  delete ui;
490 }
```

```
uint8_t diagnostic::findTest()
{  typedef const char* (*Fct)(void);
   for(int i = 1;;i++)
   {     QLibrary lib(QString("test_%1_%2").arg(i).arg(MODULE_EN));
      Fct fct = (Fct) lib.resolve("nameTest");
      if (fct){
         printNameTest(fct(),i);
      }
      else {     return i - 1;
}  }  }
void diagnostic::printNameTest(const char* a, int b)
{  switch(b){
      case 1:  ui->lblNameTest_1->setText(codec->toUnicode(a));
ui->lblNameTest_1->show();
ui->checkTest_1->show();
ui->checkTest_1->setText(CHECK_OMISSION);
         break;
…
      case 5:  ui->lblNameTest_5->setText(codec->toUnicode(a));
ui->lblNameTest_5->show();
ui->checkTest_5->show();
ui->checkTest_5->setText(CHECK_OMISSION);
         break;
      default:  ui->lblNameTest_6->setText(codec->toUnicode(a));
ui->lblNameTest_6->show();
ui->checkTest_6->show();
ui->checkTest_6->setText(CHECK_OMISSION);
         break;
}  }
void diagnostic::changeAttributeTest(QCheckBox* check, uint8_t index,
__statusTest temp)
{  switch(temp){
      case _waiting:
         check->setText(CHECK_WAITING);
         break;
      case _omission:
         check->setText(CHECK_OMISSION);
         break;
      case _running:
```

```
530             check->setText(CHECK_RUNNING);
531              break;
532           case _complete:
533              check->setText(CHECK_COMPLETE);
534              break;
535           case _interrupt:
536             check->setText(CHECK_INTERRUPT);
537              break;
538           default: case _error:
539              check->setText(CHECK_ERROR);
540              break;
541        }
542    attributeTest[index].statusTest = temp;
543    attributeTest[index].permit = (temp == _waiting ||
544                                   temp == _complete ||
545                                   temp == _interrupt
546                                   ) ? true: false;
547       /*qDebug("changeAttributeTest();");
548       qDebug()<<"attributeTest["<<index<<"].statusTest                    =
549    "<<attributeTest[index].statusTest;
550       qDebug()<<"attributeTest["<<index<<"].permit                        =
551    "<<attributeTest[index].permit;*/
552    }
553    void diagnostic::createMenu()
554    {…}
555    void diagnostic::createEthActions()
556    {  int i = 0;
557    mnEthCard = new QMenu(this);
558    mnEthCardGroup = new QActionGroup(this);
559       connect(mnEthCardGroup,      SIGNAL(triggered(QAction      *)),      this,
560    SLOT(switchCard(QAction *)));
561       // Retrieve the device list from the local machine
562       if (pcap_findalldevs(&alldevs, errbuf) == -1)  {
563         writeLog(ETH_ERR_FIND.arg(errbuf)));
564       } else {     for(d = alldevs; d; d = d->next) {
565             QAction *action = new QAction(tr("%0").arg(d->description), this);
566             action->setCheckable(true);
567             if(i == 0)action->setChecked(true);
568    mnEthCard->addAction(action);
```

```
569 mnEthCardGroup->addAction(action);
570 ethDevs[i].name = d->name;
571 ethDevs[i].description = d->description;
572 ethDevs[i].ptr = d;
573         pd[i++] = d;
574       }
575 numEthCards = i;
576       writeLog(ETH_FIND_CARD.arg(numEthCards)));
577 numActiveCard = 0;
578       openEthernetCard();
579       on_mac_address();
580 }   }
581 void diagnostic::switchCard(QAction *action)
582 { int i = 0;
583    for(d = alldevs; d; d = d->next) {
584       if(action->text() == (QApplication::translate
585                   ("RawEthernet",          pd[i]->description,          0,
586 QApplication::UnicodeUTF8))) {
587 numActiveCard = i;  }
588       i++;  }
589    openEthernetCard();
590 }
591 void diagnostic::on_mac_address()
592 {  int i=0, j=0;
593    for(i = 0; i < 6; i++)
594    {  QNetworkInterface iface = QNetworkInterface::interfaceFromIndex(i);
595       if (iface.isValid()) {  // получаем данные о выбранном интерфейсе
596          for(j = 0; j <numEthCards; j++)
597          {                          if(ethDevs[j].name.right(38)          ==
598 (QString::fromUtf8("%1").arg(iface.name())))
599             {ethDevs[j].valid = iface.isValid();
600 ethDevs[j].strMACAddr = iface.hardwareAddress();
601                getMAC(ethDevs[j].strMACAddr, ethDevs[j].macAddress);
602 }  }  }  }  }
603 void diagnostic::openEthernetCard()
604 {  struct bpf_program filter; // Скомпилированноевыражениедляфильтра
605    char filter_app[] = ETH_FILTER; // Выражениедляфильтра
606    char errbuf[PCAP_ERRBUF_SIZE];// Строкасошибкой
607 bpf_u_int32 mask;// Сетевая маска интерфейса
```

```
608     bpf_u_int32 net; // IP адрес нашего интерфейса
609     //! \brief открытие сессии
610   if ((fp =  pcap_open(pd[numActiveCard]->name,  // имя устройства
611                 65536, // макс.число байт сетевого кадра, кот. захватывается
612   библиотекой
613                 false,// установлен в true, интерфейс переходит в promiscuous
614   mode
615                       (перехватываются пакеты, адресованные другим
616   станциям сети)
617                 1,    // тайм-аут в мс (в случае, если значение установлено 0,
618   чтение будет
619                       происходить до первой ошибки, в минус единицу -
620   бесконечно)
621   NULL, // authentication on the remote machine
622   errbuf   // сообщение об ошибке
623                 )) == NULL)
624   writeLog(ETH_ERR_OPEN_CARD.arg(pd[numActiveCard]->name));
625   //! \brief настройка фильтра для перехвата пакетов
626   pcap_lookupnet(pd[numActiveCard]->name, &net, &mask, errbuf);
627     pcap_compile(fp, &filter, filter_app, 0, net);
628     pcap_setfilter(fp, &filter);
629     qDebug()<<ETH_FILTER;
630     qDebug()<<QString(errbuf);
631   }
632   void diagnostic::getMAC(QString src, uint8_t mas[])
633   {  int index_1 = 0, index_2 = 0, i = 0;
634     bool ok;
635     QString toHex;
636     while(index_1 < src.length() - 1)
637     {  index_2 = src.indexOf(':', index_1);
638       if (index_2 ==  -1) index_2 =src.length();
639       toHex = src.mid(index_1, index_2-index_1);
640       mas[i++] = toHex.toInt(&ok, 16);
641       index_1 = index_2 + 1;
642   }  }
643   void diagnostic::writeLog(QString q)
644   {ui->textLog->append(q);
645   }
646   void diagnostic::writeLog(char* q)
```

```
{ui->textLog->append(QApplication::translate ("diagnostic",q,0,
QApplication::UTF8));
}
unsigned char diagnostic::lit2num(unsigned char lit)
{lit -= '0';
  if(lit < 10) return lit;
  else return lit - 7;
}
void diagnostic::on_buttonPerform_clicked()
{  createDirs();
    showModalMap();
verPP_78 = -1;
    QFile fileLargeProt;
    QFile fileShortProt;
dateTime = QDateTime::currentDateTime();
    QString path = QString("Protocols/Detailed/protocol %0 %1
.txt").arg(MODULE_EN).arg(dateTime.toString("MM-dd-yyyy hh.mm.ss"));
    fileLargeProt.setFileName(path);
    fileLargeProt.open(QIODevice::WriteOnly | QIODevice::Text);
    path = QString("Protocols/Generic/protocol %0 %1
.txt").arg(MODULE_EN).arg(dateTime.toString("MM-dd-yyyy hh.mm.ss"));
    fileShortProt.setFileName(path);
    fileShortProt.open(QIODevice::WriteOnly | QIODevice::Text);
    headProt(&fileLargeProt);
    headProt(&fileShortProt);
    typedef const int (*Fct)(pcap_t*, uint8_t[], QString*, char *, int *);
    typedef const char* (*NAM)(void);
isInterrupt = false;
isError = false;
numCycles = ui->spinCycles->value();
ui->textLog->clear();
    QTextStream streamNames(&fileShortProt);
    QTextStream streamLarge(&fileLargeProt);
    for (actCycle = 1; actCycle<= numCycles; actCycle++) {
       if (numCycles> 1) {
          streamNames << MESS_CYCLE.arg(actCycle);
          streamLarge << MESS_CYCLE.arg(actCycle);
          streamLarge << endl;
          writeLog(MESS_CYCLE.arg(actCycle));
```

```
686         }
687         for(uint8_t i = 1; i <= numTests; i++) {
688           /* qDebug("on_buttonPerform_clicked();");
689            qDebug()<<"attributeTest["<<i-1<<"].statusTest = "<<attributeTest[i-
690 1].statusTest;
691            qDebug()<<"attributeTest["<<i-1<<"].permit = "<<attributeTest[i-
692 1].permit;*/
693            if (attributeTest[i-1].permit) {
694               QLibrary lib(QString("test_%0_%1").arg(i).arg(MODULE_EN));
695               NAM nam = (NAM) (lib.resolve("nameTest"));
696               QString toProtocol = "";
697               if (nam) {
698                  toProtocol.append(codec->toUnicode(nam()));
699                  toProtocol.append("\n");
700                  writeLog(codec->toUnicode(nam()));
701                  streamNames << QString(codec->toUnicode(nam()));
702                  streamNames << endl;
703               }
704               Fct fct = (Fct) (lib.resolve("startTest"));
705               if (fct) {
706                  changeAttributeTest((i == 1) ? ui->checkTest_1 :
707                                      (i == 2) ? ui->checkTest_2 :
708                                      (i == 3) ? ui->checkTest_3 :
709                                      (i == 4) ? ui->checkTest_4 :
710                                      (i == 5) ? ui->checkTest_5 :
711 ui->checkTest_6, i-1, _running);
712                  int result = (fct(fp, ethDevs[numActiveCard].macAddress,
713 &toProtocol,
714                        pd[numActiveCard]->name, &verPP_78));
715                  QTextStream stream(&fileLargeProt);
716                  stream << toProtocol;
717                  switch (result) {
718                     case TEST_COMPLETE:
719                        changeAttributeTest((i == 1) ? ui->checkTest_1 :
720                                            (i == 2) ? ui->checkTest_2 :
721                                            (i == 3) ? ui->checkTest_3 :
722                                            (i == 4) ? ui->checkTest_4 :
723                                            (i == 5) ? ui->checkTest_5 :
724 ui->checkTest_6, i-1, _complete);
```

```
725                     break;
726                     case TEST_INTERRUPT:
727                        changeAttributeTest((i == 1) ? ui->checkTest_1 :
728                                            (i == 2) ? ui->checkTest_2 :
729                                            (i == 3) ? ui->checkTest_3 :
730                                            (i == 4) ? ui->checkTest_4 :
731                                            (i == 5) ? ui->checkTest_5 :
732    ui->checkTest_6, i-1, _interrupt);
733    isError |= true;
734    isInterrupt = true;
735                        break;
736                     case TEST_ERROR:
737                        changeAttributeTest((i == 1) ? ui->checkTest_1 :
738                                            (i == 2) ? ui->checkTest_2 :
739                                            (i == 3) ? ui->checkTest_3 :
740                                            (i == 4) ? ui->checkTest_4 :
741                                            (i == 5) ? ui->checkTest_5 :
742    ui->checkTest_6, i-1, _complete);
743    isError |= true;
744                        break;
745                     default:
746                        changeAttributeTest((i == 1) ? ui->checkTest_1 :
747                                            (i == 2) ? ui->checkTest_2 :
748                                            (i == 3) ? ui->checkTest_3 :
749                                            (i == 4) ? ui->checkTest_4 :
750                                            (i == 5) ? ui->checkTest_5 :
751    ui->checkTest_6, i-1, _error);
752                     break;
753                  }
754               }
755               else
756                  changeAttributeTest((i == 1) ? ui->checkTest_1 :
757                                      (i == 2) ? ui->checkTest_2 :
758                                      (i == 3) ? ui->checkTest_3 :
759                                      (i == 4) ? ui->checkTest_4 :
760                                      (i == 5) ? ui->checkTest_5 :
761    ui->checkTest_6, i-1, _error);
762    }  }  }
763    actCycle--; //! \brief `cause actCycle starts with 1
```

```
764   dateTime = QDateTime::currentDateTime();
765      fooProt(&fileLargeProt);
766      fooProt(&fileShortProt);
767   }
768   void diagnostic::on_buttonExit_clicked()
769   {  pcap_freealldevs(alldevs);    // Free the device list
770      close();
771   }
772   void diagnostic::initGUI()
773   {  setWindowTitle(TITLE_MAIN);
774   ui->mainToolBar->hide();
775      QApplication::setStyle(new QPlastiqueStyle);
776   ui->buttonExit->setStyle(new QCleanlooksStyle);
777   ui->buttonPerform->setStyle(new QCleanlooksStyle);
778   ui->lblNameTest_1->hide();
779   …
780   ui->lblNameTest_6->hide();
781   ui->checkTest_1->hide();
782   …
783   ui->checkTest_6->hide();
784   ui->checkTest_1->setChecked(false);
785   …
786   ui->checkTest_6->setChecked(false);
787   ui->checkALL->setText(CHECK_INVERSE);
788   ui->checkALL->setChecked(false);
789   ui->checkALL->setTristate(true);
790   fullProtAction      = new QAction(ACTION_FULL, this);
791      connect(fullProtAction,                SIGNAL(triggered()),                this,
792   SLOT(openFullProtDir()));
793   shortProtAction   = new QAction(ACTION_SHORT, this);
794      connect(shortProtAction,               SIGNAL(triggered()),                this,
795   SLOT(openShortProtDir()));
796   showMapAction  = new QAction(ACTION_SCHEMA, this);
797      connect(showMapAction,                 SIGNAL(triggered()),                this,
798   SLOT(showModalMap()));
799   aboutMeAction   = new QAction(ACTION_ABOUT, this);
800      connect(aboutMeAction,                 SIGNAL(triggered()),                this,
801   SLOT(showAboutMe()));
802      createMenu();
```

```
numTests = findTest();
}
void diagnostic::on_checkALL_stateChanged(int arg1)
{  switch (arg1) {
      case Qt::Unchecked:
ui->checkALL->setText(CHECK_INVERSE);
         on_checkTest_1_clicked(false);
         …
         on_checkTest_6_clicked(false);
ui->checkTest_1->setChecked(false);
…
ui->checkTest_6->setChecked(false);
         break;
      case Qt::PartiallyChecked:
ui->checkALL->setText(CHECK_S_ALL);
         on_checkTest_1_clicked(! ui->checkTest_1->isChecked());
         …
         on_checkTest_6_clicked(! ui->checkTest_6->isChecked());
ui->checkTest_1->setChecked(! ui->checkTest_1->isChecked());
…
ui->checkTest_6->setChecked(! ui->checkTest_6->isChecked());
         break;
      case Qt::Checked:
ui->checkALL->setText(CHECK_S_NONE);
         on_checkTest_1_clicked(true);
         …
         on_checkTest_6_clicked(true);
ui->checkTest_1->setChecked(true);
…
ui->checkTest_6->setChecked(true);
         break;
      default:
         break;
}  }
void diagnostic::on_checkTest_1_clicked(bool checked)
{  changeAttributeTest(ui->checkTest_1, 0, (checked) ? _waiting : _omission);}
void diagnostic::on_checkTest_2_clicked(bool checked)
{  changeAttributeTest(ui->checkTest_2, 1, (checked) ? _waiting : _omission);}
void diagnostic::on_checkTest_3_clicked(bool checked)
```

```
842  {  changeAttributeTest(ui->checkTest_3, 2, (checked) ? _waiting : _omission);}
843  void diagnostic::on_checkTest_4_clicked(bool checked)
844  {  changeAttributeTest(ui->checkTest_4, 3, (checked) ? _waiting : _omission);}
845  void diagnostic::on_checkTest_5_clicked(bool checked)
846  {        changeAttributeTest(ui->checkTest_5, 4, (checked) ? _waiting :
847  _omission);}
848  void diagnostic::on_checkTest_6_clicked(bool checked)
849  {        changeAttributeTest(ui->checkTest_6, 5, (checked) ? _waiting :
850  _omission);}
851  void diagnostic::openFullProtDir()
852  {  QString FileName = (QDir::current().absolutePath());
853     QUrl url = QUrl::QUrl("file:///"+ FileName + "/Protocols/Detailed/");
854     QDesktopServices::openUrl(url);}
855  void diagnostic::openShortProtDir()
856  {  QString FileName = (QDir::current().absolutePath());
857     QUrl url = QUrl::QUrl("file:///"+ FileName + "/Protocols/Generic/");
858     QDesktopServices::openUrl(url);}
859  void diagnostic::showModalMap()
860  {  map* vMap = new map(0);
861     vMap->setModal(true);
862     vMap->setWindowTitle(TITLE_SCHEMA);
863     vMap->exec();}
864  void diagnostic::showAboutMe()
865  {  about* vAbout = new about(0);
866     vAbout->setModal(true);
867     vAbout->setWindowTitle(TITLE_ABOUT);
868     vAbout->exec();}
869  void diagnostic::createDirs()
870  {  QDir dir;
871     QString fileName = ((QDir::current().absolutePath()) + "/Protocols/");
872     if(dir.exists(fileName) == false) dir.mkdir("Protocols");
873     fileName = ((QDir::current().absolutePath()) + "/Protocols/Generic/");
874     if(dir.exists(fileName) == false) dir.mkdir("Protocols/Generic");
875     fileName = ((QDir::current().absolutePath()) + "/Protocols/Detailed/");
876     if(dir.exists(fileName) == false) dir.mkdir("Protocols/Detailed");}
877  void diagnostic::headProt(QFile* src)
878  {  QTextStream stream(src);
879     stream << QString(codec->toUnicode("                    Протокол"));
880     stream << endl;
```

```
881     stream << endl;
882     stream                  <<                  QString(codec->toUnicode("Типблока:
883  %0")).arg(MODULE_RU);
884     stream << endl;
885     stream << QString(codec->toUnicode("Датаивремясозданияпротокола:  ")
886           + dateTime.toString("dd.MM.yyyy  hh:mm:ss"));
887     stream << endl;
888     stream << endl;
889     if (ui->textNblock->toPlainText().length() != 0)
890        stream << QString(codec->toUnicode
891          ("Заводскойномер:  ___%0___").arg(ui->textNblock->toPlainText()));
892     else
893        stream              <<              QString(codec->toUnicode("Заводскойномер:
894  ____________"));
895     stream << endl;
896     stream << endl;
897     stream << QString(codec->toUnicode("Условияпроверки:"));
898     stream << endl;
899     stream << QString(codec->toUnicode
900          ("          температура          окружающего          воздуха,          град.С          -
901  _____"));
902  stream << endl;
903     stream << QString(codec->toUnicode
904          ("          относительная          влажность          воздуха,%                    -
905  _____"));
906  stream<<endl;
907  stream<<QString(codec->toUnicode
908          ("       атмосферное       давление,kPa                                          -
909  _____"));
910  stream << endl;
911     stream << QString(codec->toUnicode
912          ("       прочие       воздействия                                                -
913  _____"));
914  stream<<endl;
915  stream<<endl;
916  stream<<QString(codec->toUnicode("Список тестов:"));
917  stream << endl;
918  }
919  void diagnostic::fooProt(QFile* src)
```

```
{  QTextStream stream(src);
   stream << endl;
   stream << endl;
   stream << endl;
   stream << endl;
   stream << QString(codec->toUnicode("Результат:"));
   stream << endl;
   stream << QString(codec->toUnicode
      ("  версия ПО пульта проверки ПП-78     %0")).arg(verPP_78, 0, 10);
stream << endl;
   stream << QString(codec->toUnicode
      ("  количество пройденных циклов  %0  из
%1")).arg(actCycle).arg(numCycles);
stream << endl;
   stream << endl;
   if(isInterrupt) {
      if(isError) stream << QString(codec->toUnicode("  блокнегоден"));
      else  stream << QString(codec->toUnicode("  блокгоден"));
      stream << endl;
      stream << QString(codec->toUnicode
("Проверка  %0  прервана
%1")).arg(MODULE_RU).arg(dateTime.toString("dd.MM.yyyyhh:mm:ss"));
} else {if(isError) stream << QString(codec->toUnicode("  блокнегоден"));
      else  stream << QString(codec->toUnicode("  блокгоден"));
      stream << endl;
      stream << QString(codec->toUnicode
("Проверка  %0  завершена
%1")).arg(MODULE_RU).arg(dateTime.toString("dd.MM.yyyyhh:mm:ss"));;
}
   stream << endl;
   stream << endl;
   stream << endl;
   stream << QString(codec->toUnicode
      ("Проверяющий  _____  _____  _____"));
stream << endl;
   stream << QString
      (codec->toUnicode("  (Ф.И.О.)  подпись  дата"));
stream<<endl;
src->close();
```

```
959  }
```

test_1.cpp

```
960  #include "test_1.h"
961  #include "ui_test_1.h"
962  QTextCodec *codec = QTextCodec::codecForName("Windows-1251");
963  extern uint8_t checkCRC_h(uint8_t*, unsigned long);
964  extern uint16_t calckCRC_i(uint8_t*, unsigned long);
965  extern uint16_t ALT_CI_SWAPH_INST(unsigned long);
966  extern uint16_t Crc16(uint8_t *pData, uint16_t len);
967  void delay(int n)
968  {  QEventLoop loop;
969     QTimer::singleShot(n, &loop, SLOT(quit()));
970     loop.exec();
971  }
972  uint8_t mess_1[SIZE_ANSV] =
973                     {0xF1, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x10,
974                      0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x20,
975                      0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x30};
976  uint8_t mess_2[SIZE_ANSV] =
977                     {0xF2, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x10,
978                      0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x20,
979                      0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x30};
980  test_1::test_1(QWidget *parent) :
981     QDialog(parent),
982  ui(new Ui::test_1)
983  {ui->setupUi(this);
984     QApplication::setStyle(new QPlastiqueStyle);
985  ui->buttonAbort->setStyle(new QCleanlooksStyle);
986     setWindowTitle(NAME_TEST_1);
987  syncFrame = ( __request*) &bodyFrame[1];
988  reqFrame = ( __request*) &bodyFrame[0];
989  pTim_test = new QTimer(this);
990     connect(pTim_test, SIGNAL(timeout()), this, SLOT(sendForSync()));
991  isError = false;
992  param.result = DEFAULT_VALUE;
993  toProt = "";
994  }
```

```
995  test_1::~test_1()
996  {  delete ui;}
997  unsigned int test_1::genFrame(__request* pkt)
998  {  unsigned int size = 0;
999     __req_type_3* t_3 = (__req_type_3*) (&pkt->req.data);
1000    pkt->req.addr    = MAC_PP_78_5;
1001    pkt->req.res     = DEFAULT_VALUE;
1002    pkt->req.TypeMes = TYPE_SERVICE;
1003    t_3->ReadWrite   = FLAG_READ;
1004    t_3->K25_K30     = DEFAULT_VALUE;
1005    t_3->stateAdr    = DEFAULT_VALUE;
1006    t_3->K17_K24     = DEFAULT_VALUE;
1007    t_3->K9_K16      = DEFAULT_VALUE;
1008    t_3->K1_K8       = DEFAULT_VALUE;
1009    for (unsigned int i = 0; i < sizeof(t_3->reserve); i++) t_3-
1010 >reserve[i]=DEFAULT_VALUE;
1011    size += SIZE_SERVICE_TYPE;
1012    __eth_head* head = (__eth_head*) &pkt->header;
1013    for (int i = 0; i < 6; i++) head->src_mac[i] = param.mac[i];
1014    head->dest_mac[0] = MAC_PP_78_0;
1015    head->dest_mac[1] = MAC_PP_78_1;
1016    head->dest_mac[2] = MAC_PP_78_2;
1017    head->dest_mac[3] = MAC_PP_78_3;
1018    head->dest_mac[4] = MAC_PP_78_4;
1019    head->dest_mac[5] = MAC_PP_78_5;
1020    head->EtherType = ETH_TYPE_0713;
1021    head->length = size + 18;
1022    head->id_cadr.num_mess = (uint16_t) DEFAULT_VALUE;
1023    head->id_cadr.reserve = DEFAULT_VALUE;
1024    head->dest_addr = head->dest_mac[5];
1025    head->control = CTRL_BYTE;
1026    head->src_addr = head->src_mac[5];
1027    head->inf_length = size;
1028    head->id_message.kui = KUP_ID;
1029    head->id_message.kup = KUP_ID;
1030    head->id_message.reserve = DEFAULT_VALUE;
1031    head->id_cadr.crc_i = calckCRC_i((uint8_t *) &pkt->req.addr, size);
1032    head->id_cadr.crc_h = checkCRC_h((uint8_t *) &head->id_cadr.num_mess,
1033 15);
```

```
1034    return size + 18;
1035    }
1036    //! \brief сообщения к КСТ
1037    unsigned int test_1::genFrame(uint8_t mode)
1038    {reqFrame->req.addr = MAC_PP_78_5;
1039    reqFrame->req.res = DEFAULT_VALUE;
1040        unsigned int size = 2;
1041    reqFrame->req.TypeMes = mode;
1042        switch (mode) {
1043            case TYPE_MODEM_1: {
1044                __mess_type_m *mod =(__mess_type_m *) &reqFrame->req.data;
1045                mod->Lans = SIZE_ANSV;
1046                uint8_t *data = (uint8_t *) &mess_1;
1047                for (int i = 0; i < SIZE_ANSV; i++)
1048                mod->DataAns[i] = *data++;
1049                break;
1050            }
1051            case TYPE_MODEM_2: {
1052                __mess_type_m *mod =(__mess_type_m *) &reqFrame->req.data;
1053                mod->Lans = SIZE_ANSV;
1054                uint8_t *data = (uint8_t *) &mess_2;
1055                for (int i = 0; i < SIZE_ANSV; i++)
1056                    mod->DataAns[i] = *data++;
1057                break;
1058        }    }
1059        size += SIZE_ANSV + 2;
1060        __eth_head* head = (__eth_head*) &reqFrame->header;
1061        for (int i = 0; i < 6; i++) head->src_mac[i] = param.mac[i];
1062        head->dest_mac[0] = MAC_PP_78_0;
1063        head->dest_mac[1] = MAC_PP_78_1;
1064        head->dest_mac[2] = MAC_PP_78_2;
1065        head->dest_mac[3] = MAC_PP_78_3;
1066        head->dest_mac[4] = MAC_PP_78_4;
1067        head->dest_mac[5] = MAC_PP_78_5;
1068        head->EtherType = ETH_TYPE_0713;
1069        head->length = size + 18;
1070        head->id_cadr.num_mess = mode;
1071        head->id_cadr.reserve = DEFAULT_VALUE;
1072        head->dest_addr = head->dest_mac[5];
```

```
1073        head->control = CTRL_BYTE;
1074        head->src_addr = head->src_mac[5];
1075        head->inf_length = size;
1076        head->id_message.kui = KUP_ID;
1077        head->id_message.kup = KUP_ID;
1078        head->id_message.reserve = DEFAULT_VALUE;
1079        head->id_cadr.crc_i = calckCRC_i((uint8_t *) &reqFrame->req.addr, size);
1080        head->id_cadr.crc_h = checkCRC_h((uint8_t *) &head->id_cadr.num_mess,
1081    15);
1082        return size + 18;
1083    }
1084    bool test_1::sendMessage(uint8_t to_modem, uint8_t from_modem)
1085    {  bool isFind = false, result = false;
1086        __asknowledge *rdFrame;
1087        QTime time;
1088    ui->textBrowser->append
1089              (STR_MODEM_TO_FROM.arg(to_modem,  0,  16).arg(from_modem,  0,
1090    16));
1091        delay(DELAY_100MS);
1092        start_session();
1093    sizeFrame = genFrame(to_modem);
1094        pcap_sendpacket(param.fp,                  // Adapter
1095    bodyFrame[0],              // buffer with the packet
1096    sizeFrame + 14);           // size
1097        time.restart();
1098        while (!isFind) {
1099            if (time.elapsed() >= DELAY_100MS) break;
1100            struct pcap_pkthdr *pkt_header;
1101            const u_char *pkt_data;
1102            if (pcap_next_ex(param.fp, &pkt_header, &pkt_data) == 1) {
1103                rdFrame = (__asknowledge *) pkt_data;
1104                if  (rdFrame->ask.TypeMes  ==  from_modem  ||  rdFrame->ask.TypeMes
1105    ==
1106              TYPE_ERROR)  isFind = true;
1107        }    }
1108        result = isFind;
1109        //! \brief check frame on correct address set
1110        if (isFind) {
1111            result = checkErrors(rdFrame);
```

```
1112        if(!result) {ui->textBrowser->
1113  append(STR_MODEM_ERR_NO.arg(from_modem, 0, 16, QLatin1Char('0')));
1114  result = true;}
1115        else result = false;
1116     } else  {ui->textBrowser->append(STR_ERR_PP_1); result = false;}
1117     return result;
1118  }
1119  void test_1::mainTest()
1120  {  delay(DELAY_500MS);
1121     bool isEnd = false;
1122  st = _sync_modems;
1123     while(!isEnd) {
1124        switch(st) {
1125           case _sync_modems: {
1126              if (!sync(TYPE_MODEM_1)) st = _goto_interrupt;
1127              else
1128  st = _send_to_modem_1;
1129  ui->progressBar->setValue(_sync_modems);
1130              break;
1131           }
1132           case _send_to_modem_1: {
1133              if(sendMessage(TYPE_MODEM_1,        TYPE_MODEM_2))        st
1134  =_send_to_modem_2;
1135              else st = _goto_interrupt;
1136  ui->progressBar->setValue(_send_to_modem_1);
1137              break;
1138           }
1139           case _send_to_modem_2: {
1140              if(sendMessage(TYPE_MODEM_2,     TYPE_MODEM_1))     st     =
1141  _goto_end;
1142              else st = _goto_interrupt;
1143  ui->progressBar->setValue(_send_to_modem_2);
1144              break;
1145           }
1146           case  _goto_interrupt: { //! \brief If test interrupted
1147              isEnd = true;
1148              endTest(TEST_INTERRUPT);
1149              break;
1150           }
```

```
1151            case  _goto_end: { //! \brief If test end with errors or not
1152               isEnd = true;
1153               if (isError) endTest(TEST_ERROR);
1154               else        endTest(TEST_COMPLETE);
1155               break;
1156            }
1157            default:
1158    st = _goto_end;
1159               break;
1160    }   }   }
1161    void test_1::on_buttonAbort_clicked()
1162    {  endTest(TEST_INTERRUPT);
1163    }
1164    void test_1::start_session()
1165    {  struct bpf_program filter;                  // Скомпилированное выражение для
1166    фильтра
1167    char filter_app[] = ETH_FILTER_MAC;        // Выражениедляфильтра
1168       char errbuf[PCAP_ERRBUF_SIZE];             // Строкасошибкой
1169       bpf_u_int32 mask;                          // Сетевая маска интерфейса
1170    bpf_u_int32 net;                           // IP адрес нашего интерфейса
1171    //! \brief Create session
1172    param.fp =  pcap_open_live(param.name, 65535, false, 1, errbuf);
1173    //! \brief настройка фильтра для перехвата пакетов
1174    pcap_lookupnet(param.name, &net, &mask, errbuf);
1175       pcap_compile(param.fp, &filter, filter_app, 0, net);
1176       pcap_setfilter(param.fp, &filter);
1177    }
1178    bool test_1::checkErrors(__asknowledge *src)
1179    {bool isWrong = false;
1180     switch (src->ask.TypeMes) {
1181        case TYPE_MODEM_1: {
1182           __mess_type_m *t_1 = (__mess_type_m *) (&src->ask.data);
1183           if   (t_1->Lans   !=   SIZE_ANSV)                              ui->textBrowser-
1184    >append(STR_MODEM_ERR_LEN.arg(src->ask.TypeMes, 0, 16));
1185           uint8_t *data1 = (uint8_t *) &t_1->DataAns;
1186           uint8_t *data2 = (uint8_t *) &mess_2;
1187           for (int i = 0; i < SIZE_ANSV; i++)
1188          if (*data1++ != *data2++) {
1189           isWrong |= true;
```

```
1190 ui->textBrowser->append(STR_MODEM_ERR_DATA.arg(src->ask.TypeMes,
1191 0, 16));
1192                 break;
1193             }
1194             data1 = (uint8_t *) &t_1->DataAns;
1195 if (isWrong){QString buf=getFormFrame(data1,SIZE_ANSV);ui->txtBrower-
1196 >append(buf);}
1197             break;
1198         }
1199         case TYPE_MODEM_2: {
1200             __mess_type_m *t_2 = (__mess_type_m *) (&src->ask.data);
1201             if (t_2->Lans != SIZE_ANSV) ui->textBrowser-
1202 >append(STR_MODEM_ERR_LEN.arg(src->ask.TypeMes, 0, 16));
1203             uint8_t *data1 = (uint8_t *) &t_2->DataAns;
1204             uint8_t *data2 = (uint8_t *) &mess_1;
1205             for (int i = 0; i < SIZE_ANSV; i++)
1206         if (*data1++ != *data2++) {
1207             isWrong |= true;
1208 ui->textBrowser->append(STR_MODEM_ERR_DATA.arg(src->ask.TypeMes,
1209 0, 16));
1210                 break;
1211             }
1212             data1 = (uint8_t *) &t_2->DataAns;
1213 if (isWrong) {QString buf=getFormFrame(data1,SIZE_ANSV);ui->txtBwser-
1214 >append(buf);}
1215             isWrong = false;
1216             break;
1217         }
1218         case TYPE_ERROR: {
1219             __ask_type_4 *t_4 = (__ask_type_4 *) (src->ask.data);
1220             if (t_4->ER_CRCH)      ui->textBrowser->append(STR_ERR_PP_4);
1221             if (t_4->ER_KUP)       ui->textBrowser->append(STR_ERR_PP_5);
1222             if (t_4->ER_CTRLB)     ui->textBrowser->append(STR_ERR_PP_6);
1223             if (t_4->ER_KUI_1)     ui->textBrowser->append(STR_ERR_PP_7);
1224             if (t_4->ER_KUI_2)     ui->textBrowser->append(STR_ERR_PP_8);
1225             if (t_4->ER_L)         ui->textBrowser->append(STR_ERR_PP_9);
1226             if (t_4->ER_CRCI)      ui->textBrowser->append(STR_ERR_PP_10);
1227             if (t_4->ER_ADDR)      ui->textBrowser->append(STR_ERR_PP_11);
1228             if (t_4->ER_RES)       ui->textBrowser->append(STR_ERR_PP_12);
```

```
1229            if      (t_4->ER_TYPEMES)                       ui->textBrowser-
1230    >append(STR_ERR_PP_13);
1231            if      (t_4->ER_OVER_M1)                       ui->textBrowser-
1232    >append(STR_ERR_PP_14);
1233            if      (t_4->ER_OVER_M2)                       ui->textBrowser-
1234    >append(STR_ERR_PP_15);
1235            if (t_4->ER_LUPR)       ui->textBrowser->append(STR_ERR_PP_16);
1236            isWrong = true;
1237            break;
1238          }
1239          default:
1240            isWrong = true;
1241            break;
1242      }
1243      return isWrong;
1244    }
1245    bool test_1::sync(uint8_t numModem)
1246    {    bool isSync = false;
1247      __asknowledge *rdPack;
1248      start_session();
1249    pTim_test->start(DELAY_100MS);
1250    ui->textBrowser->append(STR_SYNC_MOD.arg(numModem));
1251      delay(DELAY_SYNC);
1252      QTime time; time.restart();
1253      while (1) {  if (time.elapsed() >= DELAY_SYNC) break;
1254        struct pcap_pkthdr *pkt_header;
1255        const u_char *pkt_data;
1256        if (pcap_next_ex(param.fp, &pkt_header, &pkt_data) == 1) {
1257          rdPack = (__asknowledge *) pkt_data;
1258          if  (rdPack->ask.TypeMes  ==  TYPE_MODEM_1)    {isSync  =  true;
1259    break;}
1260      }   }   pTim_test->stop();
1261      if                (isSync)              {                 ui->textBrowser-
1262    >append(STR_SYNC_MOD_YES.arg(numModem)); }
1263      else                                                {       ui->textBrowser-
1264    >append(STR_SYNC_MOD_NO.arg(numModem));  }
1265      return isSync;
1266    }
1267    QString test_1::getFormFrame(uint8_t *src, uint32_t cnt)
```

```
{  uint16_t shift = 0x0000; uint32_t i = 0;
   QString result = "";
   result.append(QString(codec->toUnicode("
_Принятоесообщение:_____"")));
   while(i < cnt) {
      if (i % 16 == 0) {result.append(QString(codec->toUnicode("\n      %0
")).arg(shift, 4, 16, QLatin1Char('0')).toUpper()); shift += 0x10;}
      if (i % 16 == 8)  result.append(QString(codec->toUnicode("  ")));
      result.append(QString(codec->toUnicode("%0    ")).arg(*src++,   2,   16,
QLatin1Char('0')).toUpper());
      i++;  }
   result.append("\n   _____\n");
   return result;
}
void test_1::sendForSync()
{  sizeForSync = genFrame(TYPE_MODEM_1);
   pcap_sendpacket(param.fp,              // Adapter
bodyFrame[0],          // buffer with the packet
sizeForSync + 14);       // size
pTim_test->setInterval(DELAY_100MS);
}
```

ДОДАТОК В

Мультимедійна презентація

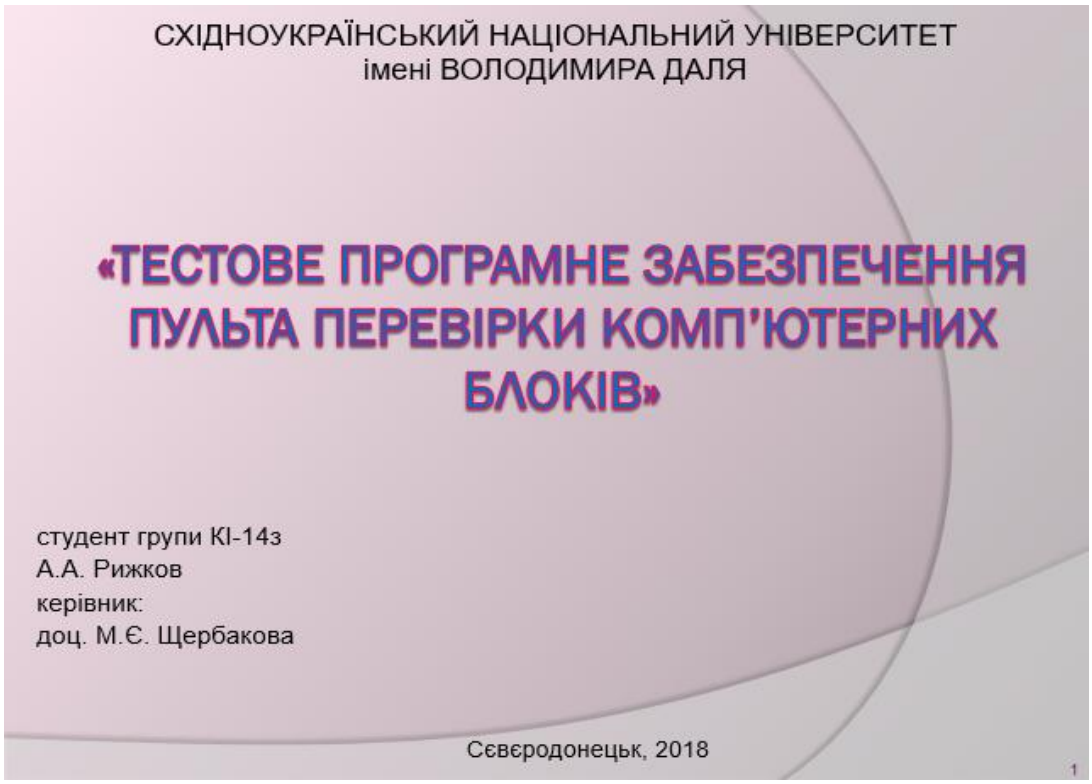

Рисунок В.1 - Титульний аркуш мультимедійної презентації

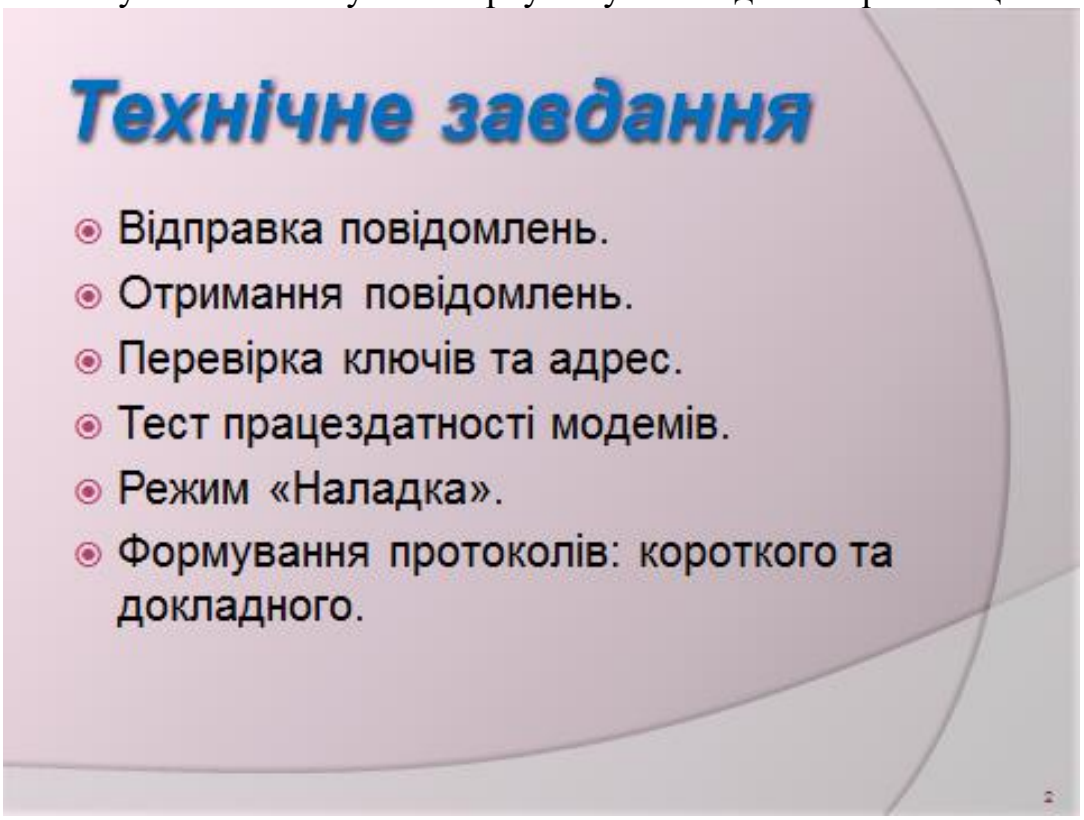

Рисунок В.2 - Слайд мультимедійної презентації №2

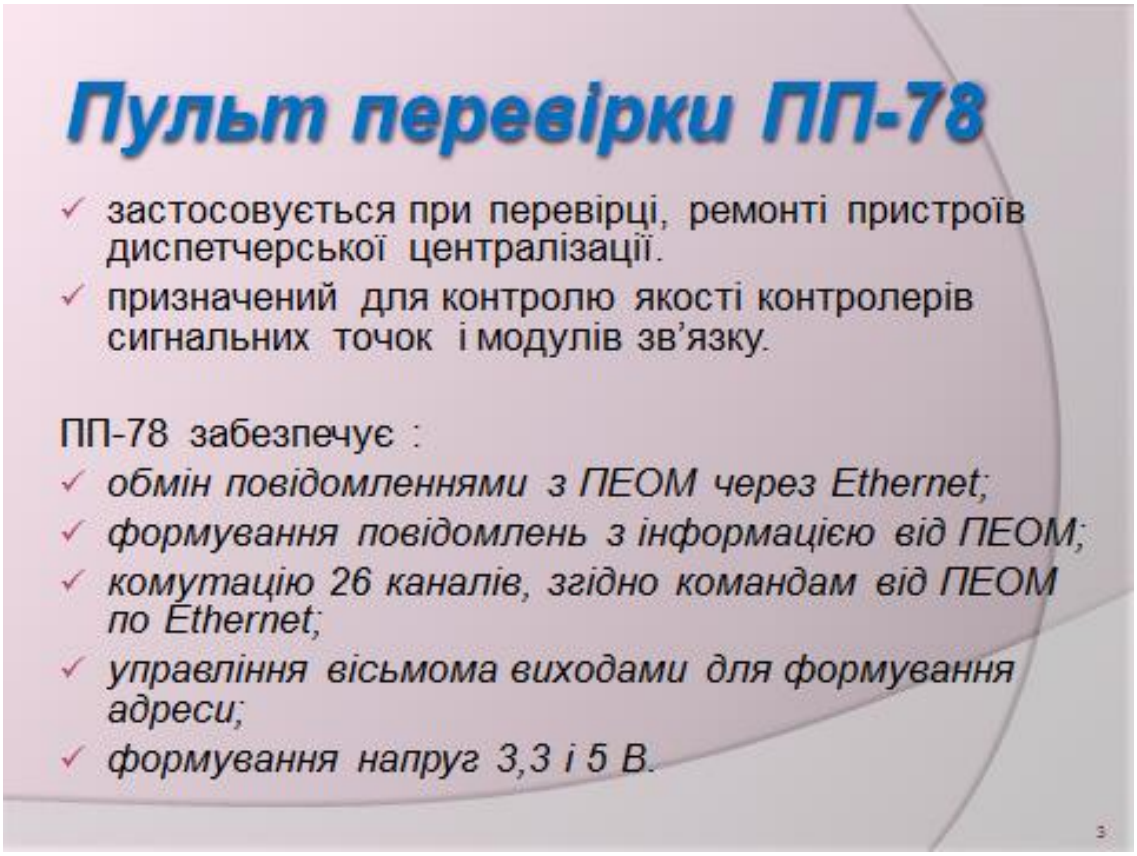

Рисунок В.3 - Слайд мультимедійної презентації №3

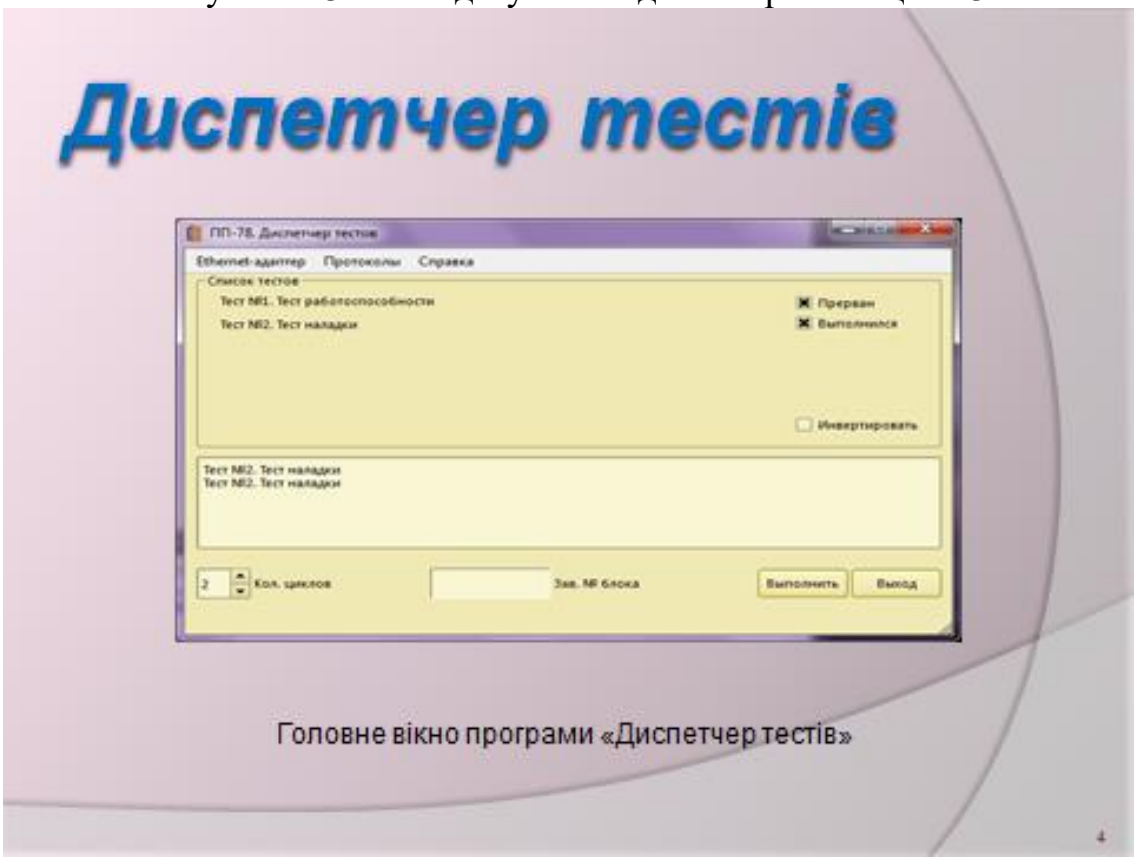

Рисунок В.4 - Слайд мультимедійної презентації №4

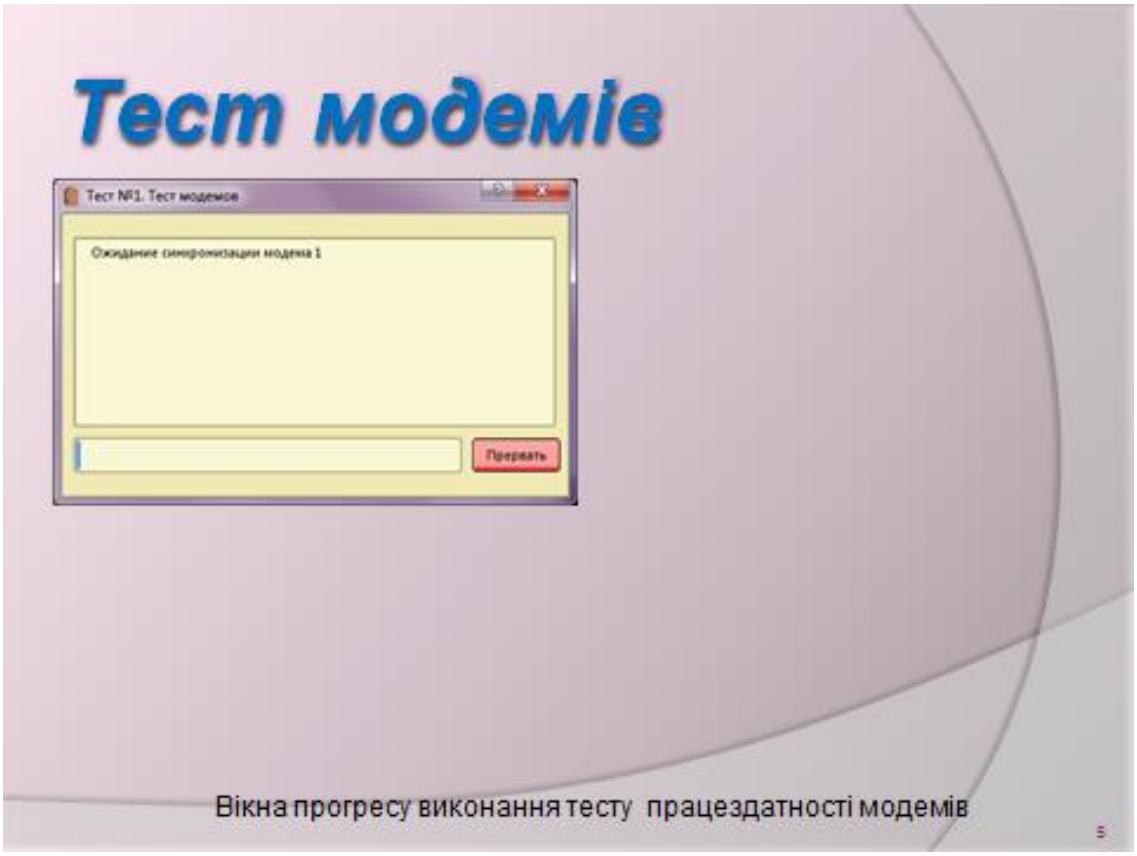

Рисунок В.5 - Слайд мультимедійної презентації №5

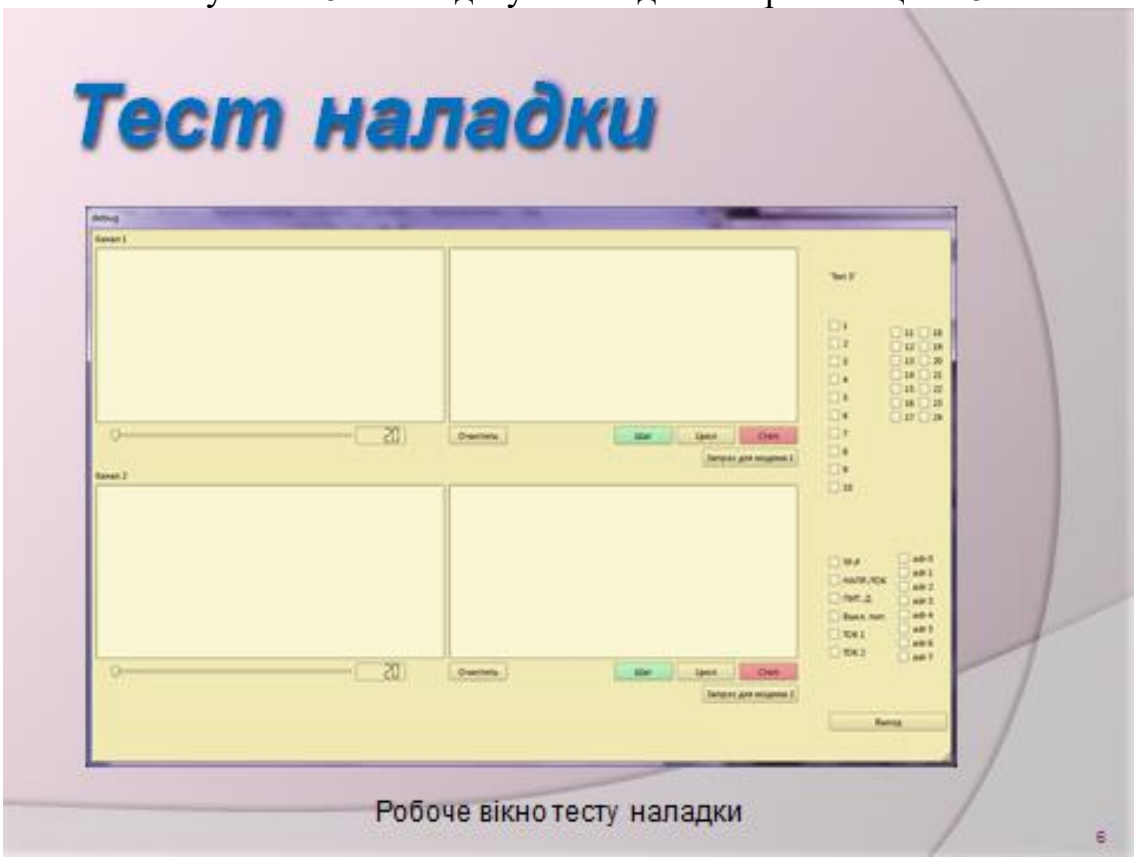

Рисунок В.6 - Слайд мультимедійної презентації №6

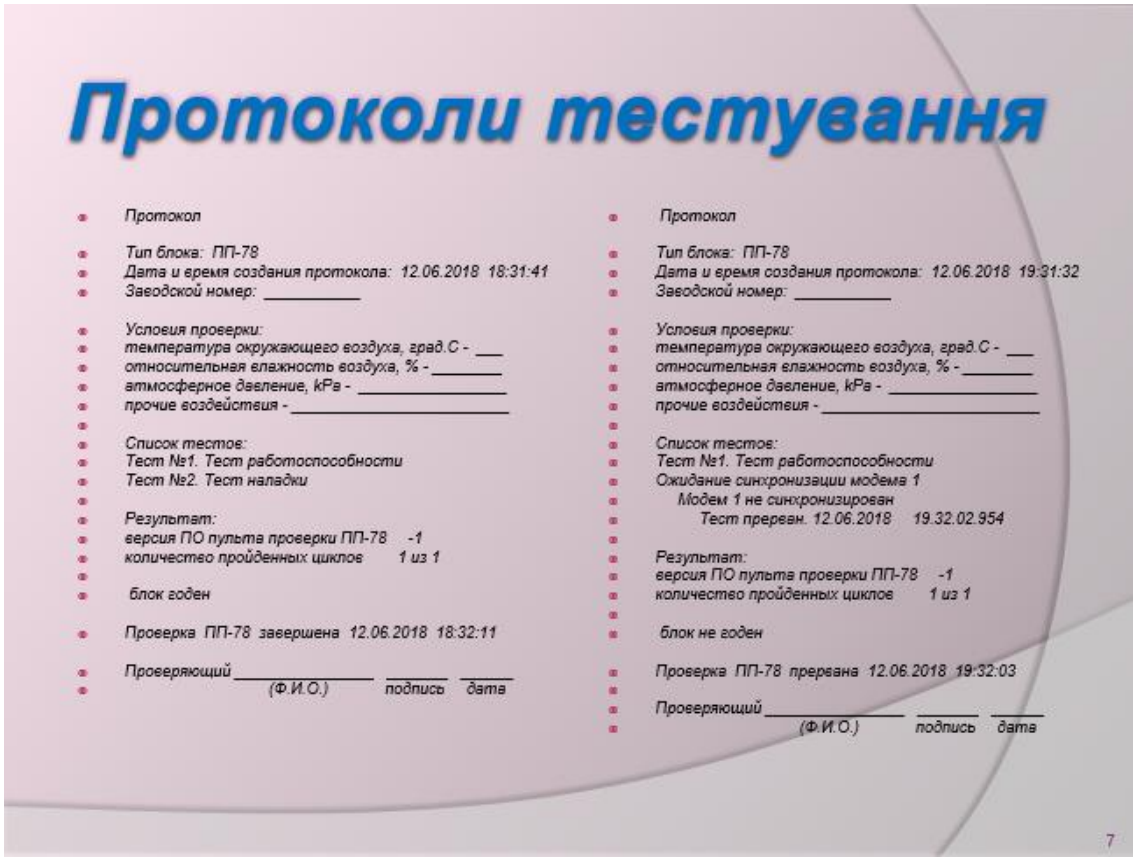

Рисунок В.7 - Слайд мультимедійної презентації №7

Рисунок В.8 - Слайд мультимедійної презентації №8

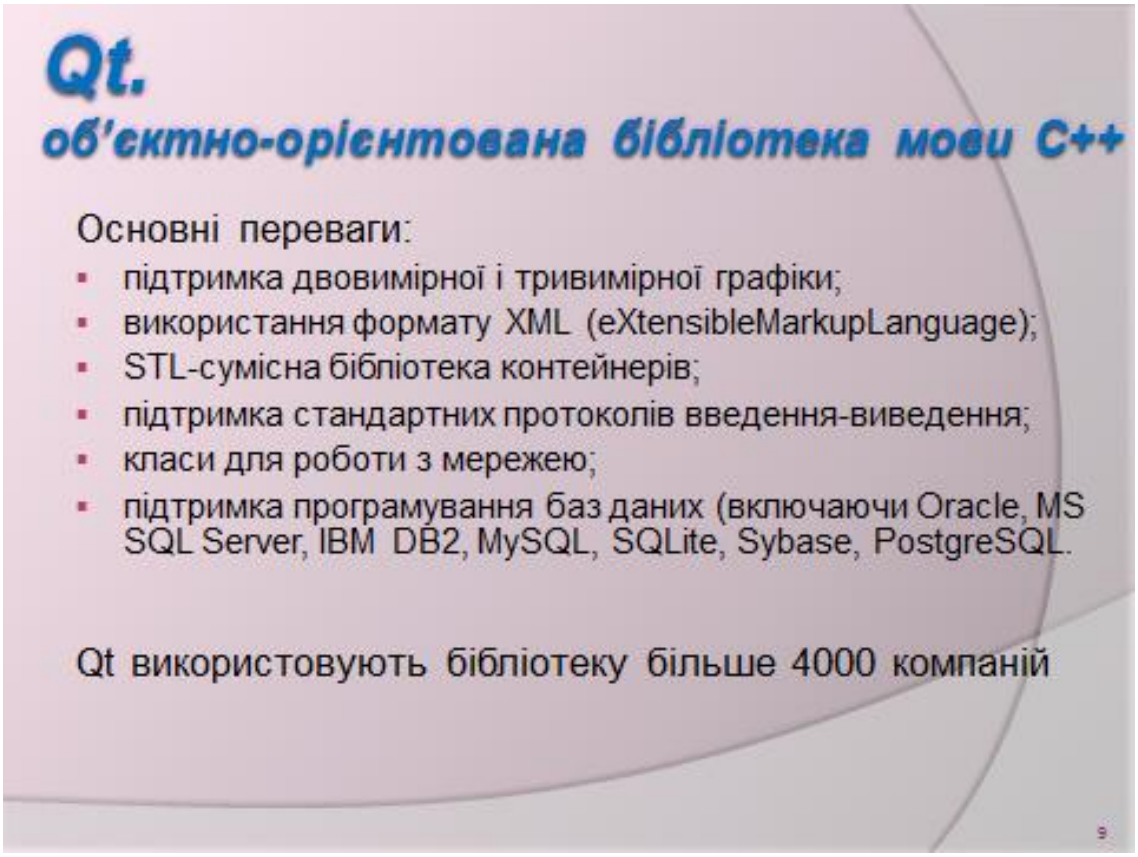

Рисунок В.9 - Слайд мультимедійної презентації №9

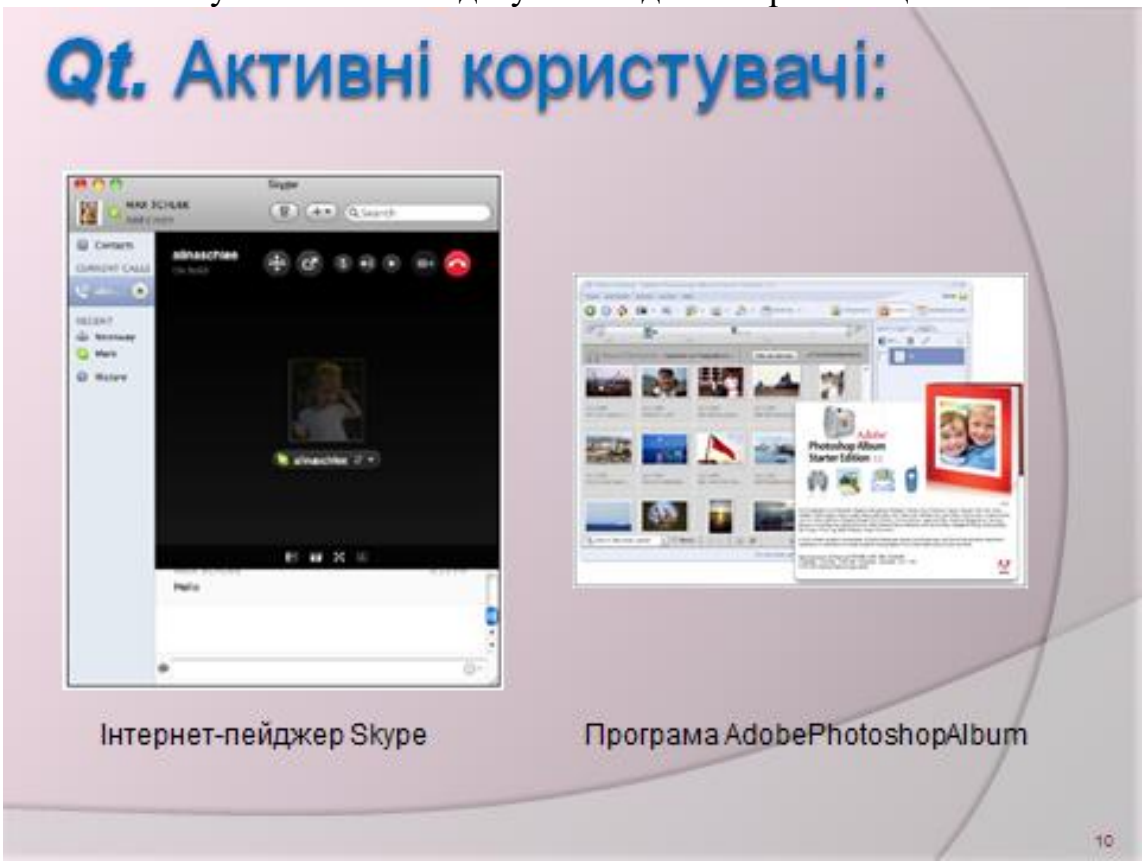

Рисунок В.10 - Слайд мультимедійної презентації №10

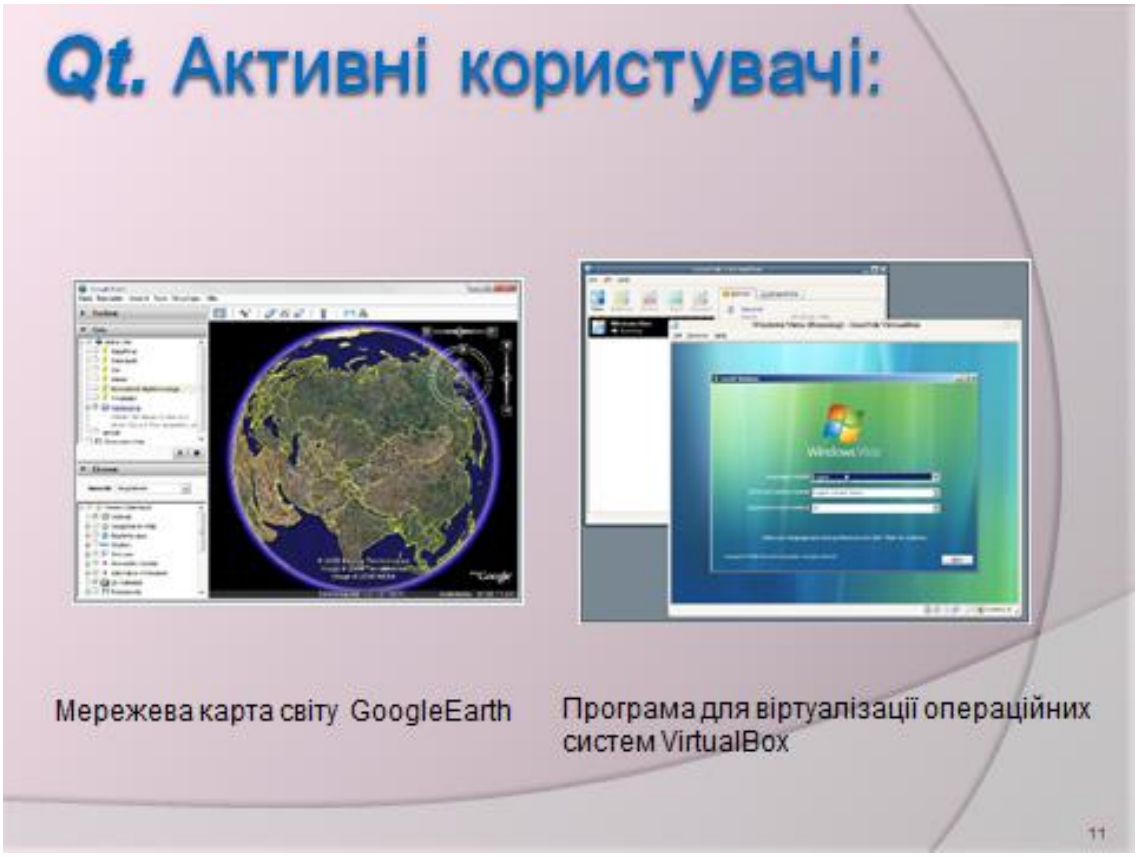

Рисунок В.11 - Слайд мультимедійної презентації №11

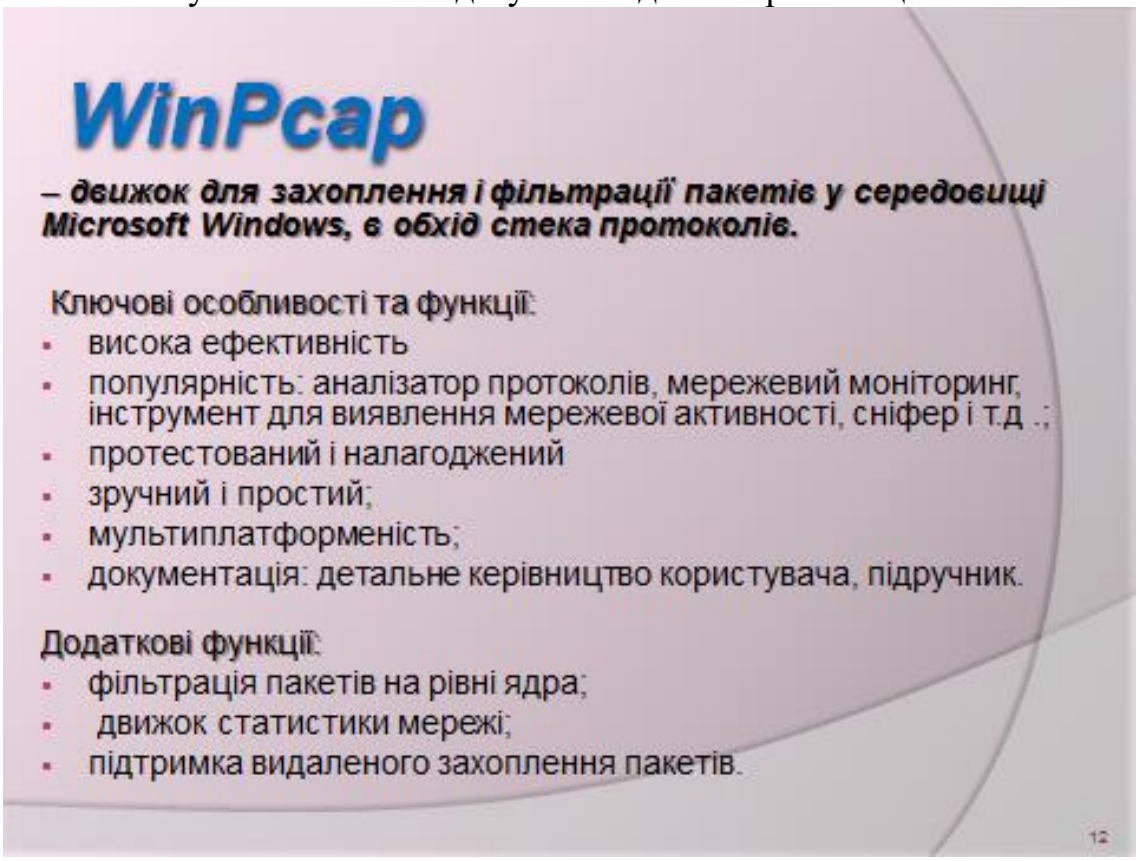

Рисунок В.12 - Слайд мультимедійної презентації №12